Bowei Wang
1462495
Objectives:
I should first understand the process of lanc works, know the usage of different part in Flood2 and Flood3.
In part 2, I should understand the process of the programs, handle several cases of lanc to make the program have the ability to handle corrupted packets.

part1:
1.
The variable seqno is used in stop and wait protocol, with initial value of 1
The value of it is defined in structure to represent the number of date.
Another variable MAXHOPS is also used, with initial value of 4.
This variable is used to monitior how many ACK is arrived that not the expected one.

2.
Vary MAXHOPS will have affect on algrithm performance.
with MAXHOPS = 4 in node 1, after 100 secs, we get:
generated: 14 received:14
with MAXHOPS = 2 in node 1, after 100 secs, we get:
generated:27 received:22
with MAXHOPS = 1 in node 1, after 100 secs, we get:
generated: 46 received:39
With a largher MAXHOPS, the host will wait a longer time for each packets, that is it will jumps to next packet when 4 irrelevant data received, therefor it can get more ACK.
However the MAXHOPS is smaller, the host will wait for a shorter time, that is when 1 irrelevant data received, it will jump to next one, so it can send more packets with a lower back-information get rate.

3.
ALL LINKS & ~(1<<arrived on) will calculate the seq number of the pacaket that haven't receive a ACK.
The floods use this expression in
"flood2(packet, length, ALL_LINKS & ~(1<<arrived_on));"
to retransmit the pacakets that not yet received the correspond ACK.

4.
a.if CHECK(CNET_disable_application(p.dest)); is comment out, the program will terminate with 8 nodes on network for 5 mins, besides, many error is generated, the receive rate is low.

b.Under the case unmodified program runs on a network of 21 nodes, the program will terminate befroe 5 mins, with most of the sent packets received.
static EVENT_HANDLER(down_to_network)

5.
a. It will comes to NL_savehopcount in line 135 and p->hopcount will be saved in NL_table[t].minihops.

b. It will comes to NL_savehopcount in line 135 and p->hopcount will be compared with the previous minihops, if it is smaller than previous minhops, it will be saved.

c.It will check whether it is the packet that the host is expected if so, pass that into NL_savehopcount and save the correspond link and minihop.

Part2:
Design Overview:
The design of my protocol is first look through all the programs and comments that given by the lab3-lan.c. Then remember several points that mentioned in the comment. Go over to the LANC_manager. Handle all the mentioned situations.
-For app case, do several asserts, then set state to busy, start a timer for cs.
-For collosion case, printf the debug information, do several assert and stop timer.
    -if backoff is not less than max backoff, drop the frame by reinit the lanc, implement the dropped frame number then enable all application layer.
    -if backoff is less than max backoff, set state to collieded, implement the backoff number, generate a new backoff_rn, start a new timer for backoff.
-For cs case, get the temp frame length.
    -if cs_flag == 1, check whether the link is available, if so, send current frame and start a timer for frame transmit. If link is not available, set a timer for next cs.
    -if cs_flag is not 1. Send current frame and start a timer for frame transmit.
-For backoff case,set a timer for cs.
-For frame transmit case, enable application layer for all nodes. Implement success frame number, then reinitial the lanc.
-For default case, print error happened, show debug info, then exit the process.

Program Status: The program currently works well. The difficulty I met is that I didn't find the way to drop the frame, finally I found that I can drop the frame by reinit the lanc. Besides, I was confused that the dropped frames is 0 at first, which shouldn't happen, finally I found that I should implement the dropped frame number while reinit the frame.

Testing: I tested my program by running all 4 files under cs_flag = 0 and cs_flage = 1 two cases.

Results:

Cs_flag = 0

Network	Global Statistics			Protocol Statistics (for the first node)			
	Messages generated	Messages delivered	Frame collisions	tx_frames	success frames	dropped frames	rx_frames
LAN-5	2872	2514	1453	624	544	80	498
LAN-10	5851	4229	6671	596	440	156	406
LAN-15	8929	5422	14747	616	374	242	341
LAN-20	11838	6258	23920	610	330	280	314

Cs_flag = 1

Network	Global Statistics			Protocol Statistics (for the first node)			
	Messages generated	Messages delivered	Frame collisions	tx_frames	success frames	dropped frames	rx_frames
LAN-5	2941	2941	0	571	571	0	563
LAN-10	5891	5891	0	609	609	0	613
LAN-15	8638	8637	0	583	583	0	597
LAN-20	11426	11424	0	541	541	0	564

We can find that when cs_flag = 1, which means the carrier_sense is used, the collisions come to 0, the message deliver rate is extremely high. Obvious, the way that user cs_flag = 1 is much better than 0 case.

Acknowledgements: The help of TAs and comments is very useful. Especially the comments in the lab3-lan.c file, it gave several basic ideas about what I should begin on this assgnment.