Comment *(Technical pg: 186 Cl: 144.2.1 line: 27)*
Use of the term timestamp is ambiguous:
187/28 - defined as a variable
187/30 - something other than (LocalTime?) the variable that is being defined
187/33 - the variable that is being defined
187/52 - a non italicized variable
187/53 - "timestamp value" (which apparently is not the same as the variable)
188/2 - a field name "Timestamp field"
191/40 - the value of the variable (or maybe field?) "the Timestamp value pre-compensated"
192/24 - a field value "the Timestamp field value"
I could go on;
there are 29 instances of "Timestamp" most of which (but not all) are in italics (including a lone instance of "Timestamp drift" in DeregistrationTrigger definition),
there are 29 instances of "timestamp" none of which are in italics (including 11 instance of "timestamp value" and 11 instances of "timestamp drift")

We can be nicer to the first time reader.

Editing Instruction: The concepts of localtime, timestamps, and timestamp drift is never introduced in Cl 144.
In 144.1.1.1 at pg 180 line 48 add the following para:
"To enable transmission arbitration the OLT and each ONU includes a counter to track *LocalTime* (see 144.3.1.2). MPCPDUs include a field, the *TimeStamp* field, which is used to convey the time of the MPCPDU transmission. At the receiving station this field is compared to the receiving stations *LocalTime*, this difference is referred to as timestamp drift. If the timestamp drift value is above a set threshold (i.e., what is expected) the station declares a timestamp drift error. Additional details on these concepts are included in the subcauses below."

Editing Instruction: At the locations indicated make the changes shown in the table below.

Pg	***Line***	***From*** ***To***
186	'27-30	The Control Parser also extracts the timestamp value from all MPCPDUs that contain the *Timestamp* field and checks whether the timestamp drift value is within the acceptable range. The Control Parser also extracts the value of the *Timestamp* field from all MPCPDUs that contain this field and checks whether the timestamp drift value is within the acceptable range.
186	'32-35	The Control Multiplexor inserts the timestamp value into all MPCPDUs that carry the *Timestamp* field.

		The Control Multiplexor inserts the *Timestamp* field value into all MPCPDUs that carry this field.
186	'40	This constant holds the maximum amount of drift allowed for a timestamp received at the given device. This constant holds the maximum amount of drift allowed before a timestamp drift error is declared.
187	'2	In the ONU, this variable is updated with the received timestamp value by the Control Parser Process (see 144.2.1.5). In the ONU, this variable is updated with the value of *TimestampRx* by the Control Parser Process (see 144.2.1.5).
187	'30-33	Timestamp Type: 32-bit unsigned integer Description: In the Control Multiplexor state diagram, this variable holds the PLID-specific value of timestamp to be inserted into an outgoing MPCPDU. In the Control Parser state diagram this variable represents the value of the *Timestamp* field of the received MPCPDU. TimestampRx Type: 32-bit unsigned integer Description: In the Control Parser state diagram this variable represents the *Timestamp* field value of the received MPCPDU. TimestampTx Type: 32-bit unsigned integer Description: In the Control Multiplexor state diagram, this variable holds the PLID-specific *Timestamp* field value to be inserted into an outgoing MPCPDU.
187	'40	A list of all MPCPDU opcodes that contain the *Timestamp* field (see Table 31A-1). ***OK as is, no change required.***
187 188	'53 2	ProcessTimestamp (Plid, Timestamp) This function takes the PLID and the timestamp value from a received MPCPDU and checks whether the timestamp drift has exceeded the predefined device-specific threshold *DRIFT_THOLD*. In the ONU, this function sets the *LocalTime* variable to the value of the received Timestamp field (see 144.3.1.1). … ProcessTimestamp (Plid, TimestampRx) This function takes the PLID and *TimestampRx* from a received MPCPDU *Timestamp* field and checks whether the timestamp drift has exceeded the predefined device-specific threshold *DRIFT_THOLD*. In the ONU, this function also sets the *LocalTime* variable to the value of *TimestampRx* (see 144.3.1.1). …
188	'5-17	`ProcessTimestamp( Plid, Timestamp )` `{`

		`` if(FirstTimestamp[Plid]) { // The following line is executed only in the ONU LocalTime = Timestamp; Rtt[Plid] = LocalTime – Timestamp; TimestampDrift[Plid] = false; FirstTimestamp[Plid] = false; } Else TimestampDrift[Plid] = abs(LocalTime – Timestamp) > DRIFT_THOLD } ProcessTimestamp(Plid, TimestampRx) { if(FirstTimestamp[Plid]) { // The following line is executed only in the ONU LocalTime = TimestampRx; Rtt[Plid] = LocalTime – TimestampRx; TimestampDrift[Plid] = false; FirstTimestamp[Plid] = false; } Else TimestampDrift[Plid] = abs(LocalTime – TimestampRx) > DRIFT_THOLD } ```
189	'49-54	The *LocalTime* counters supply the timestamp value for MPCPDUs transmitted by either device. The time reference point for the timestamp value is the transmission time of the Envelope Start Header (ESH) of the envelope that includes the MPCPDU (see 143.3.2). In situations where multiple MPCPDUs are transmitted within a single envelope, all these MPCPDUs shall have the same timestamp value, referencing the transmission time of ESH. The *LocalTime* counters supply the *Timestamp* field value for MPCPDUs transmitted by either device. The time reference point for TimestampTx is the transmission time of the Envelope Start Header (ESH) of the envelope that includes the MPCPDU (see 143.3.2). In situations where multiple MPCPDUs are transmitted within a single envelope, all these MPCPDUs shall have the same *Timestamp* field value, referencing the transmission time of ESH.
190	'11	The OLT Discovery Process transmits DISCOVERY MPCPDU with timestamp value equal to the *LocalTime* counter at the time when the MCRS_CTRL.request primitive is generated by the Envelope Activation Process. The OLT Discovery Process transmits DISCOVERY MPCPDU with Timestamp field value equal to the *LocalTime* counter at the time when the MCRS_CTRL.request primitive is generated by the Envelope Activation Process.

190	'16	When an unregistered ONU receives the DISCOVERY MPCPDUs, it sets its *LocalTime* counter according to the value in the *Timestamp* field in the received MPCPDU. When an unregistered ONU receives the DISCOVERY MPCPDUs, it sets its *LocalTime* counter to the *Timestamp* field value in the received MPCPDU.
190	'20	After the ONU's *LocalTime* counter reached the value of *GrantStartTime* and an additional random delay, the ONU Registration Process transmits the REGISTER_REQ MPCPDU with timestamp value equal to the *LocalTime* counter at the time when the MCRS_CTRL.request primitive is generated by the Envelope Activation Process. After the ONU's *LocalTime* counter reaches the value of *GrantStartTime* and an additional random delay, the ONU Registration Process transmits the REGISTER_REQ MPCPDU with *Timestamp* field value equal to the *LocalTime* counter at the time when the MCRS_CTRL.request primitive is generated by the Envelope Activation Process.
190	'26-30	When the OLT receives MPCPDUs, it uses the received timestamp value to calculate the round trip time between the OLT and the ONU. The Round Trip Time (RTT) is equal to the difference between the OLT's *LocalTime* counter at the moment when the ESH is read from the MCRS *EnvRx* buffer and the *Timestamp* field value in the REGISTER_REQ MPCPDU. When the OLT receives MPCPDUs, it uses the received *Timestamp* field value to calculate the round trip time between the OLT and the ONU. The Round Trip Time (RTT) is equal to the difference between the OLT's *LocalTime* counter at the moment when the ESH is read from the MCRS *EnvRx* buffer and the *Timestamp* field value in the REGISTER_REQ MPCPDU.
191	'28	(the *Timestamp* field value in the REGISTER_REQ MPCPDU) ***OK as is, no change required.***
191	'40-45	All MPCPDUs sent by the OLT on unicast PLIDs have the *Timestamp* value pre-compensated by the RTT associated with this PLID: *Timestamp*[*LLID*] = *LocalTime* + *RTT*[*LLID*] The effect of such timestamp pre-compensation is that the first ESH in any burst from an ONU arrives to the OLT MCRS *EnvRx* buffer approximately 32 EQT before their *GrantStartTime* values and this ESH is read from *EnvRx* buffer into the associated MAC instance at the time when the OLT's *LocalTime* counter value is equal to *GrantStartTime* (see Figure 144–8). All MPCPDUs sent by the OLT on unicast PLIDs have the *Timestamp* field value pre-compensated by the RTT associated with this PLID: *TimestampTx*[*LLID*] = *LocalTime* + *RTT*[*LLID*] The effect of such pre-compensation is that the first ESH in any burst from an ONU arrives to the OLT MCRS *EnvRx* buffer

		approximately 32 EQT before their *GrantStartTime* values and this ESH is read from *EnvRx* buffer into the associated MAC instance at the time when the OLT's *LocalTime* counter value is equal to *GrantStartTime* (see Figure 144–8).
192	'24-39	Another effect of the timestamp pre-compensation is that the *Timestamp* field value in a received MPCPDU is expected to match the *LocalTime* value at the time the MPCPDU is received. (As stated above, the timestamp reference point is the time the ESH is read from the *EnvRx* buffer). A condition of timestamp drift error occurs if the OLT's and ONU's *LocalTime* counters lose their synchronization or mutual alignment. This condition can be independently detected by the OLT or an ONU. This condition is detected when an absolute difference between the Timestamp value received in an MPCPDU and the *LocalTime* counter exceeds the timestamp drift threshold limit *DRIFT_THOLD* (see 144.2.1.4). The timestamp drift error causes an immediate ONU deregistration. After the ONU receives the REGISTER MPCPDU with its assigned PLID and MLID, it stops processing any MPCPDUs received in envelopes with DISC_PLID. At this time, the ONU is ready to accept its first GATE received on the newly-assigned unicast PLID. The timestamp in this GATE MPCPDU is pre-compensated with ONU's RTT, and therefore, the ONU is expected to measure a large difference between the received *Timestamp* value and its *LocalTime* counter. This large difference is detected immediately after the registration is expected and the ONU shall not recognize it as a timestamp drift error. Another effect of the pre-compensation is that the *Timestamp* field value in a received MPCPDU is expected to match the *LocalTime* value at the time the MPCPDU is received. (As stated above, the Timestamp field reference point is the time the ESH is read from the *EnvRx* buffer). A timestamp drift error occurs if the OLT's and ONU's *LocalTime* counters lose their synchronization or mutual alignment. This condition can be independently detected by the OLT or an ONU. This condition is detected when an absolute difference between the *Timestamp* field value received in an MPCPDU and the *LocalTime* counter exceeds the timestamp drift threshold limit *DRIFT_THOLD* (see 144.2.1.4). The timestamp drift error causes an immediate ONU deregistration. After the ONU receives the REGISTER MPCPDU with its assigned PLID and MLID, it stops processing any MPCPDUs received in envelopes with DISC_PLID. At this time, the ONU is ready to accept its first GATE received on the newly-assigned unicast PLID. The Timestamp field in this GATE MPCPDU is pre-compensated with ONU's RTT, and therefore, the ONU is expected to measure a large difference between the received *Timestamp* field value and its *LocalTime* counter. This large difference detected immediately after the registration is expected and the ONU shall not recognize it as a timestamp drift error.
199	'41	Up to seven LLIDs can be reported by a single REPORT MPCPDU. REPORT MPCPDUs also carry the *Timestamp* value that is used by the OLT to check for the timestamp drift condition (see 144.3.1.1). Up to seven LLIDs can be reported by a single REPORT MPCPDU. REPORT MPCPDUs also carry the *Timestamp* field value that is used by the OLT to check for the timestamp drift error (see 144.3.1.1).
201	'6	The REGISTER_REQ MPCPDUs carry the *Time-stamp* value that is used by the OLT to measure the round-trip time of that ONU (see 144.3.1.1).

		The REGISTER_REQ MPCPDUs carry the *Timestamp field* value that is used by the OLT to measure the round-trip time of that ONU (see 144.3.1.1). ***Note to Editor: make the word Timestamp in the above sentence non-breaking.***
243	'17	When multiple MPCPDUs are transmitted within a single envelope, all these MPCPDUs have the same timestamp value, referencing the transmission time of ESH. When multiple MPCPDUs are transmitted within a single envelope, all these MPCPDUs have the same *Timestamp* field value, referencing the transmission time of the ESH.
247	'39	The timestamp field is generated by MAC Control and is not exposed through the client interface. The *Timestamp* field is generated by MAC Control and is not exposed through the client interface.

Editing Instruction: In Figure 144-5 in states PARSE_OPCODE, PROCESS_TIMESTAMP and TO_OPCODE_SPECIFIC_PROCESS change "Timstamp" to "TimstampRx" where it refers to the variable.

Editing Instruction: In Figure 144-6 in state INSERT_TIMESTAMP change "Timstamp" to "TimstampTx" in 2 places.