UNITED STATES
NUCLEAR REGULATORY COMMISSION
WASHINGTON, D. C. 20555

APR 1 2 1982

MEMORANDUM FOR: Walter R. Butler, Chief, Containment Systems Branch, DSI

FROM: Peter S. Kapo, Containment Systems Branch, DSI

SUBJECT: NUREG-0737, ITEM II.F.1.4, CONTAINMENT PRESSURE MONITOR SYSTEM (CPMS): METHOD FOR ESTIMATING THE COMBINED TIME CONSTANT OF A STRING OF COMPONENTS EACH OF WHICH HAS A KNOWN TIME CONSTANT

In discussing the time response of the CPMS with licensees, it became evident that almost all licensees had specifications for the time constants of the individual components in their CPMS, but none had a method for computing the time constant of the overall system.

It was clear that the review of the CPMS would be expedited if we (the NRC) prescribed a standard method for combining the time constants of the individual components that make up the CPMS. In our initial discussions, all the licensees were amenable to using the NRC prescription.

We used our time constant prescription for all CPMS reviews completed in the second fiscal quarter, and do not anticipate any difficulty in using it for CPMS reviews to be completed in the third and fourth quarters.

The mathematical development of our time constant prescription and an HP-67 program which computes this prescription is given in the attachment to this memorandum.

The prescription assumes that all components have first order transfer functions. However, it is shown in the attachment that if the components have higher order transfer functions, then the prescription which assumes all first order transfer functions gives a conservative (higher than) estimate of all overall system time constant.

Our prescription is valid only for components which display a linear response to the driving function, which means that it is applicable to all components in the CPMS except the strip chart recorder. This presents no serious problem because, depending on the magnitude of the pressure transient, the time response of the CPMS is dominated either by the strip chart recorder or by the rest of (the linear part of) the system. The time response of the system can be considered to be identical to the time response of that part of the system which dominates the time response.

8205070408 XA

While our prescription was developed specifically to use in computing the overall time constant of the CPMS, it can obviously be used to compute the overall time constant of any string of components with linear transfer functions. The exact application to higher order systems is discussed in the attachment.

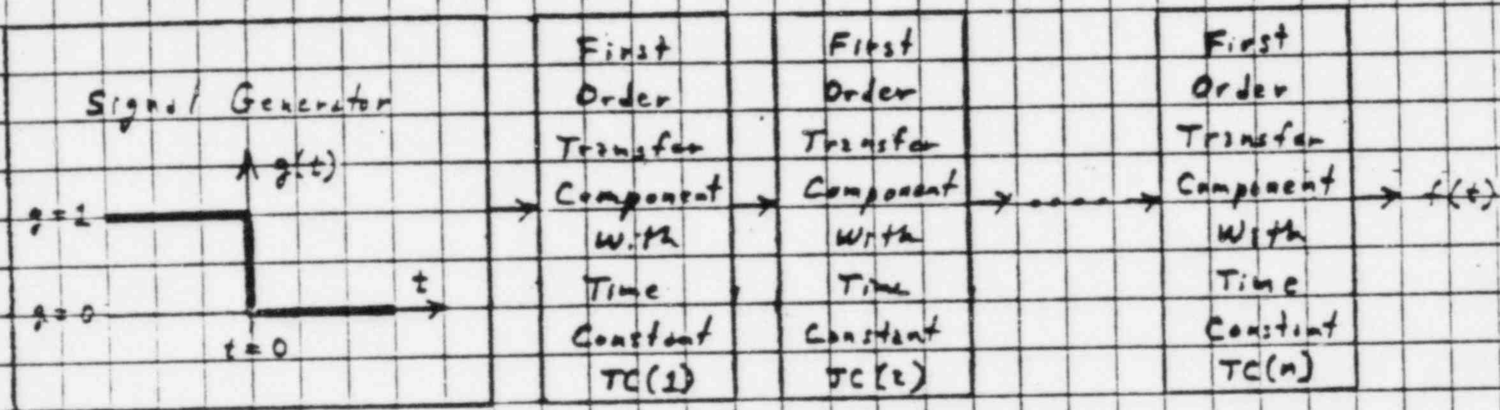
Peter S Kapo

Peter S. Kapo
Division of Systems Integration
Office of Nucelar Reactor Regulations

Attachment: As Indicated

cc: R. Mattson
T. Speis
H. Nicholaras
W. Wade, EG&G
J. Kudrick
M. Fields

HP-67 PROGRAM TO FIND THE COMPOSITE TIME CONSTANT, TC of $f(t)$, [ie TC such that $f(TC) = e^{-1}$], where $f(t)$ is GIVEN BY:



Program Input Consists of

RA = n
R0 = TC(1)
R1 = TC(2)
R2 = TC(3)
R3 = TC(4)
R4 = TC(5)

n is the number of Transfer Components

Only TC(1) thru TC(n) need be input.

When Program is Finished Computing it Stops with TC in RX [ie TC in Display]

Program Interpolates on TC and Stops when

$ABS\left[f(TC) - e^{-1}\right] < 10^{-6}$

$$\left[\begin{array}{l}\text{Laplace Transform of Unit}\\\text{Step Function at } t=0\end{array}\right] = \left[\frac{1}{s}\right]$$

$$\left[\begin{array}{l}\text{Laplace Transform of the}\\\text{Transfer Function of a First}\\\text{Order Component with Time}\\\text{Constant} = TC(i)\end{array}\right] = \left[\frac{1}{1 + s * TC(i)}\right]$$

$$[\text{Laplace Transform}] = \mathcal{L}$$

$$[\text{Inverse Laplace Transform}] = \mathcal{L}^{-1}$$

$$f(t) = 1.0 - \mathcal{L}^{-1}\left[\frac{1}{s}\right]\left[\frac{1}{1 + s * TC(1)}\right]\left[\frac{1}{1 + s * TC(2)}\right]\cdots\left[\frac{1}{1 + s * TC(n)}\right]$$

It is easier to see the next steps if we assume a specific value for n, and then generalize the procedure when we write the program. Use n = 3

$$\left[\frac{1}{s}\right]\left[\frac{1}{1 + s * TC(1)}\right]\left[\frac{1}{1 + s * TC(2)}\right]\left[\frac{1}{1 + s * TC(3)}\right]$$

$$= \left[\frac{1}{s}\right] - \left[\frac{1}{1/TC(1) + s}\right]\left[\frac{TC(1)}{TC(1) - TC(2)}\right]\left[\frac{TC(1)}{TC(1) - TC(3)}\right]$$

$$- \left[\frac{1}{1/TC(2) + s}\right]\left[\frac{TC(2)}{TC(2) - TC(1)}\right]\left[\frac{TC(2)}{TC(2) - TC(3)}\right] -$$

$$- \left[\frac{1}{1/TC(3) + s}\right]\left[\frac{TC(3)}{TC(3) - TC(1)}\right]\left[\frac{TC(3)}{TC(3) - TC(2)}\right]$$

$$= \left[\frac{1}{s}\right] - A(1)\left[\frac{1}{1/TC(1) + s}\right] - A(2)\left[\frac{1}{1/TC(2) + s}\right] - A(3)\left[\frac{1}{1/TC(3) + s}\right]$$

$$\text{where } A(1) = \left[\frac{TC(1)}{TC(1) - TC(2)}\right]\left[\frac{TC(1)}{TC(1) - TC(3)}\right] \quad \text{etc.}$$

$$f(t) = 1.0 - 1.0 + A(1) * \exp[-t/TC(1)] + A(2) * \exp[-t/TC(2)] + A(3) * \exp[-t/TC(3)]$$

$$= A(1) * \exp\left[\frac{-t}{TC(1)}\right] + A(2) * \exp\left[\frac{-t}{TC(2)}\right] + A3 * \exp\left[\frac{-t}{TC(3)}\right]$$

MEMORY REGISTERS

For convenience the TC(i) are input in the primary memory registers. However throughout the whole execution of the program the TC(i) and A(i) are stored in the secondary registers as follows:

R10 = TC(1)   R11 = TC(2)   R12 = TC(3)   R(13) = TC(4)   R14 = TC(5)
R15 = A(1)   R16 = A(2)   R17 = A(3)   R18 = A(4)   R19 = A5

PRIMARY REGISTER: USED IN COMPUTING A(i) s.

R0 = Index of Register Containing TC(i)
R1 = Index of Register Containing TC(j)
R2 = Index of Register Containing A(i) = R0 + 5
R3 = RA + 10 = One more than last R0 or R1

PRIMARY REGISTERS USED IN INTERPOLATING TO FIND TC.
[Subroutine a computes f(t), and this is the only place in the interpolation scheme where the indices of the registers containing TC(i) and A(i) are needed.]

Subroutine a Memory Registers

R0 = Index of Register Containing TC(i)
R1 = Index of Register Containing A(i)
R2 = Highest Value of R0 to use.
R8 = t (Subroutine a Input)
[illegible] = f(t) (Subroutine a Output)

Memory Registers Used in Interpolation Scheme

R9 = Upper Bound for TC (= TCU)
R8 = Interpolation Value for TC (= TCI)
R7 = Lower Bound for TC (= TCL)
R6 = f(TCU)
R5 = f(TCI)
R4 = f(TCL)
R3 = $f(TCI) - e^{-1}$

We found that good starting values for the interpolation are $TCL = \sum TC(i)$ $TCU = 1.2 * TCL$

These values do not always bracket TC, but the interpolation scheme still works well.

Note: Although the mathematical problem is solvable if two or more of the TC(i) are identical or if one or more of the TC(i) is negative, the HP-67 program will bomb out if it is run with the input meeting either of these two conditions.

PROGRAM BLOCK DIAGRAM

Input

$RA = n \leq 5$
$R0 = TC1$
$R1 = TC2$
....
$R(n-1) = TC(n)$

↓

Compute $A(i)$ such That

$$f(t) = \sum A(i) * \exp\left[\frac{-t}{TC(i)}\right]$$

↓

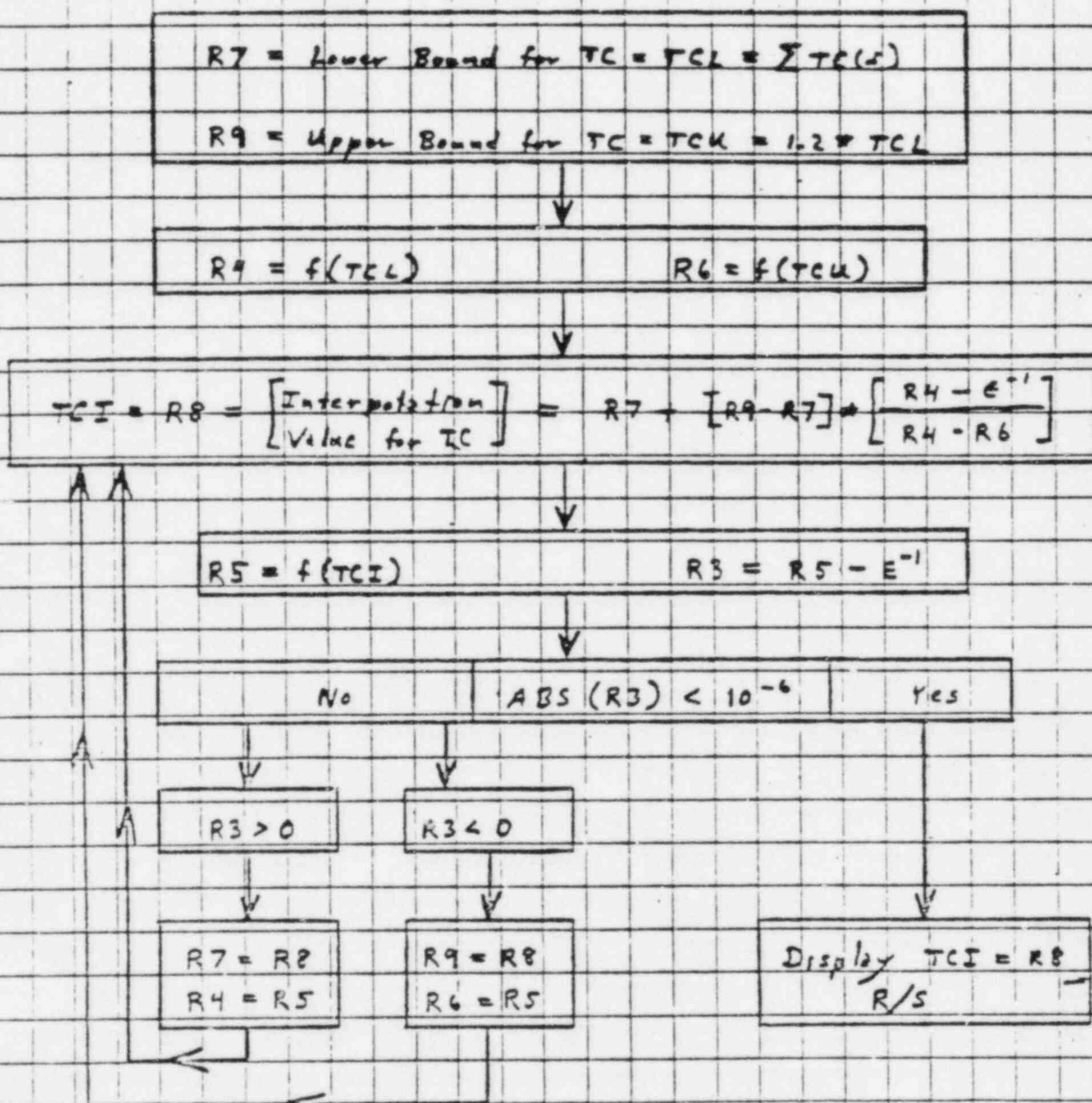
Interpolate to find $t = TC$
such that $ABS\left[f(TC) - e^{-1}\right] < 10^{-6}$

In the Interpolatron Scheme
$f(t)$ is computed using
Subroutine a.

↓

Stop When Interpolation is
Finished with TC in the
Display (ie in RX)

BLOCK DIAGRAM OF INTERPOLATION SCHEME



HP-67 PROGRAM START

Compute A(i)s [Start]

f LBL A	1	
	STO 5	Initialize
	STO 6	
	STO 7	A(1) = A(2) = A(3) = A(4) = A(5) = 1
	STO 8	
	STO 9	Put TC(i)s and A(i)s in
	P↔S	Secondary Memory Registers
	9	Initialize Indices to One Less
	STO 0	than Starting Point.
	14	
	STO 2	R0 = 9 = Location of TC(i)
	10	R2 = 14 = Location of A(i)
	RCL A	R3 = RA + 10 = One more than last
	+	R0 or R1
	STO 3	
f LBL 0	1	
	STO + 0	R0 = R0 + 1 R2 = R2 + 1
	STO + 2	
	RCL 0	End Looping on R0
	RCL 3	if R0 = R3 ie Computation
	x = y	of A(i)s is Finished.
	GTO 2	
	9	Initialize
	STO 1	R1 = 9 = Location of TC(j)
f LBL 1	1	
	STO + 1	R1 = R1 + 1
	RCL 1	
	RCL 3	End Looping on R1
	x = y	if R1 = R3
	GTO 0	

Compute A(i)s [Continued]

TC 8

RCL 0 RCL 1 x = y GTO 1	Skip A(i) computation if $i = j$ (R0 = R1)
RCL 0 STI RCL (I) ENTER ENTER ENTER RCL 1 STI CLX RCL (I)	Fetch Numbers for A(i) Computation RZ = TC(i) RY = TC(i) RX = TC(j)
− ÷	Compute Factor for A(i) $\left[\dfrac{TC(i)}{TC(i) - TC(j)}\right]$
RCL 2 STI R↓ STO × (I) GTO 1	Multiply A(i) = R(R2) by Factor Go back and continue Looping on j = R1

Compute Starting Values for Upper and Lower Bounds on TC.

TC9

R7 = Lower Bound for TC = TCL = Σ TC(i)
R9 = Upper Bound for TC = TCU = 1.2 * TCL

```
f LBL 2     9          Initialize
            STI
            RCL A      I = 9 = Location of TC(i)
            +
            STO 1      R1 = 9+n = 9+RA = Highest I
            0
            STO 7      R7 = 0 = Lower Bound for TC

f LBL 3     ISZ        Increment  I = I+1
            RCL (i)
            STO + 7    Add TC(i) = R(I) to TCL = R7

            RCI
            RCL 1      Loop Back if R1 > I
            x > y
            GTO 3

            RCL 7
            1.2        Compute TCU = 1.2 * TCL
            x          ie R9 = 1.2 * R7
            STO 9

            RCL 7
            STO 8      Compute R4 = f(TCL) = f(R7)
            g GSB a
            STO 4

            RCL 9
            STO 8      Compute R6 = f(TCU) = f(R9)
            g GSB a
            STO 6
```

Iterate to Find TC [cont]

```
f LBL 4     RCL 4
            -1
            e^x
            -
            RCL 4
            RCL 6
            -
            ÷
            RCL 9
            RCL 7
            -
            ×
            RCL 7
            +
            STO 8
```

Interpolate Between R7 = TCL and R9 = TCU to find R8 = TCI, our new best guess of TC.

$$R8 = TCI = R7 + [R9 - R7] * \left[\frac{R4 - e^{-1}}{R4 - R6}\right]$$

```
            GSB f a
```

$R5 = f(TCI) = f(R8)$

```
            RCL 5
            -1
            e^x
            -
            STO 3
```

$R3 = f(TCI) - e^{-1}$
$= f(R8) - e^{-1}$

TC 12

Iterate to Find TC [Continued]

	-6	Program is Finished if
	10^x	$ABS[f(TCI) - e^{-1}] = ABS[R3] < 10^{-6}$.
	RCL 3	Display TCI = R8 and R/S
	ABS	
	$x > y$	
	GTO 5	Otherwise Continue Iterating
	RCL 8	
	R/S	
f LBL 5	RCL 3	
	$x < 0$	For $[f(TCI) - e^{-1}] = R3 > 0$
	GTO 6	
	RCL 8	Set TCL = TCI f(TCL) = f(TCI)
	STO 7	
	RCL 5	ie R7 = R8 R4 = R5
	STO 4	
	GTO 4	Go Back to Recompute R8 = TCI
f LBL 6	RCL 8	For $[f(TCI) - e^{-1}] = R3 < 0$
	STO 9	
	RCL 5	Set TCL = TCI f(TCL) = f(TCI)
	STO 6	ie R9 = R8 R6 = R5
	GTO 4	Go Back to Recompute R8 = TCI

TC-12

Subroutine a — Computes: f(t)

Subroutine Inputs: t = R8 n = RA
TC(i) and A(i) in secondary memory registers

Subroutine Output: f(t) = R5 = RX

g LBL a	0	Initialize
	STO 5	R5 = 0 = f(t)
	9	R0 = 9 = One less than location of TC(1)
	STO 0	
	RCL A	
	+	R1 = 14 = One less than location of A(1)
	STO 2	
	14	R2 = 9 + RA = last R0 to use
	STO 1	
g LBL b	1	R0 = R0 + 1 R1 = R1 + 1
	STO + 0	
	STO + 1	
	RCL 8	Add A(i) * exp[-t/TC(i)] to R5
	RCL 0	
	ST I	
	CLX	
	RCL (I)	
	÷	
	CHS	
	e^x	
	RCL 1	
	ST I	
	CLX	
	RCL (I)	
	×	
	STO + 5	

RCL 0	Switching
RCL 2	
x ≠ y	
GTO f b	
RCL 5	
RTN	

HP-67 Program Exp

TEST PROBLEM:

The HP-67 Program is written so that the number of TC(i)s is variable. This required memory register indexing that was a little complicated. It seems worthwhile to run off a few problems by hand to check that we got all our memory register indexing right.

Test Computation of A(i)s.

RA = 1 TC(1) = 2 A(1) = 1

RA = 2 TC(1) = 2 TC(2) = 5

$A(1) = [2/(2-5)] = -2/3 = -0.6667$

$A(2) = [5/(5-2)] = 5/3 = 1.6667$

RA = 3 TC(1) = 2 TC(2) = 5 TC(3) = 6

$A(1) = [2/(2-5)][2/(2-6)] = +[2/3][2/4] = +1/3 = +0.3333$

$A(2) = [5/(5-2)][5/(5-6)] = -[5/3][5/1] = -25/3 = -8.3333$

$A(3) = [6/(6-2)][6/(6-5)] = +[6/4][6/1] = +36/4 = +9.0000$

RA = 4 TC(1) = 2 TC(2) = 5 TC(3) = 6 TC(4) = 8

$$A(1) = \left[\frac{2}{2-5}\right]\left[\frac{2}{2-6}\right]\left[\frac{2}{2-8}\right] = -\frac{2}{3} \times \frac{2}{4} \times \frac{2}{6} = -\frac{1}{9} = -0.1111$$

$$A(2) = \left[\frac{5}{5-2}\right]\left[\frac{5}{5-6}\right]\left[\frac{5}{5-8}\right] = +\frac{5}{3} \times \frac{5}{1} \times \frac{5}{3} = +\frac{125}{9} = +13.8889$$

$$A(3) = \left[\frac{6}{6-2}\right]\left[\frac{6}{6-5}\right]\left[\frac{6}{6-8}\right] = -\frac{6}{4} \times \frac{6}{1} \times \frac{6}{2} = -27 = -27.0000$$

$$A(4) = \left[\frac{8}{8-2}\right]\left[\frac{8}{8-5}\right]\left[\frac{8}{8-6}\right] = +\frac{8}{6} \times \frac{8}{3} \times \frac{8}{2} = +\frac{128}{9} = +14.2222$$

$P(I) = 5 \qquad TC(1) = 2 \qquad TC(2) = 5 \qquad TC(3) = 6 \qquad TC(4) = 8 \qquad TC(5) = 12$

$$A(1) = \left[\frac{2}{2-5}\right]\left[\frac{2}{2-6}\right]\left[\frac{2}{2-8}\right]\left[\frac{2}{2-12}\right] = +\frac{2}{3} * \frac{2}{4} * \frac{2}{6} * \frac{2}{10} = +\frac{1}{3} * \frac{1}{1} * \frac{1}{3} * \frac{1}{5} = +\frac{1}{45} = +0.0222$$

$$A(2) = \left[\frac{5}{5-2}\right]\left[\frac{5}{5-6}\right]\left[\frac{5}{5-8}\right]\left[\frac{5}{5-12}\right] = -\frac{5}{3} * \frac{5}{1} * \frac{5}{3} * \frac{5}{7} = -\frac{625}{63} = -\frac{625}{63} = -9.9206$$

$$A(3) = \left[\frac{6}{6-2}\right]\left[\frac{6}{6-5}\right]\left[\frac{6}{6-8}\right]\left[\frac{6}{6-12}\right] = +\frac{6}{4} * \frac{6}{1} * \frac{6}{2} * \frac{6}{6} = +\frac{3}{1} * \frac{3}{1} * \frac{3}{1} * \frac{1}{1} = +27 = +27.0000$$

$$A(4) = \left[\frac{8}{8-2}\right]\left[\frac{8}{8-5}\right]\left[\frac{8}{8-6}\right]\left[\frac{8}{8-12}\right] = -\frac{8}{6} * \frac{8}{3} * \frac{8}{2} * \frac{8}{4} = -\frac{4}{3} * \frac{8}{3} * \frac{4}{1} * \frac{2}{1} = -\frac{256}{9} = -28.4444$$

$$A(5) = \left[\frac{12}{12-2}\right]\left[\frac{12}{12-5}\right]\left[\frac{12}{12-6}\right]\left[\frac{12}{12-8}\right] = +\frac{12}{10} * \frac{12}{7} * \frac{12}{6} * \frac{12}{4} = +\frac{6}{5} * \frac{12}{7} * \frac{2}{1} * \frac{3}{1} = +\frac{432}{35} = +12.3429$$

TC 12

TC15

Subroutine a Test Problems

n = RA	TC1 = R10 = 12 / A1 = R15	TC2 = R11 = 8 / A2 = R16	TC3 = R12 = 6 / A3 = A17	TC4 = R13 = 5 / A4 = A18	TC5 = R14 = 2 / A5 = A19	t = R8: 0	4	8
						Sum of $A_i * \exp(-t/TC_i)$		
1	1					1	0.71653	0.51342
2	2	-1				1	0.82653	0.65875
3	3	-3	1			1	0.84342	0.70021
4	4	-5	3	-1		1	0.92439	0.80317
5	5	-7	8	-8	3	1	0.25365	0.54048

Check to see how convolution of TCs of two first order components compares with convolution of TCs of two higher order components.

Test No	TCs of 3 first order systems that make up one third order system	TC of third order system	TCs of 2 first order systems that make up one second order system	TC of second order system	TCs of third and second order systems	TC of combined third and second order systems [Convolute 5 second order systems]	Convoluted TC of two first order systems with TCs same as the third and second order system TCs
1	3 4 5	12.98	6 7	13.94	12.98 13.94	27.08	28.88
2	1 2 3	6.43	12 15	28.92	6.43 28.92	35.19	36.11
3	10 12 14	39.03	1 2	3.17	39.03 3.17	42.11	42.34
4	2 4 6	12.8598	5 7	12.8266	12.8598 12.8266	25.98	27.56
5	3 4 5	12.9843	2 10	12.2163	12.9843 12.2163	25.67	27.04

TC 16

Interpretation of the Exercise on Page TC16

The time constant of a component or system is defined as the time required for the component or system to attain 63.2% ($= 1 - E^{-1}$) of its final response following a step change in the input. This definition was originally applied to first order systems only, because for first order systems the time constant has an obvious physical interpretation. Despite the fact that a time constant defined in this way for higher order systems does not have a simple physical interpretation, the definition has come to be used for components and systems of all orders.

What the exercise of page TC16 shows is as follows: Suppose we have two first order components with time constants TC1 and TC2. Suppose we also have two higher order (in this case third order and second order) components with the same time constants, TC1 and TC2. If we combine the two first order components in a chain, we have a second order system with time constant TC3. If we combine the third and second order components in a chain, we have a fifth order system with time constant TC4. What page TC16 shows is that we will always have TC3 > TC4.

In other words if we have a chain of higher order components with known time constants, and we wish to estimate the time constant for the whole chain, we can do so by assuming that all the components are first order, and we are guaranteed that the time constant for the string of first order components is a conservative (higher than) estimate than the time constant for the string of higher order components.

HIGHER ORDER TRANSFER FUNCTIONS

All linear transfer functions are completely specified if their zeroes and poles are specified. For example, a third order transfer function has the general form

$$\frac{(s-z_1)(s-z_2)}{(s-p_1)(s-p_2)(s-p_3)}$$

where z_1 & z_2 are the zeroes of the transfer function and p_1, p_2 & p_3 are the poles of the transfer function.

In most applications transfer functions have poles, but no zeroes. There are two principal reasons for this: (1) Many transfer functions inherently have poles, but no zeroes, and (2) the zeroes of a transfer function are difficult to measure, and in many instances experimental results can be fit to a transfer function only if the simplifying assumption is made that there are no zeroes.

Thus in many applications the general form for a third order transfer function is

$$\frac{1}{(s-p_1)(s-p_2)(s-p_3)}$$

In our HP-67 program the third order transfer function can be considered to be three first order transfer functions in a row.

$$\left[\frac{1}{s-p_1}\right] \cdot \left[\frac{1}{s-p_2}\right] \cdot \left[\frac{1}{s-p_3}\right]$$

In this way the HP-67 program can handle higher order functions