SLACP Overview

- SLACP - a logical link layer protocol
- SLACP - to improve the performance of TCP in satellite links
- Combines ARQ - FEC to reduce residual packet loss rate
- Uses selective repeat sliding window mechanism for data transfer
- Provides partially reliable in-order delivery of frames
- Flow control between IP – SLACP - MAC

SLACP Environment

SDL Support for SLACP Design

- SLACP was modelled in SDL and validated using the Telelogic Tau 4.4 SDT
- SDL Model and Validation guided the development of SLACP using a Validation Oriented Design Approach
- SDL model served as the basis of implementation of SLACP in Linux

SLACP Design: Validation-Oriented Design Approach

- Initial Abstract Model (SDL model) with minimum signals and processes
- Validate the internal consistency (using Telelogic SDT) and functionality of the abstract model
- Incrementally add features to the model and validate at each step
- Iterate till the design is complete
- Incremental approach helps in controlling the complexity of the protocol and in evolving a design that could be validated

Validation Oriented Design Approach
SLACP - SDL Model

SDL allows to define the system model in a top down manner
- System Model
  - Blocks
- Processes

SLACP - SDL Model consists of
- IP Layer
- SLACP Sender
- MAC Layer
- SLACP Receiver

Only the essential elements to be validated in the SDL model are retained in the SDL model.
Inessential aspects are stripped off:
- Only one SLACP channel is used
- No QoS and scheduling
- No Segmentation and reassembly
- No FEC coding/decoding algorithm
- No processing of accepted frames at the receiver

SDL model incorporates SLACP interface to IP and MAC layers.
SLACP protocol engine consists of both sender and receiver.
In SDL model we separate sender and receiver for ease of implementation.
No processing of received frames by higher layers in the SDL model.

IP Layer

Packet Generator Process
- functionalities associated with packet generation

Signaling Process
- channel establishment, channel reset and channel disconnection
- packet delivery status
- packet loss notifications
In the actual implementation there are N senders for N logical channels. As senders are independent, we model just one sender. Sender block consists of:
- Sender process
- Sender control process

Sender process
- The entity carrying out the processing of signals for ordinary data channel

Sender control
- The entity carrying out the processing of signals for the control channel

Sender Process
- Main Functions
  - Flow control between IP and SLACP
  - Flow control between SLACP and MAC
  - Data transmission, ACK reception

Sender Control
- Main Functions
  - Retransmission of data frames
  - FEC encoding & decoding
  - Instead of using FEC coding to compute redundancy frames, actual frame in the FEC block is duplicated as redundancy frame

MAC Layer
- Delivery of frames between slacp sender and receiver
- Flow control information to both sender and receiver
- Modeling delay and packet losses in the link
MAC Process

- Modelling delay
  - Mac process queues all the frames it receives and sends when a link timer expires
  - Timeout of the link timer represents the delay in the wireless link
- Modelling packet loss
  - Drops packets in a random way to model packet losses
  - We simulated the packet loss by dropping every mth packet

SLACP Receiver

- Receiver process
  - Entity carrying out the processing of signals for ordinary data channel
- Receiver control
  - Entity carrying out the processing of signals for the control channel

Receiver Process

- Main Functions
  - Flow control between SLACP and MAC
  - Data reception
  - ACK transmission

Receiver Control

- Main Functions
  - Processing of retransmitted data
  - FEC encoding and decoding
  - A duplicate frame present among the redundancy frames, simulates fec recovery
Data structures used in SDL model

- SLACP Frames (Data, ACK)
- Queues
- Scoreboard
- Timers
- Choose small sizes for the frames, queues to reduce the complexity of the validation

MAC Queue

newtype t_Macframe choice
  a t_slacp_frame;
  b t_ACKframe;
  c FEC_data_frame;
  .
  .
  k integer;
end newtype;
newtype t_Macqueue array(integer,t_Macframe)
end newtype;

Scoreboard

newtype fec_scoreboard array(integer,fec_sb_entry)
end newtype;
newtype fec_sb_entry struct
  block_no integer;
  seq_no integer;
  retransmitted boolean;
  frame t_slacp_frame;
end newtype;

Timer

Timer Link_Timer;
dcl delay duration := 3;
set(now+delay, Link_Timer)
MAC Process

Levels of Modelling

1. Opening and closing of a channel
2. Simplest data transfer
3. Data transfer with flow control
4. Data transfer with error control
5. Full-fledged SLACP Model

Each level of modelling builds on the previous level.
Level 1: Opening and Closing of a channel

- The functionalities for opening and closing a channel
- Involve all the protocol blocks and processes
- Help to establish the basic signalling between the blocks and processes
- Timer is used to set the duration of the connection

Level 2: Simplest Data Transfer

- Transfer of a single packet
- IP layer opens the channel and sends a single packet to the SLACP sender
- Sender forwards the frame to MAC layer
- SLACP receiver receives the frame
- Closes the channel
- Extends the model with functionalities for generating the packet, framing, sending and receiving data as well as ACK frames

Level 3: Flow Control

- Flow control between IP and SLACP and between SLACP and MAC
- Adding functionalities
  - for queuing frames at the SLACP sender and MAC buffers
  - for flow control mechanisms based on buffer availability
- This model is tested by setting the size of SLACP buffer and MAC buffer to small values

Level 4: Error Control

- Incorporates FEC/ARQ error control schemes
- Includes SACK mechanism
- Adds buffering and timing features to handle FEC encoding
- Duplication of frames for FEC encoding
- Error recovery scheme is tested by sending several data frames of which only the first frame is lost

Level 5: Full-fledged SLACP Model

- Data Transfer with error control and flow control
- Loss of several frames in the MAC layer
- Protocol recovery after a channel reset
- At this level SLACP model has all the functionalities of the protocol

Validation Scenarios

1. Opening/closing a channel
2. Sending and receiving a frame without frame loss
3. Sending and receiving frames with flow control
4. Sending and receiving frames with frame loss
5. Sending and receiving frames with frame loss and flow control
6. Resetting a channel
SDT Support for Validation

- Validation reports
  - Error situations found during exploration
- Execution and error traces in SDL graphs and MSC (Message Sequence Charts).
  - This helped to eliminate many design errors
- Symbol Coverage
  - Percentage of coverage of SDL symbols

Validation Results

- SDT validation supported the design of SLACP
- Justified the inclusion of new signals such as nothing-to-send, rxmtpt-loss
- Validation provided confidence in the working of the protocol
- SDL Model of SLACP is about 100 pages and took 3 man months to develop

SLACP-SDL model vs SLACP Implementation

- SDL model
- Explicit flow control between IP and SLACP
- FEC coding/decoding not modeled
- Single SLACP channel
- SACK block is a structure
- Coverage 94.99%
  - Uncovered symbols are either invalid end states or conditions that do not occur

- Implementation in C for Linux
- Implicit flow control, SLACP protocol engine accepts a packet from IP underbelly interface only when buffer is available in SLACP
- FEC coding/decoding is implemented
- Many SLACP channels
- SACK block is bitmap
- Many more invalid end states and exceptional conditions

Conclusions

- SDL modeling and validation enabled the design of SLACP
- Validation oriented design approach
  - to control the complexity of the protocol design
  - modularity of the design
- This methodology can be used in general for protocol design.