ALBERT-LUDWIGS-
UNIVERSITÄT FREIBURG

Network Protocol Design and Evaluation

Exercise 1

Stefan Rührup

University of Freiburg
Computer Networks and Telematics

Summer 2009

CoNe
Freiburg

IIF
INSTITUT FÜR
INFORMATIK
FREIBURG

Exercise 1

1. *Clayton Tunnel protocol*
    Try to fix the Clayton Tunnel protocol so that the semaphore is not reset as long as there is a train in the tunnel.
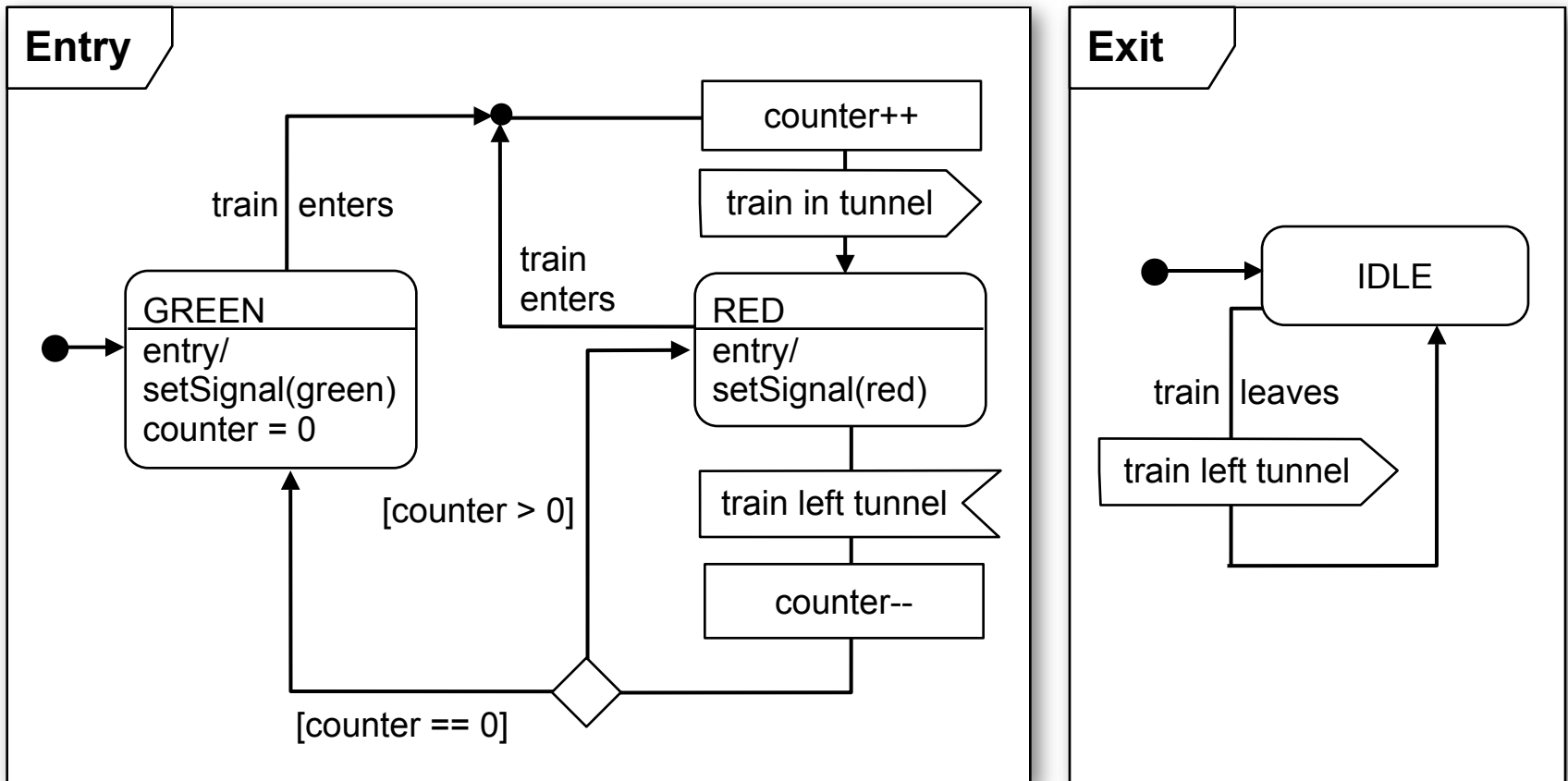
2. *Lynch's Protocol*
    Try to fix the duplication problem in Lynch's protocol. When can a character be accepted though the two preceding messages transmissions were erroneous?

The Clayton Tunnel protocol

Entry

Exit
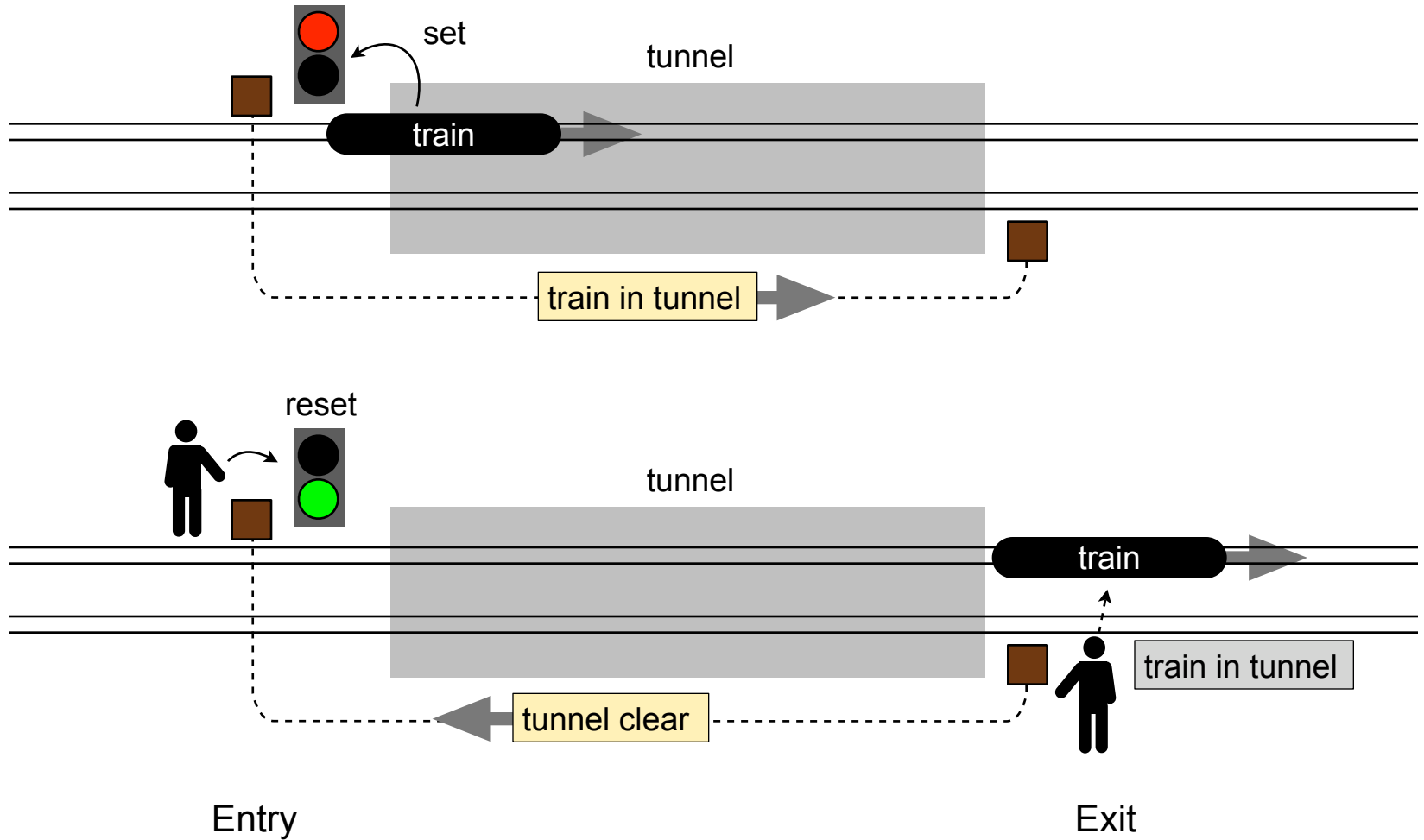
Exercise 1, Task 1
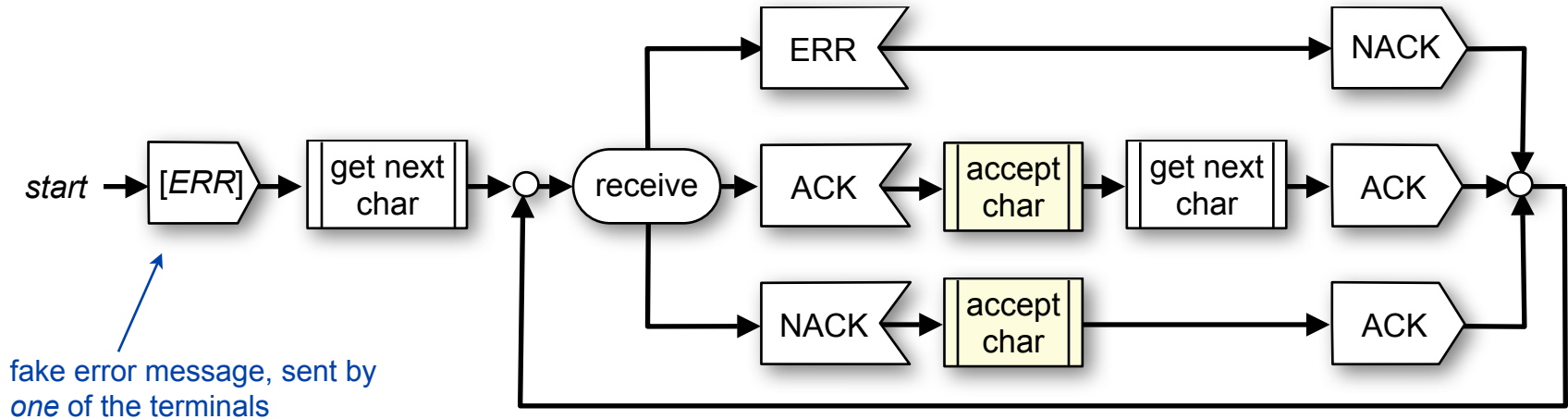
Idea: count the trains in the tunnel

Exercise 1, Task 1

‣ On the entry side the signalman maintains a counter for the trains that enter the tunnel

‣ The signal will be reset to green only if enough corresponding "train left tunnel" messages arrived.

‣ It covers the situation that a train enters the tunnel though the signal is red (or the signalman waves the red flag)

‣ Drawback: If a "train left tunnel" message gets lost, the counter will not reach 0 and the signal will stay red.
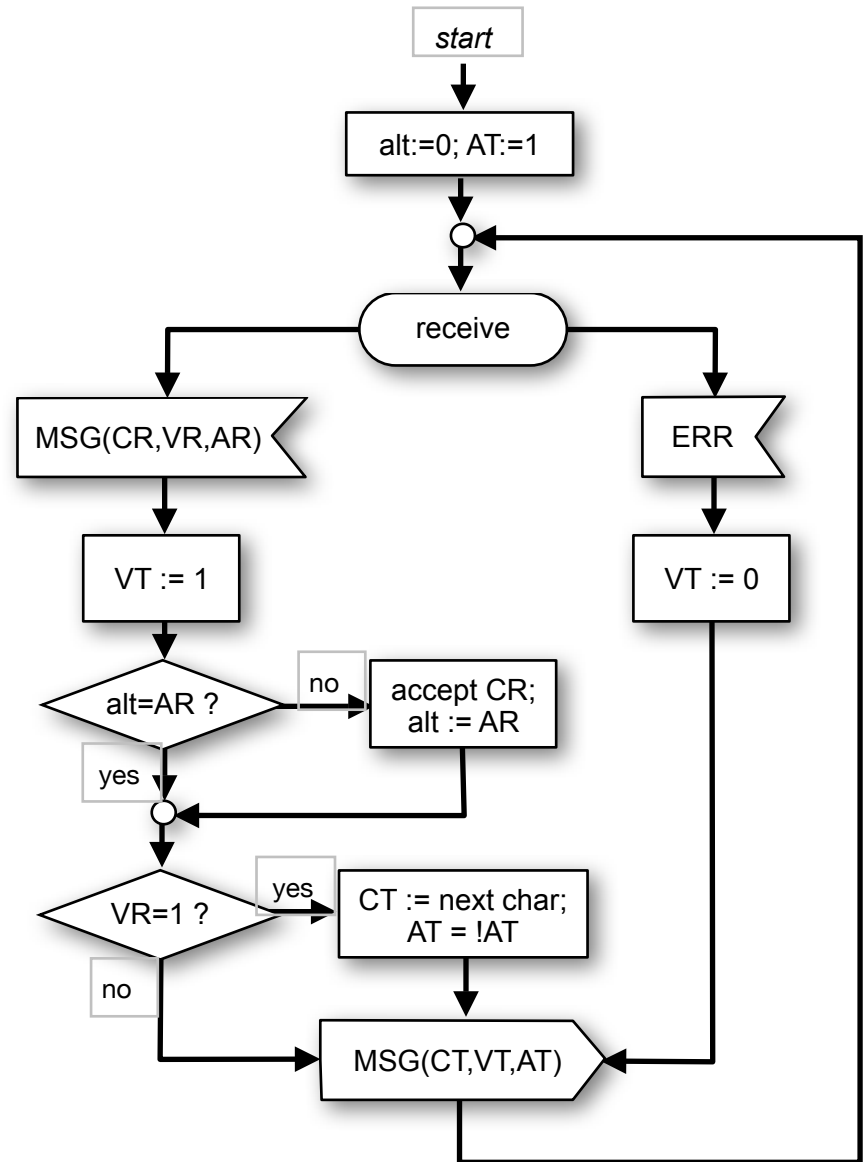
Exercise 1, Task 2: Lynch's Protocol (extended)

start → [ERR] → get next char → ○ → receive

- ERR ← NACK
- ACK ← accept char → get next char → ACK
- NACK ← accept char → ACK

fake error message, sent by *one* of the terminals

Example for the duplication problem:

	Terminal A			Terminal B		
	msg	error	ack	msg	error	ack
initiate	--	[ERR]	ACK			
				x	OK	**NACK**
accept x	a	ERR	(ACK)			
				x	ERR	(NACK)
	a	OK	**NACK**			
				x	OK	ACK
accept x						

ACK(x) is lost
retransmit x
accept a

Exercise 1, Task 2

Solution with verify bit and alternating bit
[Lynch 1969]

MSG(contents) = message(contents)
ERR = error message
CR = character received
CT = character to transmit
VR = verify bit (received)
VT = verify bit (to transmit)
AR = alternating bit (received)
AT = alternating bit (transmitted)

start

alt:=0; AT:=1

receive

MSG(CR,VR,AR)

ERR

VT := 1

VT := 0

alt=AR ?

no

accept CR;
alt := AR

yes

VR=1 ?

yes

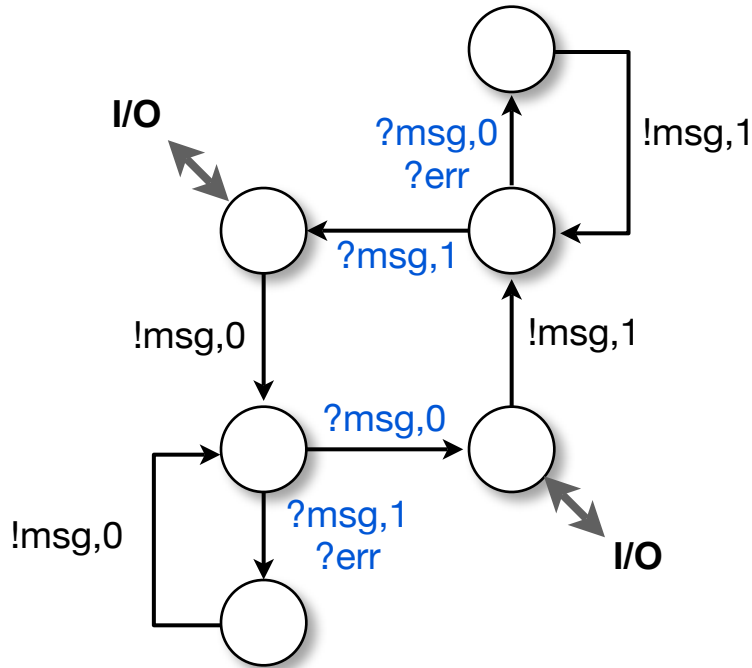CT := next char;
AT = !AT

no

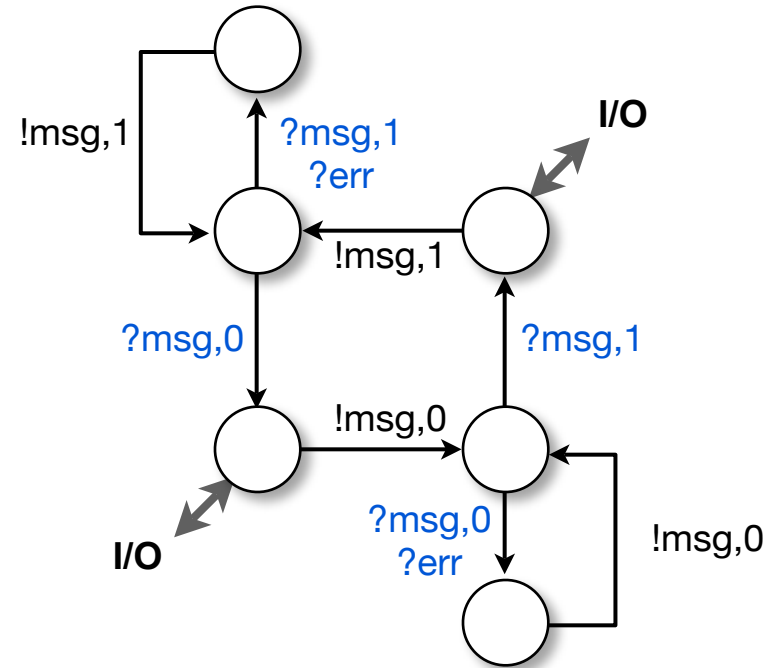MSG(CT,VT,AT)

Exercise 1, Task 2

The alternating bit protocol
[Bartlett, Scantleburst, Wilkinson 1969]

Notation: ! = send, ? = receive,
"msg,0" = message with 0 bit appended
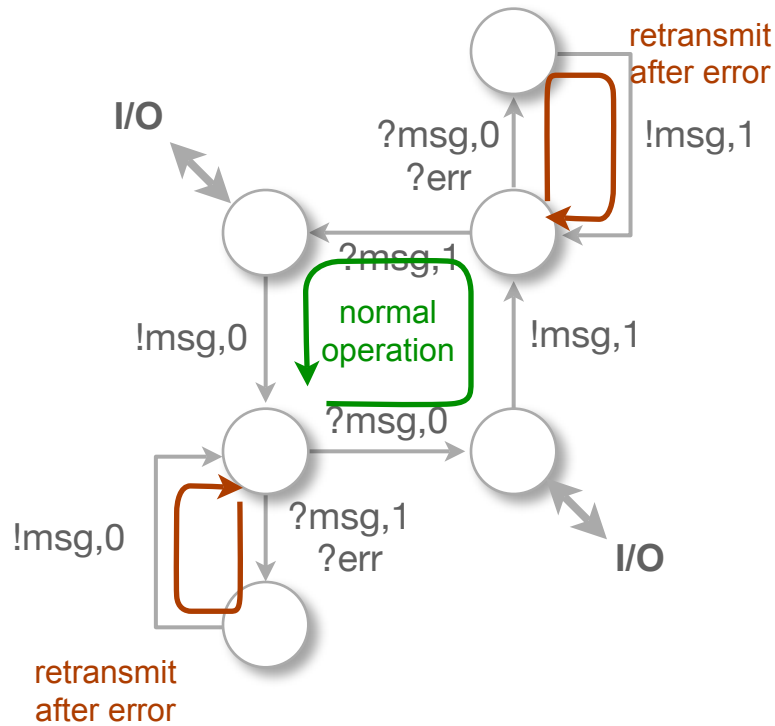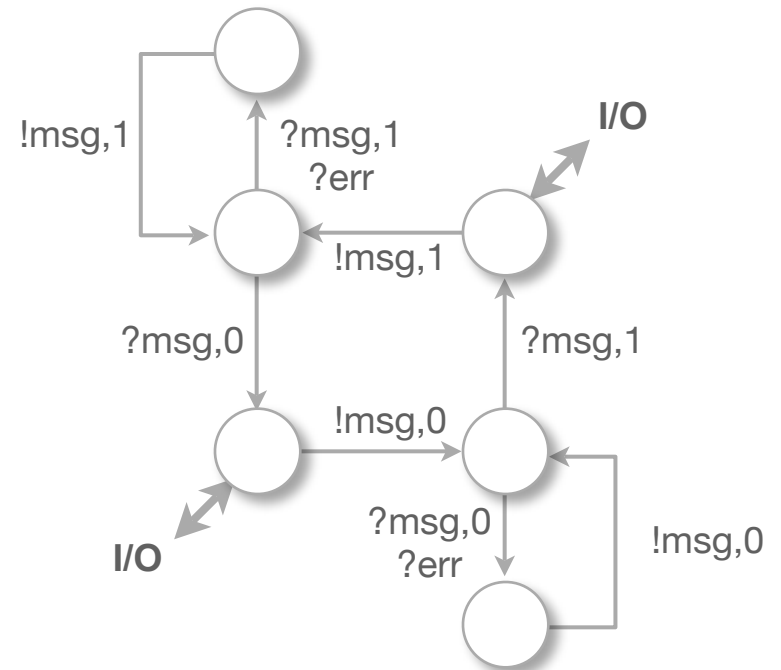I/O = accept message, fetch new message

Terminal A

Terminal B

Exercise 1, Task 2

The alternating bit protocol
[Bartlett, Scantleburst, Wilkinson 1969]

retransmit
after error

?msg,0
?err

!msg,1

I/O

?msg,1

normal
operation

!msg,0

!msg,1

?msg,0

!msg,0

?msg,1
?err

I/O

retransmit
after error

Terminal A

!msg,1

?msg,1
?err

I/O

!msg,1

?msg,0

?msg,1

!msg,0

I/O

?msg,0
?err

!msg,0

Terminal B