The HX5MT Ultrasonic Positioning Component

The HX5MT has all the features of the older HX5M1, it also has the main features of the HX5T. It can be set up to be one or the other. The HX5MT replaces the HX5M1, and can operate in asynchronous mode. It can track the position of any HX5T transmitting tags. It can function with the HX5NC network controller, and provide the XYZ coordinate program and other HX5 programs with required data for 3D asynchronous tracking. When the HX5MT is shipped it is configured as a slave for the network controller, no configuration is required for the 3D asynchronous tracking. The HX5MT can also operate as synchronized clusters of tags. This allows the HX5MT to operate with other tags in close proximity without any overrun problems. If plenty of power is available, this feature makes guidance for autonomous apparatus more feasible.

But the HX5MT has many other features, features that allow it to work in varying degrees of synchronous modes as well. This document attempts to explain the operation of the HX5MT.

HX5MT Operating Modes

The older versions HX5M works in only one operating mode. This mode is Slave Receiving Mode (program control byte = 00) and the termination byte is fixed a # and a carriage return. The HX5MT has many modes, receiving and transmitting.

HXMT Receiver Modes

Slave Receiving Mode
Program Control byte = 00

In this mode the device listens for sonic tag id signals. When these signals are received; the time elapsed from synchronization until the specific tag id is received, is appended to the identification number of the tag and saved in a memory ring buffer. The HX5MT ring buffer can hold up to 30 tag ids, and must be emptied before it overruns. The time elapsed is a 24 bit number in hexadecimal format.

Master Synchronizer Mode
Program Control Byte = 01

When bit 0 of the program control byte is set, the device becomes a master. The master synchronizes all other devices on the same network. Only one master can run the network, if the PC needs to configure the network, it must terminate the master. The master transmits ASCII character $ followed by an ASCII numeric character. The numeric character has no significance other than to help identify the acquisition cycle. It helps the data monitor (PC) determine if any acquisition cycle has been missed. Given that the termination word has been set to # (ASCII code 35) and space (ASCII code 32), the following is what the master synchronizer transmits over the network.

$1# $2# $3# $4# $5# $6# $7# $8# $9# $0# $1# $2# $3# ……CONTINUES

NOTE: in Master Synchronizer mode the HX5MT is still receiving data just like it does in slave receiving mode.

Timers are cleared upon the reception of the first start bit following $

Master Synchronizer plus Output Mode
Program Control Byte = 03

The HX5MT transmits $ followed by a numeric character between 0 and 9. When the acquisition cycle ends, the HX5MT transmits the entire contents of the ring buffer. And by monitoring the data using a serial terminal program on a PC, the example string may be encountered.

Example: Given that the termination word is #&. And given that, tag 03CC transmitted two times during the acquisition cycle and tag 01A2 transmitted once during the acquisition cycle; the results might look like follows:

$3 03CC003A66 01A201453B 03CC01F201 #&

First letter is the synchronization byte followed by the acquisition cycle number. This is followed by the tag id (16 bits), and the appended (24 bits) time elapsed from last synchronization.

HX5MT Transmitter Modes

Transmits Tag ID at the end of Interval Count
Program Control Byte = 04

At the end of the interval count determined by the Interval Byte in the work registers, the tag will transmit it's id.

Transmits Synchronization and Tag ID
Program Control Byte = 05

Outputs the synchronization signal onto the network, and transmits tag id at the beginning of the interval count. This process is repeated at the end of the interval count, and the repetition is determined by the Interval Byte in the work registers.

Boot Up (online reset)

On startup the device loads the work registers with program parameters from EEPROM. It looks at the checksum to verify that the parameters are unscrambled. If for some reason the parameters and the check sum don't correspond, the device loads default parameters into the work registers.

Default Parameters

Program Control Byte = 00
Termination Byte High = 26 (Number Sign #)
Termination Byte Low = 0D (Carriage Return)
Interval Byte = FF

Note that the controller also checks the Program Control Byte. If it is higher than 5 the default parameters are loaded.

Communication and Control

Characters $, %, & and ASCII code 27 (ESCAPE) are control characters and should never be used on the network for other purposes. Device addresses are binary in nature consisting of two bytes running from 0 to 255. The binary values 36, 37 and 38 will never occur in device addresses. All other data is in hexadecimal form binary values 48 through 57 and from 65 through 70.

Character $

This character, will set all devices on the network into readiness to receive synchronization. The synchronization occurs on the start bit of the first character following $. When the synchronization start bit is received, the timers of all devices are set to 0 and the timers will commence incrementing until they high significant byte matches with the Interval Byte in the work registers.

Character % (percent sign)

Typing this character onto the network, will force every HX5MT device to respond with it's own primary address. The address value determines the delay until the device transmits it's own address. HX5MT has addressing span of 32000 unique addresses. Each device is given a slot to transmit 6 characters. Only 5 are transmitted 6^{th} character is timeslot tolerance, so that transmissions don't overrun. Hence 32000 * 6 characters are transmitted on the network, or a total of 192000. If the network is running at 250000 baud, or 25000 characters per second it will take 192000/25000 or roughly 8 seconds to receive all the devices on the network.

Use a terminal program and type % onto the network wait about 8 seconds and all devices on the network will have identified themselves.

Character & (ampersand)

Ampersand will cause all the devices on the network to boot up (online reset).

ASCII code 27 (ESCAPE)

This character will always clear the program control byte, it is mainly used to remove a master from the network so that the PC can transmit control parameters freely on the network.

Entering Parameters into the HX5MT

The HX5MT has a secondary address and a primary address, to help distinguish between data entry and data polling. When the HX5MT receives it's secondary address, it responds with a carriage return and the prompt character > expecting entry to follow. If at this point the characters FF followed by a carriage return (ENTER key) the device will respond by transmitting the content of it's EEPROM containing the control parameters.

At the > the user is invited to enter new control parameters. If a total of 4 hexadecimal bytes followed with the correct checksum are entered, then the data is stored on EEPROM. This means that next time the HX5MT comes up or is re-booted it will use these parameters for operation.

The parameters typed always go directly into the work registers. If carriage return (ENTER key) is received at any time during the typing, the device will resume operation using the new parameters. If on the other hand the ASCII code 27 (ESCAPE key) is typed, the device resumes operation with the work register Program Control Byte cleared.

Parameters which can be entered

1. Program Control Byte = ##
2. Termination Byte High = ##
3. Termination Byte Low = ##
4. Interval Byte = ##

must be one of the following hex character: 0,1,2,3,4,5,6,7,8,9,A,B,C,E,F

Getting Data from the HX5MT

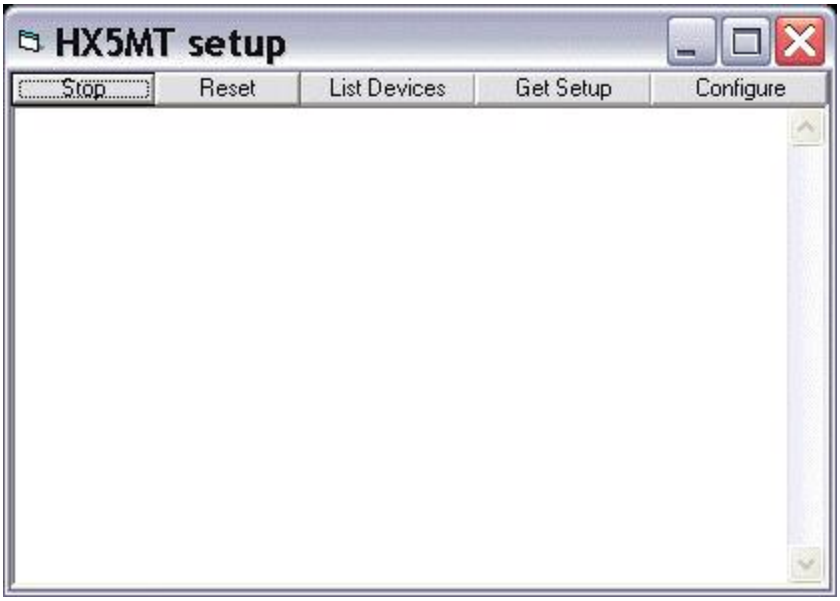
To poll devices for data, all the HX5MT units should be in a slave mode; i.e. not active. It is possible to poll an active device, but that has to be done without conflicting with it's transmissions.

When the device receives it's primary address, it responds by sending all data available in it's round buffer. This would include all not polled or none depleted data. If a long time elapses between polling, the round buffer can overlaps and the data becomes useless. Once data has been withdrawn even if there has been an overlaps operation will become normal again.

The Primary Address is always an even number, one unit lower in value than the Secondary Address. Hence the secondary address must be an odd number. When a HX5MT receives it's Primary Address it responds by transmitting the contents of it's round buffer. When it receives it's Secondary Address it responds with a prompt > for entry mode.

HX5MT Setup Program

The following program HX5MT setup, is shipped with the devices to make configuration a bit easier and can be downloaded of our HX5 utilities web storage.



HX5MT Setup Program Operation

The Get Setup button gets the current control parameters out of the EEPROM of HX5MT devices on the network. It uses a list of these devices ***previously*** obtained using the List Devices button. If no list of devices exists the Get Setup function will reply with an error message.

The Get Setup function will convert the hexadecimal control parameters string, into a more readable decimal string. The decimal version of the parameters is stored on a file called Hx5setup.txt.

Ordinary text editor can be used to change and modify these parameters in the HX5setup.txt file. The Configure function converts the Hx5setup.txt back into hexadecimal representation and sends the modified data back to the corresponding devices.

Data received over the serial lines, is displayed in the white window below the control buttons. The same data is also stored on a file called Hx5output.txt.

Calculating Distance

The mathematical relationship for calculation of distance is as follows:

Let N be the clock count	(obtained from hx5output.txt)
Let C be the speed of sound	(known)
Let I(n) be the interval Byte	(obtained from hx5setup.txt)
Let S be signal capture overhead (97700)	(Given)

*Distance = (N – S – I (n) * 65536) * C / 16000000*

See the application notes for implementation of this relationship

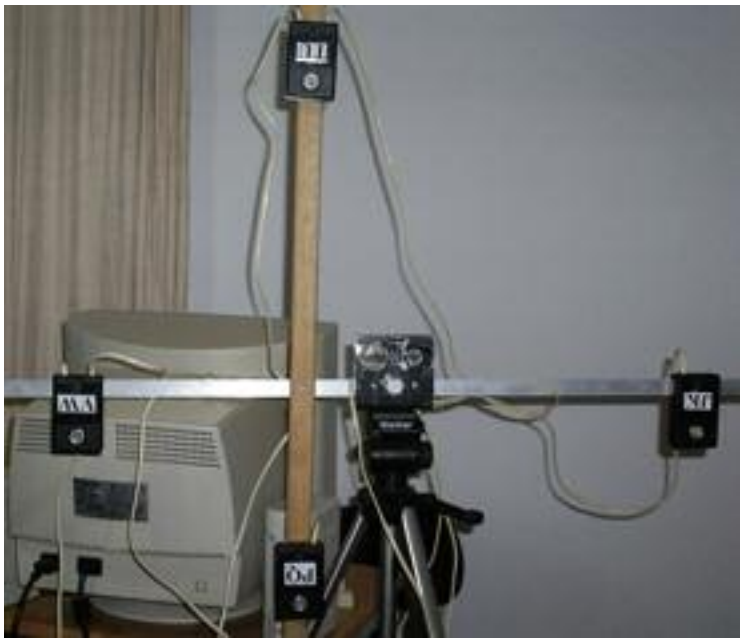
The given value S is not known precisely, it is consistent over time and temperature. It is also a close match from device to device.

For precise measurements the units should be calibrated, and it must be known that speed of sound varies slightly with temperature. Temperature variation is fairly linear.

Speed of sound should be calculated or selected from tables based on temperature; the signal overhead S should be varied to bridge the gab between theory and measurement.

Application example setup

The following example utilizes six HX5MT units total. With 4 units addressed, 8522, 8528, 8532 and 8534 mounted on the fixture shown below.



The 4 units shown above are facing the units shown below placed approximately 5 meters apart for the first data set.

8536 and 8564 were mounted on the structure shown in the illustration above.

Example 1.

The contents of the HX5setup.txt file is as follows:

Hx5setup.txt

DeviceID	PGM CTRL	Termination	Interval
8522	4	221	25
8528	4	222	35
8532	4	220	5
8534	4	219	15
8536	3	8973	130
8564	0	8973	233

The setup file above contains a total of six HX5MT devices, four are set up as transmitters and two are set up as receivers. The termination word for the devices setup as transmitters (PGM CTRL = 4) is 221, 222, 220 and 219. This translates to DD, DE, DC and DB respectively. Device 8564 is set up as a receiving slave; the data is not being called out of it's ring buffer, so it won't have an appearance on the network. Example 3 shows what happens when device 8564 is called.

A different interval byte is associated with each tag. Therefore each tag transmits it's data at different times ensuring there's no overrun. For example id 221 transmits it's id 25 x 65536 clock cycles or 1638400 cycles after receiving the synchronization signal. Similarly tag 220 transmits id 5 x 65536 clock cycles after synchronization. The clock cycle of the HX5MT is 16Mhz at +/- 30 PPM. Hence the tag 220 delays it's transmission for 20.5 milliseconds. This clock cycle delay after synchronization, must be subtracted from 24-bit count result following the tag id in the Hx5output.txt file.

The following is a clip of the contents of the Hx5output.txt file. The file hx5output.txt is the response to the Hx5setup file above. Each data set in the Hx5output.txt contains the hex id of the Tag followed by the number of clock cycles (time) elapsed from sync to the time the respective monitor received the tag id.

Hx5output.txt

```
$000DC0A1A8D 00DB141B50 00DD1E10CD 00DE2817B7 #
$100DC0A1A8C 00DB141B52 00DD1E10CF 00DE281622 #
$200DC0A1A8A 00DB141B51 00DD1E10CD 00DE281626 #
$300DC0A1A89 00DB141B4E 00DD1E10D1 00DE2817B3 #
$400DC0A1A8C 00DB141B4F 00DD1E1261 00DE281620 #
$500DC0A1A8D 00DB141B52 00DD1E1268 00DE281620 #
$600DC0A1A8F 00DB141B51 00DD1E10D2 00DE28161B #
$700DC0A1A8F 00DB141B52 00DD1E10D4 00DE28161C #
$800DC0A1A92 00DB141B57 00DD1E10D4 00DE28161D #
```

To calculate the distance the first data segment in hx5output.txt above can be used. Take 000DC0A1A8D discard the first digit because it enumerates the whole data line. The next four digits are the two tag id (transmitter id) bytes. And the remaining six digits are the three bytes (24 bits), representing the time elapsed from synchronization until the transmission was captured. From the hx5setup.txt tag DC (220) has interval equal to 5.

Let N be the clock count obtained from hx5output
Let C be the speed of sound
Let I(n) be the interval Byte
Let S be signal capture overhead (97700)

Calculation for the distance from master (8536) to tag (220)

N = 0A1A8D (hex) = 662157 (decimal
I(n) = 5

Distance = (N – S – I (n) * 65536) * C / 16000000

Distance = (662157-97700-5*65536)*344(m/s)/16000000
Distance = 5.0907 meters

Program called Hx5parse.exe, converts the Hx5ouput.txt to the format shown in the Hx5parsed.txt file below. It uses the above relationship to calculate the distance, as shown below.

Hx5parsed.txt

```
000.00 > 08536 |  0220 = 05.091  0219 = 05.095  0221 = 05.038  0222 = 05.076
000.53 > 08536 |  0220 = 05.091  0219 = 05.095  0221 = 05.038  0222 = 05.067
001.06 > 08536 |  0220 = 05.091  0219 = 05.095  0221 = 05.038  0222 = 05.067
001.60 > 08536 |  0220 = 05.091  0219 = 05.095  0221 = 05.038  0222 = 05.076
002.13 > 08536 |  0220 = 05.091  0219 = 05.095  0221 = 05.046  0222 = 05.067
002.66 > 08536 |  0220 = 05.091  0219 = 05.095  0221 = 05.046  0222 = 05.067
003.19 > 08536 |  0220 = 05.091  0219 = 05.095  0221 = 05.038  0222 = 05.067
003.73 > 08536 |  0220 = 05.091  0219 = 05.095  0221 = 05.038  0222 = 05.067
004.26 > 08536 |  0220 = 05.091  0219 = 05.096  0221 = 05.038  0222 = 05.067
```

Considerations

The distance from receiver 08536 to tag 0220 is 05.091 meters. The parsing program below leaves out some of the least significant numbers in the Hx5output.txt file. The clock count of the HX5MT is 16MHZ at +/-30PPM. The least significant digit in the Hx5output.txt is 62.5 nano seconds. A change in a single timing digit in Hx5output represents a change of 21.5 micrometers, using speed of sound 344m/s. Looking closer at the data column of tag 222 the data flip flops between 5.067 and 5.076 with deviation approximately 9 mm. This is the truncated actual wavelength of the ultrasonic signal, which is a 40khz wave. The wavelength is 8.6mm, currently there is nothing Hexamite can do to correct this problem (we are working on it). Depending on how the tags are situated, sometimes the receivers will not be able to decide which wave to lock on.

Example 2.

The only difference between Example 1 and Example 2, is that the interval timing is shorter for Example 2. The Master interval setting is 50 meaning 50 x 65536 x 1/16Mhz or 205 milliseconds. All tags must be able to transmit within this period, and because of the signal overhead the last tag cannot signal later than interval 46.

Hx5setup.txt

DeviceID	PGM CTRL	Termination	Interval
8522	4	221	3
8528	4	222	10
8532	4	220	17
8534	4	219	24
8536	3	8973	50
8564	0	8973	233

There isn't any limit to how fast the Master interval setting can be, but the tags that signal outside the interval will disrupt the operation. Hexamite doesn't have exact test results for how far apart the transmission signal can be in terms of time. Successful readings have been had with interval = 5, or 20 mS space between signals. Maximum sampling rate is therefore higher than 20mS (space) + 13mS (signal) = 33 mS or 30 samples per second.

Hx5output.txt

```
$000DD0810DF 00DE0F162E 00DC161AA5 00DB1D1B6B #
$100DD0810E1 00DE0F1630 00DC161AA8 00DB1D1B6A #
$200DD0810DB 00DE0F162D 00DC161AA2 00DB1D1B66 #
$300DD0810DC 00DE0F162B 00DC161AA4 00DB1D1B6A #
$400DD0810DA 00DE0F162F 00DC161AA5 00DB1D1B69 #
$500DD0810DC 00DE0F162C 00DC161AA4 00DB1D1B68 #
$600DD0810DD 00DE0F162F 00DC161AA8 00DB1D1B6B #
$700DD0810DC 00DE0F162F 00DC161AA5 00DB1D1B68 #
$800DD0810DD 00DE0F162E 00DC161AA5 00DB1D1B6B #
$900DD0810DB 00DE0F162C 00DC161AA4 00DB1D1B6A #
$000DD0810DD 00DE0F1630 00DC161AA5 00DB1D1B69 #
```

hx5parsed.txt

```
000.00 > 08536 |  0221 = 05.038  0222 = 05.067  0220 = 05.092  0219 = 05.096
000.20 > 08536 |  0221 = 05.038  0222 = 05.067  0220 = 05.092  0219 = 05.096
000.41 > 08536 |  0221 = 05.038  0222 = 05.067  0220 = 05.092  0219 = 05.096
000.61 > 08536 |  0221 = 05.038  0222 = 05.067  0220 = 05.092  0219 = 05.096
000.82 > 08536 |  0221 = 05.038  0222 = 05.067  0220 = 05.092  0219 = 05.096
001.02 > 08536 |  0221 = 05.038  0222 = 05.067  0220 = 05.092  0219 = 05.096
001.23 > 08536 |  0221 = 05.038  0222 = 05.067  0220 = 05.092  0219 = 05.096
001.43 > 08536 |  0221 = 05.038  0222 = 05.067  0220 = 05.092  0219 = 05.096
001.64 > 08536 |  0221 = 05.038  0222 = 05.067  0220 = 05.092  0219 = 05.096
001.84 > 08536 |  0221 = 05.038  0222 = 05.067  0220 = 05.092  0219 = 05.096
```

Example 3.

The only difference between this configuration and the configuration of example 1, is that the master 8536 calls slave 8564 into action. If like in example 1 a receiving slave is not called, it still gets the $ synchronization signal. The timers are reset and the tag ids and elapsed time is stored in the slave's ring buffer. This data is cleared out of the ring buffer when the slave unit is called. The slave can be polled by call anytime by any master, and the master can be a PC issuing $ synchronization initiation + (dummy character for the start bit) and calling each individual slave for their ring buffers.

Hx5setup.txt

DeviceID	PGM CTRL	Termination	Interval
8522	4	221	3
8528	4	222	10
8532	4	220	17
8534	4	219	24
8536	3	8564	150
8564	0	8973	233

The lines of the hx5output.txt and hx5parse.txt are becoming to wide for the page, therefore a text format version easier to read can be found with this manual.

Hx5output.txt (Better view found on Example3_hx5output.txt)

```
$000DD0746DB 00DE0E4A28 00DC154F37 00DB1C4FBB !t00DE0E478C 00DC154C46 00DB1C4830 00DD074FDE #
$100DD0746D6 00DE0E4A25 00DC154F30 00DB1C4FB8 !t00DE0E478D 00DC154C43 00DB1C46A0 00DD074FDF #
$200DD0746D5 00DE0E4A22 00DC154F30 00DB1C4FB6 !t00DE0E478A 00DC154C44 00DB1C46A0 00DD074FDF #
$300DD0746D7 00DE0E4A22 00DC154F31 00DB1C4FB7 !t00DE0E4788 00DC154C43 00DB1C469F 00DD074FDF #
$400DD0746D8 00DE0E4A25 00DC154F33 00DB1C4FB8 !t00DE0E478B 00DC154C44 00DB1C46A0 00DD074FDE #
$500DD0746D6 00DE0E4A25 00DC154F30 00DB1C4FB6 !t00DE0E478B 00DC154C42 00DB1C469F 00DD074FDE #
$600DD0746DA 00DE0E4A26 00DC154F34 00DB1C4FB9 !t00DE0E4789 00DC154C42 00DB1C469E 00DD074FDF #
$700DD0746D8 00DE0E4A25 00DC154F35 00DB1C4FBB !t00DE0E45FB 00DC154C46 00DB1C46A2 00DD074FDC #
$800DD0746D9 00DE0E4A28 00DC154F32 00DB1C4FBA !t00DE0E478B 00DC154C40 00DB1C469F 00DD074FE1 #
$900DD0746D8 00DE0E4A25 00DC154F30 00DB1C4FB9 !t00DE0E45FD 00DC154C44 00DB1C46A3 00DD074FE1 #
$000DD0746D9 00DE0E4A22 00DC154F31 00DB1C4FB9 !t00DE0E45FB 00DC154C45 00DB1C46A2 00DD074FDD #
```

Hx5parse.txt (Better view found on Example3_hx5parse.txt)

```
000.00 > 08536 |  0221 = 03.926  0222 = 03.944  0220 = 03.972  0219 = 03.975  | 08564 |  0222 = 03.930  0220 = 03.956  0219 = 03.933  0221 = 03.976
000.61 > 08536 |  0221 = 03.926  0222 = 03.944  0220 = 03.972  0219 = 03.975  | 08564 |  0222 = 03.930  0220 = 03.956  0219 = 03.925  0221 = 03.976
001.23 > 08536 |  0221 = 03.926  0222 = 03.944  0220 = 03.972  0219 = 03.975  | 08564 |  0222 = 03.930  0220 = 03.956  0219 = 03.925  0221 = 03.976
001.84 > 08536 |  0221 = 03.926  0222 = 03.944  0220 = 03.972  0219 = 03.975  | 08564 |  0222 = 03.930  0220 = 03.956  0219 = 03.925  0221 = 03.976
002.46 > 08536 |  0221 = 03.926  0222 = 03.944  0220 = 03.972  0219 = 03.975  | 08564 |  0222 = 03.930  0220 = 03.956  0219 = 03.925  0221 = 03.976
003.07 > 08536 |  0221 = 03.926  0222 = 03.944  0220 = 03.972  0219 = 03.975  | 08564 |  0222 = 03.930  0220 = 03.956  0219 = 03.925  0221 = 03.976
003.69 > 08536 |  0221 = 03.926  0222 = 03.944  0220 = 03.972  0219 = 03.975  | 08564 |  0222 = 03.930  0220 = 03.956  0219 = 03.925  0221 = 03.976
004.30 > 08536 |  0221 = 03.926  0222 = 03.944  0220 = 03.972  0219 = 03.975  | 08564 |  0222 = 03.921  0220 = 03.956  0219 = 03.925  0221 = 03.976
004.92 > 08536 |  0221 = 03.926  0222 = 03.944  0220 = 03.972  0219 = 03.975  | 08564 |  0222 = 03.930  0220 = 03.956  0219 = 03.925  0221 = 03.976
005.53 > 08536 |  0221 = 03.926  0222 = 03.944  0220 = 03.972  0219 = 03.975  | 08564 |  0222 = 03.921  0220 = 03.956  0219 = 03.925  0221 = 03.976
000.00 > 08536 |  0221 = 03.926  0222 = 03.944  0220 = 03.972  0219 = 03.975  | 08564 |  0222 = 03.921  0220 = 03.956  0219 = 03.925  0221 = 03.976
```

Example 4.

In the following example, two HX5MT are configured as tags (transmitters) and the remaining four are configured as monitors (receivers). Every receiving slave is polled by master call through a daisy chain setup. It is important to note, that the interval for a slave should always be longer than that of a master. If it is shorter, then the timers of the respective slave will time out before the master has called for the results. In order not to make the files look chaotic, the fonts are made small and possibly too small. Therefore text files containing the data should be found with the manual.

Hx5setup.txt

DeviceID	PGM CTRL	Termination	Interval
8522	3	8528	80
8528	0	8532	255
8532	0	8534	255
8534	0	8973	255
8536	4	333	5
8564	4	334	15

Hx5output.txt (Better view found on Example4_hx5output.txt)

```
$0014D094687 014E135105 !P014D0949BE 014E13470B !T014D095069 014E134D58 !V014D09526A 014E134929 #
$1014D094682 014E135100 !P014D0949BD 014E13470C !T014D095064 014E134D55 !V014D09526A 014E13492B #
$2014D094682 014E135100 !P014D094B4B 014E134708 !T014D095065 014E134D55 !V014D095268 014E134929 #
$3014D094686 014E135106 !P014D0949BD 014E13470D !T014D095065 014E134D56 !V014D0953F9 014E13492B #
$4014D094683 014E135101 !P014D094B4E 014E13470D !T014D095065 014E134D55 !V014D09526A 014E13492D #
$5014D094687 014E135103 !P014D094B4D 014E13489B !T014D095068 014E134D56 !V014D09526A 014E13492B #
$6014D094684 014E1350FF !P014D094B4D 014E13470B !T014D095066 014E134D53 !V014D095267 014E134929 #
$7014D094681 014E135102 !P014D094B48 014E13489C !T014D095064 014E134D57 !V014D095266 014E13492E #
$8014D094687 014E135104 !P014D0949BB 014E13470B !T014D09506B 014E134D59 !V014D095269 014E13492D #
$9014D094685 014E135103 !P014D094B4A 014E13470B !T014D095066 014E134D55 !V014D095265 014E134ABC #
```

Hx5parse.txt (Better view found on Example4_hx5parse.txt)

```
000.00 > 08522 | 0333 = 03.924 0334 = 03.982 | 08528 | 0333 = 03.942 0334 = 03.927 | 08532 | 0333 = 03.979 0334 = 03.962 | 08534 | 0333 = 03.990 0334 = 03.939
000.33 > 08522 | 0333 = 03.924 0334 = 03.982 | 08528 | 0333 = 03.942 0334 = 03.927 | 08532 | 0333 = 03.979 0334 = 03.962 | 08534 | 0333 = 03.990 0334 = 03.939
000.66 > 08522 | 0333 = 03.924 0334 = 03.982 | 08528 | 0333 = 03.951 0334 = 03.927 | 08532 | 0333 = 03.979 0334 = 03.962 | 08534 | 0333 = 03.990 0334 = 03.939
000.98 > 08522 | 0333 = 03.924 0334 = 03.982 | 08528 | 0333 = 03.942 0334 = 03.927 | 08532 | 0333 = 03.979 0334 = 03.962 | 08534 | 0333 = 03.998 0334 = 03.939
001.31 > 08522 | 0333 = 03.924 0334 = 03.982 | 08528 | 0333 = 03.951 0334 = 03.927 | 08532 | 0333 = 03.979 0334 = 03.962 | 08534 | 0333 = 03.990 0334 = 03.939
001.64 > 08522 | 0333 = 03.924 0334 = 03.982 | 08528 | 0333 = 03.951 0334 = 03.936 | 08532 | 0333 = 03.979 0334 = 03.962 | 08534 | 0333 = 03.990 0334 = 03.939
001.97 > 08522 | 0333 = 03.924 0334 = 03.982 | 08528 | 0333 = 03.951 0334 = 03.927 | 08532 | 0333 = 03.979 0334 = 03.962 | 08534 | 0333 = 03.990 0334 = 03.939
002.29 > 08522 | 0333 = 03.924 0334 = 03.982 | 08528 | 0333 = 03.950 0334 = 03.936 | 08532 | 0333 = 03.979 0334 = 03.962 | 08534 | 0333 = 03.990 0334 = 03.939
002.62 > 08522 | 0333 = 03.924 0334 = 03.982 | 08528 | 0333 = 03.942 0334 = 03.927 | 08532 | 0333 = 03.979 0334 = 03.962 | 08534 | 0333 = 03.990 0334 = 03.939
002.95 > 08522 | 0333 = 03.924 0334 = 03.982 | 08528 | 0333 = 03.950 0334 = 03.927 | 08532 | 0333 = 03.979 0334 = 03.962 | 08534 | 0333 = 03.990 0334 = 03.947
```

Controlling Tag Clusters

The HX5MT possesses the ability to work in a group. This opens some interesting possibilities. If the HX5MT units are wired together, one of them can be set up to control the other units. It is possible to set the Program Control Byte of a transmitting master unit to 05 and the rest of the devices to transmitting slave modes 04. In this case the Master unit will transmit it's ultrasonic positioning Id as it outputs $ (synchronization) over the wire network. Hence by setting the Interval Bytes of the transmitting slave units, the user can control the sequence. This ensures that no tags overrun.

Two ultrasonic positioning applications are now more achievable:

a. The user can string a network of transmitters overhead, and then run an autonomous apparatus with 3 receivers under the transmitters. Each autonomous apparatus can contain own laptop running the XYZ program real time and hence know own position.
b. If tags must constantly be close together to perhaps observe the orientation of an object, the synchronized transmission may be of advantage.

Following is a sample setup file for the clustered tag application:

Hx5setup.txt

DeviceID	PGM CTRL	Termination	Interval
8522	4	220	30
8528	4	221	24
8532	4	222	18
8534	4	223	12
8536	4	224	6
8564	5	225	36

The tags transmit Ids from 220 through 225, at the transmission of $ (sync) device 8564 transmits it's tag Id 225 approximately 6 x 4ms or 24mS later 8536 transmits it's tag Id 224. and this continues at 24mS intervals. The cycle is repeated every 36 x 4mS or 144mS.