

**Possession of the TED Footprints API and Serial Communication Specification are by License only.  
Any unauthorized possession will be prosecuted to the full extent of the law.  
www.theenergydetective.com**

TED Footprints 1.30 (RDU Firmware 9.0) Serial Communication Specification					
© Copyright 2008 Energy, Inc. All Rights Reserved					
Pos	Bytes	Field	Type	Modifier	Description
0	2	Start of Packet Header			0x10, 0x04
<b>Setup Data</b>					
2	1	Flag - Season - Daylight Savings Time - Plan Type - Country Code	bit flag		ynSeason = (byteValue & 0x01) Summer/Winter ynDLS = (byteValue & 0x02) > 0 Daylight Savings Time PlanType = (byte)((byteValue & 0x0C) / 4) - 0:Flat, 1:Tiered, 2:Time of Use (TOU) CountryCode = (byte)((byteValue & 0xF0) / 0x10) - 0:USA 1:Canada
3	1	NumEnergyRates	byte		Number of energy rates in use by the RDU.
4	2	SummerFixMonthlyCharge	int16	* 0.01	Summer Fixed Monthly Charge
6	2	WinterFixMonthlyCharge	int16	* 0.01	Winter Fixed Monthly Charge
8	2	SummerMinMonthlyCharge	int16	* 0.01	Summer Minimum Monthly Charge
10	2	WinterMinMonthlyCharge	int16	* 0.01	Wintr Minimum Monthly Charge
12	2	SummerEnergySurcharge	int16	* 0.0001	Summer Energy Surcharge Value
14	2	WinterEnergySurcharge	int16	* 0.0001	Winter Energy Surcharge Value
16	2	SummerStart	int16		Day of Year that Summer Starts(128 = May 1st, 2009)
18	2	SummerEnd	int16		Day of Year that Summer Ends(128 = May 1st, 2009)

Possession of the TED Footprints API and Serial Communication Specification are by License only.

Any unauthorized possession will be prosecuted to the full extent of the law.

www.theenergydetective.com

Pos	Bytes	Field	Type	Modifier	Description
20	64	KwhBreakPeakTimeRanges	32 position array of Int16		<p>An array of two byte values indicating the breaks for tiered and TOU plans. Not used for a flat plan.</p> <p>For a tiered plan, the kWh tier break values are listed together followed by a "-1" to indicate the end of the season data. Winter break tiers are listed first.</p> <p>In the example of a 3-tier plan w/ seasons.  pos 0: 0 (winter tier 1 starts at 0kWh)  pos 1: 10000 (winter tier 2 starts at 10000kWh)  pos 2: 29999 (winter tier 3 starts at 29999kWh)  pos 3: -1  pos 4: 0 (summer tier 1 starts at 0kWh)  pos 5: 20000 (summer tier 2 starts at 20000kWh)  pos 6: 25000 (summer tier 3 starts at 25000kWh)  pos 7: -1</p> <p>If there is not a seasonal plan in effect, then only positions 0-3 would have been used.  -----</p> <p>For TOU plans, these indicate the start/end times (military time based) of the various peaks. NOTE: Seasonal data still starts with winter peaks. However, there is not a "-1" to indicate when the winter peaks end and the summer peak definitions start. This is indicated solely by position based on the number of TOU peaks in use. Times are represented as the number of minutes since midnight (e.g. 119 = 1:59am)</p> <p>An example of a seasonal 2-TOU plan:  pos 0: 0 (Winter Plan 1 starts at mid-night)  pos 1: 119 (Winter Plan 1 ends at 1:59am)  pos 2: 119 (Winter Plan 2 starts at 1:59am)  pos 3: 239 (Winter Plan 2 ends at 3:59am)  pos 4: 0 (Summer Plan 1 starts at mid-night)  pos 5: 839 (Summer Plan 1 ends at 1:59PM)  pos 6: 839 (Summer Plan 2 starts at 1:59PM)</p>

**Possession of the TED Footprints API and Serial Communication Specification are by License only.  
Any unauthorized possession will be prosecuted to the full extent of the law.  
www.theenergydetective.com**

Pos	Bytes	Field	Type	Modifier	Description
84	20	KwhRate	10 position array of Int16	*.0001	kWh rates as they related to the various tiers/TOU peaks specified in KwhBreak-PeakTimeRanges.  For example, using the 3-tier seasonal plan defined above, the array would look like:  pos 0: 1000 (Winter Tier 1 Rate of .010) pos 1: 1500 (Winter Tier 2 Rate of .015) pos 3: 1100 (Summer Tier 1 Rate of .011) pos 4: 1200 (Summer Tier 2 Rate of .012)
104	2	SalesTax	int16	*.0001	% of sales tax
106	1	MeterReadDay	byte		Meter Read Day. Zero based value so "0" is the first of the month, 1 is the 2nd of the month, etc.
107	2	MeterCalib	int16	*.0001	Meter Calibraton Value
109	1	isDualMTU	bit flag		Apply logical AND to get true/false value (byte & 0x01)
110	1	MTU Code #1	byte		The id of MTU #1
111	1	MTU Code #2	byte		The id of MTU #2
112	1	PlanID	byte		The Plan ID of the RDU
113	4	CompanyID	byte		The Company ID of the RDU
117	4	Not Used			
<b>Alarm Data</b>					
121	1	Flags	bit flags - Alarm Enable - Buzzer Enable		ynAlarms = (byteValue & 0x01) ynAlarmBuz = (byteValue & 0x02)
122	2	AlmPeakHrCost	int16	* 0.01	Alarm trigger for peak Cost per hour
124	2	AlmPeakKw	int16	* 0.01	Alarm trigger for peak kWh per hour
126	2	AlmPeakMtdCost	int16	* 0.1	Allarm trigger for peak month-to-date dlr cost
128	2	AlmPeakMtdKwh	int16		Alarm trigger for peak month-to-date kWh usage
130	2	AlmLoVrms	int16	* 0.1	Allarm trigger for low voltage alarm
132	2	AlmHiVrms	int16	* 0.1	Alarm trigger for high voltage alarm
<b>Log Data</b>					
134	2	LoVrmsTdy	double	* 0.1	Low Voltage reading for the day
136	2	FUTURE USE			
138	2	HiVrmsTdy	double		High Voltage reading for the day
140	2	FUTURE USE			

**Possession of the TED Footprints API and Serial Communication Specification are by License only.**  
**Any unauthorized possession will be prosecuted to the full extent of the law.**  
[www.theenergydetective.com](http://www.theenergydetective.com)

Pos	Bytes	Field	Type	Modifier	Description
142	2	LoVrmsMtd	double	* 0.1	High Voltage reading Month-To-Date
144	1	FUTURE USE			
145	2	HiVrmsMtd	double	* 0.1	High Voltage reading Month-To-Date
147	1	FUTURE USE			
148	2	KwPeakTdy	double	* 0.01	Peak kWh reading for the day
150	2	DlrPeakTdy	double	* 0.01	Peak Dlr Spent reading for the day
152	2	KwPeakMtd	double	* 0.01	Peak kWh reading Month-to-Date
154	2	DlrPeakMtd	double	* 0.01	Peak Dlr spent Month-to-Date
156	4	DlrTdySum	double	div by 600,000	Total spend today
160	4	KWhrTdySum	uint32	div by 60,000	Total kWh used today
164	4	KwhMtdCnt	double		Number of minutes since the beginning of the month. Used for internal calculations.
168	4	KwhMtdSum	uint32	div by 60,000	Total kWh used Month-To-Date
172	4	DlrMtdSum	double	div by 60	Total Dlr Cost Month-To-Date
<b>History Data</b>					
176	1	Next Index	Byte		The array position where the next month will be written.
177	72	History Records	12 position array of 6 bytes		The first two bytes of the packet are the month and year.
BYTE Mth : 4; //Month of history record. (byte[0] & 0x04 to get value) BYTE YrL : 4; //Last digit of year. (byte[0] >> 4 to get value) BYTE YrH : 3; //First digit of year [0 - 7] (byte[1] & 0x07 to get value) BYTE Day : 5; //Day of month [1-31] (byte[1] >> 5 to get value)  Bytes 3 and 4 are the Int16 value of the kWh used that month Bytes 5 and 6 are the Int16 values of the Dlr amount spent that month. (multiply by 0.10 to get value)					
<b>Live Data</b>					
249	2	KwNowDsp	double	* 0.01	Current kWh Usage Value
251	2	DlrNowDsp	double	* 0.01	Current Dollar Spent Value
253	2	VrmsNowDsp	double	* 0.1	Current Voltage Value
255	2	DlrMtdDsp	double	* 0.1	Dollar Month to Date Value
257	2	DlrMthProjDsp	double	* 0.1	Dollar Projected Monthly Value
259	2	KwhMthProjDsp	double		Monthly Projected kWh Value
261	1	LED Status flag	bit flag		FLAG_MSGSTART = 0x01 FLAG_BUZZON = 0x02 FLAG_GREENLED = 0x04 FLAG_YELLOWLED = 0x08 FLAG_REDLED = 0x16 FLAG_LEDTEST= 0x32
262	4	WattsNow1	double	* 0.001	Current Watts used by MTU1 (only used in multiple MTU environment)
266	4	WattsNow2	double	* 0.001	Current Watts used by MTU2 (only used in multiple MTU environment)
270	4	VrmsNow1	double	* 0.1	Current Voltage Reported by MTU 1 (only-used in multiple MTU environment)

**Possession of the TED Footprints API and Serial Communication Specification are by License only.**

**Any unauthorized possession will be prosecuted to the full extent of the law.**

**www.theenergydetective.com**

Pos	Bytes	Field	Type	Modifier	Description
274	4	VrmsNow2	double	* 0.1	Current Voltage Reported by MTU 2 (only-used in multiple MTU environment)
278	2	Current Rate	double	* 0.001	Current Rate being used by the RDU for calculations
280	2	END OF PACKET			0x10, 0x03
Total Size	282				
<b>NOTES:</b>					
If the 0x10 byte occurs within the packet (between positions 2 - 280), it is escaped with another 0x10 byte. This will increase the size of the packet and all additional values should be offset.					
Byte values are little-endian ordered					
All positions assume a zero-based index					
Modifiers are values that are multiplied					