Time-Base Control

PMAC's motion language expresses the position trajectories as functions of time. Whether the moves are specified directly by time or by speed, ultimately the trajectory is defined as a position-vs.-time function. This is fine for a great number of applications. However, in many applications, the PMAC axes must be slaved to an external axis not under PMAC control (or occasionally, an independent axis under PMAC control in a different coordinate system). In these applications, the PMAC trajectories are defined as functions of master position, not of time.

Real-Time Input Frequency

PMAC's method for doing this leaves the language expressing position as a function of time, but makes time proportional to the distance covered by the master. This is done by defining a real-time input frequency (RTIF) from the master's position sensor, in units of counts per millisecond. For example, define an RTIF of 32 cts/msec. Then, in time-base mode, when the program refers to a millisecond, what it is really referring to is 32 counts of the master encoder, whatever physical distance that is. If a program move in the slave program should take 2 seconds, it will really take 64,000 counts of the master encoder to complete.

Constraints on Selection of RTIF

If PMAC had infinite resolution and infinite dynamic range in its time base calculations, the choice of real-time input frequency would be entirely arbitrary – any frequency could be selected as the RTIF and the motion program could be written for that RTIF. However, PMAC does its time-base calculations in integer arithmetic, which limits the resolution and in 24-bit registers, which limits the dynamic range.

These limitations lead to three restraints on the selection of the RTIF:

1. The time base scale factor (TBSF) derived from the RTIF must be an integer. The value that PMAC needs for its calculations is not the frequency in cts/msec, but the inverse of the frequency in msec/ct. In order for this number to be in the range of integer values, the rule is to multiply the frequency inverse by 2^{17} (131,072).

   If a value of 100 cts/msec were chosen for RTIF, then the TBSF would be 131,072/100 = 131.072, which is not an integer. PMAC could only accept the integer part of 131, and drift would occur.

   A choice of real-time input frequency that is a power of 2 in cts/msec (e.g. 32, 64, 128) will always produce an integer TBSF. In addition, RTIF values that are equal to a power of 2 divided by an integer typically will work. For example, 204.8 cts/msec (=2048/10, = 2^{10}/10) will yield a TBSF of $2^{17}/2^{10}*10 = 640$.

2. The time base calculations will saturate at an input frequency (IF) where IF/RTIF equals the servo update frrequency in kHz. At the default servo update frequency of 2.25 kHz and an RTIF of 32 cts/msec, the maximum input frequency that can be accepted without saturation is 32*2.25 = 72 cts/msec. If the system could operate to 100 cts/msec, the choice of RTIF=32 cts/msec would not be acceptable, but a choice of RTIF=64 cts/msec would be acceptable (100/64=1.5625<2.25).

   A choice of RTIF greater than the maximum input frequency is always acceptable.

3. If PVT or SPLINE mode moves are used, the segment times at the RTIF must be an integer number of milliseconds. This means that the RTIF must be chosen so that the total cycle time at the RTIF is an integer number of milliseconds.

Sometimes this will not be possible unless the resolution of the master encoder is a power of 2. For this reason, it is suggested that the master encoder resolution be selected as a power of 2 (e.g. 1024 lines/rev instead of 1000 lines/rev).

For example, with a 1000 line/rev encoder (4000 cts/rev) on a spindle motor, an RTIF of 200 cts/msec corresponds to a speed of 50 revs/sec (3000 rpm), or exactly 20 msec/rev. However, it yields a TBSF of 655.36, which is not an integer.

With this encoder, an RTIF that yields an integer TBSF – 256 cts/msec yields 512, or 204.8 cts/msec yields 640 – corresponds to a cycle time that is not an integer.

However, with a 1024 line/rev encoder (4096 cts/rev), an RTIF of 204.8 cts/msec corresponds to a speed of 50 rps (3000 rpm) for a revolution time of 20 msec, and it yields a time base scale factor of exactly 640. In this case, the time base function stays locked, and a SPLINE or PVT sequence can be written that corresponds to an exact number of spindle revolutions.

How It Works

Time-base control works by lying to the commanded position update equations that occur every servo cycle about the amount of elapsed time since the last servo cycle. (Variable I10 contains the actual amount of time.) Note that the actual time between servo cycles does not change, nor do the dynamics of the servo loops. It is only the rate of the commanded trajectories that change with the external frequency and since all of the trajectories in the coordinate system change together, the path through space does not.

Make sure that it is counting up in the direction that master signal is going – counting down would imply a negative time-base, which PMAC cannot handle.

Instructions for Using an External Time-Base Signal

Using an external time-base signal requires several steps to set up. However, once the setup is complete, the time-base control is transparent to the user and the program – it is automatic. The steps in the set-up are detailed below.

Step 1: Signal Decoding

The signal is input to the PMAC at one of the incremental encoder inputs (Channels A and B). The signal must be either a quadrature signal (as out of an encoder) or a pulse and direction signal (pulse into A, direction into B). For the Encoder inputs used (one of the Encoders 1 to 16), Encoder I-variable 0 (I900 for Encoder 1, I905 for Encoder 2, etc.) controls the decode method, and defines what a count is. For instance, with a quadrature signal into Encoder 4 lines, I915 = 3 or 7 defines four counts per encoder cycle, whereas I915 = 2 or 6 defines only two counts per encoder cycle. The difference between 3 and 7, or 2 and 6 is for which sense of the signal does the decoder count up.

Analog Source for Frequency

PMAC has a single on-board voltage-to-frequency (V-to-F) converter that allows a voltage level input to the Wiper line of the JPAN connector (Pin 20 of J2) to control the time base. The input is 0 to +10V analog signal that is converted to a nominal 0 to 250 KHz frequency (25KHz/V). Jumpers E72 and E73 ON connect this signal to the Encoder 4 decoder-counter (there is no choice about which encoder). Make sure jumper E24 connects pins 1 and 2 (the default). I-variable I915, which controls the decoding of this signal, should be set to 4 (pulse and direction, counting up on this signal).

From this point on, the time-base control can be treated just as if it came from an external frequency source. Note that the default conversion table is set up to handle time-base information from this encoder counter. Refer to the diagram under Control-Panel I/O in the Connecting PMAC to the Machine section.

Step 2: Interpolation

Once decoded and counted, the value from the signal is brought into the encoder conversion table once per servo cycle, exactly as a position feedback signal would be. Using the 1/T conversion method here is recommended because this method gives a good sub-count interpolation of the signal (using timers associated with the counter) that significantly enhances the smoothness of the time base information. Make sure that the conversion table is set up to process the counter from the input signal this way. The encoder conversion table is set up at the factory to process 1/T conversion on encoder counters 1 through 8. See the description of the encoder conversion table for more details.

Step 3: Time Base Calculation

A separate entry in the encoder conversion table takes the interpolated position information from the above step, subtracts out the interpolated position information from the previous servo cycle, and multiplies this difference by a scale factor to produce the time base value for the servo cycle. (This time base value is then a multiplying factor in the position update calculations, so the amount of update is proportional to the number of counts received from the time base signal in the last servo cycle.)

The two set-up items in this step are the source of information (the interpolated position register) and the scale factor. Both of these are entries in the encoder conversion table. See the description of the table for more details on how to enter these.

The equation for the time base conversion is:

$$\% \text{ value} = (100.0 * \text{SCALE\_FACTOR} * \text{INPUT\_FREQ}) / 2^{17}$$

where the % value (also known as feedrate override value) is what controls the rate of position update – when it equals 100.0, programs and moves operate in real time (i.e. at the times and speeds specified in the program). SCALE_FACTOR is the integer value that must be determined to set up time base following properly. INPUT_FREQ is the count rate (as determined by the signal and Encoder I-variable 0) in counts/millisecond. 2^{17} is 131,072.

To set the scale factor, decide on a real-time input count frequency – the rate of input counts at which the program and moves should execute at the specified rate. Since this is the rate at which the % value will be 100.0, it can be solved simply for the scale factor:

$$\text{SCALE\_FACTOR} = 131{,}072 / (\text{REAL\_TIME\_INPUT\_FREQ})$$

Since the scale factor must be an integer, and 131,072 is a power of 2, make the real time input frequency a power of 2 in units of counts/msec. For instance, in a system where the typical full-speed input count frequency is 60,000 counts/second, define the real-time input frequency to be 64 counts/msec. This would then make the scale factor 131,072 / 64 = 2,048.

Step 4: Using the Time-Base Calculation

Time-base values work on a coordinate system. Each coordinate system has an I-variable that tells it where to look for its time base information. This variable is Ix93 for Coordinate System x. The default values for Ix93 are the addresses of registers that are under software control, not the control of an external frequency. For a coordinate system that should be under external time-base control, put the address of the scaled time-base value determined above. For instance, in the default conversion table, this value is at address $729 (1833 decimal), so if Coordinate System 1 were to be controlled by this frequency, I193 would be set to 1833 (this is always an Xmemory word, so X' does not need to be specified).

Once this I-variable has been set up, all motors assigned to this coordinate system will be under the control of the external frequency, in programmed and non-programmed moves.

I-variable Ix94 controls the maximum rate of change of the time-base value for Coordinate System x. When commanding the time-base value from the host (with a **%n** command), this value should be set fairly low to produce a nice slewing to the new commanded value. However, to keep synchronized to an external signal as time-base source, this value should be set as high as possible (maximum value is 8,388,607) so the time base can always slew as fast as the signal. Setting the value low can improve following smoothness at the cost of some slip in the following. If the Ix94 limit is ever used in external time base, position synchronization to the master is lost.

Step 5: Writing the Program

When the program is written that is to be under external time-base control, simply write it as if the input signal were always at the real-time frequency. When run, the program will execute at a rate proportional to the input frequency. There will be full floating-point resolution on the move times and feedrates are specified. Remember that **DWELL** commands always execute in real time, regardless of the input frequency. To place pauses in the program that are proportional to an input frequency, use the **DELAY** command, not **DWELL**.

Time-Base Example

A web of material is moving at a nominal speed of 50 inches per second and an encoder on the web that gives 500 lines per inch. There is a crosscutting axis under PMAC control. When the web is moving at nominal speed, make a cutting move in 0.75 seconds and be ready to start another move 2.50 seconds later. The web encoder is attached to Encoder 2 input lines.

Step 1: Signal Decoding

Since the web encoder is Encoder 2, I905 controls the decoding. For maximum resolution, set I905 to 3 or 7 for 4x decode. Try 3 first. Looking in the list of suggested M-variables in the manual, it shows that the encoder position M-variable for this encoder is M201. Make the definition for M201 and query its value repeatedly (probably using the Executive Program Watch window) while turning the web encoder in the direction it will be going in the application. If the value increases as the encoder is turned, I905 is set properly. If it decreases, change I905 to 7. (If it does not change, check the connections.)

Step 2: Interpolation

Next, look at the current set-up of the encoder conversion table. The easiest way to do this is through the Configuration menu of the PMAC Executive program. If this is not available, command PMAC with **RHY:$720,16**, which causes PMAC to report the contents of addresses Y:$720 to Y:$72F – the set-up data for the table. The following response is received:

```
00C000 00C004    00C008 00C00C 00C010    00C014  00C018
00C01C 400723    000295 000000 000000    000000  000000
000000 000000
```

(The values shown here are the default values for the table.) The second value returned (from address Y:$721) shows a 1/T conversion of Encoder 2, which occupies registers $C004 to $C007 (49156 to 49159). This gives the desired sub-count data for smoothness. It is not necessary to change anything here. However, if the entry read *C0C004*, change it by commanding `WY:$721,$00C004`.

Step 3: Time-Base Calculation

Now set up an entry in the table to convert the interpolated position to time base format. Looking at the values reported above, it shows that the ninth entry (from address Y:$728), *400723*, is a time-base conversion.

However, its source is address $723, which is the interpolated position from Encoder 4, not Encoder 2 as needed. To change it, command `WY:$728,$400721`.

Now compute the scaling factor. Look at the nominal speed of 50 inches/sec, the resolution of 500 cycles/inch, and the 4x decode, and calculate:

50 inches/sec * 500 cycles/inch * 4 counts/cycle
= 100,000 counts/sec
= 100 counts/msec

Since the math works out more easily if this number is a power of two, declare the real-time count rate to be 128 counts/msec. Then calculate the scale factor as 131,072 / 128 = 1024. Enter the scale factor by commanding `WY:$729,1024` (note that the value can be entered as a decimal number by omitting the dollar sign).

Step 4: Using the Time-Base Calculation

Since working in Coordinate System 1, assign I193 to $729 (1833 decimal) to point to this time base value. Set I194 to the maximum value of 8,388,607 so synchronicity is not lost on rapid changes.

Step 5: Writing the Program

In writing the program, work at the real-time input frequency, which differs from the starting nominal speed – in this case, it is exactly 28% faster. Therefore, any programmed speeds would be 28% higher; any programmed times would be 28% less. Take the nominal cut time of 750 msec (0.75 sec) and multiply it by 100/128 to get exactly 585.9375 msec. The 2500 msec return is scaled similarly to 1953.125 msec. (If these numbers do not come out exactly in the program, put the math directly in the program; PMAC calculates with 48-bit floating-point precision.) A main program loop would look something like this:

```
WHILE (M11=1)                ; Cut as long as input is true
      TM 585.9375            ; Cut move time
      X10000                 ; Actual cut move
      DELAY 500              ; Hold; part of 1953.125 msec return
      TM 953.125             ; Return time; part of 1953.125 msec
      X0                     ; Actual return move
      DELAY 500              ; Hold; part of 1953.125 msec return
ENDWHILE
```

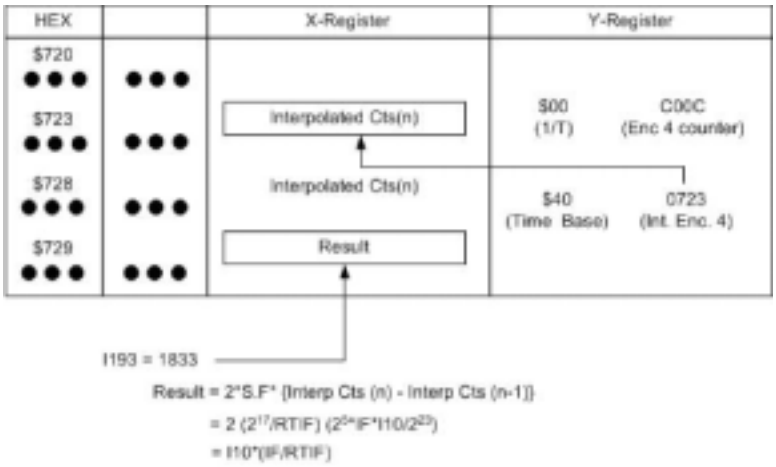
Time-Base Conversion Entries

A time-base conversion is a scaled digital differentiation. When the source data is a counter, the result is a frequency value. Every servo cycle, the table calculates the difference between the value of the source register for this cycle and the value for the last cycle, and multiplies the difference by the scale factor. The most common use for the resulting value is for time-base (feedrate override) control, which makes the speed of PMAC execution proportional to an external frequency (usually the speed of a master device). For a time-base conversion, there are two entries in each column, and the format is:

X-Words	Y-Words
1. Last cycle's source data:	1. Source and process
Bits 0-4: Fractional Bits Bits 5-23: Integer Bits	Bits 0-15: X Address of source data (usually a converted position register). Bits 16-23: = $40 for time base
2. Actual time-base value: product of scale factor and difference between last two source values	2. Time-base scale factor (supplied by user)

For example, the default conversion table creates a time-base value from the data in the Encoder 4 counter. In this time-base conversion the source data should have sub-count interpolation – this significantly smoothes out the process by reducing the quantization error created by digital differentiation. To do this, the source register should be from the conversion table itself, not from the encoder counter. In the default conversion table, the converted data from Encoder 4 is found in X:$0723 (1827 decimal). Therefore, the first setup (Y) word for the time-base conversion entry is $400723 – the $40 specifies time-base conversion and the $0723 specifies the source address.

Conversion Table Example for Time-Base Entry



Scale Factor

The second setup (Y) word is the scaling factor – the value that multiplies the difference between the current source data and the last source data. Usually, setting its value requires some computation; this subject is covered in the Time-Base Control portion of the Coordination Features section.

Converted Data

The last source data word is stored in the first X word of the entry in the table and the net result is stored in the second X word. The value of the net result is 2 * Scale_factor * (New_source - Old_source). To use this value to control the time-base of a coordinate system, enter this address as the value of Ix93 (Time-Base Source Address) for the coordinate system.

Triggered Time-Base Conversion Entries

On applications where it is necessary to synchronize exactly to the position of the master encoder, the conversion table provides the capability to freeze the time-base while calculating the first moves of the synchronized sequence. Then the time base start up is referenced exactly to the master position that occurred when the starting trigger occurred (usually the index channel of the master encoder). This provides a complete position lock of the slave to the master; there is no need for subsequent adjustment to make sure that they are phased in, as would be the case for normal (untriggered) time-base control.

Entry Format

For a triggered time-base conversion, there are two entries in each column for the following format:

X-Words	Y-Words
1. Last cycle's source data: Bits 0-4: Fractional bits Bits 5-23: Integer Bits	1. Source and process: Bits 0-15: Address of source (always a set of DSPGATE encoder registers) Bits 16-23: $90 (frozen; for preparation) $B0 (armed; waiting for trigger) $A0 (running; post-trigger)
2. Actual time-base value: when running, the product of scale factor and difference between last two source values	2. Time-base scale factor: supplied by user; equal to 131,072 / real-time-input-frequency in cts/msec

Unlike the normal (untriggered) time-base conversion, the source address must be that of the encoder registers in the DSP-GATE with the raw (unprocessed) data. The triggered time-base conversion does the 1/T interpolation itself. The valid addresses for triggered time-base entries are the same as those for the Incremental Encoder Entries: $C000 for Encoder 1, $C004 for Encoder 2, and so on, to $C03C for Encoder 16.

Setting the Trigger State

The process bits – bits 16 to 23 of the first Y-word in the conversion table entry – of a single triggered time-base entry will take on three values during the normal course of use. Usually, this is done with an 8-bit M-variable. First, with the slave axes dwelling at their starting position, these process bits should be set to $90 in the sequence of motion program calculations for the first move. This forces the time-base value to zero, putting the coordinate system in feed-hold mode.

Next, another program, usually a PLC program, changes the bits from $90 to $B0. This arms the time-base, so that it is waiting for the position-capture trigger on the source encoder, as defined by Encoder/Flag I-variable 2 for that encoder. When the capture occurs, the time base starts up with the captured-position register is used as the initial value for the time-base difference equations. When this happens, PMAC changes the process bits from $B0 to $A0 automatically. For untriggered use of this format, the user can simply set the process bits to $A0 himself.

Example:
For example, if adding to the end of the standard conversion table a triggered time-base entry working from Encoder 8 with a real-time input frequency of 64 cts/msec, these entries would reside in registers $072A(1834) and $072B (1835). Initially, a value of $A0C01C (running time-base from Encoder 8 registers) would be written to Y:$072A and a value of $800 (131,072 / 64 = 2048 = $800) to Y:$072B. An M-variable is defined to the process bits with the command **`M199->Y:$072A,16,8`**.

With the slave axes dwelling at the start-up position, freeze the time base with the motion program command **`M199=$90`**. If Ix93 is not already pointing to X register $072B, do this at this time. The motion program commands immediately following this calculate the move, but with a zero time-base value, the move execution is stuck at the starting point. Meanwhile, a PLC program is looking for M199 to be equal to $90, at which time it changes it to $B0, arming for the trigger. Since a PLC program cannot interrupt motion program calculations for a move, it is assured that this will not be done until after the calculations are completed. This change can be done with three program lines in a PLC program:

```
IF (M199=$90)
M199=$B0
ENDIF
```

Once the trigger is armed by the PLC program, when the capture trigger occurs, PMAC starts the time base and changes the process bits to $A0 automatically.

I10 Servo Interrupt Time

The I10 parameter tells the PMAC how much time there is between servo interrupts (which is controlled by hardware), so that the software knows how much time to increment each servo interrupt. Although this parameter can be changed at any time and immediately effects how the present feedrate override value (% command) is reported, the new value will take effect only at the next power-up, reset, or **`%{constant}`** command. The fundamental equation for I10 is:

$$I10 = \frac{8{,}388{,}608}{ServoFrequency(\,kHz\,)} = 8{,}388{,}608 * ServoTime(\,m\,sec)$$

I10 is used to provide the delta-time value in the position update calculations, scaled such that 2^{23} – 8,388,608 – means one millisecond. Delta-time in these equations is I10 * (%value/100). The % (feedrate override) value can be controlled in any of several ways: with the **%** on-line command, with a direct write to the % command register, with an analog voltage input, or with a digital input frequency. The default % value is 100, and many applications can always leave it at 100.

%{constant}

This command specifies the feedrate override value for the currently addressed coordinate system. The rate of change to this newly specified value is determined by coordinate system I-variable Ix94. **%{constant}** sets the addressed coordinate system's feedrate override value where {constant} is a non-negative floating point value specifying the desired feedrate override (time-base) value (100 represents real-time) I-variable Ix93 for this coordinate system must be set to its default value (which tells to coordinate system to take its time-base value from the %-command register) in order for this command to have any effect.

The maximum % value that PMAC can implement is equal to (2^{23}/I10)*100 or the (servo update rate in kHz)*100. If a value greater than this is specified, PMAC will saturate at this value instead. To control the time base based on a variable value, assign an M-variable (M197 is suggested) to the commanded time base register (X:$0806, X:$08C6, etc.), then assign a variable value to the M-variable. The value assigned here should be equal to the desired % value times (I10/100).

```
&0                          ; Command value of 0, stopping motion
%33.333                     ; Command 1/3 of real-time speed
%100                        ; Command real-time speed
%500                        ; Command too high a value
%                           ; Request current value
225.88230574                ; PMAC responds ; this is max allowed value
M197->X:$0806,24            ; Assign variable to C.S. 1 % command reg.
M197=P1*I10/100             ; Equivalent to &1%(P1)
```

Ix93 Coordinate System x Time Base Control Register Address

Variable	Hex	Decimal	Register
I193	$0806	2054	C.S.1 '%' cmd reg
I293	$08C6	2246	C.S.2 '%' cmd reg
I393	$0986	2438	C.S.3 '%' cmd reg
I493	$0A46	2630	C.S.4 '%' cmd reg
I593	$0B06	2822	C.S.5 '%' cmd reg
I693	$0BC6	3014	C.S.6 '%' cmd reg
I793	$0C86	3206	C.S.7 '%' cmd reg
I893	$0D46	3398	C.S.8 '%' cmd reg

This parameter tells coordinate system x where to look for its time base control (feedrate override) information by specifying the address of the register that will be used. The default value of this parameter for each coordinate system (see above) specifies the register that responds to on-line **%** commands. If the time base is left alone, or is under host or programmatic control, this parameter should be left at the default. Alternatively, if the time base is controlled externally from a frequency or voltage, usually the register containing the time-base information will be in the conversion table (which starts at address $720 [1824 decimal]). With the default conversion table, there is a time-base register at $0729 (1833) related to the frequency into the Encoder 4 counter. This frequency can be controlled by an input voltage on the WIPER pin of the Control Panel Port if jumpers E72 and E73 are ON. If another register is to be used for the time base, it must have the units of I10 so that 8388608 (2^{23}) indicates 1 msec between servo interrupts.