Simplifications and enhancements for P802.1ASdm D1.1 around ptpInstanceState

Author: Johannes Specht
Affiliation: Self
Date: 31.07.2023

Introduction

This document illustrates the simplifications proposed on July 24, 2023 during the discussions of comment #75 (https://www.ieee802.org/1/files/private/asdm-drafts/d1/802-1ASdm-d1-1-pdis-v03.pdf) against D1.1 of P802.1ASdm (https://www.ieee802.org/1/files/private/asdm-drafts/d1/802-1ASdm-d1-1.pdf).

The author's original idea documented in the comment was to eliminate two literals of variable ptpInstanceState, an enumeration with 4 literals, because two literals are effectively not used in D1.1 of P802.1ASdm. This makes the variable binary, and it can become a Boolean variable named isSynced. To the author's belief, this name may more accurately reflect its semantic and therefore being a benefit to the reader.

This work started with documenting the changes needed for the aforesaid. During the course, several other locations in D1.1 of P802.1ASdm were discovered with the potential for simplification and clarification to the benefit of the readers. The associated simplifications and clarifications are likewise found in this document.

The simplifications and clarifications found in this document can be summarized as follows:

1. ptpInstanceState is replaced by a Boolean called isSynced.
2. The similar named function isSynced() is entirely removed, because it is only called once, from Figure 17-2, in D1.1 of P802.1ASdm.
3. Instead, Figure 17-2 is changed to include the contents of the removed isSynced() function, and the overall code structure in that figure is reorganized to enhance readability, while maintaining the intended operation.
4. Integer function slavePort() is replaced by the new boolean function isGm(), because slavePort() was only used to determine if the given PTP Instance is a Grandmaster or not, so the replacement is functional equivalent with enhanced readability and simplicity.
5. Various smaller changes to align with the items above.
6. Removal of redundant text.

The aforesaid is details in form of editorial instructions. The editorial instructions are followed by an alternative version for Figure 17-2 than the one presented throughout the editorial instructions.
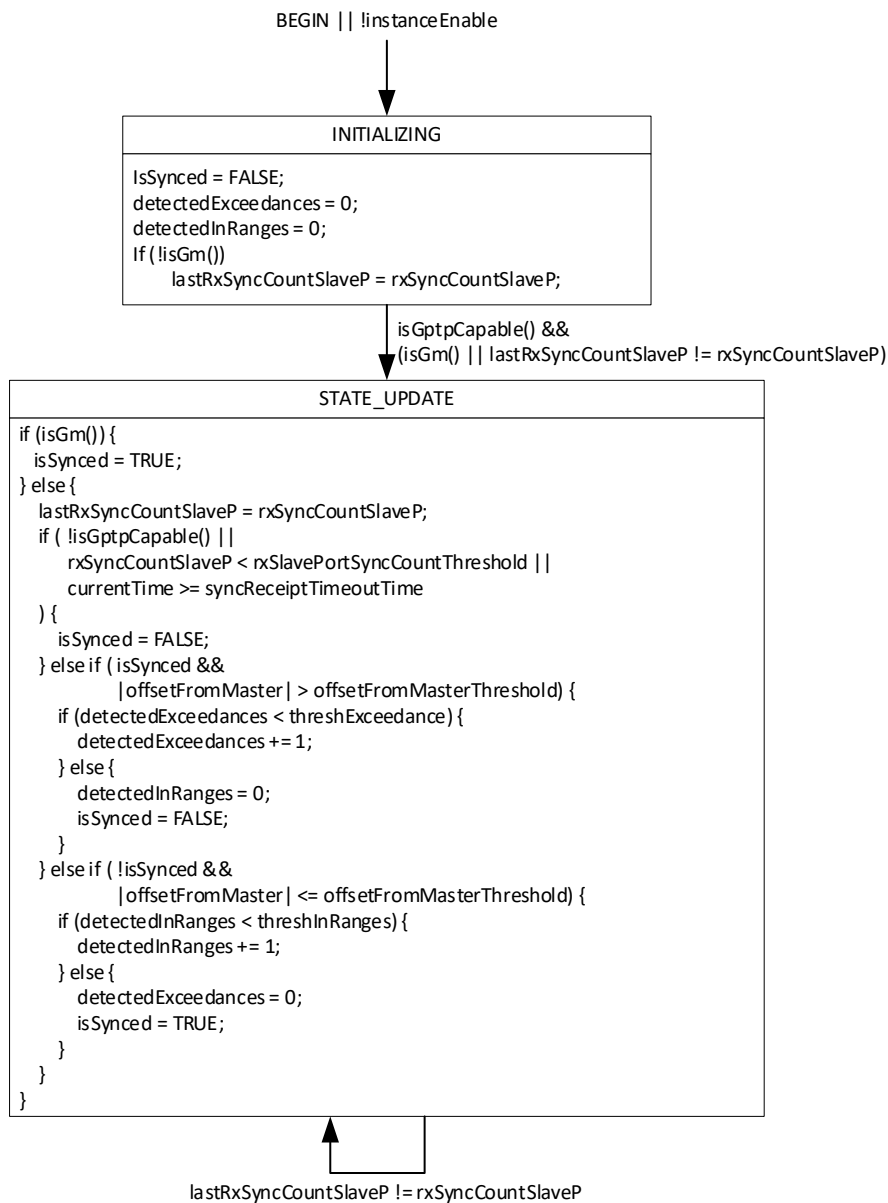
Throughout the editorial instructions, **P**<*page number*>**L**<*line number*> is used as shortcut notation for indicating positions in P802.1ASdm D1.1.

Editorial Instructions

1. <u>Change</u> on P31L8 „ptpInstanceState changes to NOT_SYNCED" to „isSynced changes to FALSE".
2. <u>Change</u> 9.3.3.4 ptpInstanceState to read „9.3.3.4 isSynced(Boolean): The value of isSynced variable (see 17.5.1.1). This parameter shall be present if the optional hot standby feature is implemented (see Clause 17).
   NOTE: „may be present otherwise" is to be omitted, because in absence of the underlying state machine from hot standby, the variable has no specified semantics.
3. <u>Change</u> 9.5.3.3 to read „9.5.3.3 isSynced As specified in 9.4.3.4.". Do similar for 9.6.2.6. NOTE: Before, there was just redundant text from 9.4.3.4. This is a problem in the long run. Other variables may be affected as well, although I do no list these here.
4. <u>Change</u> „ptpInstanceState" to „isSynced" on P38L33, P37L47, P36L43 and P35L46.
5. <u>Change</u> 14.8.2 to read „14.8.2 isSynced The value of the global variable isSynced (17.5.1.1)."
6. <u>Add</u> the following sentence to the end of 14.8.3: "For values smaller than or equal to zero, equation offsetFromMaster = 0 needs to be satisfied for considering the PTP instance as synchronized. For values greater than zero, equation
   -offsetFromMasterThreshold ≤ offsetFromMaster ≤ offsetFromMasterThreshold needs to be satisfied considering the PTP instance as synchronized."
7. In Table 14-6, <u>change</u> the „Name" and „Data type" text in the first row to „isSynced" and „Boolean", respectively.
8. <u>Change</u> 17.5.1.1 to read „17.5.1.1 isSynced The synchronization status of the PTP Instance. The variable is TRUE if the PTP instance is synchronized and FALSE otherwise."
   REMARK: I omitted the entire profile standard description. To my understanding, the profile standards can define the threshold parameters in according to the needs of the respective profile standards.
9. <u>Delete</u> 17.5.2.2.
10. <u>Replace</u> 17.5.3.2 slavePort() by function „isGm(): This function determines if the PTP instance is a grandmaster (TRUE) or not (FALSE) by searching in the selectedState array (see 10.2.4.20) for absence or presence of at least one value equal to SlavePort (see Table 10-2).

    ```
    isGm() {
      for (int i=1; i <= numberOfPorts; i++) // see 8.6.2.8 for numberPorts
        if (selectedState[i] == SlavePort) return FALSE;
      return TRUE;
    }"
    ```
11. <u>Delete</u> 17.5.3.3.
12. <u>Replace</u> Figure 17-2 as shown.

BEGIN || !instanceEnable

↓

```
INITIALIZING
─────────────────────────────────────
IsSynced = FALSE;
detectedExceedances = 0;
detectedInRanges = 0;
If (!isGm())
      lastRxSyncCountSlaveP = rxSyncCountSlaveP;
```

isGptpCapable() &&
(isGm() || lastRxSyncCountSlaveP != rxSyncCountSlaveP)

↓

```
STATE_UPDATE
─────────────────────────────────────
if (isGm()) {
  isSynced = TRUE;
} else {
  lastRxSyncCountSlaveP = rxSyncCountSlaveP;
  if ( !isGptpCapable() ||
      rxSyncCountSlaveP < rxSlavePortSyncCountThreshold ||
      currentTime >= syncReceiptTimeoutTime
  ) {
    isSynced = FALSE;
  } else if ( isSynced &&
         |offsetFromMaster| > offsetFromMasterThreshold) {
    if (detectedExceedances < threshExceedance) {
      detectedExceedances += 1;
    } else {
      detectedInRanges = 0;
      isSynced = FALSE;
    }
  } else if ( !isSynced &&
         |offsetFromMaster| <= offsetFromMasterThreshold) {
    if (detectedInRanges < threshInRanges) {
      detectedInRanges += 1;
    } else {
      detectedExceedances = 0;
      isSynced = TRUE;
    }
  }
}
```

lastRxSyncCountSlaveP != rxSyncCountSlaveP

NOTES:

- offsetFromMaster is a signed integer. Therefore, |offsetFromMaster| is used instead of offsetFromMaster.
- The code is equivalent with that from D1.1 of P802.1ASdm. However, the intended function may be different than both, that in D1.1 of P802.1ASdm and likewise that in the diagram, because the in-range/out-range counting logic may be intended to scan for consecutive syncs in sequence, rather than over the entire phase while the PTP instance is considered to be "sufficiently" or "insufficiently" synced (e.g., right now, 5 "bad" syncs with threshExceedance=5 would lead to a transition from "sufficiently" to "insufficiently" synched, irrespectively if the 5 "bad" syncs appear in a sequence of 10 or 10000000 syncs).
  An alternative for discussion could be a window with the most recent M sync events for which either "good" (|offsetFromMaster| <= offsetFromMasterThreshold) or "bad"

Johannes Specht

(|offsetFromMaster| > offsetFromMasterThreshold) is stored, and of which at least N entries must be "good" with N ≤ M and with configurable N and M.

In any case, the author's hope is that a compact form eases reviewing and commenting.

13. <u>Change</u> „ptpInstanceState != SYNCED" to „isSynced != FALSE" on P122L46. Do similar on P123L23.
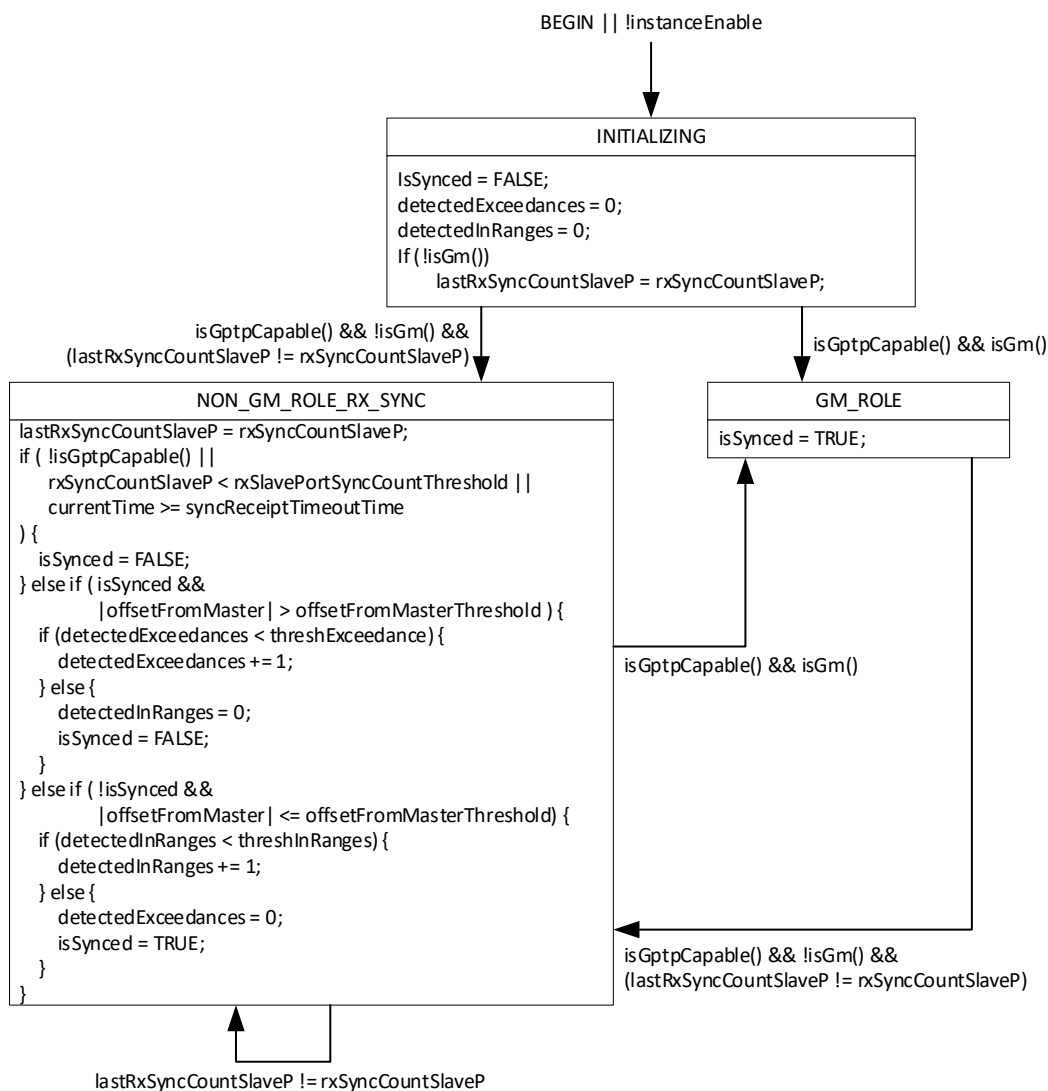
14. <u>Replace</u> numerous ptpInstanceState comparisons in Figure 17-3 by simplified versions.

15. <u>DISCUSS:</u> Rename variables for consistency (including associated modifications in previous editorial instructions):

    a.   offsetFromMaster   → offsetFromMaster [TimeInterval]

    b.   offsetFromMasterThreshold   → offsetFromMasterMax [TimeInterval]

    c.   detectedExceedances   → offsetMaxExceededCount [UInteger32]

    d.   threshExceedance   → offsetMaxExceededCountThresh [UInteger32]

    e.   detectedInRanges   → offsetMaxMetCount [UInteger32]

    f.   threshInRanges   → offsetMaxMetCountThresh [UInteger32]

    g.   rxSlavePortSyncCountThreshold → rxSyncCountSlavePThresh [UInteger32]

    h.   isSynced   → syncQualitySufficient [Boolean]

NOTE: Harmonized names for thresholds and counters, name syncQualitySufficient derived from PAR scope.

Alternative version of Figure 17-2

BEGIN || !instanceEnable

INITIALIZING

```
IsSynced = FALSE;
detectedExceedances = 0;
detectedInRanges = 0;
If (!isGm())
        lastRxSyncCountSlaveP = rxSyncCountSlaveP;
```

isGptpCapable() && !isGm() &&
(lastRxSyncCountSlaveP != rxSyncCountSlaveP)

isGptpCapable() && isGm()

NON_GM_ROLE_RX_SYNC

```
lastRxSyncCountSlaveP = rxSyncCountSlaveP;
if ( !isGptpCapable() ||
     rxSyncCountSlaveP < rxSlavePortSyncCountThreshold ||
     currentTime >= syncReceiptTimeoutTime
) {
   isSynced = FALSE;
} else if ( isSynced &&
         |offsetFromMaster| > offsetFromMasterThreshold ) {
   if (detectedExceedances < threshExceedance) {
      detectedExceedances += 1;
   } else {
      detectedInRanges = 0;
      isSynced = FALSE;
   }
} else if ( !isSynced &&
         |offsetFromMaster| <= offsetFromMasterThreshold) {
   if (detectedInRanges < threshInRanges) {
      detectedInRanges += 1;
   } else {
      detectedExceedances = 0;
      isSynced = TRUE;
   }
}
```

GM_ROLE

```
isSynced = TRUE;
```

isGptpCapable() && isGm()

isGptpCapable() && !isGm() &&
(lastRxSyncCountSlaveP != rxSyncCountSlaveP)

lastRxSyncCountSlaveP != rxSyncCountSlaveP

Remarks:

- The idea behind this version is to have a stronger separation between non-GM and GM role. In the GM role, it is always assumed that the GM is perfectly synchronized to itself, which is obvious. But this becomes dominantly visible in this version, compared to the previous figure, and even stronger compared to D1.1 of P802.1ASdm. A benefit of this version is the reduced branch depth in NON_GM_ROLE_RX_SYNC.
- Other state arrangements (e.g., a vertically aligned in the order (top to bottom) INITIALIZING, GM_ROLE and NON_GM_ROLE_RX_SYNC) may provide even more visual enhancement, but the author did not evaluate what is the best arrangement.
- The counters used and increased in non-GM role are not reset prior to transition from a GM role back to a non-GM role. This is functional equivalent to D1.1 of P802.1ASdm, however, it may be reasonable to do so by resetting these in GM_ROLE (similar to the resets in INITIALIZING).

Johannes Specht