

1/12/93

SERIAL BUS PROTOCOL FUNDAMENTALS

GERALD MARAZAS

IBM, BOCA RATON, FLORIDA

(407) 982-4423 VOICE

(407) 982-0067 FAX

MARAZAS@BCRUMPC2.VNET.IBM.COM

NEXT SBP EDITOR'S MEETING

→ JAN 27 & 28, 1993 (WED. & THURS)
→ 9 AM - 5 PM EACH DAY
HOSTED BY SCOTT SMYERS
APPLE COMPUTER
CUPERTINO, CALIFORNIA

SERIAL BUS PROTOCOL

<> ELEMENTS REQUIRED IN:

- THE TARGET
- THE INITIATOR

<> THE COMMAND BLOCK CHAIN

<> THE TAP PROTOCOL FOR
COMMAND DELIVERY

<> THE "LOG-IN" PROCEDURE

<> REQUEST FOR TAP SLOTS

<> DEDICATED TAP SLOTS
VS POOL TAP SLOTS

<> THE "SIGN-IN" PROCEDURE

<> ASYNCHRONOUS EVENTS

<> FIRST FAILURE REGISTER

<> STATUS DELIVERY

<> ISOCHRONOUS

SERIAL BUS PROTOCOL

A TARGET HAS:

- <7 NORMAL FIFO
- <7 URGENT FIFO
- <7 ACA FIFO

.....
<7 A FIFO HAS TAP SLOTS

.....
CONFIGURATION ROM

<7 BASE ADDRESS OF THE FIFO
FIFO ADDRESS HAS THE STRUCTURE



<7 INITIATOR_ID = 8 bits,
ASSIGNED BY THE TARGET

.....
WORKING STORAGE TO HOLD
COMMAND BLOCKS

.....
OPTIONALLY .. ISOCRONOUS CONTROL REG.
.. FIRST FAILURE REGISTER

②

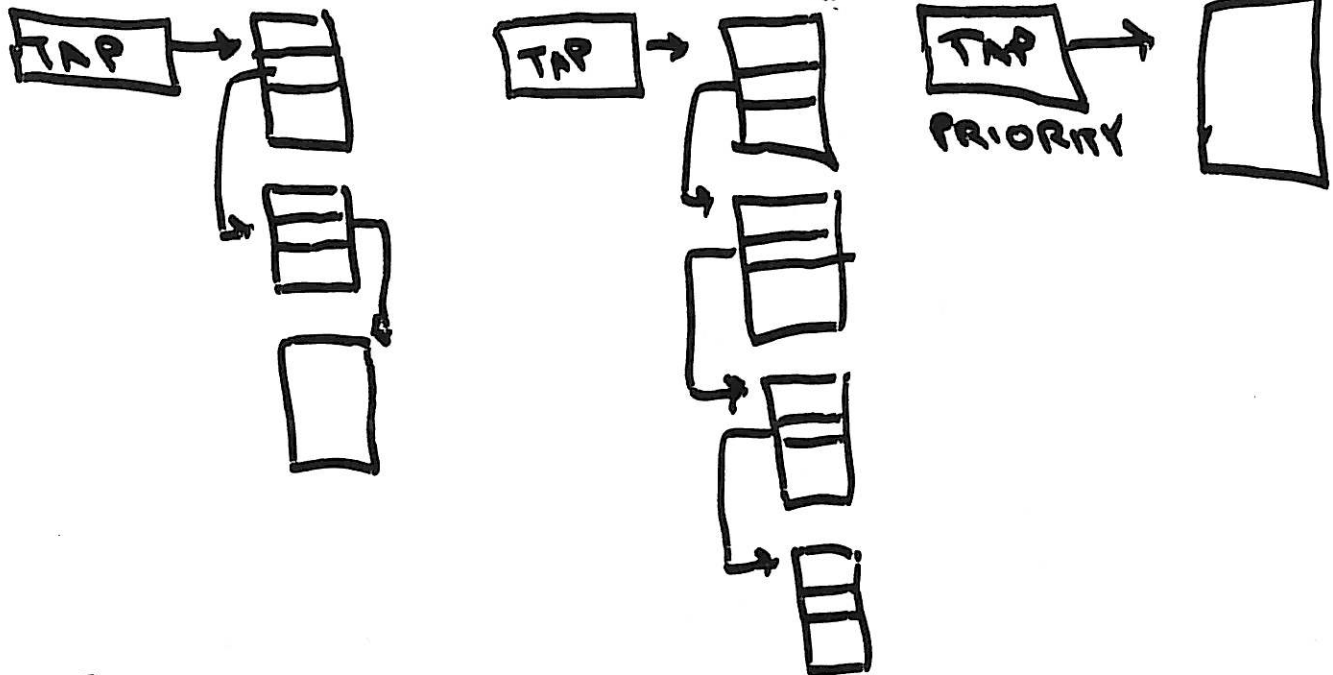
SERIAL BUS PROTOCOL

AN INITIATOR HAS:

- <7 STATUS FIFO
- <7 COMMAND BLOCK CHAINS
- <7 OPTIONALLY AN
ISOCHRONOUS CONTROL REGISTER

SERIAL BUS PROTOCOL

COMMAND BLOCK CHAINS



- <? THE INITIATOR MAY DIRECT MULTIPLE CHAINS TO THE SAME TARGET
- <? COMMAND BLOCKS IN THE SAME CHAIN:
 - MAY INCLUDE DIFFERENT QUEUE TYPES
 - MAY BE DIRECTED TO DIFFERENT LUNs at same Target
 - INDICATE FETCH POLICY



- <? EACH CHAIN REQUIRES 1 TAP SLOT IN THE TARGET
- <? A CHAIN MAY GO TO THE URGENT FIFO

SERIAL BUS PROTOCOL

COMMAND DELIVERY - PROTOCOL

INITIATOR

TARGET

SEND
TAP PACKET
(MUST HAVE
INIT-ID)
DEDICATED TAP SLOT: ?

STORE TAP PACKET
INTO A
TAP SLOT

SUPPLY FIRST
COMMAND BLOCK

REQUEST FIRST
COMMAND BLOCK
{ MORE ?
END SUB CHAIN ?
LINKED ?
HEAD OF QUEUE ?

SUPPLY
~~NEXT~~
COMMAND BLOCK

REQUEST NEXT
COMMAND BLOCK

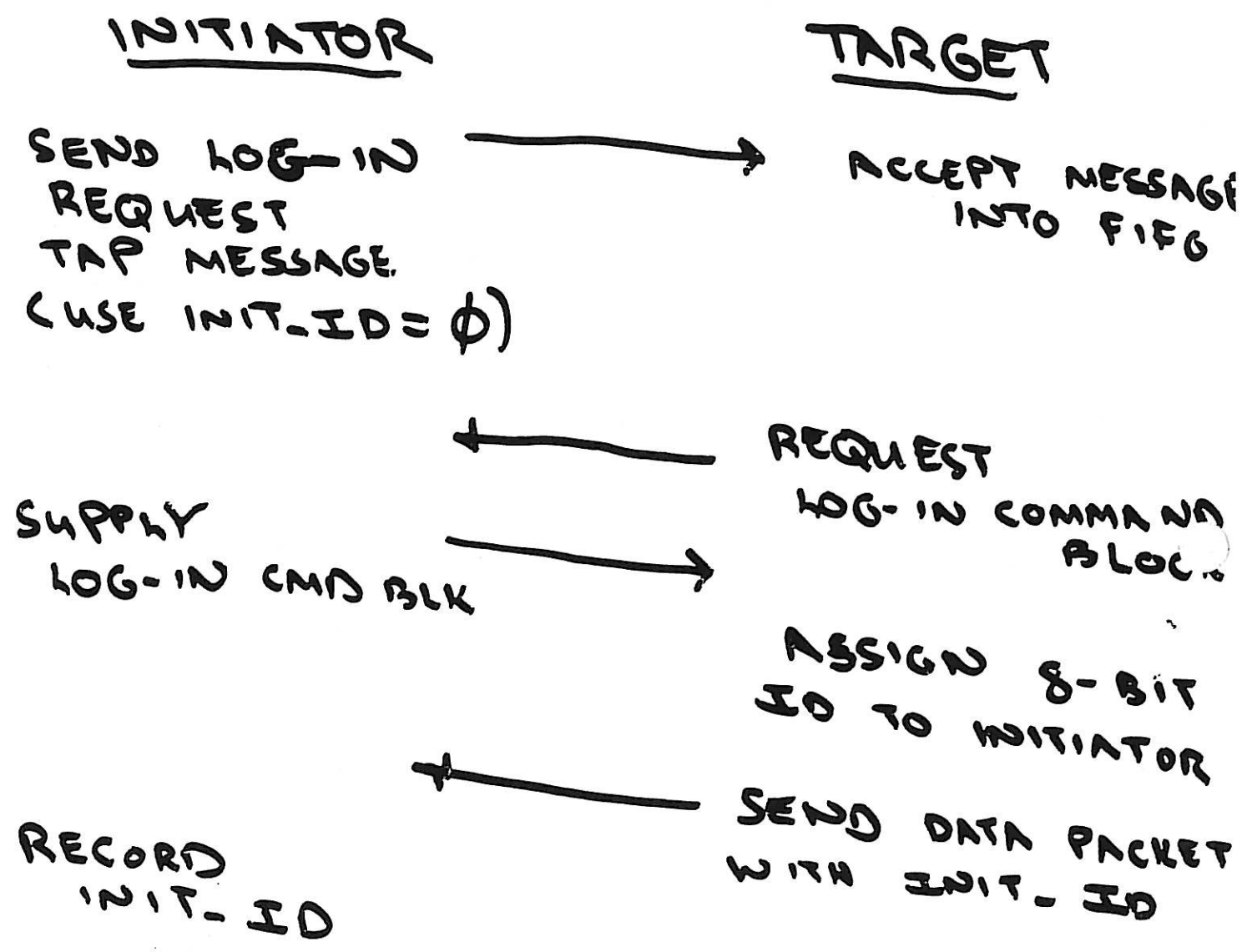
SUPPLY
LAST
COMMAND
BLOCK

REQUEST LAST COMMAND
BLOCK

LATEST TIME TO
RELEASE TAP SLOT

SERIAL BUS PROTOCOL

THE LOG-IN PROCEDURE



NOTE: THE SAME INITIATOR IS LIKELY TO HAVE A DIFFERENT INIT-ID FROM EACH TARGET.

SERIAL BUS PROTOCOL

REQUEST FOR TAP SLOTS

INITIATOR

TARGET

SEND TAP SLOT
REQUEST MSG
TO FIFO

USE INIT_ID

ACCEPT MESSAGE
INTO FIFO

REQUEST
TAP SLOT REQUEST
COMMAND BLOCK

SUPPLY
TAP SLOT REQUEST
COMMAND BLOCK

GATHER UP
TAP SLOTS TO BE
DEDICATED TO
THE INITIATOR

SEND DATA PACKET

USE ALLOCATION
OF DEDICATED TAP SLOTS
TO MANAGE
ENDING OF
TAP PACKETS

OLD # SLOTS
NEW # SLOTS

SERIAL BUS PROTOCOL

DEDICATED TAP SHOTS VS POOL TAP SLOTS

INITIATOR-A

TARGET

SLOTS
DEDICATED
TO INIT.A

POOL
SLOTS

..... 2 4

SEND TAP → MUST ACCEPT 1 4

SEND TAP → MUST ACCEPT $\emptyset$ 4

SEND TAP → MAY REJECT $\emptyset$ 3

SEND TAP → MAY REJECT $\emptyset$ 2

⋮
SEND TAP → MAY REJECT $\emptyset$ $\emptyset$

SEND TAP → DID REJECT $\emptyset$ $\emptyset$

↓ RECOVERY

..... $\emptyset$ 2

TRY TO
SEND TAP AGAIN → MAY ACCEPT

SERIAL BUS PROTOCOL

SIGN-IN PROCEDURE

INITIATOR

SEND SIGN-IN
REQUEST TAP
MESSAGE
USE INIT-ID

SUPPLY SIGN-IN
COMMAND BLOCK

TARGET

ACCEPT MESSAGE
INTO FIFO

REQUEST SIGN-IN
COMMAND BLOCK

RECORD AEN:
- STATUS FIFO @
- SENSE BUFFER @

AEN OCCURS

SEND STATUS TO
AEN STATUS FIFO

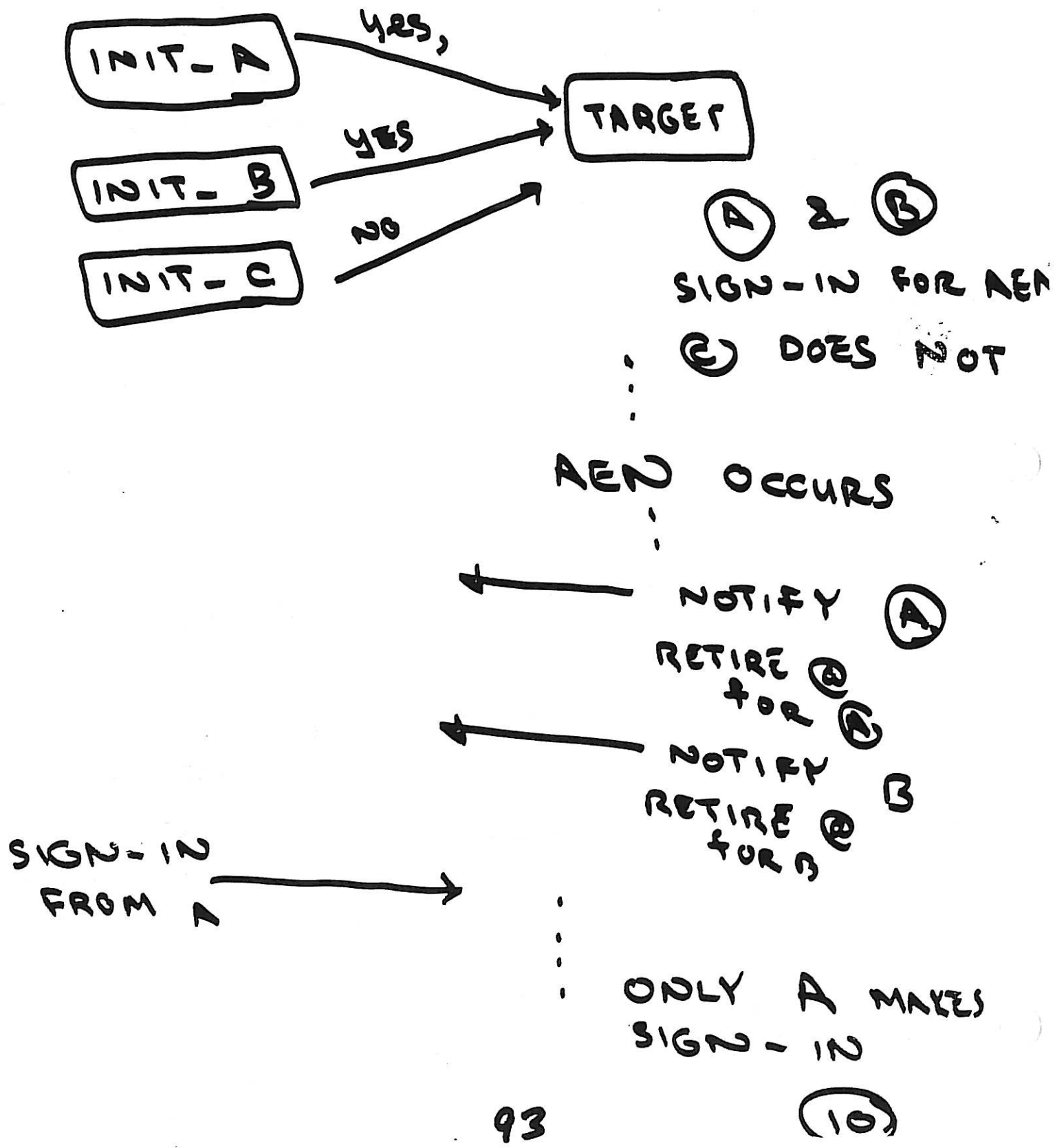
SEND SENSE DATA
TO SENSE BUF @

RETIRE STATUS FIFO
RETIRE SENSE BUF @

SEND "NEW"
SIGN-IN REQUEST TAP

SERIAL BUS PROTOCOL

ASYNCHRONOUS EVENTS



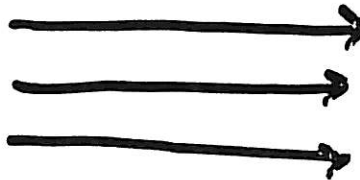
SERIAL BUS PROTOCOL

FIRST FAILURE REGISTER; FF-REG
FF-CNTL

INITIATOR-A

TARGET

SUPPLY
COMMANDS



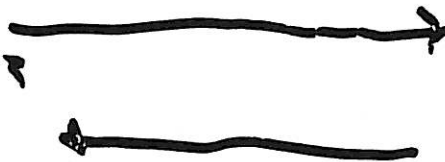
FF-REG UNLOCKE

FAILURE OCCURS

RECORD IN FF-RE
LOCK FF-CNTL

TIME OUT

READ REQUEST
TO FF-REG



SUPPLY CONTENTS OF
FF-REG.

UNLOCK REQ
TO FF-CNTL



TARGET UNLOCKS
FF-REG

SERIAL BUS PROTOCOL

STATUS DELIVERY

- ✓ EACH COMMAND BLOCK CONTAINS : - STATUS FIFO @
- SENSE BUFFER @
- ✓ INITIATOR MAY USE UNIQUE STATUS FIFO @ IN EACH COMMAND BLOCK
- ✓ INITIATOR MAY USE SAME STATUS FIFO @ FOR SEVERAL COMMAND BLOCKS
- ✓ WRITE OPERATION TO STATUS FIFO MAY OR MAY NOT CAUSE INTERRUPT TO THE INITIATOR
- ✓ INITIATOR MUST RECEIVE STATUS BLOCK FROM THE TARGET
- ✓ INITIATOR MAY TELL TARGET NOT TO SEND A STATUS BLOCK

SERIAL BUS PROTOCOL

ISOCHRONOUS

- PORT-SETUP COMMAND BLOCK
(NO CDB)
LOTS OF PARAMS
- PORT-APPEND COMMAND BLOCK
(CARRIES A CDB)
→ ENABLES ISO XFER,
DOES NOT CAUSE IT!
- WRITE OPERATION
TO ISOCRONOUS CONTROL REGISTER
START ISO XFER
PAUSE ISO XFER
- PORT-TENR DOWN COMMAND BLOCK
(NO CDB)