# IEEE 802.1Q$^{TM}$-2018 Fault Reporting Issue

Subtitle

Marc Holness

February 2023

# Fault Notification Specification Issues/Deficiencies

1. Lower priority fault notifications are not suppressed when a higher priority fault notification is generated

For example,

- At t0, priority 3 fault becomes active on the system
  - The notification for priority 3 fault gets generated
  - Fault-Present variable is set to TRUE
- At t1, a priority 5 fault arrives on the service
  - The notification for priority 5 fault gets generated
  - Fault-Present variable continues to be TRUE
- No CLEAR notification is seen for the lower priority fault (priority 3), as the system is tracking all the faults through one variable – Fault-Present

# Fault Notification Specification Issues/Deficiencies

2. Fault clear notification is not generated when multiple faults are present on a service
3. Fault notification is not generated for the new active fault, when multiple faults were present on a service and the highest priority fault present on that service gets cleared

For example,

- At t0, priority 4 fault becomes active on the system
  - The fault notification for priority 4 fault gets notified
  - Fault-Present variable is set to TRUE
- At t1, a priority 2 fault arrives on the service
  - Since, it has lower priority than the current active fault, no notification is generated
- At t2, fault of priority 4 clears from the service
  - This leaves priority 2 fault as the active fault on the service
- However, since there is no mechanism to track individual faults, the priority 4 fault's notification will not be cleared
- Additionally, no notification is generated for the current active fault (which is the priority 2 fault)

# IEEE 802.1Q$^{TM}$-2018 References

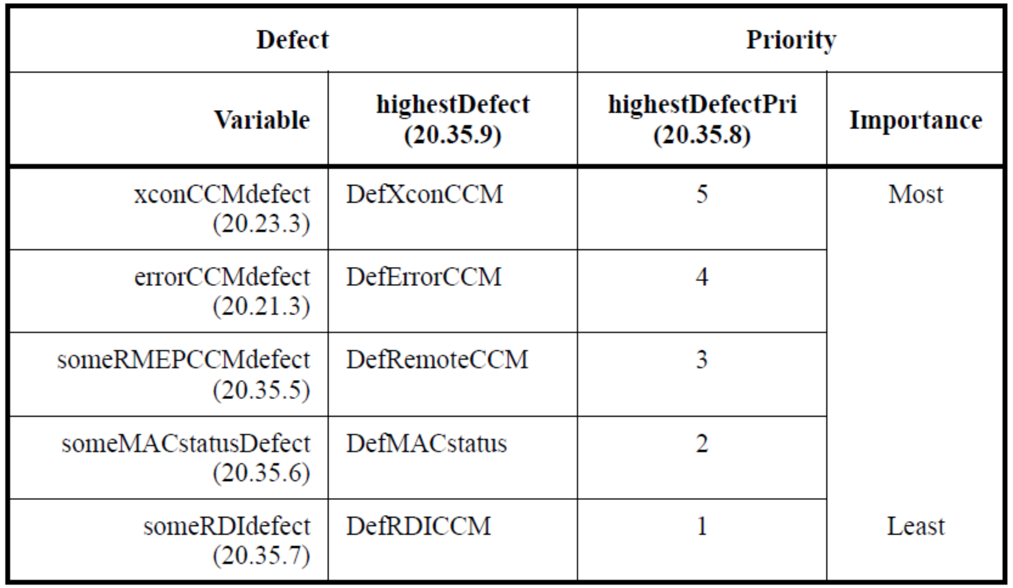
- Subclause 20.1.2 **Defects and Fault Alarms**

  "… Defects are separated from Fault Alarms (19.2.16), as is standard practice for service providers …

  … A number of separate defects are maintained by a MEP, as shown in Table 20-1. The defects are ranked by priority. If a higher priority defect occurs after a lower priority defect has triggered a Fault Alarm, but before the Fault Alarm has reset, then the MEP will immediately issue another Fault Alarm. This enables the operator to reliably prioritize Fault Alarms. For example, cross connect errors are typically of greater concern in a service provider environment than loss of connectivity errors …

  … Only the highest priority defect is reported in the Fault Alarm. Table 20-1 shows the relationships between the variables that indicate defects, the priorities of these defects, and the enumerated value reported in the Fault Alarm for each defect …"

**Table 20-1—Fault Alarm defects and priorities**



| Defect | | Priority | |
|---|---|---|---|
| **Variable** | **highestDefect (20.35.9)** | **highestDefectPri (20.35.8)** | **Importance** |
| xconCCMdefect (20.23.3) | DefXconCCM | 5 | Most |
| errorCCMdefect (20.21.3) | DefErrorCCM | 4 | |
| someRMEPCCMdefect (20.35.5) | DefRemoteCCM | 3 | |
| someMACstatusDefect (20.35.6) | DefMACstatus | 2 | |
| someRDIdefect (20.35.7) | DefRDICCM | 1 | Least |

# IEEE 802.1Q™-2018 References

- Subclause 20.37 **MEP Fault Notification Generator state machine**

  "… A MEP creates a single instance of the MEP Fault Notification Generator state machine. The MEP Fault Notification Generator state machine implements the function specified by the state diagram in Figure 20-13 …"

- **fngPriority** (20.35.1)

  "… An integer specifying the priority of the last defect reported in a Fault Alarm. fngPriority takes the same values as highestDefectPri (20.35.8) …"

- **fngDefect** (20.35.2)

  "… An enumerated value specifying the last defect reported in a Fault Alarm. fngDefect takes the same values as highestDefect (20.35.9) …"

- **fngAlarmTime** (20.35.3)

  "… The time that one or more defects must be present before a Fault Alarm is issued. Default value 2.5 s …"

- **fngResetTime** (20.35.4)

  "… The time, after a Fault Alarm, that no defects must be present before another Fault Alarm is enabled. Default value 10 s …"

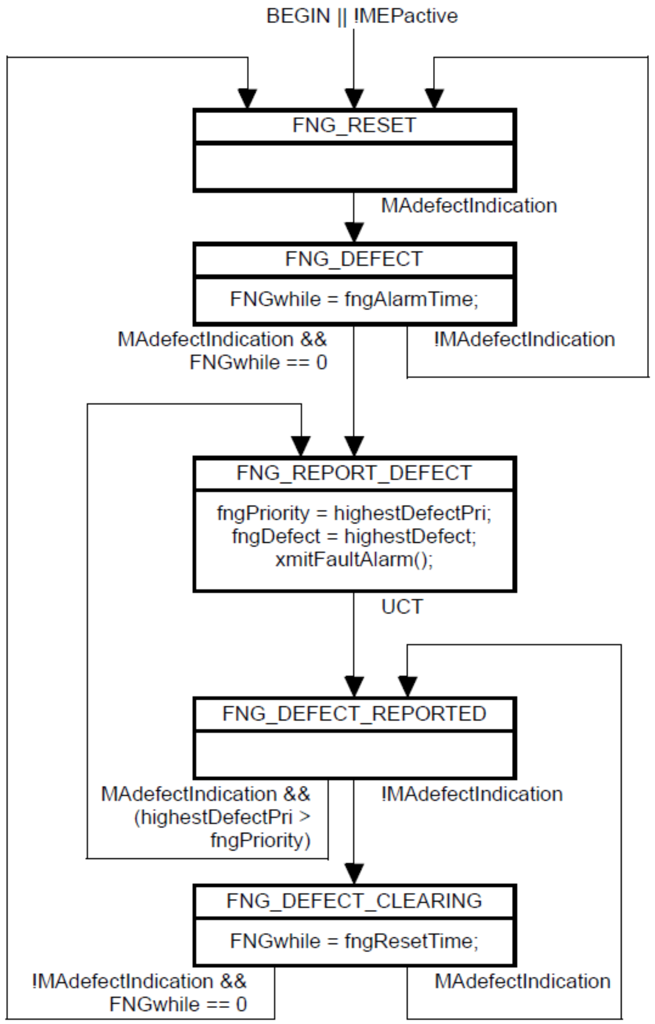


**Figure 20-13—MEP Fault Notification Generator state machine**

# IEEE 802.1Q$^{TM}$-2018 References

- **MAdefectIndication** (20.9.3)

  "… A Boolean indicating the operational state of the MEP's MA. True indicates that at least one of the remote MEPs configured on this MEP's MA has failed, and false indicates either that all are functioning, or that the MEP has been active for less than the time-out period. MAdefectIndication is true whenever an enabled defect is indicated. That is, MAdefectIndication is true if and only if, for one or more of the variables someRDIdefect, someRMEPCCMdefect, someMACstatusDefect, errorCCMdefect, or xconCCMdefect, that variable is true and the corresponding priority of that variable from Table 20-1 is greater than or equal to the value of the variable lowestAlarmPri …"

- **lowestAlarmPri** (20.9.5)

  "… An integer value indicating the lowest defect priority (see Table 20-1) that can trigger the generation of a Fault Alarm …"

- **highestDefectPri** (20.35.8)

  "… An integer value indicating the priority of the defect named in the variable highestDefect …"

- **xmitFaultAlarm()** (20.36.1)

  "… Transmits a Fault Alarm (12.14.7.7). The identity of the MEP and the variable fngDefect (20.35.2), specifying the cause of the Fault Alarm, are transmitted in the Fault Alarm PDU …"

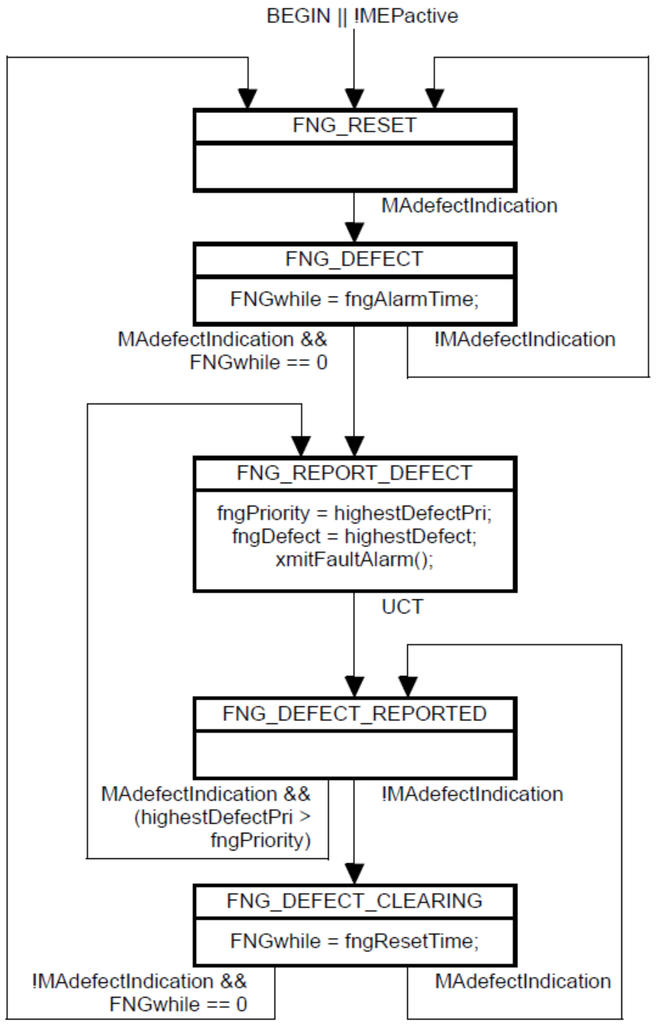


**Figure 20-13—MEP Fault Notification Generator state machine**

# Suggested IEEE 802.1Q Specification Modification(s)

1. Extend the definition of the **MAdefectIndication** variable (20.9.3)
    - Change the variable type from a Boolean to a bit vector

| | | | DefXconCCM | DefErrorCCM | DefRemoteCCM | DefMACstatus | DefRDICCM |
|---|---|---|---|---|---|---|---|
| x | x | x | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

    - The individual bits of the variable **MAdefectIndication** will correspond to the different defects that are currently supported by CFM (currently 5 defects are supported)
    - The variable can be expanded further in future if CFM supports more defects

2. Introduce a new variable called **fngMAdefectIndication**
    - This variable will have the same type and size as the new **MAdefectIndication** variable

| | | | DefXconCCM | DefErrorCCM | DefRemoteCCM | DefMACstatus | DefRDICCM |
|---|---|---|---|---|---|---|---|
| x | x | x | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

    - This variable is used to track the defects that are present before there is a change in the new **MAdefectIndication** variable

# Suggested IEEE 802.1Q Specification Modification(s)

3. Introduce a new variable called **MAdefectIndicationBitChange**

   "… A Boolean indicating that the valid (significant) bits, with a bit position greater than the variable lowestAlarmPri of the MAdefectIndication variable have been updated …"

4. Modify the definition of the variable **highestDefectPri** (20.35.8)

   "… An integer value indicating the priority of the highest priority defect found in the MAdefectIndication variable, as limited by lowestAlarmPri (20.9.5) …"

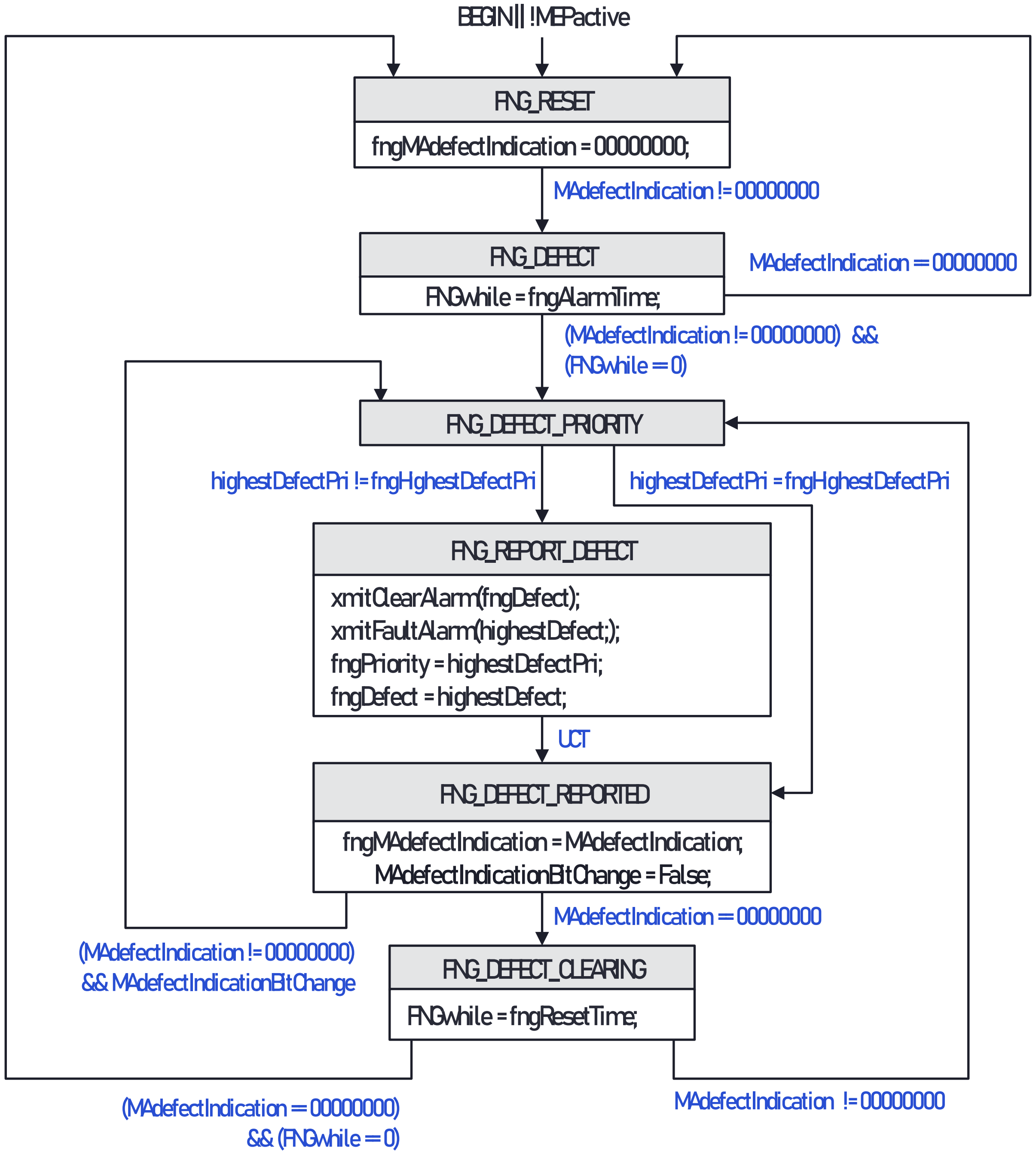
5. Introduce a new variable called **fngHighestDefectPri**

   "… An integer value indicating the priority of the highest priority defect found in the fngMAdefectIndication variable, as limited by lowestAlarmPri (20.9.5) …"

6. Introduce a new function called **xmitClearAlarm()**

   "… This function transmits a Clear Alarm Notification. It takes fngDefect (20.35.2) as an argument. The identity of the MEP and the variable fngDefect (20.35.2), specifying the Fault Alarm that is cleared, and is transmitted in the Clear Alarm PDU …"

7. Update the **MEP Fault Notification Generator state machine** as illustrated on next slide



BEGIN || !MEPactive

FNG_RESET

fngMAdefectIndication = 00000000;

MAdefectIndication != 00000000

FNG_DEFECT

FNGwhile = fngAlarmTime;

MAdefectIndication == 00000000

(MAdefectIndication != 00000000) &&
(FNGwhile == 0)

FNG_DEFECT_PRIORITY

highestDefectPri != fngHighestDefectPri

highestDefectPri = fngHighestDefectPri

FNG_REPORT_DEFECT

xmitClearAlarm(fngDefect);
xmitFaultAlarm(highestDefect;);
fngPriority = highestDefectPri;
fngDefect = highestDefect;

UCT

FNG_DEFECT_REPORTED

fngMAdefectIndication = MAdefectIndication;
MAdefectIndicationBitChange = False;

MAdefectIndication == 00000000

(MAdefectIndication != 00000000)
&& MAdefectIndicationBitChange

FNG_DEFECT_CLEARING

FNGwhile = fngResetTime;

(MAdefectIndication == 00000000)
&& (FNGwhile == 0)

MAdefectIndication != 00000000

ciena®

# Thank You