P-ROV ACOUSTIC TIMING SYSTEM
#TIM Version 4.1
V-7-2013
A. Bradley

0. OVERVIEW

The #TIM SAIL device passes out DC isolated 1ms wide trigger pulses to whatever needs them. Currently, it has four DC isolated channels and can trigger four systems. The trigger pulses can be either polarity. The basic timing resolution is to a millisecond. The timing is slaved to an external trigger which must start the timer for each pulse group.

To function, the timer has to be given a basic re-arm period and a set of delays for the outputs. The re-arm period may be up to 9999 mS. In operation, after the re-arm period has timed out, another period will be started by the next positive edge of the trigger line. (Usually the trigger is a 1 Hz pulse. Triggers are ignored while the period clock is running.)

The four outputs repeat with each period with up to eight individually specified delays in each of the four channels after the start of the period. These delays are in the range of 1 to the period milliseconds-1. If a delay is set to longer than the current cycle period, that channel won't put out that pulse. The delays for a channel must be entered in ascending order.

Commands are included to start and stop the trigger pulses. When a Go command (!G) is given, the cycle starts on the next trigger pulse.

A command (?T) is included to report the current value of the timer.

1. INTERFACES

1.1 CONTROL

True RS-232 (+/- about 10V)
9600 baud, 8 bits, no parity bit, 1 stop bit
All characters sent to the device echo back
Address '#TIM'

1.2 TRIGGER

Positive going 5V pulse of any length > 2 microsec

1.3 OUTPUT PULSES

There are four output channels each with DC isolation and custom interface
hardware. See the circuit diagram for details. The output pulses are

always 1 ms wide and may be positive going or negative going.

## 2. COMMANDS

Any command may be entered directly after the address string or after a prompt. Any invalid command will put the interface in the unaddressed condition. A response of a prompt (which always ends with the [etx] (aka ^C)) verifies that the command was accepted.

In general, all numeric arguments accept hexadecimal, (0-9, A-F), values. The numeric entry routine takes only the last number of required input characters. Therefore 12345678 will be seen as 8 for a 1 digit argument, 78 for a 2 digit arg, 5678 for a 4 dig. arg. Numeric entries may be terminated by a [space] or by a [cr]. Thus commands that include numeric parameters all end with a [space] or [cr]. The numeric entry routine also assumes leading zeros, so !P12[cr] is the same as !P0012[[cr].

The #TIM device uses only decimal arguments. If you input a hex character, you will get unpredictable results.

Non-numeric commands (such as '?T') generally do _NOT_ need a terminating [cr] but immeadiately return an answer (if any) then the prompt. Adding a [cr] to these will cause a conflict, so you need to supress your instinctive 'add a cr' reaction. Deal with it!

### 2.1 [space] is a nul command that returns the prompt, [cr][lf]:[etx]

### 2.2 [cr] is another nul command that returns the prompt

### 2.3 H

Dumps a Help file. I stuff as much Help File in the EEPROM as there is room after the program. It just gives the program version and a terse version of the command syntax-

```
#TIMHelp TIM 4.1 Mon, May 6, 2013, 15:56
!Pdddd, dddd=ms, re-arm time
!Dn+ dd dddd[cr] ms dlys, (n=0-3, +/-, 8 dlys max)
?D see all dlys
!G, !S, Go, Stop
?T, see 1-shot ck
?N, see NEGATOR byte
?I, see ext trig count
!FH, !FL sets all GFCh Hi or Lo, !FX floats
?V see GF Chs
+ mon
        [the prompt follows]
```

2.4 !Pdddd[cr]
Sets the re-arm time. The argument dddd is milliseconds in the range 1 to 9999. Note this command needs a terminating [cr]. It will return a prompt. (Note that a [space] will also terminate the entry and return the prompt.)

2.5 ?P
View the current re-arm period in milliseconds.

'#TIM?P=dddd[prompt]'

If you change the period setting while it's running, it will go into the stop mode and you'll have to restart it with the !G (on the next second).

2.6 !G
This starts cycling and sending pulse outputs at the current period starting on the next second as defined by the trigger.

2.7 !S
This stops all pulse outputs immediately.

2.8 !Dns dddd dddd[cr]
This command sets the delays for channel n to a list of dddd ms entries. The value of n ranges from 0->3. The 's' is either + or - for a positive or negative going pulse. The delays are in decimal format. Each channel can have a maximum of 8 specified delay arguments which must be in ascending order. They are separated by a space character except the last which MUST have a terminating [cr].

Entering any !D command immediately stops trigger pulses if they are running. They must be restarted with !G (2.6 above.)

As soon as the '!Dn' is entered, all eight delay values for that channel are preset to FFFF. (These values are "safe" since they never equal the decimal formatted RTC and can never output a pulse.)

A delay value of 1 (or 0001) will work but 0 will not.

A typical setup might be-

```
        #TIM!P1998[cr]          ...trigger every 2 seconds
        #TIM!D0+ 1 100[cr]      ...for a 'double tap' at 1 and 100 ms
        #TIM!D1- 50[cr]         ...a single neg pulse in the middle of the
above
        #TIM!D2+ 1997           ...a 1 ms pulse just before the second mark
        #TIM!D3- 1 250 1750[cr] ... 3 neg going pulses at 1, 250, and 1750
```

ms

#TIM!G  ... start on next external trigger, NO [cr]!

You don't need to set all channels if you don't want them. You can set them in any order.

2.9 ?Dn

View the table of delays in 'memory dump' format. The results look like-

```
#TIM?D
0080 0001 0100 FFFF FFFF FFFF FFFF FFFF FFFF;
0090 0050 FFFF FFFF FFFF FFFF FFFF FFFF FFFF;
00A0 1997 FFFF FFFF FFFF FFFF FFFF FFFF FFFF;
00B0 0001 0250 1750 FFFF FFFF FFFF FFFF FFFF
```

The first column is the ram address of the next 8 hex data words. There are 4 lines of delays for CH0->CH3

2.10 ?N

This reports the "NEGATOR' byte which determines the pulse polarity for all four channels. If the bit is a zero, the pulses are positive going and 1 ms long. If a bit is a one, the pulse is negative going and also 1 ms long. The format is x.Ch0.Ch1.Ch2.Ch3.x.x.x. Thus '40' means only Ch0 is neg going.

2.11 ?T

This reports the current value of the ms count. The format is-

'#TIM?T dddd[prompt]'

2.13 ?I

This dumps the current trigger pulse count as a hex byte. It rolls over from FF to 00. It's useful to see if you've got trigger pulses coming in.

2.14 ?V

The timer module includes an 8 channel ground fault monitor. It is important to understand your system's grounding scheme to understand its use. (See section 3 below.)

3.0 SENTRY's GROUND FAULT DETECTION SYSTEM

Sentry uses a transformer SAIL interface which is DC isolated. The 1 Hz timing bus in Sentry is also DC isolated. This allows the #TIM system to be grounded to any subsystem we wish and makes the 8 channel ground fault

system useful.

Since RS232 has an obligatory ground and the P-ROV timing pulse is NOT isolated, you will have to decide if the included ground fault detection scheme is applicable.

### 3.1 Timing Module's Ground Tester

The module has 8 independent channels that are normally floating at high impedence between 0 and 5V with respect to the modules ground. They can be driven either high (to 5v) or low (to gnd) through 100k resistors (one for each of the eight channels) and are being continually scanned by the A-D converter. All 8 channels are scanned every 2 ms. The ?V command views these digitized voltages.

#### 3.1.1 ?V View 8 ground fault detect channels

The A-D inputs are protected by a 1M series resistor and 0.1 µFd capacitor to eliminate noise. A typical reply might look like-

```
?V 34 35 34 34 34 35 34 34
:[prompt]
```

#### 3.1.2 !FH Command, Set GF channels high

This command sets all GF channels to high (5v) through a 100k resistor. They will revert to floating (hi-Z) after 5 seconds. The user should allow at least 1 second after sending this command to query the voltages to allow their low pass filters to settle. The output should look like-

```
?V FF FF FF FF FF FF FF FF
:[prompt]
```

#### 3.1.3 !FL Command, Set GF channels low

This command sets all GF channels to low (0v) through a 100k resistor. They will revert to floating (hi-Z) after 5 seconds. The user should allow at least 1 second after sending this command to query the voltages to allow their low pass filters to settle. The output should look like-

```
?V 00 00 00 00 00 00 00 00
:[prompt]
```

#### 3.1.4 !FX Float Command

If you don't want to wait 5 seconds, this will float PORTC right away. Actually, any '!Fx' with x anything not H or L will do the same.

### 3.2 How to do Ground Fault Detection

Presumably each of the possble four sonar devices you are triggering are isolated from ground. If you add a resistive divider on the 5V supply of

the #TIM module to generate a 2.5V level with respect to #TIM's ground and hold the sonar system's "ground" at this 2.5V, a ?V request should return '80' (half scale) for that channel. If you get anything else, there's a leak somewhere!