1904.2 Support for additional/future subtypes

Glen Kramer, Broadcom

Special VLC subtypes

- In the Receive and Transmit state diagrams, only the ***VLC_config***, ***OMCI_subtype***, and ***OAM_subtype*** need to have special treatment.
  - *VLC_config* need to be receivable without provisioning of any rules (in order to setup the first rule).
  - *OMCI_subtype* payload is not an 802.3 compliant frame and cannot go through MA_DATA interface. It must be sent to a special interface.
  - *OAM_subtype* has special processing to also make it receivable without a rule, in order to make it on par with OMCI and to cut on unnecessary management traffic (the rule is always the same)
- Nothing else requires a special processing.
- After system deployment, new protocols can be tunneled simply by provisioning new rules

Location of LLC sublayer (802.1Q)

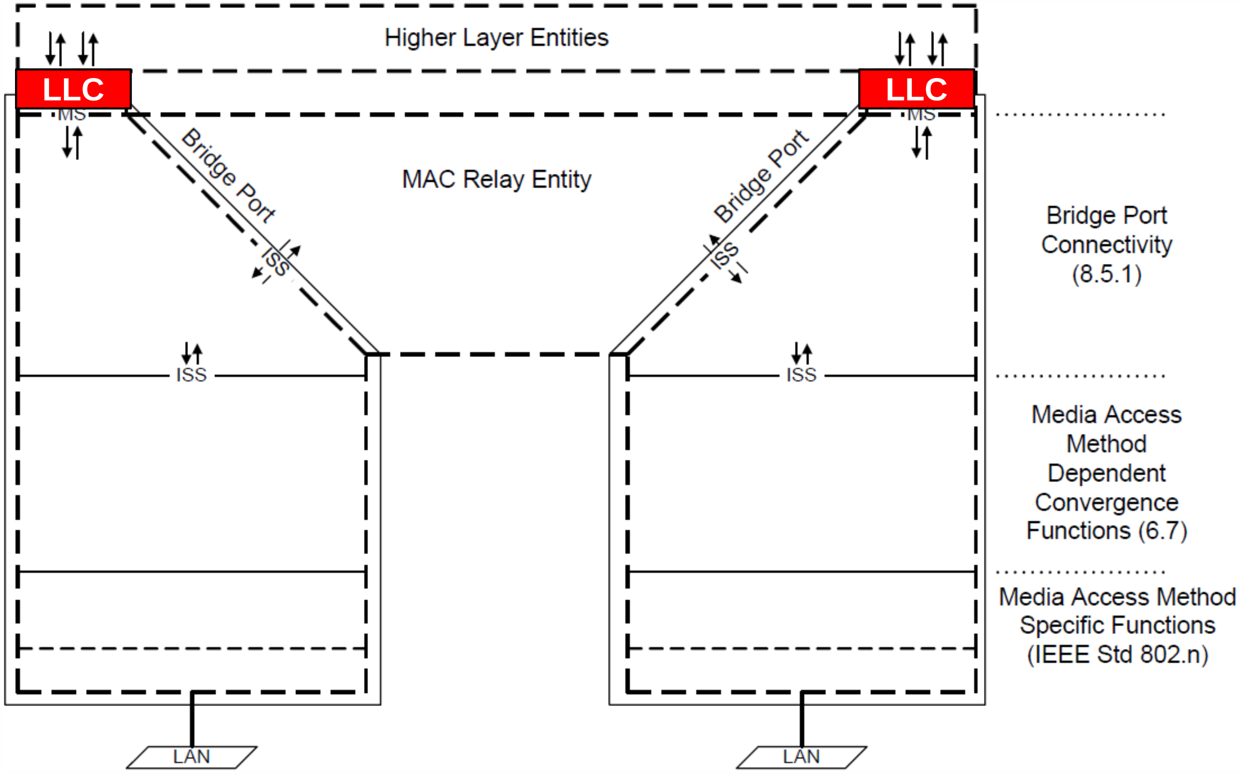

Higher Layer Entities
LLC
LLC
MS
MS
MAC Relay Entity
Bridge Port
Bridge Port
ISS
ISS
ISS
ISS
Bridge Port Connectivity (8.5.1)
Media Access Method Dependent Convergence Functions (6.7)
Media Access Method Specific Functions (IEEE Std 802.n)
LAN
LAN

Figure 8-3—MAC Bridge architecture

- Each Bridge Port also functions as an end station and shall provide the MAC Service to an LLC Entity that operates LLC Type 1 procedures to support protocol identification, multiplexing, and demultiplexing, for PDU transmission and reception by the Spanning Tree Protocol Entity and other higher layer entities. Other types of LLC procedures can also be supported for use by other protocols.

LLC is defined in IEEE 802

- Ipv4,
- IPv6,
- Slow Protocols (except OAM)
- ARP,
- many others

- > 3500 Ethertypes allocated by IEEE RA

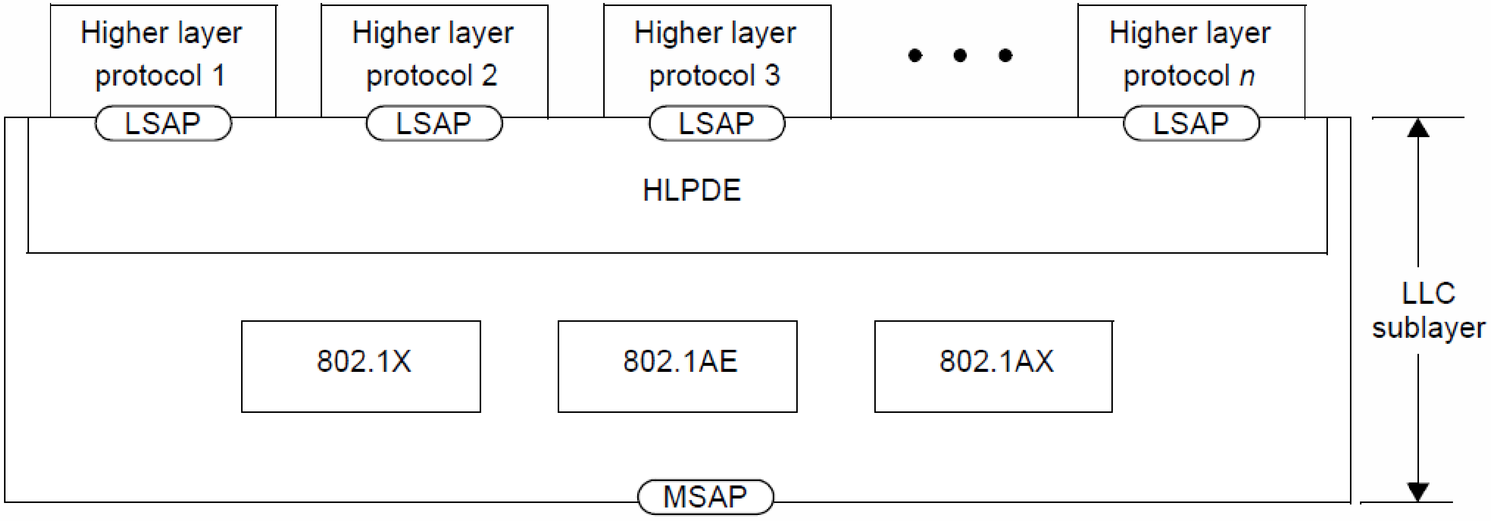


Figure 6—LLC sublayer in 802 RM

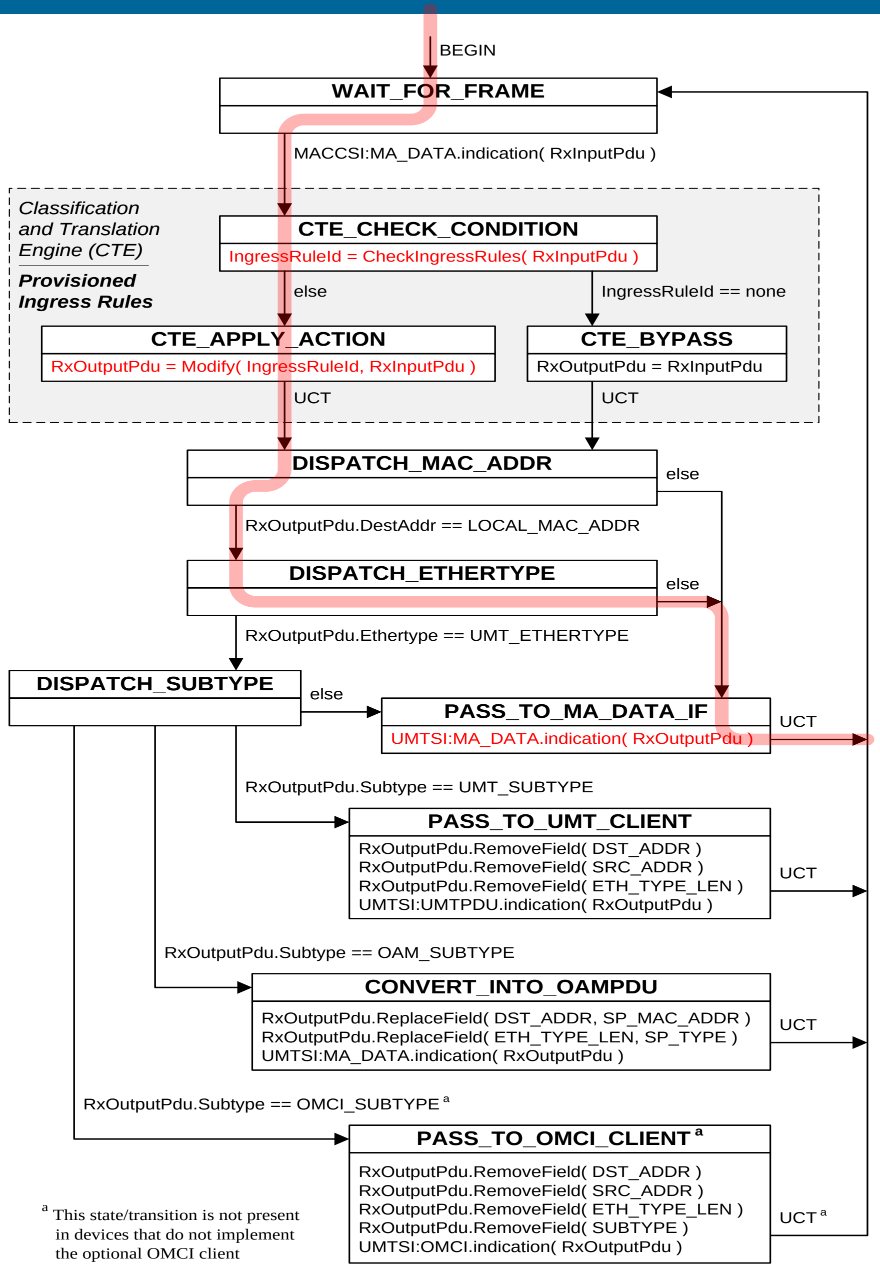
The higher layer protocol discrimination entity (HLPDE) is used by the LLC sublayer to determine the higher layer protocol to which to deliver an LLC sublayer protocol data unit (PDU). Two methods may be used in the HLPDE. The two methods are:

1) EtherType protocol discrimination (EPD), which uses the EtherType value made available to the LLC sublayer through the MSAP

2) LLC protocol discrimination (LPD), which uses the addresses defined in ISO/IEC 8802-2, including the Subnetwork Access Protocol (SNAP) format

The EPD method shall be the primary specified means for protocol identification at the LLC sublayer in IEEE 802 standards developed after January 2011, excluding amendments to existing standards.

Path through Rx State Diagram



BEGIN

WAIT_FOR_FRAME

MACCSI:MA_DATA.indication(RxInputPdu)

Classification and Translation Engine (CTE)

Provisioned Ingress Rules

CTE_CHECK_CONDITION

IngressRuleId = CheckIngressRules(RxInputPdu)

else

IngressRuleId == none

CTE_APPLY_ACTION

RxOutputPdu = Modify(IngressRuleId, RxInputPdu)

CTE_BYPASS

RxOutputPdu = RxInputPdu

UCT

UCT

DISPATCH_MAC_ADDR

else

RxOutputPdu.DestAddr == LOCAL_MAC_ADDR

DISPATCH_ETHERTYPE

else

RxOutputPdu.Ethertype == UMT_ETHERTYPE

DISPATCH_SUBTYPE

else

PASS_TO_MA_DATA_IF

UMTSI:MA_DATA.indication(RxOutputPdu)

UCT

RxOutputPdu.Subtype == UMT_SUBTYPE

PASS_TO_UMT_CLIENT

RxOutputPdu.RemoveField(DST_ADDR)
RxOutputPdu.RemoveField(SRC_ADDR)
RxOutputPdu.RemoveField(ETH_TYPE_LEN)
UMTSI:UMTPDU.indication(RxOutputPdu)

UCT

RxOutputPdu.Subtype == OAM_SUBTYPE

CONVERT_INTO_OAMPDU

RxOutputPdu.ReplaceField(DST_ADDR, SP_MAC_ADDR)
RxOutputPdu.ReplaceField(ETH_TYPE_LEN, SP_TYPE)
UMTSI:MA_DATA.indication(RxOutputPdu)

UCT

RxOutputPdu.Subtype == OMCI_SUBTYPE [a]

PASS_TO_OMCI_CLIENT [a]

RxOutputPdu.RemoveField(DST_ADDR)
RxOutputPdu.RemoveField(SRC_ADDR)
RxOutputPdu.RemoveField(ETH_TYPE_LEN)
RxOutputPdu.RemoveField(SUBTYPE)
UMTSI:OMCI.indication(RxOutputPdu)

UCT [a]

[a] This state/transition is not present in devices that do not implement the optional OMCI client

- Once the tunnel exit rule is applied, the xPDU will pass to Frame Relay sublayer via VLCSI:MA_DATA.indication() primitive.
- If xPDU `DST_ADDR ≠ local`, then the xPDU frame is relayed to another port by the Frame Relay entity.
- If xPDU `DST_ADDR = local`, then the xPDU is passed to the LLC sublayer for dispatching to an appropriate protocol client based on Ethertype (EPD method).

Supporting other protocol types

- All protocol types that are supported by LLC can also be supported by VLC by adding appropriate rules and without any changes to the Receive and Transmit state diagrams.

- There are three methods to support higher-layer protocols:

  1) Define a dedicated VLC subtype for each protocol we want to support

  2) Use L2 encapsulation

  3) Define one catch-all subtype for all higher-layer protocols

Method 1: VLC subtype for each protocol

Table 5-1—*Subtype* field encoding

Value	Designation	Description
0x00	*VLC_config*	*VLC_config* subtype identifies *VLC_Request* and *VLC_Response* VLCPDUs used for configuring the VLC Classification and Translation Engine (see 6.1).
0x01, 0x02	n/a	Reserved for VLC Discovery protocol; ignored on reception.
0x03	*OAM_Subtype*	*OAM_Subtype* represents the OAMPDU payload carried within the VLCPDU (see 5.2.2).
0x04	n/a	Reserved; ignored on reception.
0x05	*L2_subtype*	*L2_Subtype* represents a generic Ethernet frame carried within the VLCPDU (e.g., MAC-in-MAC) (see 5.2.4).
0x06	*L3_Subtype*	*L3_Subtype* represents a generic L3 packet (plus TPID) carried within the VLCPDU (see 5.2.5).
0x07-0x0B	n/a	Reserved; ignored on reception.
0x0C	*OMCI_Subtype*	*OMCI_Subtype* represents the OMCI payload carried within the VLCPDU (see 5.2.3).
0x0D	***IPv4_Subtype***	**…**
0x0E	***IPv6_Subtype***	**…**
0x0F	***ARP_Subtype***	**…**
…	**…**	**…**
… to 0xFD	n/a	Reserved; ignored on reception.
0xFE	*OUI24_Subtype*	*OUI24_Subtype* represents an organization-specific payload carried within the VLCPDU. The organization is identified by a unique OUI/CID value (see 5.2.6).
0xFF	*OUI36_Subtype*	*OUI36_Subtype* represents an organization-specific payload carried within the VLCPDU. The organization is identified by a unique OUI-36 value (see 5.2.6).

With Method 1, we need one entrance and one exist rule per each subtype per each port

Exit rule:

```
IF( DA == local AND
    Ethertype == VLC_TYPE AND
    Subtype = XX_subtype )
THEN
    REPLACE( Ethertype, TPID);
    REMOVE( Subtype );
```

Overhead is 1 byte per VLCPDU

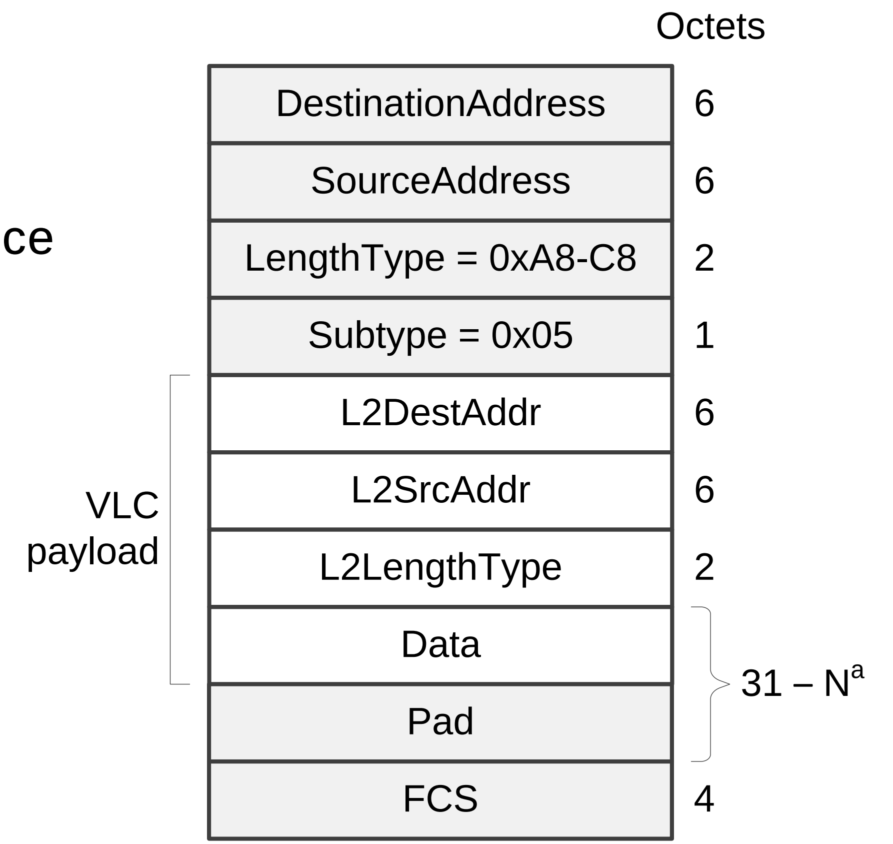
(xPDU Ethertype is replaced with VLC Ethertype + Subtype)

Method 2: L2 Encapsulation

- ❑ This is a universal subtype that can carry any L2 frame as a payload
  - – Can carry any protocol
- ❑ With Method 2, we need one entrance and one exist rule per port
  - – Exit rule:

```
IF( DA == local AND
    Ethertype == VLC_TYPE AND
    Subtype = 0x05 )
THEN
    REMOVE( DST_ADDR );
    REMOVE( SRC_ADDR );
    REMOVE( ETHERTYPE );
    REMOVE( SUBTYPE )
```

- ❑ Overhead per VLCPDU is 15 bytes (an entire Ethernet frame header + Subtype)



		Octets
	DestinationAddress	6
	SourceAddress	6
	LengthType = 0xA8-C8	2
	Subtype = 0x05	1
VLC payload	L2DestAddr	6
	L2SrcAddr	6
	L2LengthType	2
	Data	31 – N[a]
	Pad	
	FCS	4

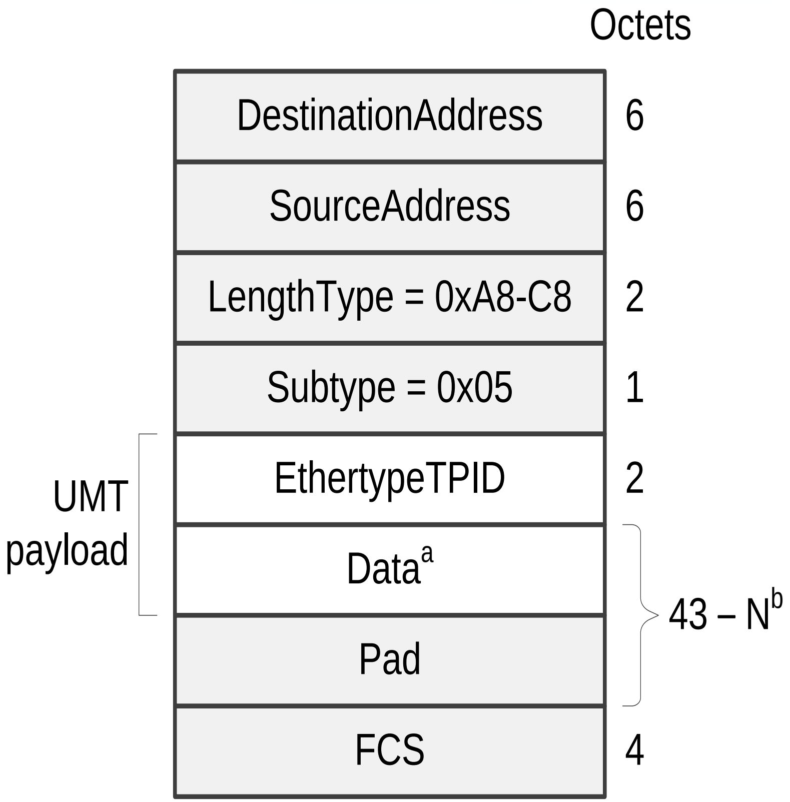
a – Maximum field length depends on frame type (see Figure 5-1).

Method 3: Catch-all subtype

- ❑ Catch-all subtype (*L3_subtype*) is already defined
- ❑ *L3_subtype* represents all protocols that are to be handled by LLC sublayer
  - – *EthertypeTPID* field identifies the actual protocol
- ❑ With Method 3, we need one entrance and one exist rule per port
  - – Exit rule:

```
IF( DA == local AND
    Ethertype == VLC_TYPE AND
    Subtype = 0x06 )
THEN
    REMOVE( Ethertype );
    REMOVE( Subtype )
```

- ❑ Overhead per VLCPDU is 3 bytes (VLC Ethertype + Subtype)



		Octets
	DestinationAddress	6
	SourceAddress	6
	LengthType = 0xA8-C8	2
	Subtype = 0x05	1
UMT payload	EthertypeTPID	2
	Data[a]	43 – N[b]
	Pad	
	FCS	4

a – Field format depends on the value of *EthertypeTPID* field.
b – Maximum field length depends on frame type (see Figure 5-1).

Conclusion

- In the state diagrams, only the *VLC_config*, *OMCI_subtype*, and *OAM_subtype* need to have special treatment.
  - *VLC_config* need to be receivable without provisioning of any rules (in order to setup the first rule).
  - *OMCI_subtype* is not an 802.3 compliant frame and cannot go through MA_DATA interface. It must be sent to a special interface.
  - *OAM_subtype* has special processing to also make it receivable without a rule, in order to make it on par with OMCI and to cut on unnecessary management traffic (the rule is always the same)
- Nothing else requires a special processing.
- After system deployment, new protocols can be tunneled, by simply provisioning new rules

Comparing 3 methods

	Method 1 Dedicated subtype	**Method 2** *L2_Subtype*	**Method 3** *L3_Subtype*

Number of rules needed
S = number of subtypes
P = number of ports
	☒
	S x P
		☑
		P
			☑
			P

Tunneling overhead per frame	☑ 1 byte	☒ 15 bytes	☑ 3 bytes
Scalability (how many protocols can be supported)	☒ A new subtype is required for each supported protocol. Subtype code-point space is limited	☑ One subtype covers all protocols	☑ One subtype covers all protocols
Extendibility (how easy it is to add support for new protocol)	☒ Standard needs to be modified every time a user wants to tunnel a new protocol	☑ Any protocol can be tunneled without changes to the standard	☑ Any protocol can be tunneled without changes to the standard

Conclusion

- *L3_subtype* is useful as a catch-all subtype. We should keep it.
- *L3_subtype* provides the best balance between scalability, extendibility, tunneling overhead, and the number or rules.
- *L3_subtype* is a bad name, because any protocol (any Ethertype/TPID) can be tunneled, not only L3 protocols
- A better name is ***EPD_subtype***
  - **EPD = Ethertype Protocol Discrimination**
    - A method to distinguish protocols based on Ethertype
    - Defined in IEEE 802
    - Default method for LLC sublayer
  - See slide 4

Thank You