STANDARD OPERATING PROCEDURE

TITLE: TECHNICAL RESOURCE GUIDE

SUBJECT: TYMKON COMMUNICATIONS PROTOCOL

1. SCOPE

This document details information regarding the communications protocol used between one or more Tymkons™ and a personal computer (hereafter referred to as 'Node Computer'). This document contains information that is not required by the typical end-user, but may be of value to anyone developing a 'user interface' for use with Tymkons™.

2. HARDWARE DESCRIPTION

Though Tymkons can be provided with other communications settings, the standard values are 115,200 baud, no parity, 7 data bits and 1 stop bit.

2.1. Host Connected to Tymkon via Isolated EIA-232 Adapter 810-0382

The simplest, highest performance way to connect a single Tymkon to a computer is to use a standard EIA-232 serial port on the computer. Most PC's have at least (2) comm ports and they always support EIA-232. This standard supports hardware handshake and allows for high baudrate communications with little risk of corrupted messages. The limitation of this approach is that a separate serial port is required for each Tymkon to be connected. The EIA-232 standard does not support multidrop connections.

In this type of installation, the Tymkon CPU board has a 6-pin RJ-12 female connector on Port D for connection to the EIA-232 DTE Comm Port on the Host Computer. The extension cable between the Tymkon and the Host is a standard serial extension cable with a 9-pin female 'D' on the end that plugs into the Host and a 9-pin male 'D' on the end that plugs into an adapter that plugs into the Tymkon. The adapter consists of a 9-pin female 'D' mated to a female 6P6C RJ12 described below. A short 'straight-thru' modular telephone cable (Pan Pacific part number DC-506P-3SV, or equivalent) connects the adapter to the Tymkon.

RJ-12 to 9-pin female 'D' adapter			
RJ-12 on Tymkon	Wire Color (inside adapter)	9-Pin 'D' Connector To PC	Signal
1	Blue	5	SG
2	Yellow	3	TDX from PC to Tymkon
3	Green	2	RDX from Tymkon to PC
4	Red	7	RTS from PC to Tymkon
5	Black	8	CTS from Tymkon to PC
6	White		

Note: Pin 1 on the RJ-12 receptacle is on the left when looking into the opening of the receptacle with the contacts across the top and the catch for the plug's latch on the bottom. Pins are labeled consecutively from left to right. Note: 'straight-thru' means that pin 1 goes to pin 1 (which looks reversed when the connectors are held end-to-end).

In some cases, a female 'D' connector is mounted directly on the Tymkon enclosure. In that case, a short RJ12 cable with tinned leads (available from L-COM as part number TDC029-1) is soldered to a female 9-pin 'D' which is then secured to the enclosure. The pinout of the connector is as follows:

RJ-12 plug to 9-pin female 'D' cable			
RJ-12 plug	Wire Color (inside adapter)	9-Pin 'D' Connector To PC	Signal
6	Blue		
5	Yellow	8	CTS from Tymkon to PC
4	Green	7	RTS from PC to Tymkon
3	Red	2	RDX from Tymkon to PC
2	Black	3	TDX from PC to Tymkon
1	White	5	SG

2.2. Host Connected to Tymkon's via Isolated 4-Wire EIA-485 Adapter 810-0384

The hardware interface used to connect a computer to multiple Tymkons is referred to as a multi-drop, 4-wire, full-duplex EIA-485 network. Since standard PC's do not come equiped with EIA-485 comm ports, a special add-in board or adapter is required. Tymkon's can be supplied to support either 4-wire EIA-485, 2-wire EIA-485 or 4-wire EIA-422, but 4-wire EIA-485 is highly preferred.

In a 4-wire, full-duplex EIA-485 network, the computer transmits to all of the Tymkons on one twisted pair of conductors. The Tymkon receivers are all wired in parallel on this data path. In a similar manner, all of the Tymkon transmitters are wired in parallel on the second twisted pair. A receiver at the computer then monitors this data path.

For optimum noise immunity, the EIA-485 receiver on the computer must be terminated with a 120-ohm resistor. Likewise, a termination resistor must be placed across the receiver pair at the Tymkon at the end of the network most distant from the computer. All apprppriate ITS communications adapters include a jumper to activate a built-in termination resistor.

In this protocol implementation, the computer serves as the 'master' and each of the Tymkons serves as a 'slave'. Since process timing and integrity is not dependent on network messaging, a polled protocol is used. In this arrangement, the master (Node Computer) EIA-485 transmitter is always enabled and is monitored by each of the slaves (Tymkons). All of the slaves share a single return data path to the master and, therefore, must be able to enable or disable their individual transmitters as required to avoid message conflicts. The Tymkon Host Communications option provides this capability.

In this type of installation, the first Tymkon has a 6-pin RJ-12 female connector on Port D for connection to the EIA-485 Master Port on the Host Computer. A second 6-pin RJ-12 female connector on Port D may be used to link to the first Port D connector on the next Tymkon and so on. The details of the EIA-485 connection between the first Tymkon and the Host Computer will depend on the type of EIA-485 I/O board or adapter used on the PC's serial port.

RJ-12 on Tymkon			Signal
1			Ground
2			RX- from PC to Tymkon
3			TX- from Tymkon to PC
4			TX+ from Tymkon to PC
5			RX+ from PC to Tymkon
6			Ground

Note: Pin 1 on the RJ-12 receptacle is on the left when looking into the opening of the receptacle with the contacts across the top and the catch for the plug's latch on the bottom. Pins are labeled consecutively from left to right. Note: 'straight-thru' means that pin 1 goes to pin 1 (which looks reversed when the connectors are held end-to-end).

3. Protocol Version

An 8-digit string is used to identify the capabilities of a particular Tymkon with regard to host communications. When the host software makes contact with a Tymkon, it receives the 8-digit string from the Tymkon in response to a "Version" query. The string consists of a 3-digit product code ("101" for all Tymkons) followed by a 5-digit protocol version. The original Tymkon communications protocol was identified as version "10100000".

On 9/27/2001, the version was changed to "10100001" when the temperature flags were relocated in the cycle definition. This coincided with the addition of a Temperature Segment Table to the Tymkon memory map.

On 5/27/2002, the version was changed to 10100002 when support for an Extended Status Message was added.

On 11/25/2005, the version was changed to 10100003 when support for clearing an individual Runs Counter was added.

4. Protocol Description

All messages sent from the computer to one or more of the Tymkons are bounded by an ASCII control character at each end. The start character of any such message is the ASCII control character 02H (<STX> = start-of-transmission). The terminal character of any computer-to-Tymkon message is the ASCII control character 0AH (<LF> = line-feed).

All messages sent from any Tymkon to the computer are also bounded by an ASCII control character at each end. The start character in these messages is the ASCII control character 01H (<SOH> = start-of-header). The terminal character in these messages is the ASCII control character 0DH (<CR> = carriage-return).

There are currently no timing requirements in this protocol. As long as the character bit timing matches the specified baud rate, inter-character delays, though undesirable, will not cause a message to be aborted.

The next section after the start character contains a 2-byte Device ID. In messages from the computer to the Tymkons, the Device ID identifies the particular Tymkon to which the message is directed. Valid Device ID's range from 01 to 99. In the future, special 'broadcast' messages

may be added to the protocol for generic messages from the computer to all Tymkons. These unique messages will be broadcast to Device ID=00. In this case, all of the Tymkons will receive and decode the message, but none will send a reply message.

Messages from a Tymkon to the computer also contain a Device ID. In these messages, the Device ID identifies the particular Tymkon from which the message originated.

The next four bytes after the Device ID may be any four ASCII characters, excluding the ASCII control characters. These four characters are used as a Serial Tag and are echoed back to the host in the reply message to confirm that the Tymkon received the message. In the ITS host software, the Serial Tag is a four digit number used to serialize and identify each message.

The next byte after the Serial Tag is a single ASCII character representing the type of command, query or data being transmitted. This section of the message is referred to as the message 'qualifier'. The qualifier character may be any 7-bit ASCII character excluding the ASCII control characters. Currently, approximately (24) message types have been defined.

Following the qualifier byte is a variable number of data bytes as determined by the requirements of the particular message type. Each particular qualifier has a specific message length associated with that type of message. The data section may be 0 length as are many messages from the computer to the host or they may be hundreds of bytes long as is the detailed status reply from a Tymkon to the computer.

All data bytes contained in the messages are ASCII characters. Most data is numeric, but those bytes containing binary data are sent as a 4-bit nibble or'd with 30 Hex resulting in ASCII characters "0123456789:;<=>?" depending on the 4-bit value.

In summary, each message in this protocol requires a minimum of (9) bytes in the following order: <STX>ddiiiiY[abcdefg…]<LF> where <STX> is the start character, 'dd' is the Device ID, 'iiii' is the Serial Tag, 'Y' is the qualifier, [abcdefg…] represents 0 to 1024 bytes of data and <LF> is the terminator.

5. HOST-TO-TYMKON STATUS AND CONTROL MESSAGES

Message Format	Bytes Req'd	Message to Tymkon	Reply from Tymkon	Effect on Download Mode & EEROM STORE/RECALL
<STX>ddiiiiD<LF>	9	Detailed status request	Detailed + Simple Status	None
<STX>ddiiiid<LF>	9	Extended status request	Extended + Simple Status	None
<STX>ddiiiiPrr<LF>	11	Select recipe & hold	Simple Status	Exit Download Mode & RECALL
<STX>ddiiiiG<LF>	9	Start or continue current recipe	Simple Status	Exit Download Mode & RECALL
<STX>ddiiiiS<LF>	9	Simple status request	Simple Status	None
<STX>ddiiiiK<LF>	9	Query Temperatures	Temperatures	None
<STX>ddiiiiu<LF>	9	Query Runs Counters	Runs Counters	None
<STX>ddiiiiUxx<LF>	11	Clear Runs Counter & query	Runs Counters	STORE
<STX>ddiiiiH<LF>	9	Go to hold mode	Simple Status	Exit Download Mode & RECALL
<STX>ddiiiiJ<LF>	9	Step to next cycle while in hold	Simple Status	Exit Download Mode & RECALL
<STX>ddiiiiRrr<LF>	11	Select recipe & run	Simple Status	Exit Download Mode & RECALL
<STX>ddiiiiI<LF>	9	Reset alarms & go to idle	Simple Status	Exit Download Mode & RECALL
<STX>ddiiiiA<LF>	9	Silence alarms	Simple Status	None
<STX>ddiiiiM<LF>	9	Manual Abort	Simple Status	Exit Download Mode & RECALL
<STX>ddiiiiV<LF>	9	Version and configuration request	Version etc.	Exit Download Mode & RECALL
<STX>ddiiiiX<LF>	9	Multipurpose message	Simple Status	None
<STX>ddiiiiZ[0:9]<LF>	19	Set time-of-day clock	Simple Status	None
<STX>ddiiiiQ[0:31]<LF>	41	Set Equipment Unique Identifier	Simple Status	STORE
<STX>ddiiiiw<LF>	9	Query Usage Timers	Output Timers	None
<STX>ddiiiiWxx<LF>	11	Clear Usage Timers & Query	Output Timers	STORE

5.1. Host-to-Tymkon Status and Control Messages Overview

5.1.1. Query Runs Counters [<STX>ddiiiiu<LF>]

This command issues a query to update the Runs Counter displays.

5.1.2. Clear Runs Counters & Query [<STX>ddiiiiUxx<LF>]

This command clears one or more Runs Counters and issues a query to update the Runs Counter displays.

- The counter for Recipe 0 is a user-defined counter. It increments whenever any any of the other counters increment and may be cleared individually by the user.
- A command argument (xx) ranging from 00 to 31 clears a particular function Runs Counter.
- A command argument (xx) ranging from 32 to 99 clears all Runs Counter including the User-Defined Runs Counter.

5.1.3. Query Usage Timers [<STX>ddiiiiw<LF>]

This command issues a query to update the Usage Timer displays.

5.1.4. Clear Usage Timers & Query [<STX>ddiiiiWxx<LF>]

This command clears one or more Usage Timers Counters and issues a query to update the Usage Timer displays.

- A command argument (xx) ranging from 0 to 31 clears a particular output function timer.
- A command argument (xx) ranging from 32 to 47 clears a particular digital input timer.
- A command argument (xx) of 48 clears the Process Timer.
- A command argument (xx) of 49 clears the User-Defined Timer.
- A command argument (xx) ranging from 50 to 99 clears all output function timers, all digital input timers, the Process Timer and the User-Defined Timer.

6. HOST-TO-TYMKON RECIPE DOWNLOAD MESSAGES

Downloading a recipe file from a computer to a Tymkon involves 258 (more or less) separate message transactions. The format of the individual messages involved in the transfer is described below:

Message Format	Bytes Req'd	Message to Tymkon	Reply from Tymkon	Effect on Download Mode & EEROM STORE/RECALL
<STX>ddiiiib<LF>	9	Prepare for download (Overwrite)	Simple Status	Enter Download Mode
<STX>ddiiiiB<LF>	9	Prepare for download (Clear all)	Simple Status	Enter Download Mode
<STX>ddiiiiEss[0:79]<LF>	64 x 91	Process Segment download	Simple Status	None
<STX>ddiiiiTtt[0:31]<LF>	64 x 43	Temperature Segment download	Simple Status	None
<STX>ddiiiiNss[0:15]<LF>	64 x 27	Process Segment Names download	Simple Status	None
<STX>ddiiiiCrr[0:15]<LF>	32 x 27	Recipe Names download	Simple Status	None
<STX>ddiiiiYrrcc[0:15]<LF>	32 x ? x 29	Cycle download (As Req'd)	Simple Status	None
<STX>ddiiiiF[0:63]<LF>	73	File Name & Timestamp download	Simple Status	Exit Download Mode & STORE

Key to Message Tables

dd Device ID (00 to 99)
iiii Serial Tag (0000 to 9999 or any ASCII characters)
ss process segment identifier (00 to 63)
tt temperature segment identifier (00 to 63)
rr recipe identifier (00 to 31)
cc cycle identifier (00 to 63)
nn last cycle (00 to 63)
xx usage timer to clear
xx = 00 to 15 for Digital Inputs,
xx = 16 to 47 for Output Functions,
xx = 48 for total recipe time
xx = 49 for master timer
xx = 49 or greater, clears all

6.1. Host-to-Tymkon Download Message Overview

In general, a download may follow two different approaches:

6.1.1. Download without Memory Clear

A complete database file download involves clearing the entire area of Tymkon RAM used for recipe storage and file ID storage and, then, downloading a comlpete set of replacement data including a file identifier for tracking purposes. In some applications, clearing memory in advance of the download may cause all of the outputs to be turned off for several seconds during the interval after the memory is cleared and before the replacement data arrives.

The other approach is to avoid clearing memory in advance of the download and, instead, overwriting existing data with new data. This approach avoids unwanted glitches in output status, but requires the host software to be more deligent in insuring a complete, successful overwrite. ITS host software uses this 'overwrite' approach, but the 'memory clear' command is provided for independent software developers who would prefer that approach.

Following is a listing of the overwrite download procedure:

Check to insure that the Tymkon is in the proper state to accept a download (Cycle 0, Program Key in place and not in a Program Mode).

Send a <STX>ddiiiib<LF> command to clear the File ID and put the Tymkon in a Download Mode. ***Note the lowercase 'b' in the command that causes only the File ID, <u>not</u> the recipes, to be cleared.**

Download each of the (64) Process Segments using the <STX>ddiiiiEss[0:79]<LF> command for each Process Segment.

Download each of the (64) Temperature Segments using the <STX>ddiiiiTtt[0:31]<LF> command for each Temperature Segment. ***Note: This section is optional. Tymkons that control a single temperature zone do not require a Temperature Segment table.**

Download each of the (64) Process Segment Names using the <STX>ddiiiiNss[0:15]<LF> command for each Process Segment Name.

Download each of the (32) Recipe Names using the <STX>ddiiiiCrr[0:15]<LF> command for each Recipe Name.

Download each of the (32) Recipes in any order (preferably in order from 0 to 31). Send one cycle at a time beginning with Cycle 0 using the <STX>ddiiiiYrrcc[0:15]<LF> command. The Tymkon supports a total of (64) cycles per recipe, but unused cycles at the end of a recipe need not be downloaded. As each cycle is downloaded, the Tymkon sets the 'Last Cycle' variable to the number of the current cycle. When Cycle 0 of any recipe is downloaded, the Tymkon additionally clears the remainder of that recipe in preparation for the remaining new cycle definitions.

Download the File ID using the <STX>ddiiiiF[0:63]<LF> command.

This entire download proceedure requires the transfer of 27,995 bytes from the computer to the Tymkon and 9,030 bytes from the Tymkon to the computer. At 9 bits per character and 115,200 baud, transferring these 333,225 bits takes about 2.9 seconds. However, since both the Tymkon CPU and the computer have other responsibilities during the transfer, the total transfer time approaches 30 seconds.

With this transfer approach, the sequence in which the messages are sent is irrelevant. In addition, complete messages may be omitted. For example, unused recipes need not be sent or the entire Temperature Segment table may be omitted. The only requirement is that the Memory Clear command must be issued first to prepare the Tymkon for downloads.

Note that the File ID may be used to identify to the host what database file currently resides in Tymkon memory.

Most commands not related to the download function cause the Tymkon to exit the Download Mode. In some cases, sending a non-download command causes the Tymkon to recall the previous contents of the EEROM, while other commands save whatever data has been transferred. For example, sending the File ID command terminates the Download mode and issues a STORE request to the EEROM, while sending a Reset command during a download terminates the Download mode and issues a RECALL request to the EEROM (see Node Computer-to-Tymkon Messages).

6.1.2. Download with Memory Clear

The primary difference with this approach is that Tymkon memory is cleared before the downlaod begins.

Check to insure that the Tymkon is in the proper state to accept a download (Cycle 0, Program Key in place and not in a Program Mode).

Send a <STX>ddiiiiB<LF> command to clear all of memory and put the Tymkon in a Download Mode. ***Note the uppercase 'B' in the command that causes both the File ID and the recipes to be cleared.**

Download any subset of the segment tables, names or recipes set as described above.

Optionally download a File ID.

Issue any command to the Tymkon that initiates an EEROM STORE (see Host-to-Tymkon Messages).

6.2. Host-to-Tymkon Download Message Details

6.2.1. Process Segment download, <STX>ddiiiiEss[0:79]<LF>

ss=process segment identifier (00 to 63)

8 bytes – Digital Outputs (4 outputs per character)
4 bytes – Digital Inputs (4 inputs per character)
4 bytes – Flags
64 bytes – (32) Analog Setpoints

```
001   Digital Output Mask 31 - 28, Nibble 7, "0123456789:;<=>?"
002   Digital Output Mask 27 - 24,    "    6,
003   Digital Output Mask 23 - 20,    "    5,
004   Digital Output Mask 19 - 16,    "    4,
005   Digital Output Mask 15 - 12,    "    3,
006   Digital Output Mask 11 - 08,    "    2,
007   Digital Output Mask 07 - 04,    "    1,
008   Digital Output Mask 03 - 00,    "    0,

009   Digital Input Mask 15 - 12, Nibble 3, "0123456789:;<=>?"
010   Digital Input Mask 11 - 08,    "    2,
011   Digital Input Mask 07 - 04,    "    1,
012   Digital Input Mask 03 - 00,    "    0,

009   "0123456789:;<=>?", 4 bits available, defaults to "0"
010   "0123456789:;<=>?", 4 bits available, defaults to "0"
011   "0123456789:;<=>?", x1xx = Segment Alarm ("4")
012   "0123456789:;<=>?", 4 bits available, defaults to "0"

013 to 080   see Detailed Status for byte-order of analog setpoints
```

6.2.2. Temperature Segment download, <STX>ddiiiiTtt[0:31]<LF>

tt = temperature segment identifier (00 to 63)

Tymkon software supports a Temperature segment table that operates much like the Process Segment table. The Temperature Segment Table consists of (64) rows of (8) 4-digit values per row. Many Tymkon versions continue to use a single temperature setpoint value per cycle and do not require a Temperature Segment Table.

All Temperature values are processed the same whether their distination is the Temperature Segment Table or the Temperature field of Cycle data. Each value is saved in two 8-bit bytes of Tymkon memory. These two bytes hold three single-bit flags and a value ranging from 0 to 1999.

When temperature values are transmitted from the Host to the Tymkon, they are broken up into four 4-bit nibbles which are then converted to four ASCII characters by adding 48 (30h) to the 4-bit value.

Each two-byte temperature value is formatted as follows:

Bits: [15 14 13 12][11 10 9 8][7 6 5 4][3 2 1 0]

Bits	Purpose
15	1 indicates 'Value Present', 0 indicates Null setpoint
14	1 indicates 'Profile', 0 indicates 'Spike'
13	1 indicates Negative value, 0 indicates Positive value
12	**1**999 Contains first digit of setpoint (0 or 1)
11 to 8	1**9**99 Contains second BCD digit of setpoint (0 to 9)
7 to 4	19**9**9 Contains third BCD digit of setpoint (0 to 9)
3 to 0	199**9** Contains fourth BCD digit of setpoint (0 to 9)

6.2.3. Process Segment Names download, <STX>ddiiiiNss[0:15]<LF>

ss = process segment identifier (00 to 63)

The 16-byte content of this message contains a single Precess Segment name. A Segment name may contain any 7-bit ASCII character excluding the ASCII control characters. Only the first (8) characters of Process Segment names are displayed on existing Tymkon models, but all (16) characters will be displayed on future models.

6.2.4. Recipe Names download, <STX>ddiiiiCrr[0:15]<LF>

rr = recipe identifier (00 to 31)

The 16-byte content of this message contains a single recipe name. A Recipe name may contain any 7-bit ASCII character excluding the ASCII control characters. Recipe Names are not displayed on existing Tymkon models, but will be on future models.

6.2.5. Cycle download (As Req'd), <STX>ddiiiiYrrcc[0:15]<LF>

rr = recipe identifier (00 to 31)
cc = cycle identifier (00 to 63)

2 bytes – Process Segment
2 bytes – Branch Cycle or Auxiallary Command argument
4 bytes – Time This Cycle
2 bytes – Flags (Cycle Alarm, Time base etc – see below)
4 bytes – Temperature (see "Temperature Segment download" above)
2 bytes – unused

Flags, MSB Byte

Bit 7 = 0 set to '0' to force a valid ASCII character
Bit 6 = 1 set to '1' to force a valid ASCII character
Bit 5 = unused
Bit 4 = unused
Bit 3 = unused
Bit 2 = unused
Bit 1 = unused
Bit 0 = unused

Flags, LSB Byte

Bit 7 = 0 set to '0' to force a valid ASCII character
Bit 6 = 1 set to '1' to force a valid ASCII character
Bit 5 = unused
Bit 4 = unused
Bit 3 = unused
Bit 2 = 1 indicates Programmable Cycle Alarm
Bit 1 = 1 indicates Minutes time base
Bit 0 = 1 indicates Seconds time base

6.2.6. File Name & Timestamp download, <STX>ddiiiiF[0:63]<LF>

The 64-byte content of this message may be used by Host software to track recipe downloads. This message may contain any 7-bit ASCII character excluding the ASCII control characters.

7. TYMKON-TO-HOST MESSAGES

7.1. Tymkon-to-Host Message Overview

There are currently seven Tymkon-to-Node Computer message types defined. These include a "Simple Status Reply", a "Detailed Status Reply", an "Extended Status Reply", a "Temperature" reply, a "Version and Configuration Reply", a "Runs Counter" reply and an "Output Timer" reply.

- (8) + Simple Status (28) + (1) = 37
- (8) + TimeStamp (11) + Detailed Status (184) + Simple Status (28) + (1) = 232
- (8) + TimeStamp (11) + Extended Status (288) + Simple Status (28) + (1) = 336
- (8) + TimeStamp (11) + Temperatures (64) + Simple Status (28) + (1) = 112
- (8) + TimeStamp (11) + Version (208) + (1) = 228
- (8) + TimeStamp (11) + Runs Counters (128) + (1) = 148
- (8) + TimeStamp (11) + Usage Timers (400) + (1) = 420

7.1.1. Simple Status Reply

The Simple Status Reply contains an (8) byte header as previously defined, (28) bytes of Simple Status data and a (1) byte terminator for a total message length of (37) characters.

```
001   <SOH>
002   Device ID, MSD
003   Device ID, LSD
004   Serial Tag, MSD
005   Serial Tag
006   Serial Tag
007   Serial Tag, LSD
008   "S" (qualifier)
      <28 bytes Simple Status inserted here>
037   <CR> (terminator)
```

7.1.2. Detailed Status Reply (Obsolete – replaced by Extended Status)

The Detailed Status Reply contains an (8) byte header as previously defined, (11) bytes of timestamp data, (184) bytes of Detailed Status data, (28) bytes of Simple Status data and a (1) byte terminator for a total message length of (232) characters.

```
001   <SOH>
002   Device ID, MSD
003   Device ID, LSD
004   Serial Tag, MSD
005   Serial Tag
006   Serial Tag
007   Serial Tag, LSD
008   "D" (qualifier)
      <11 byte Timestamp inserted here>
      <184 bytes Detailed Status inserted here>
      <28 bytes Simple Status inserted here>
232   <CR> (terminator)
```

7.1.3. Extended Status Reply

The Extended Status Reply contains an (8) byte header as previously defined, (11) bytes of timestamp data, (184) bytes of Detailed Status data, (104) bytes Extended Temperature, (28) bytes of Simple Status data and a (1) byte terminator for a total message length of (336) characters.

```
001   <SOH>
002   Device ID, MSD
003   Device ID, LSD
004   Serial Tag, MSD
005   Serial Tag
006   Serial Tag
007   Serial Tag, LSD
008   "d" (qualifier)
      <11 byte Timestamp inserted here>
      <184 bytes Detailed Status inserted here>
      <104 bytes Extended Temperature inserted here>
      <28 bytes Simple Status inserted here>
336   <CR> (terminator)
```

7.1.4. Temperature Reply (Obsolete – replaced by Extended Status)

The Temperature Reply contains an (8) byte header as previously defined, (11) bytes of timestamp data, a (64) byte temperature table, (28) bytes of Simple Status data and a (1) byte terminator for a total message length of (112) characters.

```
001   <SOH>
002   Device ID, MSD
003   Device ID, LSD
004   Serial Tag, MSD
005   Serial Tag
006   Serial Tag
007   Serial Tag, LSD
008   "K" (qualifier)
      <11 byte Timestamp inserted here>
      <64 bytes Temperature Status inserted here>
      <28 bytes Simple Status inserted here>
112   <CR> (terminator)
```

7.1.5. Version and Configuration Reply

The Version and Configuration Reply contains (24) bytes of software version information, (80) bytes of configuration data, (64) bytes of File info, and (9) bytes of overhead for a total message length of (177) characters.

```
001   <SOH>
002   Device ID, MSD
003   Device ID, LSD
004   Serial Tag, MSD
005   Serial Tag
006   Serial Tag
007   Serial Tag, LSD
008   "V" (qualifier)
      <11 byte Timestamp inserted here>
      <24 byte product name and configuration number inserted here>
```

```
      <8 byte product code and rev level>
      <80 bytes I/O parameters inserted here>
      <64 bytes file name and timestamp inserted here>
      <32 bytes Equipment Unique Identifier>
228   <CR> (terminator)
```

7.1.6. Runs Counters Reply

The Runs Counters Reply contains an (8) character header as previously defined, (11) characters of timestamp data, a (128) character counter table and a (1) character terminator for a total message length of (232) characters.

```
001   <SOH>
002   Device ID, MSD
003   Device ID, LSD
004   Serial Tag, MSD
005   Serial Tag
006   Serial Tag
007   Serial Tag, LSD
008   "U" (qualifier)
      <11 byte Timestamp inserted here>
      <128 character Runs Counter Table inserted here>
148   <CR> (terminator)
```

7.1.7. Usage Timers Reply

The Output Timers Reply contains an (8) byte header as previously defined, (11) characters of timestamp data, a (392) character timer table and a (1) character terminator for a total message length of (412) characters.

```
001   <SOH>
002   Device ID, MSD
003   Device ID, LSD
004   Serial Tag, MSD
005   Serial Tag
006   Serial Tag
007   Serial Tag, LSD
008   "W" (qualifier)
      <11 byte Timestamp inserted here>
      <400 character Usage Timer Table inserted here>
412   <CR> (terminator)
```

7.2. Tymkon-to-Host Message Details

7.2.1. Structure of Timestamp Message

```
001 to 004  Day of Year (dddd, 0000 to 9999 day counter)
005 to 006  Hours (hh, 00 to 23)
007 to 008  Minutes (mm, 00 to 59)
009 to 010  Seconds (ss, 00 to 59)
011         Tenths of Seconds (0.0 to 0.9)
```

7.2.2. Structure of Detailed Status Message

```
001   Digital Output Status 31 - 28, Nibble 7, "0123456789:;<=>?"
002   Digital Output Status 27 - 24,   "    6,
003   Digital Output Status 23 - 20,   "    5,
004   Digital Output Status 19 - 16,   "    4,
005   Digital Output Status 15 - 12,   "    3,
006   Digital Output Status 11 - 08,   "    2,
007   Digital Output Status 07 - 04,   "    1,
008   Digital Output Status 03 - 00,   "    0,

009   Digital Output Faults 31 - 28, Nibble 7, "0123456789:;<=>?"
010   Digital Output Faults 27 - 24,   "    6,
011   Digital Output Faults 23 - 20,   "    5,
012   Digital Output Faults 19 - 16,   "    4,
013   Digital Output Faults 15 - 12,   "    3,
014   Digital Output Faults 11 - 08,   "    2,
015   Digital Output Faults 07 - 04,   "    1,
016   Digital Output Faults 03 - 00,   "    0,

017   Digital Input Status 15 - 12, Nibble 3, "0123456789:;<=>?"
018   Digital Input Status 11 - 08,   "    2,
019   Digital Input Status 07 - 04,   "    1,
020   Digital Input Status 03 - 00,   "    0,

021   Digital Input Mask 15 - 12, Nibble 3, "0123456789:;<=>?"
022   Digital Input Mask 11 - 08,   "    2,
023   Digital Input Mask 07 - 04,   "    1,
024   Digital Input Mask 03 - 00,   "    0,
```

```
025   Aout 31,  9x          057   Aout 15,  9x
026   Aout 31,  x9          058   Aout 15,  x9
027   Aout 30,  9x          059   Aout 14,  9x
028   Aout 30,  x9          060   Aout 14,  x9
029   Aout 29,  9x          061   Aout 13,  9x
030   Aout 29,  x9          062   Aout 13,  x9
031   Aout 28,  9x          063   Aout 12,  9x
032   Aout 28,  x9          064   Aout 12,  x9
033   Aout 27,  9x          065   Aout 11,  9x
034   Aout 27,  x9          066   Aout 11,  x9
035   Aout 26,  9x          067   Aout 10,  9x
036   Aout 26,  x9          068   Aout 10,  x9
037   Aout 25,  9x          069   Aout 09,  9x
038   Aout 25,  x9          070   Aout 09,  x9
039   Aout 24,  9x          071   Aout 08,  9x
040   Aout 24,  x9          072   Aout 08,  x9
041   Aout 23,  9x          073   Aout 07,  9x
042   Aout 23,  x9          074   Aout 07,  x9
043   Aout 22,  9x          075   Aout 06,  9x
044   Aout 22,  x9          076   Aout 06,  x9
045   Aout 21,  9x          077   Aout 05,  9x
046   Aout 21,  x9          078   Aout 05,  x9
047   Aout 20,  9x          079   Aout 04,  9x
048   Aout 20,  x9          080   Aout 04,  x9
049   Aout 19,  9x          081   Aout 03,  9x
050   Aout 19,  x9          082   Aout 03,  x9
051   Aout 18,  9x          083   Aout 02,  9x
052   Aout 18,  x9          084   Aout 02,  x9
053   Aout 17,  9x          085   Aout 01,  9x
054   Aout 17,  x9          086   Aout 01,  x9
055   Aout 16,  9x          087   Aout 00,  9x
056   Aout 16,  x9          088   Aout 00,  x9
```

089	Ainp 31,	9x.x	137	Ainp 15,	9x.x
090	Ainp 31,	x9.x	138	Ainp 15,	x9.x
091	Ainp 31,	xx.9	139	Ainp 15,	xx.9
092	Ainp 30,	9x.x	140	Ainp 14,	9x.x
093	Ainp 30,	x9.x	141	Ainp 14,	x9.x
094	Ainp 30,	xx.9	142	Ainp 14,	xx.9
095	Ainp 29,	9x.x	143	Ainp 13,	9x.x
096	Ainp 29,	x9.x	144	Ainp 13,	x9.x
097	Ainp 29,	xx.9	145	Ainp 13,	xx.9
098	Ainp 28,	9x.x	146	Ainp 12,	9x.x
099	Ainp 28,	x9.x	147	Ainp 12,	x9.x
100	Ainp 28,	xx.9	148	Ainp 12,	xx.9
101	Ainp 27,	9x.x	149	Ainp 11,	9x.x
102	Ainp 27,	x9.x	150	Ainp 11,	x9.x
103	Ainp 27,	xx.9	151	Ainp 11,	xx.9
104	Ainp 26,	9x.x	152	Ainp 10,	9x.x
105	Ainp 26,	x9.x	153	Ainp 10,	x9.x
106	Ainp 26,	xx.9	154	Ainp 10,	xx.9
107	Ainp 25,	9x.x	155	Ainp 09,	9x.x
108	Ainp 25,	x9.x	156	Ainp 09,	x9.x
109	Ainp 25,	xx.9	157	Ainp 09,	xx.9
110	Ainp 24,	9x.x	158	Ainp 08,	9x.x
111	Ainp 24,	x9.x	159	Ainp 08,	x9.x
112	Ainp 24,	xx.9	160	Ainp 08,	xx.9
113	Ainp 23,	9x.x	161	Ainp 07,	9x.x
114	Ainp 23,	x9.x	162	Ainp 07,	x9.x
115	Ainp 23,	xx.9	163	Ainp 07,	xx.9
116	Ainp 22,	9x.x	164	Ainp 06,	9x.x
117	Ainp 22,	x9.x	165	Ainp 06,	x9.x
118	Ainp 22,	xx.9	166	Ainp 06,	xx.9
119	Ainp 21,	9x.x	167	Ainp 05,	9x.x
120	Ainp 21,	x9.x	168	Ainp 05,	x9.x
121	Ainp 21,	xx.9	169	Ainp 05,	xx.9
122	Ainp 20,	9x.x	170	Ainp 04,	9x.x
123	Ainp 20,	x9.x	171	Ainp 04,	x9.x
124	Ainp 20,	xx.9	172	Ainp 04,	xx.9
125	Ainp 19,	9x.x	173	Ainp 03,	9x.x
126	Ainp 19,	x9.x	174	Ainp 03,	x9.x
127	Ainp 19,	xx.9	175	Ainp 03,	xx.9
128	Ainp 18,	9x.x	176	Ainp 02,	9x.x
129	Ainp 18,	x9.x	177	Ainp 02,	x9.x
130	Ainp 18,	xx.9	178	Ainp 02,	xx.9
131	Ainp 17,	9x.x	179	Ainp 01,	9x.x
132	Ainp 17,	x9.x	180	Ainp 01,	x9.x
133	Ainp 17,	xx.9	181	Ainp 01,	xx.9
134	Ainp 16,	9x.x	182	Ainp 00,	9x.x
135	Ainp 16,	x9.x	182	Ainp 00,	x9.x
136	Ainp 16,	xx.9	184	Ainp 00,	xx.9

Note:

- The decimal points are not included in the transmission.
- If an analog input is >99.9, the first character sent is ">"; the remaining two characters may or may not represent actual data. For example ">07" may represent a value of 100.7 or it might just be garbage.

7.2.3. Structure of Simple Status Message

```
001   Temperature Setpoint, 9xxx
002   Temperature Setpoint, x9xx
003   Temperature Setpoint, xx9x
004   Temperature Setpoint, xxx9

005   Temperature Actual, 9xxx
006   Temperature Actual, x9xx
007   Temperature Actual, xx9x
008   Temperature Actual, xxx9

009   Recipe, "0x" to "3x", high nibble
010   Recipe, "x0" to "x1", low nibble

011   Cycle, "0x" TO "9x", high nibble
012   Cycle, "x0" TO "x9", LOW nibble

013   Segment "0x" TO "9x", high nibble
014   Segment "x0" TO "x9", LOW nibble

015   Time This Cycle, 9xx.x
016   Time This Cycle, x9x.x
017   Time This Cycle, xx9.x
018   Time This Cycle, xxx.9

019   Total Time Remaining, 9x:xx:xx
020   Total Time Remaining, x9:xx:xx
021   Total Time Remaining, xx:5x:xx
022   Total Time Remaining, xx:x9:xx
023   Total Time Remaining, xx:xx:5x
024   Total Time Remaining, xx:xx:x9

025   Status Flag byte 1, see below
026   Status Flag byte 2, see below
027   Status Flag byte 3, see below
028   Status Flag byte 4, see below
```

Status Flag Byte 1

```
Bit 7 = 0 set to ‘0’ to force a valid ASCII character
Bit 6 = 1 set to ‘1’ to force a valid ASCII character
Bit 5 = 1 when Tymkon is in a Program Mode
Bit 4 = 1 when Tymkon is in count-up cycle at end of a recipe (EOC)
Bit 3 = 1 when Time Base bit is set indicating non-default time base
Bit 2 = 1 when Tymkon is Reset
Bit 1 = 1 when Tymkon is in Hold Mode
Bit 0 = 1 when Tymkon is in Manual Abort Mode
```

Status Flag Byte 2

```
Bit 7 = 0 set to ‘0’ to force a valid ASCII character
Bit 6 = 1 set to ‘1’ to force a valid ASCII character
Bit 5 = 1 negative acknowledgement of previous command
Bit 4 = 1 when Tymkon key is in Program position
Bit 3 = 1 when an analog or digital HOLD input is unsafe
Bit 2 = 1 when an analog or digital WAIT input is unsafe
Bit 1 = 1 when an analog or digital LOCK input is unsafe
Bit 0 = 1 when an analog or digital BUZZ input is unsafe
```

Status Flag Byte 3

```
Bit 7 = 0 set to ‘0’ to force a valid ASCII character
Bit 6 = 1 set to ‘1’ to force a valid ASCII character
Bit 5 = 1 indicates center zone is ‘Spike/Process’ capable
Bit 4 = 1 indicates center zone control mode is ‘Process T/C’
Bit 3 = 1 Power Fail – request to reset Tymkon time-of-day clock
Bit 2 = 1 End-Of-Process Alarm (also see Status Flag Byte 1, Bit 4)
Bit 1 = 1 Programmable Cycle or Segment Alarm
Bit 0 = 1 indicates the FileID in Tymkon memory has been altered
```

Status Flag Byte 4

```
Bit 7 = 0 set to ‘0’ to force a valid ASCII character
Bit 6 = 1 set to ‘1’ to force a valid ASCII character
Bit 5 = 1 indicates Eurotherm’s <NAK>’s or Temperature ILK
Bit 4 = 1 indicates waiting at end-of-cycle before Wait Alarm
Bit 3 = 1 indicates single temp zone with single T/C
Bit 2 = 1 indicates Wait Alarm (wait has timed out)
Bit 1 = 1 unused
Bit 0 = 1 unused
```

7.2.4. Structure of Temperature Message (Obsolete)

```
001 to 008  Z7 Setpoint (4 bytes) and Actual (4 bytes)
009 to 016  Z6 Setpoint (4 bytes) and Actual (4 bytes)
017 to 024  Z5 Setpoint (4 bytes) and Actual (4 bytes)
025 to 032  Z4 Setpoint (4 bytes) and Actual (4 bytes)
033 to 040  Z3 Setpoint (4 bytes) and Actual (4 bytes)
041 to 048  Z2 Setpoint (4 bytes) and Actual (4 bytes)
049 to 056  Z1 Setpoint (4 bytes) and Actual (4 bytes)
057 to 064  Z0 Setpoint (4 bytes) and Actual (4 bytes)
```

7.2.5. Structure of Extended Temperature Message

```
001 to 013  Z7 FLG/Set(4 bytes)/Spk(4 bytes)/Pro(4 bytes)
014 to 026  Z6 FLG/Set(4 bytes)/Spk(4 bytes)/Pro(4 bytes)
027 to 039  Z5 FLG/Set(4 bytes)/Spk(4 bytes)/Pro(4 bytes)
040 to 052  Z4 FLG/Set(4 bytes)/Spk(4 bytes)/Pro(4 bytes)
053 to 065  Z3 FLG/Set(4 bytes)/Spk(4 bytes)/Pro(4 bytes)
066 to 078  Z2 FLG/Set(4 bytes)/Spk(4 bytes)/Pro(4 bytes)
079 to 091  Z1 FLG/Set(4 bytes)/Spk(4 bytes)/Pro(4 bytes)
092 to 104  Z0 FLG/Set(4 bytes)/Spk(4 bytes)/Pro(4 bytes)

FLG = ASCII character indicating that setpoint is S/P/+/-
FLG = +/- is used when analog ramper board is present
FLG = ? if no setpoint is set
Spk = Thermocouple 1: Actual Spike T/C
Pro = Thermocouple 2: Actual Process/Profile T/C
```

7.2.6. Structure of Version and Configuration Message

```
001 to 008  Configuration Number, ex: '800-0420'
009 to 016  Configuration Date, ex: '01/02/99'
017 to 024  Product Name, ex: ' TYMKON '
025 to 032  Product Code & Rev level

033 to 048  16 Digital Inputs x 1 byte per input (15 thru 0)
```

Digital Inputs = 01f0blwh (characters 025 to 040):
- f = Function exists
- blwh = Buzz, Lock, Wait, Hold

```
049 to 112  32 Output Functions x 2 bytes per function (31 thru 0)
```

Output Functions, 1st byte of 2 byte pair:= 01fxyddd
- f = Function exists
- x = Analog Input,
- y = Analog Output,
- ddd: 000='A99/T99%', 001='99.9/99%', 010=' 999/99%', 011='9.99/99%'

Output Functions, 2nd byte of 2 byte pair = 01ulblwh
- u = Upper Limit,
- l = Lower Limit,
- blwh = Buzz, Lock, Wait, Hold

```
113 to 176  64 byte file name and timestamp
177 to 208  Equipment Unique Identifier
```

7.2.7. Structure of Runs Counter Message

```
001 to 004  Number of Runs completed using Recipe 31
005 to 008  Number of Runs completed using Recipe 30
009 to 012  Number of Runs completed using Recipe 29
   etc
117 to 120  Number of Runs completed using Recipe 2
121 to 124  Number of Runs completed using Recipe 1
125 to 128  Total Number of Runs completed since cleared
```

7.2.8. Structure of Usage Timers Message

```
001 to 008  Master Timer (range: 99,000,000 seconds per timer)
009 to 016  Recipe Timer     (all timers >3 year range)
017 to 024  Output Timer, Function 31
025 to 032  Output Timer, Function 30
   etc
265 to 272  Output Timer, Function 00
273 to 280  Input Timer, Function 15
   etc
392 to 400  Input Timer, Function 00
```