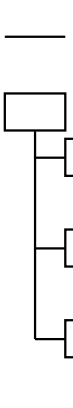
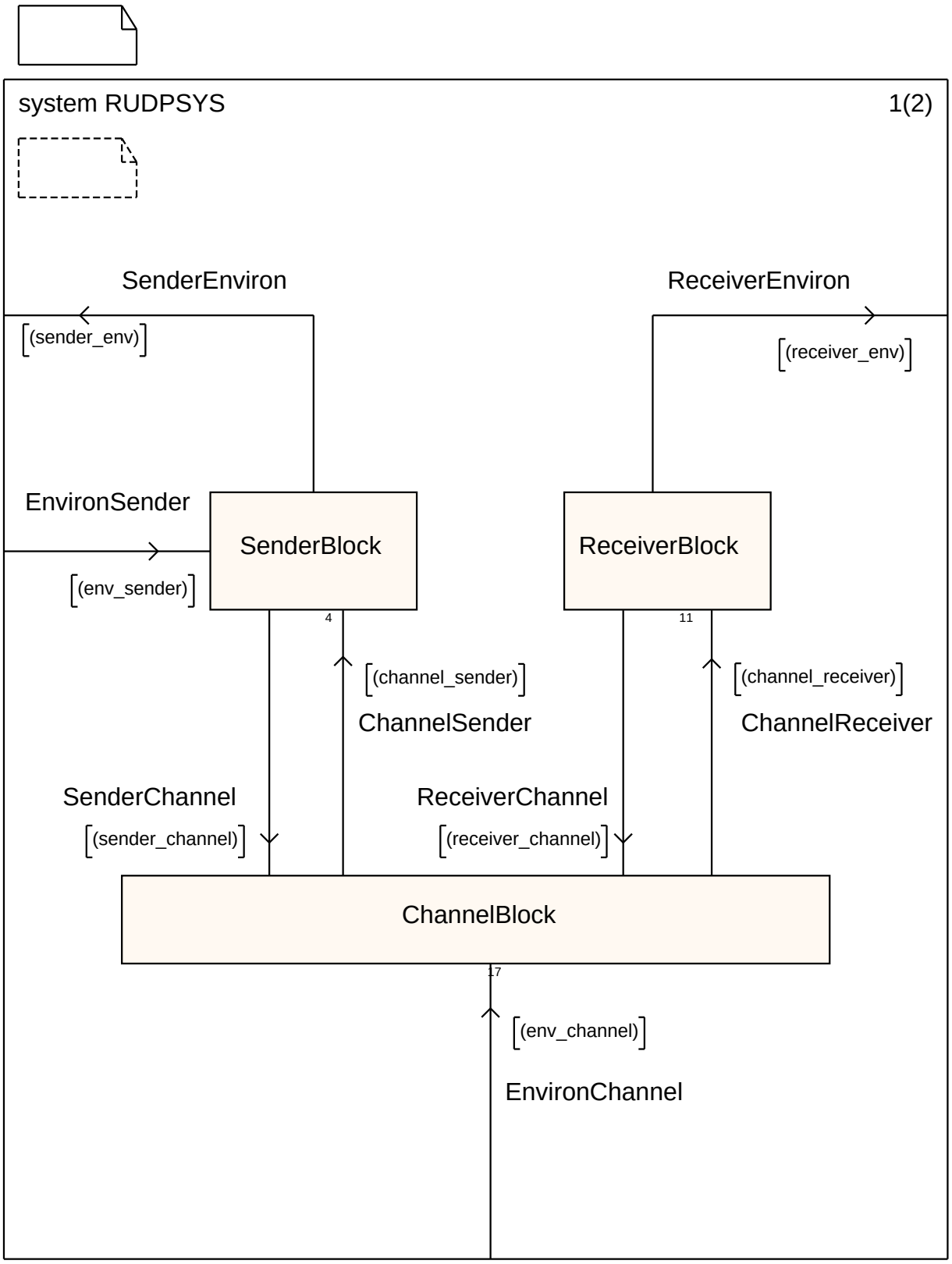
Contents

rw /home/fs−0/2/yding/SDL2008/rudp.sdt

rw /home/fs−0/2/yding/SDL2008/




system RUDPSYS
1(2)
SenderEnviron
ReceiverEnviron
(sender_env)
(receiver_env)
EnvironSender
SenderBlock
ReceiverBlock
(env_sender)
4
11
(channel_sender)
ChannelSender
(channel_receiver)
ChannelReceiver
SenderChannel
(sender_channel)
ReceiverChannel
(receiver_channel)
ChannelBlock
17
(env_channel)
EnvironChannel

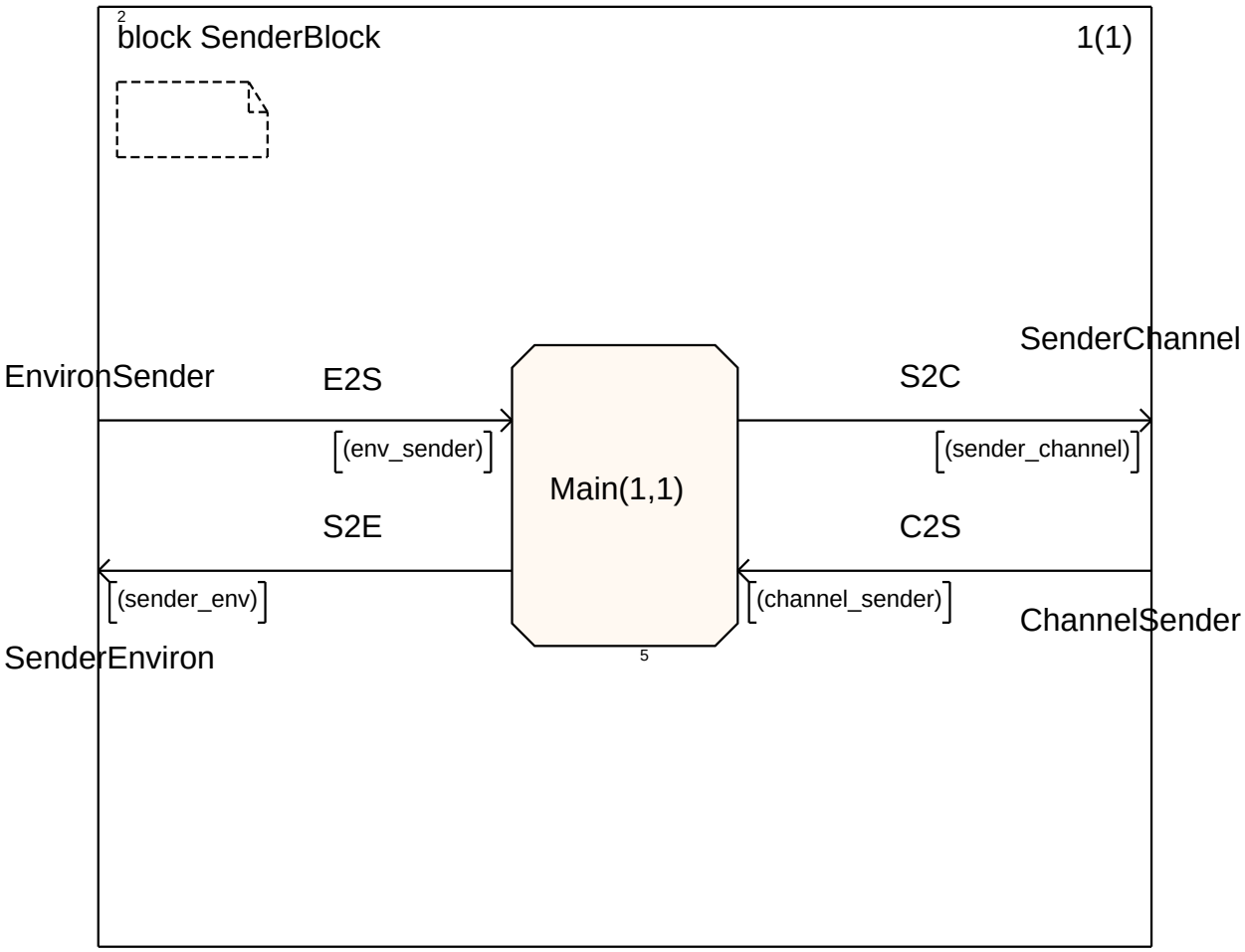
system RUDPSYS 2(2)

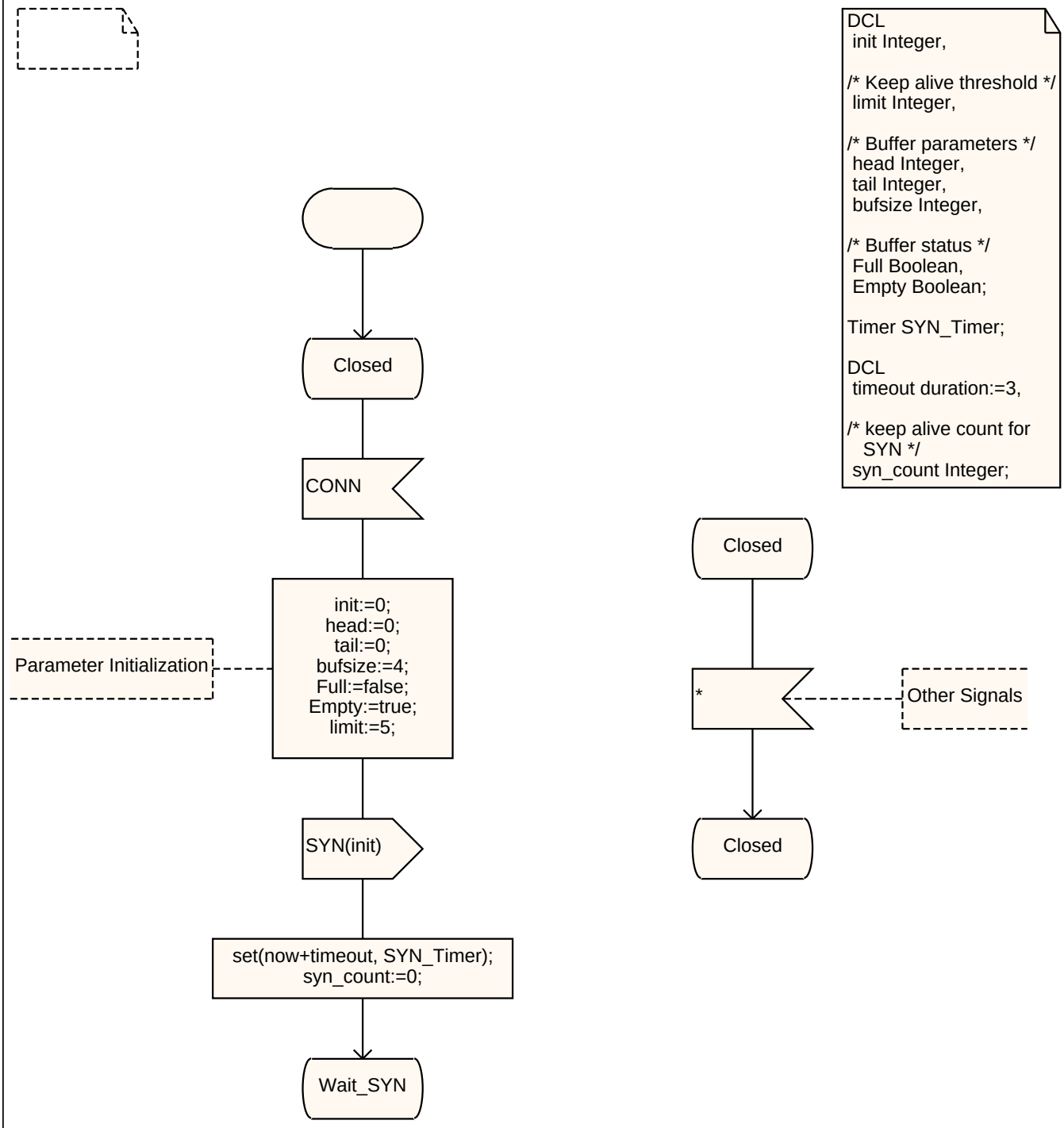
```
/*Packet structure definition*/

NEWTYPE Packet STRUCT
 seq Integer;
 payload Integer;
ENDNEWTYPE;
```

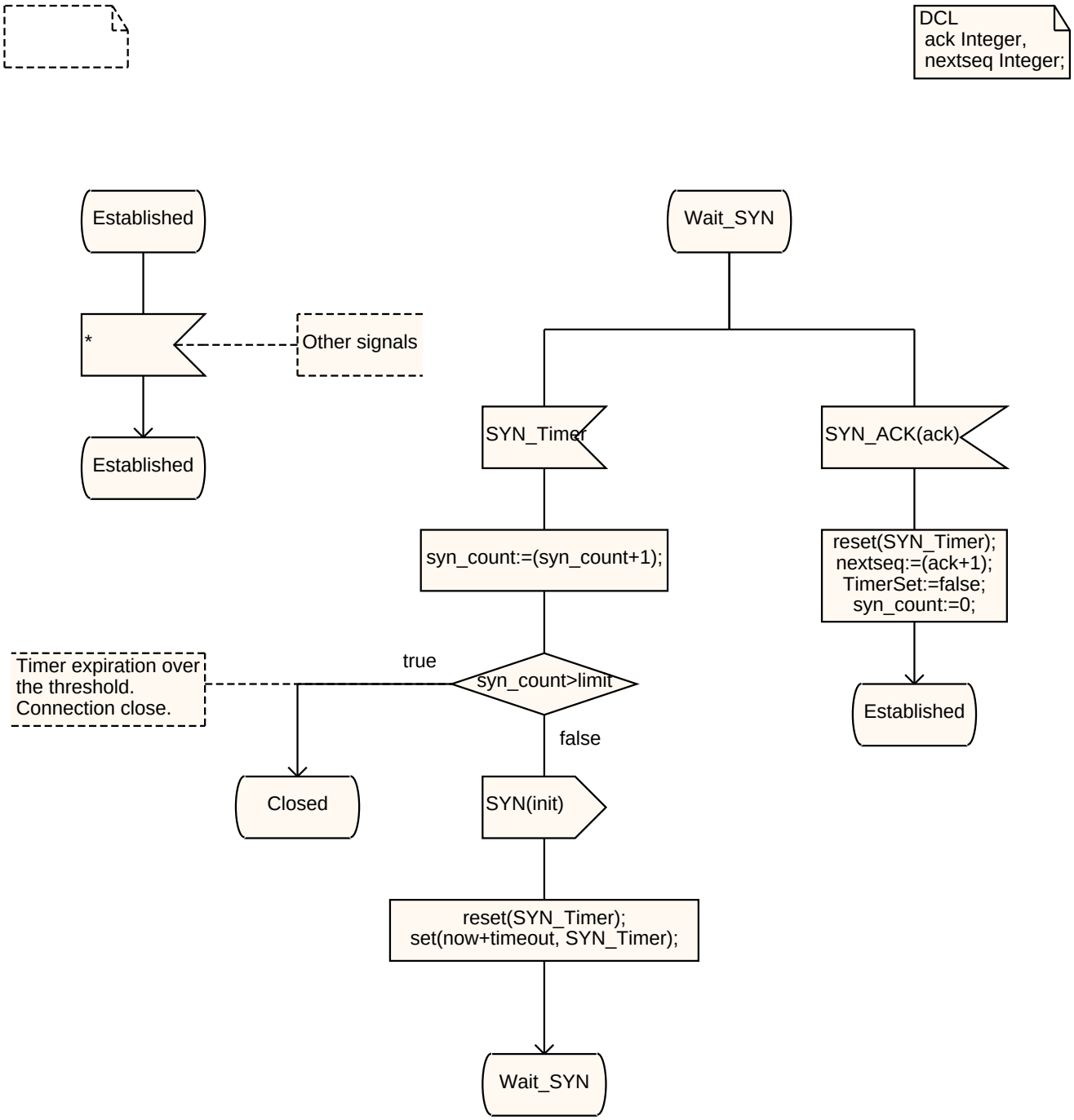
```
SIGNAL

/* Application Signals */
CONN, CLOSE, ERR(Integer), SEND(Integer), RECEIVED(Integer),

/* Control Signals */
Lose, Duplicate,

/* Communication Signals */
DATA(Packet), DATA_ACK(Integer), SYN(Integer), SYN_ACK(Integer),
FIN, FIN_ACK;

/* Signal list */
SIGNALLIST sender_env = ERR;
SIGNALLIST env_sender= CONN, SEND, CLOSE;
SIGNALLIST receiver_env = RECEIVED;

SIGNALLIST env_channel = Lose, Duplicate;

SIGNALLIST channel_sender = DATA_ACK, SYN_ACK, FIN_ACK;
SIGNALLIST sender_channel = DATA, SYN, FIN;
SIGNALLIST channel_receiver = DATA, SYN, FIN;
SIGNALLIST receiver_channel = DATA_ACK, SYN_ACK, FIN_ACK;
```

```
/* Buffer for sender, 4 positions for testing
   The size of buffer can be changed later */

syntype index1 = Integer
  constants 0:3
endsyntype;

NEWTYPE
Buffer1 array(index1, Packet)
ENDNEWTYPE;
```

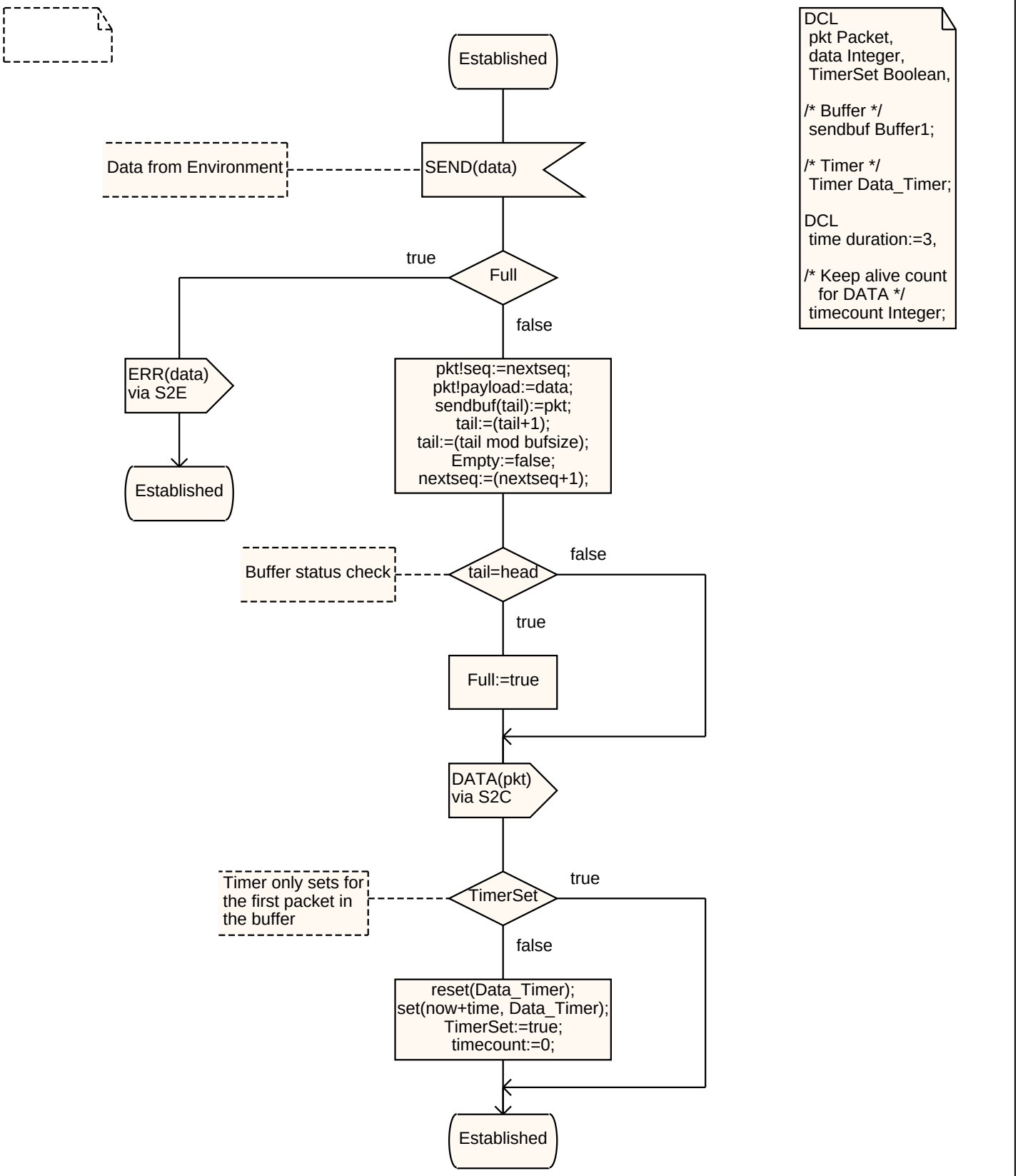
```
/* Buffer for receiver, 2 positions for testing.
   The size of buffer can be changed later */

syntype index2 = Integer
  constants 0:1
endsyntype;

NEWTYPE
Buffer2 array(index2, Packet)
ENDNEWTYPE;
```


2
block SenderBlock
1(1)
EnvironSender
E2S
(env_sender)
Main(1,1)
S2E
(sender_env)
SenderEnviron
5
SenderChannel
S2C
(sender_channel)
C2S
(channel_sender)
ChannelSender

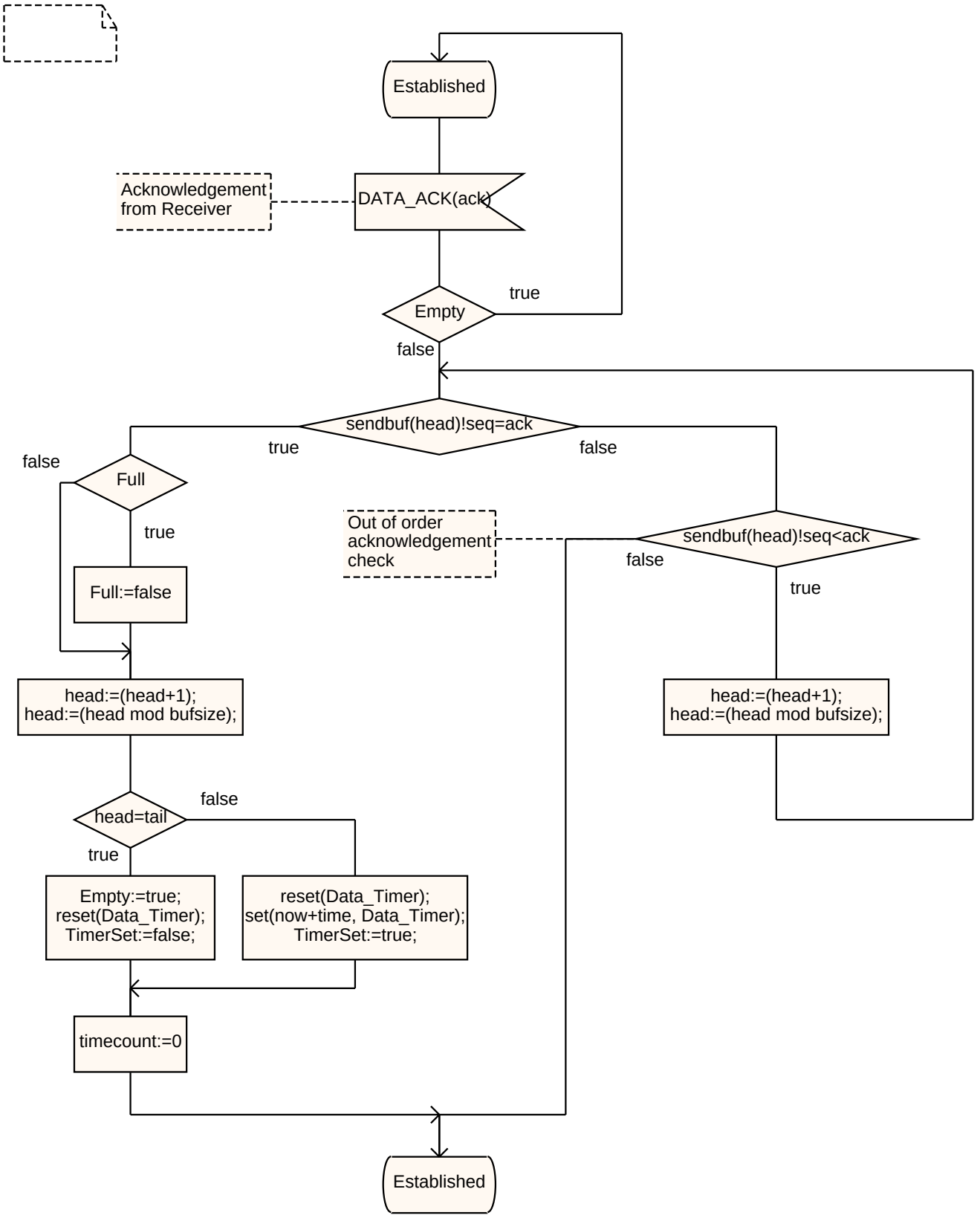

DCL
init Integer,
/* Keep alive threshold */
limit Integer,
/* Buffer parameters */
head Integer,
tail Integer,
bufsize Integer,
/* Buffer status */
Full Boolean,
Empty Boolean;
Timer SYN_Timer;
DCL
timeout duration:=3,
/* keep alive count for
SYN */
syn_count Integer;
Closed
CONN
init:=0;
head:=0;
tail:=0;
bufsize:=4;
Full:=false;
Empty:=true;
limit:=5;
Parameter Initialization
SYN(init)
set(now+timeout, SYN_Timer);
syn_count:=0;
Wait_SYN
Closed
*
Other Signals
Closed

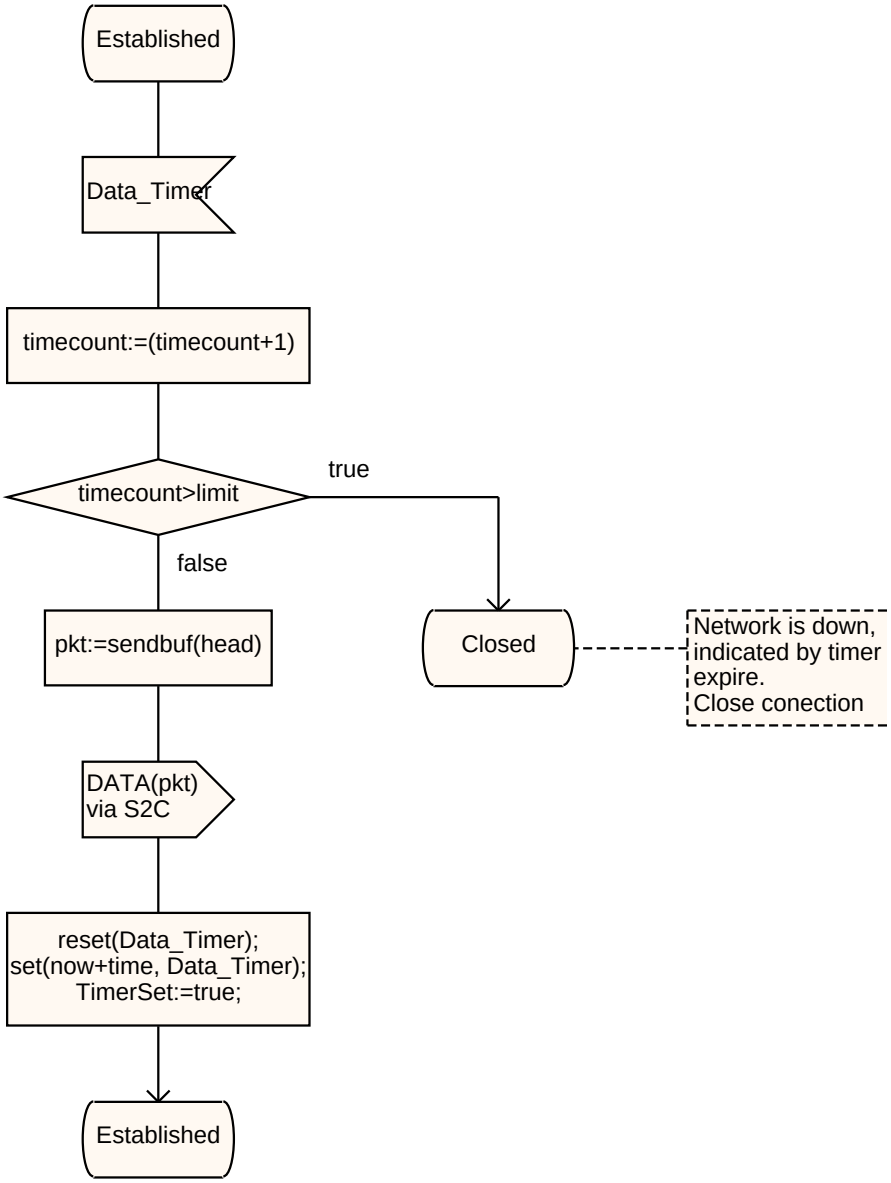

DCL
ack Integer,
nextseq Integer;
Established
*
Other signals
Established
Wait_SYN
SYN_Timer
SYN_ACK(ack)
syn_count:=(syn_count+1);
reset(SYN_Timer);
nextseq:=(ack+1);
TimerSet:=false;
syn_count:=0;
Timer expiration over
the threshold.
Connection close.
true
syn_count>limit
false
Established
Closed
SYN(init)
reset(SYN_Timer);
set(now+timeout, SYN_Timer);
Wait_SYN

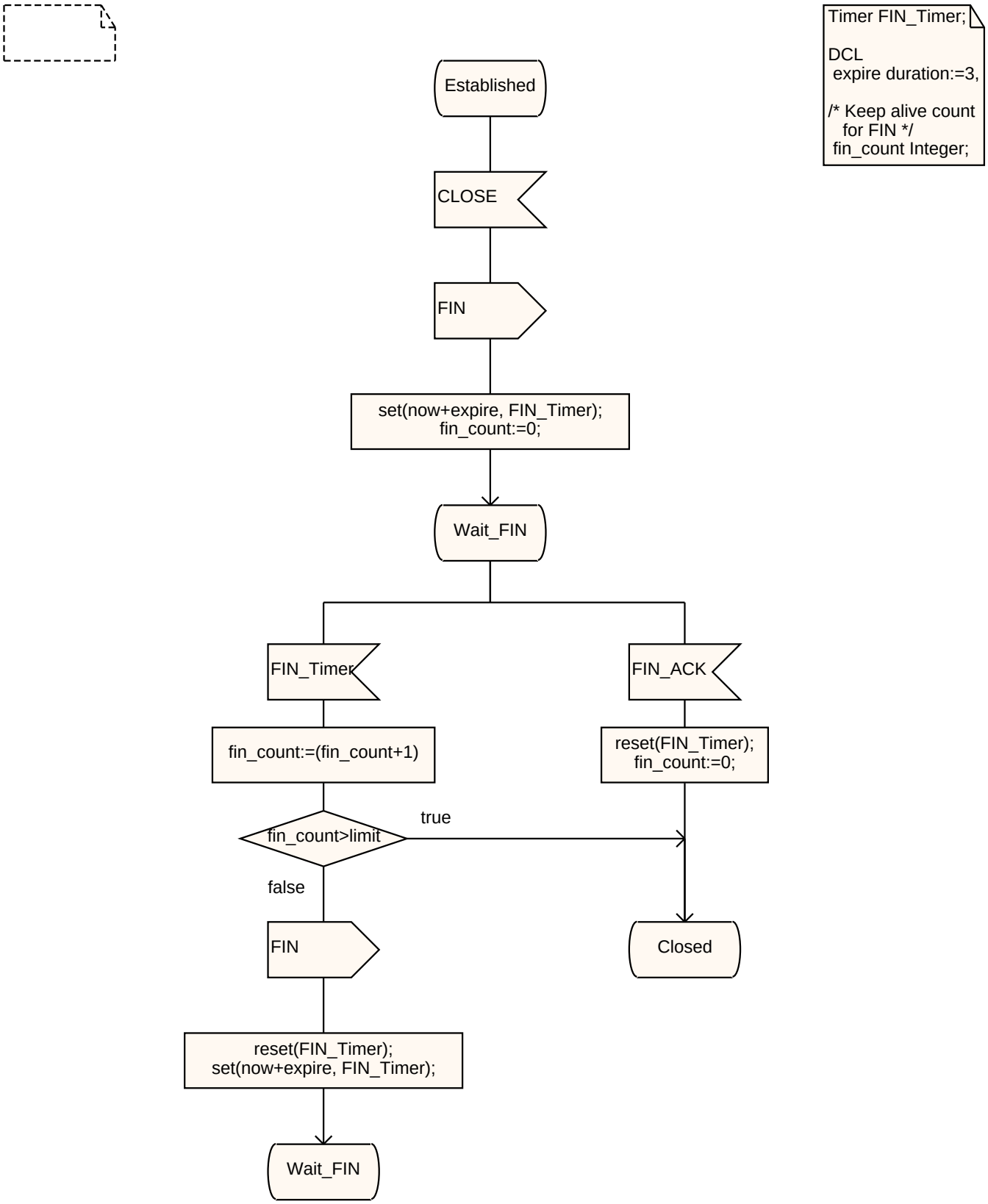
process <<block SenderBlock>> Main

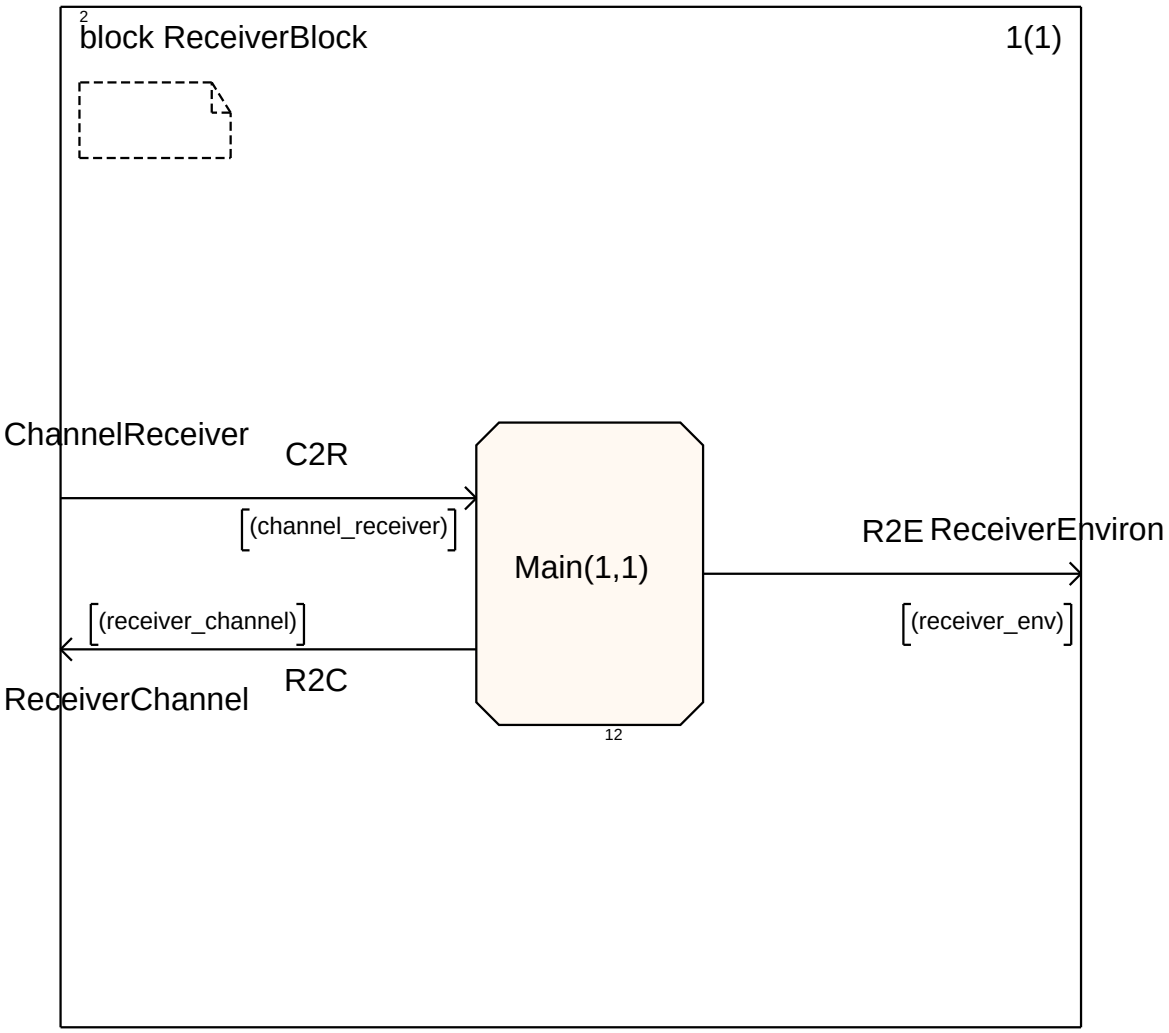


process <<block SenderBlock>> Main

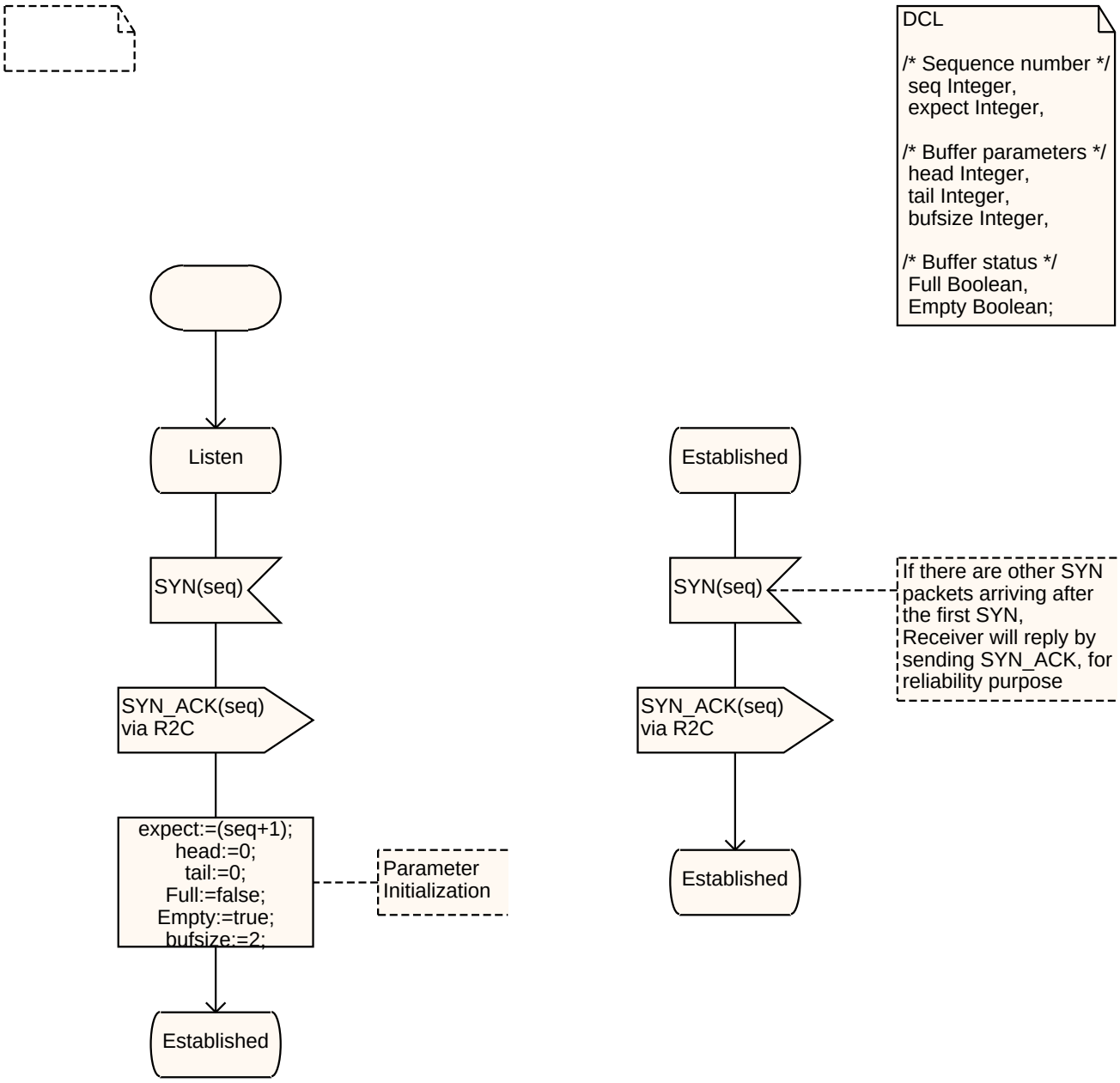



Established
Data_Timer
timecount:=(timecount+1)
timecount>limit
true
false
pkt:=sendbuf(head)
Closed
Network is down, indicated by timer expire.
Close conection
DATA(pkt)
via S2C
reset(Data_Timer);
set(now+time, Data_Timer);
TimerSet:=true;
Established


Timer FIN_Timer;
DCL
expire duration:=3,
/* Keep alive count
for FIN */
fin_count Integer;
Established
CLOSE
FIN
set(now+expire, FIN_Timer);
fin_count:=0;
Wait_FIN
FIN_Timer
FIN_ACK
fin_count:=(fin_count+1)
reset(FIN_Timer);
fin_count:=0;
fin_count>limit
true
false
FIN
Closed
reset(FIN_Timer);
set(now+expire, FIN_Timer);
Wait_FIN

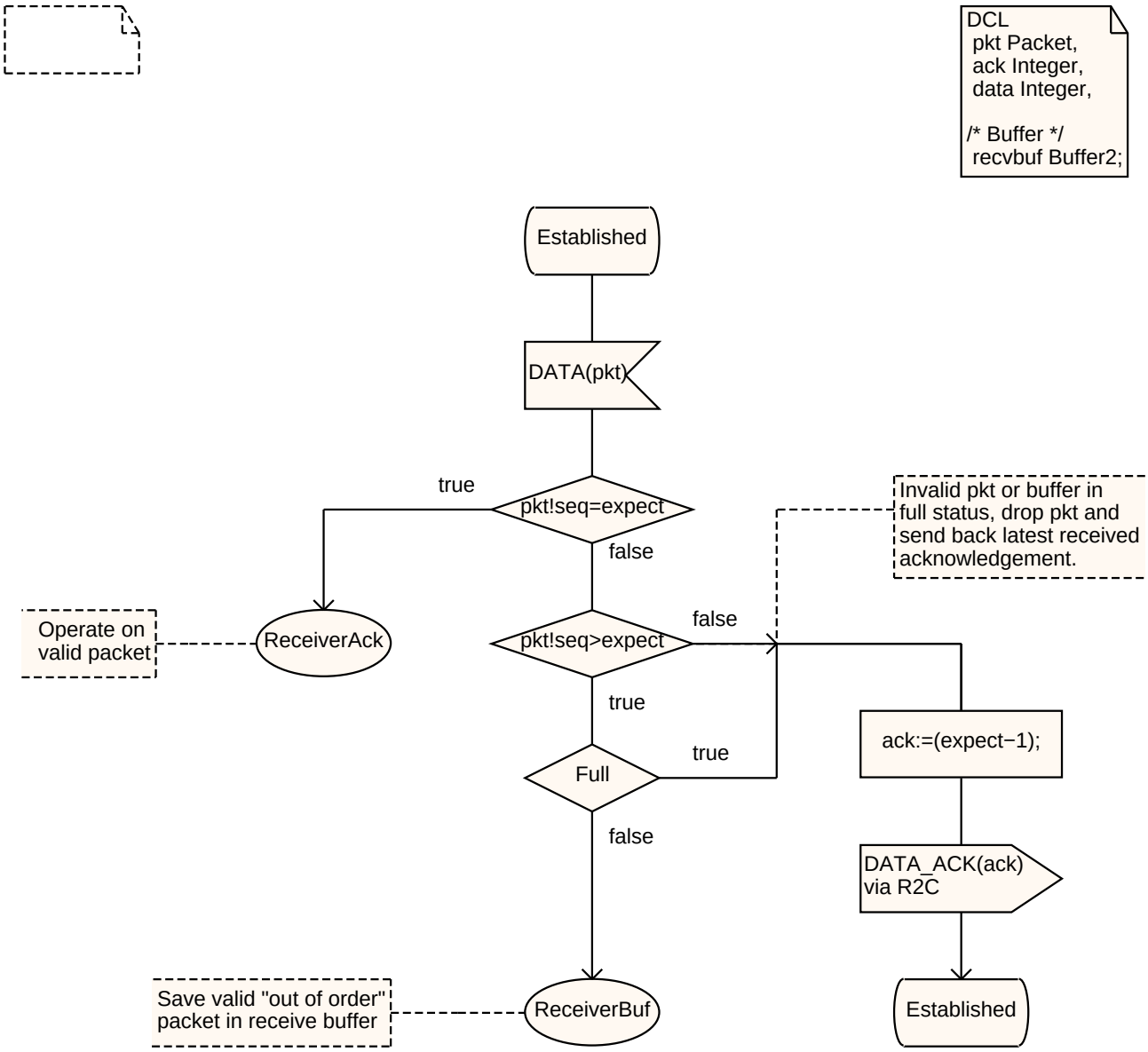

2
block ReceiverBlock
1(1)
ChannelReceiver
C2R
(channel_receiver)
Main(1,1)
R2E ReceiverEnviron
(receiver_channel)
(receiver_env)
R2C
ReceiverChannel
12

process <<block ReceiverBlock>> Main



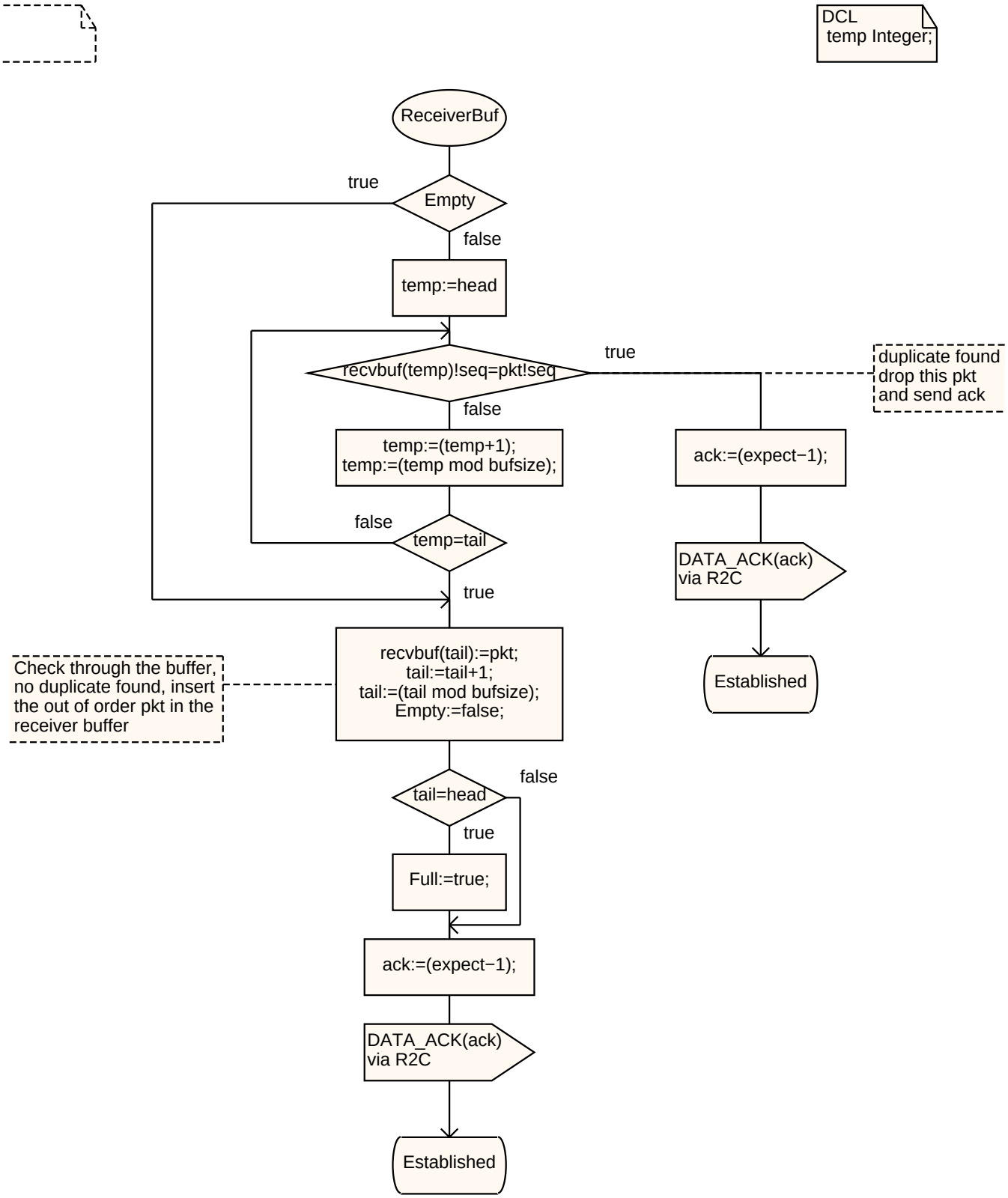
11

process <<block ReceiverBlock>> Main

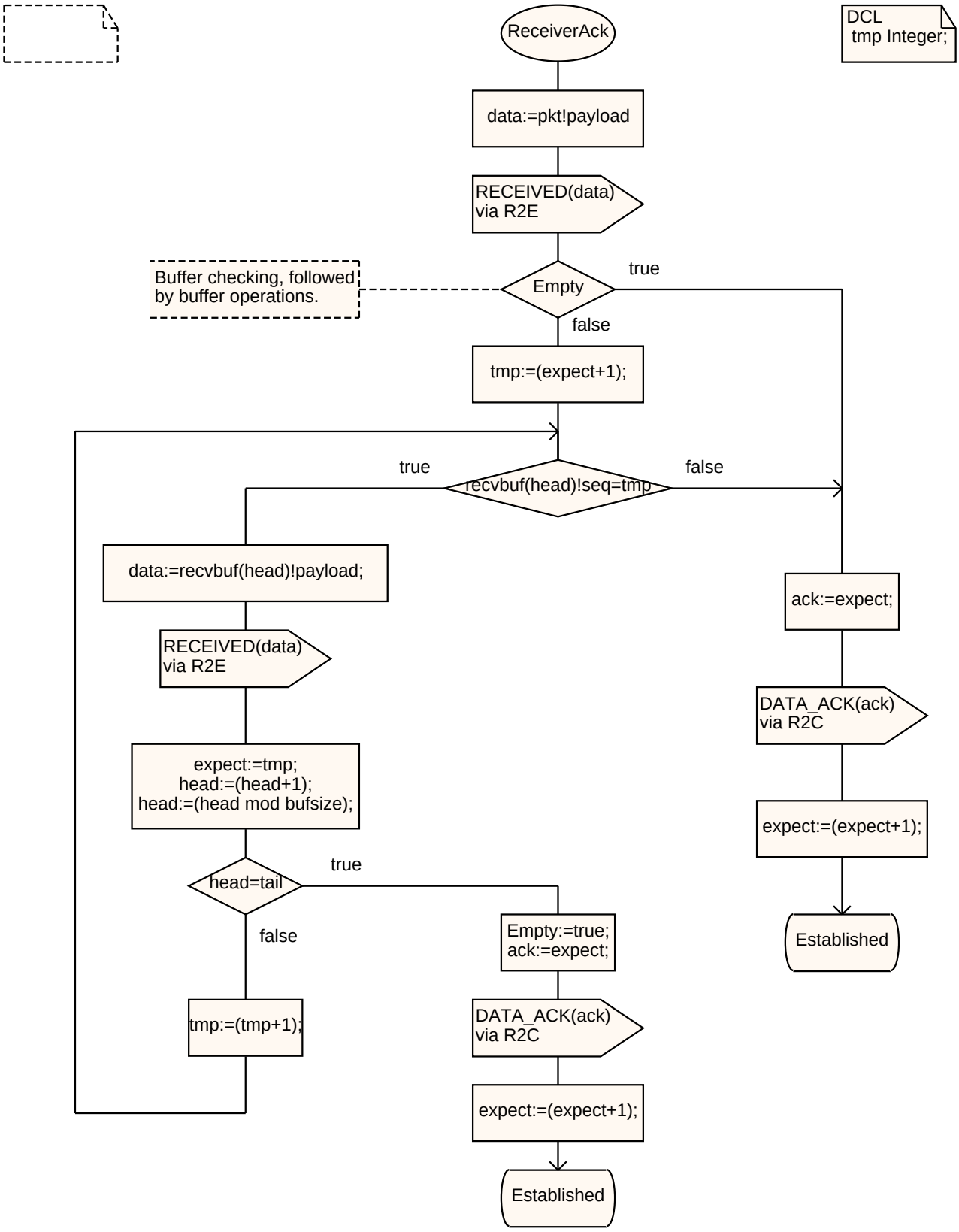


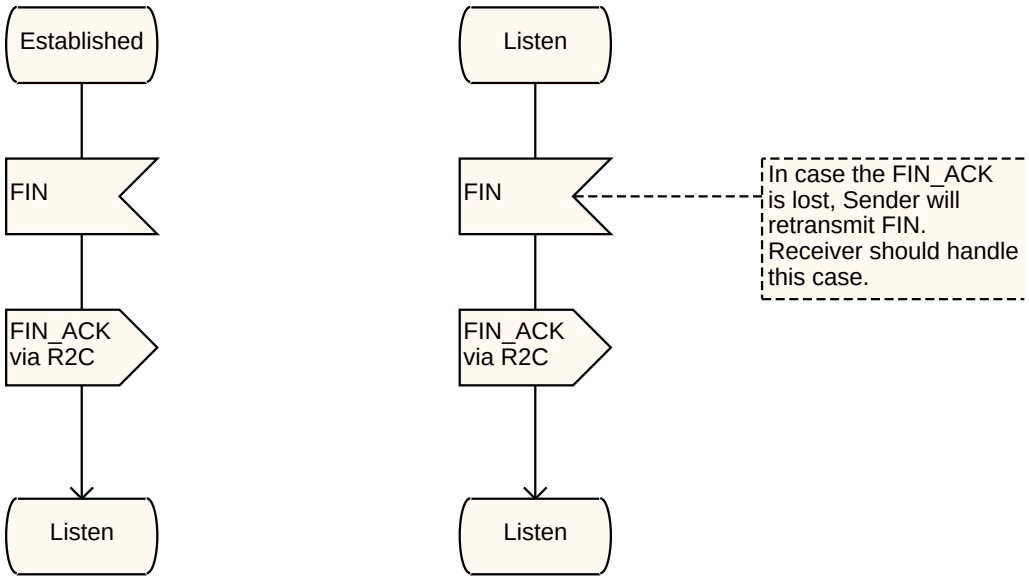
11

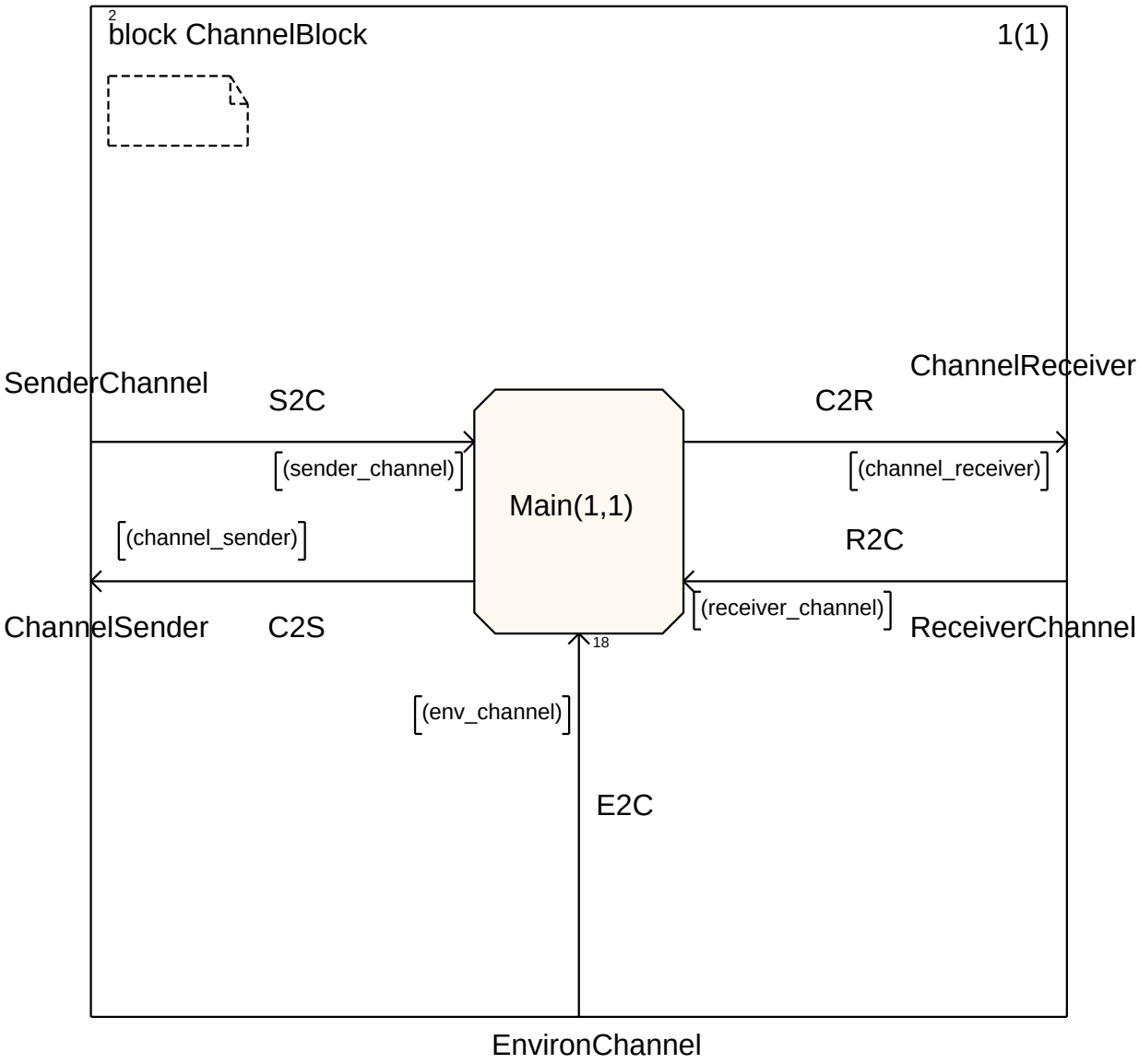
process <<block ReceiverBlock>> Main



process <<block ReceiverBlock>> Main



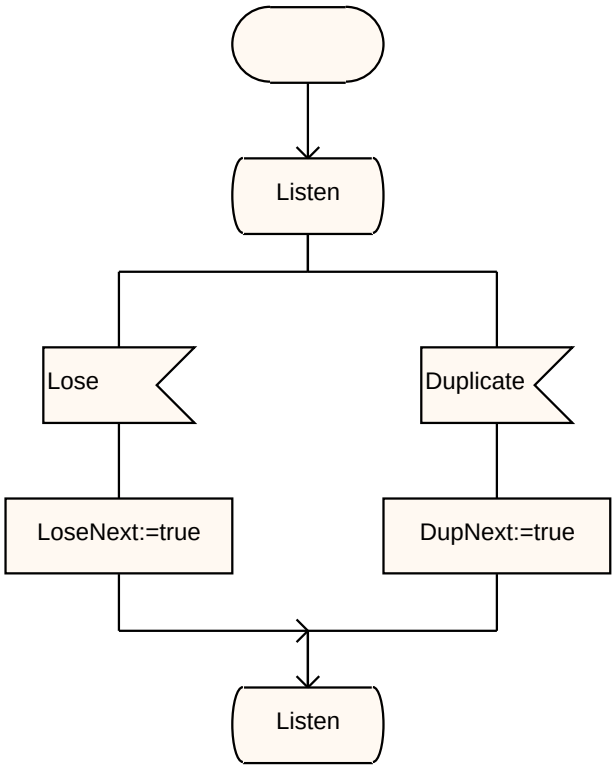

Established
FIN
FIN_ACK
via R2C
Listen
Listen
FIN
FIN_ACK
via R2C
Listen
In case the FIN_ACK
is lost, Sender will
retransmit FIN.
Receiver should handle
this case.


2
block ChannelBlock
1(1)
SenderChannel
S2C
(sender_channel)
(channel_sender)
ChannelSender
C2S
Main(1,1)
18
(env_channel)
E2C
ChannelReceiver
C2R
(channel_receiver)
R2C
(receiver_channel)
ReceiverChannel
EnvironChannel

17

process <<block ChannelBlock>> Main

DCL
/* Control signal status */
LoseNext Boolean,
DupNext Boolean;



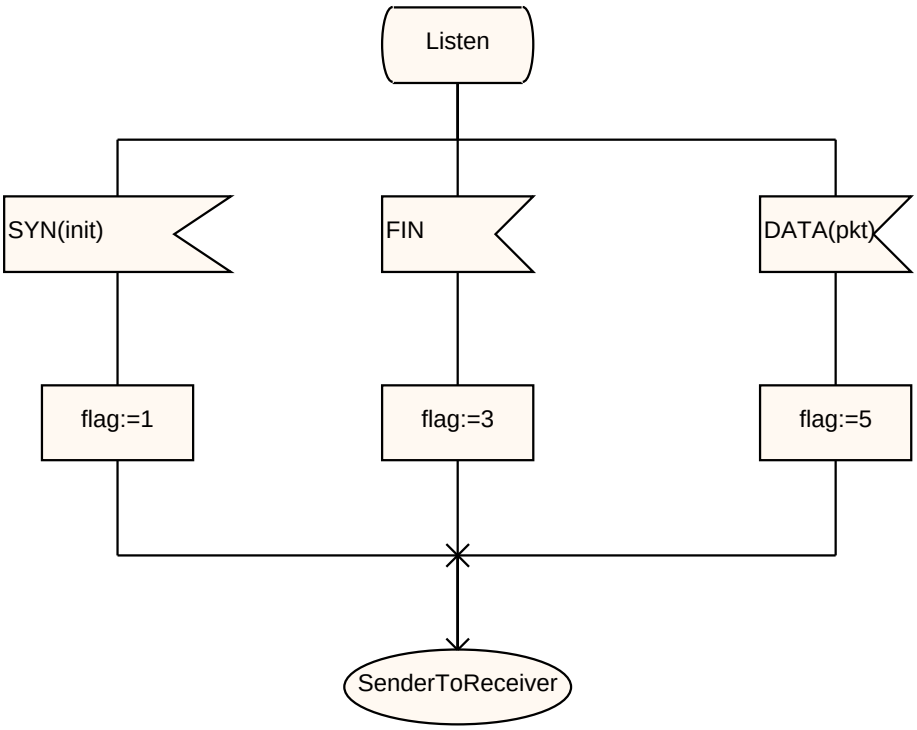
17

process <<block ChannelBlock>> Main 2(5)

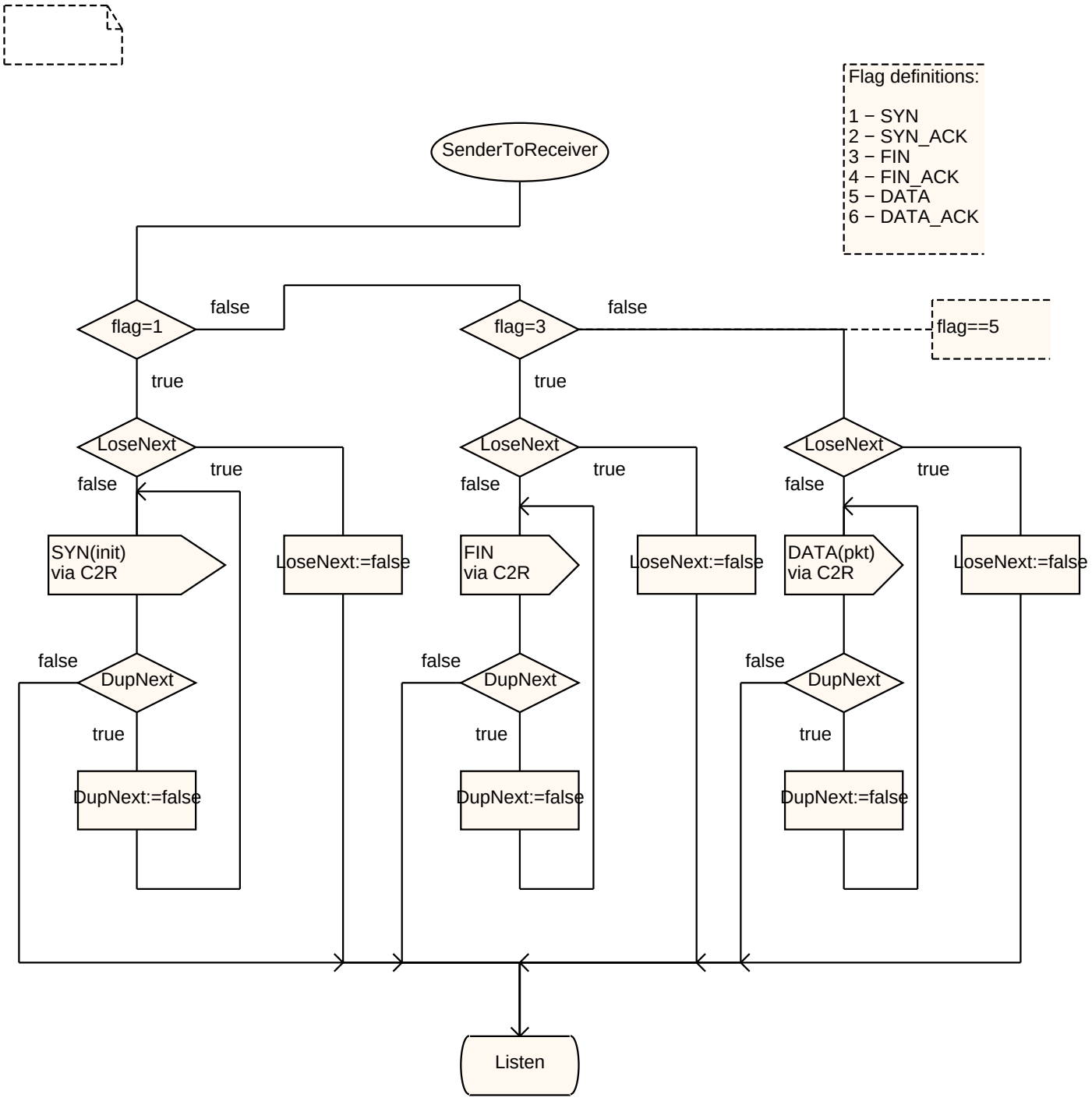
DCL
init Integer,
pkt Packet,
flag Integer;

Flag definitions:

1 − SYN
2 − SYN_ACK
3 − FIN
4 − FIN_ACK
5 − DATA
6 − DATA_ACK



process <<block ChannelBlock>> Main 3(5)

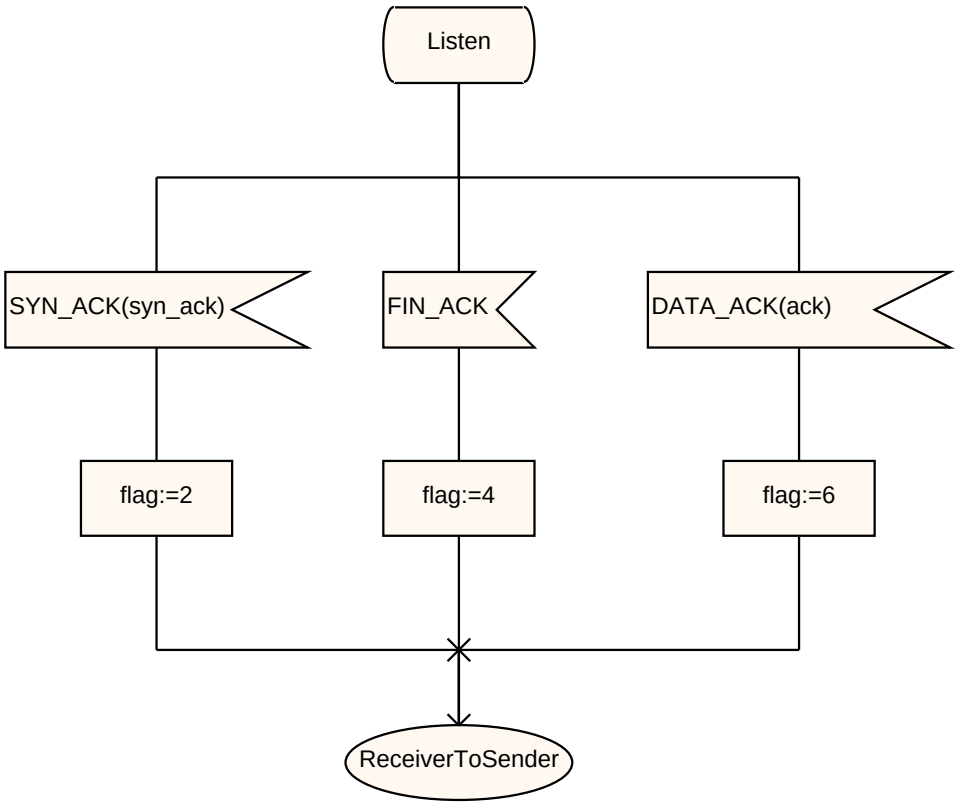


process <<block ChannelBlock>> Main

DCL
syn_ack Integer,
ack Integer;

Flag definitions:

1 − SYN
2 − SYN_ACK
3 − FIN
4 − FIN_ACK
5 − DATA
6 − DATA_ACK



process <<block ChannelBlock>> Main