

P1394b
Maui - October 24th 1997

P1394b Start-up

Connectivity Management Task Group
(*"upstarts"*)

upstarts@zayante.com

Introduction

- ▢ Scope: start-up procedure for Beta-capable links
 - when a port becomes connected, this procedure has to decide whether to start up in P1394a mode or Beta mode
 - for Beta mode, the procedure also determines the operating speed and starts up the scrambler
- ▢ Arbitration state machine operation (bus reset, tree-ID, self-ID and normal operation) will follow AFTER this start-up procedure
- ▢ Start-up in Beta mode if
 - both ends are Beta capable
- ▢ Corollary: start-up in DS mode if
 - one end is only DS capable
- ▢ Develop appropriate modifications to P1394a state machines (build on the S/R mechanism), C code etc
- ▢ Ensure interoperability with both P1394a and 1394-1995

Port properties - 1

a P1394b port

- may be capable of operating at any speed range from S100 upwards
- may be capable of operating at any speed range from S800 upwards
- shall be capable of operating in Beta mode
- when capable of operating at S100, shall be capable of DS at S100-S400, may be capable of Beta mode at S100-S400
- may be brought out to a 1394-1995 connector ONLY when capable of operating in DS mode
- NB improved connector electrical properties are required for S800
- open issue - connector for S1600 and S3200
- may be connected directly to a suitable transceiver for long haul connection (thus also providing DC isolation)
- may use DC (e.g. capacitive or galvanic) isolation when operating in Beta mode

 this is also the "short-haul" port on a P1394b sub-PHY

Port properties - 2

The "long-haul" port on a P1394b sub-PHY

- may be capable of operating at any single speed (S100 - S3200)
- may be capable of operating at any speed range from S100 upwards (i.e. S1, or S1+S2, or S1+S2+S4, etc)
- may be capable of operating at any speed range from S800 upwards
- operates in Beta mode only
- is not required to interoperate with a 1394-1995 or P1394a port
- shall not be brought out to an interface using a 1394-1995 connector
- connects through a suitable transceiver (this may be integrated), galvanic isolation and a RJ45 connector for S100 UTP
- connects through a suitable transceiver and, optionally, a suitable optical connector for optical fibre transmission
- "understands" the speed limitations of the connected long-haul transceiver by an implementation dependent mechanism

Sub-PHY long-haul port

 A subPHY long haul port is essentially a stripped down 1394b port.

It's a 1394b port:

- without DS
- 1394 connector forbidden
- may operate at only one speed (S100 through S3200), or a range of speeds
 - ✓ provided the range includes either S100 or s800 or both (for start up)
- speed(s) dictated by attached transceiver.

Port types and properties

S100* DS

- a port as defined in 1394-1995/1394a
(S100* means S100, or S100+S200, or S100+S200+S400)

S100* DS/Beta

- as 1) but also capable of Beta mode (not necessarily all Beta speeds)

S100* Beta

- Beta mode only at S100* i.e. a long haul only port for slower speeds
- cannot be brought out to a 1394 connector

S100* DS; S800* Beta

- as 1) but also capable of operation in Beta mode only at higher speeds (\geq S800)

S100* DS/Beta; S800* Beta

- as 4) but also capable of beta mode at S100*
- this is the bells and whistles port

S800* Beta

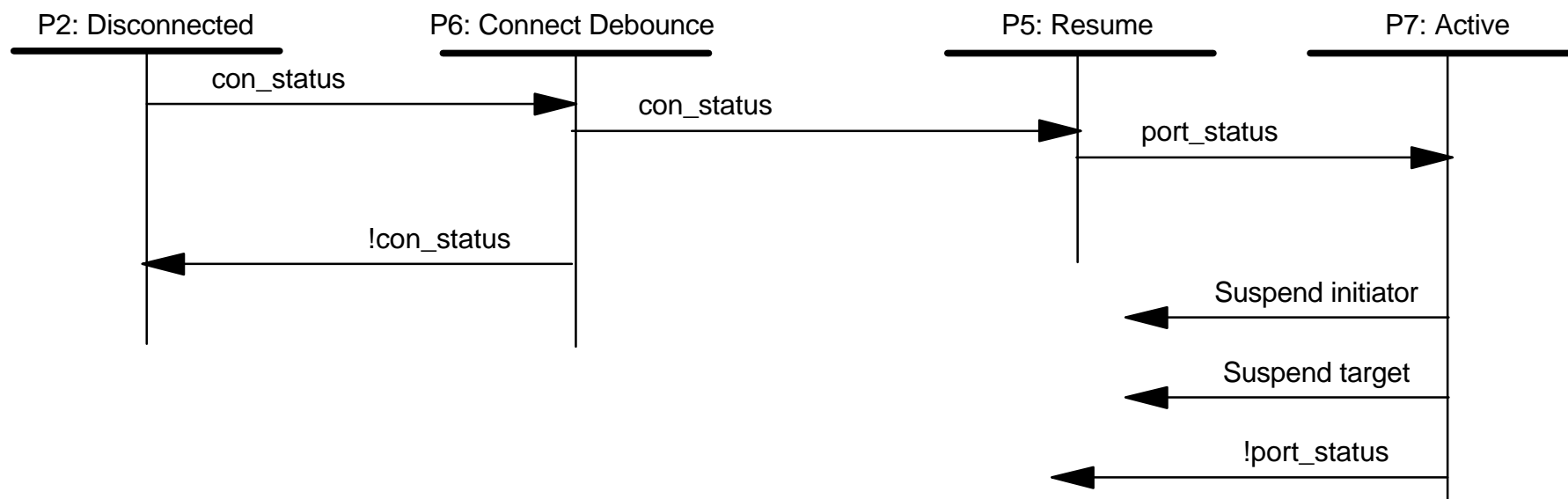
- only operates at speeds of S800 upwards, and does not support the 1394-1995 signalling interface (i.e. long haul for higher speeds)
- cannot be brought out to a 1394 connector

★ The above does not fully comprehend single-speed Beta ports

Interoperability table

	S100* DS	S100* DS/Beta	S100* Beta	S100* DS; S800* Beta	S100* DS/Beta; S800* Beta	S800* Beta
S100* DS	yes DS	yes DS	no	yes DS	yes DS	no
S100* DS/Beta		yes Beta	yes Beta	yes Beta	yes Beta	no
S100* Beta			yes Beta	no	yes Beta	no
S100* DS; S800* Beta				yes Beta	yes Beta	yes Beta
S100* DS/Beta; S800* Beta					yes Beta	yes Beta
S800* Beta						yes Beta

P1394a Suspend/Resume State Machine (part, simplified)



State	Do	Then wait for
P2: Disconnect	TpBias = Z Turn off port	con_status
P6: Connect Debounce	TpBias = Z (bug?) Turn on port do reset if none seen	CONNECT_TIMEOUT or 2 * CONNECT_TIMEOUT incoming reset
P5: Resume	TpBias = on	incoming TpBias (port_status)
P7: Active	Normal operation	suspend command or TpBias going away

P1394b Eager Beta/Lazy a Algorithm for P2: Disconnected

- Immediately start well-spaced (low power) pulse-toning on TPB, listen on TPA (details later)
- if we hear the tone, then we know we're Beta mode
 - set Beta mode and go to P6: debounce
- if we don't hear a tone, then, if bi-lingual, look for con_status
 - if set, then set DS mode and go to P6: debounce, otherwise repeat toning
- Following table gives the parameters that a bi-lingual or Beta mode only port will detect during P2: Disconnected

Type of connection	con_status	tone exchange	action
No connection	no	fails	stay in P2:
DC connection to P1394a	yes	fails	set DS
DC connection to bi-lingual	yes	succeeds	set Beta
DC connection to Beta only	yes	succeeds	set Beta
AC connection to bi-lingual	no	succeeds	set Beta
AC connection to Beta only	no	succeeds	set Beta
DC connection to optical transceiver (but no end-to-end connection) <i>(outlawed by mandating AC coupling)</i>	yes?	fails	want to stay in P2, so OUTLAW THIS
AC connection to optical transceiver (but no end-to-end connection)	no	fails	stay in P2

Notes on Eager Beta for P2: Disconnected

Transmit tone on TPB, receive on TPA

- TPA remains high impedance
- a 1394-1995 node at the far end will assert TpBias on its TPA,
 - ✓ but will not see TpBias on its TPB, and so will not think it is connected
 - ✓ will therefore ignore anything received on its TPA
- a P1394a node at the far end will sense con_status
 - ✓ start its debounce timeouts
 - ✓ Eager Beta will not see a tone, and so will also go into Lazy a
 - ✓ need to ensure that we don't wait for a tone longer than a CONNECT_DEBOUNCE time

DC connection to an optical transceiver can result on con_status being set

- false impression of DC connection to a P1394a or 1394-1995 node
- con_status is sensed on TPA, which will be connected to the optical transceiver's receiver
- problem only for bi-lingual port

Rule: Optical transceiver running the risk of being connected to a bi-lingual port must provide an AC coupled interface on its receiver

- normally a couple of external capacitors
- may also be needed on the transmitter, depending on transceiver specification

Eager Beta for P6: Connect debounce

- ▢ If in Beta mode, continue sending tones
 - power up the port
 - wait CONNECT_DEBOUNCE
 - if no tones then received, false connection, go back to P2: Disconnected
 - exchange timed tones to determine operating speed
 - ✓ by this time, the port is powered up, so we can use the normal accurate timer
 - start transmitting Idle at the agreed speed
 - synchronise scrambler on incoming Idle
 - acquire code-word synch on incoming Idle
 - set short arbitrated or normal reset as appropriate
 - ✓ possibly wait another CONNECT_DEBOUNCE
 - go to P5: Resume with the link able to exchange control and data symbols

Lazy-a for P6: Connect debounce

- ▢ If in DS mode, power up the port and apply a modified form of the P1394a algorithm
- ▢ Lazy-a algorithm
 - Far end must be DS only, or else Eager Beta would have detected Beta mode
 - ✓ but not always, corner case of the user making a DC connection just after giving up toning
 - Far end will have asserted TpBias if it is 1394-1995 node
 - Far end will assert TpBias straight away if it is a p1394a node
 - ✓ (possible bug in current S/R actions?)
 - Wait for TpBias on TPB for 2 x bias hold notify??
 - ✓ generate TpBias in response
 - ✓ if not go back to P2: Disconnected and Eager Beta
 - wait CONNECT_DEBOUNCE
 - see con_status for longer than ping tone time
 - ✓ if not go back to P2: Disconnected and Eager Beta
 - set short arbitrated or normal reset as appropriate
 - ✓ possibly wait another CONNECT_DEBOUNCE
- ▢ Rule: Beta capable devices never initiate the assertion of TpBias
 - ✓ catches the corner cases

P1394b Eager Beta/Lazy a Algorithm for P5: Resume

Beta mode - if not already running (come from Connect_Debounce)

- exchange timed tones to determine operating speed
 - ✓ by this time, the port is powered up, so we can use the normal accurate timer
- start transmitting Idle at the agreed speed
- synchronise scrambler on incoming Idle
- acquire code-word synch on incoming Idle

Beta mode - now the port is running

- set port_status and go to P7: Active
 - ✓ port_status is now a logical signal, not a direct copy of TpBias
- loss of continuous operation at any time sets port_status to false
- but allow for possible attempt to resynchronise without going all the way back to P2: Disconnected?
 - ✓ perhaps should only set port_status to false when resynchronisation attempts fail?

DS mode - Lazy-a again (modified P1394a algorithm)

- Wait for TpBias on TPB for 2 x bias hold notify??
 - ✓ generate TpBias in response
 - ✓ if not go back to P2: Disconnected and Eager Beta
- TpBias is mirrored in port_status, go to P7: Active

Pulse-tone low power connect detect

- Use a periodic burst of signals during P2: Disconnected
 - send a tone for B microseconds, followed by S microsecs of silence
 - ✓ $B = S/500?$
 - receiver's 'Connected' is true if any transitions are detected within B+S usecs.
 - ✓ signal does not have to be recovered - just the presence of transitions detected (ie no PLL or decoding at this point in time).
- How long does B need to be to activate an optical transmitter, AC coupling etc?
- Does it conserve enough power?
 - tone frequency needs to be above pass-band filter cut-off, say 5MHz, but not so high as to cause EMC problems
 - Receiver needs to detect transitions. Both tx and rx need to have (share) a crude timer.
- How long can S be before there is the possibility of a disconnect/reconnect occurring during the interval S?
 - quite long for human controlled disconnects,
 - electronic switches are outlawed!

Issues list - 1

- How to handle the extra error conditions (sync fail etc)?
- Do the other state machines need to know whether we're in Beta or not?
- Is there a need for a software override to force P1394a mode or to force Beta mode?
 - should it be possible to have software force the port back though P2: Disconnected in order to reconnect using the "other" mode, or to change the line rate, or something
- Is there a need for a hardware configuration (strap) option to force one or the other mode?
- What level of software observability is required?
 - line rate
 - Beta mode or P1394a mode
 - in P7 or not, or state of state machine?
 - more detail (e.g. loss of synch flag)?
- What higher level of control may be required, e.g.
 - try to resynch coz there's too many 8B10B error codes
- Is it allowed for some ports to be P1394a only, others bilingual, others Beta only?

Issues - 2

- What is the lowest signal frequency that can be sent via an optical transceiver simply to exercise a connect/disconnect mechanism?
 - in discussion with Del Hanson (HP) and Ron Soderstrom (IBM)
- Connect_debounce state: don't understand exactly why connected can be true when con_status is false, and vice versa - there's an interaction with reset_detected(). Should we be looking for reset this early in determining whether we are P1394a or Beta?
- How can we send pings or something through a differential pair before TpBias is asserted. Have to provide our own bias.
 - Jim Doyle and Mike Brown working on this problem!
- Can we design a tone or ping system which does not require the receiver to be operational all the time?
 - ditto
- How else can we get to a very low power state?
- Lazy a has a bug - does not work if the "other" end gets into P8: Resume failed
 - we still have to work through all the states anyway