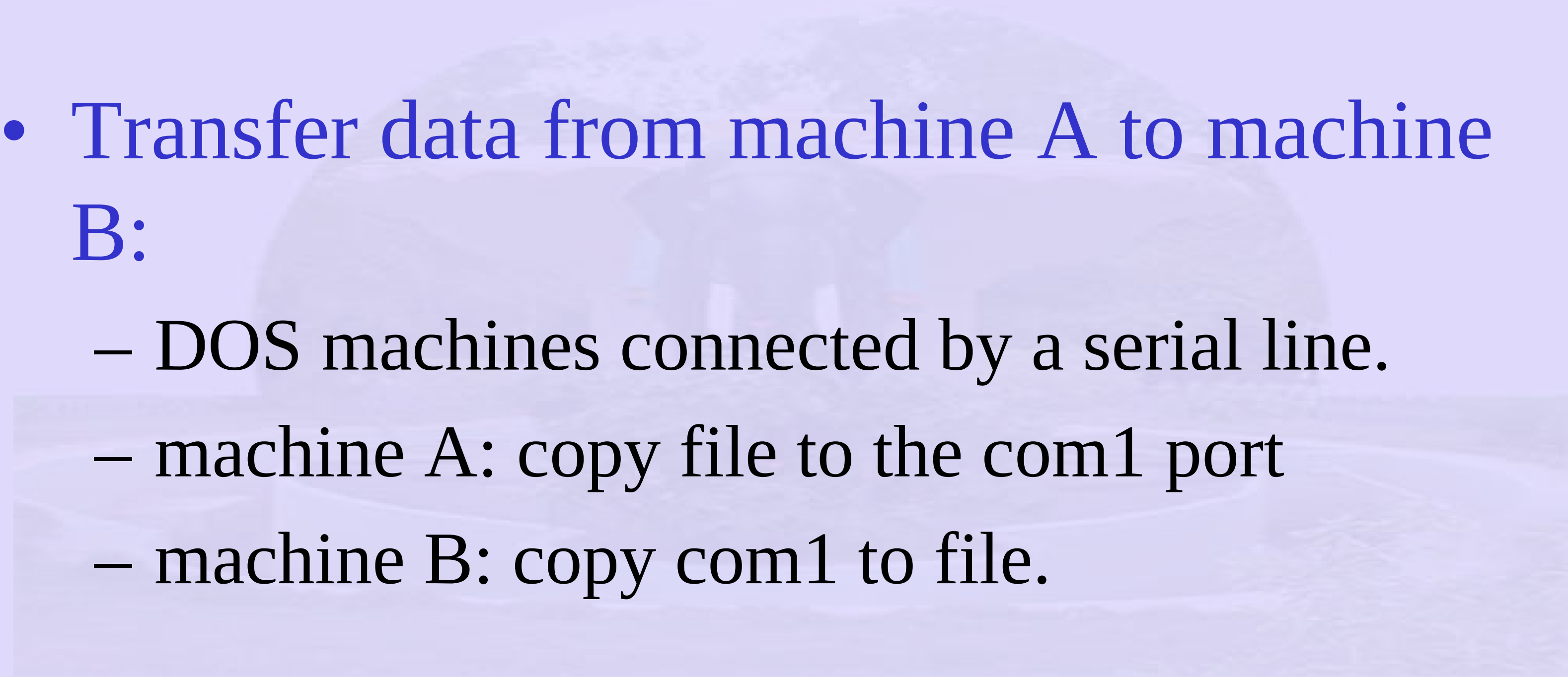
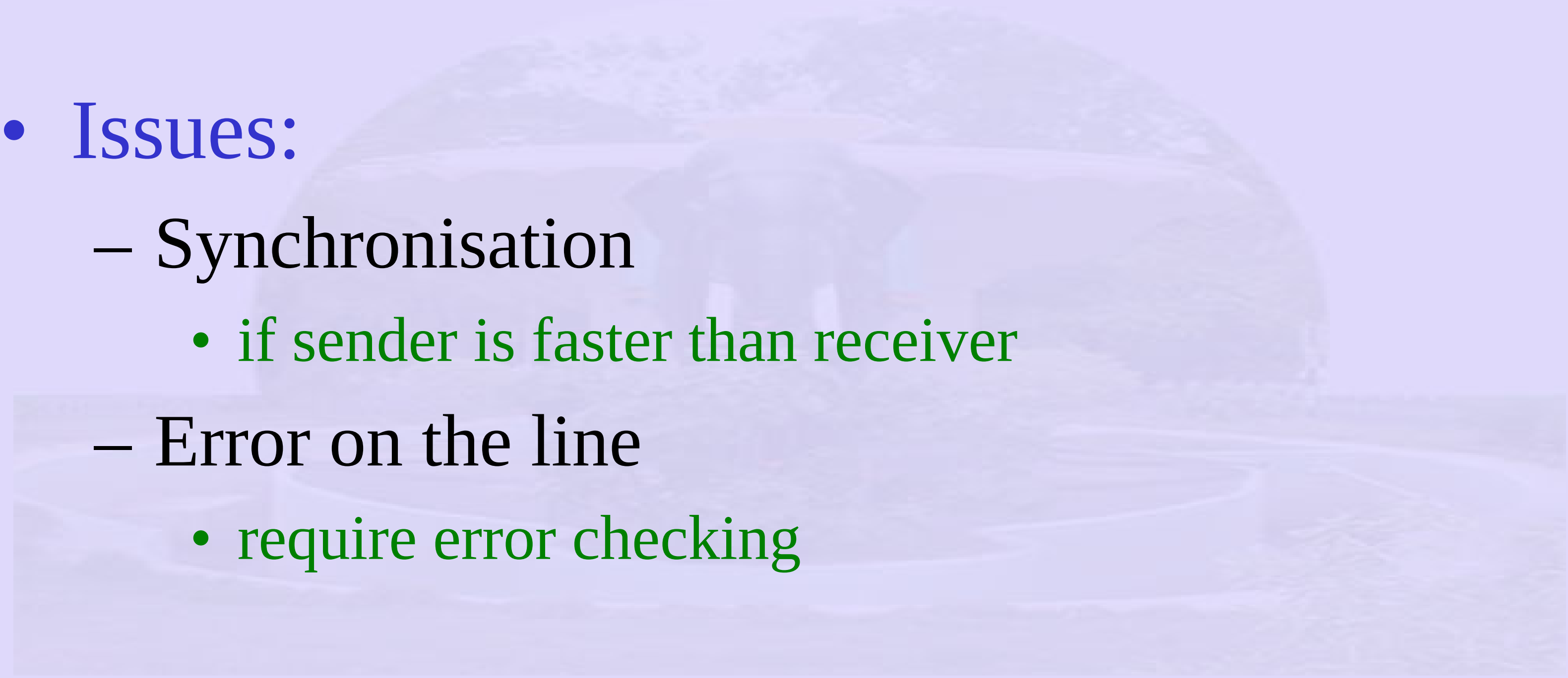
Sharing of Data on DOS machines



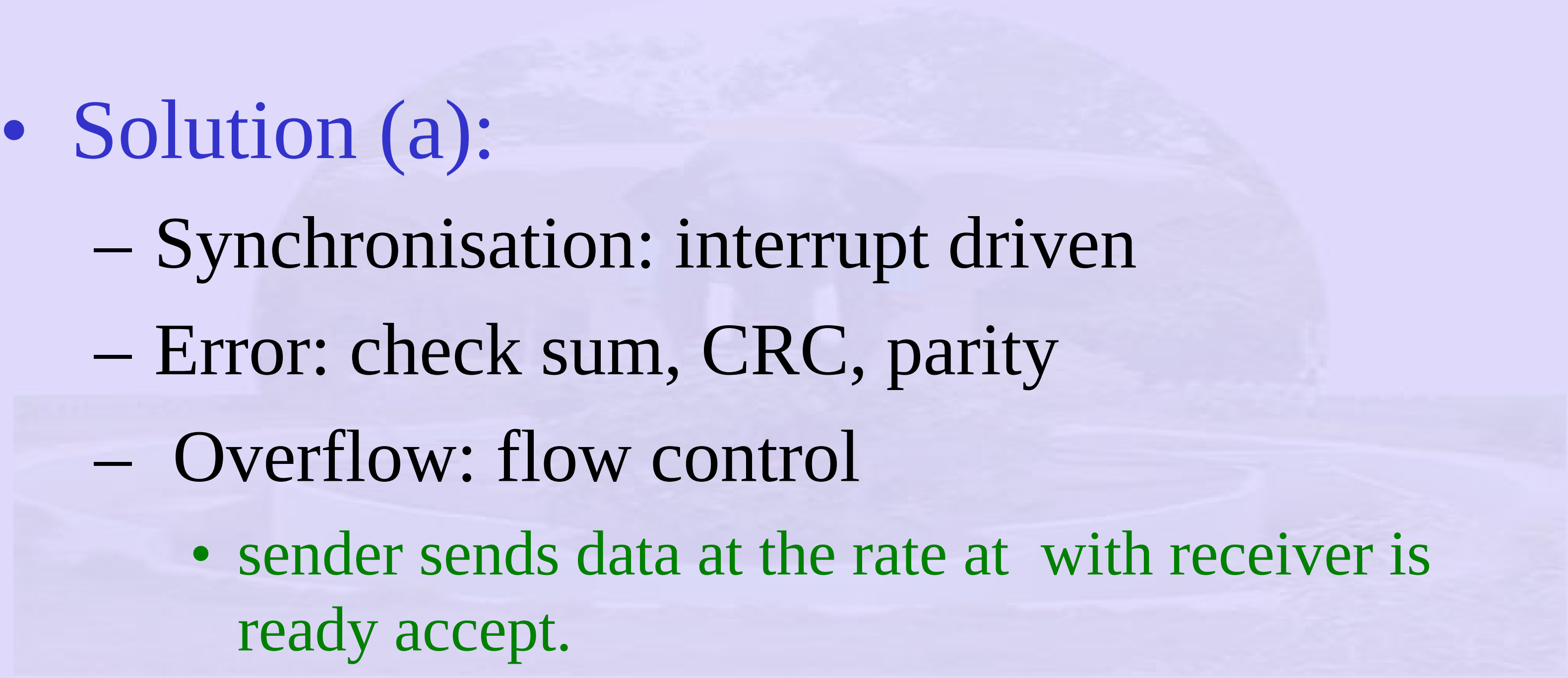
- Transfer data from machine A to machine B:
  - – DOS machines connected by a serial line.
  - – machine A: copy file to the com1 port
  - – machine B: copy com1 to file.

Sharing of Data on DOS machines



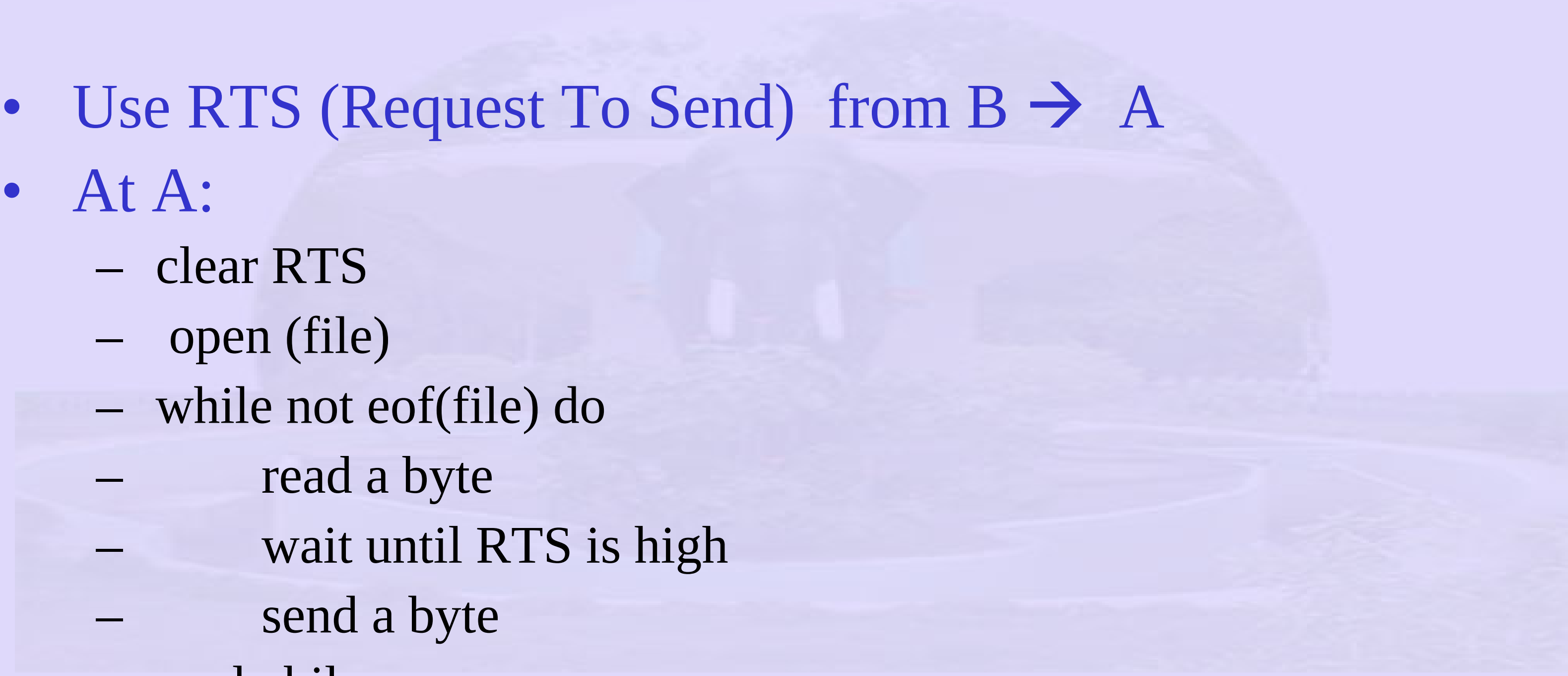
- Issues:
  - Synchronisation
    - if sender is faster than receiver
  - Error on the line
    - require error checking

A Solution



- Solution (a):
  - – Synchronisation: interrupt driven
  - – Error: check sum, CRC, parity
  - – Overflow: flow control
    - sender sends data at the rate at with receiver is ready accept.

Another Solution



- Use RTS (Request To Send) from B → A
- At A:

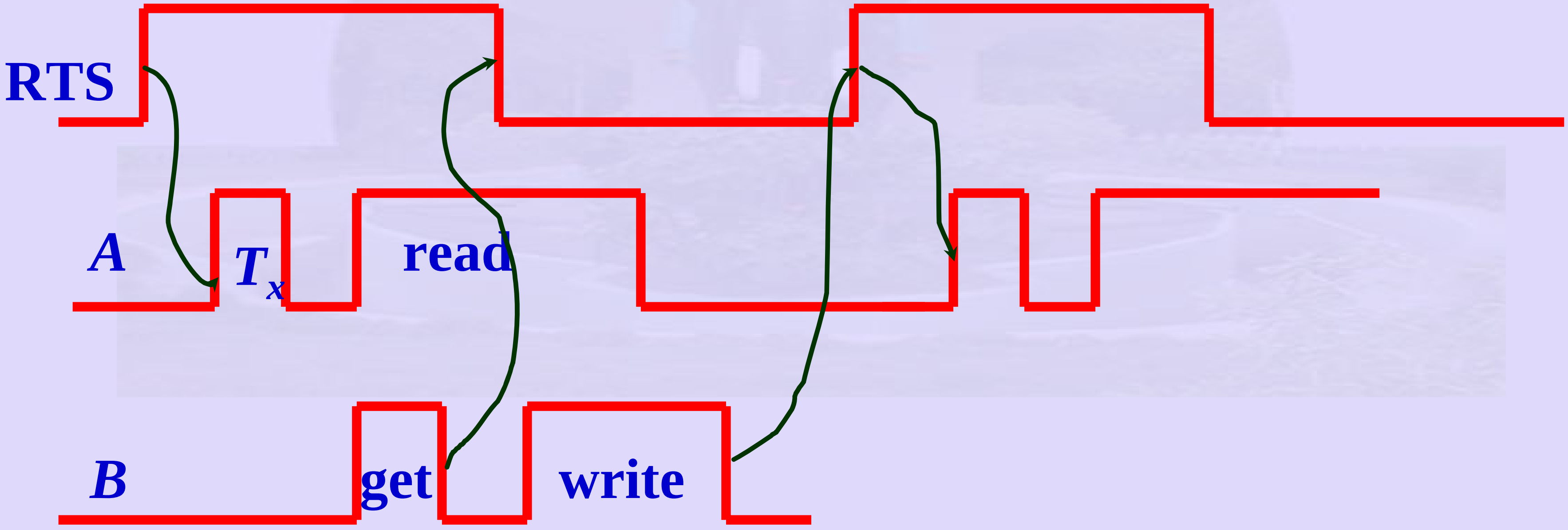
```
– clear RTS
– open (file)
– while not eof(file) do
–       read a byte
–       wait until RTS is high
–       send a byte
– endwhile
– send eof
– close(file)
```

Another Solution (contd)

- At B:

```
open(file)
repeat
    set RTS
    get a byte
    clear RTS
    write byte to file
until eof
close(file)
```

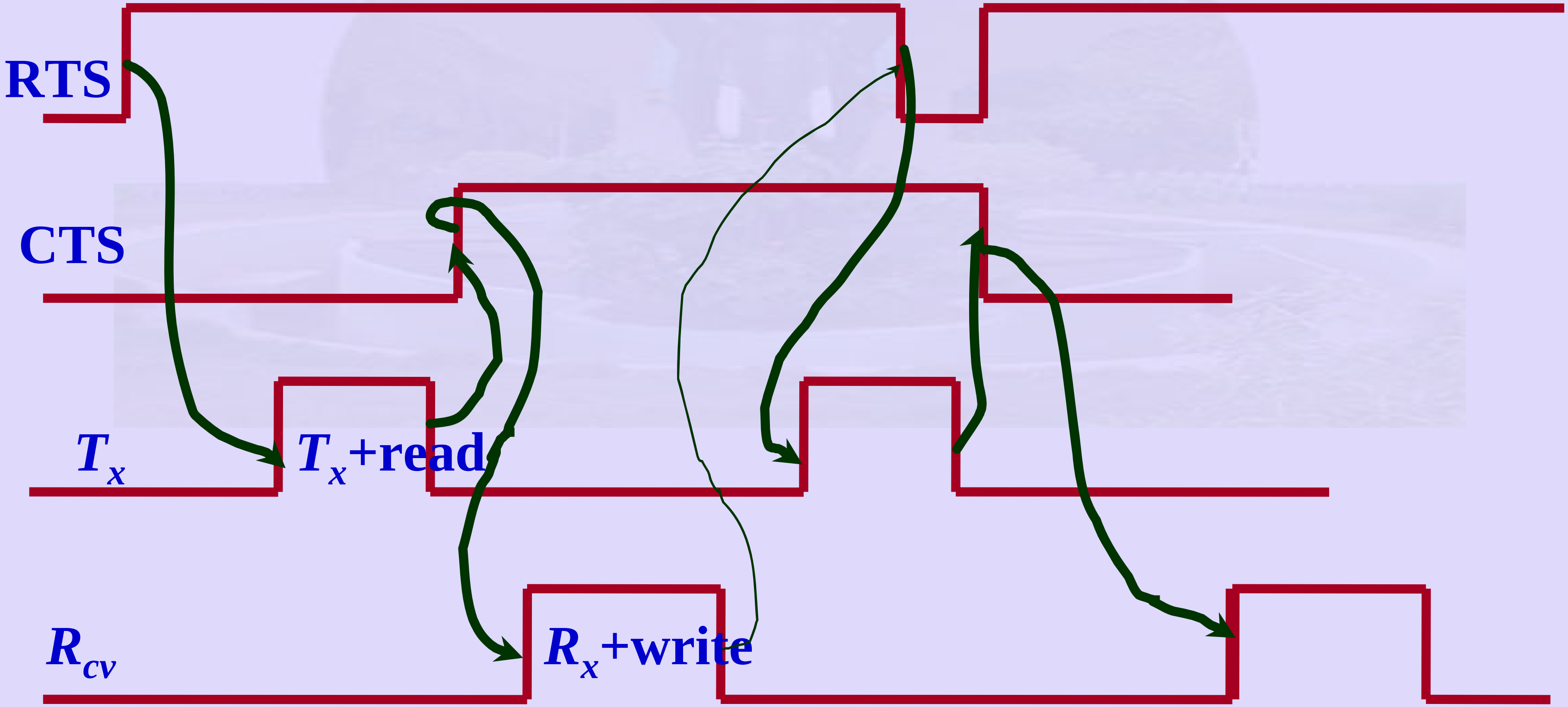
Data Flow Diagram using RTS



Issues

- If read at A is faster than get at B
  - read at A is completed before RTS is reset by B.
  - A will transmit another byte.
  - B will be swamped by A.
- One more signal is required:
  - RTS alone is not sufficient.
  - CTS (Clear To Send) A → B
  - RTS (Request To Send) B → A

Data Flow Diagram using RTS and CTS

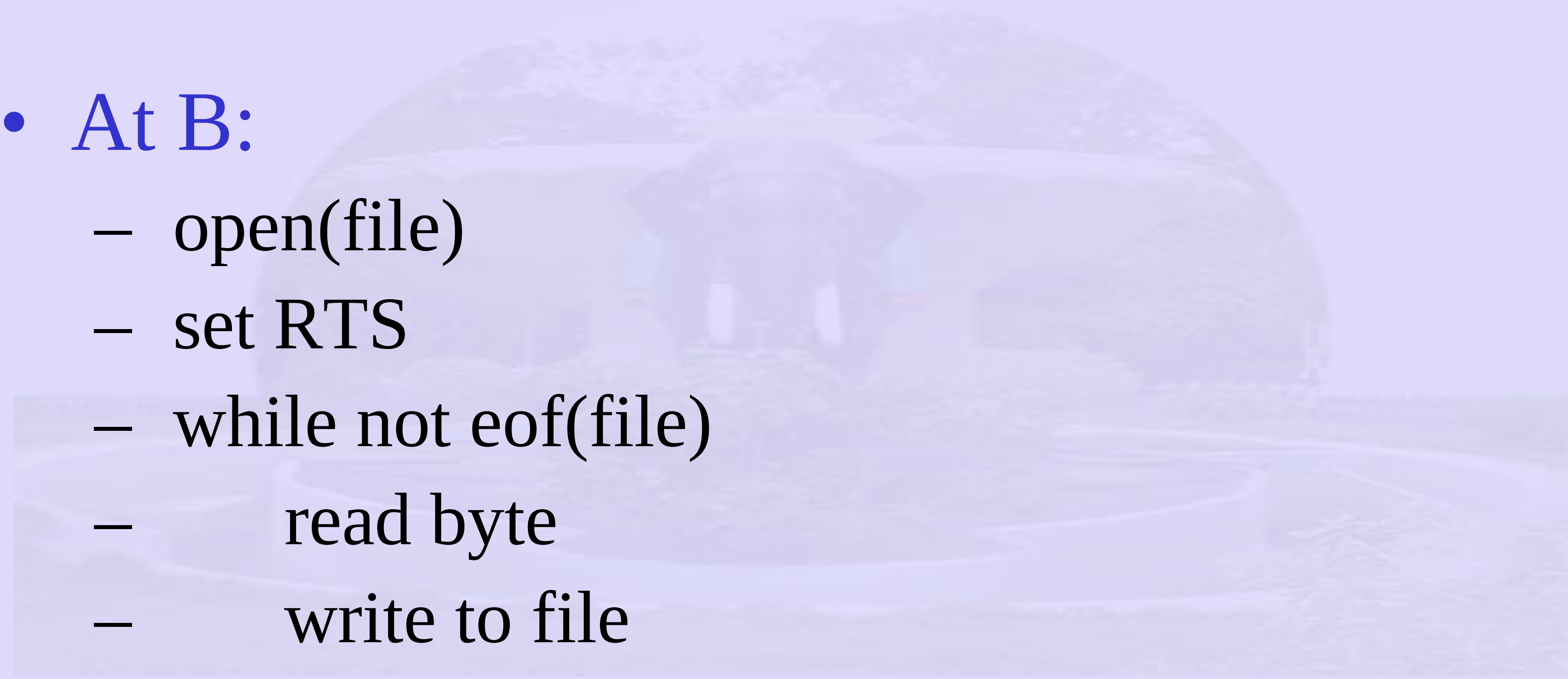


The Algorithm

- At A:

```
–   clear CTS
–   open(file)
–   wait for RTS to go High
–   while not eof(file) do
–         read byte
–         send byte
–         toggle CTS
–         wait for RTS toggle
–   endwhile
–   wait for RTS toggle
–   send eof
```

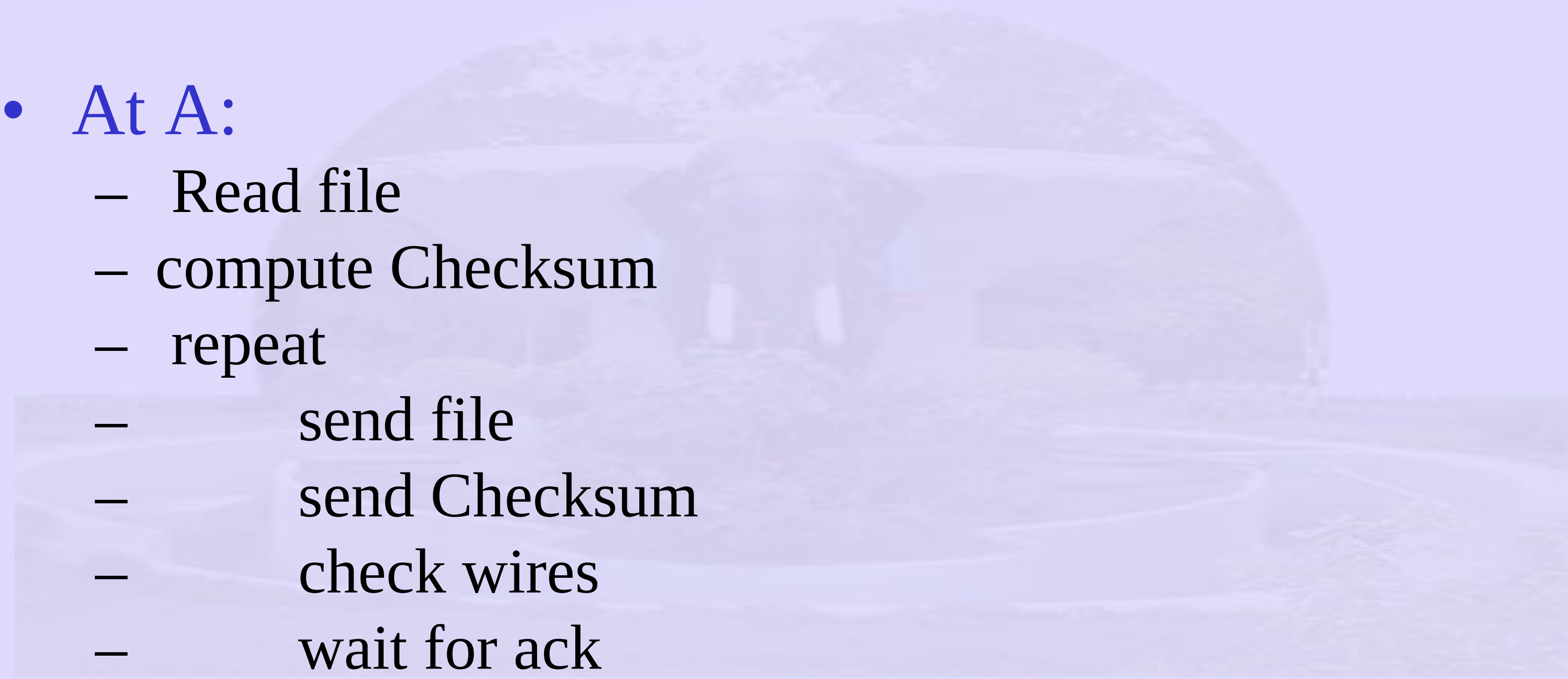
The Algorithm



- At B:

```
– open(file)
– set RTS
– while not eof(file)
–     read byte
–     write to file
–     toggle RTS
– endwhile
```

Error Control



- At A:
  - Read file
  - compute Checksum
  - repeat
  - send file
  - send Checksum
  - check wires
  - wait for ack
  - get ack
  - until ack
  - send finish

Error Control

- At B:

```
–   open(file)
–   while not (finish) do
–       get file
–       get checksum from A
–       compute Checksum from file received
–       compare the two
–       if same then send send acknowledgement
–   endwhile
```