AX.25 Link Multiplexor State Machine

Eric L. Scace K3NA
10701 Five Forks Road
Frederick MD 21701
USA

0. Summary

This paper is part of a series of papers which provide extended finite state machine representations for AX.25 and related protocols. The state machines are depicted using state description language (SDL) graphic conventions from the 2.100 series of Recommendations developed by the International Telegraph and Telephone Consultative Committee (CCITT) of the International Telecommunications Union (ITU). An extended finite state machine representation of a communications protocol such as AX.25 avoids the ambiguities associated with prose descriptions. These descriptions also compel1 the protocol designer to confront many of the error scenarios which arise on a communications path, and simplify the implementor's task of producing correct solutions which will interwork with solutions created by others.

This particular paper describes an extended finite state machine which supports multiple simultaneous AX.25 links. "Link" here embraces both AX.25 connections (established between two stations with the SABM command) and connectionless operation between two stations (using UI frames exclusively). The main responsibilities of the link multiplexor SDL machine are to insure that each link has a fair and equal opportunity to access the radio channel, and to handle incoming frames which require digipeating.

1. status of Proposal

The link multiplexor SDL description here is a draft. The ARRL Digital Committee intends to include this machine as an Annex of the upcoming publication of AX.25 Revision 2.1.

The present AX.25 Revision 2.0 includes very little information about the use of simultaneous links. It has been my personal observation that a number of AX.25 implementations which permit simultaneous links fail to operate in a graceful manner. For example, some implementations transmit every outstanding frame, regardless to whom it is destined, in a single (and sometimes quite lengthy) burst... and then sit back and expect that twenty or more remote stations will successfully acknowledge receipt before twenty-plus concurrent retry timers expire. Even under optimum conditions extensive polling and retries result. The proposal for a standard link multiplexor is intended to provide helpful guidance which will lead to more effective implementations of stations which use simultaneous connections.

The following material is still in a draft state. You are invited to review and comment on this material. Comments are desired so that the final publication is as useful as possible to its readers.

2. Features of the Link Multiplexor SDL Machine

The link multiplexor SDL machine includes the following features:

a) preferential treatment for frames to be digipeated.
b) discard of incoming frames containing FCS errors.
c) round-robin rotation for access to the radio channel between all links with frames awaiting transmission.

[Note -- An equivalent link multiplexor SDL machine with priority assignments for each link has also been developed, but is not included here. If there is sufficient interest expressed, it will be made available for inclusion in AX.25 Revision 2.1.]

3. Location in Overall Model

This SDL machine resides within the data link layer of the Open Systems Interconnection reference model. The link multiplexor SDL machine interacts with data link SDL machines above it and a single physical layer SDL machine below.

The physical layer SDL machine may be either the simplex physical SDL machine or the simple full duplex physical SDL machine (as described in companion papers), or any other physical layer SDL machine which is prepared to accept and generate the corresponding primitives.

The link multiplexor SDL machine works with multiple data link SDL machines above it. A data link SDL machine exists for each ongoing communication (connection-oriented or connectionless or both) with a remote station. The individual data link SDL machines are distinguished by hypothetical "identifiers". In formal protocol standardization, these hypothetical identifiers are known as service access points; however, in practical terms for AX.25 the identifier is the **callsign** of the remote station (as found in the address field of the AX.25 **frame).**

3.1 Interaction with the Data Link

The data link SDL machine directs the operation of the link multiplexor SDL machine through the primitives described below. **"LM"** in some primitive names stands for "link multiplexor".

LM Seize Request -- This primitive requests the link **multiplexor** SDL machine to arrange for transmission at the next available opportunity. The data link SDL machine uses this primitive when an acknowledgement must be ma&, but the exact frame in which the **ackowledgement** will be sent will be chosen when the actual time for transmission arrives. The link multiplexor SDL machine uses the **LM Seize** Confirm to indicate that the transmission opportunity has arrived. After the data link SDL machine has provided the acknowledgement, the data link SDL machine gives permission to stop transmission with the **LM Release Request** primitive.

Frame -- This primitive from the data link SDL machine provides an AX.25 frame of any type (UI, SABM, I, etc.) which is to be transmitted. An unlimited number of frames may be provided. The link multiplexor SDL machine accumulates the frames in a **first-in** first-out queue until it is time to transmit them.

During reception, the link multiplexor SDL machine delivers an **incoming** AX.25 **frame to the addressed data link SDL machine in a Frame primitive.**

3.2 Interface to the Physical Layer

The link multiplexor SDL machine works with **a** specific physical **layer SDL machine. The** exact type of physical Iayer **SDL machine** used is dependent on the characteristics of the radio channel. **Two** companion papers described physical layer **SDL machines** suitable for simplex channels typical of most 2 meter amateur packet frequencies, and for simple (used only by two stations) full duplex channels which would be part of a backbone networking trunk circuit. Importantly, the variation in operating characteristics of radio channels is kept hidden from the link multiplexor SDL machine. The link **multplexor** SDL machine uses the same primitives to communicate with the physical layer SDL machine, regardless of which type it is.

PH Seize **Request** is used by the link multiplexor SDL machine before each transmission to request access to the radio channel. When access has been obtained (i.e., the transmitter is operating, any **intervening** repeater has had an opportunity to be activated, the remote station's received has had an **opportunity** to become synchronized, and the channel is considered ready to send traffic), the link multiplexor SDL machine is notified by a **PH Seize Confirm primitive.**

At **this** point **the link multiplexor** SDL machine **delivers** each **frame to be** sent **in a Normal Frame primitive to the** physical layer **SDL machine.** When **all** frames ***which have been awaiting transmission for a given link*** have been submitted for transmission, the link multiplexor SDL machine concludes with a **PH Release Request** primitive. The intention here is that **a single** transmission **will** contain frames for only one remote station. The PH Release Request primitive permits the physical layer **SDL** machine to release the channel for use by others, for digipeating, and for receipt of acknowledgements in a contention environment (such as shared **simplex** channels).

The physical layer SDL machine provides incoming frames to the link multiplexor SDL machine via **Frame** primitives. The link multiplexor SDL machine checks each incoming frame for FCS errors. Correctly-received frames are checked to see if digipeating by the station has been requested and if the digipeat function is enabled (a user specified parameter); if so, the frame is resubmitted to the physical layer SDL machine **in** a **Digipeat Frame primitive.** (PH Seize Request and PH Release Request are not used for **digipeat** operation.) Correctly-received **frames** addressed to this station are delivered to the indicated higher-layer data link SDL machine (described earlier in. **§** 3.1).

The physical layer SDL machine **also** provides the **PH Busy Indication,** which is used by the link multiplexor SDL machine **to** suspend **all AX.25 data link timers.** Timers resume ticking when **the PH Quiet Indication is** received. The suspension of timers overcomes **a** problem noted in some implementations on busy channels. This problem occurs when frames are transmitted and **a** response is expected. If the channel is busy, it is possible for the retry timers (AX.25 timer **T1)** to expire before the remote station had an opportunity to send any acknowledgement. This premature expiration causes needless retries and polling, further cluttering an already busy frequency.

4. Internal Operation of the Machine

The internal states, queues, and flags are summarized on the first page of the SDL diagram.

All queues are **first-in first-out** queues. Three queues are utilized, in conjunction with two **flags,** to implement round-robin rotation amongst the various data link SDL machines.

The Awaiting Queue contains all primitives received from data link SDL machines which have not yet had an opportunity to transmit.

When a primitive pops off the Awaiting Queue, it and all other primitives from that same data link SDL machine are placed in order on the Current Queue. The identity of this data link SDL machine is maintained in the Current DL flag. The link multiplexor SDL machine then proceeds to obtain a transmission opportunity for that data link SDL machine. Any further primitives received from that particular data link SDL machine are added to the Current Queue. When the transmission opportunity arrives, everything in the Current Queue is conveyed to the physical layer SDL machine for transmission. (In the event of an overly large amount of information to be sent, the physical layer SDL machine makes whatever breaks in transmission are appropriate for reasonable channel sharing. This is done within the physical layer SDL machine and hidden from the higher layers.)

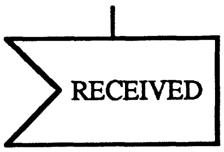
Once everything has been sent for the current data link SDL machine, its identity is moved to the Served List. Any subsequent primitives from this data link SDL machine are added to the ServedQueue.

The link multiplexor SDL machine then goes back to the Awaiting Queue to pop off the next primitive, and thereby identify which data link SDL machine has the next transmission opportunity. If the Awaiting Queue is empty, then the link multiplexor SDL machine concludes that all data link SDL machines which had frames to be sent have now been served. The queue system is reset by converting the Served Queue into a new Awaiting Queue, and by purging all identifiers from the served List.

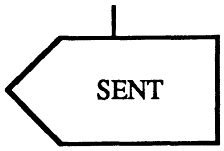
LINK MULTIPLEXOR

Summary of Primitives, States, Flags, Errors, and Timers

LM Primitives

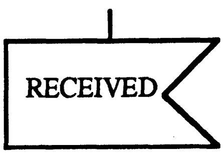


LM Seize Request
LM Release Request
frame (any type)

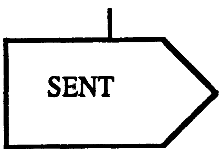


LM Seize Confirm
frame (any type)

PH Primitives



PH Seize Confirm
PH Quiet Indication
PH Busy Indication
frame



PH Seize Request
PH Release Request
digipeat frame
normal **frame**

States

O-Idle
1 -- Seize Pending
2 -- Seized

Error Codes

No error codes used.

Queues

Awaiting Queue -- queue of primitives received from data link machines which are not presently using the transmitter.
Current Queue -- queue of primitives received from the data link machine which is presently using the transmitter.
Served Queue -- queue of primitives received from data link machines which already have used the transmitter.
Note -- After all data link machines have had an opportunity to be served, then the Served Queue is converted to the Awaiting Queue.

Flags

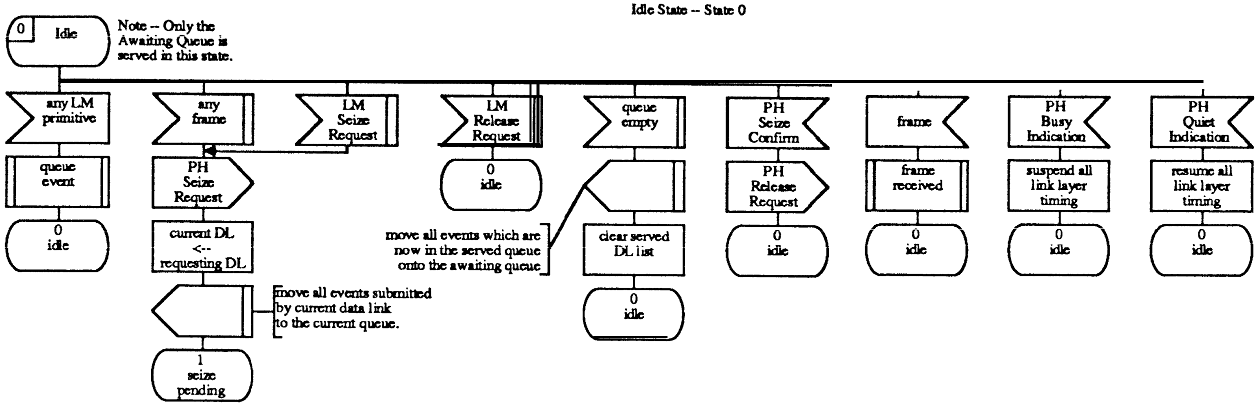
Current DL -- Identifies the data link machine currently using the transmitter.
Served List -- Identifies the data link machines which have already used the transmitter. This list is cleared when all data link machines with **frames** to send have been served.

Timers

No timers used.

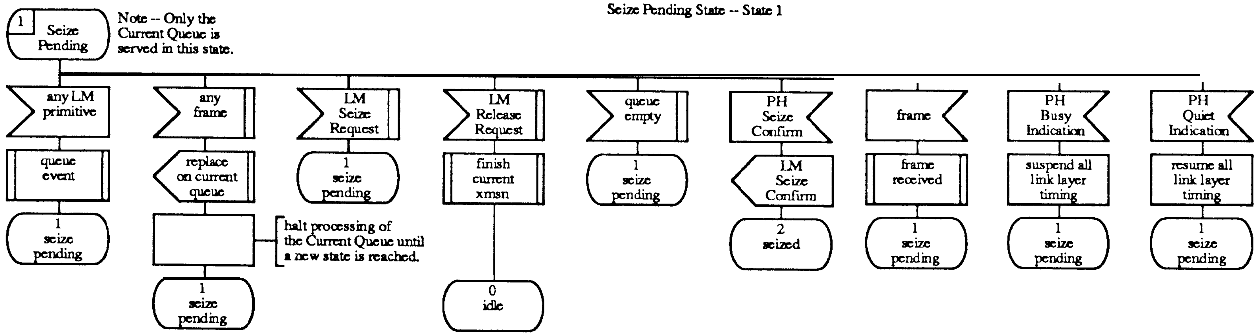
Link Multiplexor

Idle State -- State 0



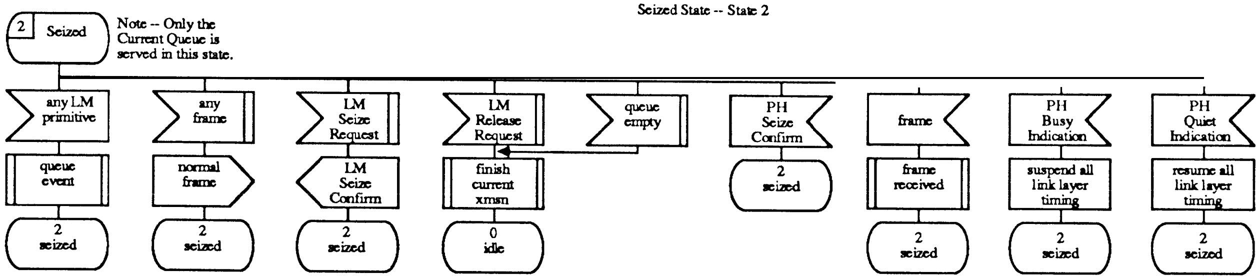
Link Multiplexor

Seize Pending State -- State 1



Link Multiplexor

Seized State -- State 2



Link Multiplexor

Subroutines