FRED in support of multi ULPs

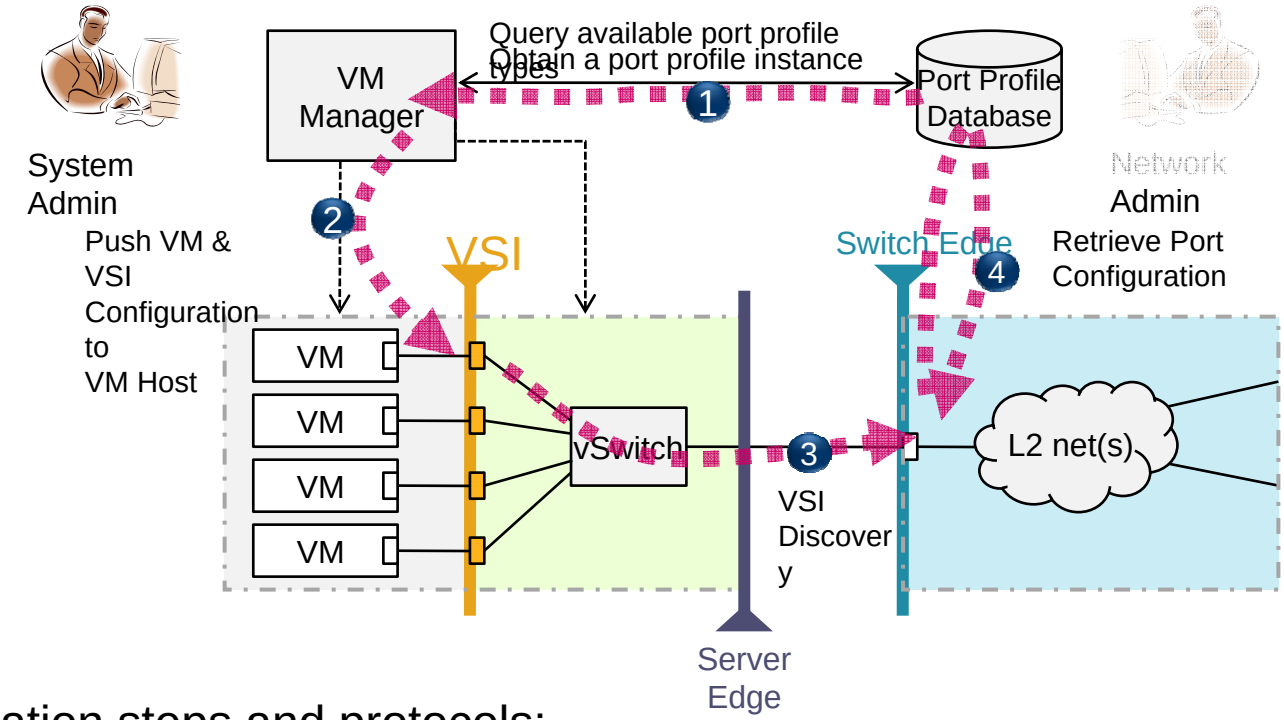
For EVB Discovery and Configuration

Please, help Fred get to the next level…

v52
February 9, 2010

Uri Elzur, Broadcom Ilango Ganga, Intel Manoj Wadekar, Qlogic

VSI Discovery in the context of EVB



Configuration steps and protocols:

- DCBX (Optional) over LLDP
- EVB Discovery is an ordered set of protocols
  - Multi Channel Discovery (Optional) - over LLDP
  - Multi Channel Setup (e.g. SVID assignment) over LLDP?
  - Discovery of Host Switching (VEPA, VEB, Direct Access, PE) over LLDP
- VSI Discovery and Configuration new Transport
- Port Extender Multicast tables distribution new Transport

Per channel MUTUALLY EXCLUSIVE

Clarifying Requirements

❶ **Classic: App and Transport vs. IEEE: "ULP" and "Transport"**
- What required Low Level semantics are sufficient to relieve ULP of ANY transport chores?
- Assuming ULPs have their own Reliability, Flow Control and Recovery
  - What is the benefit of the low level mechanism?
    - Service for multiple ULPs i.e. Multiplexing/Demux? Sharing Rx buffer?

- **Combine "ULP" and "Transport"?**

❷ **What are the ULPs that need to share the wire?**
- Current understanding is that VSI and MC table distribution are NOT sharing a single SVID channel!
- Final list? Similar requirements?

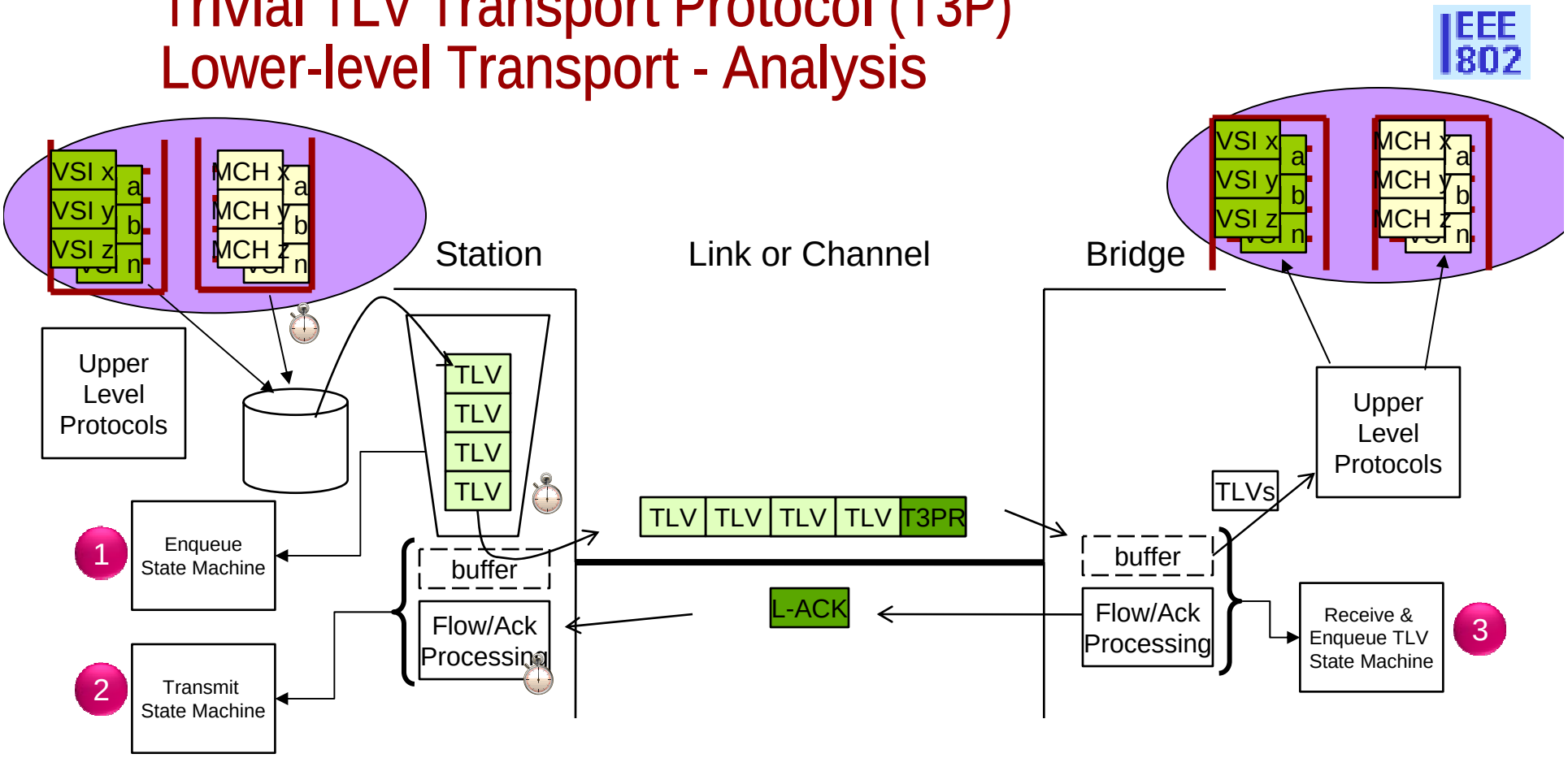
- **Will there be a future ULP using this underlying Transport?**
  - Some people mention collecting Statistics ????

❸ **Design point (Station and Bridge)**
- Constraints (buffers, processing speed, external access to DB etc.)
- What is the realistic pace of info exchange?
- What is the required "Real Timeliness" of the whole architecture?
- Amount of outstanding ULP "state": 1500B, multiple MTU?

Roles and features of the “Transport”

	Feature	ULP	FRED		T3P	
Transport				Total "processing Energy"		Total "processing Energy"
	Reliability	ACK	Fast Re-Tx	1	ACK	2
	Flow Control	yes	Timer	1	ACK	2
	Sequencing	yes	no		yes	
	ULP independent Multiplexing	n/a	yes	1	Tx "starvation", Rx dependency	1
	Immediate TX		yes, per ULP	0	yes, per TLV; "walk the Tx Q"	1
	multi outstanding PDU		yes, per ULP		One	
	"walk" the PDU on Rx	1	1st TLV	1.1	1 (T3P) + 1 (ULP)	2
Failures						
	ULP Queue stuck		loose efficiency, but other ULP not blocked		Other ULP may be blocked	
	PDU loss	ULP Sequencing and re-Tx	Fast Re-Tx makes this rare (so ULP can handle), keep low level simple. No detection scheme		ACK, Sequencing and Health at low level	

Trivial TLV Transport Protocol (T3P) Lower-level Transport - Analysis



A. Double Tx queue. ULP must have an independent queue anyhow. A Lock mechanism needed to share Enqueue FSM's single queue

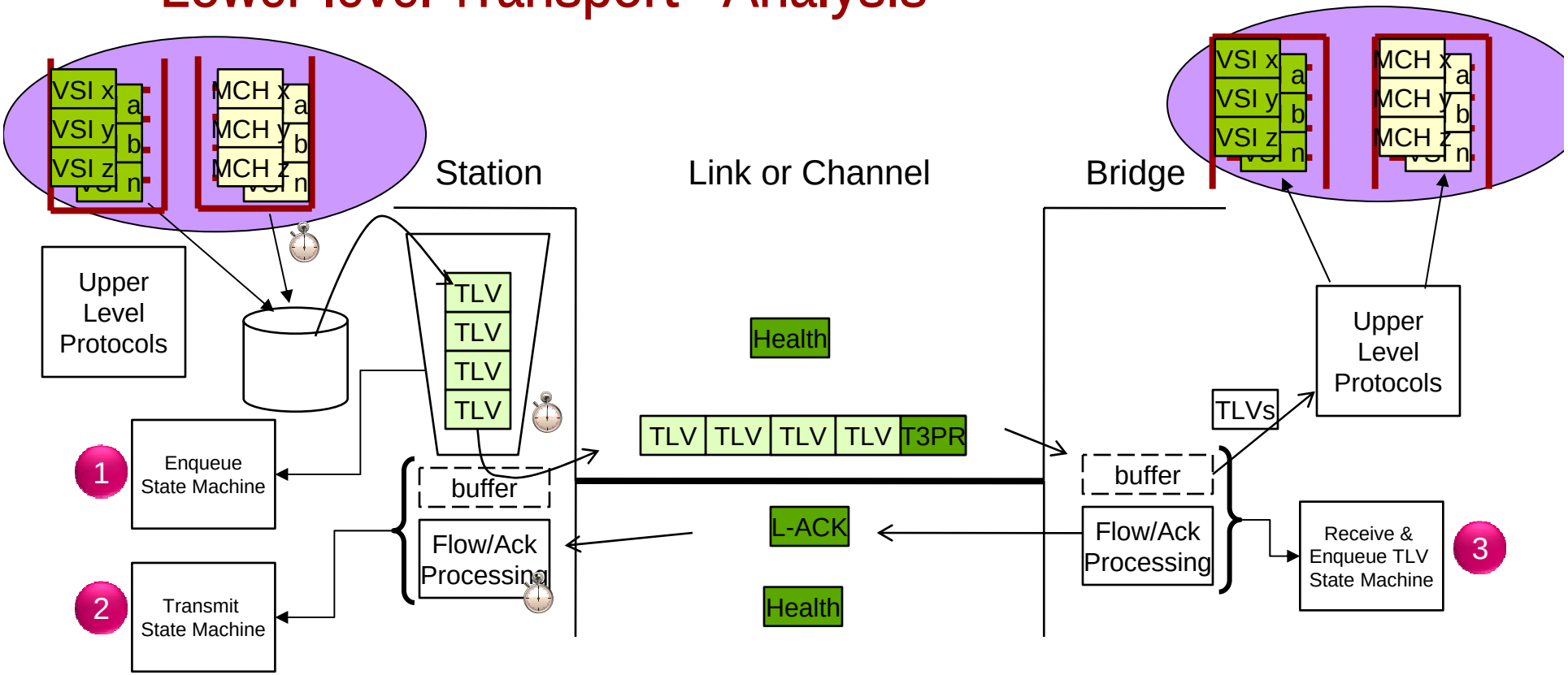
B. Starvation is possible, as Enqueue's queue at Origin can be overtaken by one ULP and one "stuck" ULP stops the other one

C. Rate determined by slowest ULP (if Rx queues are full)

D. Each ULP may have different TLV aggregation policies and OTimer value preferences, while only one is avail. "IMMEDIATE" may require walking over the Q!

With FRED: each ULP transmits at will. The Low level mechanism prevents starvation and provides fair arbitration.

Trivial TLV Transport Protocol (T3P) Lower-level Transport - Analysis



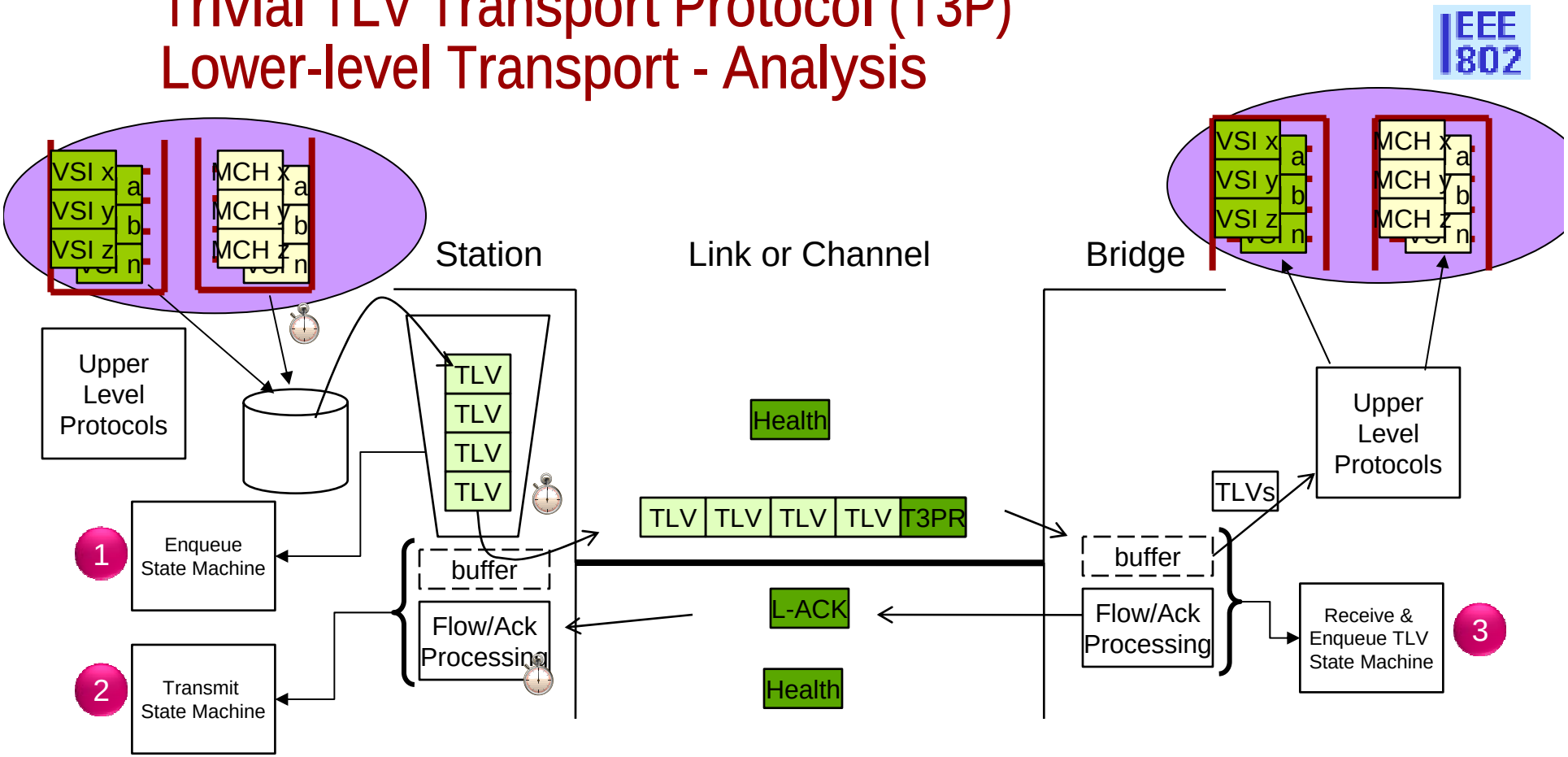
E. ULP must have an ACK, Sequencing for in-order processing and echo of TLV to sender as well as checks over request content and state transitions. Why Duplicate at the lower level?

F. Assuming, rare drops on link and on end points, ULP re-transmit is more economical than L_ACK

G. Low level ACK, doesn't mean the ULP has received the TLV. It simply means the TLV arrived to the other end. A ULP ACK is still needed to Tx the next TLV, IFF in-order ULP processing is to be GURANTEED. PACING vs ACK!

With FRED: ACK, sequencing at ULP layer only.

Trivial TLV Transport Protocol (T3P) Lower-level Transport - Analysis



H. mixed TLVs in each PDU, requires walking over the TLV for 2 times: once by T3P and another time by the ULP

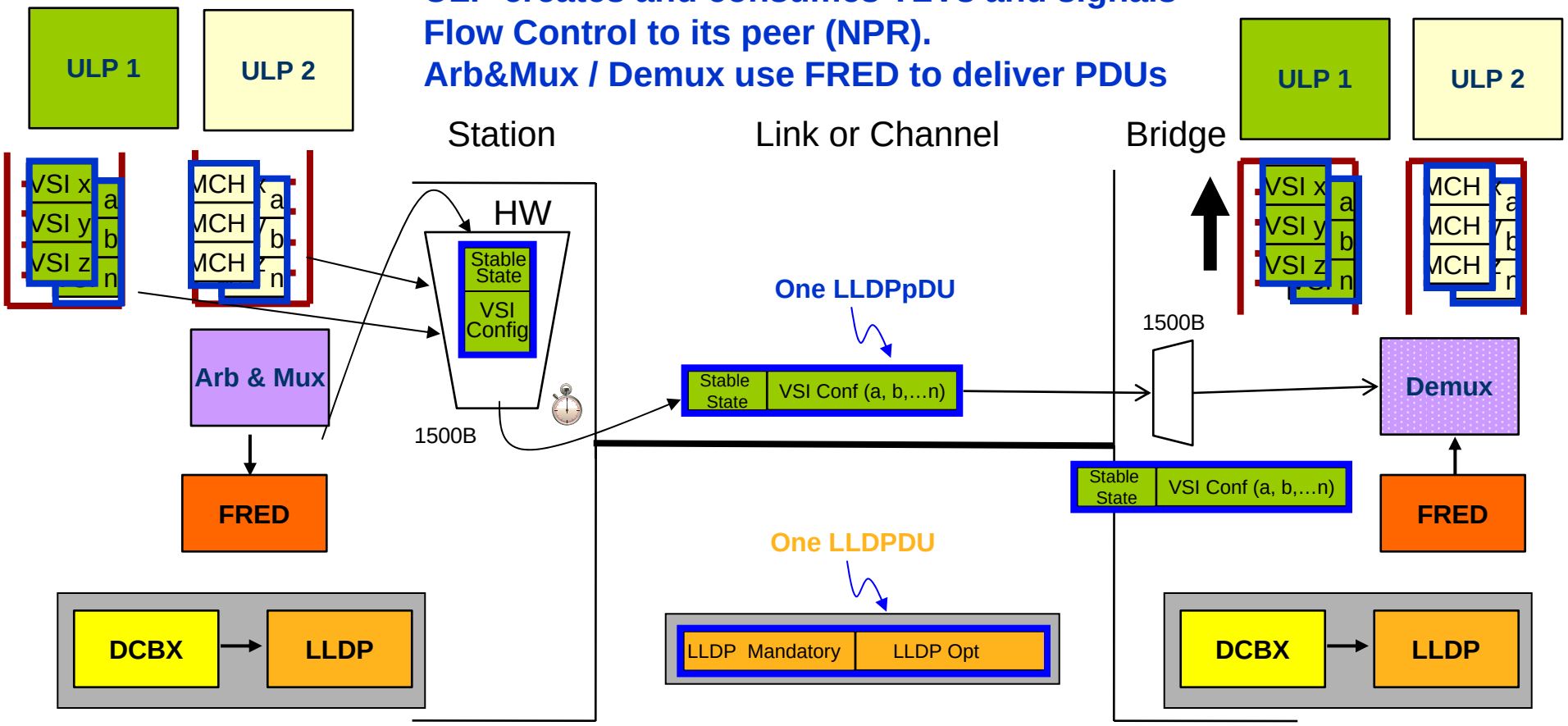
I. "low Level" Transport needs to know all TLVs of the ULP and needs to be updated every time the ULP changes TLV coding

With FRED: one ULP in each PDU, TLV Type decouples Low level form knowing all TLVs

FRED / Multi ULP* - Design Approach

(single direction shown)

ULP creates and consumes TLVs and signals Flow Control to its peer (NPR).
Arb&Mux / Demux use FRED to deliver PDUs



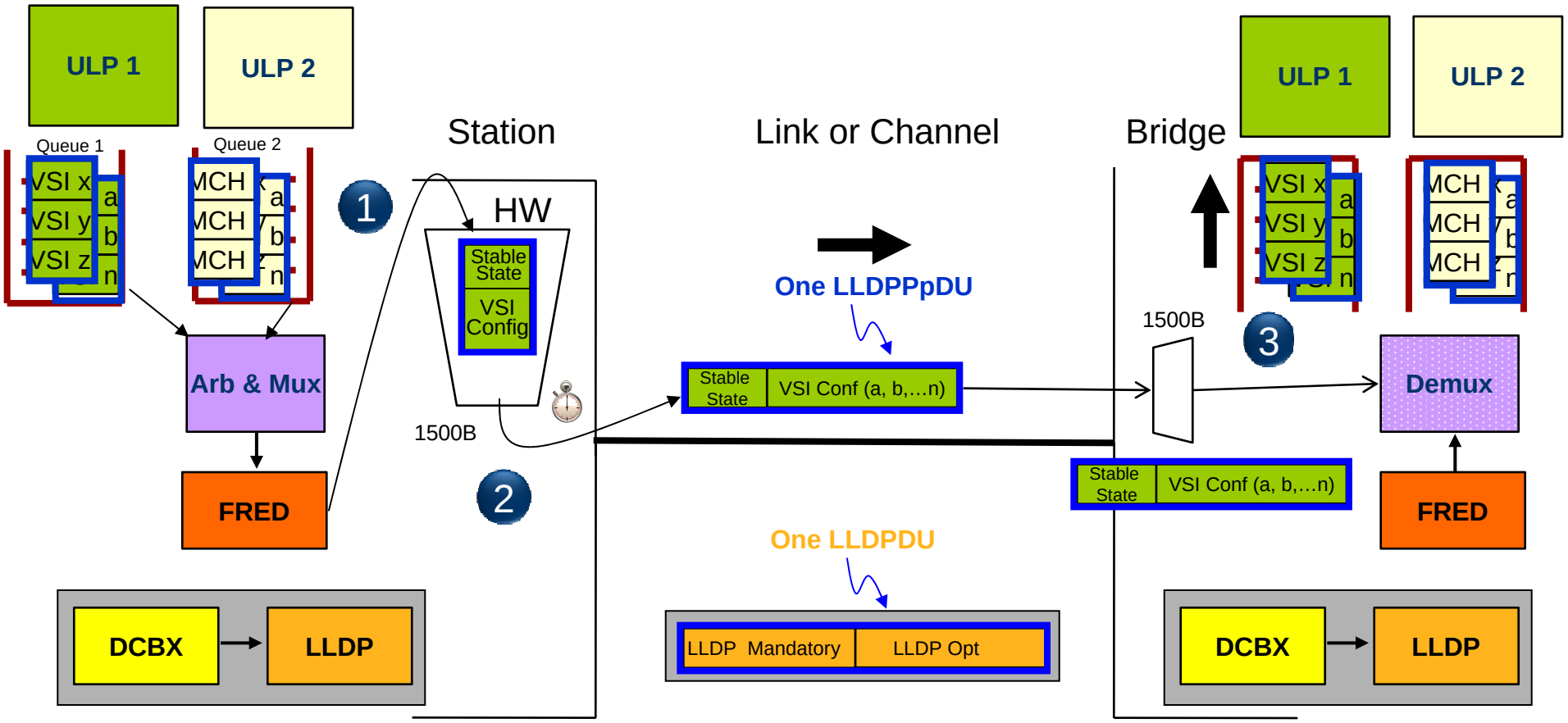
FRED (LLDP+) sets the Basic_Rate of PDU exchange both ends can reliably handle. It further guarantees no loss using LLDP+'s Fast Re-XMT

Each ULP deploys its ACK, Sequencing and checks to suit its needs.

Each ULP paces its traffic based on guaranteed resources it has (NPR).

* - "ULP" are specifically designed for a narrow purpose

FRED / Multi ULP
(single direction shown)



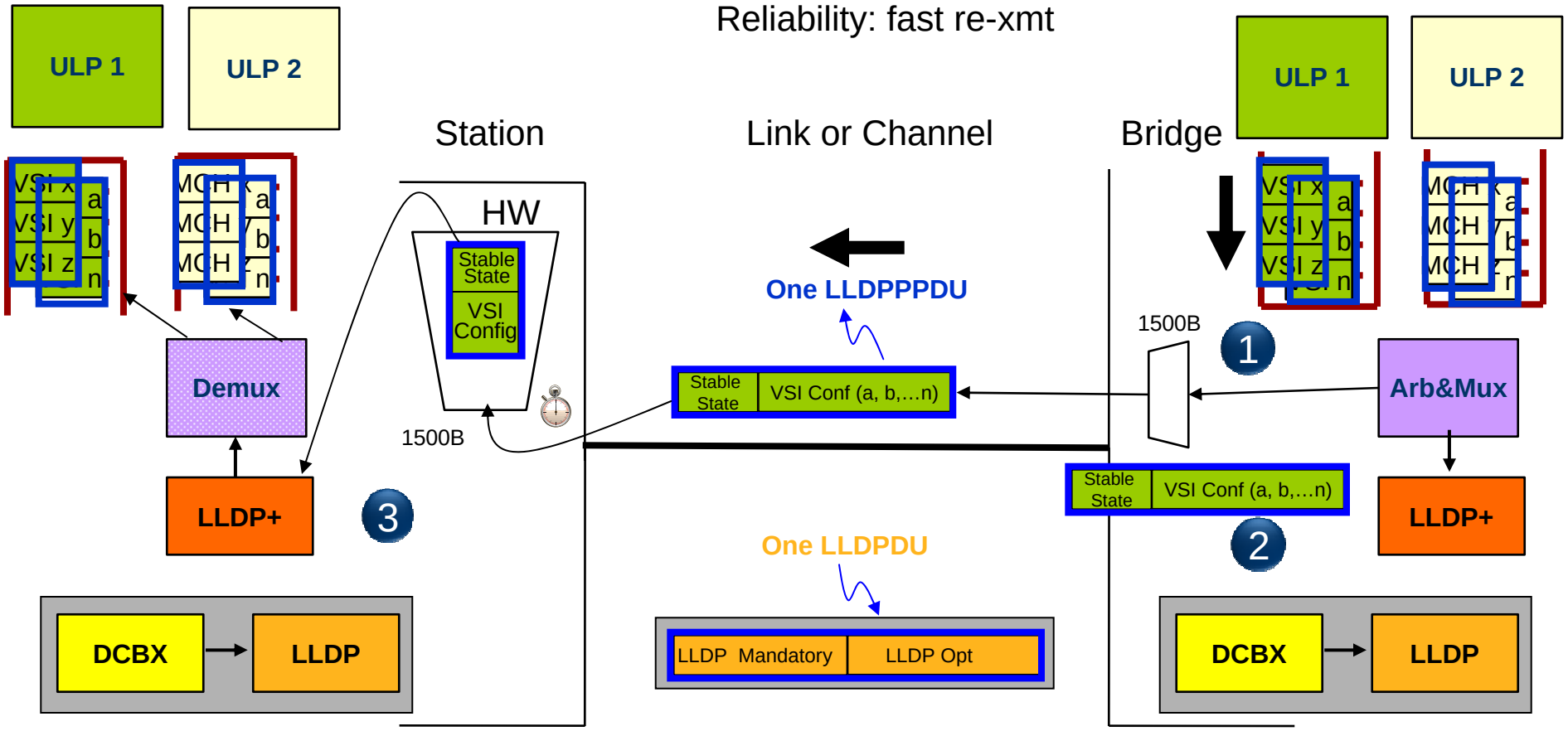
1. Each ULP places its “ULP PDU” with TLVs in its Tx Q. Arb&Mux takes “ULP PDU” worth of data from a ULP that has data. At speed <= Basic_Rate (allows LLDP+ Fast Re-Tx to complete), Arb&Mux updates the Local MIB and informs LLDP+ of “Local MIB Change” event.

2. LLDP+ transmits the new PDU due to LOCAL MIB Change event. Fast Re-Tx follows

3. LLDP+ receives the PDU and informs Demux of a “Remote MIB Change”. Demux consumes LLDPpDU, peers into TLVs to determine the target ULP and places TLVs in right ULP buffer.

FRED / Multi ULP - Bridge response
(single direction shown)

Notes:
Reliability: fast re-xmt

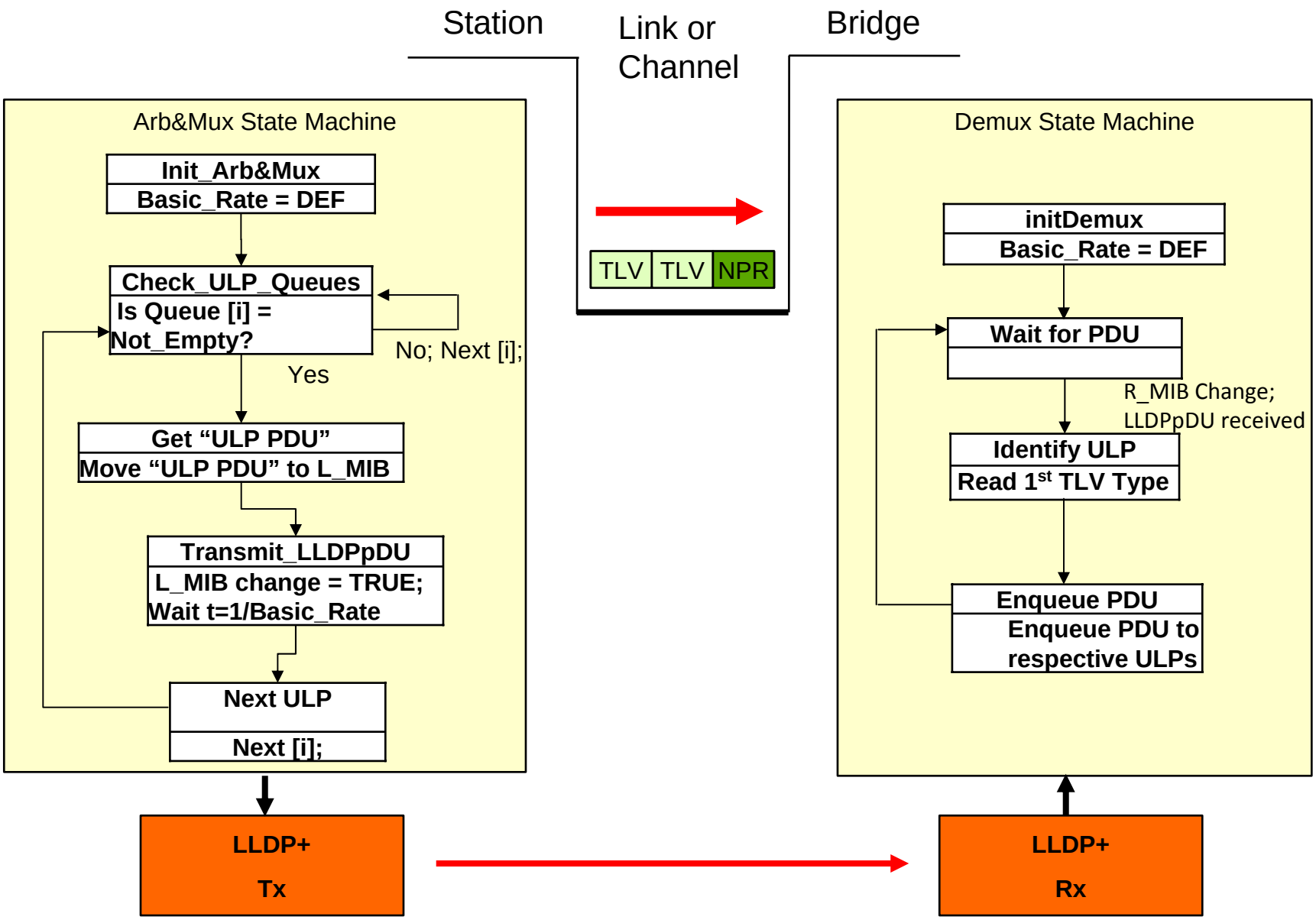


1. Bridge's ULP state machine creates TLVs (NPR is opt.) and places in its Tx Queue. Arb&Mux takes "ULP PDU" worth of data from a ULP that has data. At speed <= Basic_Rate (allows LLDP+ Fast Re-Tx to complete), Arb&Mux updates the Local MIB and informs LLDP+ of "Local MIB Change" event.

2. LLDP+ transmits the new PDU due to LOCAL MIB Change event.

3. LLDP+ receives the PDU and informs Demux of a "Remote MIB Change". Demux consumes LLDPpDU, peers into TLVs and places TLVs in right ULP buffer.

”FRED”- Multi State Machine Overview



FRED: LLDP+ Tx State Machine

MIB CONFIG: No Tx only mode. Must support TxRx mode

Use mibConstrShutdownVDCPDU() To denote other MIB, but same operation

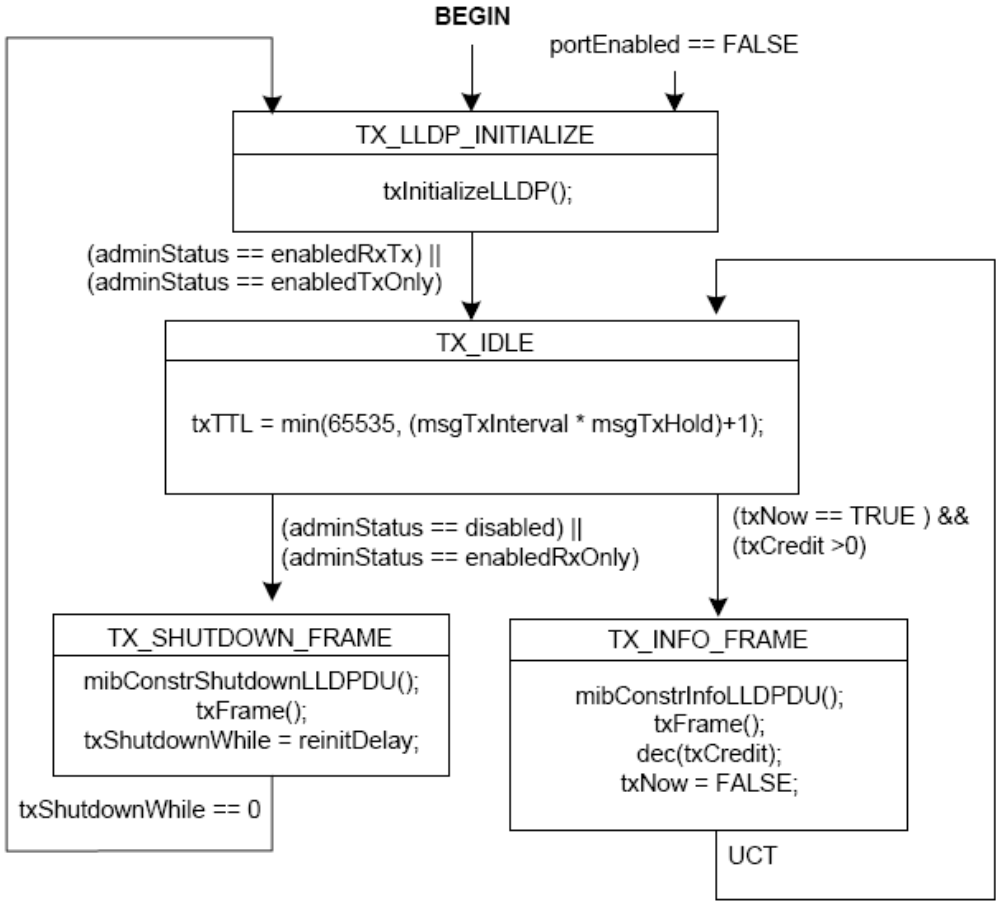


Figure 9-1—Transmit state machine

Use mibConstrinfoVDCPDU() To denote other MIB, but same operation : Mandatory TLVs + blob of “ULP PDU” + EOF TLV

FRED: LLDP+ Timer State Machine

MIB CONFIG (example):

txCreditMax	=	5
txTick	=	100uS
MsgTxInterval	=	1000uS
msgFastTx	=	100uS

Timer resolution: 100 uS

Per 9.2.7.8 localChange ⇦ TRUE causes txFast = tx FastInit

Need to determine what is negotiable if any

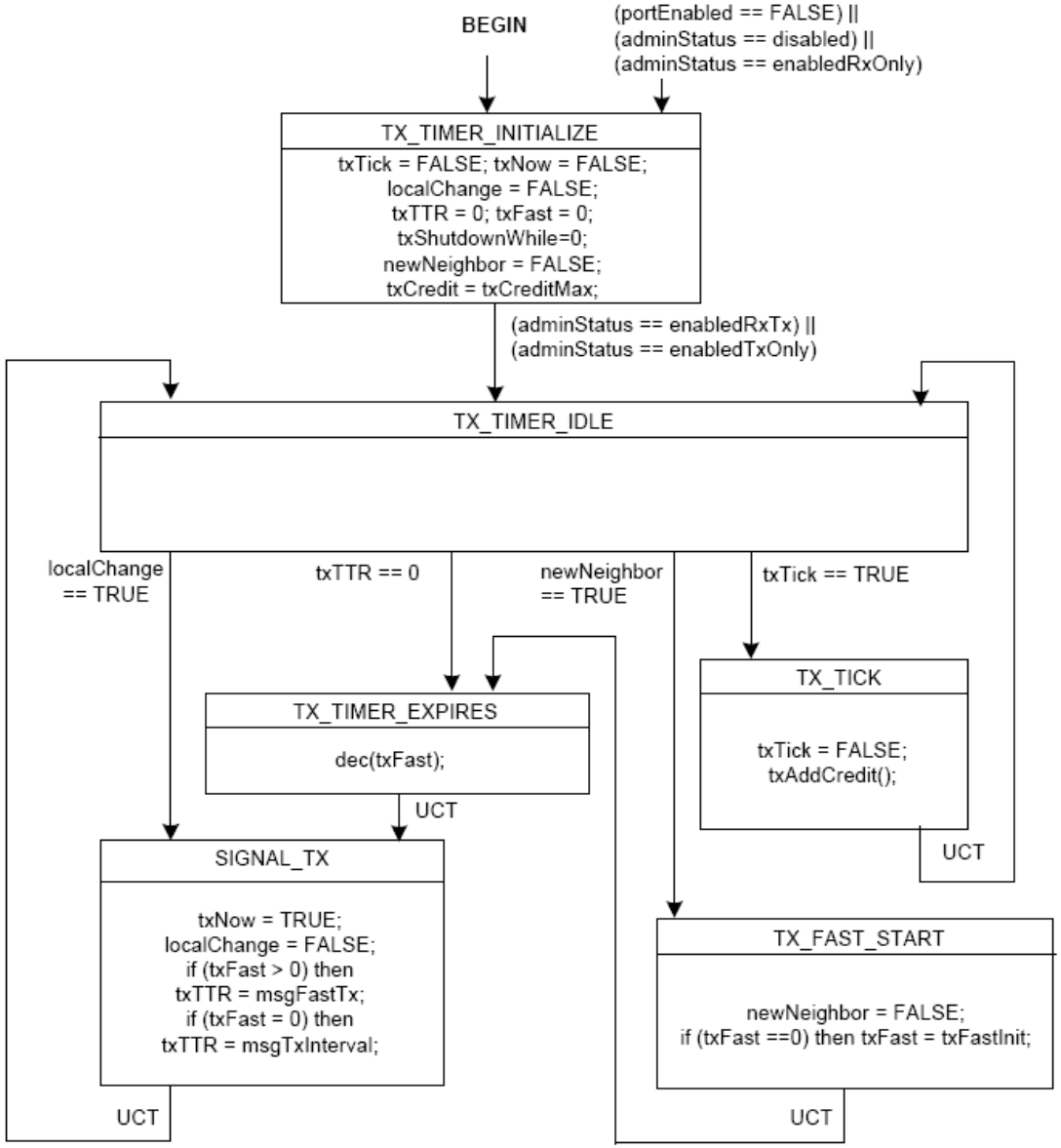


Figure 9-3—Transmit timer state machine

FRED: LLDP+ Rx State machine

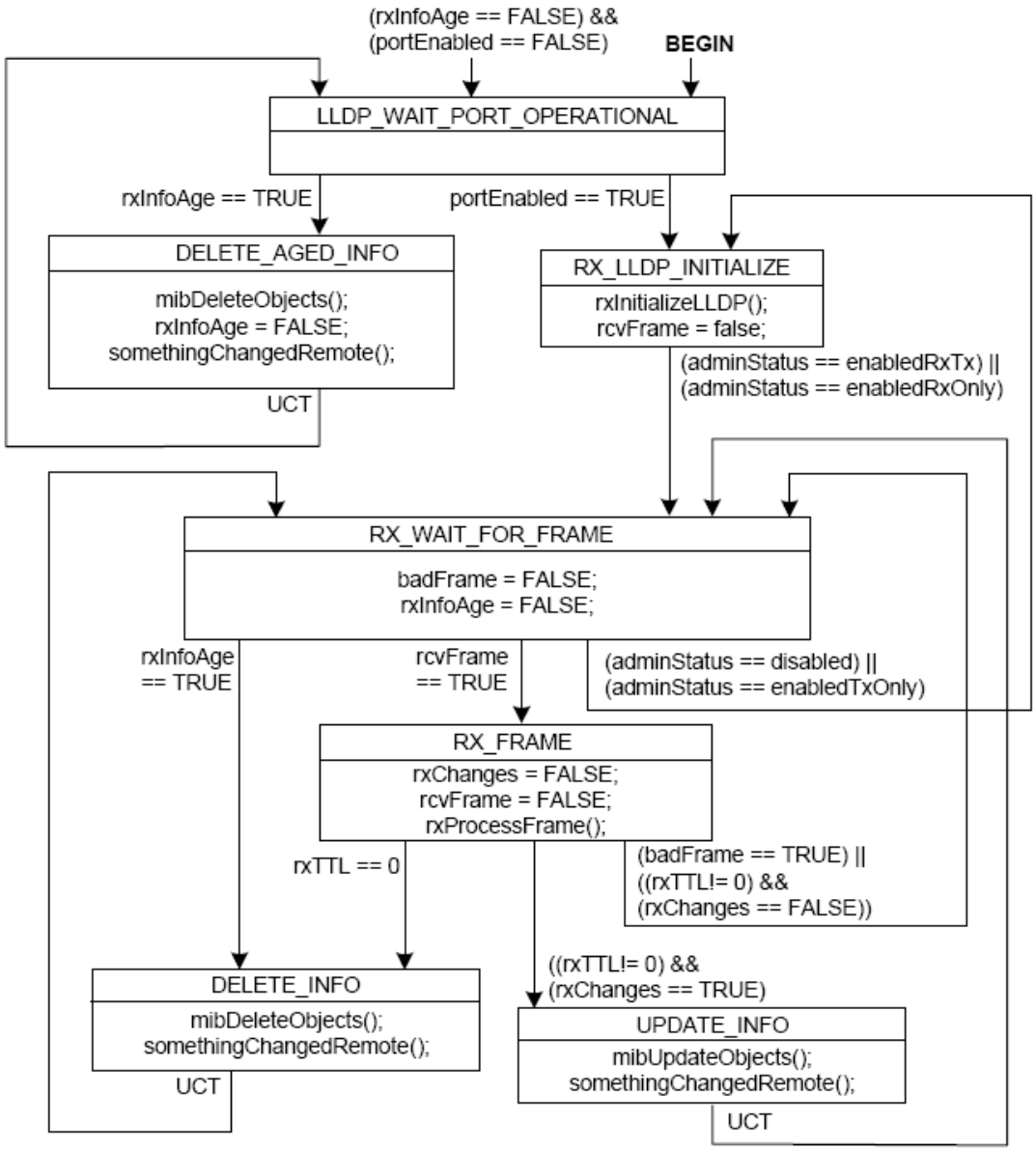


Figure 9-2—Receive state machine

MIB CONFIG: No Rx only mode.
Must support TxRx mode

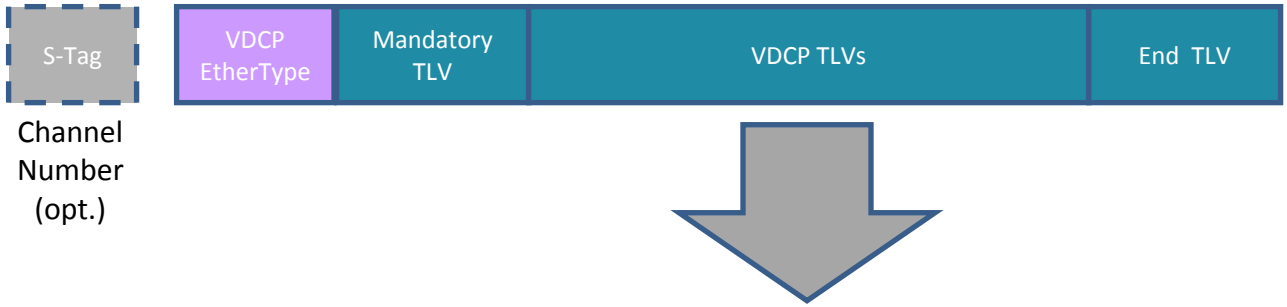
Simple ULP interfaces

- Arb&Mux and Demux interface with ULP using a simple FIFO queue. One entity places items on the queue and the other removes them.
- Like TCP/IP/Ethernet FRED notifies ULP of effective MTU size
  - i.e. fixed overhead consumed by LLDP Mandatory TLVs
- On Tx:
  - ULP places “ULP PDU” on the Tx FiFo
  - If more than one outstanding “ULP PDU” is desired, ULP adds an NPR TLV
- On Rx:
  - ULP gets a “ULP PDU” from the Rx FiFo
- ULP responsibilities
  - ACK, Sequencing, Flow Control
  - Signal the remote whether NPR is supported and how many buffers are avail.

Backup

VDCPDU Capabilities Format (A starting point…)

Frame format for per channel exchanges [B]



Sent once for configuration information exchange. May be combined with VEB configuration

VDCP Capabilities TLV –	Bytes
• FRED parameters (Basic_Rate, msgTxInterval, msgFastTx,…)	32
•EVB Type (VEPA | VEB | VNIC | …)	2
•EVB Type Status	1
•Max number of VSI supported *(V-max)*	1
•Current number of VSI *(V-x)*	2
•Max number of State changing VSI supported *(SC-max)**	2
•# of VSI State Change request in the VSI Configuration TLV *(SC-m)*	2
•*# of Data Bases supported (DB-n)*	2
•*Data base identifier/s (n)*	1
	n*16

OPTIONAL: configurable number of bits per VSI in the Stable State