Tesla Transport Protocol (TTP) Specification

Rev 2.0

9/13/2024

Eric Quinnell

Douglas Williams

Christopher Hsiong

Gerardo Navarro Hurtado

William Lemaire

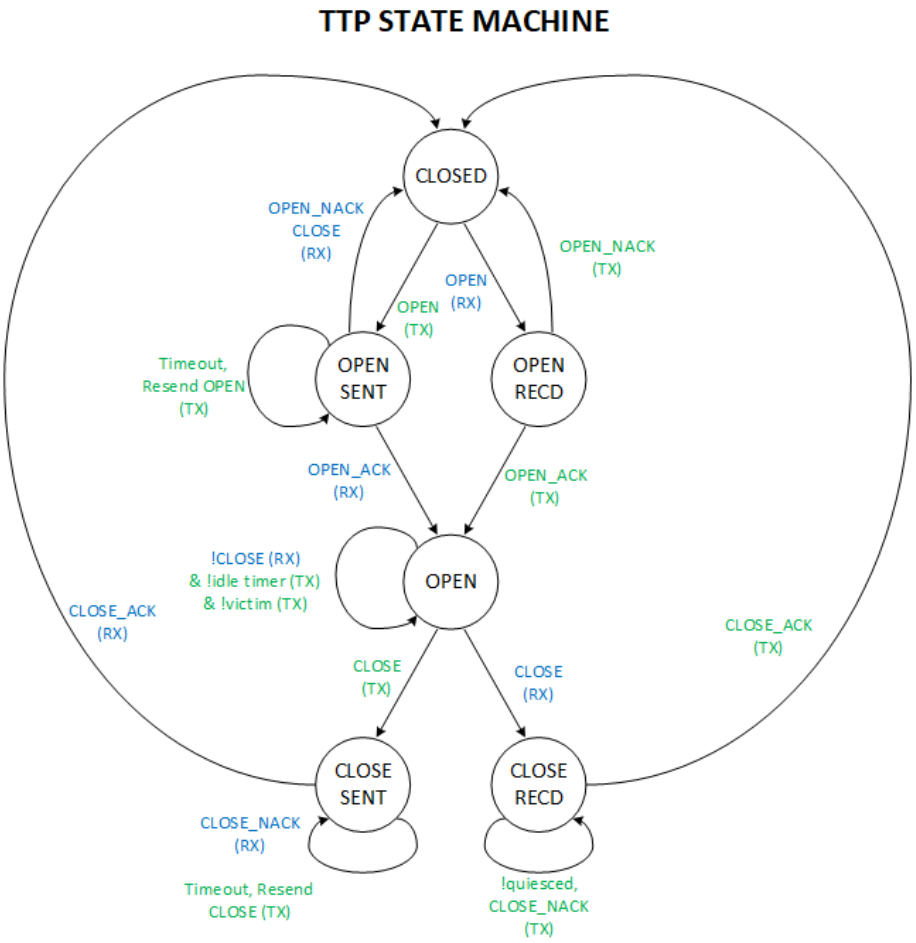
Diwakar Tundlam

Mackenzie Goodwin

Revision	Author	Date	Comment
1.0	equinnell	1/4/2022	Initial document creation, port over from spreadsheet of per-state into per-opcode
1.1	dwilliams	1/25/2022	Doug added concept of Virtual Channels (VC) to bits 7:6 of TTP_OPCODE. DATA may only be on VC 2, REQ_HI on VC 1, and REQ_LO on VC 0. Rename TTP_DATA to TTP_PAYLOAD
1.2	chsiong	1/28/2022	Changed opcode values, TTP_PAYLOAD starting at 'h6
1.3	equinnell	2/3/2022	Added case for DOJO-5717 to TTP_NACK clarifying that a TTP_CLOSE speculative ID will enforce NACKs on new PAYLOAD until the CLOSE handshake is complete
1.4	equinnell	7/28/2022	TTP_CLOSE on RX sends OPEN_SENT back to CLOSED. TTP_NACK_FULL RxID is ignored
1.5	equinnell	4/19/2023	Upon receiving TTP_CLOSE_NACK for a CLOSE_SENT state, the TTP_CLOSE original sender will not send another TTP_CLOSE until declared RxID difference is zero (prevents CLOSE storms)
1.6	equinnell	7/3/2024	Moved VC to separate byte for nxt-gen support, more VCs
2.0	equinnell	9/13/2024	Clean up for open sourcing

Table of Contents

1 TTP State Machine



Link States	Description
CLOSED	Link not established or agreed CLOSED -- REQ/DATA ignored/discarded
OPEN_SENT	TX sent OPEN, waiting on RX OPEN/OPEN_ACK
OPEN_RECD	RX sent OPEN, waiting on TX OPEN/OPEN_ACK. TX may choose aliased link to CLOSE
OPEN	TX/RX link established, REQ/DATA proceeds, IDs tracked
CLOSE_SENT	TX sent CLOSE, waiting for RX CLOSE_ACK. TX/RX send final ID endpoints
CLOSE_RECD	RX sent CLOSE, waiting for TX CLOSE_ACK. TX/RX send final ID endpoints

2 TTPoE Ethertype

https://standards-oui.ieee.org/ethertype/eth.txt

EtherType	ORGANIZATION / ADDRESS	PROTOCOL
9AC6	Tesla,Inc. 3500 Deer Creek Rd. PALO ALTO, CA 94304 US	TTP (Tesla Transport Protocol) over Ethernet

3 Opening a link in TTP

3.1 TTP_OPEN

Description: Request a new TTP link to open between two MAC endpoints

Opcode Mnemonic	Encoding[7:0]	TxID[31:0]	RxID[31:0]
TTP_OPEN	8'h00	OPEN pkt ID (1^{st} DATA is ID+1)	N/A

States/Responses:

Channel	Local LINK State		Response	Note
	Current	Next		
TX	**CLOSED**	**OPEN_SENT**	N/A	
	OPEN_SENT			Timeout, replay
	OPEN_RECD			Re-fetch remote OPEN ID
	OPEN			
	CLOSE_SENT			illegal
	CLOSE_RECD			
RX	**CLOSED**	**OPEN_RECD**	TTP_OPEN_ACK	
		CLOSED	TTP_OPEN_NACK	No available ways, create victim
	OPEN_SENT		TTP_OPEN_ACK	Always respond, timeouts or lost TTP_OPEN_ACK
	OPEN_RECD			
	OPEN			
	CLOSE_SENT			
	CLOSE_RECD			

Red = common case

Assumptions:

1. All received TTP_OPEN opcodes must issue a response in any state
2. Each endpoint communicates their own starting packet ID.
3. TTP_OPEN requires its own unique ID; TTP_OPEN_ACK does not
4. Local state is only OPEN when both a TTP_OPEN and TTP_OPEN_ACK have been processed
    - i.e. "Passing ships" OPENs on TX+RX simultaneously ***do not*** shortcut to OPENED
5. TTP_PAYLOAD is ***serialized*** and stalled until a full OPEN state is achieved

3.2 TTP_OPEN_ACK

Description: Acknowledge response to a TTP_OPEN request that the link is considered OPEN; set local starting ID

Opcode Mnemonic	Encoding[7:0]	TxID[31:0]	RxID[31:0]
TTP_OPEN_ACK	8'h01	1st local DATA ID	TTP_OPEN TxID

States/Responses:

Channel	Local LINK State		Response	Note
	Current	Next		
TX	CLOSED		N/A	timeout or lost TTP_OPEN
	OPEN_SENT			TTP_OPEN_ACK for remote TTP_OPEN, but not local TTP_OPEN
	OEN_RECD	**OPEN**		
	OPEN			timeout or lost TTP_OPEN
	CLOSE_SENT			
	CLOSE_RECD			
RX	CLOSED		N/A	Illegal?
	OPEN_SENT	**OPEN**		
	OPEN_RECD			TTP_OPEN_ACK for local TTP_OPEN, but not remote TTP_OPEN
	OPEN			timeout or lost TTP_OPEN
	CLOSE_SENT			
	CLOSE_RECD			

Red = common case

Assumptions:

1. TTP_OPEN requires its own unique ID; TTP_OPEN_ACK does not

3.3 TTP_OPEN_NACK

Description: Not-Acknowledge response to a TTP_OPEN request. The link is still considered CLOSED

Opcode Mnemonic	Encoding[7:0]	TxID[31:0]	RxID[31:0]
TTP_OPEN_NACK	8'h02	N/A	TTP_OPEN TxID

States/Responses:

Channel	Local LINK State		Response	Note
	Current	Next		
TX	**CLOSED**		N/ANOLI	
	OPEN_SENT			Illegal*
	OPEN_RECD			
	OPEN			
	CLOSE_SENT			
	CLOSE_RECD			
RX	CLOSED		N/A	Illegal*
	OPEN_SENT	**CLOSED**		
	OPEN_RECD			Illegal*
	OPEN			
	CLOSE_SENT			
	CLOSE_RECD			

Red = common case

* NACKs are far more restrictive than ACKs and should instead be specific to a particular situation.

Assumptions:

1. TTP_OPEN requires its own unique ID; TTP_OPEN_NACK does not

4 Closing a link in TTP

4.1 TTP_CLOSE

Description: Request a TTP link to close between two MAC endpoints; set local final ID

Opcode Mnemonic	Encoding[7:0]	TxID[31:0]	RxID[31:0]
TTP_CLOSE	8'h03	Last TX DATA ID + 1	Last seen RX DATA (speculative)

***Note**: TTP_CLOSE is a complex control command due to the speculative nature of unseen/outstanding packets already in flight*

States/Responses:

Channel	Local LINK State		Response	Note
	Current	Next		
TX	CLOSED	CLOSED	N/A	Illegal
	OPEN_SENT	OPEN_SENT		
	OPEN_RECD	OPEN_RECD		
	OPEN	CLOSE_SENT		
	CLOSE_SENT	CLOSE_SENT		Timeout, replay
	CLOSE_RECD	CLOSE_RECD		
RX	CLOSED	CLOSED	TTP_CLOSE_ACK	Always respond, timeouts or lost TTP_CLOSE_ACK from prior connection
	OPEN_SENT	CLOSED		
	OPEN_RECD	OPEN_RECD		
	OPEN	CLOSE_RECD	TTP_CLOSE_ACK	
			TTP_CLOSE_NACK	RxID > local TxID
	CLOSE_SENT	CLOSED	TTP_CLOSE_ACK	RxID == (local TxID + 1)
		CLOSE_SENT	TTP_CLOSE_NACK	RxID > (local TxID + 1)
	CLOSE_RECD	CLOSE_RECD	TTP_CLOSE_ACK	
			TTP_CLOSE_NACK	RxID > local TxID

Red = common case

Assumptions:

1.) TTP_CLOSE RxID is always *speculative*
   - The RxID represents the last ID processed by the TTP_CLOSE sender, even if remote has CLOSE_NACK'd at a higher ID

2.) TTP_CLOSE_ACK has a meaningless TxID
   - TTP_CLOSE always generates a response from the receiver if no DATA NACK is outstanding
   - While a PITA for DV, but if a redundant CLOSE from a previous connection appears, a TTP_CLOSE_ACK will be generated and the TxID of the former connection is already lost

3.) TTP_CLOSE received on RX while in CLOSE_SENT ***may shortcut*** to CLOSED if IDs are off by 1 or 0
   - RxID == (local TxID + 1) – the +1 represents the newly seen TTP_CLOSE

4.) TTP_CLOSE requires its own unique ID; TTP_CLOSE_ACK does not
5.) All received TTP_CLOSE opcodes must issue a response in any state

CLOSE_ACK/CLOSE_NACK Tx/Rx ID special cases:

Case	DATA sent/recd?		TX			RX		
	TX	**RX**	**OPCODE**	**TXID**	**RXID (speculative)**	**OPCODE**	**TXID**	**RXID (speculative)**
Can't send CLOSE	0	0						
	TX	**RX**	**OPCODE**	**TXID**	**RXID (speculative)**	**OPCODE**	**TXID**	**RXID**
TX sends CLOSE	1	0	TTP_CLOSE	TX(DATA+1)	RX(OPEN ID)	TTP_CLOSE_ACK		TX(DATA+1)
						TTP_CLOSE_NACK	RX(OPEN ID + N)	TX(DATA+1)
	1	1	TTP_CLOSE	TX(DATA+1)	RX(DATA)	TTP_CLOSE_ACK		TX(DATA+1)
						TTP_CLOSE_NACK	RX(DATA+N)	TX(DATA+1)
	TX	**RX**	**OPCODE**	**TXID**	**RXID**	**OPCODE**	**TXID**	**RXID (speculative)**
RX sends CLOSE	0	1	TTP_CLOSE_ACK		RX(DATA+1)	TTP_CLOSE	RX(DATA+1)	TX(OPEN ID)
			TTP_CLOSE_NACK	TX(OPEN ID + N)	RX(DATA+1)			
	1	1	TTP_CLOSE_ACK		RX(DATA+1)	TTP_CLOSE	RX(DATA+1)	TX(DATA)
			TTP_CLOSE_NACK	TX(DATA+N)	RX(DATA+1)			
	TX	**RX**	**OPCODE**	**TXID**	**RXID (speculative)**	**OPCODE**	**TXID**	**RXID (speculative)**
TX + RX send CLOSE	1	1	**"Passing ships" Sent**					
			TTP_CLOSE	TX(DATA+1)	RX(DATA)	TTP_CLOSE	RX(DATA+1)	TX(DATA)
			Responses					
			TTP_CLOSE_ACK		RX(DATA+1)	TTP_CLOSE_ACK		TX(DATA+1)
			TTP_CLOSE_NACK	TX(DATA+1+N)	RX(DATA+1)	TTP_CLOSE_NACK	RX(DATA+1+N)	TX(DATA+1)

4.2 TTP_CLOSE_ACK

Description: Acknowledge response to a TTP_CLOSE request. The link is now considered CLOSED

Opcode Mnemonic	Encoding[7:0]	TxID[31:0]	RxID[31:0]
TTP_CLOSE_ACK	8'h04	N/A	TTP_CLOSE TxID

States/Responses:

Channel	Local LINK State		Response	Note
	Current	Next		
TX	CLOSED		N/A	timeout or lost TTP_CLOSE
	OPEN_SENT			
	OPEN_RECD			
	OPEN			
	CLOSE_SENT	**CLOSED**		TTP_CLOSE_ACK can finalize CLOSE from any closing state
	CLOSE_RECD			
RX	CLOSED		N/A	timeout or lost TTP_CLOSE
	OPEN_SENT			
	OPEN_RECD			
	OPEN			
	CLOSE_SENT	**CLOSED**		TTP_CLOSE_ACK can finalize CLOSE from any closing state
	CLOSE_RECD			

Red = common case

Assumptions:

1. TTP_CLOSE requires its own unique ID; TTP_CLOSE_ACK does not

4.3 TTP_CLOSE_NACK

Description: Not-Acknowledge response to a TTP_CLOSE request. The link is still closing, but outstanding packets are not yet resolved; set local closing ID

Opcode Mnemonic	Encoding[7:0]	TxID[31:0]	RxID[31:0]
TTP_CLOSE_NACK	8'h05	(Set) Last TX ID	Last RX ID

States/Responses:

Channel	Local LINK State		Response	Note
	Current	Next		
TX	CLOSED		N/A	Illegal*
	OPEN_SENT			
	OPEN_RECD			
	OPEN			
	CLOSE_SENT			
	CLOSE_RECD			
RX	CLOSED		N/A	Illegal*
	OPEN_SENT			
	OPEN_RECD			
	OPEN			
	CLOSE_SENT			
	CLOSE_RECD			

Red = common case

* NACKs are far more restrictive than ACKs and should instead be specific to a particular situation.

Assumptions:

1. TTP_CLOSE requires its own unique ID; TTP_CLOSE_NACK does not
2. Local connection sets TxID as final local ID and may only set TxID.
3. RxID represents the remote speculative final set on its message's TxID during CLOSE negotiations
4. Receiving a TTP_CLOSE_NACK shall prevent sending any more TTP_CLOSE replays until the remote's last ID has been reached.

5 Control, data movement, and responses in TTP

5.1 TTP_PAYLOAD

Description: Sends Dojo control or data packets as payload with a TTP header packet; DATA packets are sent on VC 2 and are 1 to 16 beats (64B – 1KB); REQ_HI packets are sent on VC 1 within the TTP header beat; REQ_LO packets are sent on VC 0 within the TTP header beat

Opcode Mnemonic	Encoding[7:0]	TxID[31:0]	RxID[31:0]	LEN[15:6]
TTP_PAYLOAD	8'h06	PAYLOAD TxID	N/A	Data VC: #data beats + 1 Req VC: 1

States/Responses:

Channel	Local LINK State		Response	Note
	Current	Next		
TX	CLOSED		N/A	Illegal – stall until OPEN
	OPEN_SENT			
	OPEN_RECD			
	OPEN			
	CLOSE_SENT			Send until TxID > remote closing ID, then stall
	CLOSE_RECD			
RX	CLOSED		TTP_NACK_NOLINK	Lost TTP_PAYLOAD
	OPEN_SENT			
	OPEN_RECD			
	OPEN		**TTP_ACK**	TxID <= local RxID
			TTP_NACK	TxID > local RxID
			TTP_NACK_FULL	local is temporarily full
	CLOSE_SENT, CLOSE_RECD		TTP_ACK	TxID <= local RxID
			TTP_NACK	TxID > local RxID
			TTP_NACK_FULL	Local is temporarily full
			TTP_NACK_NOLINK	TxID > local closing ID

Red = common case

Assumptions:

1. TxID vs RxID must consider and handle 32-bit wraps for age ordering
   - e.g. 0xFFFF_FFFF vs 0x0000_0000, when the zero case is ***younger***
2. VC2: Minimum 1 beat of DATA, maximum 16 beats
3. VC1 and VC0: REQ_HI and REQ_LO control packets are embedded within the 64B TTP header
4. TTP_PAYLOAD may ***not*** be sent in OPEN_SENT or OPEN_RECD states
5. TTP_PAYLOAD requires its own unique ID; TTP_ACK/NACK/etc. do not

5.2 TTP_ACK

Description: Acknowledge response to a TTP_PAYLOAD that the packet has been received and correctly processed; local endpoint may deallocate resources as replays may no longer occur on the ID

Opcode Mnemonic	Encoding[7:0]	TxID[31:0]	RxID[31:0]
TTP_ACK	8'h07	N/A	TTP_PAYLOAD TxID

States/Responses:

Channel	Local LINK State		Response	Note
	Current	Next		
TX	CLOSED		N/A	Illegal – should be TTP_NACK_NOLINK
	OPEN_SENT			
	OPEN_RECD			
	OPEN			
	CLOSE_SENT			
	CLOSE_RECD			
RX	CLOSED		N/A	Lost TTP_ACK, discard
	OPEN_SENT			
	OPEN_RECD			
	OPEN			
	CLOSE_SENT			
	CLOSE_RECD			

Red = common case

Assumptions:

1. TTP_PAYLOAD requires its own unique ID; TTP_ACK does not
2. TTP_ACKs ***shall be sent redundantly*** to older-than-expected/redundant TTP_PAYLOAD packets in OPEN/CLOSE_SENT/CLOSE_RECD states
    - Previous TTP_ACK may have been lost, remote replays TTP_PAYLOAD and still requires TTP_ACK

5.3 TTP_NACK

Description: Not-Acknowledge response to a TTP_PAYLOAD request that the packet transfer (and any subsequent packets) has failed; RxID indicates the ID of the initial failure; ignore younger packets on the link until failed ID received

Opcode Mnemonic	Encoding[7:0]	TxID[31:0]	RxID[31:0]
TTP_NACK	8'h08	N/A	Failed TTP_PAYLOAD TxID

States/Responses:

Channel	Local LINK State		Response	Note
	Current	Next		
TX	CLOSED		N/A	Illegal – should be TTP_NACK_NOLINK
	OPEN_SENT			
	OPEN_RECD			
	OPEN			
	CLOSE_SENT			
	CLOSE_RECD			
RX	CLOSED		N/A	Lost TTP_NACK, discard
	OPEN_SENT			
	OPEN_RECD			
	OPEN			
	CLOSE_SENT			
	CLOSE_RECD			

Red = common case

Assumptions:

1. TTP_PAYLOAD requires its own unique ID; TTP_NACK does not
2. TTP_NACK allows for any younger ID traffic to be ignored by the NACK-ing receiver and therefore unresponsive on the specific NACK-ing link. The receiver must resume responses on the link when the NACK's communicated failed ID is received on an incoming packet
3. Per TTP_CLOSE assumption #1, a NACK may appear suddenly following a local point sending a TTP_CLOSE. The local point declares a TTP_CLOSE with the last accepted PAYLOAD as the final line, and therefore will NACK any incoming PAYLOAD until it confirms a response to the TTP_CLOSE.

5.4 TTP_NACK_FULL

Description: Not-Acknowledge response to a TTP_PAYLOAD request that the local receiver's resources are full; Alias to TTP_NACK

Opcode Mnemonic	Encoding[7:0]	TxID[31:0]	RxID[31:0]
TTP_NACK_FULL	8'h09	N/A	N/A

***Note**: TTP_NACK_FULL is used as an optimization to immediately discard new incoming traffic on the link until RX resources drain. This is a congestion alternative to IEEE 802.3 pause packets or faults. It is similar to TTP_NACK in protocol implementation, although the RxID loses meaning while queues drain*

States/Responses:

Channel	Local LINK State		Response	Note
	Current	Next		
TX	CLOSED		N/A	Illegal – should be TTP_NACK_NOLINK
	OPEN_SENT			
	OPEN_RECD			
	OPEN			
	CLOSE_SENT			
	CLOSE_RECD			
RX	CLOSED		N/A	Lost TTP_NACK_FULL, discard
	OPEN_SENT			
	OPEN_RECD			
	OPEN			
	CLOSE_SENT			
	CLOSE_RECD			

Red = common case

Assumptions:

1. TTP_PAYLOAD requires its own unique ID; TTP_NACK_FULL does not
2. TTP_NACK_FULL allows for any younger ID traffic to be ignored by the NACK-ing receiver and therefore unresponsive on the specific NACK-ing link. The receiver must resume responses on the link when the NACK's communicated failed ID is received on an incoming packet

5.5 TTP_NACK_NOLINK

Description: Not-Acknowledge response to a TTP_PAYLOAD request that the link is closed

Opcode Mnemonic	Encoding[7:0]	TxID[31:0]	RxID[31:0]
TTP_NACK_NOLINK	8'h0A	N/A	TTP_PAYLOAD TxID

States/Responses:

Channel	Local LINK State		Response	Note
	Current	Next		
TX	CLOSED		N/A	Lost TTP_PAYLOAD/TTP_REQ
	OPEN_SENT			
	OPEN_RECD			
	OPEN			Link down, throw interrupt
	CLOSE_SENT			Received TxID > local closing ID
	CLOSE_RECD			
RX	CLOSED		N/A	Lost TTP_PAYLOAD/TTP_REQ
	OPEN_SENT			
	OPEN_RECD			
	OPEN			Link down, throw interrupt
	CLOSE_SENT			Sent TxID > remote closing ID
	CLOSE_RECD			

Red = common case

Assumptions:

1. TTP_PAYLOAD requires its own unique ID; TTP_NACK_NOLINK does not
2. TTP_NACK_NOLINK responds to every TTP_PAYLOAD packet without a link
    - TTP_NACK_NOLINK is unlike TTP_NACK in that it does not prevent future responses nor expect a replay of the packet ID