WiThrottle Protocol v2.0

The WiThrottle protocol uses a text-based communication to link a controller (client) to an interface (server) connected to a model railroad. All commands are character values, not numeric. The client is typically a handheld device. Examples include an iOS device running the WiThrottle app or an Android device running the Engine Driver app. Some of the current servers that implement the WiThrottle protocol are JMRI software, Digitrax LNWI, and the MRC Prodigy WiFi module.

Zero Configuration Networking can be used for connection using the service type: `_withrottle._tcp.` in domain: `local`. Manual connection using IP address can also be used and the preferred port is `12090`.

The protocol is non-blocking, no request should wait for a response. That does not mean there are no responses, but a response is not guaranteed. An example occurs when requesting an address from a server. There is an affirmative response if successful but typically no response to a failure.

Each command is plain text, with a system-specific newline character at the end (typically `0x0A`). A newline could be a linefeed (`0x0A`), carriage return (`0x0D`), or a carriage return followed by a newline. (`0x0D0A`) Clients and servers should accept any newline sequence as a command termination. Multiple messages may be grouped in the same Wi-Fi packet with the newline separating each command.

There exists an early version (lower than version 2.0) of the WiThrottle protocol which is now deprecated. It has a more simple command structure but lacks many features that users now consider as standard. It is not recommended to start with the older version as it does not easily convert to the newer format. Also, being deprecated, implementations may not recognize that format. This earlier version will not be described in this document.

Upon initial connection, there are two commands that need to be sent from the client and one command from the server. The sequence of these is not critical and neither client nor server should wait for the other to initiate communication. They should be sent once the network streams have completed opening.

- The client sends the device name packet and a unique identifier packet. The name is used as a reference. The unique identifier prevents the same device from having two (or more) open connections to the server. If a client disconnects and immediately reconnects, the identifier will allow the server to recognize that event and clean up the no longer valid connection.
- The server sends `VN2.0` - the version number which is 2.0 as described in this document.

Any command which is not recognized by either client or server needs to be cleanly ignored.

N — Name

Send a human-readable name for the client to the server. Spaces and punctuation are acceptable. Should be unique, but is not used in any command structures.

```
NCasey Jones’ iPhone
```

H — Hardware

System information.

U — UUID

Device unique identifier that is sent only to server. Must be unique for each client and remain constant between program runs on that client. This is used to determine if a device has disconnected and then reconnected on a new set of streams.

```
HU480B75D9-CA18-442F-A3AE-F9F63D8A826F
```

M — Message - UI Blocking

A message that is intended to stop the user’s interaction with the client app. Typically indicates a failed action. Should only be used when necessary.

```
HMAlert - Stop interaction!
```

m — Message - Non-Blocking

A minor message that is likely to be ignored by the client’s user. Do not overuse!

```
HmIgnore me!
```

T — Hardware Manufacturer

The brand of DCC system.

```
HTMRC
```

t — Hardware Sub-Type

The type of interface to the DCC system.

```
HtMRCWi-Fi
```

M — MultiThrottle

This is the workhorse of the protocol. It allows address specific information and commands. Most are bi-directional and can be sent to a client or server. The basic format is:

- Three characters
- Decoder address
- Delimiter sequence
- Command

1. The first character is always `M`
2. Second is throttle group which is an ASCII character. Typically A-Z and 0-9. Addresses are grouped together by this for control. (Universal Consisting)
3. The third character is one of: `+`, `-`, `S`, `A`, or `L`. (Described later)
4. DCC address starting with either `S` or `L` for short or long, followed by the number. May be an `*` to send the command to all addresses in this throttle group. Number of characters varies.
5. `<;>` Delimiter sequence separating the header from the command.
6. A single command of varying length followed by a newline termination.

The following command will be used as an example. A newline character would exist at the end of each command but will be omitted for clarity throughout this document.

`MTAL757<;>V14`

In order:

- `M` - Always starts with M
- `T` - Throttle group, can be any ASCII character. Addresses are grouped together by this for control. (Universal Consisting)
- `A` - Action, an action command will follow.
- `L757` - mobile decoder address, in this example an extended address of 757.
- `<;>` - Delimiter separating the header from the command.
- `V` - Speed command
- `14` - Speed step 14. (in 128 speed step mode) Valid values 0-126. In the WiThrottle Protocol, 1 is a speed - not “emergency stop” as DCC would use.

+ — Request Address

Request mobile address commands. These commands can be stacked on the same throttle group to create a Universal Consist. Server should reply to a request before client sends another request for a different address to add to the group.

- `L` - Long (or extended) address formatted as `M` throttle group `+` mobile address `<;>` mobile address. Requests by DCC address.
  `MT+L757<;>L757`
  Server responds with `MT+L757<;>` on success.
- `S` - Short address formatted as `M` throttle group `+` mobile address `<;>` mobile address. Requests by DCC address.
  `MT+S42<;>S42`
  Server responds with `MT+S42<;>` on success.

- `E` - Request address by roster entry name formatted as `M` throttle group `+` mobile address `<;>E` roster name.
  `MA+L1071<;>ECNJ Heritage`
  Server responds with `MA+L1071<;>` on success.

Note: If all three commands above are sent, the first two would be Universal Consist to each other as they are both on 'T'. The third command is on a separate throttle group.

- — Remove Address

Yes, the character is a dash, and the heading above has a second dash before the description.
Relinquish control of an address. This command can be used to release one address of a Universal Consist without releasing the entire group.

- `r` - Release address.
  `MT-L757<;>r`
  Server responds with `MT-L757<;>` on release.
- `d` - Dispatch address. Server will need to release on systems that do not use dispatch.
  `MT-L757<;>d`
  Server responds with `MT-L757<;>` on release.

S — Steal Address

Has two meanings. Sent to client to indicate that an address is in use by another throttle. Client sends to server to force acquire the address.
Client sends: `MT+L757<;>L757` — Request address.
Server sends: `MTSL757<;>L757` — Refused, but can be forced.
Client sends: `MTSL757<;>L757` — Force acquisition.
Server sends: `MT+L757<;>` — Success.

A — Action

Action commands, after delimiter, are as follows:

- `C` - Set lead loco by DCC address, for server to forward DAC function commands. Sent only to server.
  `MAAS13<;>CL757`
  DAC with address S13 forward function commands to L757.
- `c` - Set lead loco by roster entry name, for server to forward DAC function commands. Sent only to server.
  `MAAS14<;>cCNJ Heritage`
  DAC with address S14 forward function commands to address of CNJ Heritage.
- `V` - Followed by speed. Set speed to value of 0-126. Server will need to adjust to match DCC values. Sent to server or client.
  `MTAL757<;>V25`
  Set speed of L757 in T group to step 25.
- `I` - Send an idle command. Typically interpreted as speed step 0. Sent to server.
  `MTAL757<;>I`

- X - Send an emergency stop command. Typically interpreted as speed step 1. Sent to server.
  MTAL757<;>X
- R - Followed by 0 to set direction to reverse, 1 to set forward. Sent to server or client.
  MTAL757<;>R1
- F - Send position of function "button" to server. Followed by 1 for depressed, 0 for released, and also function number (0-28). Server should evaluate how to handle momentary or latching aspect of function and respond to client with same format of packet to indicate function state if changed. (0 - off, 1 - on)
  Momentary:
  Client sends: MTAL757<;>F12 — Function button 2 pressed.
  Server sends: MTAL757<;>F12 — Function 2 on.
  Client sends: MTAL757<;>F02 — Function button 2 released.
  Server sends: MTAL757<;>F02 — Function 2 off.
  Latching (if function 16 was off):
  Client sends: MTAL757<;>F116 — Function button 16 pressed.
  Server sends: MTAL757<;>F116 — Function 16 on.
  Client sends: MTAL757<;>F016 — Function button 2 released.
  Server sends: nothing — Function 16 does not change.
- f - Force a function to on or off state. Sent only to server.
  MTAL757<;>f16
  Set function 6 to on.
  Server should respond on function change with:
  MTAL757<;>F16 — Capital F!
- s - Set speed step 1-128, 2-28, 4-27, 8-14. Sent to server or client.
  MTAL757<;>s8
  Set L757 to 14 speed step mode.
- m - Set momentary status of function. Followed by 1 for momentary, 0 for latching. Sent to server or client.
  MTAL757<;>m114
  Set status to momentary for function 14.
- q - Query a value. Sent only to server.
  MTAL757<;>qV — Speed
  MTAL757<;>qR — Direction
  MTAL757<;>qs — Speed step
  MTAL757<;>qm — Momentary status of all functions

L — Labels

After the header delimiter, each function has a separator]\[followed by its label. A typical use is 29 separators and 29 labels. Blank labels are fine with nothing between the separators. If there are no function labels for the address, this command should not be used. Sent only to client.

```
MTLL5606<;>]\[Light]\[Bell]\[Horn]\[Coupler]\[Air Feature Disable]\
[Dynamic Brakes]\[Switching Speed & Strobe]\[Crossing Horn]\[Mute]\
[Brake Squeal]\[Crossing Horn]\[]\[]\[]\[]\[]\[]\[]\[]\[]\[]\[]\[]\
[]\[]\[]\[]\[
```

P — Panel

Items that are typically found on control panels of a layout. The server has control over whether the clients can interact with each of these features.

P — Power

Track power controls. Power status shall only be sent to client if server allows power control from clients. Sent to server to change power state or to client to indicate power state. (Unknown should not be sent to server.)

- A - Action followed by 0 (off), 1 (on), or 2 (unknown) to indicate state.
  PPA1

T — Turnouts

List, control, and permitted actions of turnouts. State is represented by 1 (unknown), 2 (closed), or 4 (thrown).

- T - Title list as value/key pairs. Pairs are separated by]\[and value/key separated by }|{. Sent only to client. This command is also used to indicate that turnout control is allowed by this server. A server shall only send this command if turnout control is allowed on the connected layout.
  PTT]\[Turnouts}|{Turnout]\[Closed}|{2]\[Thrown}|{4
- L - Turnout list with a format of]\[system name }|{ user name }|{ state. This pattern repeats for each turnout stored in the server that is available to the client.
  PTL]\[LT304}|{Yard Entry}|{2]\[LT305}|{A/D Track}|{4
- A - Action. Has two patterns:
  1. Sent to server - commands are 2 (toggle), C (closed), or T (thrown).
     PTA, followed by command, followed by system name.
     PTATLT304
     Set “thrown” LT304.
  2. Sent to client - States are 1 (unknown), 2 (closed), or 4 (thrown). Server should send state to clients whenever state changes.
     PTA, followed by state, followed by system name.
     PTA4LT304
     State of LT304 is “thrown”.

R — Routes

List, control, and permitted actions of routes. State is represented by 1 (unknown), 2 (active), or 4 (inactive).

- T - Title list as value/key pairs. Pairs are separated by]\[and value/key separated by }|{. Sent only to client. This command is also used to indicate that route control is allowed by this server. A server shall only send this command if route control is allowed on the connected layout.
  PRT]\[Routes}|{Route]\[Active}|{2]\[Inactive}|{4
- L - Route list with a format of]\[system name }|{ user name }|{ state. This pattern repeats for each route stored in the server that is available to the client.
  PRL]\[IO:AUTO:0002}|{CH: Barge 1}|{2]\[IO:AUTO:0003}|{CH: Barge 2}|{4

- A - Action. Has two patterns:
  1. Sent to server - command is 2 (active). You cannot “inactive” a route.
     PRA, followed by command, followed by system name.
     PRA2IO:AUTO:0002
     Set “active” CH: Barge 1 route.
  2. Sent to client - States are 1 (unknown), 2 (active), or 4 (inactive). Server should send state to clients whenever state changes.
     PRA, followed by state, followed by system name.
     PRA4IO:AUTO:0003
     State of CH: Barge 2 is “inactive”.

W — WebServer

Indicates WebServer is available and what port number it is on. Sent only to client.
PW12080

F — Fast Clock

- T - Current time to display. Format is PFT followed by the number of seconds from the reference date of 00:00:00 UTC on 1 January 1970. It is preferred to append <;> rate value - but not mandatory. Adding the rate to the string allows self-running clocks and reduces the frequency that time updates should be sent. If no rate is sent, clocks will update only by receiving new time value. Sent only to client.
  PFT1607717340<;>1.0
  PFT74100<;>1.0
  Start clock running at 1.0:1 rate at 20:09:00.
  PFT1607718869<;>0.0
  Stop clock.
  PFT1607718910<;>4.0
  Restart clock with rate of 4.0:1 at 20:35:10.

R — Roster

This category is used for a stored roster of mobile decoders and also Decoder Assisted Consisting (DAC).

L — Roster List

A list of addresses known and stored on the server. The format is RL followed by the number of roster entries. Each roster has a format of]\[loco name }|{ address number }|{ Short or Long. Sent only to client. RL0 should be sent if the system has an empty roster or does not support a roster. This will ensure that a client that was previously connected to a layout with a roster will update properly.

```
RL3]\[BN 9710 F9}|{7}|{S]\[BNSF 9388 SD70ACe}|{9388}|{L]\[WM 146 S4}|{146}|{L
```

C — Consist

Decoder Assisted Consists (DAC). RCL or RCC command must be sent before any data commands so that client has record of number of consists.

- L - List only. Followed by number of consists. Clients are not allowed to create, edit, or delete DACs. Sent only to client.
  RCL2 - Two consists, no changes allowed.
- C - Controllable. Followed by number of consists. Clients are allowed to create, edit, or delete DACs. Sent only to client.
  RCC2 - Two consists, changes are allowed.
- D - Data for the consists. Each consist will be sent separately and has a format of RCD}|{ consist address }|{ consist name]\[lead loco DCC address }|{ is normal direction]\[next loco DCC address }|{ is normal direction]\[trail loco DCC address }|{ is normal direction. **This command uses a different address format.** The loco and direction repeat for each mobile decoder that is in this consist. "is normal direction" will be true for normal and false for reversed. The order in the command is the order of the locos from front-to-rear. Sent only to client.
  RCD}|{14(S)}|{Valley Power]\[146(L)}|{true]\[333(L)}|{false
  DAC with address 14(S) has a user name of Valley Power. It contains two mobile addresses - 146(L) as lead loco in normal direction and 333(L) as trailing loco in reversed direction.
- P - Reorder consist loco positions. Format is RCP<;> consist address <:> lead loco <;> next loco <;> trail loco. Sent only to server.
  RCP<;>S14<:>L333<;>L146
- R - Remove DAC. Format is RCR<;> consist address. Sent only to server.
  RCR<;>S14
- + - Add mobile address to DAC. Create a DAC if one with this consist address does not exist
  RC+<;>S14<;>Valley Power<:>L146<;>true
  Add loco: L146 normal direction, to consist: S14.
  RC+<;>S14<;>Valley Power<:>L333<;>false
  Add loco: L333 reversed direction, to consist: S14.

- `-` - Remove mobile address from DAC. Format is `RC-<;>` consist address `<:>` mobile address. Sent only to server.
  `RC-<;>S14<:>L333`
  Remove loco: 333(L), from consist: S14
- `F` - Set CV 21 or 22 for this DAC mobile address. Format is `RCF<;>` mobile address `<:>` CV number `<;>` decimal value of function bitmap. Sent only to server.
  `RCF<;>L146<:>21<;>131`
  Write CV: 21 of L146 to: 131 (0b10000011)
  `RCF<;>L146<:>22<;>1`
  Write CV: 22 of L146 to: 1
  `RCF<;>L333<:>22<;>2`
  write CV: 22 of L333 to: 2

* — Heartbeat

A signal sent from the client to the server to indicate that the connection is still valid. On loss of connection, the server shall send emergency stop commands to all addresses controlled by that client. Any protocol message will restart the server's timeout counter, but the asterisk by itself is a lower overhead message dedicated to this purpose.

```
*
```

— Set Timeout

Replace with a number for length of timeout in seconds. Sent only to client so that client can send heartbeat within that range.

```
*12
```

Heartbeat is set to 12 second timeout.

+ — Start Heartbeat Timer

Sent only to server.

```
*+
```

− — Stop Heartbeat Timer

Sent only to server.

```
*−
```

Q — Quit

When a user leaves the client app, the app shall send this message to the server indicating that the server needs to clean up its end of the connection.

```
Q
```

Commands sent to ‘C’lient, ’S'erver, or ‘B’oth

N			Client Name	S

H			Hardware commands	
	U		UUID	S
	M		Message - UI Blocking	C
	m		message - minor ignored by WiThrottle app	C
	T		Hardware manufacturer	C
	t		Hardware sub-type	C

*****			Heartbeat	S
	#		Timeout value	C
	+		Start	S
	-		Stop	S

Q			Quit	S

M			MultiThrottle commands	
	+		Add address	
		L	Long	B
		S	Short	B
		E	By roster entry	
	-		Remove address	
		L	Long	B
		S	Short	B
		r	Release	S
		d	Dispatch	S
	S		Steal address	B
	A		Action	
		C	DAC lead loco by address	S
		c	DAC lead loco by roster entry	S
		V	Speed (as 128 speed step mode)	B
		X	Emergency stop	S
		I	Idle	S
		R	Direction	B
		F	Function	B
		f	Function	S
		s		B
		m		B
		q		S
	L		Function Labels	C

P			Panel commands	
	P		Power	
		A	Action	B
	T		Turnout	
		T	Titles	C
		L	List	C
		A	Action	B
	R		Route	
		T	Titles	C
		L	List	C
		A	Action	B
	W		WebServer	C
	F		Fast clock	
		T	Time	B

R			Roster commands	
	L		List	C
	C		Consist	
		L	List	C
		C	Controllable	C
		D	Data	C
		P	Reorder positions	S
		R	Remove DAC	S
		+	Add decoder	S
		-	Remove Decoder	S
		F	Program functions CV 21, CV 22	S