Link Aggregation Control Protocol - Update

Presentation to the Link Aggregation Task Force, November, 1988

Tony Jeffree

State of Play

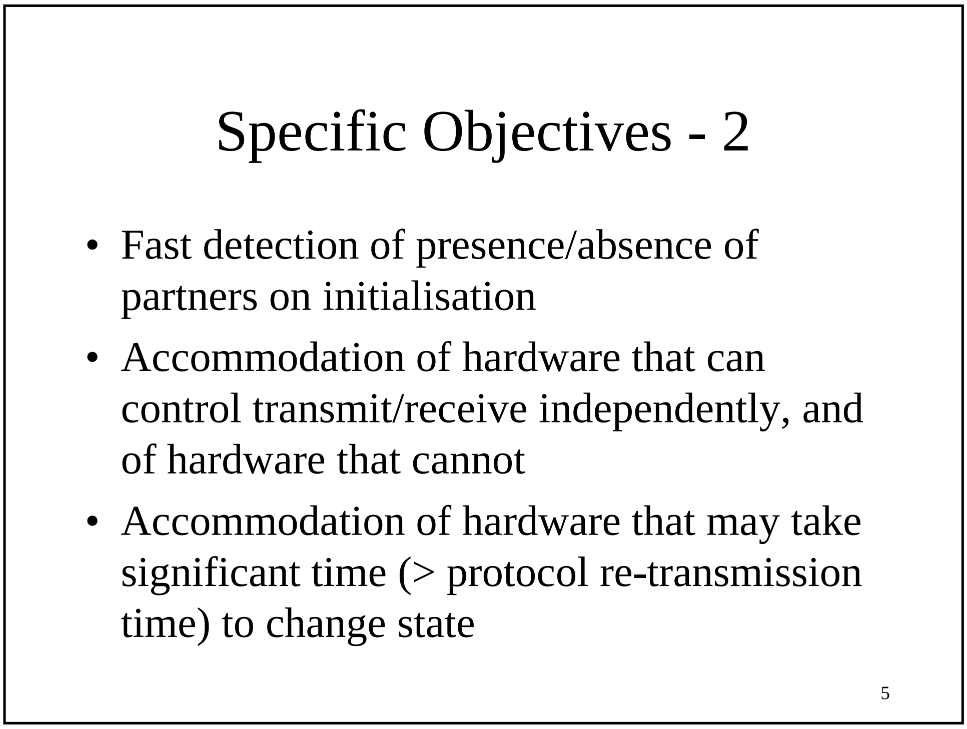
- Further refinement of the design based on functionality previously presented
- Comments taken on board from previous meeting
  - Nervous => Short Timeout/Long Timeout
  - Desirable => Active LACP/Passive LACP
  - Description of variants of Selection rules
- Further work:
  - Nervous, Desirable & TX functionality becomes *Periodic Tx* machine
  - Flush included
  - Churn Detect machine added

Basic assumptions/objectives

- If aggregation is possible, it will happen automatically
- If not possible, links operate normally
- Determinism
- Rapid convergence
- Low risk of misconfiguration
- Low risk of duplication or misordering

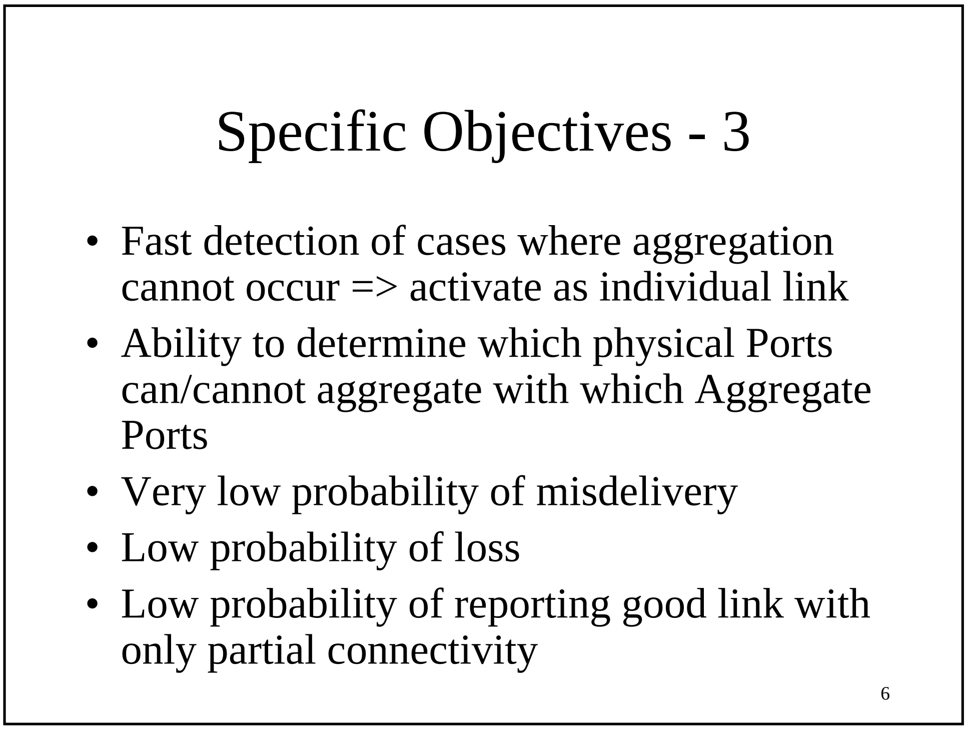
Specific Objectives - 1

- Ability to configure “speak if spoken to” Ports (= Passive LACP) and “speak anyway” Ports (= Active LACP)
- Ability to configure “Long Timeout” operation for Ports that can hardware detect link failure, or “Short Timeout” operation for Ports that cannot



Specific Objectives - 2

- Fast detection of presence/absence of partners on initialisation
- Accommodation of hardware that can control transmit/receive independently, and of hardware that cannot
- Accommodation of hardware that may take significant time (> protocol re-transmission time) to change state



Specific Objectives - 3

- Fast detection of cases where aggregation cannot occur => activate as individual link
- Ability to determine which physical Ports can/cannot aggregate with which Aggregate Ports
- Very low probability of misdelivery
- Low probability of loss
- Low probability of reporting good link with only partial connectivity

Identifying link characteristics

- Many characteristics that contribute
  - Standardised in .3: Link speed, duplex/non-duplex…etc
  - Other characteristics…e.g., administrative, non-standardised
- A Link is allocated a single **Key** value
- All Links in a system that share the same Key can potentially aggregate;
- Links that are not capable of aggregation are allocated unique Keys

Identifying Links that can Aggregate

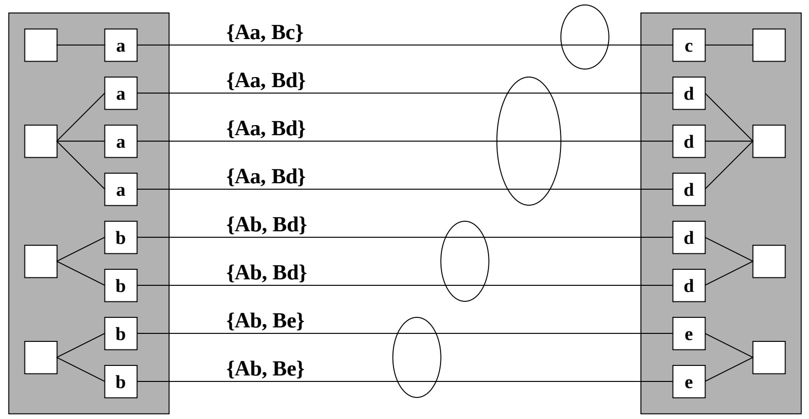
- System ID plus Key provides a global identifier
- The set of links between 2 systems that can aggregate are identified by concatenating the System ID and Key at each end of the links
- Hence, for two systems S and T that use K and L respectively as Key values for some links, then all links with {SK, TL} (interchangeably, {TL, SK}) can aggregate together

Detecting Aggregation possibility

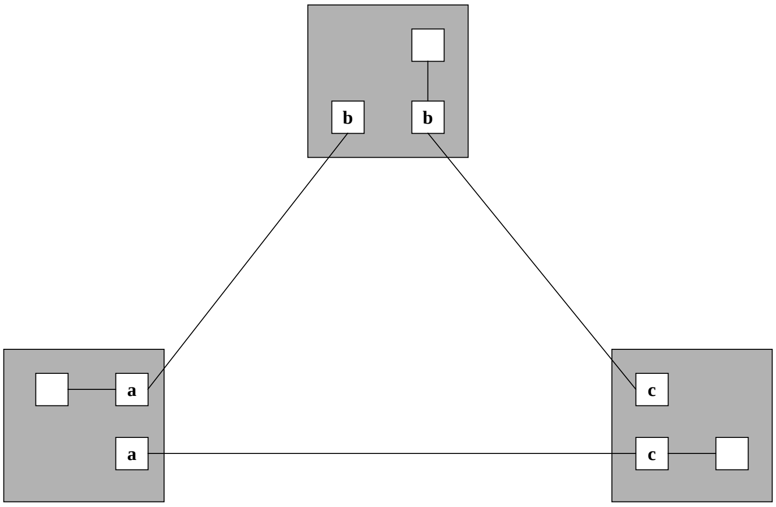
- Aggregation possibility can be detected simply by exchanging System Ids and Keys across a link; each system can then see whether any other Links exist with the same {SK,TL} value.
- If other links in a system exist with the same {SK, TL} then they can all be added to the same Aggregate
- Simplifying assumption: no limit on aggregation size - allocate more capabilities if it is necessary to impose such a limit.

Effect of Keys - Example

System ID = A **System ID = B**



Consequence of too few Aggregators



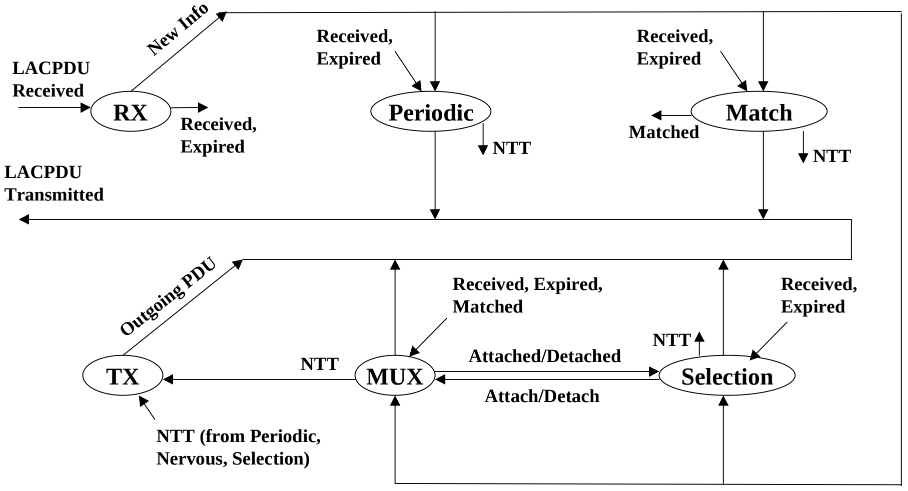
Prevention of Duplication/Reordering

- *Collect* once you are in the right aggregation
- Don't *Distribute* until you know that the other end is Collecting
- Stop Distribution/Collection on a Link prior to moving it to a new aggregation
- BUT also need to accommodate equipment which cannot switch collector/distributor independently
- Need to "flush" other links if Conversations are re-allocated as a result of adding/removing links

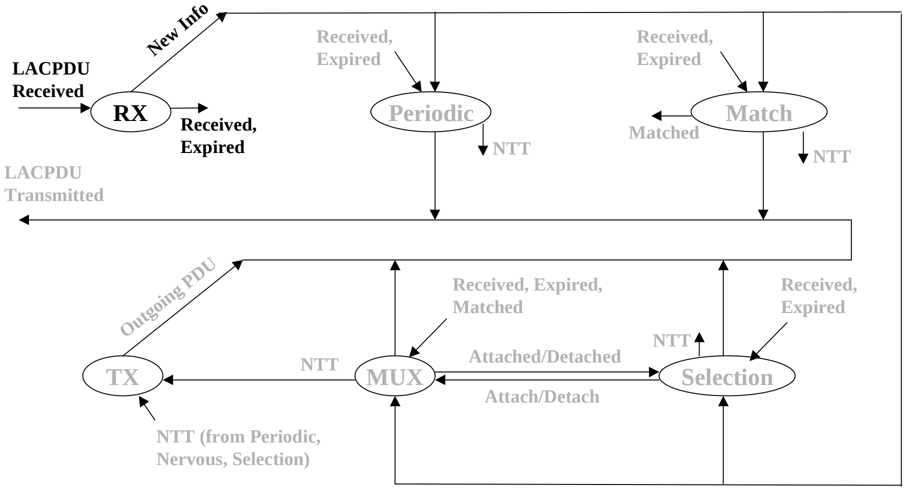
Protocol basics

- If the other guy doesn’t get it, say it again
- Assumption that packet loss is very low
- Communicate *state*, not *commands*
- *Need to Tell (NTT)* if local state has changed, if information is old, or if the other guy does not get it
- Tell the other party what you know. When you are both agreed - aggregate

The Big Picture



RX



New Info

LACPDU Received

RX

Received, Expired

Received, Expired

Periodic

NTT

Received, Expired

Match

Matched

NTT

LACPDU Transmitted

Outgoing PDU

Received, Expired, Matched

Received, Expired

NTT

TX

NTT

MUX

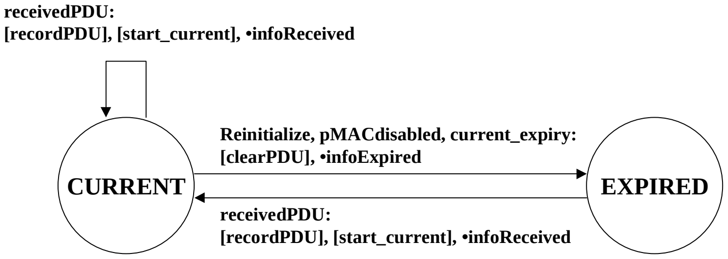
Attached/Detached

Attach/Detach

Selection

NTT (from Periodic, Nervous, Selection)

RX State Machine



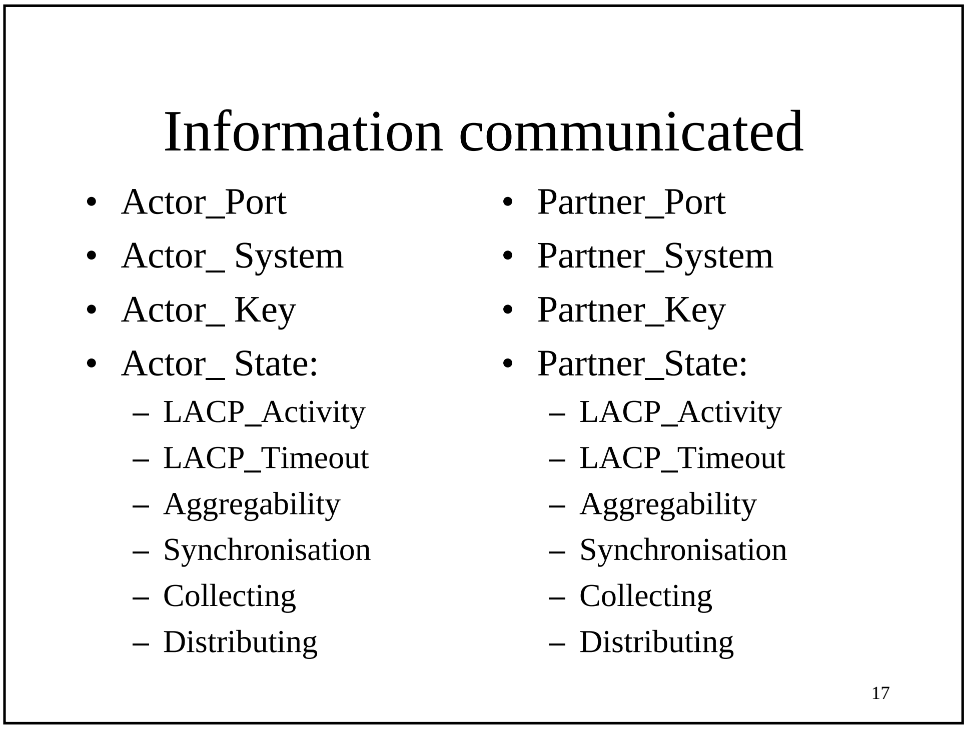
receivedPDU:
[recordPDU], [start_current], •infoReceived

CURRENT

Reinitialize, pMACdisabled, current_expiry:
[clearPDU], •infoExpired

EXPIRED

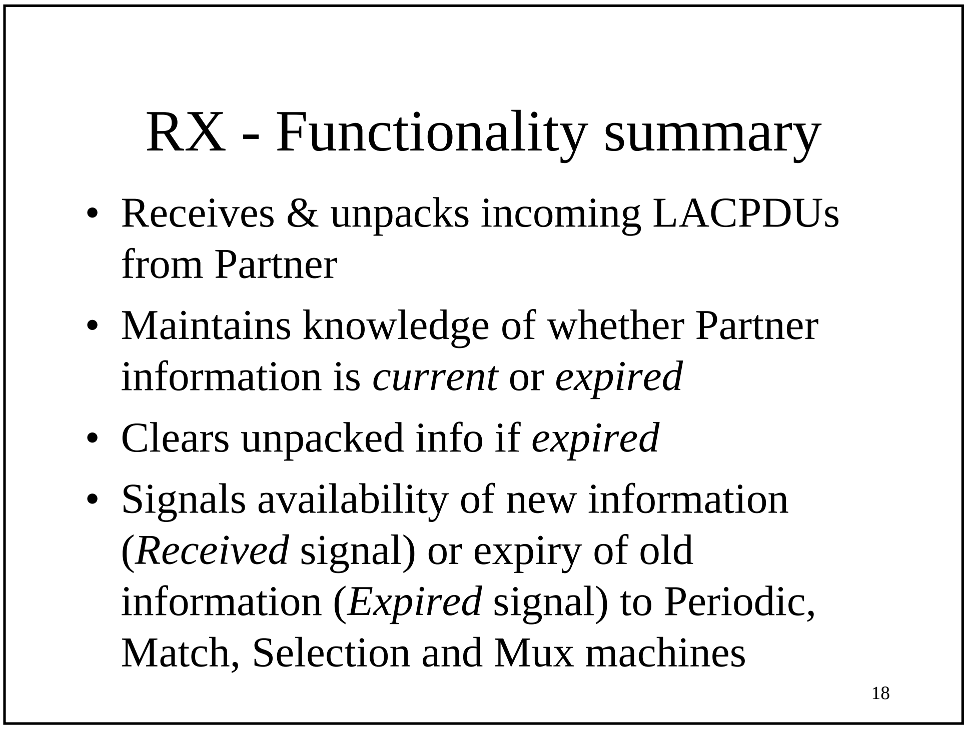
receivedPDU:
[recordPDU], [start_current], •infoReceived



Information communicated

- Actor_Port
- Actor_ System
- Actor_ Key
- Actor_ State:
  - LACP_Activity
  - LACP_Timeout
  - Aggregability
  - Synchronisation
  - Collecting
  - Distributing

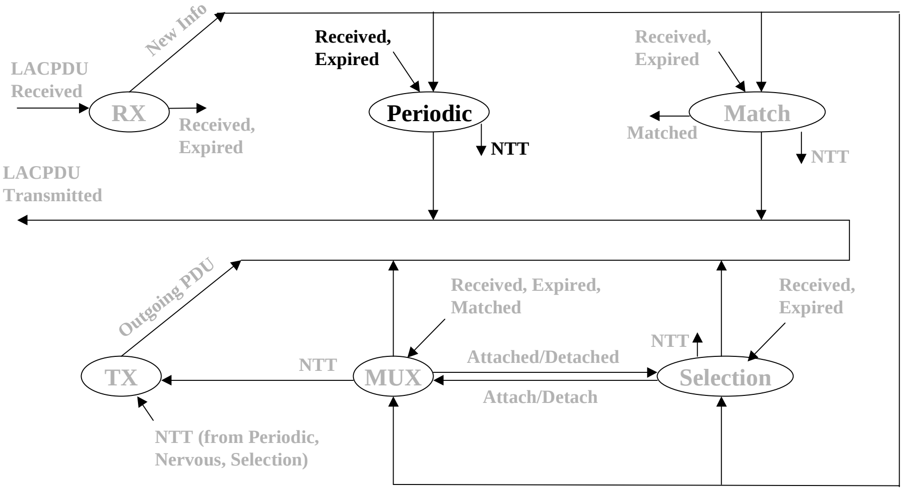
- Partner_Port
- Partner_System
- Partner_Key
- Partner_State:
  - LACP_Activity
  - LACP_Timeout
  - Aggregability
  - Synchronisation
  - Collecting
  - Distributing



RX - Functionality summary

- Receives & unpacks incoming LACPDUs from Partner
- Maintains knowledge of whether Partner information is *current* or *expired*
- Clears unpacked info if *expired*
- Signals availability of new information (*Received* signal) or expiry of old information (*Expired* signal) to Periodic, Match, Selection and Mux machines

Periodic



New Info

LACPDU Received

RX

Received, Expired

Received, Expired

Periodic

NTT

Received, Expired

Match

Matched

NTT

LACPDU Transmitted

Outgoing PDU

Received, Expired, Matched

Received, Expired

NTT

TX

NTT

MUX

Attached/Detached

Attach/Detach

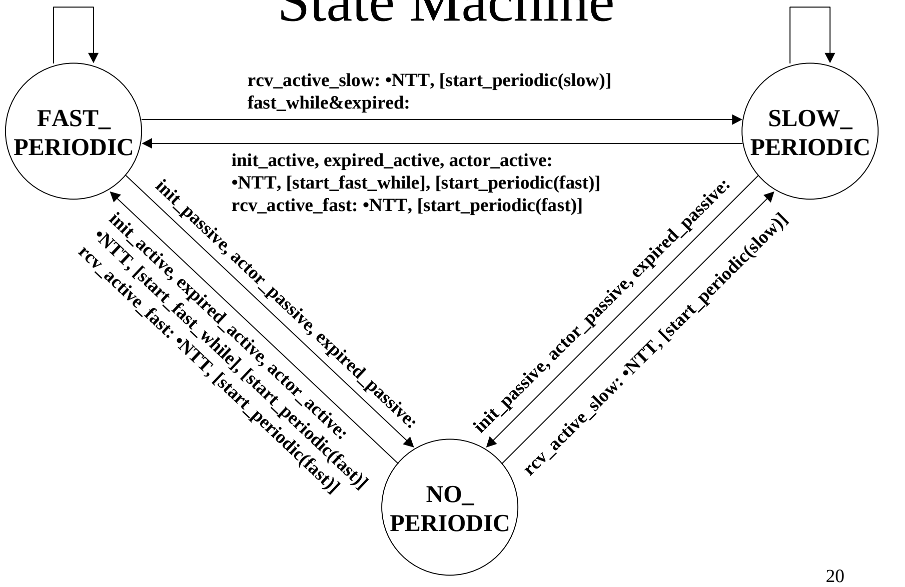
Selection

NTT (from Periodic, Nervous, Selection)

Periodic - State Machine

{expired_active, actor_active, init_active, rcv_active_fast: … }
Periodic_expired:
•NTT, [start_periodic(fast)]

Periodic_expired:
•NTT, [start_periodic(slow)]



FAST_ PERIODIC

SLOW_ PERIODIC

NO_ PERIODIC

rcv_active_slow: •NTT, [start_periodic(slow)]
fast_while&expired:

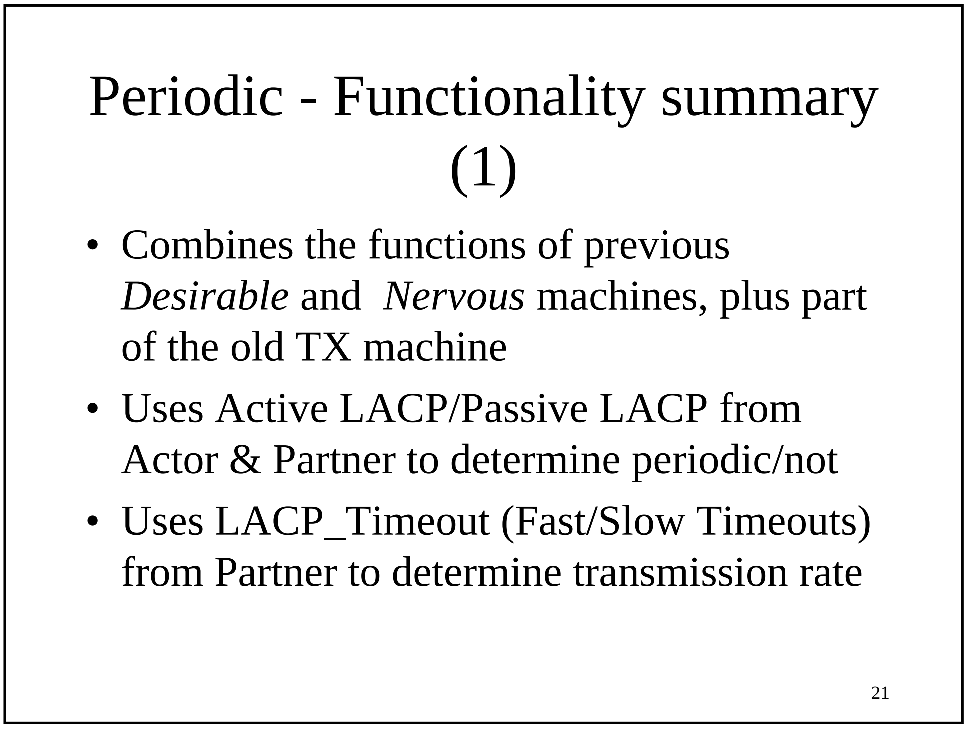
init_active, expired_active, actor_active:
•NTT, [start_fast_while], [start_periodic(fast)]
rcv_active_fast: •NTT, [start_periodic(fast)]

init_passive, actor_passive, expired_passive:

init_active, expired_active, actor_active:
•NTT, [start_fast_while], [start_periodic(fast)]
rcv_active_fast: •NTT, [start_periodic(fast)]

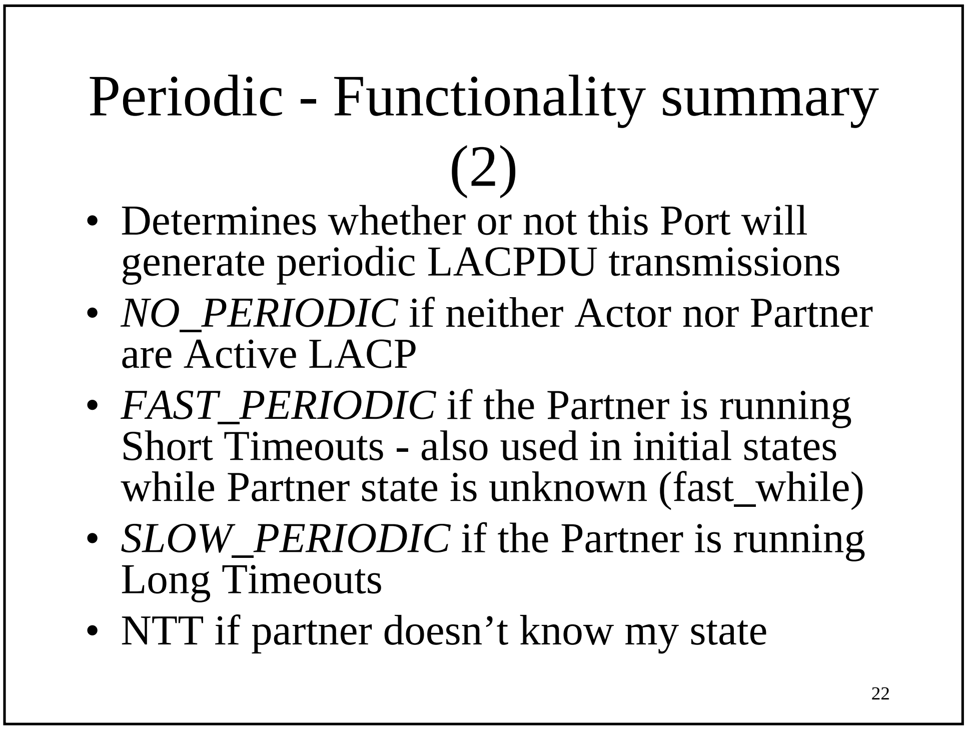
init_passive, actor_passive, expired_passive:

rcv_active_slow: •NTT, [start_periodic(slow)]



Periodic - Functionality summary (1)

- Combines the functions of previous *Desirable* and *Nervous* machines, plus part of the old TX machine
- Uses Active LACP/Passive LACP from Actor & Partner to determine periodic/not
- Uses LACP_Timeout (Fast/Slow Timeouts) from Partner to determine transmission rate



Periodic - Functionality summary (2)

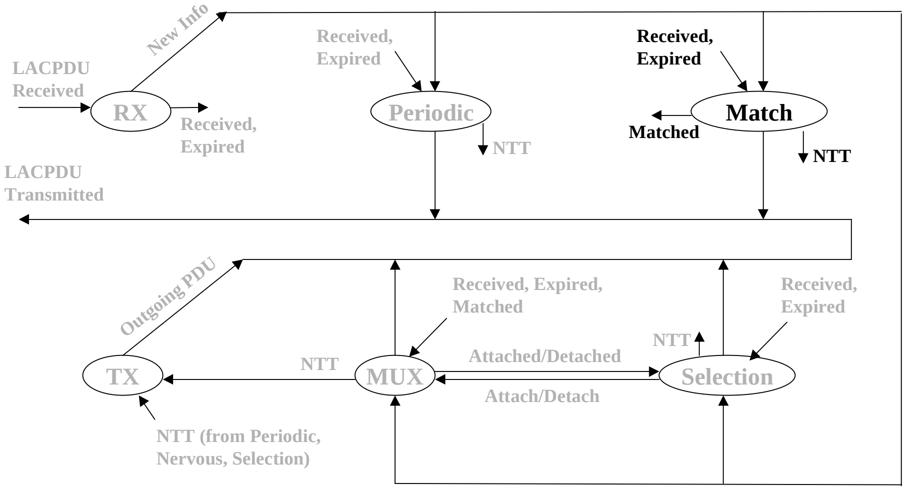
- Determines whether or not this Port will generate periodic LACPDU transmissions
- *NO_PERIODIC* if neither Actor nor Partner are Active LACP
- *FAST_PERIODIC* if the Partner is running Short Timeouts - also used in initial states while Partner state is unknown (fast_while)
- *SLOW_PERIODIC* if the Partner is running Long Timeouts
- NTT if partner doesn’t know my state

Periodic - Functionality summary (3)

- NO_PERIODIC on initialisation events (reinitialise, create, expired) if Actor is Passive LACP
- FAST_PERIODIC on initialisation events, with fast_while timer running, if Actor is Active LACP
- Reverts to SLOW_PERIODIC on receipt of slow indication from Partner, or on expiry of fast_while if Partner still EXPIRED.

23

Match



24

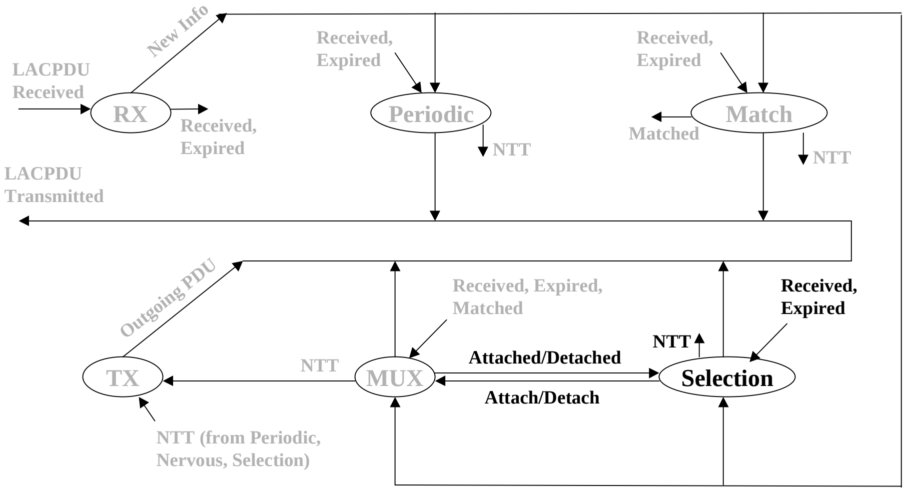
Match Logic

- Matched if:
  - No Partner
  - Matched Individual (Partner believes this link is Individual, or Actor believes this link is Individual & Partner’s view agrees)
  - Matched Aggregate

Match - Functionality summary

- Determines whether participants have both agreed on the protocol information exchanged to the extent that the physical Port can safely be used in an aggregate
- State of match feeds into Mux state machine
- Initial state: No match

Selection



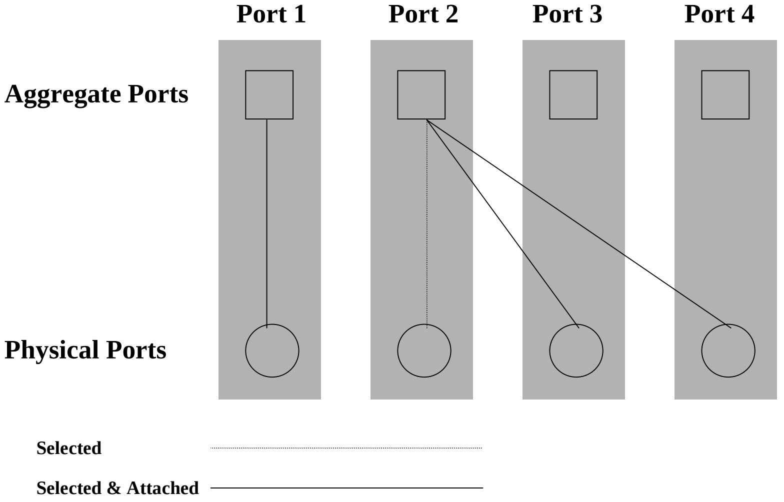
Selection - Assumptions

- No additional MAC addresses (over those allocated per physical MAC) required
- Determinism in allocation of physical Ports to Aggregators
- Result is intuitive to the user
- Compatible with alternative views

Selection - Rules

- Each MAC has a physical Port and an Aggregator
- Aggregation = attachment of a physical Port to an Aggregator (its own, or someone else's)
- Each physical Port is always attached to one Aggregator
- The physical Port of an individual link always attaches to its own Aggregator
- The lowest numbered Aggregator is always used by an aggregate, even if its physical Port is not operational

Selection - Legal Example 1



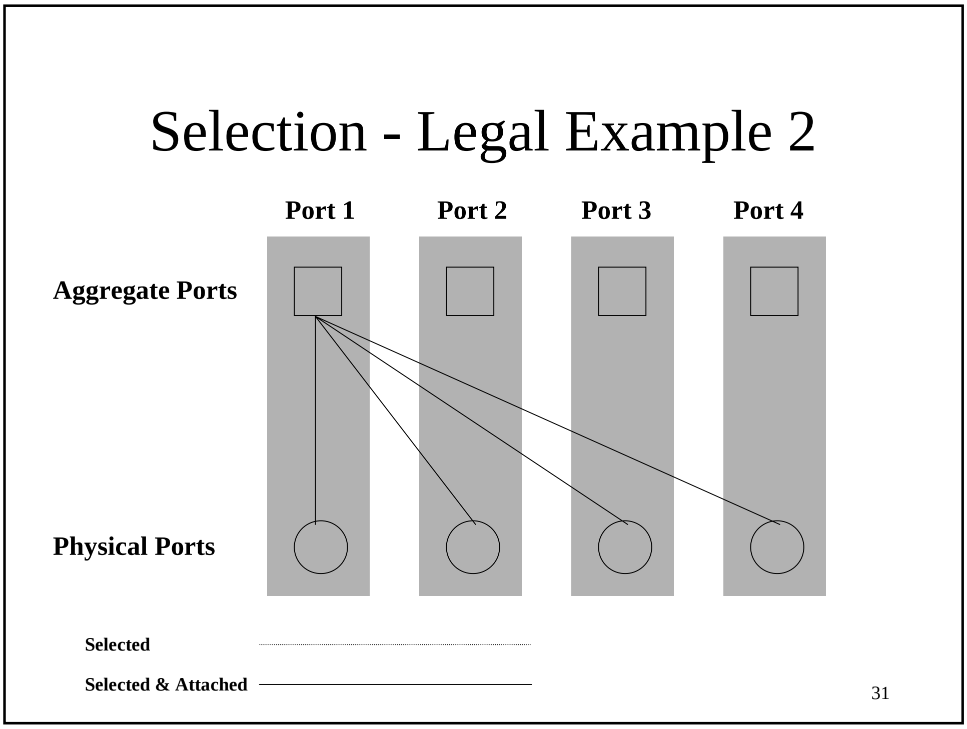
Port 1 **Port 2** **Port 3** **Port 4**

Aggregate Ports

Physical Ports

Selected

Selected & Attached



Selection - Legal Example 2

Port 1 Port 2 Port 3 Port 4

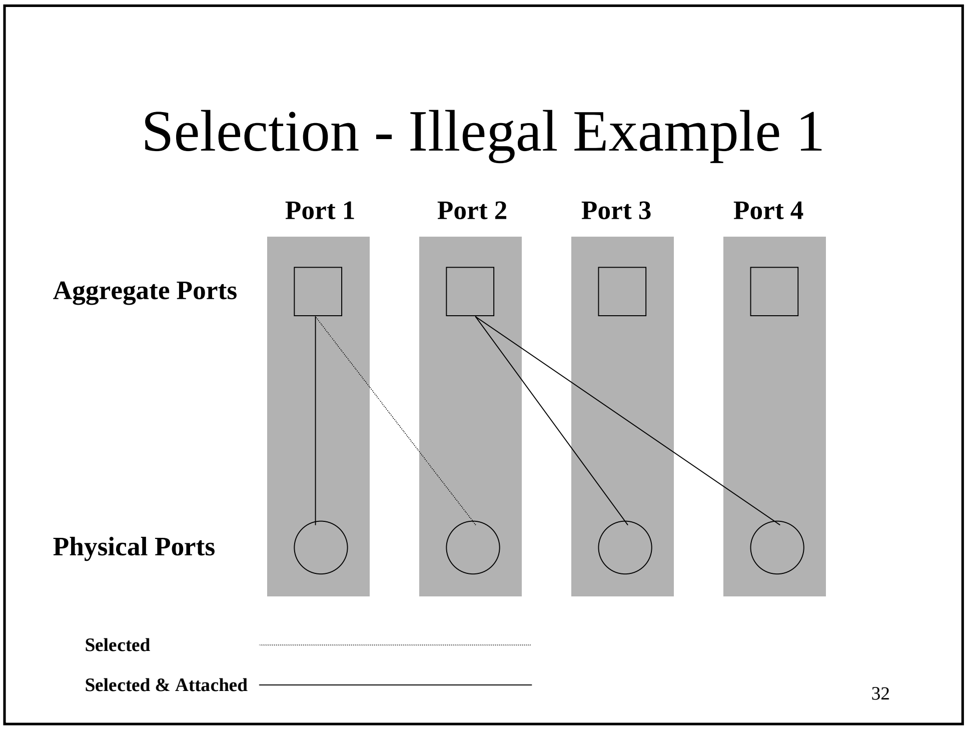
Aggregate Ports

Physical Ports

Selected

Selected & Attached

31



Selection - Illegal Example 1

Port 1 Port 2 Port 3 Port 4

Aggregate Ports

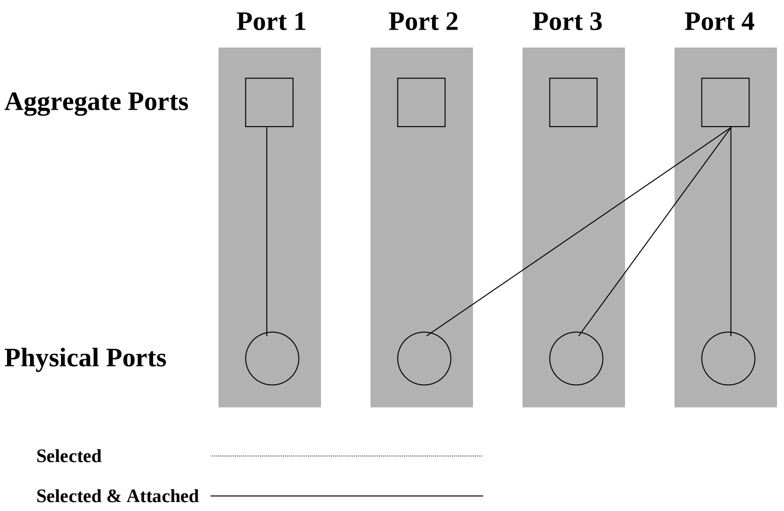
Physical Ports

Selected

Selected & Attached

32

Selection - Illegal Example 2



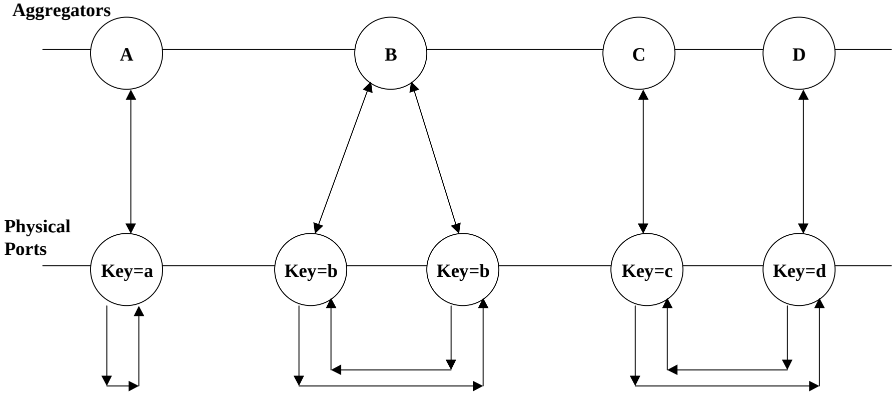
Selection Logic (1)

- Determines
  - – Partner’s System ID and Key
  - – Whether this link is an individual link
  - – Whether the partner has changed (ID or Key)
- Updated on
  - – New information received
  - – Selection Wait time expiry
  - – Management changes to my parameters

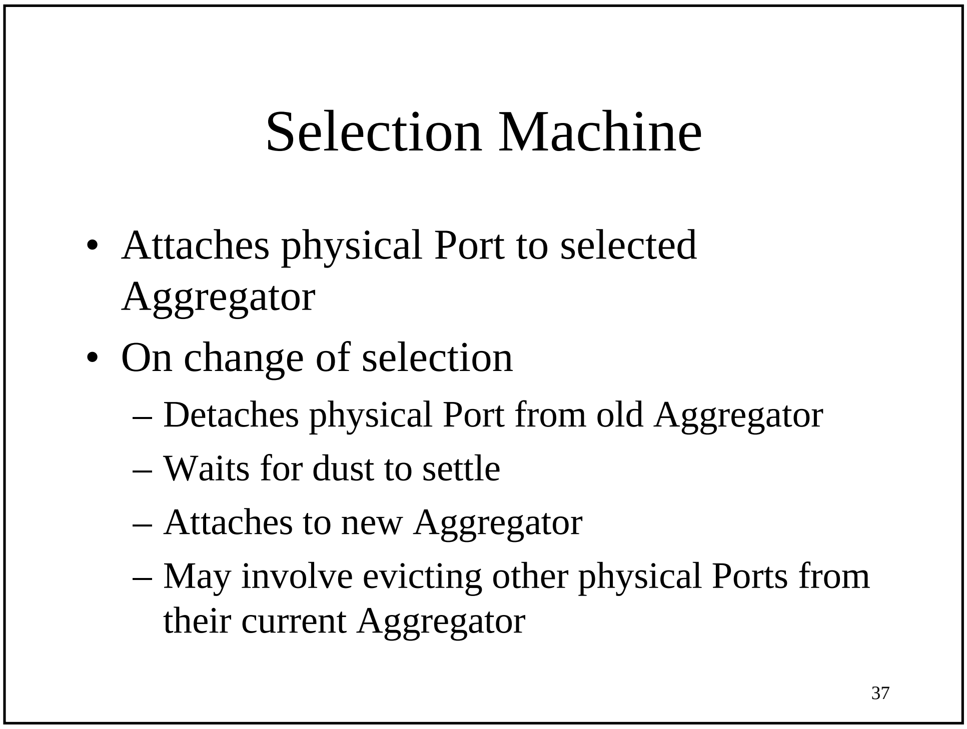
Selection Logic (2)

- Individual if
  - RX machine is expired
  - Actor believes the link to be individual
  - partner believes the link to be individual
  - Port is looped back (Partner ID/Key = Actor ID/Key)
- If Individual, Aggregator selected is own Aggregator
- If not Individual, Aggregator selected is lowest numbered Aggregator with same local/remote system ID & Key

Loopback cases

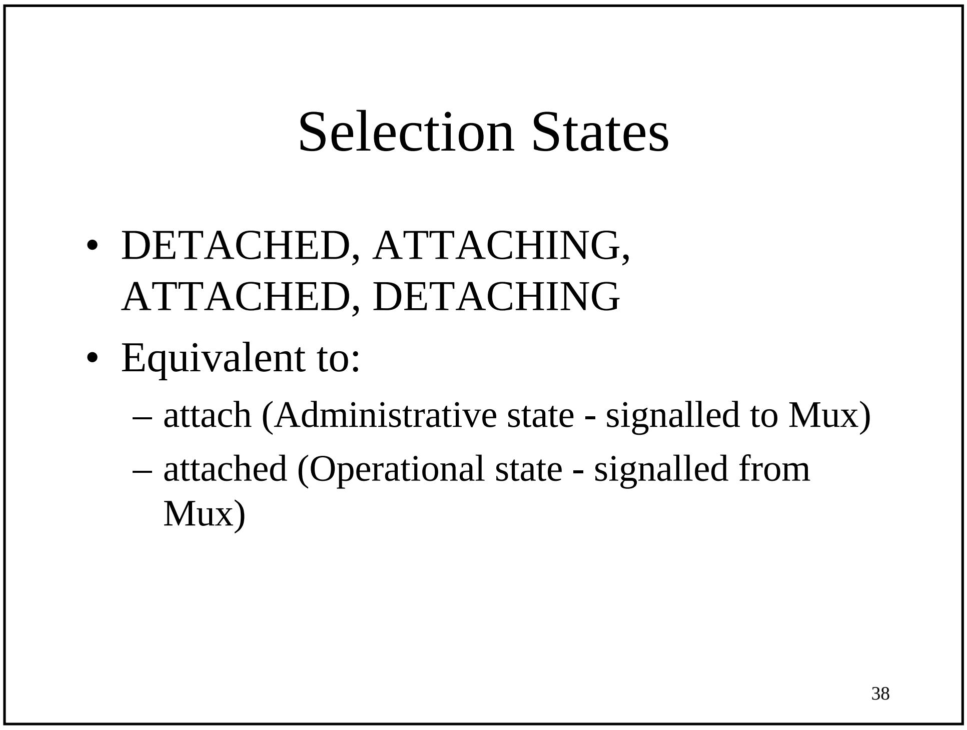


- A & B look the same to upper layers - will confuse/break Bridges
- In B need to treat physical Ports as Individual (=> same as C/D)
- In A may need to disable Physical Port (or Aggregator) to prevent it being used by upper layers
- Aggregators C, D work just fine



Selection Machine

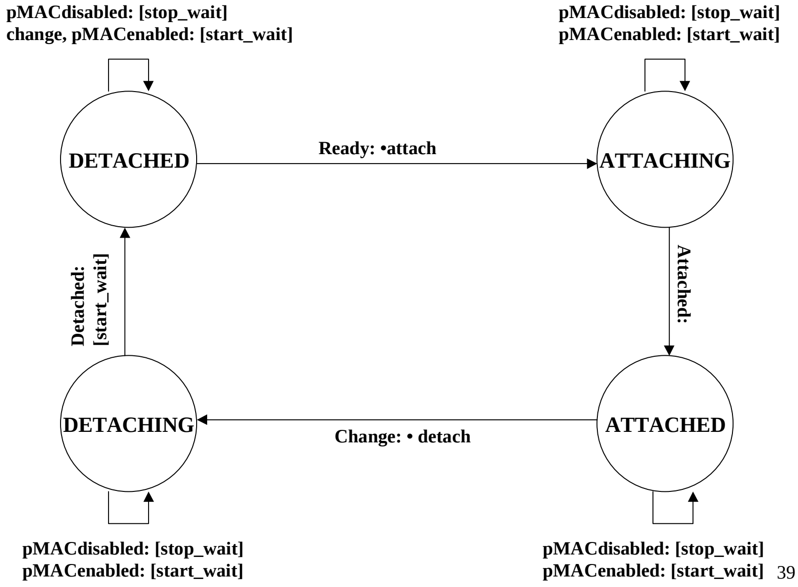
- Attaches physical Port to selected Aggregator
- On change of selection
  - – Detaches physical Port from old Aggregator
  - – Waits for dust to settle
  - – Attaches to new Aggregator
  - – May involve evicting other physical Ports from their current Aggregator



Selection States

- DETACHED, ATTACHING, ATTACHED, DETACHING
- Equivalent to:
  - – attach (Administrative state - signalled to Mux)
  - – attached (Operational state - signalled from Mux)

Selection - State Machine


pMACdisabled: [stop_wait]
change, pMACenabled: [start_wait]
pMACdisabled: [stop_wait]
pMACenabled: [start_wait]
DETACHED
Ready: •attach
ATTACHING
Detached:
[start_wait]
Attached:
DETACHING
Change: • detach
ATTACHED
pMACdisabled: [stop_wait]
pMACenabled: [start_wait]
pMACdisabled: [stop_wait]
pMACenabled: [start_wait]

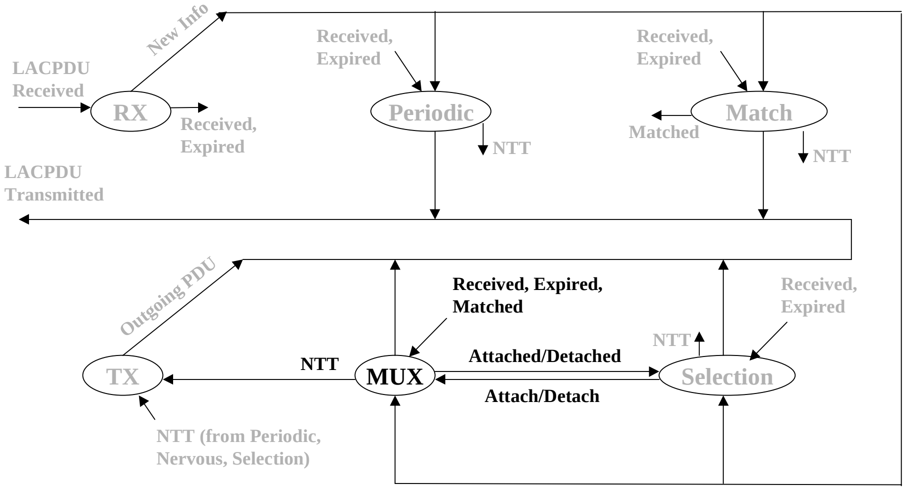
“Hard Wired” selection

- Rationale: Need to allow for devices that can aggregate, but cannot run LACP
- Solution:
  - Allow administrative configuration of “default” value for Partner Key
  - Default value used, with Partner ID of 0, if protocol not received from Partner
  - Default is overridden by any active protocol exchanges

Selection - Functionality summary

- Determines whether the link is in the right aggregate or not
- If not in the right one, removes it
- If not in an aggregate, finds the right one for it to be in and adds it
- Takes account of the need to wait for other links to select the same aggregate
- Can allow for management-specified default configuration

MUX



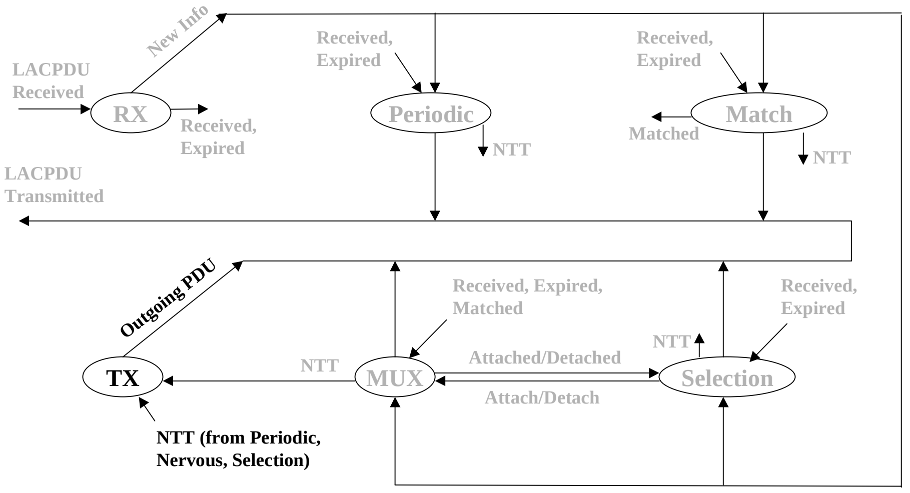
Mux - States and Goals

- States: In Sync, Out of Sync
- Goals
  - Partner or Actor Out of Sync: turn off collector & distributor
  - Actor and Partner In Sync: turn on collector
  - Actor and Partner In Sync, Partner’s Collector is on: turn on distributor
  - Above rules also apply to **coupled** mux h/w
  - If mux h/w is **independent**, and if Partner’s collector is turned off, then turn off distributor

Mux - Functionality summary

- When *in synch*, takes the necessary steps to turn on collector and distributor
- When *out of synch*, takes the necessary steps to turn off collector and distributor
- Signals *attached, detached* when it is done
- Initial state: collector/distributor off, out of sync

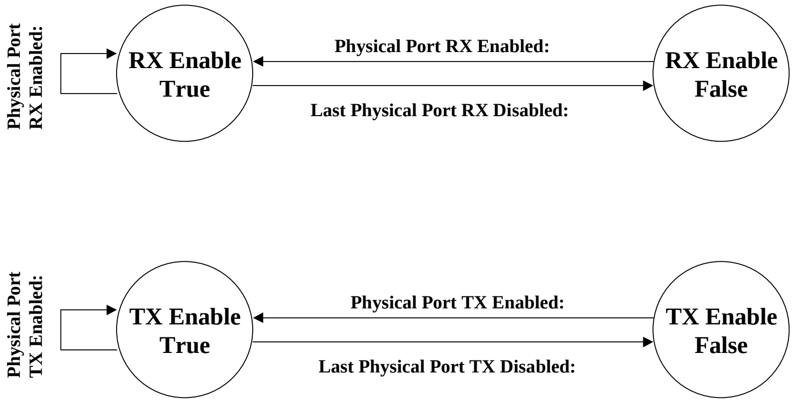
TX



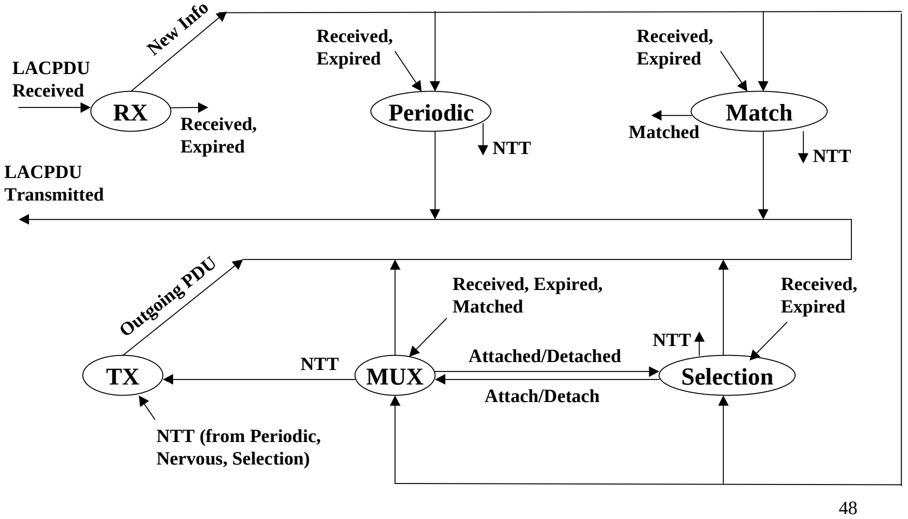
TX - Functionality summary

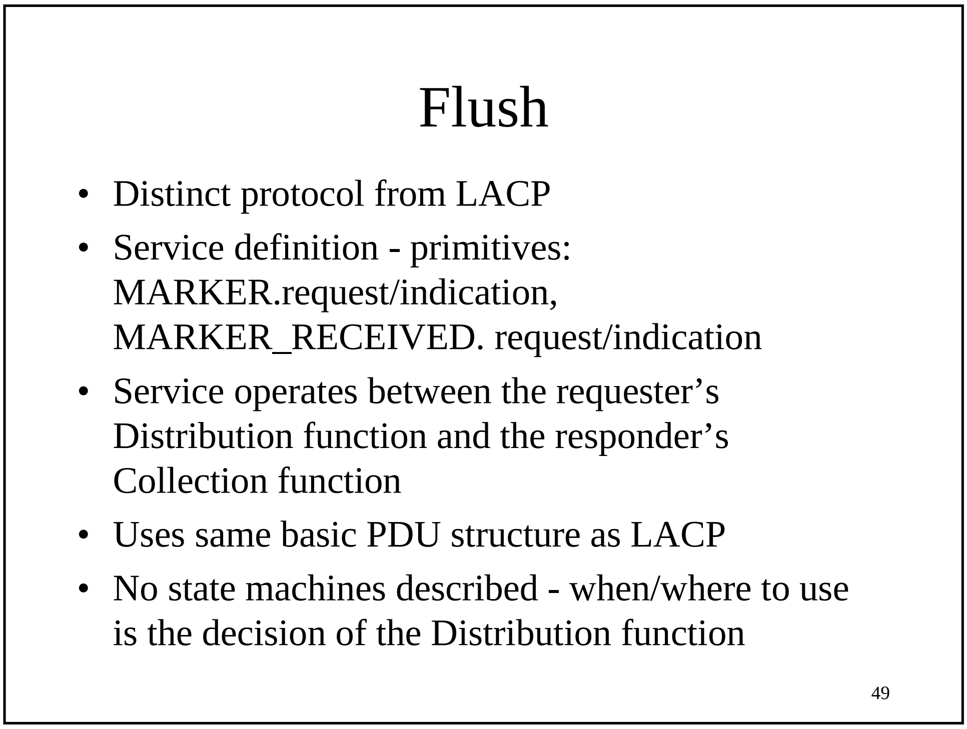
- Causes LACPDUs to be generated if NTT
- Collects & assembles the required information from the other machines
- Limits LACPDU transmission rate to no more than 3 in any Fast_Transmission_Ticks interval

Aggregate Port TX and RX States


Physical Port RX Enabled:
RX Enable True
Physical Port RX Enabled:
Last Physical Port RX Disabled:
RX Enable False
Physical Port TX Enabled:
TX Enable True
Physical Port TX Enabled:
Last Physical Port TX Disabled:
TX Enable False

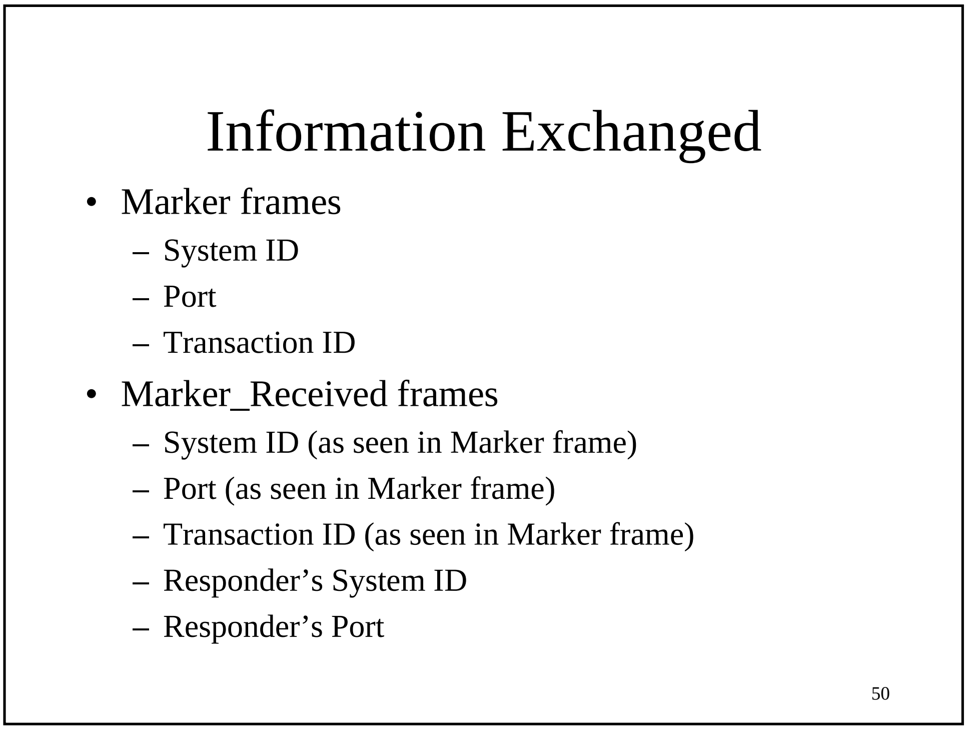
The Big Picture





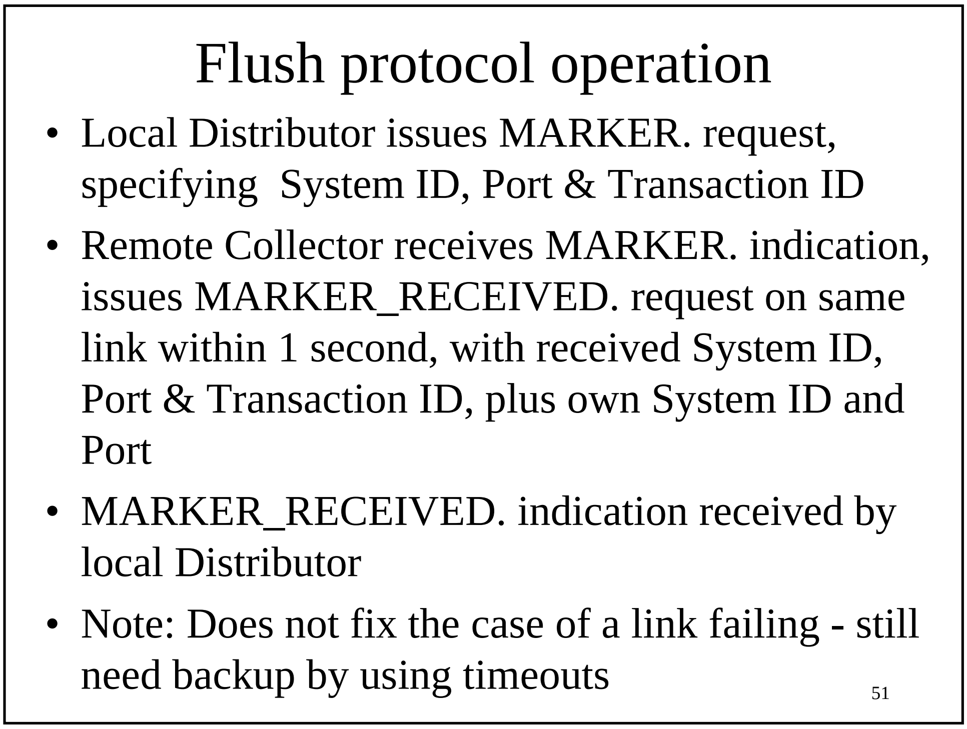
Flush

- Distinct protocol from LACP
- Service definition - primitives: MARKER.request/indication, MARKER_RECEIVED. request/indication
- Service operates between the requester’s Distribution function and the responder’s Collection function
- Uses same basic PDU structure as LACP
- No state machines described - when/where to use is the decision of the Distribution function



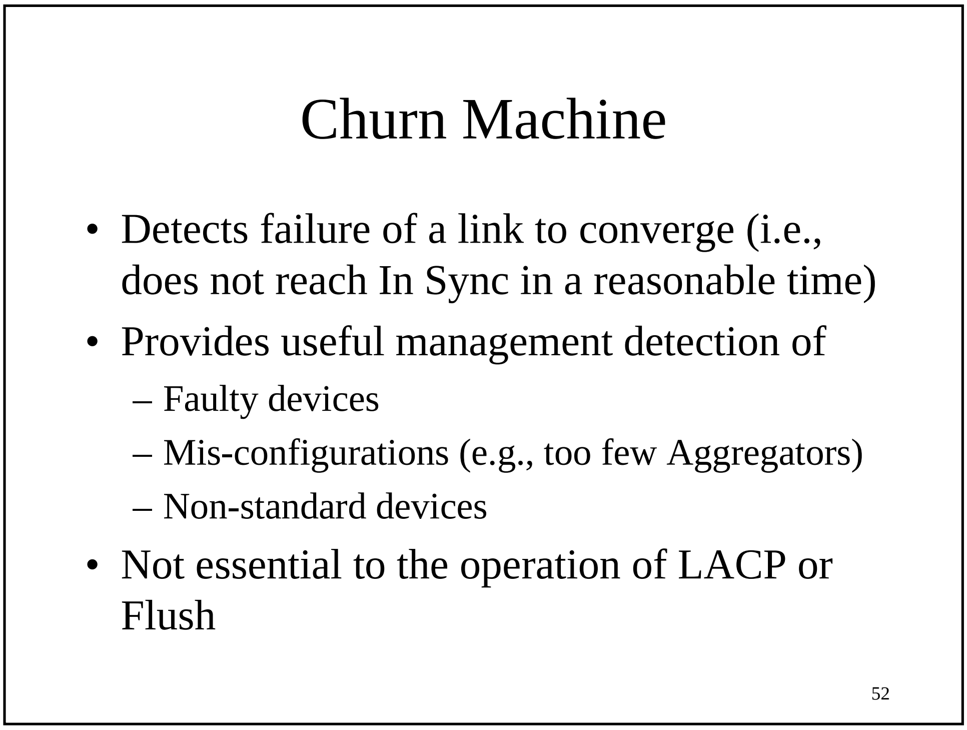
Information Exchanged

- Marker frames
  - System ID
  - Port
  - Transaction ID
- Marker_Received frames
  - System ID (as seen in Marker frame)
  - Port (as seen in Marker frame)
  - Transaction ID (as seen in Marker frame)
  - Responder’s System ID
  - Responder’s Port



Flush protocol operation

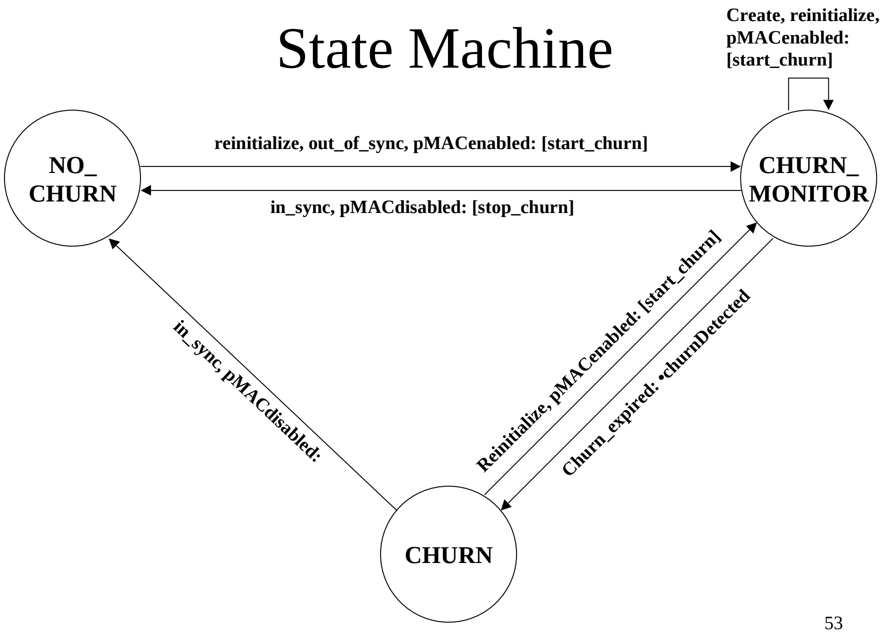
- Local Distributor issues MARKER. request, specifying System ID, Port & Transaction ID
- Remote Collector receives MARKER. indication, issues MARKER_RECEIVED. request on same link within 1 second, with received System ID, Port & Transaction ID, plus own System ID and Port
- MARKER_RECEIVED. indication received by local Distributor
- Note: Does not fix the case of a link failing - still need backup by using timeouts



Churn Machine

- Detects failure of a link to converge (i.e., does not reach In Sync in a reasonable time)
- Provides useful management detection of
  - Faulty devices
  - Mis-configurations (e.g., too few Aggregators)
  - Non-standard devices
- Not essential to the operation of LACP or Flush

Churn - State Machine


Create, reinitialize, pMACenabled: [start_churn]
NO_ CHURN
CHURN_ MONITOR
CHURN
reinitialize, out_of_sync, pMACenabled: [start_churn]
in_sync, pMACdisabled: [stop_churn]
in_sync, pMACdisabled:
Reinitialize, pMACenabled: [start_churn]
Churn_expired: •churnDetected

Summary

- Covers (majority of) LACP functionality previously described/presented
- Fully describes the process of reaching agreement & the actions taken to join & leave aggregations
- Simplifies/clarifies operation of periodic transmission control
- Flush protocol fully described
- Churn detection machine adds useful failure mode detection