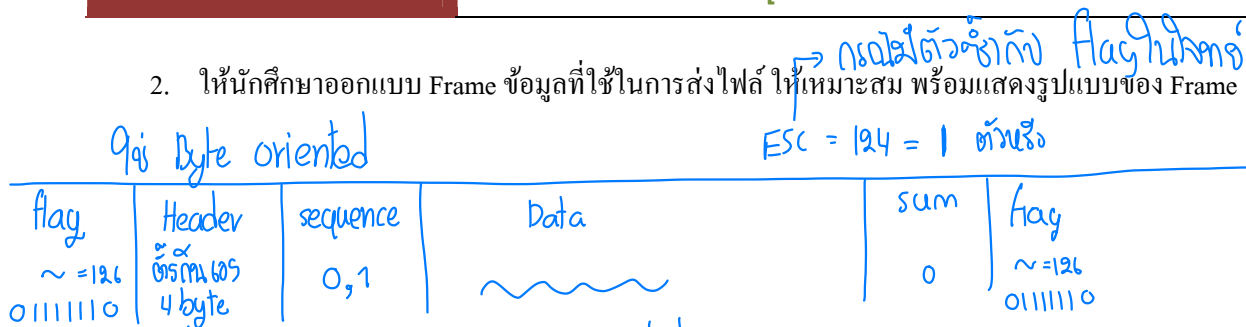


2. ให้นักศึกษาออกแบบ Frame ข้อมูลที่ใช้ในการส่งไฟล์ ให้เหมาะสม พร้อมแสดงรูปแบบของ Frame



3. ให้นักศึกษาเขียนผังงาน (Flow Chart) โปรแกรมในการรับ-ส่งไฟล์ข้อมูล

หรือ whatever

→ ตัวซ้ำใน data ให้ข้อมูลเน้นตรงกับ flag ในข้อ
→ ตัวซ้ำ: 8 byte → เลข sum

วิชา Data Communication Laboratory

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

การทดลองที่ 4 Protocol Design

วัตถุประสงค์

1. เข้าใจการทำงานของโปรโตคอลการส่งข้อมูล (Flow & Error Control Protocol)
2. สามารถเขียนโปรแกรมประยุกต์เพื่อสื่อสารผ่านพอร์ทอนุกรมได้

การทดลองที่ 4.1 การเขียนโปรแกรมส่งไฟล์ผ่านพอร์ทอนุกรม

1. ออกแบบโปรแกรมในการรับ-ส่งไฟล์ข้อมูล โดยมีข้อกำหนดต่อไปนี้
 - 1.1. เป็นการส่งข้อมูลทางเดียว (Unidirectional)
 - 1.2. เลือกไฟล์ที่จะส่งข้อมูลได้ & เลือกรับชื่อไฟล์ได้
 - 1.3. ส่งข้อมูลในรูปแบบ Frame ที่เหมาะสม
 - 1.4. ใช้ Stop and Wait ARQ Protocol ในการส่งข้อมูล
 - 1.5. Acknowledgement เป็นแบบกำหนดเอง (a : ปกติ / n : ACK หาย / r : Data หาย)
 - 1.6. Timeout เป็นแบบกำหนดเอง (เกิด Timeout เมื่อเกิด t ระหว่างส่งข้อมูล)
 - 1.7. ตัวอย่างโปรแกรมการรับส่ง

Sender	Receiver
Send file : <u>C:\dir\data.txt</u>	Sender Send : <u>data.txt</u> Save as : <u>C:\commu.txt</u>
Send frame : 0 Data : xxxxxxxxxxxxxxxx Timeout : Receive ACK1	Receive frame : 0 Data : xxxxxxxxxxxxxxxx Action frame : <u>a</u> Received & Send ACK1
Send frame : 1 Data : yyyyyyyyyyyyyyy Timeout : <u>t</u> Retransmit frame 1	Receive frame : 1 Data : yyyyyyyyyyyyyyy Action frame : <u>r</u> Reject & Sleep
Send frame : 1 Data : yyyyyyyyyyyyyyy Timeout : <u>t</u> Retransmit frame 1	Receive frame : 1 Data : yyyyyyyyyyyyyyy Action frame : <u>n</u> Received & Sleep
Send frame : 1 Data : yyyyyyyyyyyyyyy Timeout : Receive ACK0	Receive frame : 0 Data : yyyyyyyyyyyyyyy Action frame : <u>a</u> Reject & Send ACK0
Send frame : 0	

Stop-and-Wait Automatic Repeat Request

• Algorithm

กฎข้อที่ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33

```

1 Sn = 0; // Frame 0 should be sent first
2 canSend = true; // Allow the first request to go
3 while(true) // Repeat forever
4 {
5     WaitForEvent(); // Sleep until an event occurs
6     if(Event(RequestToSend) AND canSend)
7     {
8         GetData(); // Get data to send
9         MakeFrame(Sn); //The seqNo is Sn
10        StoreFrame(Sn); //Keep copy
11        SendFrame(Sn);
12        StartTimer();
13        Sn = Sn + 1;
14        canSend = false;
15    }
16    WaitForEvent(); // Sleep
17    if(Event(ArrivalNotification)) // An ACK has arrived
18    {
19        ReceiveFrame(ackNo); //Receive the ACK frame
20        if(not corrupted AND ackNo == Sn) //Valid ACK
21        {
22            StopTimer();
23            PurgeFrame(Sn-1); //Copy is not needed
24            canSend = true;
25        }
26    }
27    if(Event(TimeOut)) // The timer expired
28    {
29        StartTimer();
30        ResendFrame(Sn-1); //Resend a copy check
31    }
32 }
33

```

Handwritten notes for the sender side:

- Line 1: $S_n = 0$ (Frame 0 should be sent first)
- Line 2: `canSend = true;` (Allow the first request to go)
- Line 3: `while(true)` (Repeat forever)
- Line 4: `{` (Start of loop)
- Line 5: `WaitForEvent();` (Sleep until an event occurs)
- Line 6: `if(Event(RequestToSend) AND canSend)` (Check if data is ready to send and can send)
- Line 7: `{` (Start of if block)
- Line 8: `GetData();` (Get data to send)
- Line 9: `MakeFrame(Sn);` (The seqNo is S_n)
- Line 10: `StoreFrame(Sn);` (Keep copy)
- Line 11: `SendFrame(Sn);`
- Line 12: `StartTimer();`
- Line 13: `Sn = Sn + 1;`
- Line 14: `canSend = false;`
- Line 15: `}` (End of if block)
- Line 16: `WaitForEvent();` (Sleep)
- Line 17: `if(Event(ArrivalNotification))` (An ACK has arrived)
- Line 18: `{` (Start of if block)
- Line 19: `ReceiveFrame(ackNo);` (Receive the ACK frame)
- Line 20: `if(not corrupted AND ackNo == Sn)` (Valid ACK)
- Line 21: `{` (Start of if block)
- Line 22: `StopTimer();`
- Line 23: `PurgeFrame(Sn-1);` (Copy is not needed)
- Line 24: `canSend = true;`
- Line 25: `}` (End of if block)
- Line 26: `}` (End of if block)
- Line 27: `if(Event(TimeOut))` (The timer expired)
- Line 28: `{` (Start of if block)
- Line 29: `StartTimer();`
- Line 30: `ResendFrame(Sn-1);` (Resend a copy check)
- Line 31: `}` (End of if block)
- Line 32: `}` (End of loop)
- Line 33: `}` (End of function)

```

1 Rn = 0; // Frame 0 expected to arrive first
2 while(true)
3 {
4     WaitForEvent(); // Sleep until an event occurs
5     if(Event(ArrivalNotification)) //Data frame arrives
6     {
7         ReceiveFrame();
8         if(corrupted(frame)); //Data frame is corrupted
9         {
10            SendFrame(Rn); //Resend the frame
11        }
12        if(seqNo == Rn) //Valid data frame
13        {
14            ExtractData();
15            DeliverData(); //Deliver data
16            Rn = Rn + 1;
17        }
18        SendFrame(Rn); //Send an ACK
19    }
20 }

```

Handwritten notes for the receiver side:

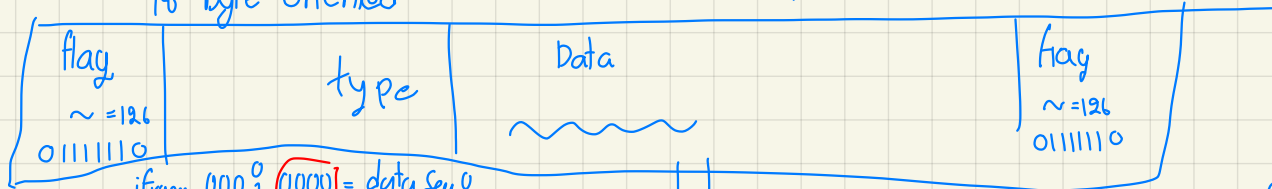
- Line 1: $R_n = 0$ (Frame 0 expected to arrive first)
- Line 2: `while(true)`
- Line 3: `{` (Start of loop)
- Line 4: `WaitForEvent();` (Sleep until an event occurs)
- Line 5: `if(Event(ArrivalNotification))` (Data frame arrives)
- Line 6: `{` (Start of if block)
- Line 7: `ReceiveFrame();`
- Line 8: `if(corrupted(frame));` (Data frame is corrupted)
- Line 9: `{` (Start of if block)
- Line 10: `SendFrame(Rn);` (Resend the frame)
- Line 11: `}` (End of if block)
- Line 12: `if(seqNo == Rn)` (Valid data frame)
- Line 13: `{` (Start of if block)
- Line 14: `ExtractData();`
- Line 15: `DeliverData();` (Deliver data)
- Line 16: `Rn = Rn + 1;`
- Line 17: `}` (End of if block)
- Line 18: `SendFrame(Rn);` (Send an ACK)
- Line 19: `}` (End of if block)
- Line 20: `}` (End of loop)

timeout เปรียบ manual ๑:

→ เปรียบตัวซ้ำกัน flag ใน packet

ESC = 124 = 1 ตัวซ้ำ

is Byte oriented



iframe 000 0000 = data seq 0

sframe 100 1111 = ack + Seq 0

~~101 1111 = ack + Seq 1~~

1100 1111 = correct → send
1110 1111 = disconnect

→ ต้องมีวิธีรับ data ว่าข้อมูลเริ่มตรงกับ flag ใน packet

วิธีที่ 1: 8 byte → คำนวณ sum

→ ถ้า byte ตัวแรก f เท่ากับ flag ก่อน เราต้อง ESC ใน packet

โครงสร้าง ตัวแปรที่ตัวรับ

```
# timer (unsigned long)
canSend (bool)
frameSequence (int)
fp (FILE *p)
filename (char array)
```

เพิ่มตัว #define flag, esc, ใช้ตัวใช้เพียงอย่างเดียว

function ที่เราต้องได้ใช้

→ ตัวรับ (bit) ให้รับ ms check timeout. #181 เป็นตัวรับ

receive => check ว่าข้อมูลตัวรับตรงกับ ESC หรือไม่

→ แล้วคำนวณ sum

→ ถ้าไม่ตรงกับ sum เขียนค่าซ้ำ