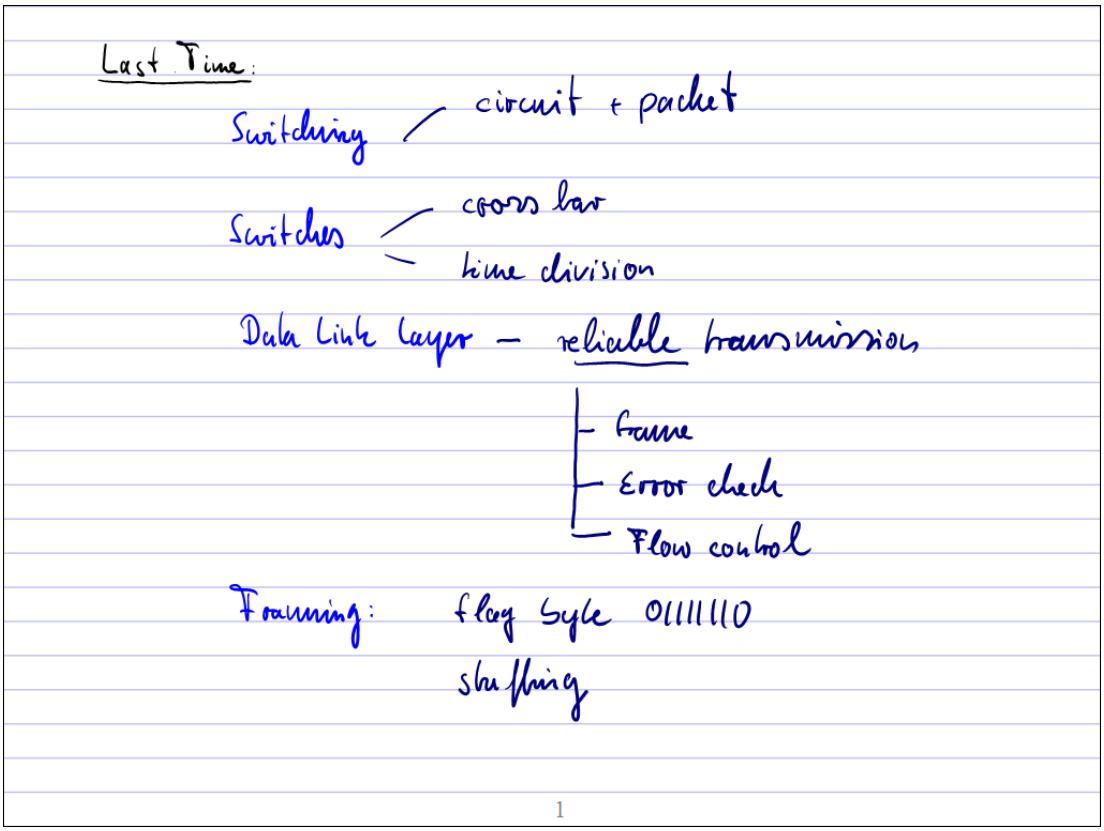
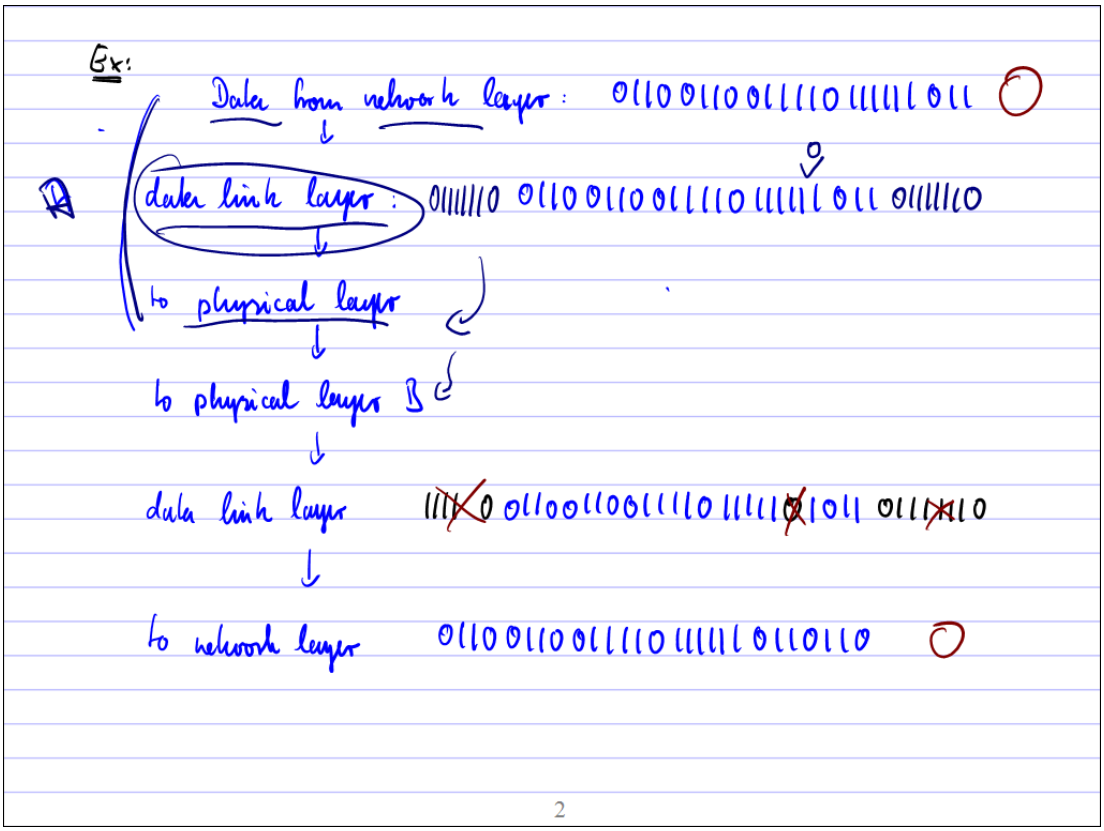
Panel 1



Panel 2



Panel 3

Handling Errors

error : inverting a bit ∈ (extra bit, lost bit)

error correction : build enough redundancy into a frame to detect + fix error.

Big overhead, used for unreliable simplex chan.

error detection : detect an error but you don't fix it; instead ask for a retransmission.

3

Panel 4

Simple Error Detection: Parity bit

even parity : add 0 or 1 to data of frame for even # of 1's

odd parity : add 0 or 1 to data of frame for odd # of 1's

Recall that an even parity bit is computed so that the total number of 1's in the data is even.

a. What is the parity of 10101111001110010010011100100111?

b. Suppose a bit pattern has been constructed by (1) computing an even parity bit and appending it to the end of the data bits (2) stuffing by replacing all sequences of 11111 by 111110, and (3) framing by adding the flag byte 01111110 at the beginning and end of each frame. Then you receive the pattern:

01111110101011001001110010010000100010111000101110000110101111100111111010010010011010101111110011111110100000101000111101111100111111101001001010011010101111110

How many frames are there, which frame is invalid, and what is the original data in each valid frame?

frame 1: valid

frame 2: valid

frame 3: invalid

frame 4: valid

next program

4

Panel 5

Parity bit : can detect single errors but not double errors
odd errors but not even ones

More powerful error detection: Cyclic redundancy check
CRC

Sender + receiver agree on Generator polynomial $G(x)$
Each frame of k bits is a $(k-1)$-degree polynomial.
$1001 \sim x^4 + x^1 + x^0$
Add checksum so that new polynomial is
divisible by $G(x)$. What you do is:
Divide original polyn. $M(x)$ by $G(x)$ and find
remainder

5

Panel 6

$$\begin{array}{r} 411\ R\,2 \\ 3\overline{)1235} \\ 12 \\ \hline 03 \\ 3 \\ \hline 05 \end{array}$$

1235 is not div. by 3 (R2)
⇒ but $1235 - 2$ is

What type of errors can you catch (divisible by $G(x)$)
If error occurs then instead of $T(x)$ you get
$T(x) + E(x)$ ⇒ check $(T(x) + E(x)) / G(x)$
$= \underbrace{T(x)/G(x)}_{\text{no remainder}} + E(x)/G(x)$

If choose $G(x)$ carefully, you can detect most
errors!

6

Panel 7

CRC-12: $G(x) = x^{12} + x^{11} + x^3 + x^2 + x + 1$

CRC-16: $G(x) = x^{16} + x^{15} + x^2 + 1$

CRC-CCIT: $G(x) = x^{16} + x^{12} + x^5 + 1$

) for 8 bit chars

CRC-16 catches: all single errors
all double errors
all errors with odd # of flipped bits
all burst errors length 16 or less
(99.997 of burst errors of length 17
99.998 of burst error lengths than 17

All calculations are done via shift registers in hardware.

7

Panel 8

Elementary Data Link Protocols

Outline 4 different protocols of increasing complexity:

Protocol 1: Unrestricted Simplex: perfect transmission phys. + network are infinitely fast always ready

Protocol 2:

8

Panel 9

File protocol.h

```
#define MAX_PKT 1024;

typedef enum {false, true} boolean;
typedef unsigned int seq_nr;
typedef struct {unsined char data[MAX_PKT] } packet;
typedef enum {data, ack, nak} frame_kind;

typedef struct
{   frame_kind kind;
    seq_nr seq;
    seq_nr ack;
    packet info;
} frame;

void wait_for_event(event_type *event);
void from_network_layer(packet *p);
void to_network_layer(packet *p);
void from_physical_layer(frame *r);
void to_physical_layer(frame *s);
void start_timer(seq_nr k);
void stop_timer(seq_nr k);
void start_ack_timer(void);
void stop_ack_timer(void);
void enable_network_layer(void);
void disable_network_layer(void);
#define inc(k) if (k < MAX_SEQ) k = k + 1; else k = 0;
```

Spring
M 4-5
ICR
C + Unix

Fall : 08
Robotics
course

9