NASA Contractor Report 3631

CARE III Phase III Report - Test and Evaluation

J. J. Stiffler, J. S. Neumann,
and L. A. Bryant

CONTRACT NAS1-15072
NOVEMBER 1982

NASA

NASA Contractor Report 3631

CARE III Phase III Report - Test and Evaluation

J. J. Stiffler, J. S. Neumann,
and L. A. Bryant
Raytheon Company
Sudbury, Massachusetts

Prepared for
Langley Research Center
under Contract NAS1-15072

NASA
National Aeronautics
and Space Administration

Scientific and Technical
Information Branch

1982

TABLE OF CONTENTS

TABLE OF CONTENTS (CONCLUDED)

LIST OF FIGURES

FIGURE

LIST OF FIGURES (CONTINUED)

FIGURE

TEST CASE 3B'

LIST OF FIGURES (CONTINUED)

LIST OF FIGURES (CONTINUED)

FIGURE

LIST OF FIGURES (CONCLUDED)

FIGURE

LIST OF TABLES

TABLE

1.0 Introduction

The Phase III version of CARE III (Computer-Aided Reliability Estimation, version three) is the product of a series of efforts designed to help estimate the reliability of complex redundant systems. Although designed specifically for use in fault-tolerant avionics systems, the approach is of a general nature and can be used to model a variety of redundant structures.

The first CARE program developed at the Jet Propulsion Laboratory in 1971, provided an aid for estimating the reliability of systems consisting of a combination of any of several standard configurations (e.g., stand by - replacement configurations, triple modular redundant configurations, etc.). CARE II and CARE III were subsequently developed by Raytheon, under contract to the NASA Langley Research Center. CARE II substantially generalized the class of redundant configurations that could be accommodated, and included a coverage model to determine the various coverage probabilities as a function of the applicable fault recovery mechanism (detection delay, diagnostic scheduling interval, isolation and recovery delay, etc.).

CARE III further generalized the class of system structures that can be modelled and greatly expands the coverage model to take into account such effects as intermittent and transient faults, latent faults, error propagation, etc. In order to accomplish this, it was necessary to depart substantially from the approaches taken in the earlier CARE efforts. The nature of, and reasons for, this departure are discussed in the CARE III PHASE II Report, Mathematical Description, Section 2. This Phase III version of CARE III is a further refinement of the CARE III approach, with the current status of the program reported on here.

2.0 INTENT OF PHASE III

The third phase of the CARE III project has been oriented towards the evaluation and refinement of the Phase II version of CARE III. Various stress tests, single and double fault, and various consistancy tests have been defined so as to test many of the assumptions and approximations used in the most recent version of the program.

During the course of this evaluation a few inadequate numerical techniques or their improper implementations have become apparent. Most of these inadequacies have been corrected, although a few are beyond the scope of the current effort. All of the problems encountered, their severity and the corrective actions taken are addressed in this report.

2.1 Selection of Test Cases

Because CARE III is so versatile and the set of possible input conditions so large, it is difficult to test it and to verify its accuracy with any degree of completeness. The major emphasis of the current effort is to restrict the allowed set of input parameters to the point that other, independent, but much simpler models could be developed to verify at least some of the results produced by CARE III. This approach was particularly useful in testing coverage model results since these intermediate results had heretofore been tested only indirectly through their influence on reliability predictions.

The specific test cases were selected so as to minimize, in CARE III, all factors influencing unreliability except those that could be independently evaluated using these simple models. These latter models were then used to verify the coverage functions (e.g. $P_B(t)$, $P_{\bar{B}}(t)$, $P_L(t)$, $P_{DP}(t)$, $P_{DF}(t)$) (See Appendix 4 and Ref. 4) produced by CARE III under various choices of the remaining unrestricted parameters. These parameters were chosen specifically to stress the numerical analysis techniques used in the CARE III coverage program, thereby attempting to expose any weaknesses in these techniques.

Other test cases, specifically, the FTMP and SIFT test cases, were selected because previous results were available for comparison, both independently derived results and results obtained using earlier versions of CARE III. These tests were useful in assessing the effect on CARE III of the modifications that have been introduced to improve its accuracy and to correct flaws discovered during the aforementioned tests.

2.2 Test Program Mathematical Model

Verification of the single and double fault models and evaluation of the computational algorithms used in CARE III, was greatly enhanced by the use of two analytical test programs; SFMODL, and DFMODL. These two programs use a Markov mathematical model to calculate, for a restricted set of input cases, some of the intermediate single and double fault results.

By restricting the input transition parameters to be time invarient, and allowing only constant rate functions the CARE III single fault (Figure I) and double fault models (FigureII) can be programmed using standard Markov-model analysis techniques.

In both cases the state diagram models are translated into matrix form as follows:

Single Fault Model Matrix

	A	B	A_E	B_E
A	$-(\alpha+\rho+P_A\delta)$	β	$(1-P_A)C\varepsilon$	0
B	α	$-\beta$	0	$(1-P_B)C\varepsilon$
A_E	ρ	0	$-(\alpha+\varepsilon)$	β
B_E	0	0	α	$-(\beta+\varepsilon)$

Double Fault Model Matrix

	B_1A_2	B_1B_2	A_1B_2
B_1A_2	ϕ_1	β_2	0
B_1B_2	α_2	ϕ_2	α_2
A_1B_2	0	β_1	ϕ_3

Where:

$$\phi_1 = -(\alpha_2 + \beta_1 + \rho_2 + \delta_2)$$
$$\dot{\phi}_2 = -(\beta_1 + \beta_2)$$
$$\phi_3 = -(\alpha_1 + \beta_2 + \rho_1 + \delta_1)$$

These matricies are subsequently reduced and their eigenvalues and eigenvectors are calculated using a series of International Mathematical and Statistical Libraries (IMSL) subroutines. A second series of IMSL subroutines is then called to solve the linear system AX=Y where the columns of A contain the previously calculated eigenvectors, and Y is the input matrix whose columns are the individual right hand sides of the equation AX = Y. These solutions are then used to calculate probability function coefficients. The probability of being in any one state at any time, t, can then be calculated. In general for a Markov-model with i states:

$$P_i = \sum_{j=1}^{i} x_{ij} \exp(-\lambda_j t)$$

where: P_i = probability of being in state i at time t.

x_{ij} = coefficients

λ_j = eigenvalues (assumed here to be distinct)

Intermediate coverage functions consist of these probabilities or combinations of these probabilities.

Source listings of both SFMODL (Single Fault Markov-Model) and DFMODL (Double-Fault Markov-Model) are included in Appendix 1.

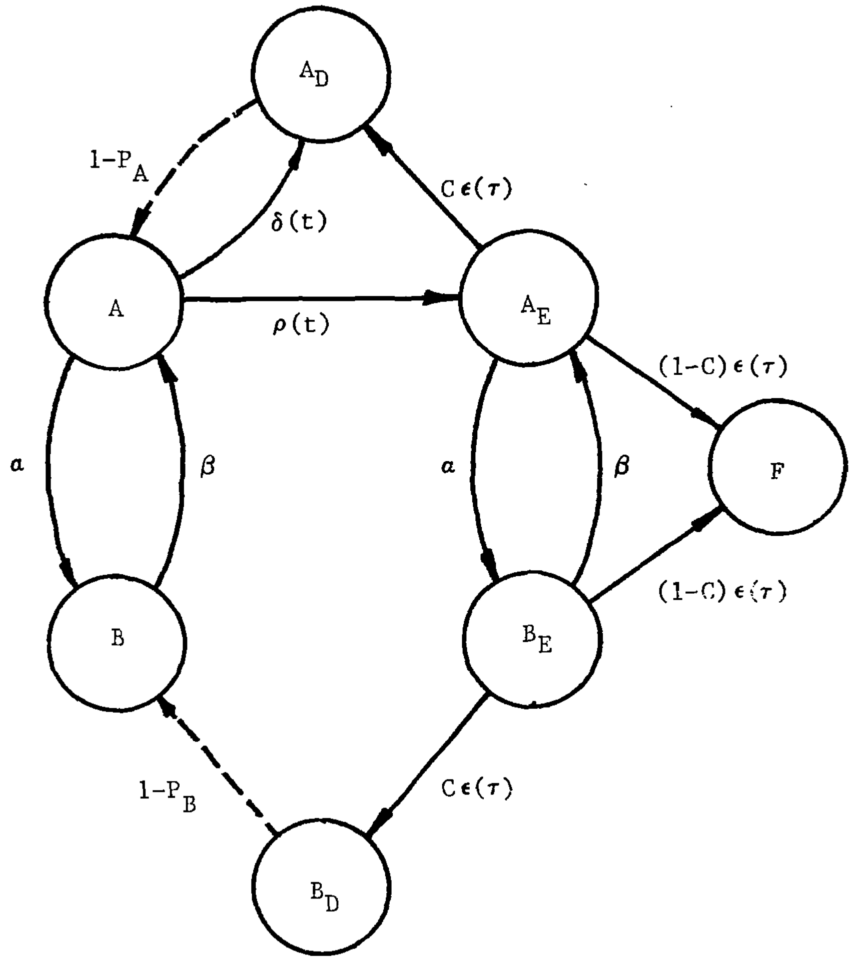

FIGURE I
CARE III SINGLE FAULT MODEL

α = active to benign transition rate

β = benign to active transition rate

$\rho(t)$ = error generation rate

$\epsilon(t)$ = error propagation rate

$\delta(t)$ = fault detection rate

t = time from entry into active state

τ = time from entry into error state

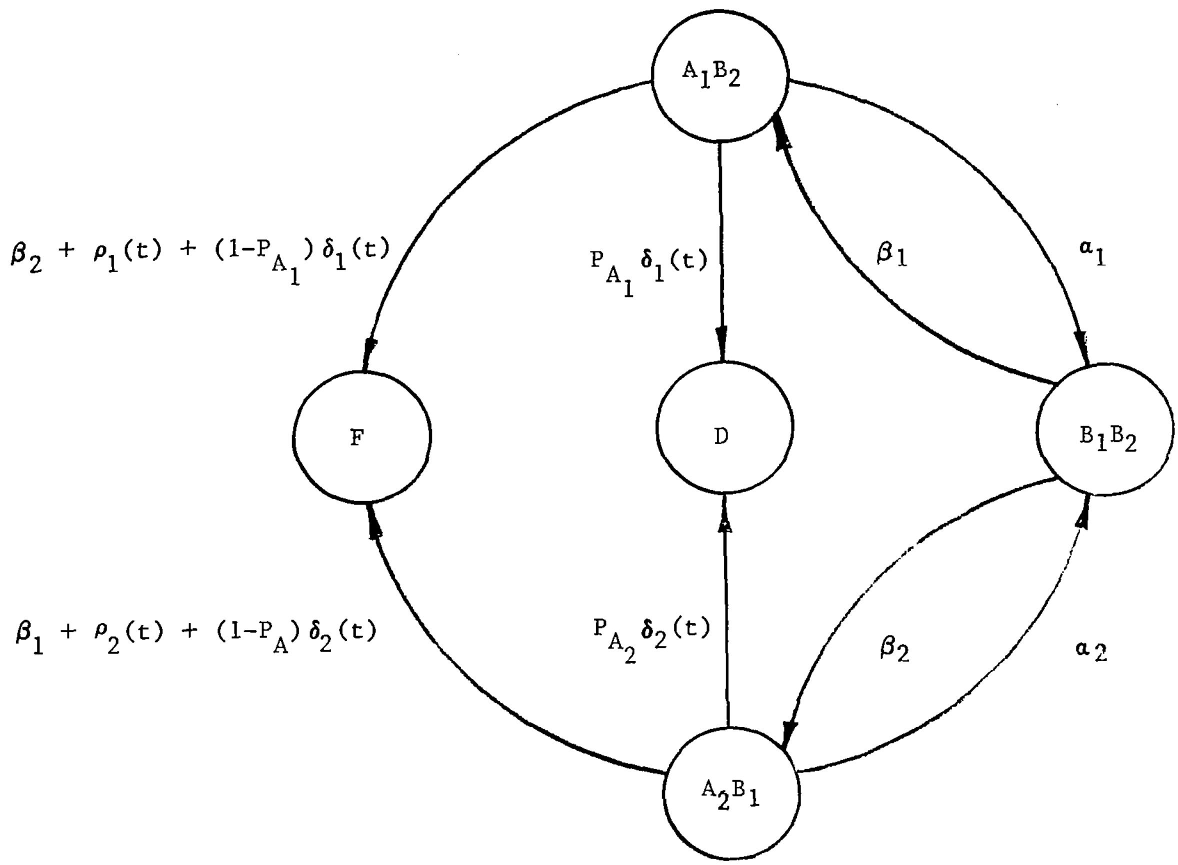

FIGURE II

CARE III DOUBLE FAULT MODEL

3.0 Accomplishments

3.1 Release One Evaluation

3.1.1 Evaluation Procedure

The initial evaluation of CARE III began with a comparison of its results to those obtained using the Markov-model program described above. The latter program was used to determine, for one special case, the probability of a system failure by time t (Q(t) values). This special case involved two stages, the first subject to failures occurring at a rate $\lambda_1 = 10^{-2}$ failures/hour, and the second failure occurring at a rate $\lambda_2 = 10^{-20}$ failures/hour. With no critical fault pairs, and at time t << 20 hours the only significant contribution to a system failure is a coverage failure in stage 1 (Q_{10} (t)). A second, program was designed to model transient faults in a similar fashion. The desired result in this case, however, was not a failure probability, but instead the probability that a fault has been detected as permanent by time t (P_{DP} (t)). CARE III was run with identical inputs so as to derive results valid for comparison. The performance of CARE III under extreme conditions was also evaluated by running a series of stress tests similar to those listed in Table 1. These stress cases include permanent, transient, intermittent and software type fault tests.

3.1.2 Evaluation Results

The comparison between the test program results and the CARE III results, at that time, were felt to be quite satisfactory, with some improvements to be gained by modification of the coverage model integration routines (e.g. doubling criteria).

The transient fault comparison, however, highlighted two errors in numerical computation:

1. The treatment of a function's steady state value. (For all times t greater than the last calculated value, the function was equated to zero).
2. An inconsistency with the function h_{DPT} $(t|x_i)$ used in the definition of $R_{x_i}(t)$ (ref. CARE III PHASE II Report, Mathematical Description, Table 1).

The results of the stress tests indicated some problems with accumulated error generated while solving the VOLTERRA type integral.

3.1.3 <u>Modifications Incorporated Into Second Release</u>

All of the above mentioned problems were addressed, and either corrected or improved for the Phase III second release. A summary of the modifications made to CARE III during this period is as follows:

1. A self modifying capability was added to the doubling difference parameter (DBLDF) (ref. CARE III PHASE II Report, User's Manual, Section 3.1) in COVRGE. In certain instances a DBLDF value may be appropriate for all but a few of the functions. Under these conditions DBLDF will be appropriatly modified, for that function only, and that function will be recomputed. If the second attempt is unsuccessful, a third try will not be attempted. Instead, the program will be haulted with diagnostic messages printed. Additionally, array sizes were increased to allow for smaller DBLDF values.
2. The inconsistency in the function $R_{x_i}(t)$ was determined to be a second, inappropriate, integration of H_{DPT} $(t|x_i)$. This integration was removed thereby correcting the calculation of $R_{x_i}(t)$ in CARE3.

3. It was discovered that when the propagated error coverage probability C equals 1.0 and no critical fault pairs exist, zero valued fault vectors were being unnecessarily computed. This was corrected in the CARE3 program.

4. The unnecessary restriction of not allowing the λ failure rate to be greater than 1.0 in CAREIN was corrected. λ may now be greater than 1.0.

5. The M input parameter was modified in CAREIN and CARE3 to allow zero values, thereby providing a software fault modelling capability.

3.2 Release Two Evaluation

3.2.1 Evaluation Procedure

The procedure used to evaluate release 2 is essentially the same as that used in the evaluation of release 1. The Markov-model programs were up-graded to include several of the coverage single fault functions; P_A, P_B (benign), P_{NB} (not benign), P_L (latent), and P_{DP} (detected as permanent), in addition to the double fault function p_{DF}. A greater emphasis was placed on the stress test results, both intermediate and final, so as to characterize any sources of accumulated error and to verify any assumptions used. A large part of the effort during this evaluation was directed towards the more complicated and error prone coverage program (COVRGE).

3.2.2 Evaluation Results

Upon examination of the stress-test results it became apparent that accumulated error in the integration routines is still a potential problem area.

Because of the extreme nature of these stress tests the integration routines are forced to convolve functions whose maximum time, tmax, differ by a much greater ratio than had been tested before. Any small amount of accumulated error seen with standard cases becomes greatly magnified in these extreme cases, to the extent that in one case (3d - see Table 1) an intermediate single fault function (P_A) became unbounded. A number of successful measures were implemented to help rectify this situation.

3.2.3 Modifications Incorporated Into Third Release

The easiest and most obvious modification, although not without cost, was to increase the array sizes, thereby allowing smaller initial step sizes. This modification improved the situation only slightly. It became apparent that it would be desirable for the step-size doubling procedure to be changed to a halving procedure after a function has reached its peak (or valley). As the slope of the function decreases, as it peak is approached, the step size goes through a series of doublings, and becomes quite large. This step size is then too large to accurately capture the function as it again begins to rapidly vary. Due to the nature of COVRGE the incorporation of a halving ability was beyond the scope of this phase. Instead, the doubling algorithm was modified to restrict doubling for 25 steps after a function peaks or dips. This approach has worked out as the most effective compromise.

During this evaluation the single fault intermediate function P_a presented the greatest problems. Because of the severe extremes of its constituent functions, P_a was often the only unacceptable function. In order to make P_a less radical, and hence more manageable, its numerical implementation was divided into a series of smaller computations, with the more rapidly varying functions separated from the slower varying functions. This approach was also successfully implemented in the second single fault recursion, F_X (t) (see Table 2A, CARE III PHASE II Report, Mathematical Description).

The effect of these changes was two-fold; the acceptable range of input parameters was greatly extended, and an increased accuracy was achieved for the 'easier' input cases. In order to detect the few situations where accumulated error could lead to erroneous results, a test has been incorporated into COVRGE to halt the program and produce a diagnostic message. Final results will be erroneous if the following situation does not hold:

$$\int_0^t \phi_x(\tau)d\tau \leq 1.0$$

where $\phi_x(t)$ = kernal of F_X

A routine to test whether user inputs are within the value range, specified in CARE III Phase II Report, User's Manual, was also incorporated into CAREIN during this phase.

4. Interpretation of Results

Some representative results of the tests conducted during this phase are plotted in Figures 1-7; other results are tabulated in Tables 1-4. (See Appendix 3 for corresponding CARE III input files.) The figures emphasize comparison of the coverage calculations obtained using CARE III to those obtained using the Markov-model discussed in Section 2. Three types of plots are used to facilitate this comparison: linear, log, and log-log.

The tables list the various parameters used in each test along with the unreliability at user defined flight time (FT) determined by CARE III under each of these sets of conditions. Since, in general, it is not possible to get an independent verification of these results, their main value is as a reasonableness test - do these results appear to be mutually consistant?

The tests can be conveniently grouped into three categories: single-fault-model tests; double-fault-model tests, and consistency tests. Some observations and conclusions about the tests in each of these categories are discussed in the following paragraphs.

4.1 Single-Fault-Model Tests

The coverage and reliability parameters used in the single-fault-model tests are listed in Table 1. Selected intermediate results obtained during these tests are plotted in Figures 1 through 5 along with, wherever possible and appropriate, the corresponding Markov-model results.

In general, the CARE III results and the Markov-model results compare extremely well. The only discrepancies that appear to be significant are those seen on the log plots. These discrepancies show up, however, only when the function in question has become insignificantly small. In any case, they are caused by the fact that the CARE III model uses a user-specified parameter (TRUNC) to determine when to truncate the calculation of an intermediate value and set that value to zero. (The Markov-model does not need an analogous parameter, since it, in effect, determines an analytic expression for the function of interest). The discontinuities evident in the CARE III log plots in Figures 1-5 are caused by these truncation events. This is easily seen by comparing the plots in Figure 4 with the corresponding plots in Figure 4'. TRUNC was changed from its normal 10^{-4} value in Figure 4 to a value of 10^{-6} in Figure 4'. As expected, the location of the discontinuities shifted and their magnitude decreased with the decreased TRUNC value. It should be noted, however, that the effect of these discontinuities on the primary result of interest (the system unreliability) is entirely negligible. The difference between the TRUNC = 10^{-4} and TRUNC = 10^{-6} unreliabilities predicted by CARE III for the Figure 4/4' test case, in particular, is .002%. The same truncation effect, incidently, explains the discontinuities seen in the P* and Q plots in Figures 1-7; again, these discontinuities are insignificant so far as the results of interest are concerned.

As previously noted, the test cases were deliberately selected to stress some of the coverage model numerical evaluation procedures. One effect of this is particularly evident in the log-log plots (and almost entirely obscured in the linear and log plots) of Figure 3 and especially Figure 5. It is seen that several of the coverage functions initially exhibit very rapid changes for a short interval followed by a period of very slow changes followed, in turn, by another interval of relatively rapid changes. Such functions severely stress the numerical integration and recursion algorithms that require them as inputs. In order to accommodate the initial, rapidly varying part of the function, the integration step size must be extremely small. In order to keep the time needed to evaluate the integration from becoming excessive, the step size must be allowed to increase rapidly as the rate of change of the function decreases. Unfortunately, any step size selection rule compatible with both of these requirements tends to introduce significant error when the function again begins its rapid variation. It was to accommodate such functions that some of the modifications discussed in Section 3.2.2 were introduced. Their effectiveness can be seen by comparing the log-log plots of the CARE III results and those obtained using the Markov-model.

One area in which CARE III evidently still needs work is the transient model. As seen in Figure 2, the P* results (and hence the P* + Q results) tend to oscillate. Apparently, this is due to round-off error resulting from the calculation of $R_x(t)$, the reliability of an element subject to well-covered transients (so that $R_x(t) \approx 1$) and then using $1 - R_x(t)$ in subsequent calculations. Such oscillations are clearly incorrect and the evaluation procedures should be modified to remove them. This should not be difficult to do; unfortunately, time and budget constraints prevented this from being accomplished as part of the current effort.

4.2 Double-Fault-Model Tests

A Markov-model was also written to provide a means of evaluating independently some of the CARE III coverage functions associated with the double-fault model. The postulated double-fault model test cases are listed in Table 2. Again because of time and budget constraints, only two of these test cases were actually run. The results of these runs are shown in Figures 6 and 7.

In general, the agreement between the Markov-model and the CARE III results appear to be satisfactory, although less exact than the agreement between the single-fault model results. This is somewhat surprising since the CARE III double-

-fault model uses techniques similar to those used in the single-fault model, and the model itself is considerably simpler. It is believed that modifications to the CARE III double-fault model numerical evaluation procedures similar to those made in the single-fault case would virtually eliminate these discrepancies. Again, however, time and budget constraints precluded this effort.

It should be remarked that the differences in the two sets of results could be at least in part due to inaccuracies in the Markov-model results. The Markov-model, for example, was unable to produce any answer in the double (1,2) fault case for test 4B, (cf. Table 2). This particular case is especially interesting in that CARE III produces a bimodal curve for the intensity of double-fault coverage failures, $p_{DF}(t)$ (Figure 7b). Since the Markov-model did not work in this case, and since bimodal coverage function curves are apparently unusual, the question arises as to whether this result is actually valid. An examination of the physical situation being modelled, however, suggests that the results are at least approximately correct. *

The only difference between the first-occurring and second-occurring faults in this case is that the former has an active-to-benign transition rate of 1/sec. while the latter has an active-to-benign transition rate of 1000/sec. Since the first fault must be benign when the second fault occurs (otherwise the system would fail immediately), and since the benign-to-active transition rate for both faults is slow (1/sec.) relative to the active-to-benign transition rate of the second fault, the rate of change of p_{DF} is dominated entirely by the active-to-benign transitions of the second fault. It is easily verified, in fact, that the initial value of $p_{DF}(t)$ is $\beta+\rho$ = 120/min. This initial activity should persist for only about .001 seconds (1.6 x 10^{-5}min.), since that is the expected time for the second fault to become benign. Since no double-

*An alternate Markov modeling technique at NASA-Langley did reproduce the CARE III results.

fault failure can occur when both faults are benign, the value of $p_{DF}(t)$ should decrease significantly at this point. Since the benign-to-active transition of both faults are identical (1/sec.), exactly half of the transitions out of the doubly benign state will be to the state in which the first fault is active and the second is benign. The occupancy probability of this state should peak at roughly 1 second (.016 mins.). Furthermore, since transitions from this state back to the doubly benign state occur at the relatively slow 1/sec. rate, the rate of change of $p_{DF}(t)$ following this second peak should be much slower than that following the first peak. These observations describe precisely the results predicted by CARE III (remember that the Figure 7 plots are log-log plots), thus verifying, at least qualitatively, that the CARE III double-fault coverage model is performing correctly in this otherwise unverified case.

4.3 Consistency Tests

Consistency tests run during this phase were of two types: 1) The test cases developed during this phase and used primarily to test the single-and double-fault coverage models were also allowed to run to completion, thereby producing reliability estimations that could be compared from test to test and evaluated for consistency. 2) Some of the test cases run during Phase I of the CARE III program were re-run, again to verify consistency of results and to determine whether any of the changes made during this phase of the program significantly altered any of the earlier results.

4.3.1 Test Cases Developed During This Phase

Several observations can be made concerning the consistency of the results presented in Table 1:

1) The failure probablity decreases when the error propagation rate increases if C = 1 and all other parameters remain the same

(compare test cases 1a and 1h, 1c and 1i, 1d and 1g). This initially counter-intuitive result is clearly correct when it is observed that, when C = 1, all propagated errors are detected; thus, the quicker they propagate, the shorter the latency of the fault.

2) The previous statement also holds for long-term transients (compare test cases 2c and 2c') but the reverse holds for short-term transients (test cases 2a and 2a'). This can be explained by the fact that, while transients behave much like permanent faults if the active period of the fault is long compared to the other coverage parameters, the quick propagation of errors resulting from short-term transients increases the likelihood that they are detected as permanent (P_A=1, P_B=0). Since the latency period for short-term transients is, by definition, short in any event, this latter effect dominates.

3) When the probability that any unit survives for the interval of interest is kept constant, the effect of a nonconstant hazard rate is to increase the failure probability regardless of whether the hazard rate is an increasing or a decreasing function of time (compare the numerical results of test case 1h with cases 1e and 1f). This is clearly true, since a non-uniform hazard rate concentrates the failures at the beginning (when the hazard rate is decreasing function of time) or at the end (when it increases with time) of the interval in question. In either case, the likelihood of double faults is increased.

4) A less-than-perfect probability of detecting propagated errors can have a profound effect on the probability of system failure

(compare test cases 1b and 1b' with all the others). These results can even be roughly verified quantitatively. When C=1, coverage failures occur only when one fault occurs within the latency period of an earlier fault. Since the latency of a fault is of the order of the faster of the fault detection and error propagation delays (reciprocal rates), it is typically of the order of 0.1 to 1 second, or about 0.01% of the total interval of concern here. Thus, the probability of failure due to a double fault is roughly $10^{-4}q^2$ with q the probability of a single fault. The probablity of a failure due to an uncovered single fault is roughly (1-C)q. Since q is approximately 10^{-5} here, this argument would predict a double-fault failure probability of roughly 10^{-14} and a single-fault failure probablity of 10^{-7} when C=.99 and 10-8 when C=.999, thus supporting the observed results.

5) A constant fault-detection rate is somewhat less effective than a constant fault-detection density function (compare test cases 1h and 1h'). When faults are detected at a constant rate δ/sec., the probability that a fault has not been detected after being active for t seconds is $e^{-\delta t}$. When the fault detection density function is a constant δ/sec., this same probability takes the form $1-\delta t$ $(0<t<1/\delta)$. Since $e^{-\delta t} > (1-\delta t)$ for all $0<\delta t<1$, this result is obviously correct.

A constant-rate fault-detection function results, for example, when a diagnostic program is randomly scheduled; a constant-density function results when it is run on a fixed schedule. Equating the two δ-parameters is meaningful, since doing so is equivalent to specifying that the same amount of time is devoted to the diagnostic program in the two cases. Nevertheless, the expected fault-latency period, when diagnostic programs are randomly scheduled at a rate δ/sec. is twice that when programs are run on a fixed schedule every δ^{-1} seconds. To equate the latency periods in the two cases, it is necessary to increase the randomly scheduled diagnostic program rate to 2δ/sec. Doing so decreases the failure probability to that shown in test 1h". In this case, then, the random scheduling strategy is better. This result is less obvious since $e^{-2\delta t}$ is neither always greater than nor always less than $1-\delta t$ over the interval $0<t<\delta t$. The fact that $e^{-2\delta t}$ initially decreases twice as rapidly as $1-\delta t$, and that the probability of a latent fault is therefore decreased correspondingly more rapidly at least adds credibility to the CARE III result.

6) The failure probability decreases as the detection rate increases, but the importance of the detection rate is diminished when the error propagation rate is high and the probability of detecting propagated errors is unity (compare, for example, cases 1a and 1c). These results are clearly as one would expect.

7) The shorter the transient, the less likely it is to cause a system failure (compare test cases 2a, 2b and 2c, and cases 2a', 2b' and 2c'). This should be expected for two reasons: a detected transient fault is less likely to be diagnosed as permanent if it is detected after it reaches the benign state (in fact, PB=0 here); and, the shorter the time spent in the active state, the

less likely will the effect of the fault be present at the time of a subsequent fault.

8) The effect of an intermittant fault depends both on the fraction of time it spends in the active state (contrast test cases 3a and 3b with test case 3d, for example) and on the rates at which it makes transitions between these states. At least when propagated errors are certain to be detected, it is significantly worse for a fault to be almost always benign than for it to be almost always active (compare, for example, test cases 3c" and 3d"). The result is due to the fact that the more time a fault spends in the active state, the more quickly it will be detected and hence the less likely it will contribute to a system failure (since C=1). This is apparently (but not obviously) more significant than the fact that a fault is harmless when it is in the benign state. That this last statement is true is suggested by the following agrument. Suppose that two intermittant faults simultaneously exist for N units of time and that each is active during a given unit of time with probability 1/N, independent both of the other fault and of its own past. Then the probability that the two are never simultaneously active is $(1-1/N)^{2N}$ which, for large N, is roughly $e^{-2} \approx .14$. If the two simultaneously exist for only half as long (N/2 time units) this probability increases, but it is still only $e^{-1} \approx .37$. Thus, even though both faults are almost always benign, the fact that they exist for an extended interval make the probability quite large that they are, at some instant, both active (and hence cause the system to fail). Thus, again, the observed results appear to be at least qualitatively consistent.

9) Additional consistency checks can be made by comparing the results for different types of faults when the parameters are such that their differences should be relatively insignificant. As already observed, for example, when intermittants spend most of the time in the active state they tend to look like permanent faults. This is confirmed by comparing the results of case 3c with those of case 1i.

4.3.2 Test Cases Developed During Phase I (FTMP)

Consistency during the evaluation of CARE III can be seen by examining the results of Table 3. This table tabulates the results of the Fault Tolerant Multi-Processor (FTMP) test cases for a number of CARE III versions. Included are some of the results reported on in the CARE III Final Report, Phase I, Volume I (Table 3.3), the Phase II (release 1) results, and the current Phase III (release 3) results.

As can be seen, there is a consistent trend in the results predicted by CARE III for each of the versions tabulated, the severity of which is proportional to the ratio α/β. This is evidently a consequence of the different restrictions placed on these models. In order to reduce the complexity of the Draper model, any three simultaneous latent faults were equated to a system failure. The Phase I model identifies three simultaneous latent faults with a system failure only if at least two of these faults constitute a critical pair. The Phase II model eliminates all restrictions on the number of simultaneous latent faults; a system failure occurs only when both faults in a critical pair are simultaneously active. This aspect of the model was not changed in Phase III. The differences between the release 1 and release 3 results can be accounted for by the changes made to the way in which coverage results are used in the reliability evaluations. In general, these changes resulted in tighter upper bounds on unreliability by eliminating double counting (e.g. by eliminating the possibility that the same fault leads to two system failures by being involved, possibly at different times, in two critical pairs). As would be expected, the effects of these different restrictions become more apparent as the ratio α/β increases since the amount of time a fault remains latent is an increasing function of this ratio.

4.3.3 Test Cases Developed During Phase I (SIFT)

Similar comparisons were made between the reliability predictions for the Software Implemented Fault Tolerance (SIFT) computer, both those derived by SRI and those produced by the Phase I version of CARE III, with those generated by the current CARE III (Table 4).

In Phase I the coverage model had not yet been programmed, so the coverage inputs to the reliability program were those corresponding to the SRI model in which every fault has a latency of exactly τ seconds. The CARE III coverage model does not allow constant-time latency periods (although this capability could be included if it were of general interest). Consequently, CARE III was exercised, for these examples, using a constant-density fault detection function with a τ seconds mean-time to detection. This is obviously only a rough approximation to a constant τ second detection delay. Even so, the results thus obtained compare very well with those derived both by CARE III using the simulated coverage model inputs and by SRI, thereby giving added credence to the current version of CARE III.

5.0 Conclusions and Recommendations for Further Study

In general, the CARE III results are in excellent agreement with those derived independently, at least in those restricted cases in which independent evaluation is readily obtained. There remain some flaws in CARE III, however, which could not be eliminated under the current budget and schedule constraints. Specifically, the transient model produces results that show instabilities at least under certain conditions, and the double-fault model is less accurate than it should be.

The following recommendations for further study are suggested by the results obtained during this effort:

1. Obviously, the flaws uncovered in the transient and double-fault cases should be analyzed and eliminated.
2. Since it is believed that many of the problems encountered during those tests, including those in the double-fault model, have to do with the fact that the integration routine step size can be changed adaptively only by doubling it, this restriction should be removed. This restriction was initially felt to be acceptable since it was believed that coverage functions tended to have monotonically decreasing derivatives. This is evidently not the case.
3. Under certain conditions, the coverage model evaluation was found to take an excessive amount of computer time. This problem could be eliminated by the integration routine modification recommended above. In addition, however, it is recommended that an alternative, Markov-type coverage model, similar to that described in Section 2.2, be incorporated into CARE III to be used whenever the user specifies only constant rate coverage parameters. Although this case is, somewhat restrictive, it is expected to be the one most commonly used. Even when it is not precisely

applicable, it could be used to obtain preliminary results and to screen out those conditions meriting more careful and more precise scrutiny. When the Markov-model can be used, the run time in many cases could be drastically reduced.

4. Although the testing accomplished here has significantly increased our confidence in CARE III, it can by no means be asserted that CARE III has been completely verified. Further testing is highly recommended.

(Function PNB vs. Time-Mins., CARE III Model)

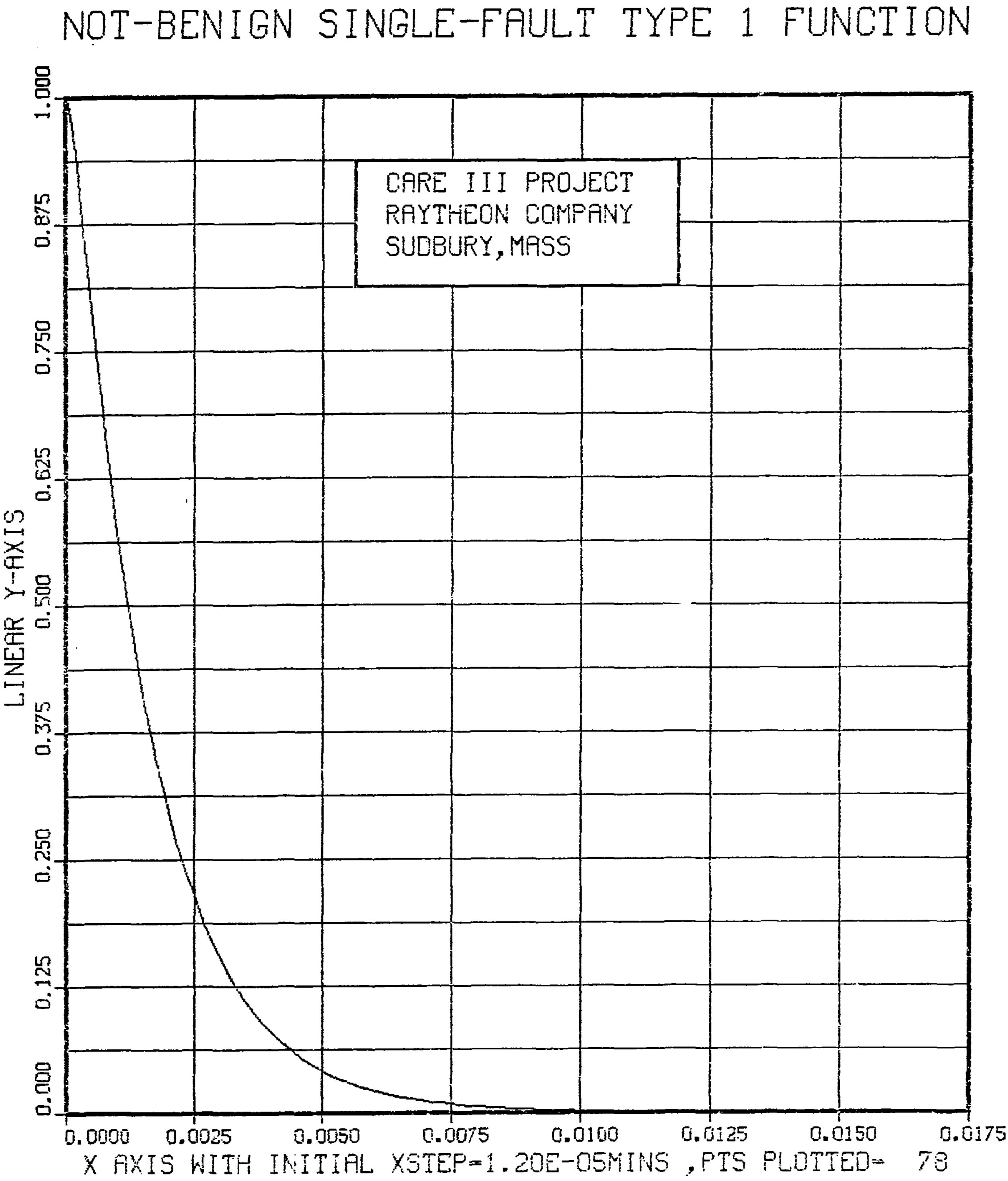

FIGURE 1a
TEST CASE 1A

(Function PNB vs. Time-Mins., Markov Model)

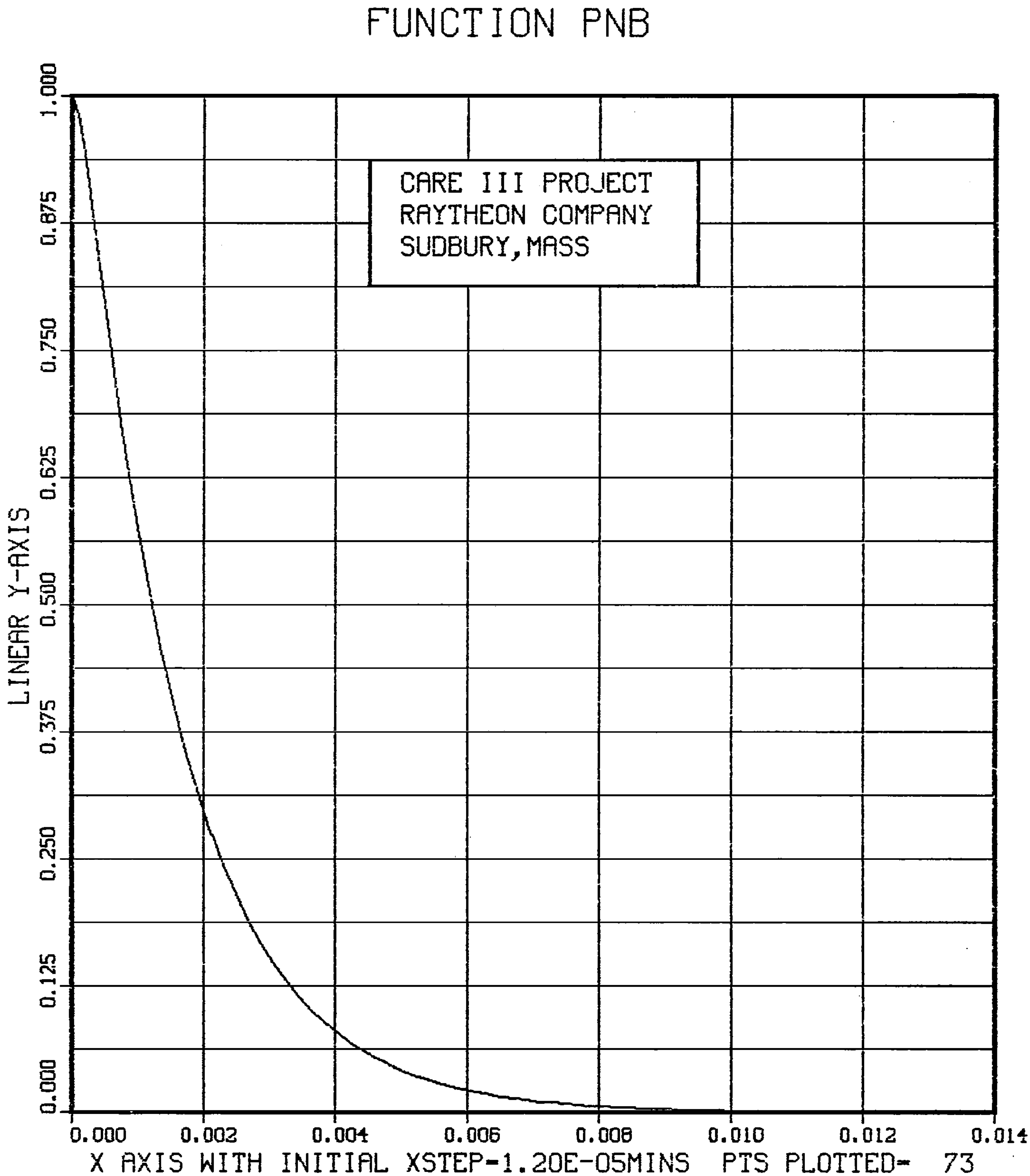

FIGURE 1a'
TEST CASE 1A

(Function PNB vs. Time-Mins., CARE III Model)

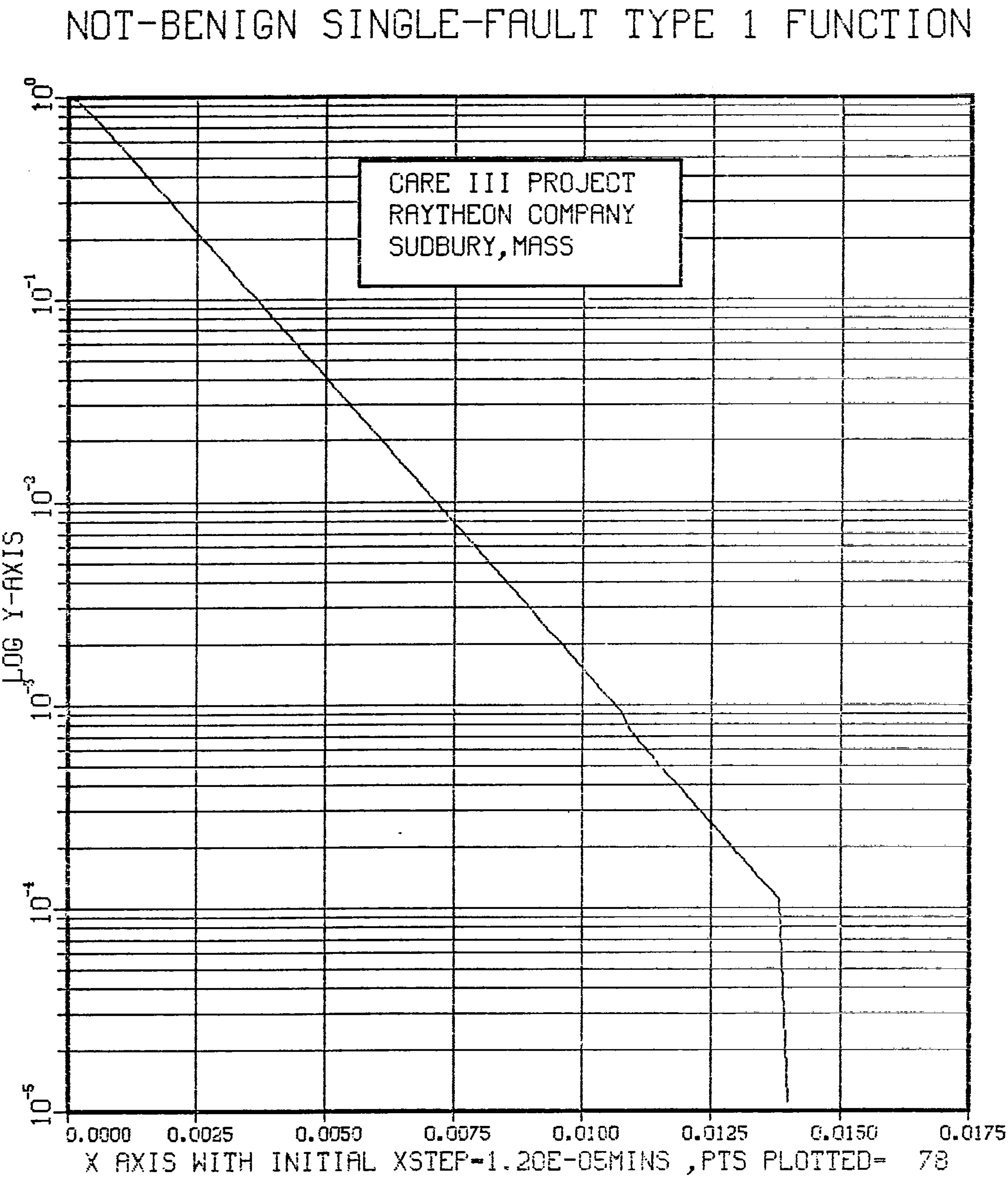

FIGURE 1b
TEST CASE 1A

(Function PNB vs. Time-Mins., Markov Model)

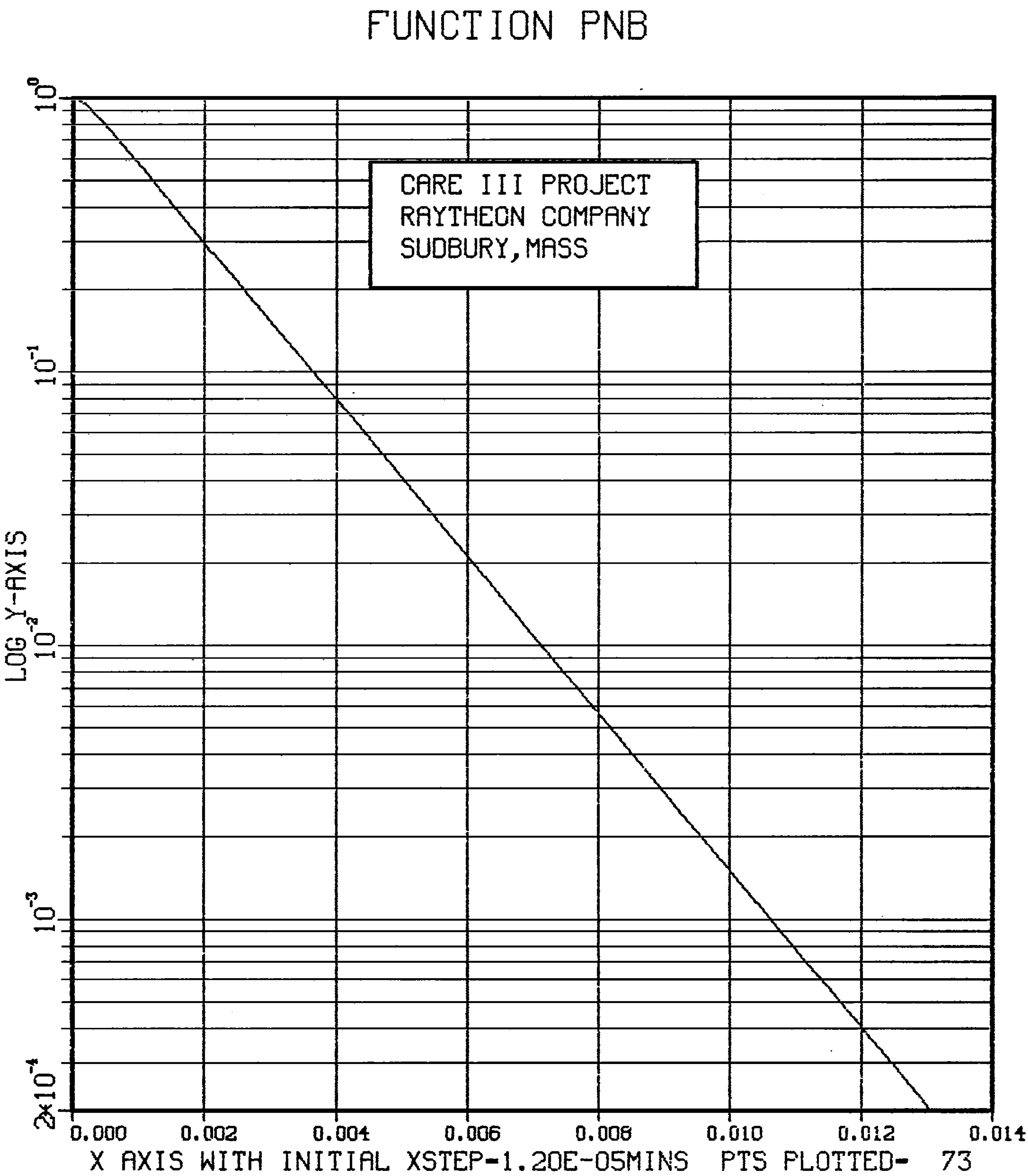

FIGURE 1b
TEST CASE 1A

(Function PL vs. Time-Mins., CARE III Model)

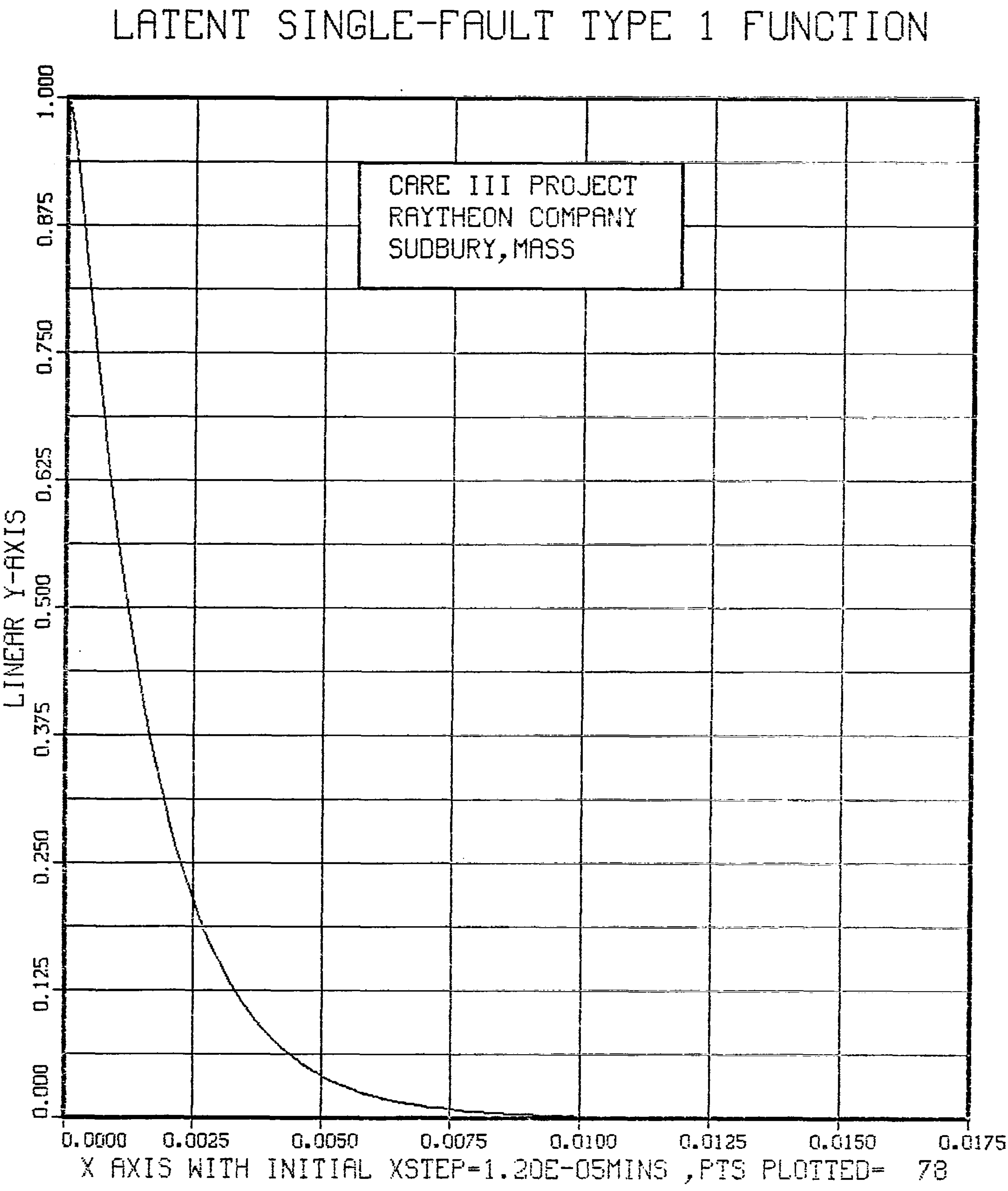

FIGURE 1c
TEST CASE 1A

(Function PL vs. Time-Mins., Markov Model)

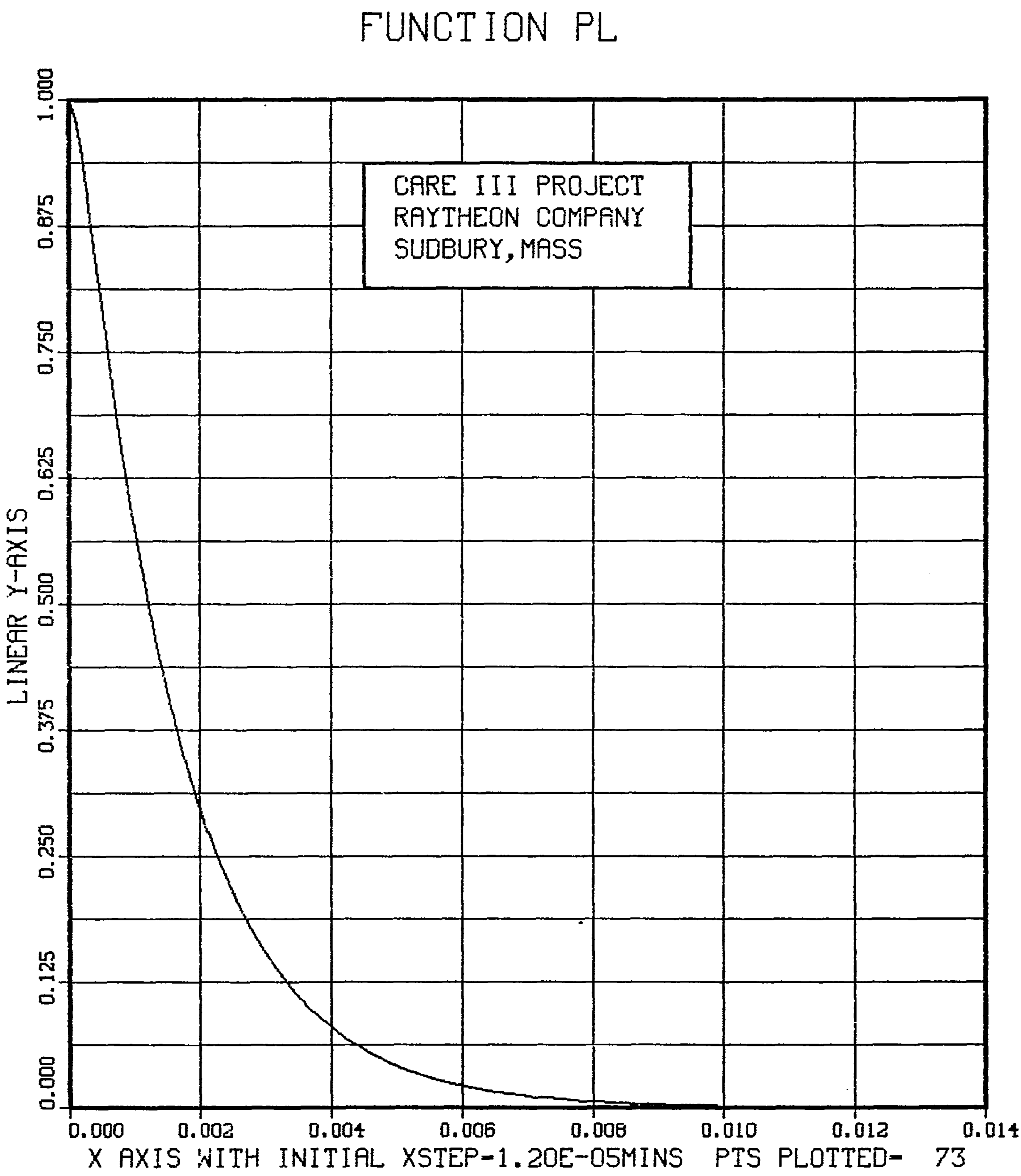

FIGURE 1c'
TEST CASE 1A

(Function PL vs. Time-Mins., CARE III Model)

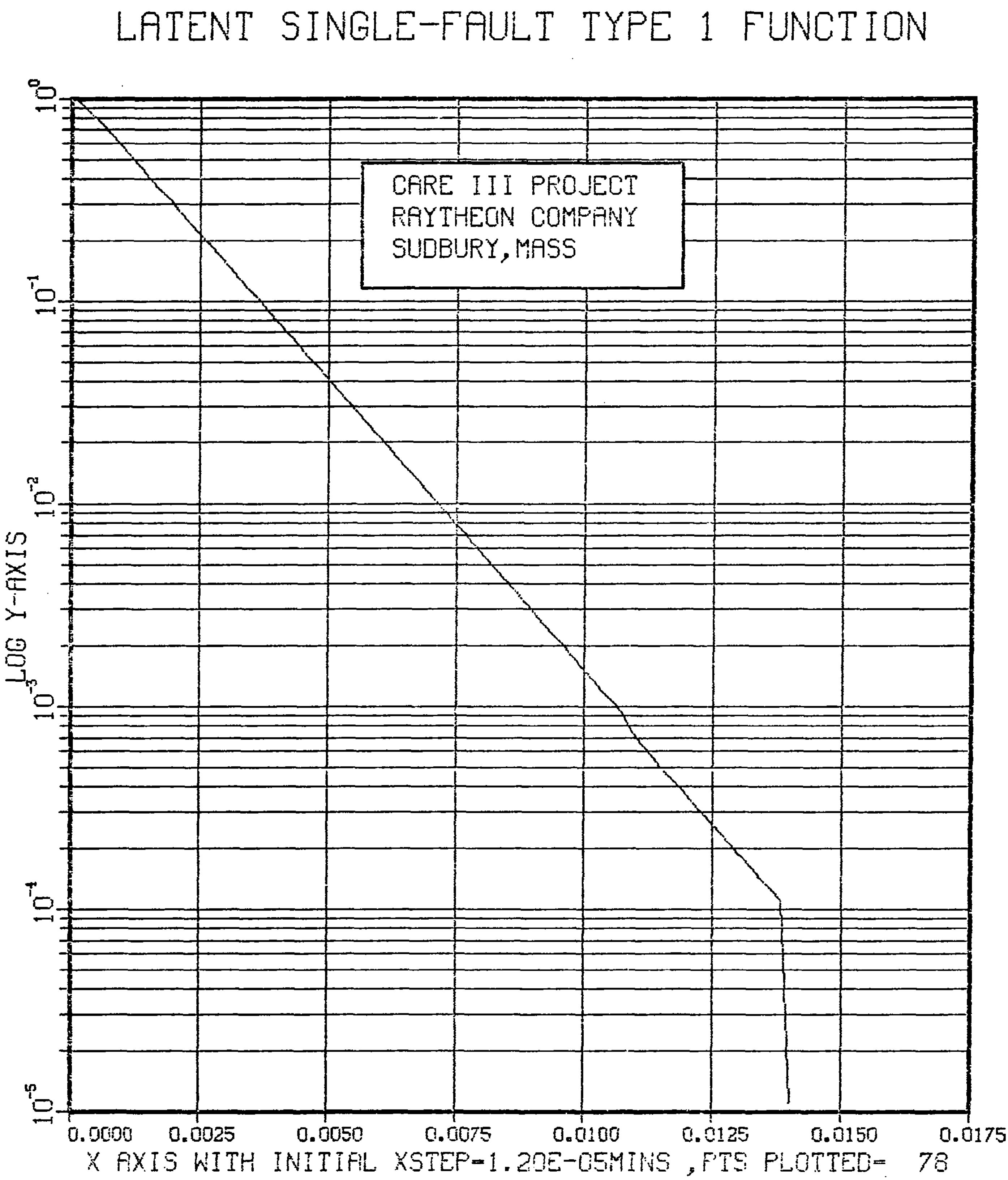

FIGURE 1d
TEST CASE 1A

(Function PL vs. Time-Mins., Markov Model)

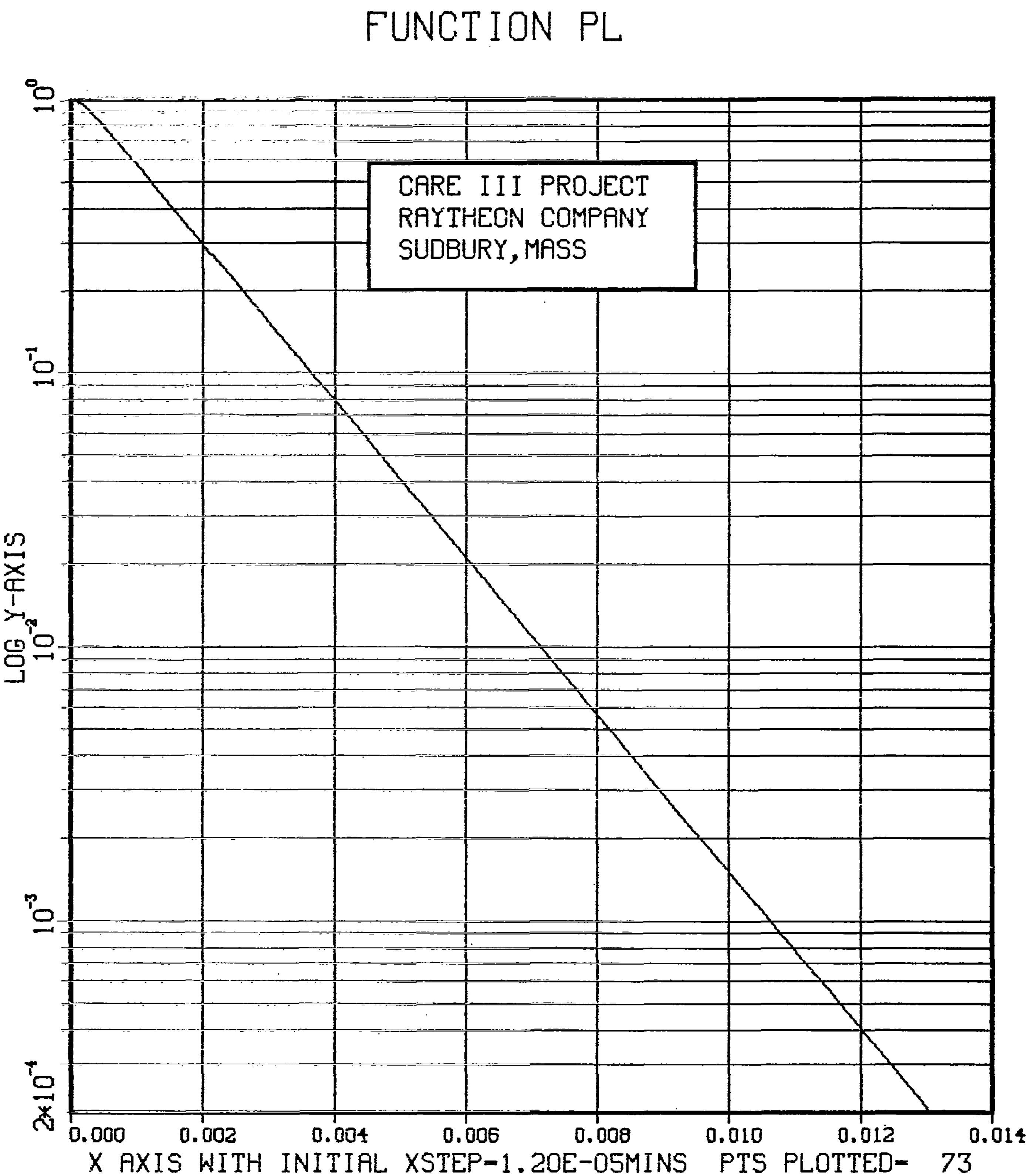

FIGURE 1d'
TEST CASE 1A

(Function PDP vs. Time-Mins., CARE III Model)

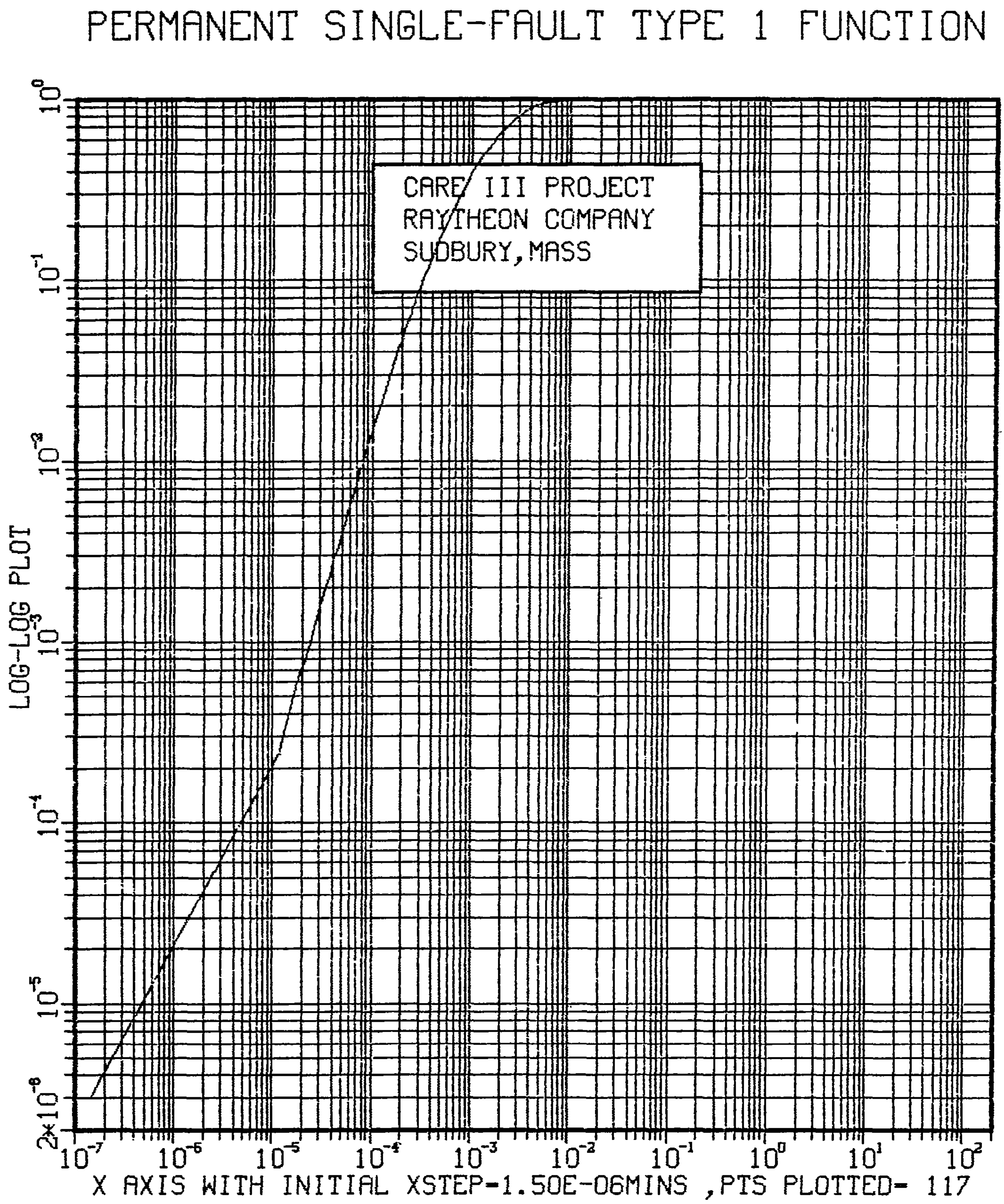

FIGURE 2a
TEST CASE 2C

(Function PDP vs. Time-Mins., Markov Model)

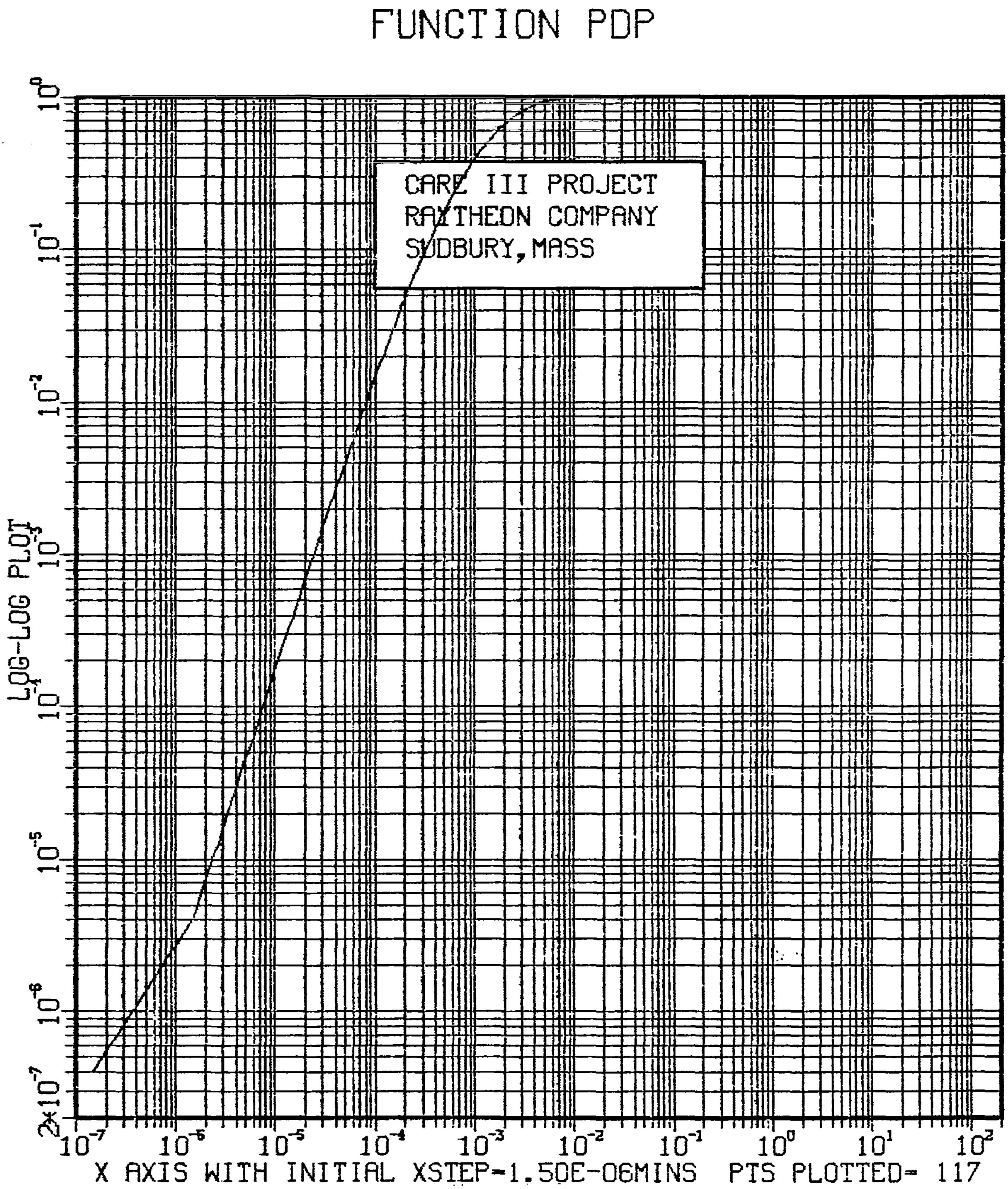

FIGURE 2a'
TEST CASE 2C

(Function PB vs. Time-Mins., CARE III Model)

BENIGN SINGLE-FAULT TYPE 1 FUNCTION

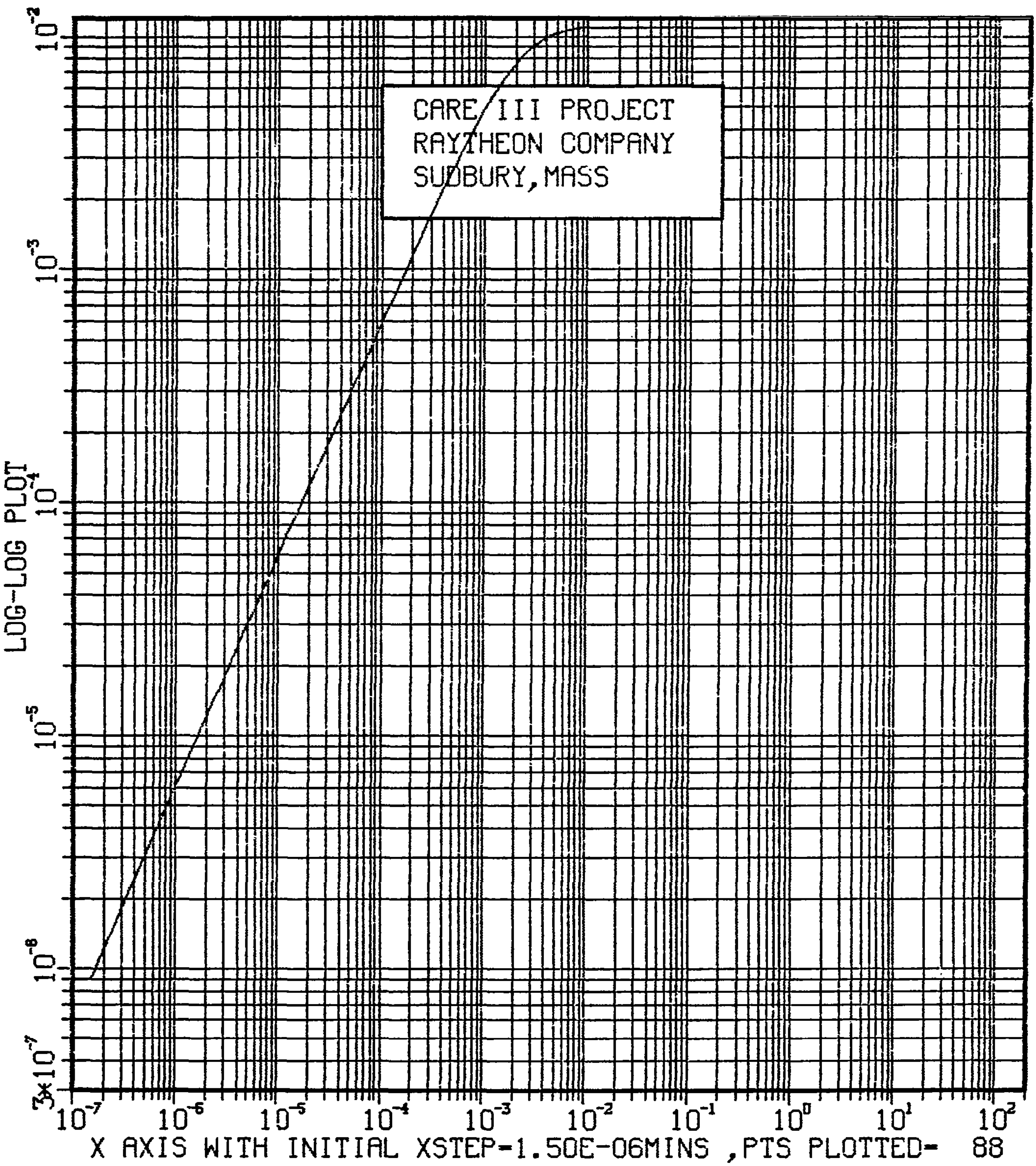

FIGURE 2b
TEST CASE 2C

(Function PB vs. Time-Mins., Markov Model)

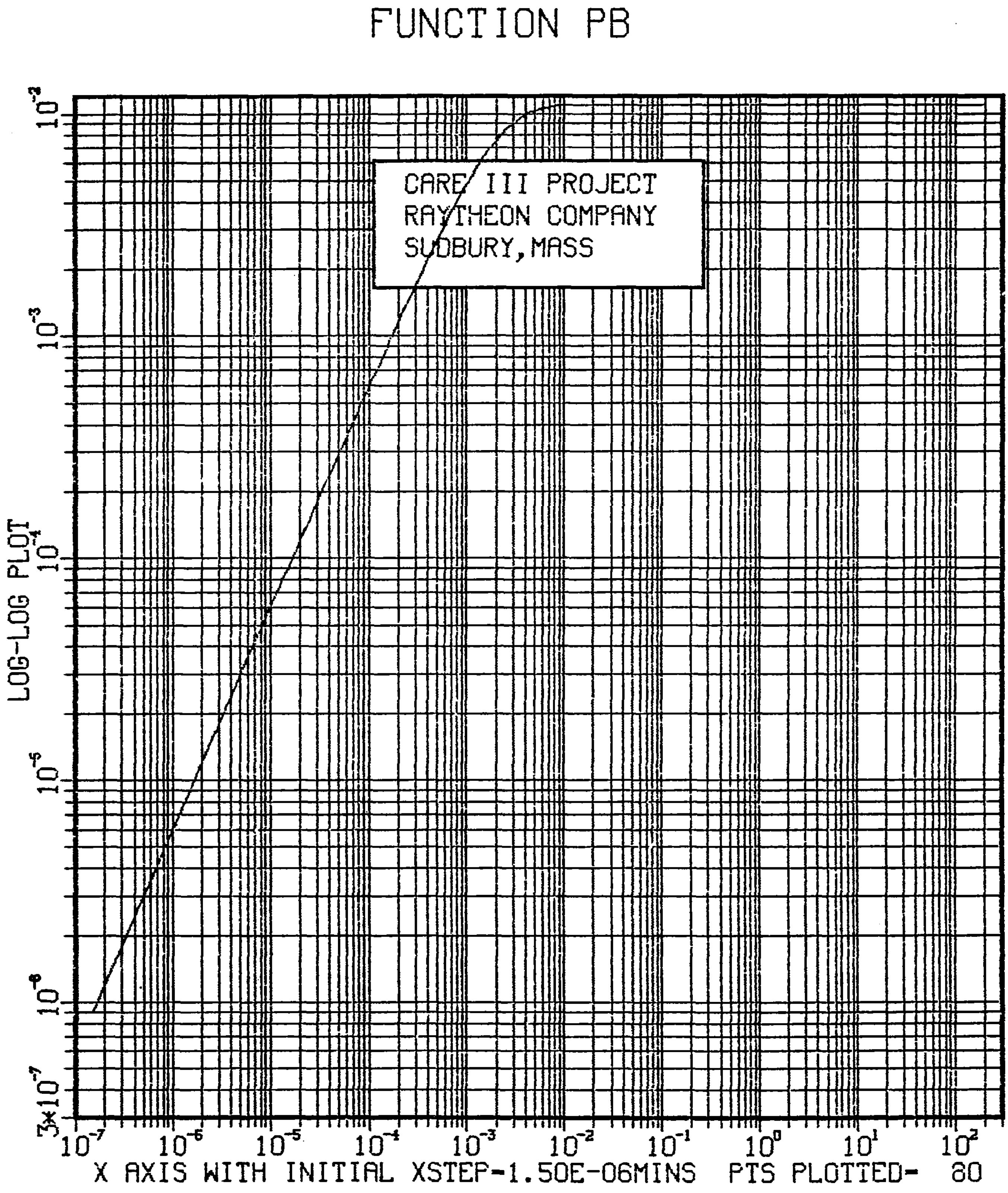

FIGURE 2b'
TEST CASE 2C

(Function PNB vs. Time-Mins., CARE III Model)

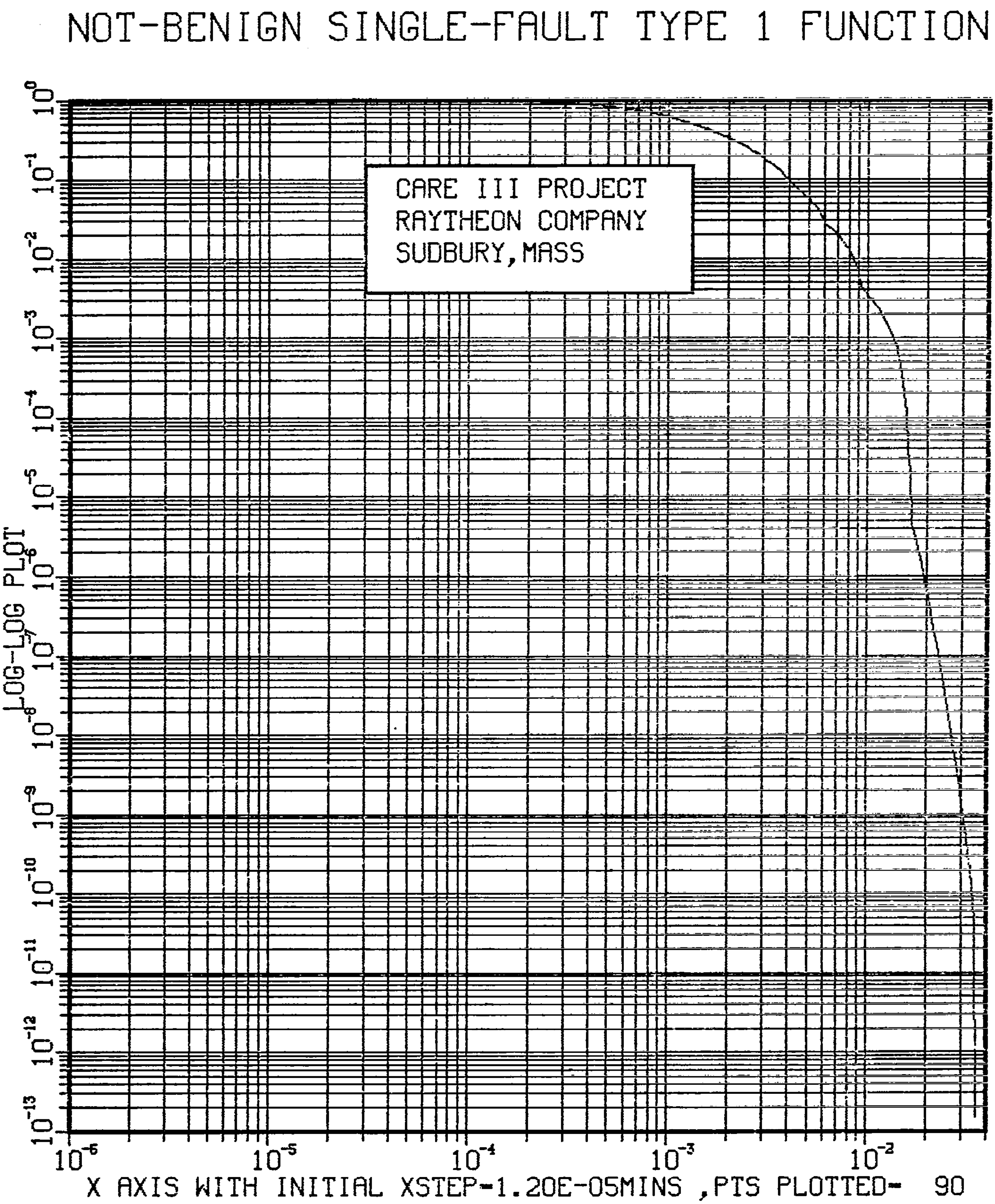

FIGURE 2c
TEST CASE 2C

(Function PNB vs. Time-Mins., Markov Model)

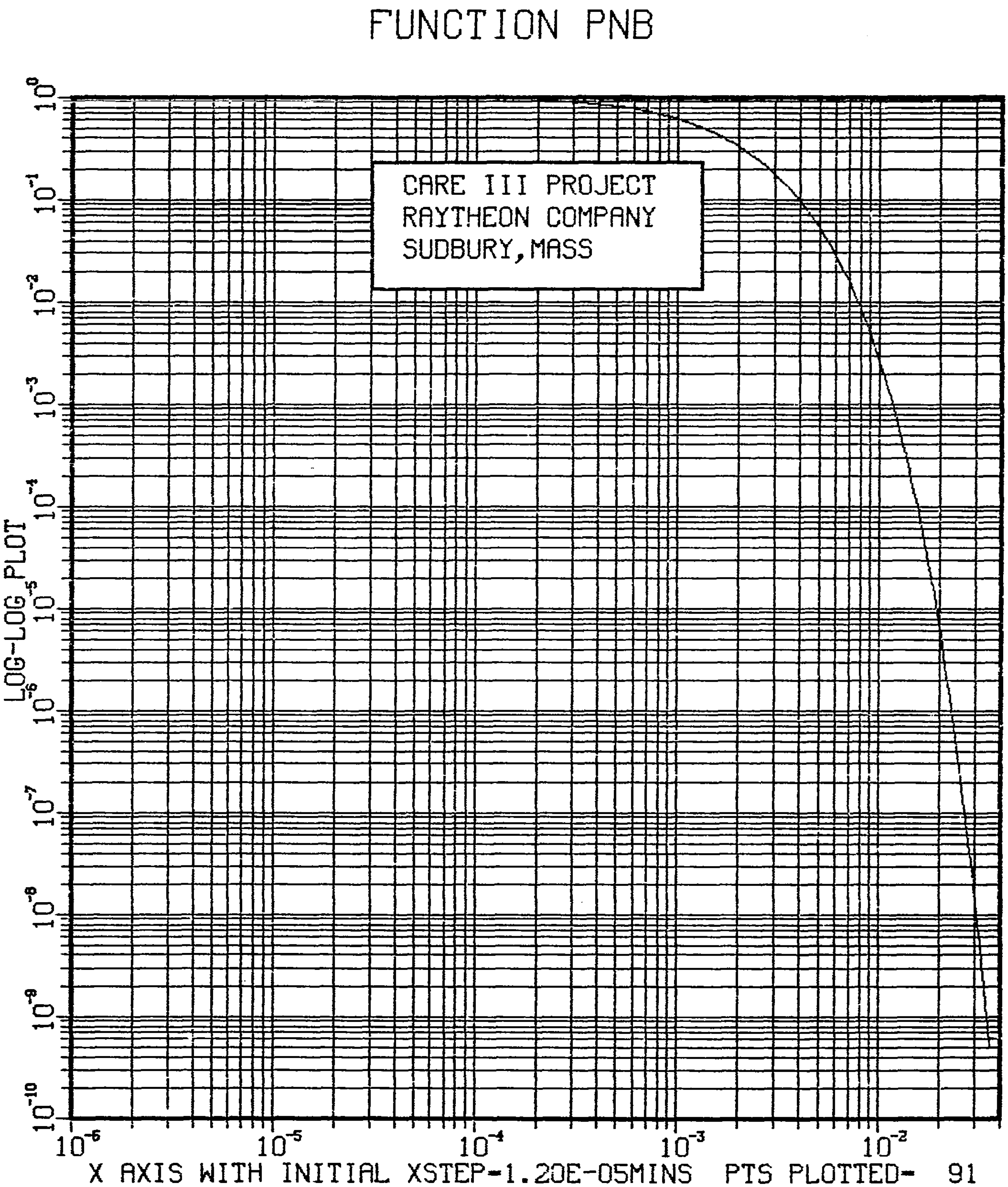

FIGURE 2c'
TEST CASE 2C

(Function PL vs. Time-Mins., CARE III Model)

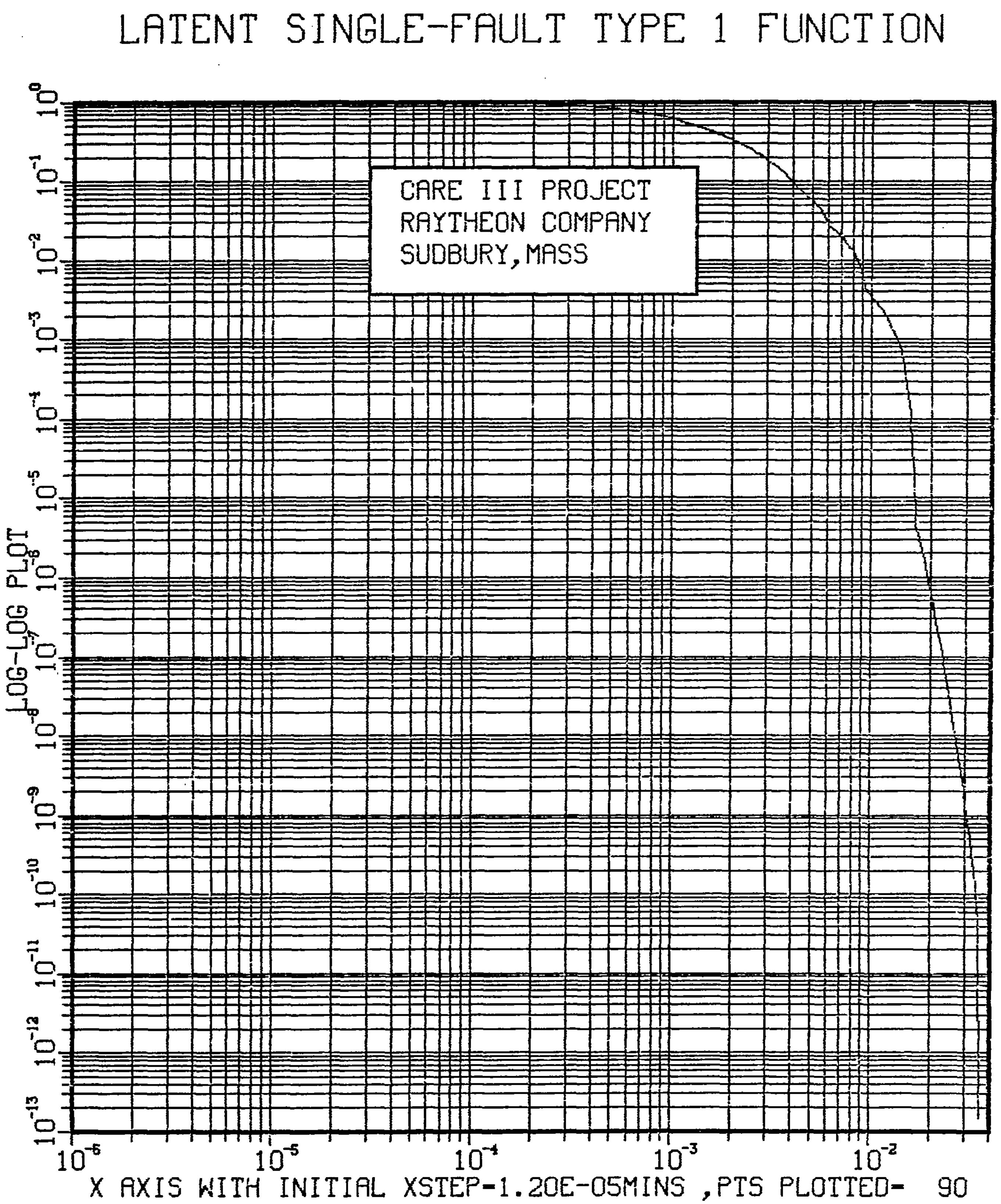

FIGURE 2d
TEST CASE 2C

(Function Q SUM vs. Time-Mins., CARE III Model)

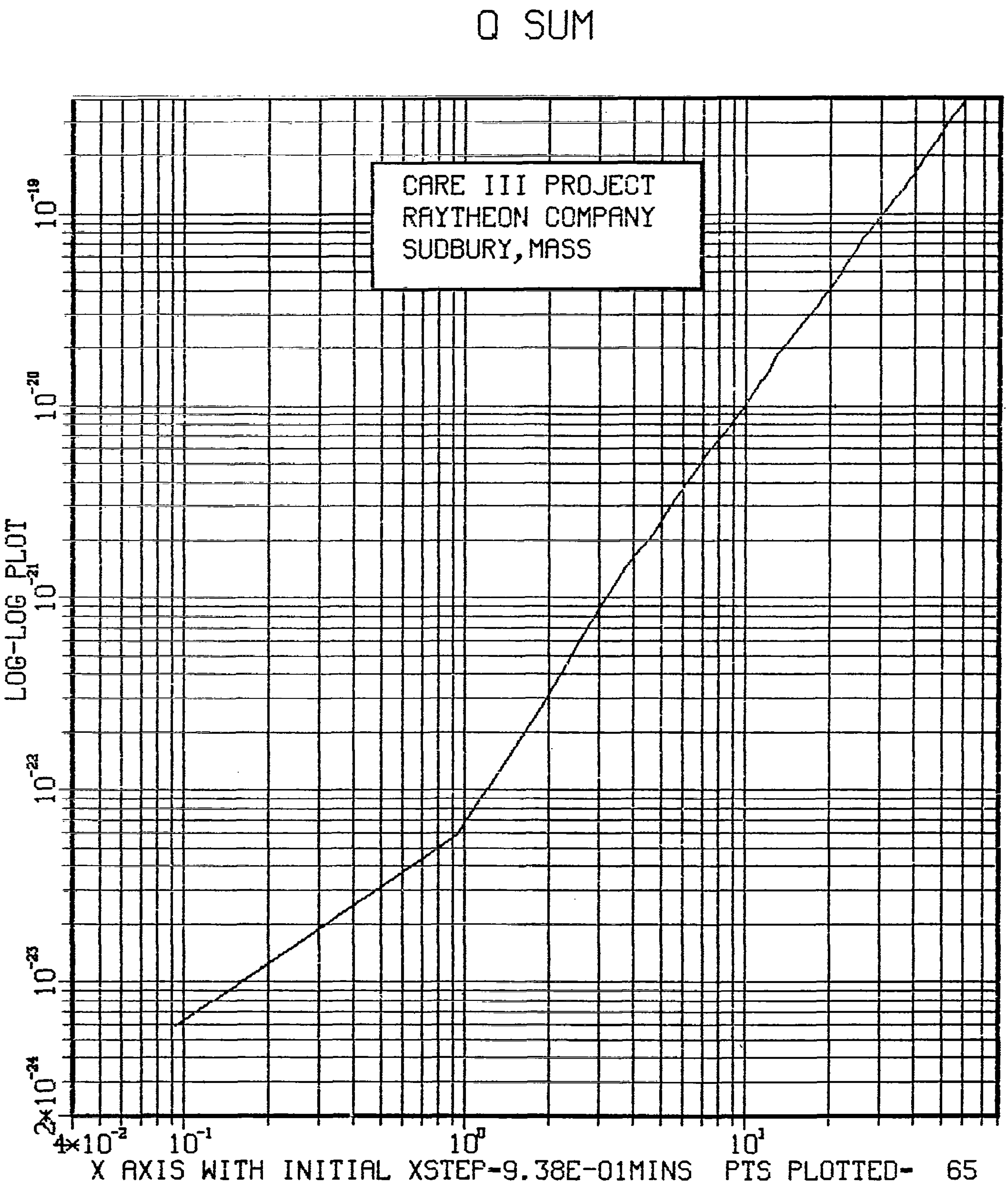

FIGURE 2e
TEST CASE 2C

(Function P* SUM vs. Time-Mins., CARE III Model)

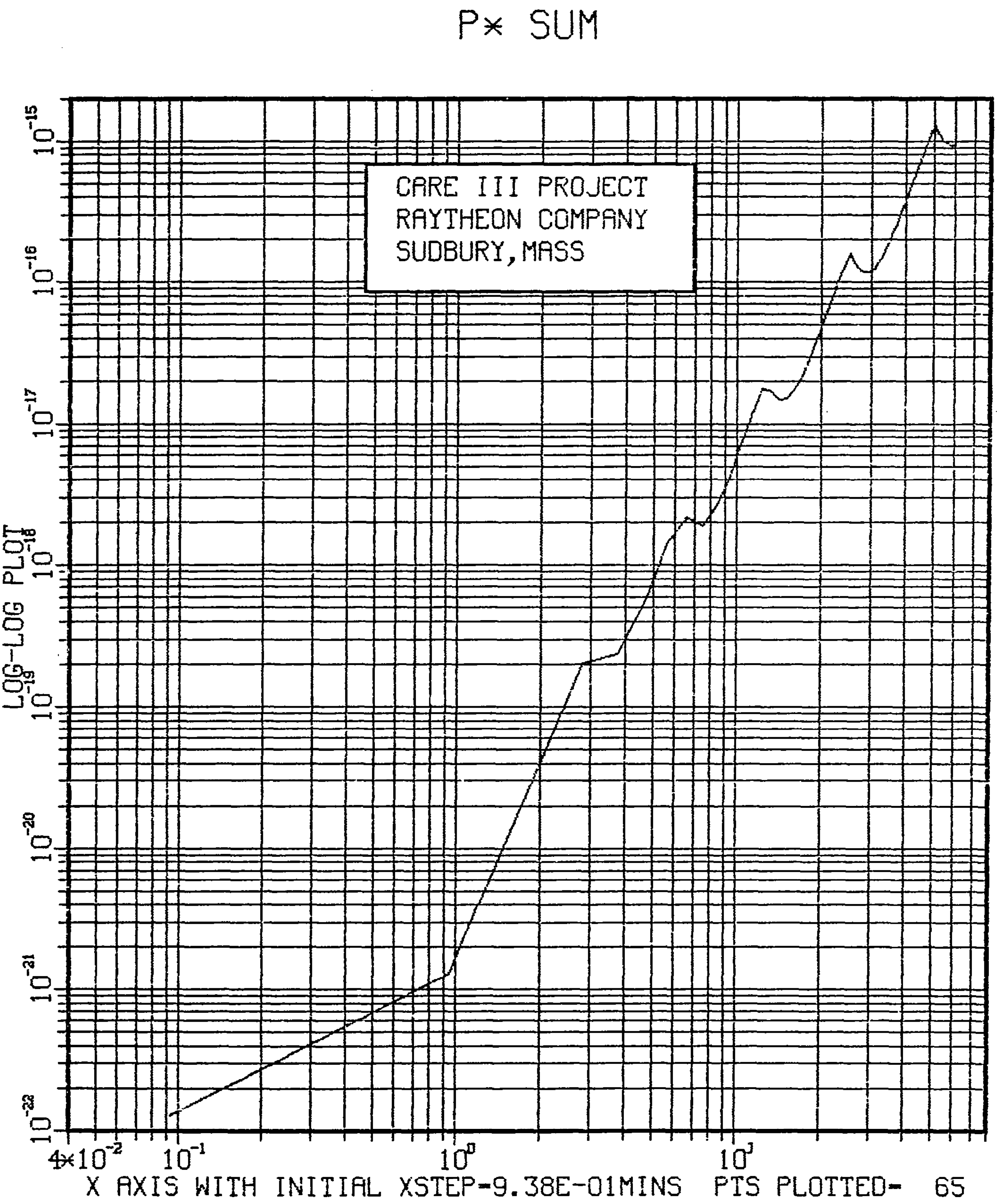

FIGURE 2f
TEST CASE 2C

(Function Q + P* SUM vs. Time-Mins., CARE III Model)

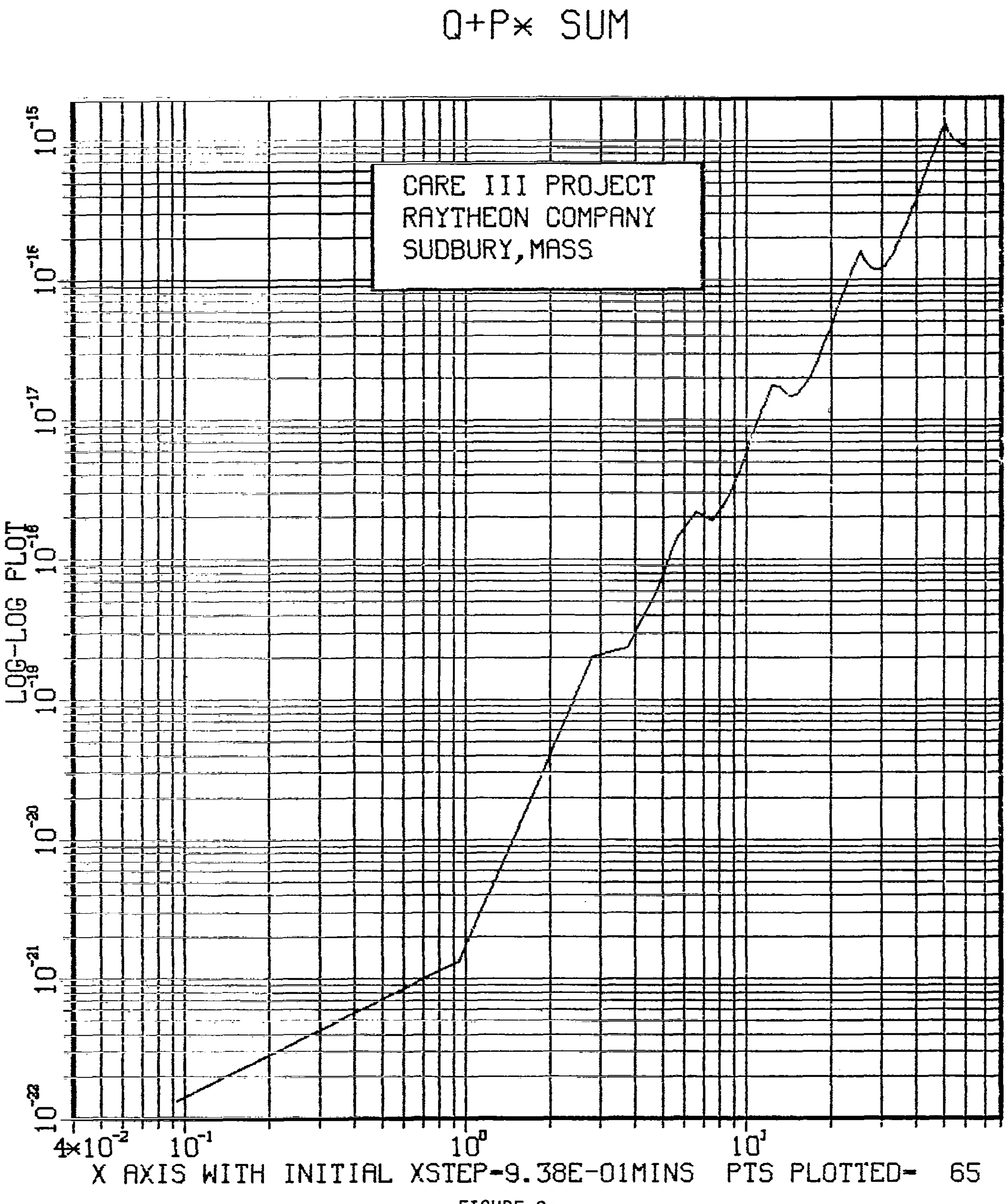

FIGURE 2g
TEST CASE 2C

(Function PB vs. Time-Mins., CARE III Model)

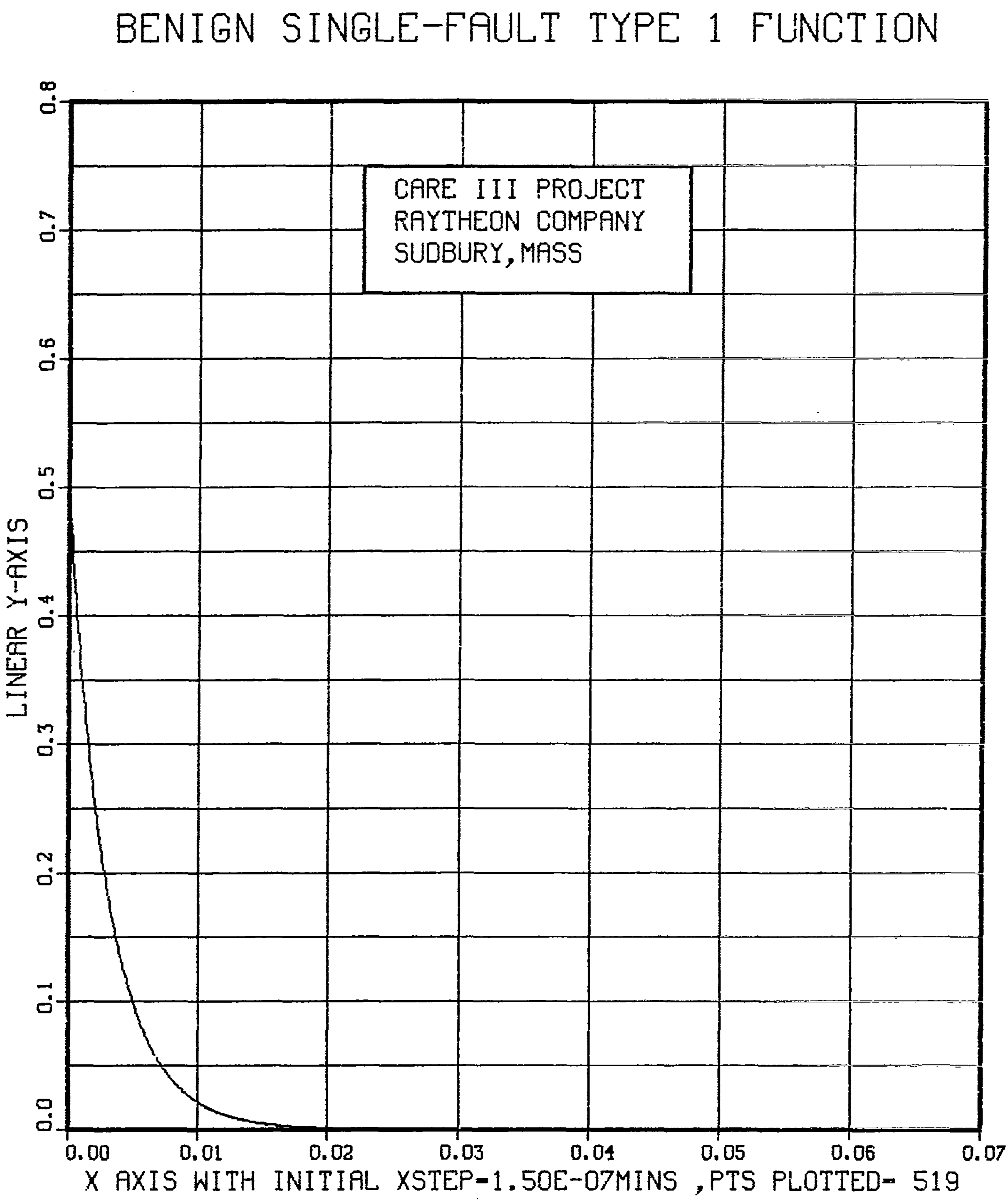

FIGURE 3a
TEST CASE 3B '

(Function PB vs, Time-Mins., CARE III Model)

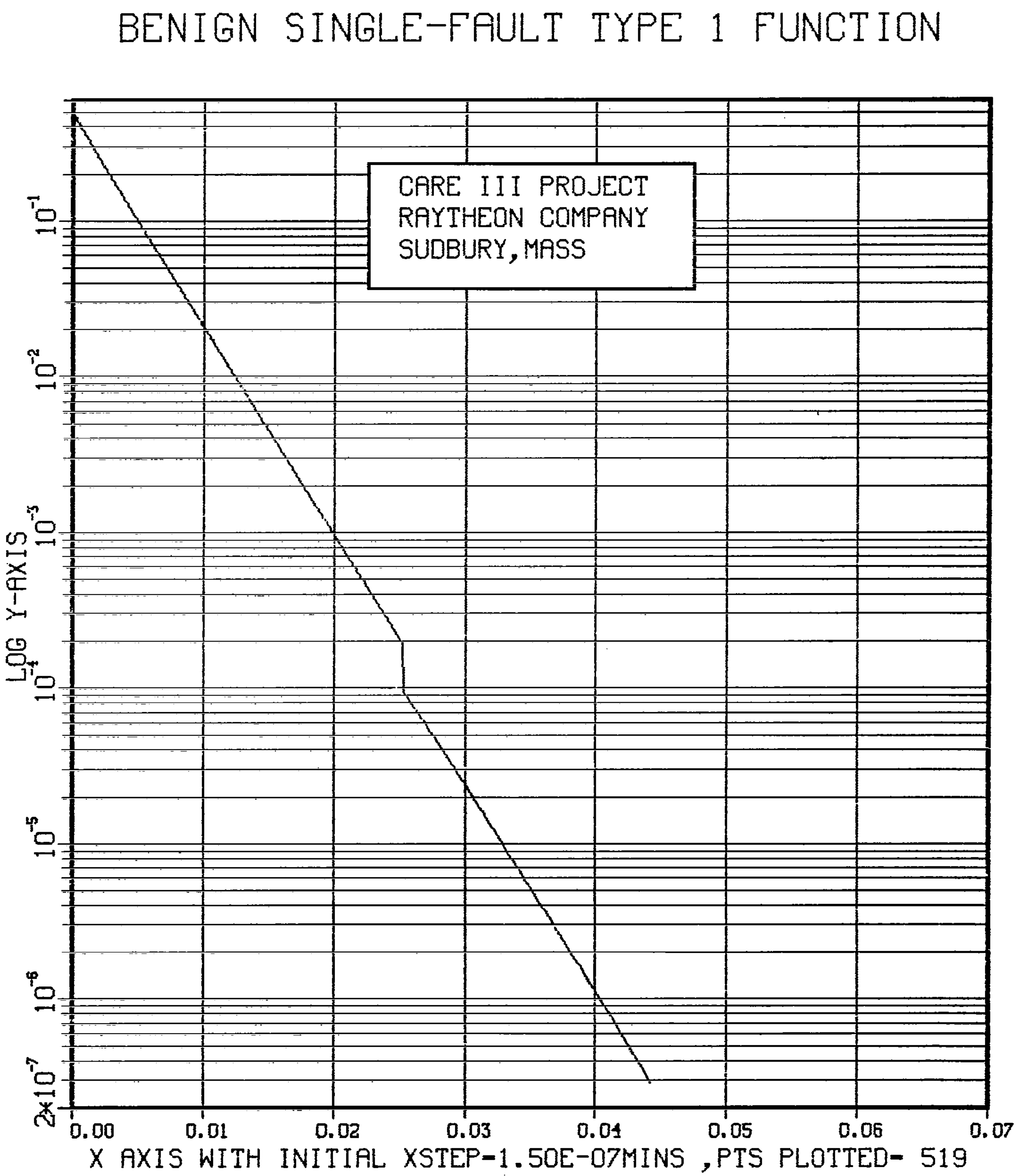

FIGURE 3a'
TEST CASE 3B '

(Function PB vs. Time-Mins., CARE III Model)

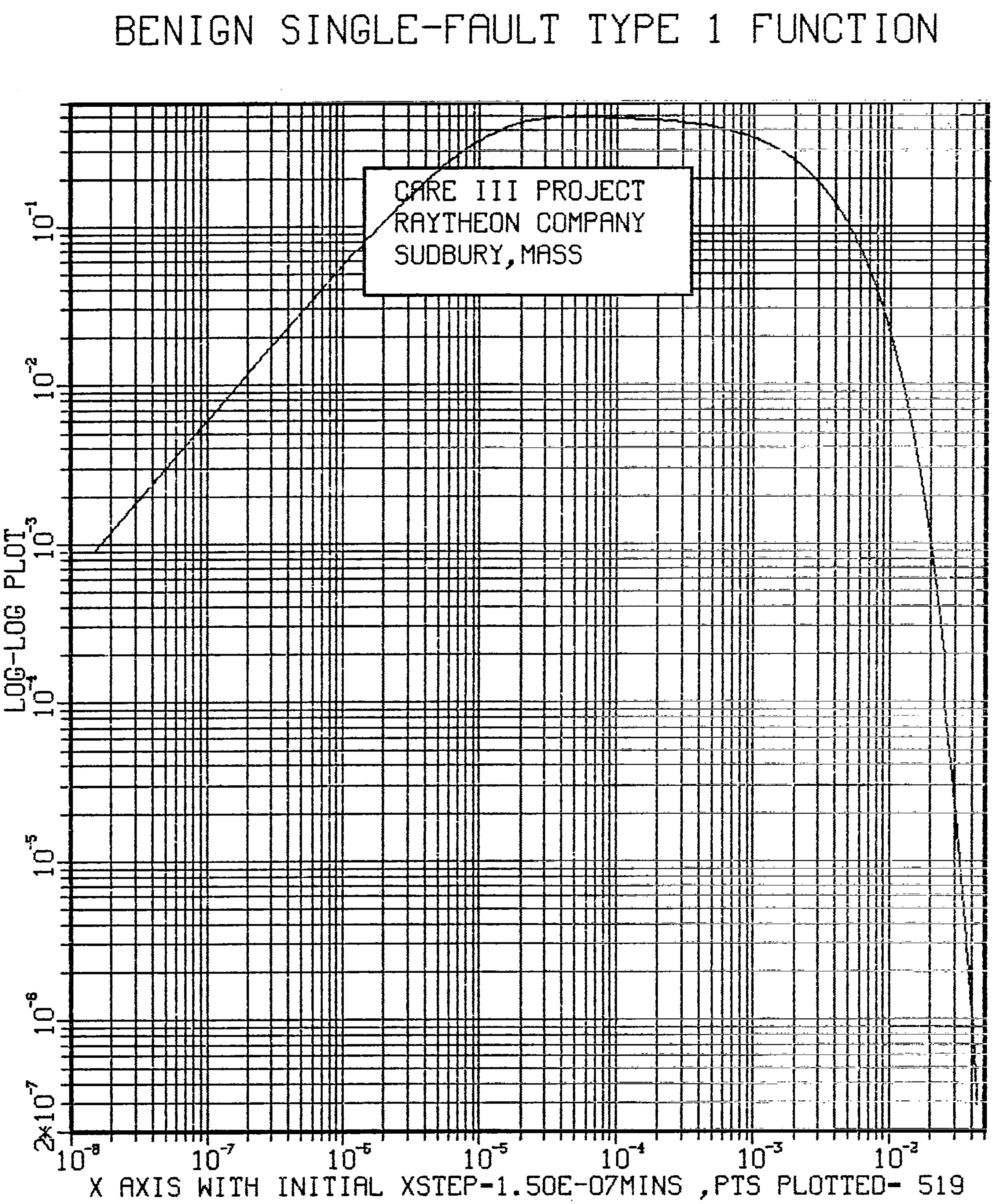

FIGURE 3a''
TEST CASE 3B '

(Function PB vs. Time-Mins., Markov Model)

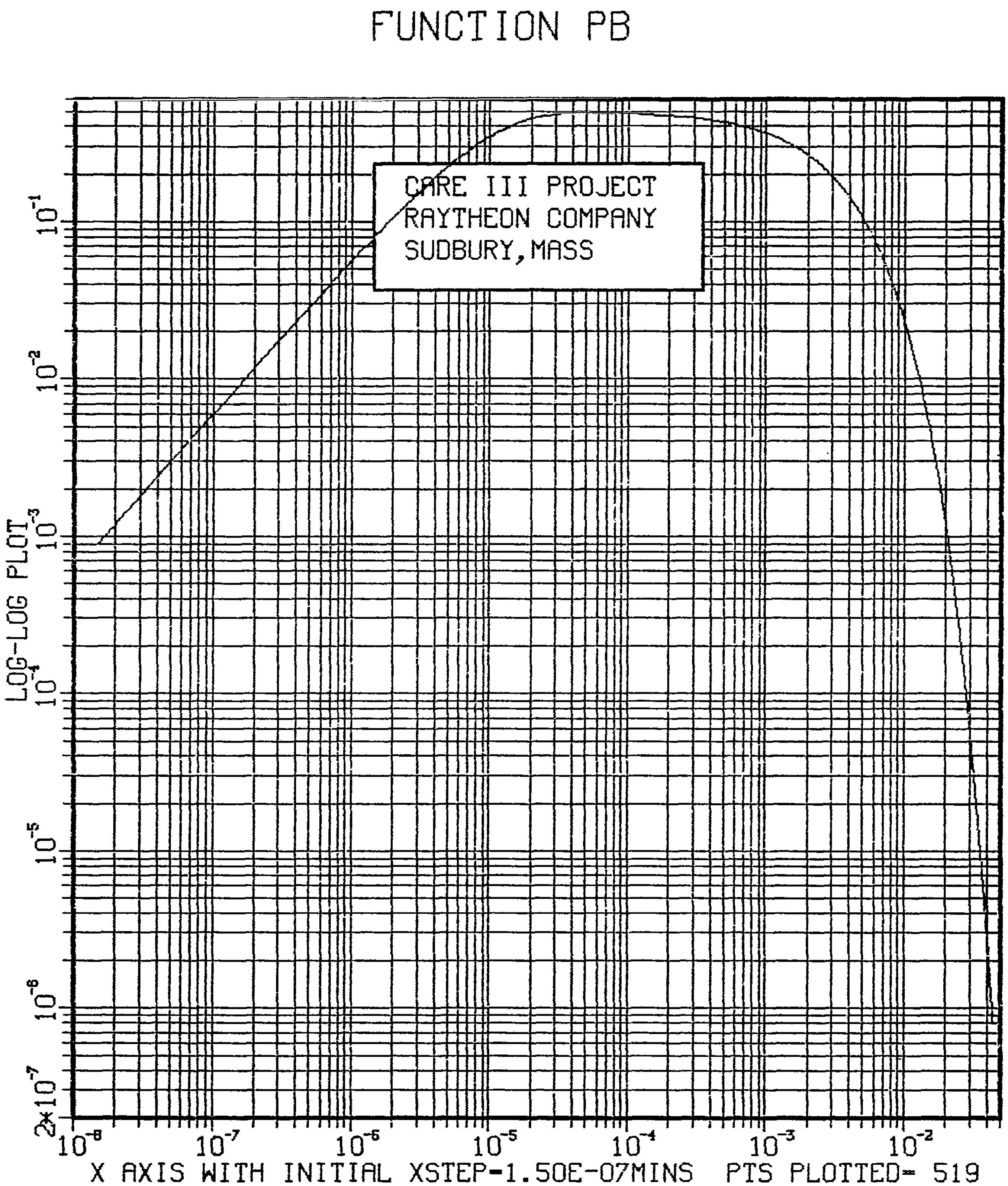

FIGURE 3a'''
TEST CASE 3B '

(Function PNB vs. Time-Mins., CARE III Model)

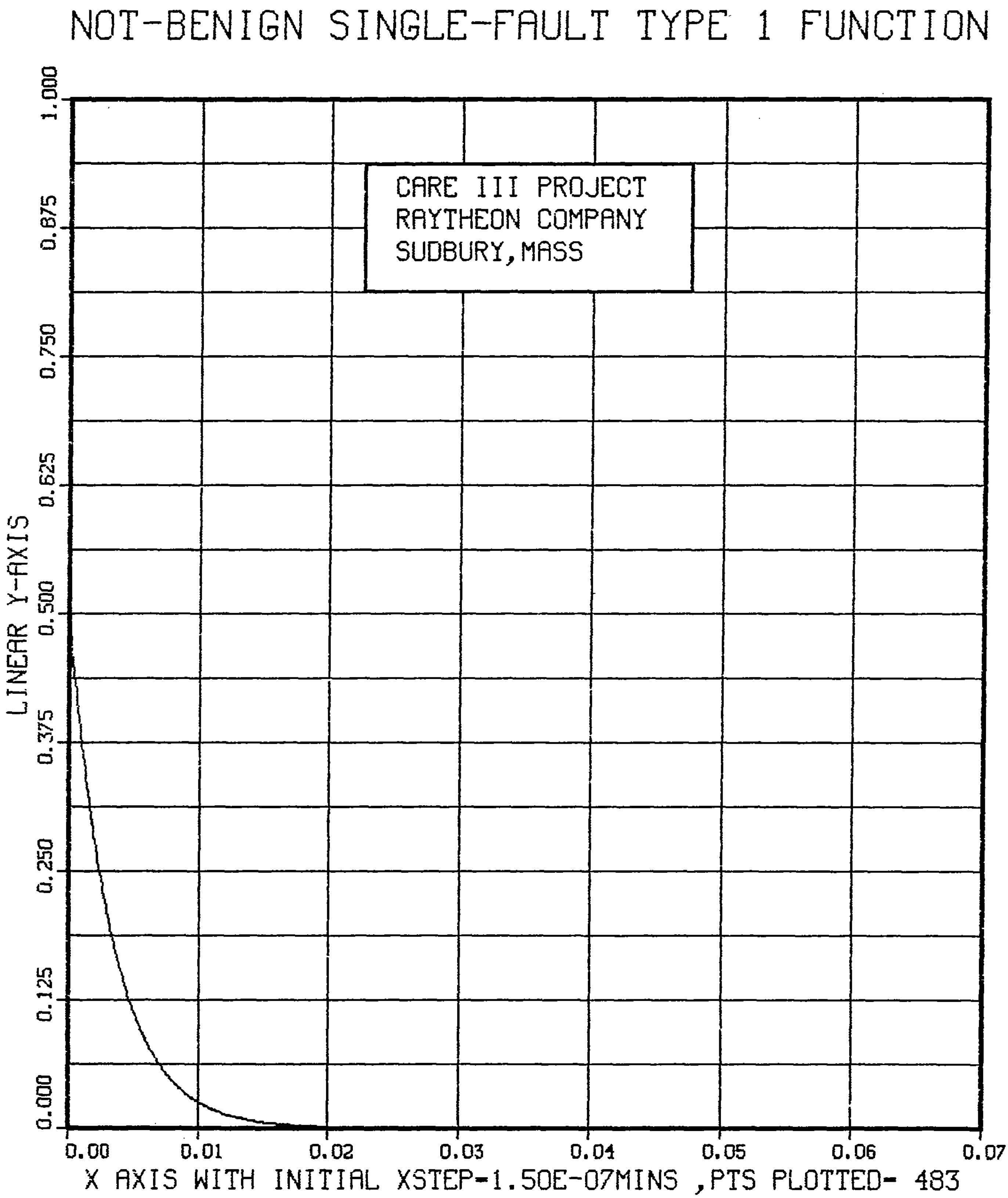

FIGURE 3b
TEST CASE 3B

(Function PNB vs. Time-Mins., CARE III Model)

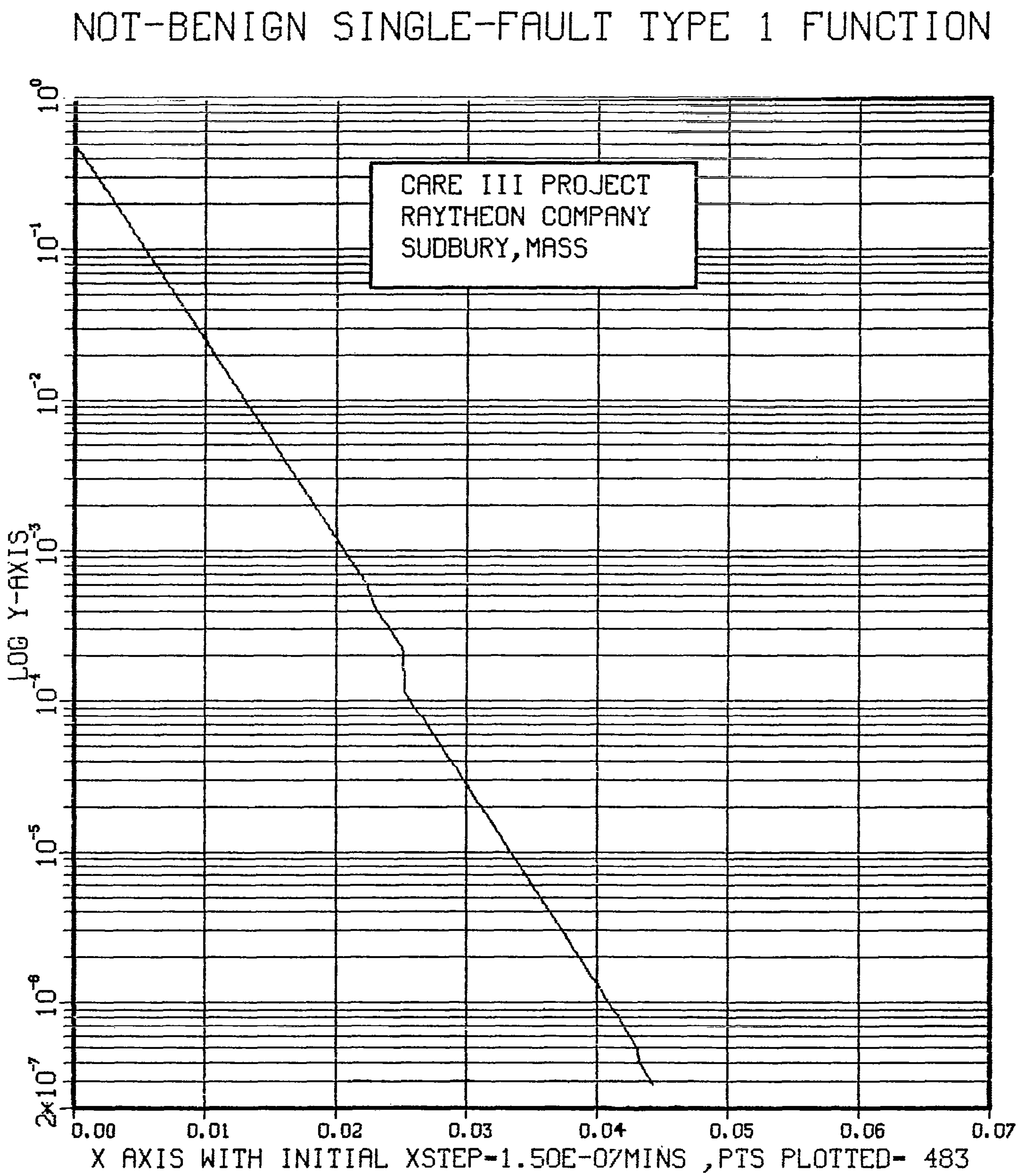

FIGURE 3b'
TEST CASE 3B '

(Function PNB vs. Time-Mins., CARE III Model)

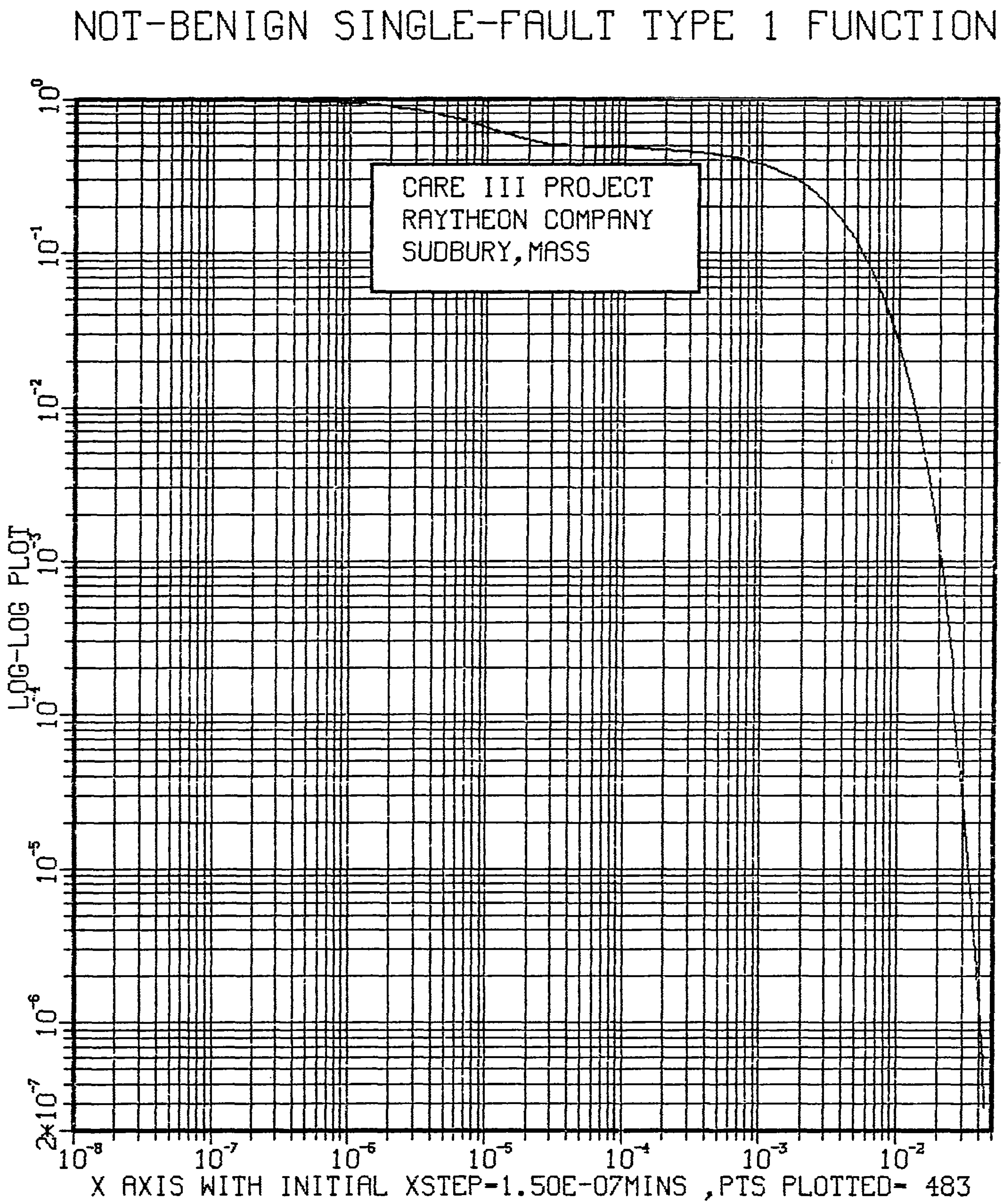

FIGURE 3b''
TEST CASE 3B '

(Function PNB vs. Time-Mins., Markov Model)

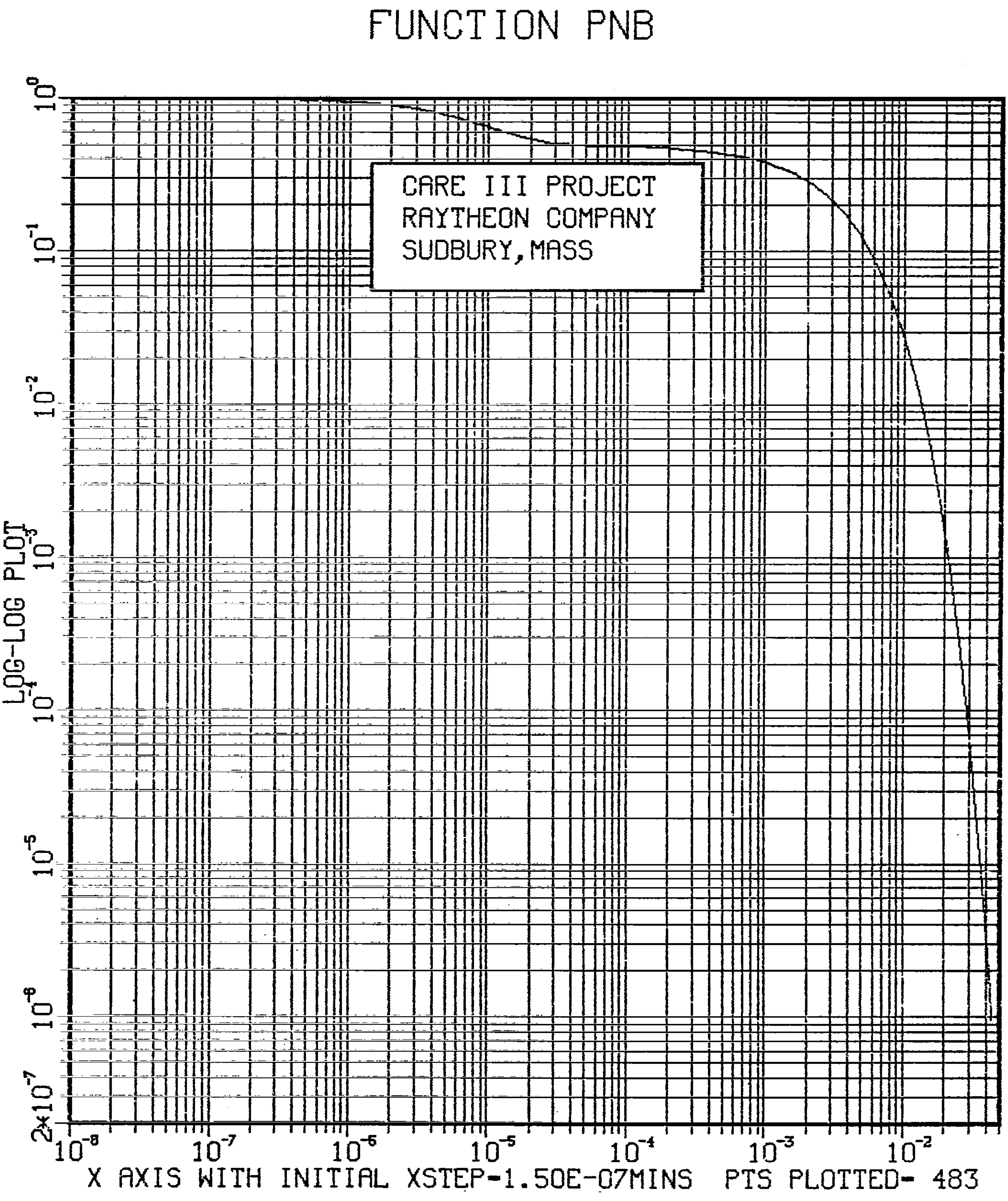

FIGURE 3b'''
TEST CASE 3B

(Function PL vs. Time-Mins., CARE III Model)

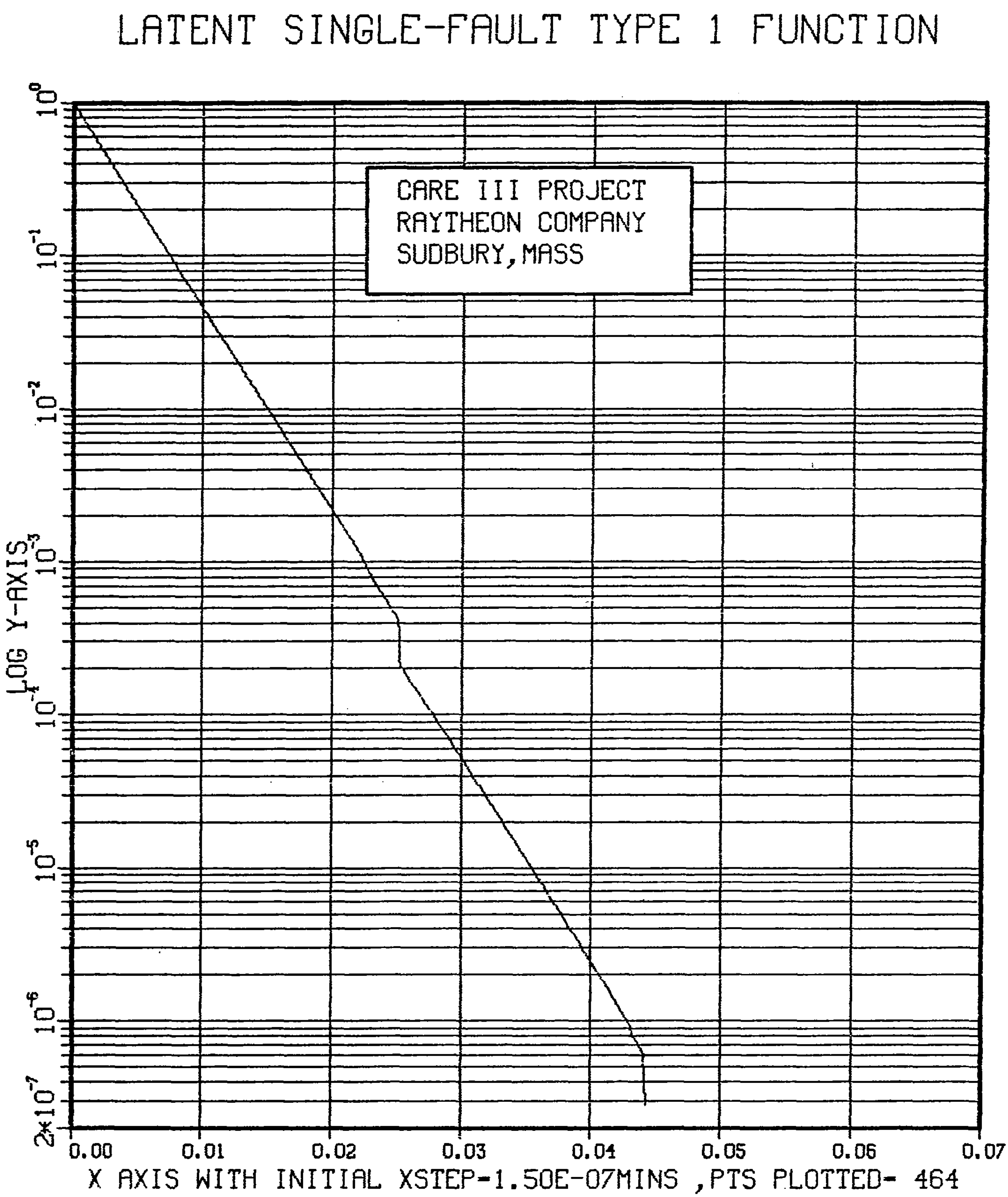

FIGURE 3c'
TEST CASE 3B '

(Function PL vs. Time-Mins., CARE III Model)

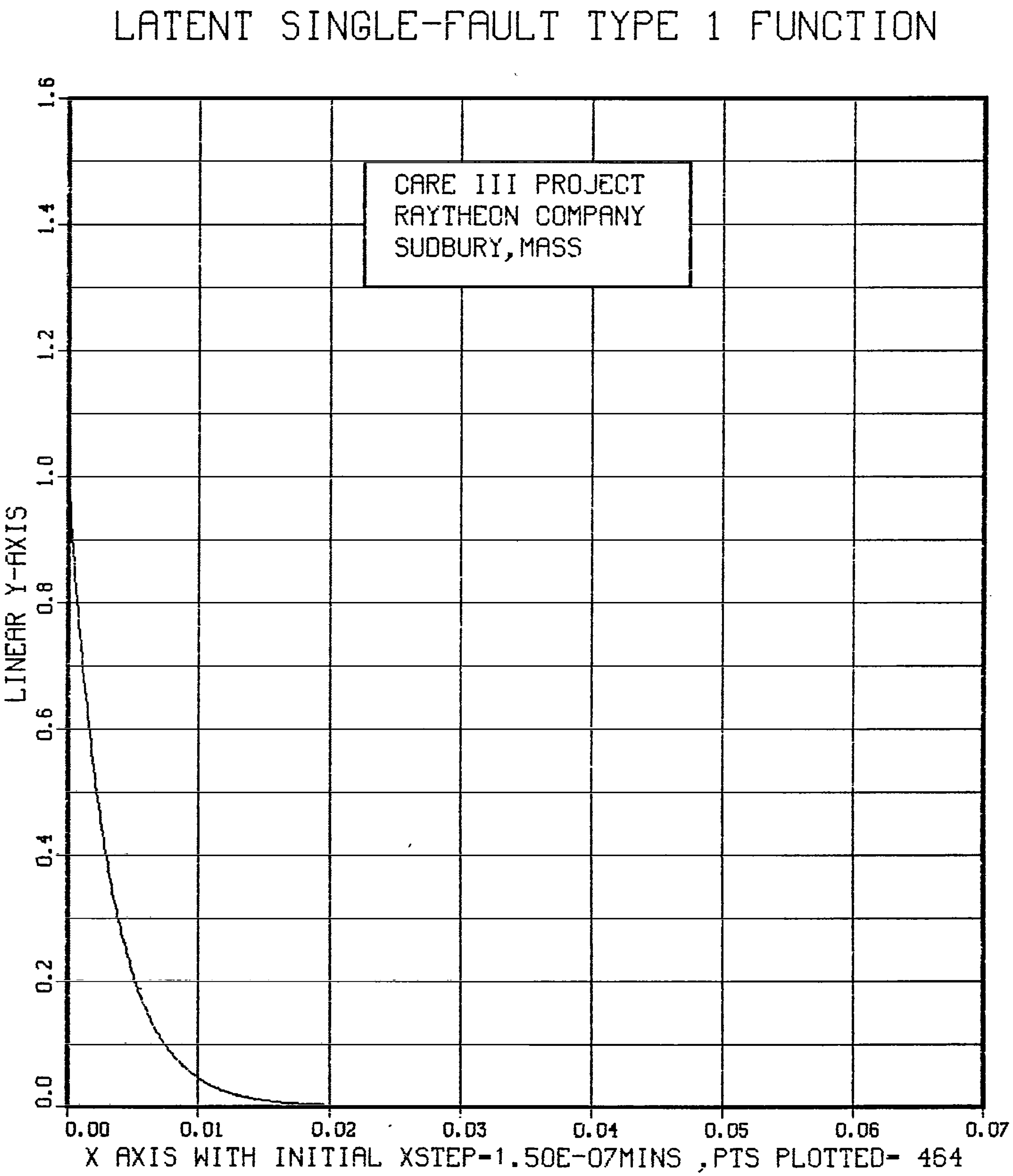

FIGURE 3c
TEST CASE 3B

(Function PL vs. Time-Mins., CARE III Model)

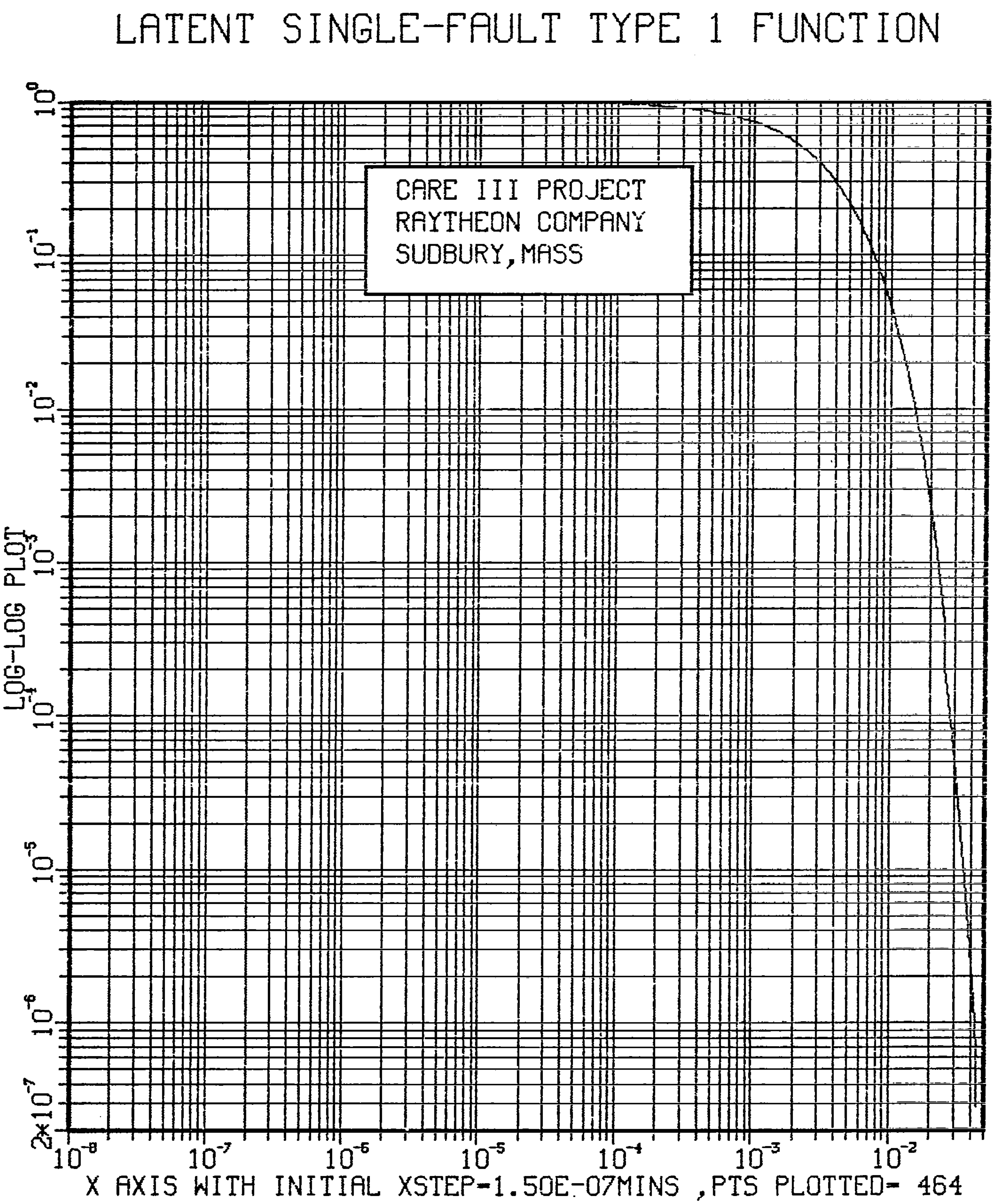

FIGURE 3c''
TEST CASE 3B '

(Function PL vs. Time-Mins., Markov Model)

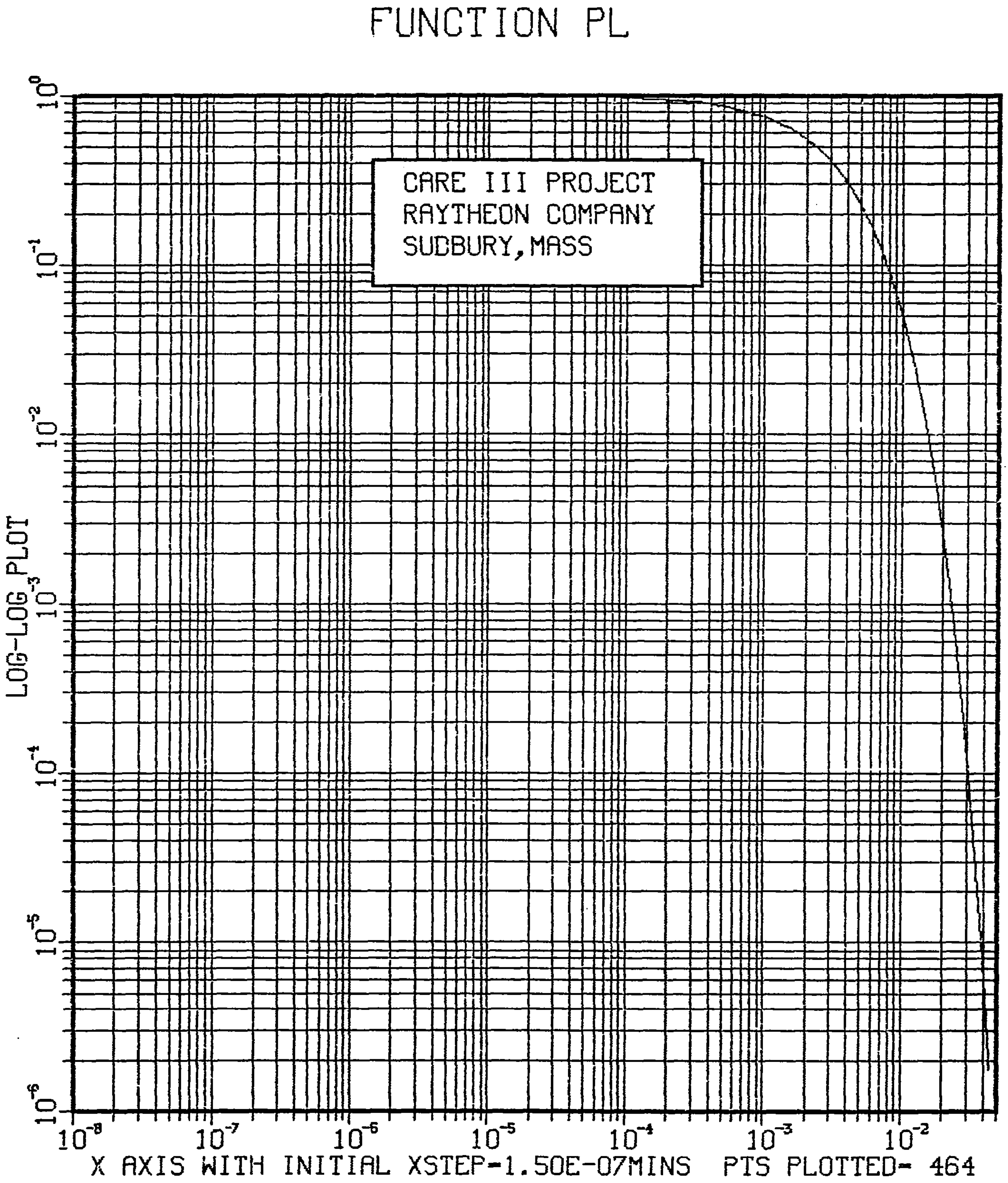

FIGURE 3c'''
TEST CASE 3B '

(Function PDF vs. Time-Mins., CARE III Model)

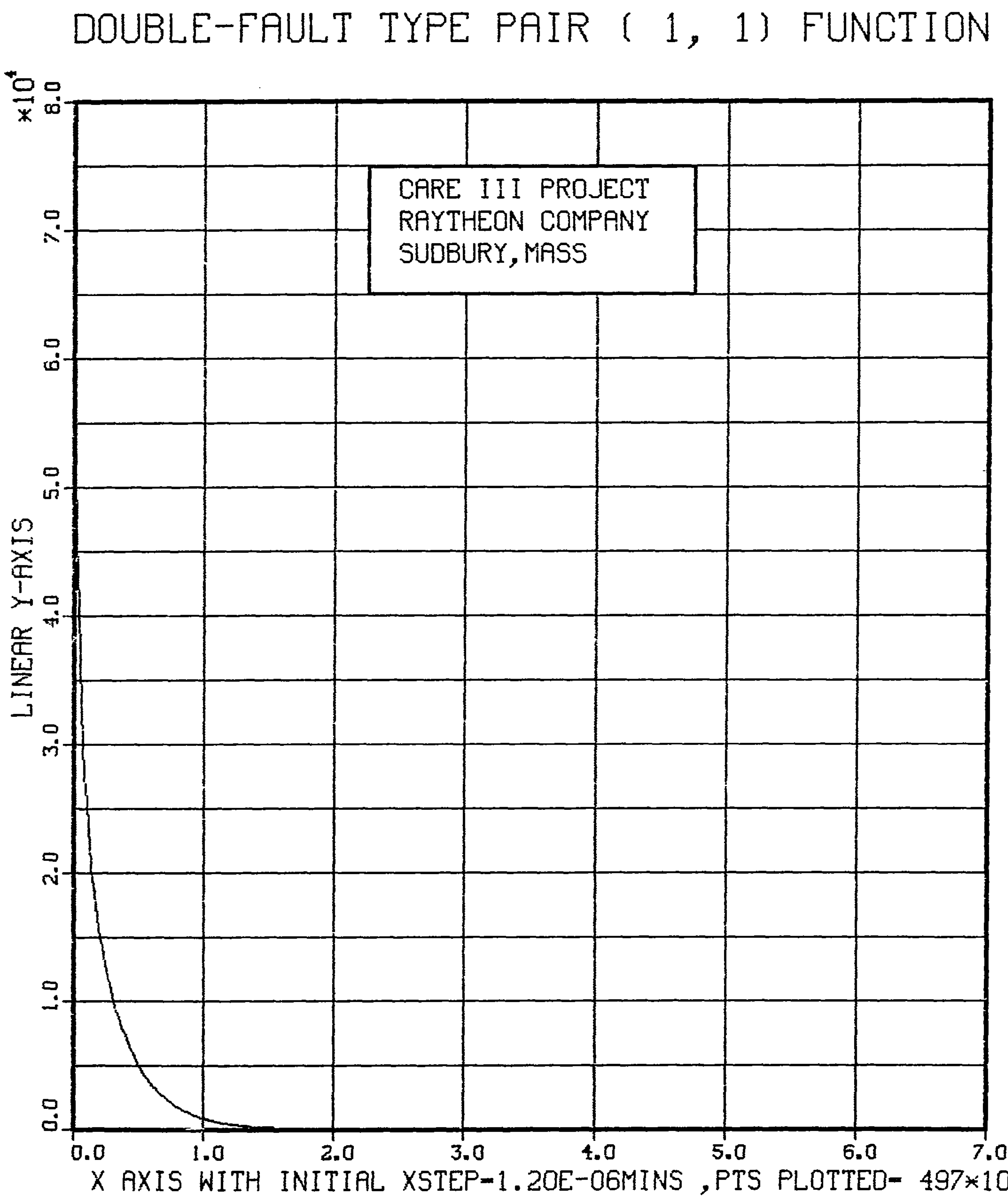

FIGURE 3d
TEST CASE 3B

(Function PDF vs. Time-Mins., CARE III Model)

DOUBLE-FAULT TYPE PAIR (1, 1) FUNCTION

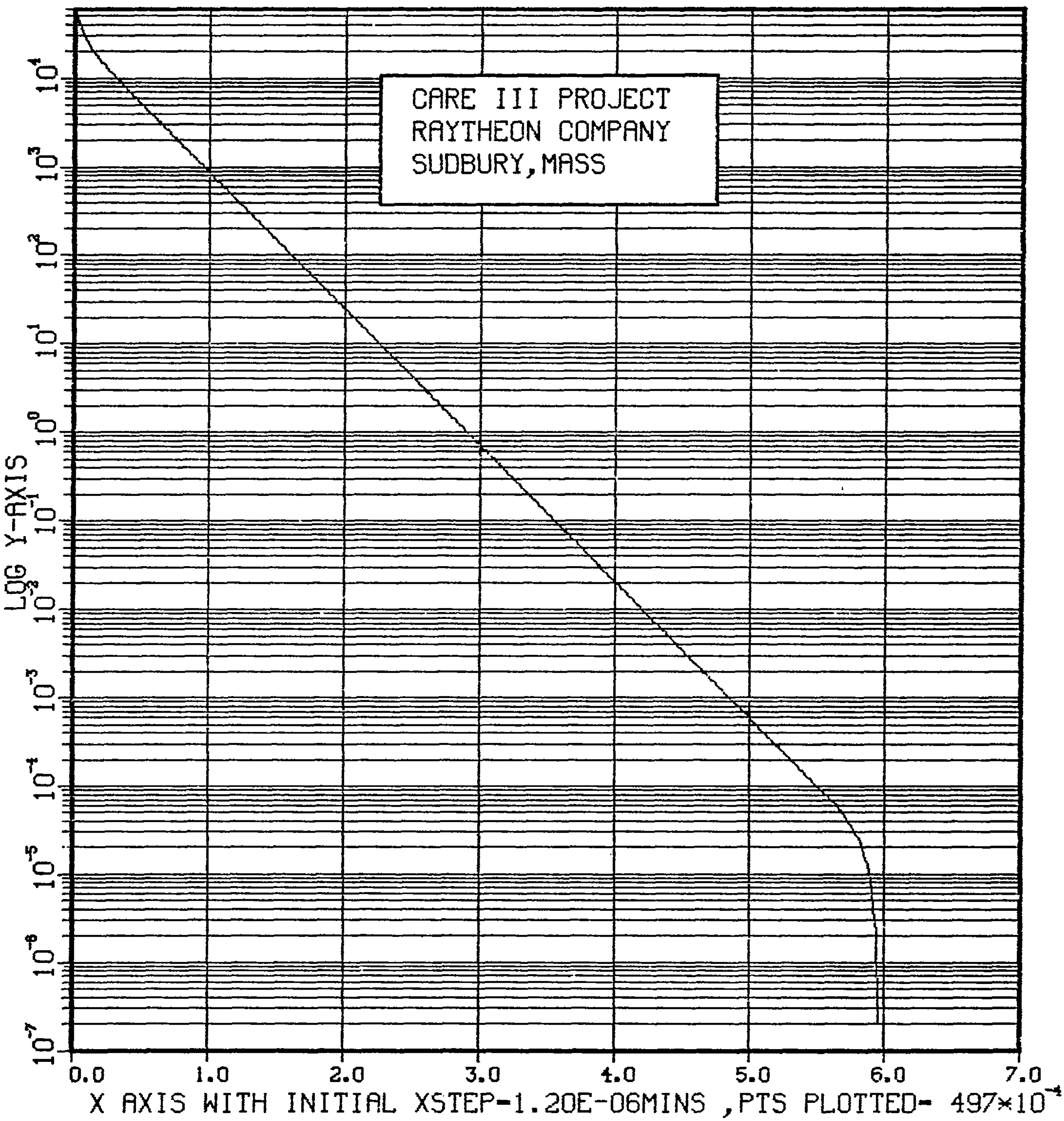

FIGURE 3d'
TEST CASE 3B '

(Function PDF vs. Time-Mins., CARE III Model)

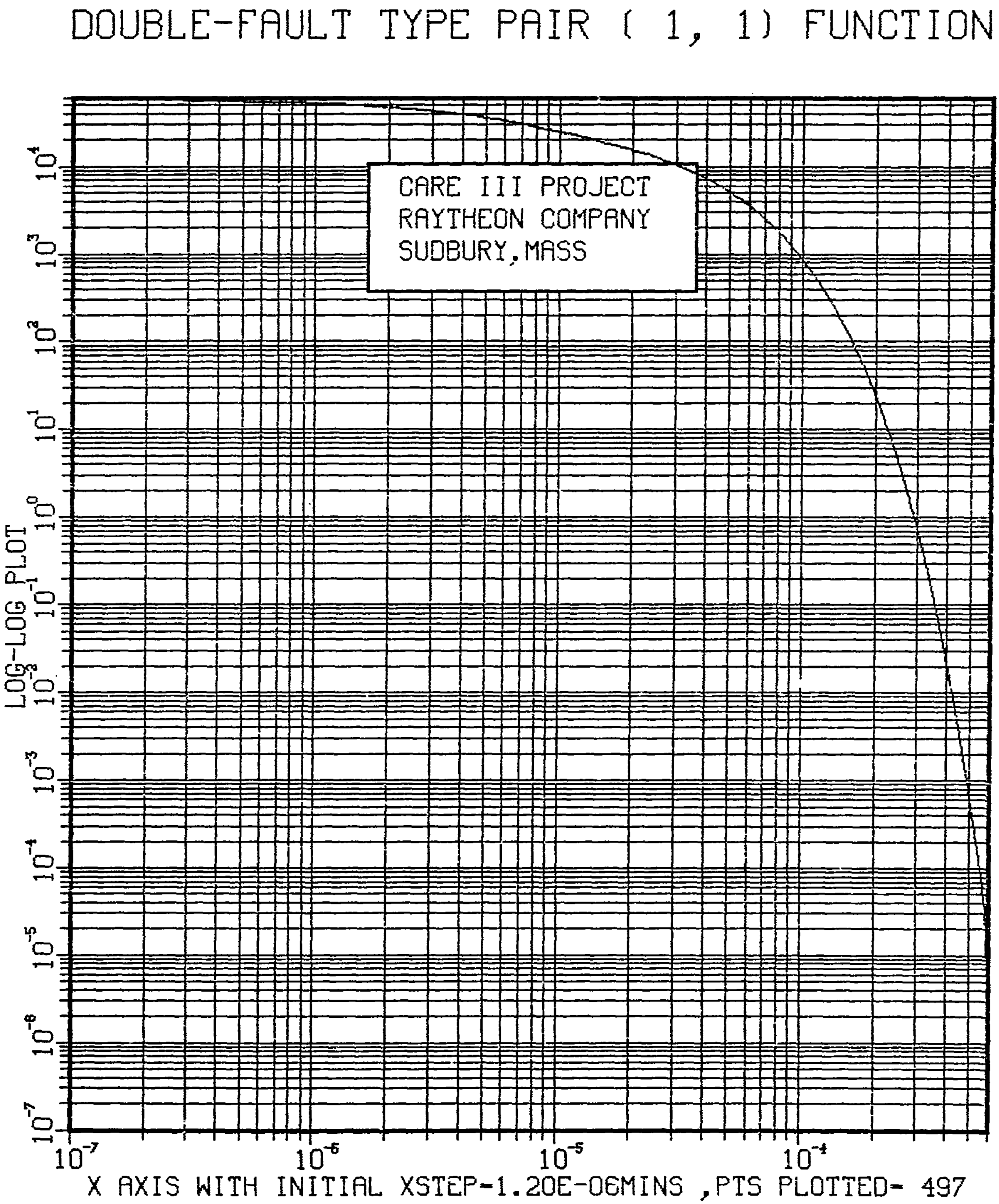

FIGURE 3d''
TEST CASE 3B '

(Function P* SUM vs. Time-Mins., CARE III Model)

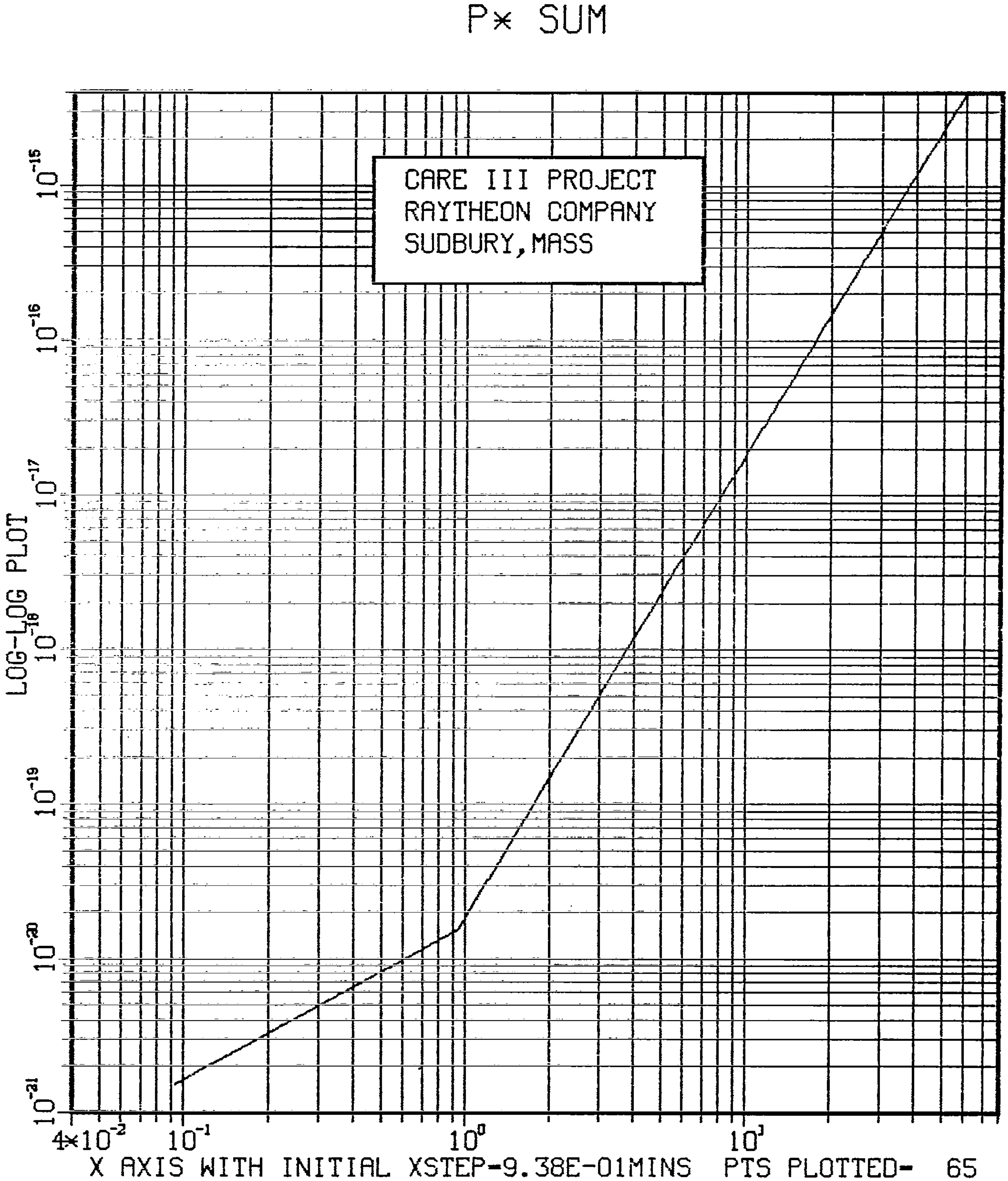

FIGURE 3e
TEST CASE 3B'

(Function Q SUM vs. Time-Mins., CARE III Model)

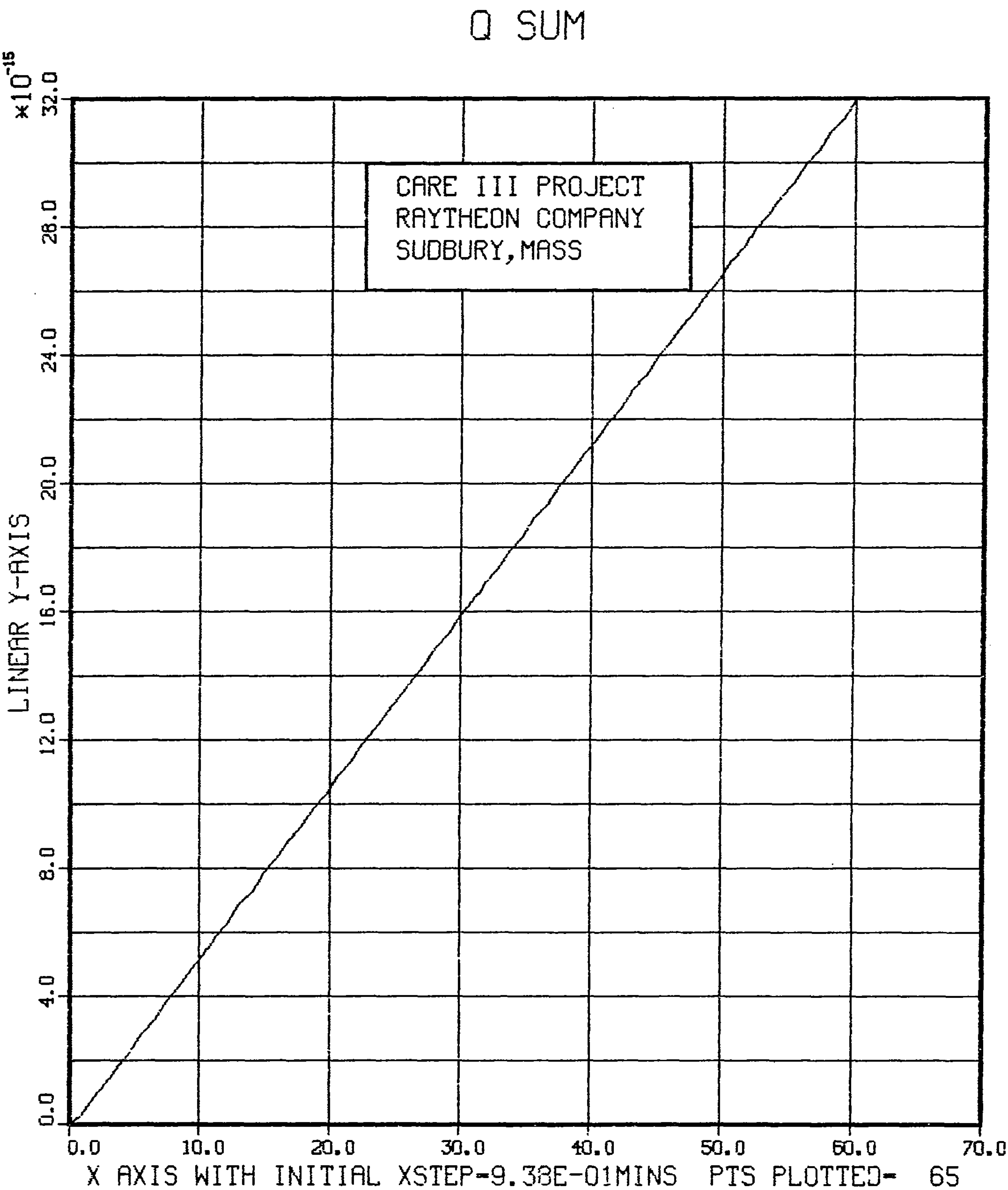

FIGURE 3f
TEST CASE 3B'

(Function Q SUM vs. Time-Mins., CARE III Model)

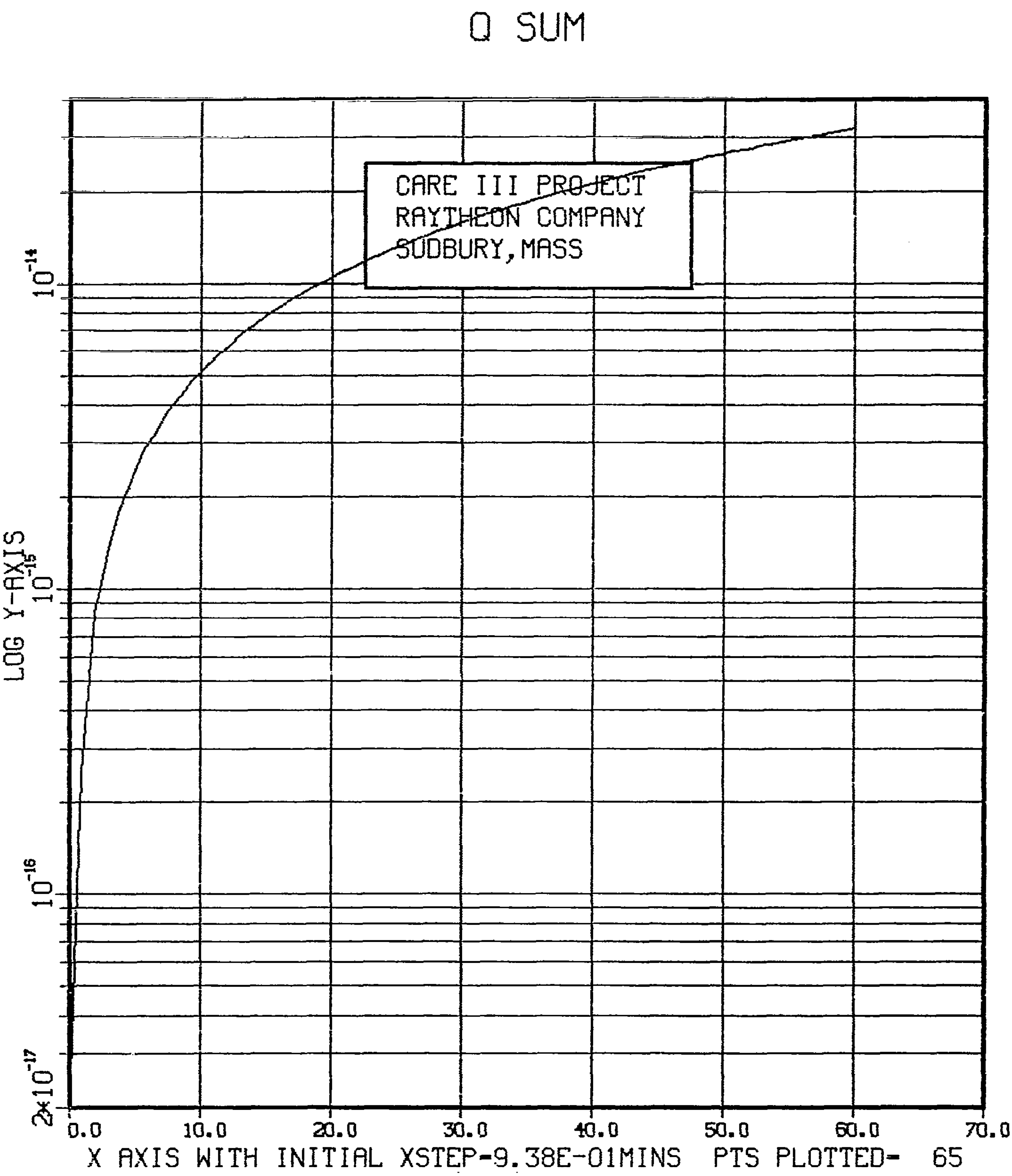

FIGURE 3f'
TEST CASE 3B'

(Function Q SUM vs. Time-Mins., CARE III Model)

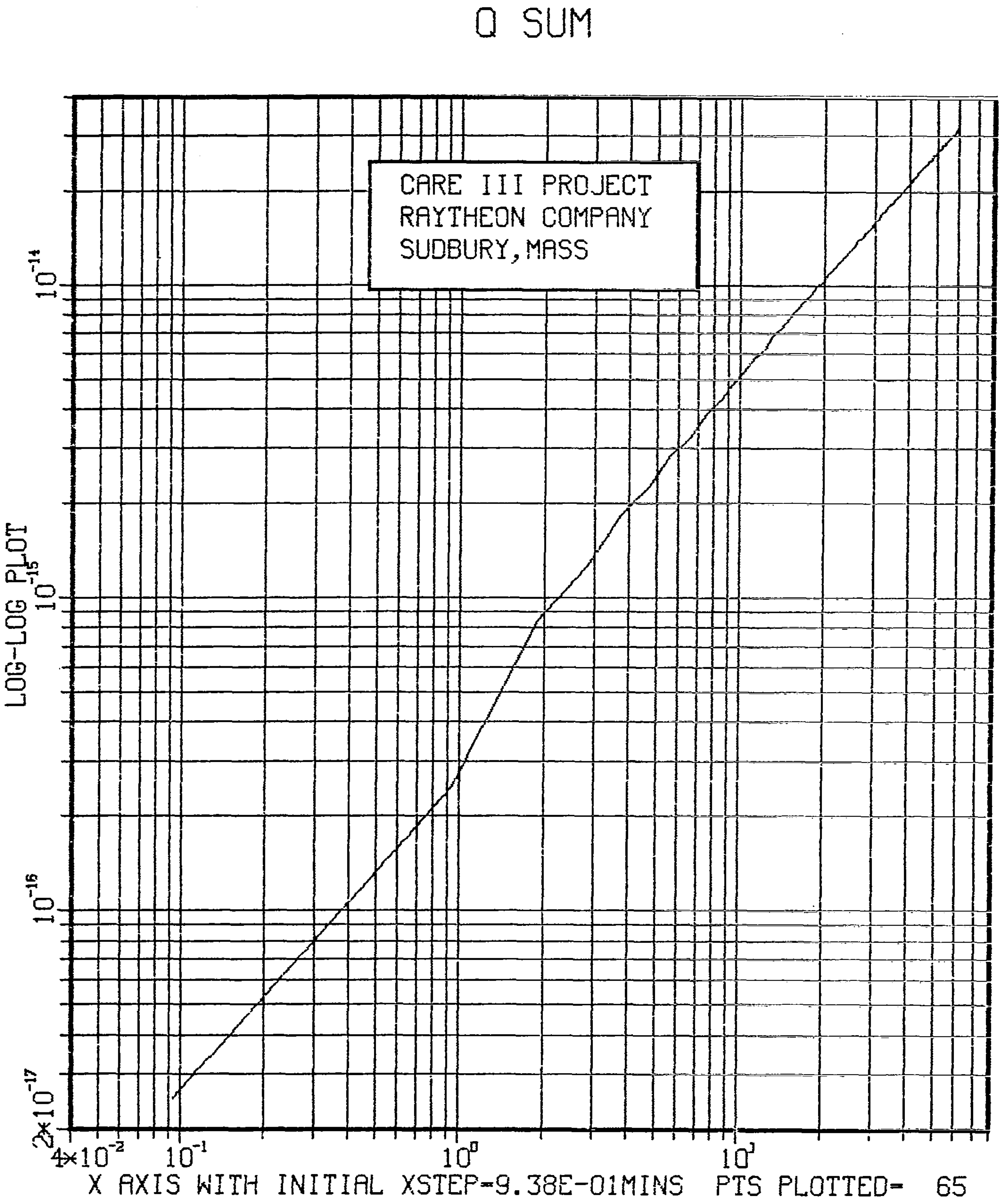

FIGURE 3f''
TEST CASE 3B'

(Function Q + P* SUM vs. Time-Mins., CARE III Model)

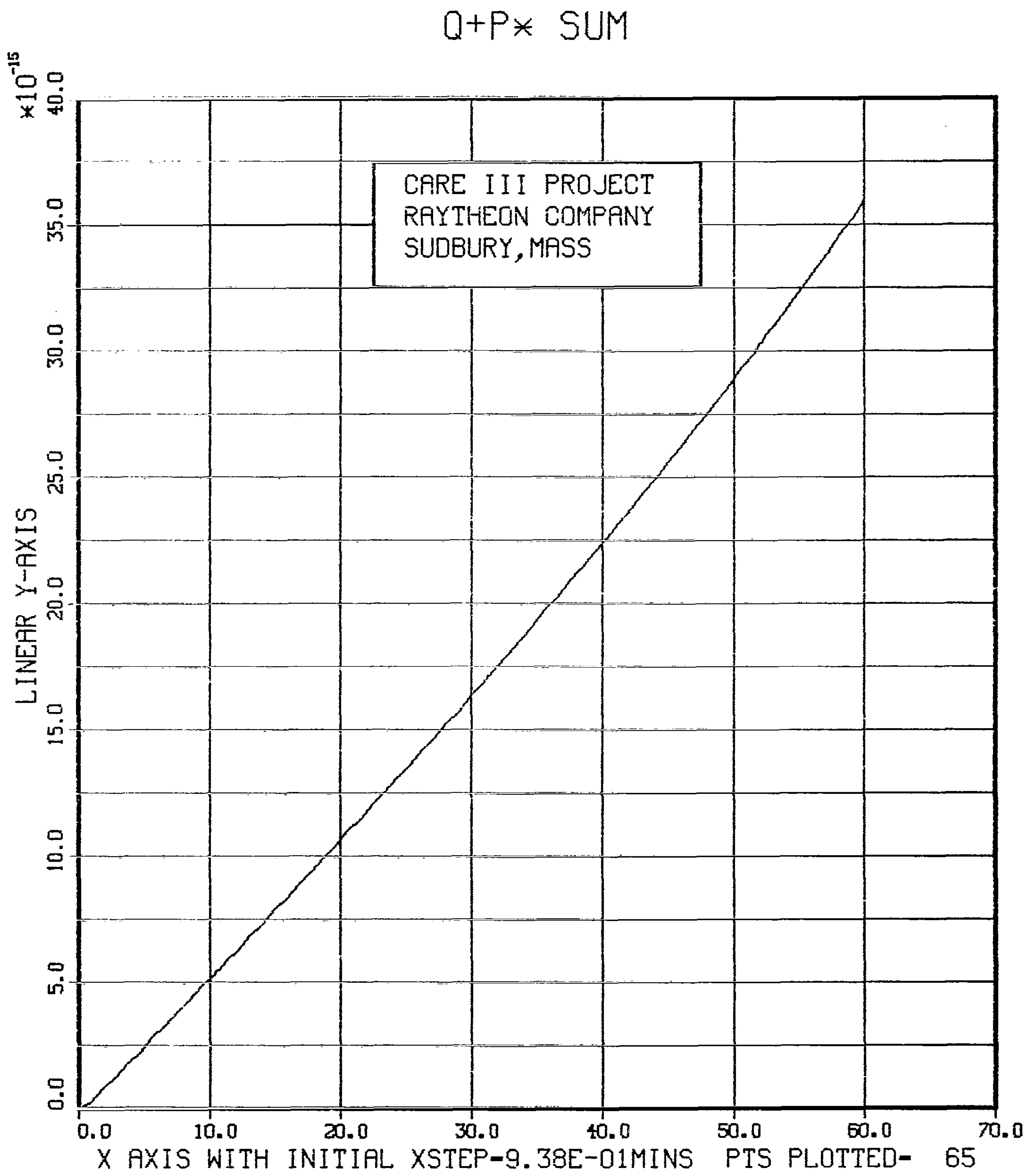

FIGURE 3g
TEST CASE 3B

(Function Q + P* SUM vs. Time-Mins., CARE III Model)

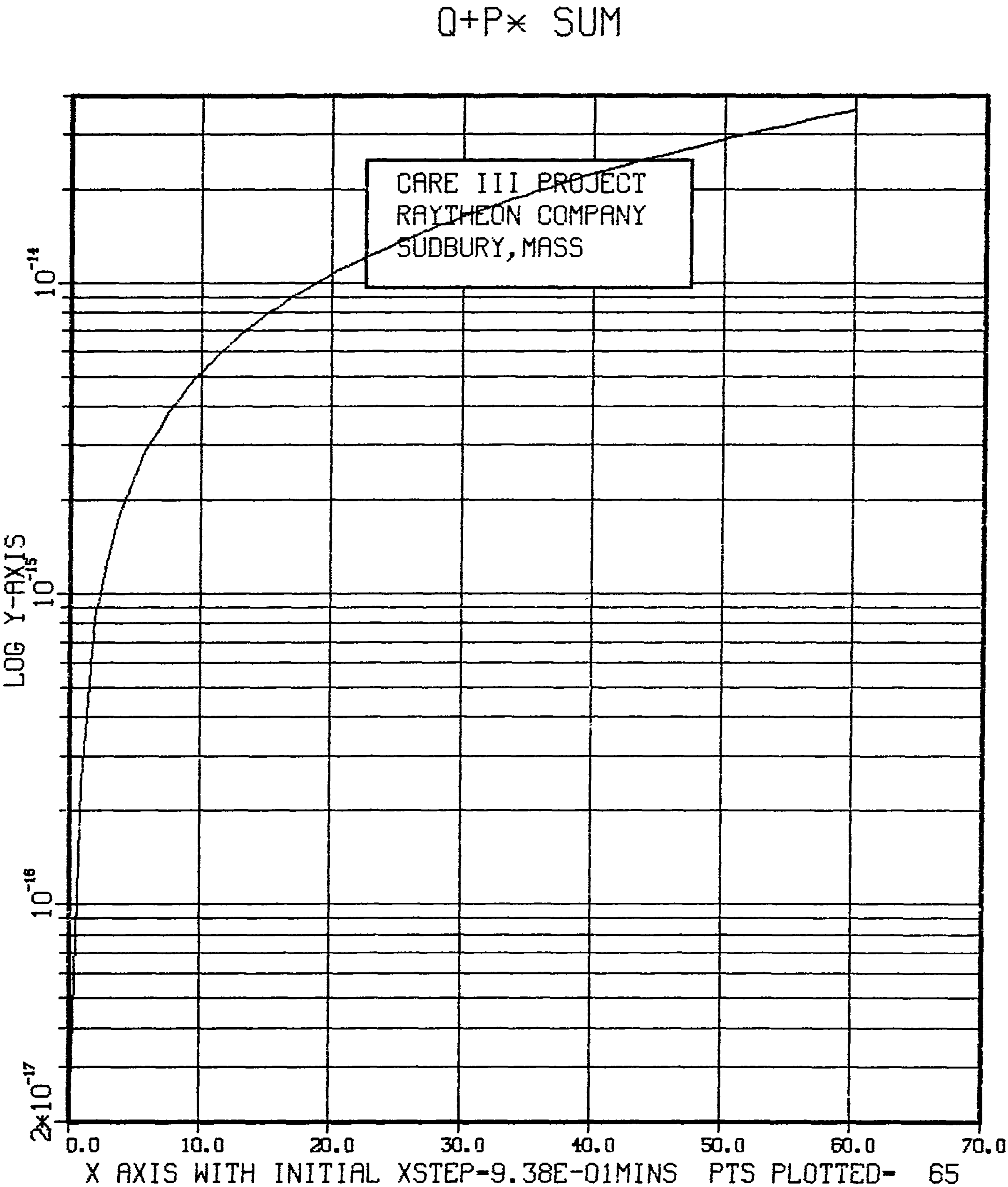

FIGURE 3g'
TEST CASE 3B'

(Function Q + P* SUM vs. Time-Mins., CARE III Model)

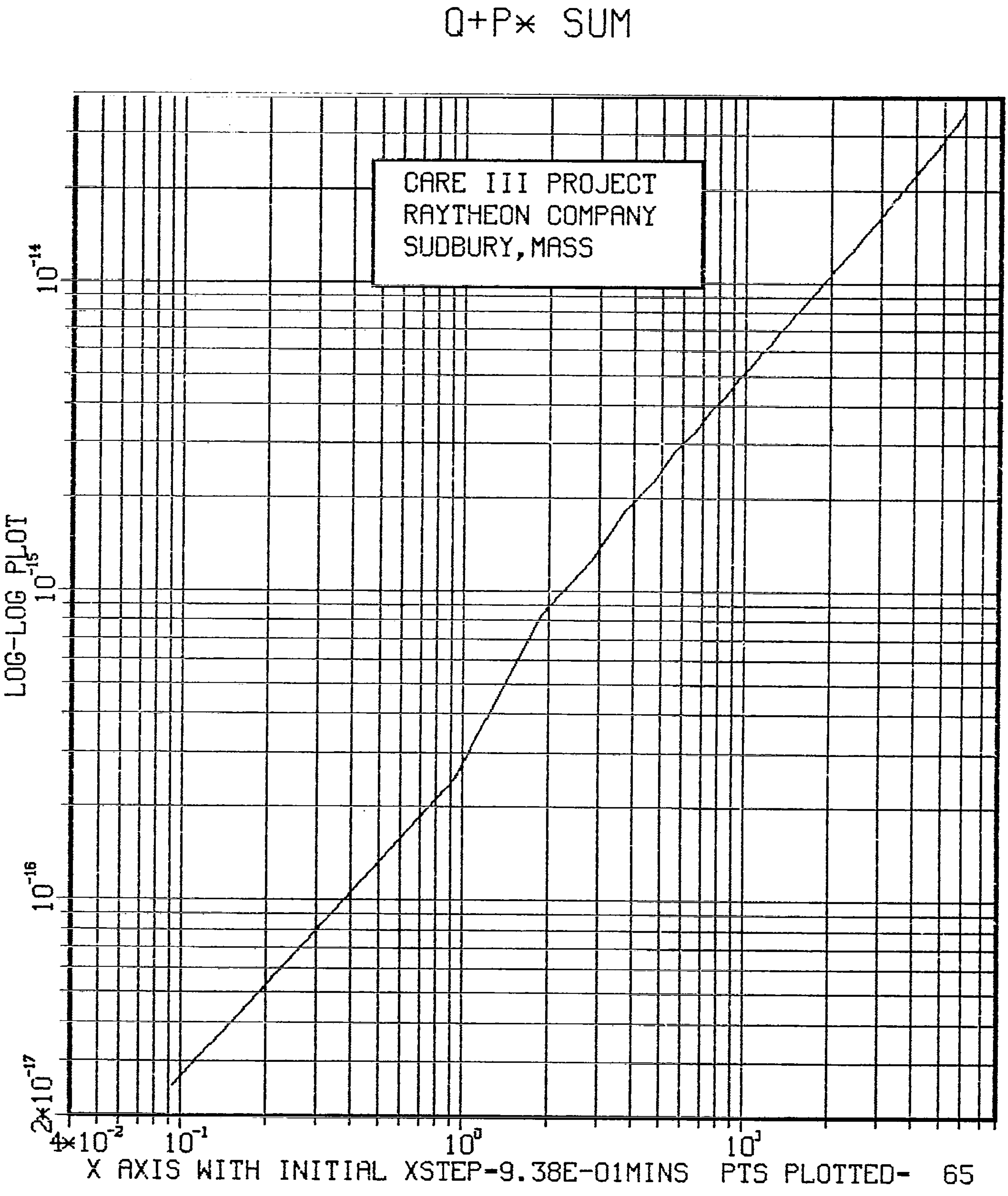

FIGURE 3g''
TEST CASE 3B'

(Function PB vs. Time-Mins., CARE III Model)

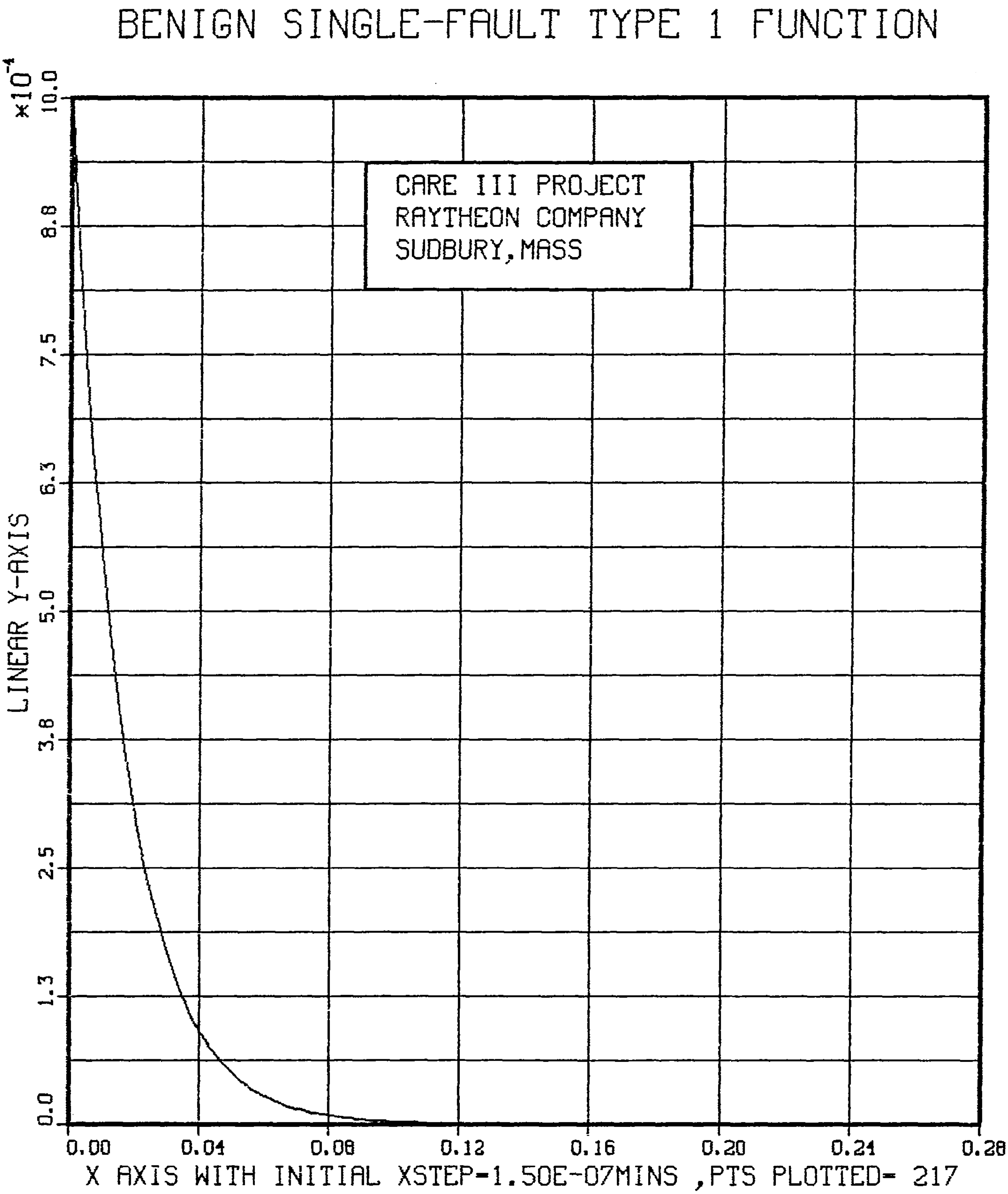

FIGURE 4a
TEST CASE 3C
TRUNC = 10^{-4}

(Function PB vs. Time-Mins., Markov Model)

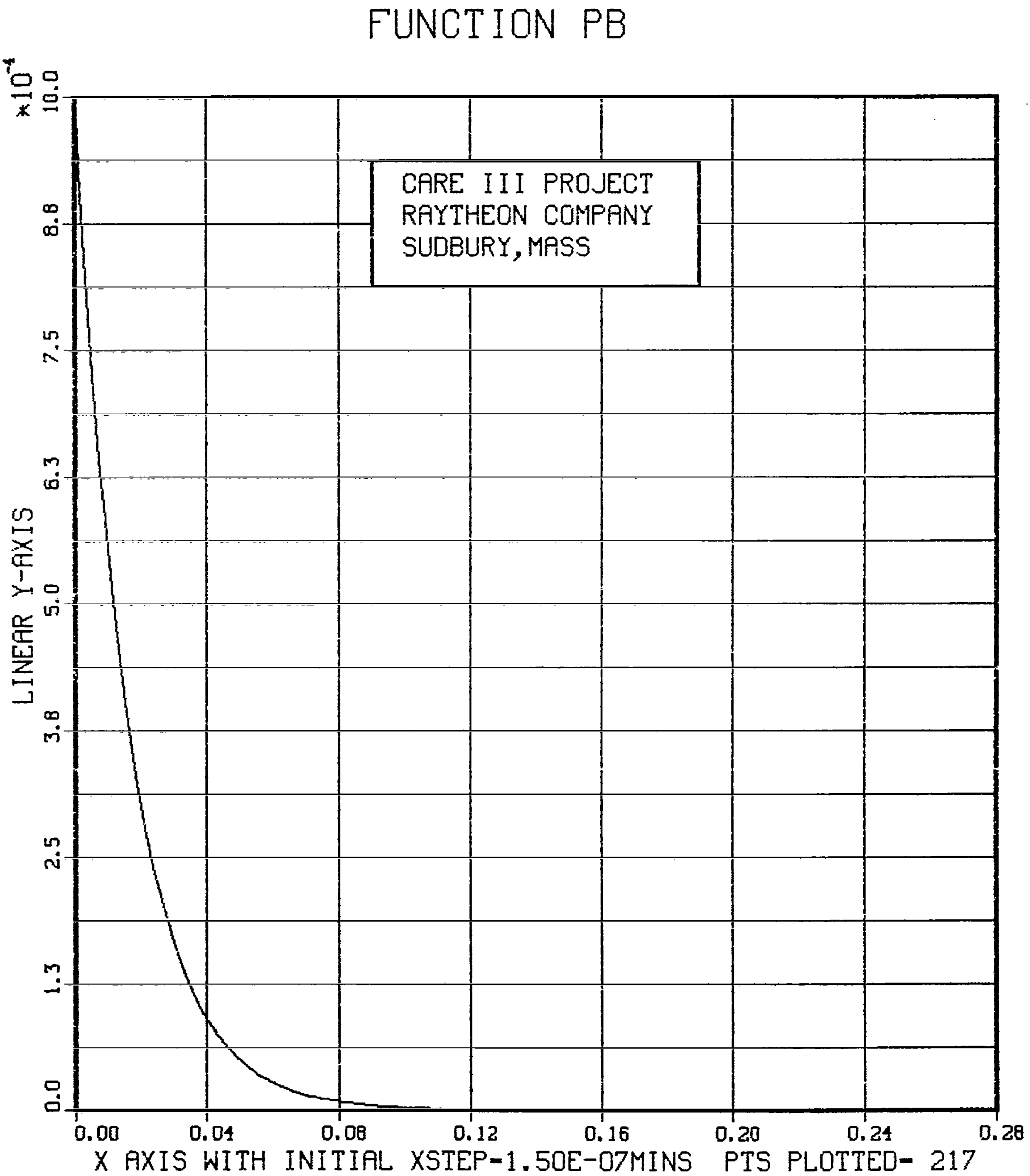

FIGURE 4a'
TEST CASE 3C

(Function PB vs. Time-Mins., CARE III Model)

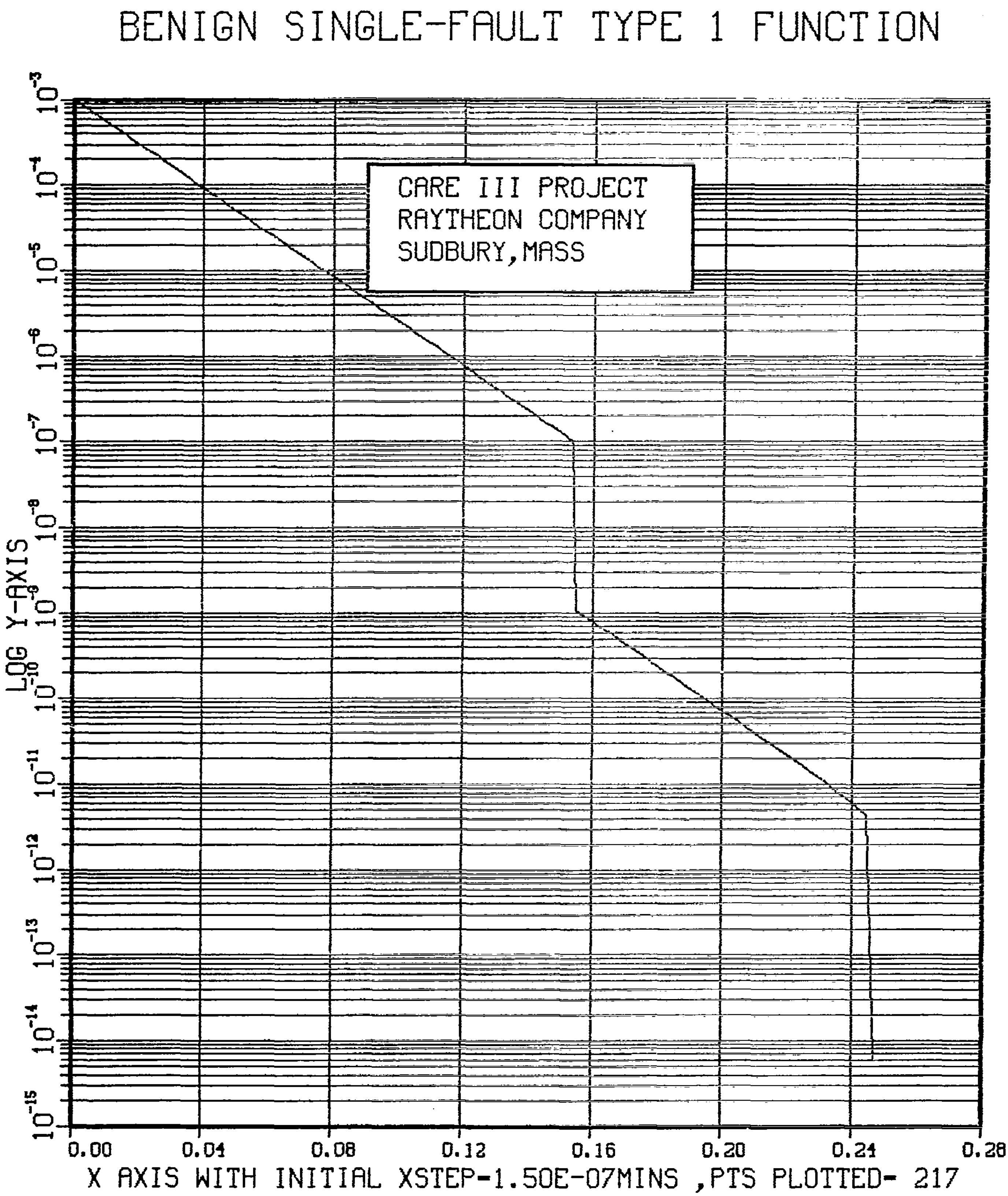

FIGURE 4b
TEST CASE 3C
TRUNC = 10^{-4}

(Function PB vs. Time-Mins., Markov Model)

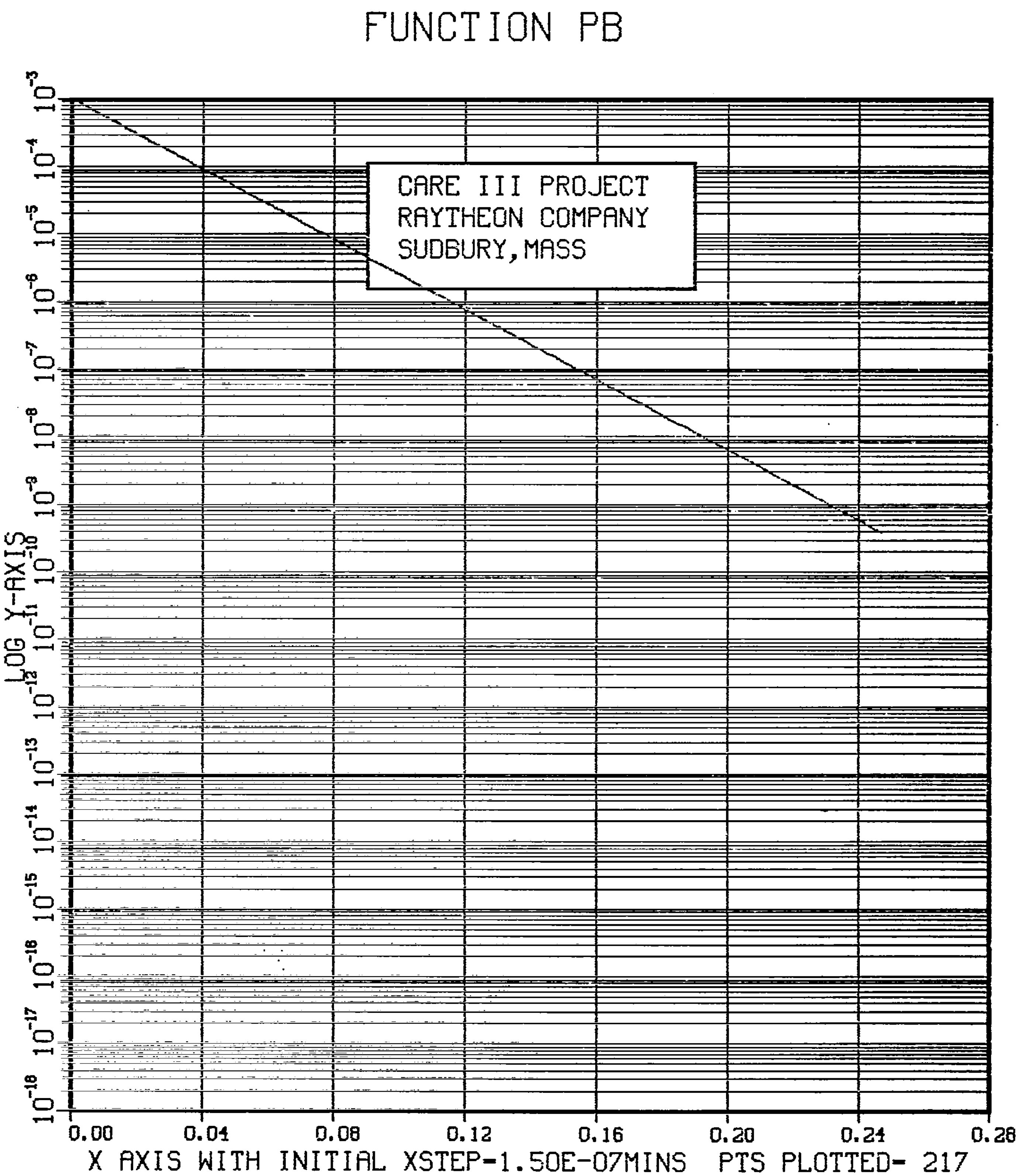

FIGURE 4b'
TEST CASE 3C

(Function PNB vs. Time-Mins., CARE III Model)

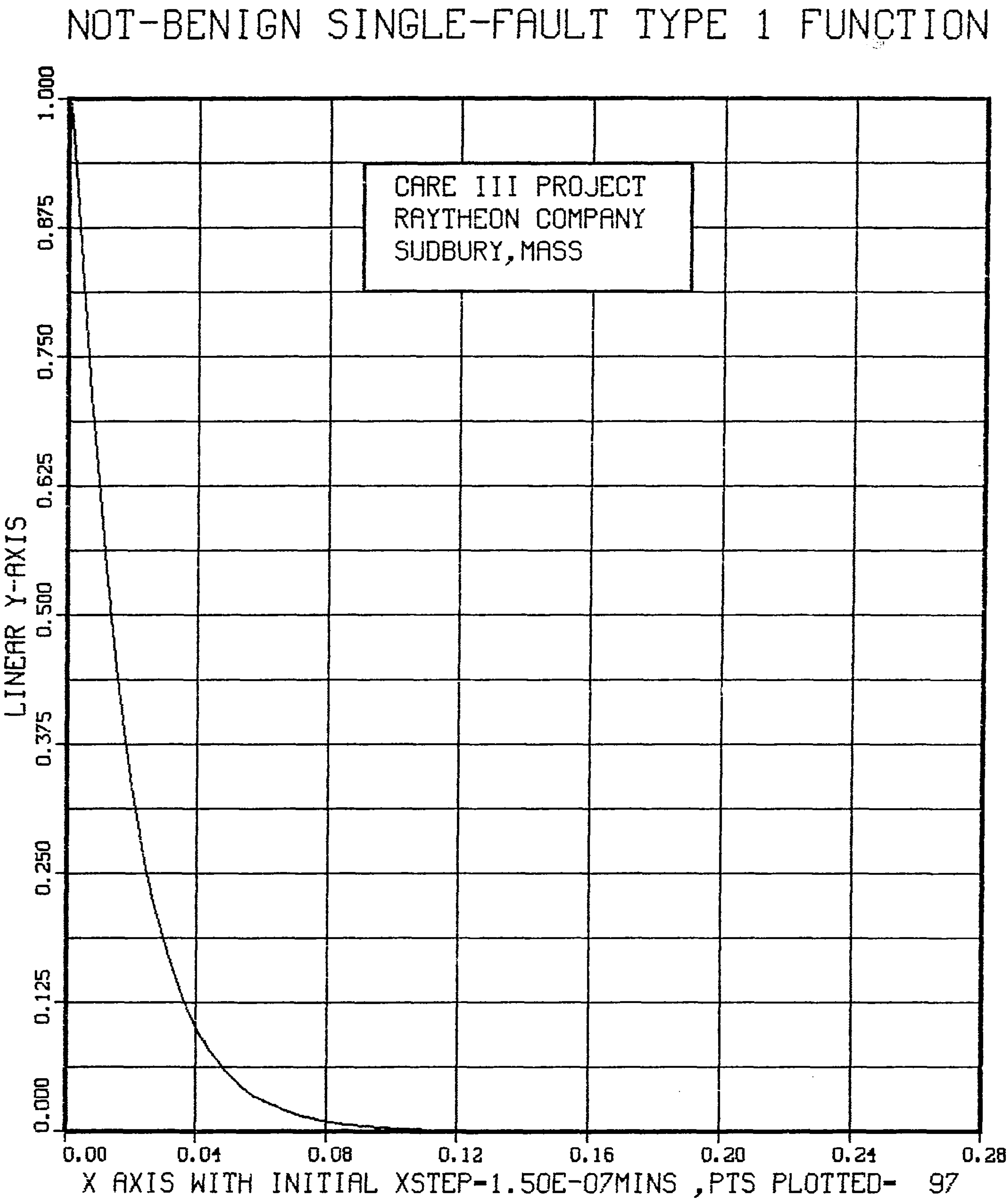

FIGURE 4c
TEST CASE 3C
TRUNC = 10^{-4}

(Function PNB vs. Time-Mins., Markov Model)

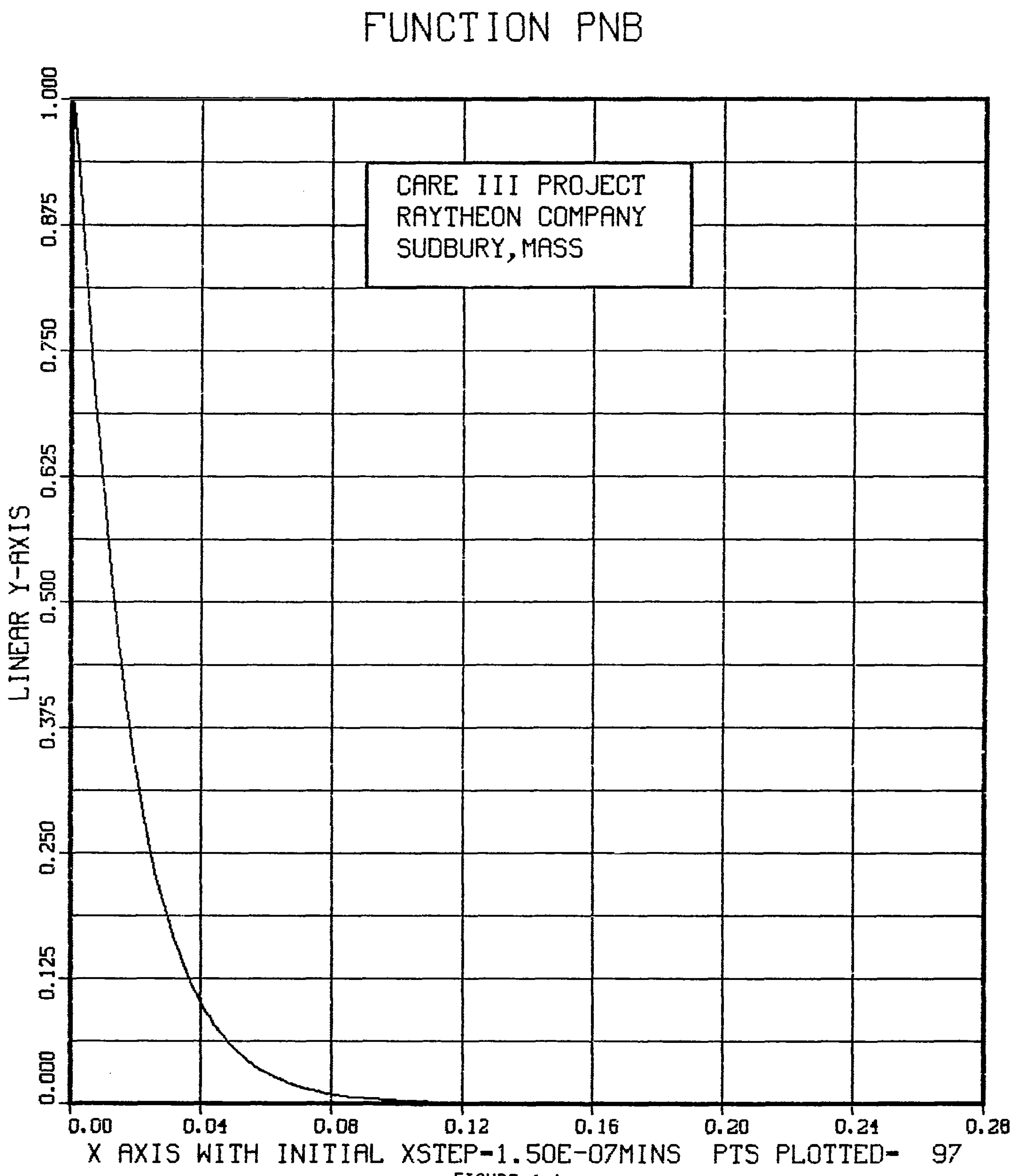

FIGURE 4c'
TEST CASE 3C

(Function PNB vs. Time-Mins., CARE III Model)

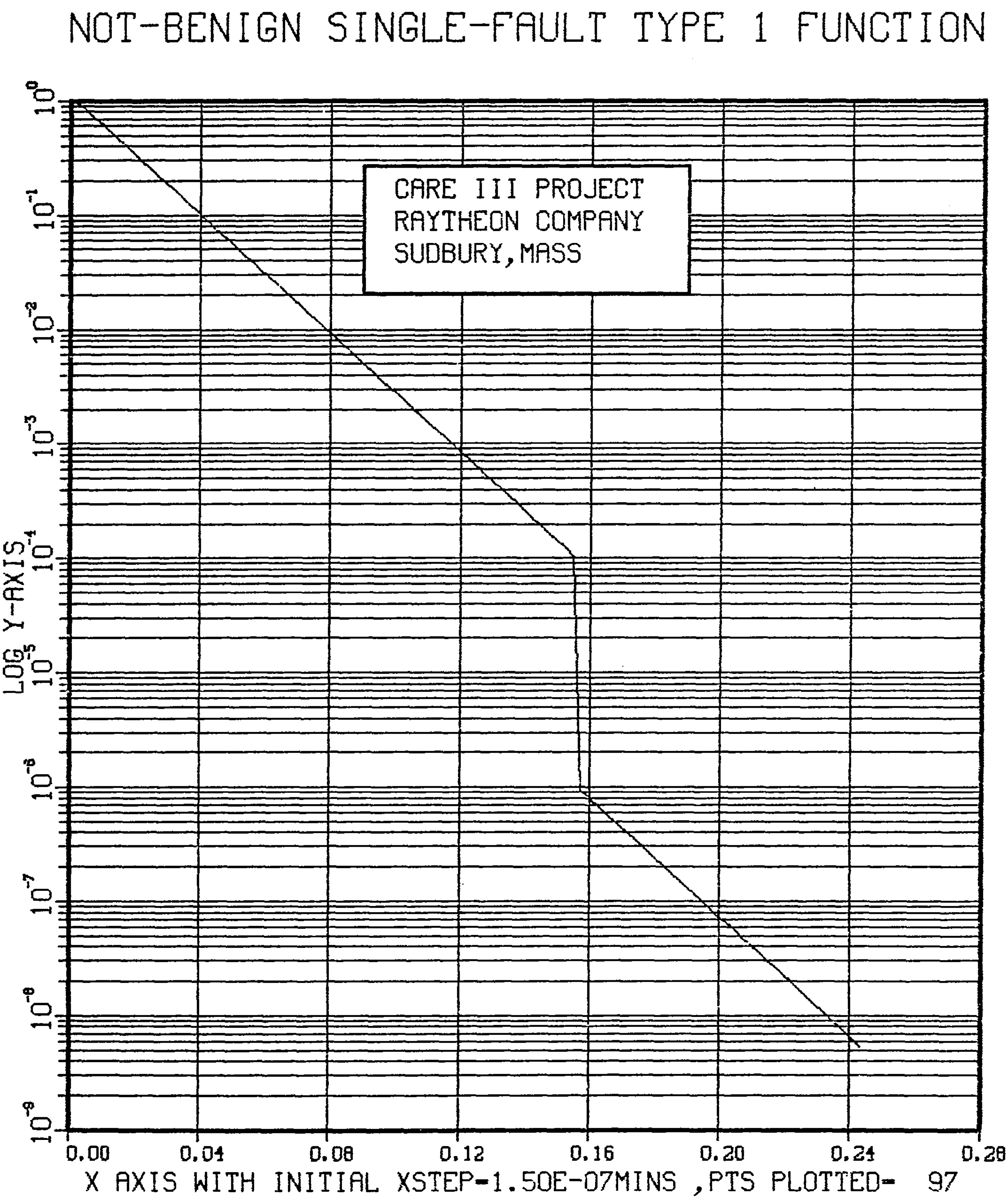

FIGURE 4d
TEST CASE 3C
TRUNC = 10^{-4}

(Function PNB vs. Time-Mins., Markov Model)

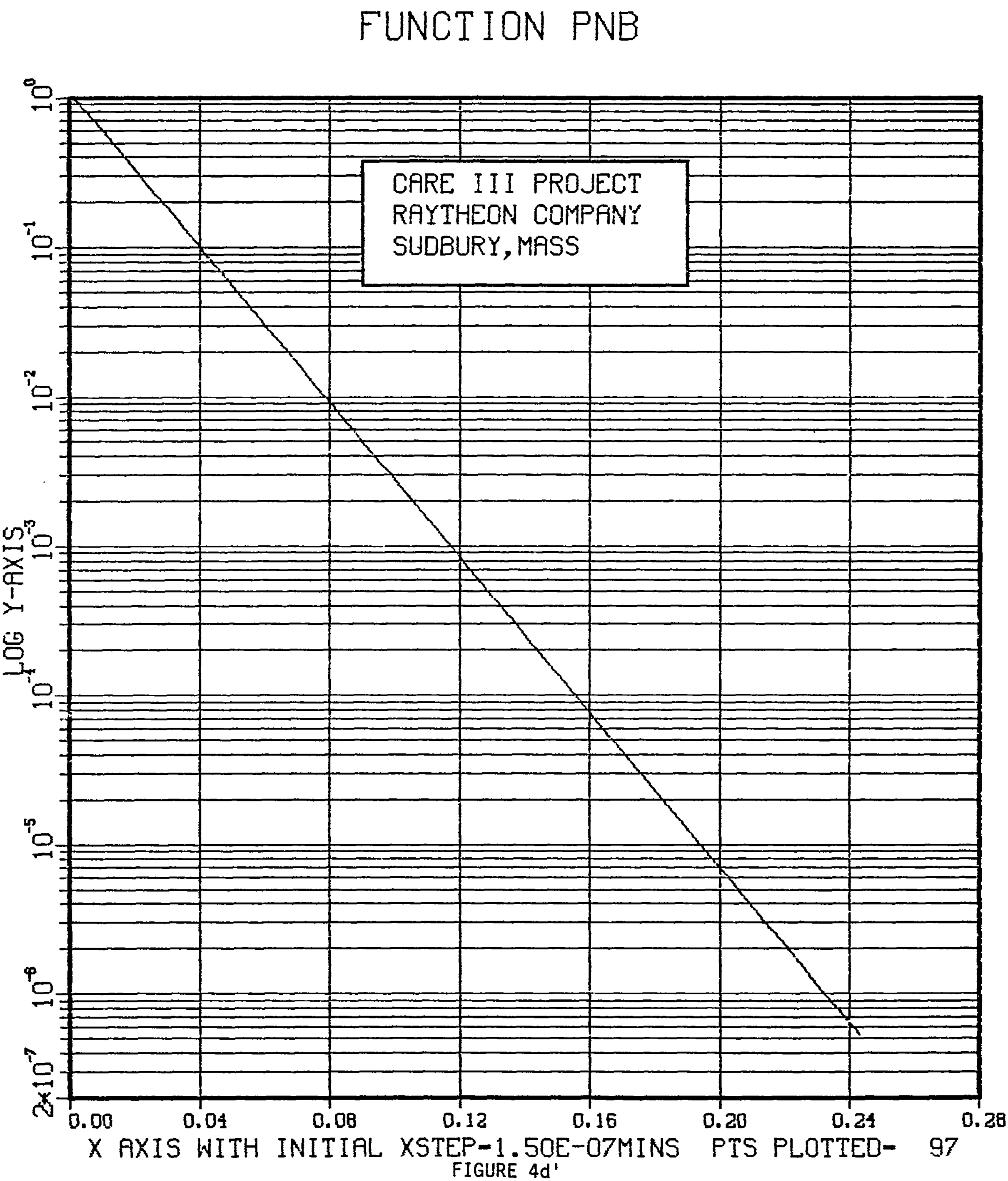

FIGURE 4d'

TEST CASE 3C

(Function PL vs. Time-Mins., CARE III Model)

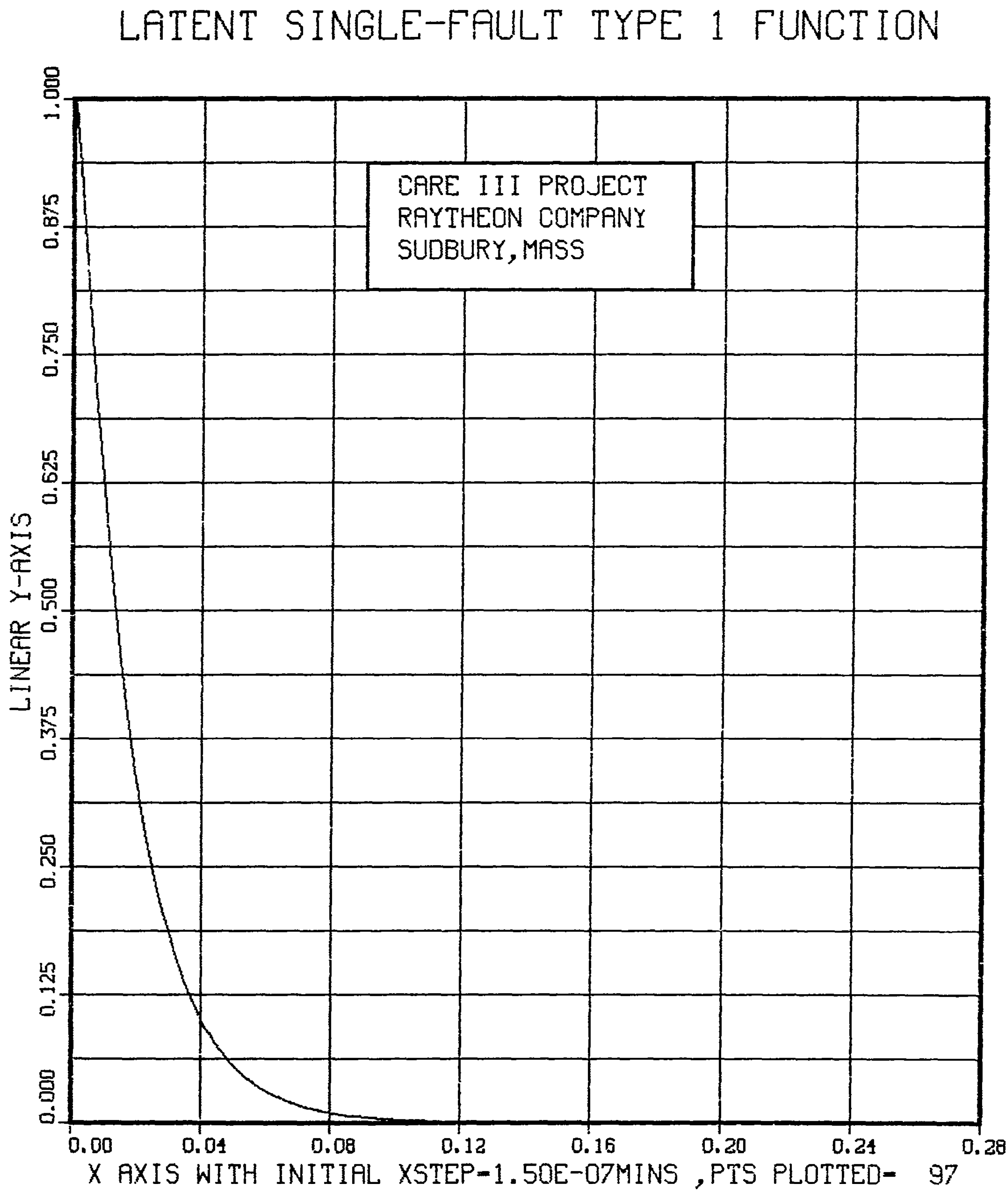

FIGURE 4e
TEST CASE 3C
TRUNC = 10^{-4}

(Function PL vs. Time-Mins., Markov Model)

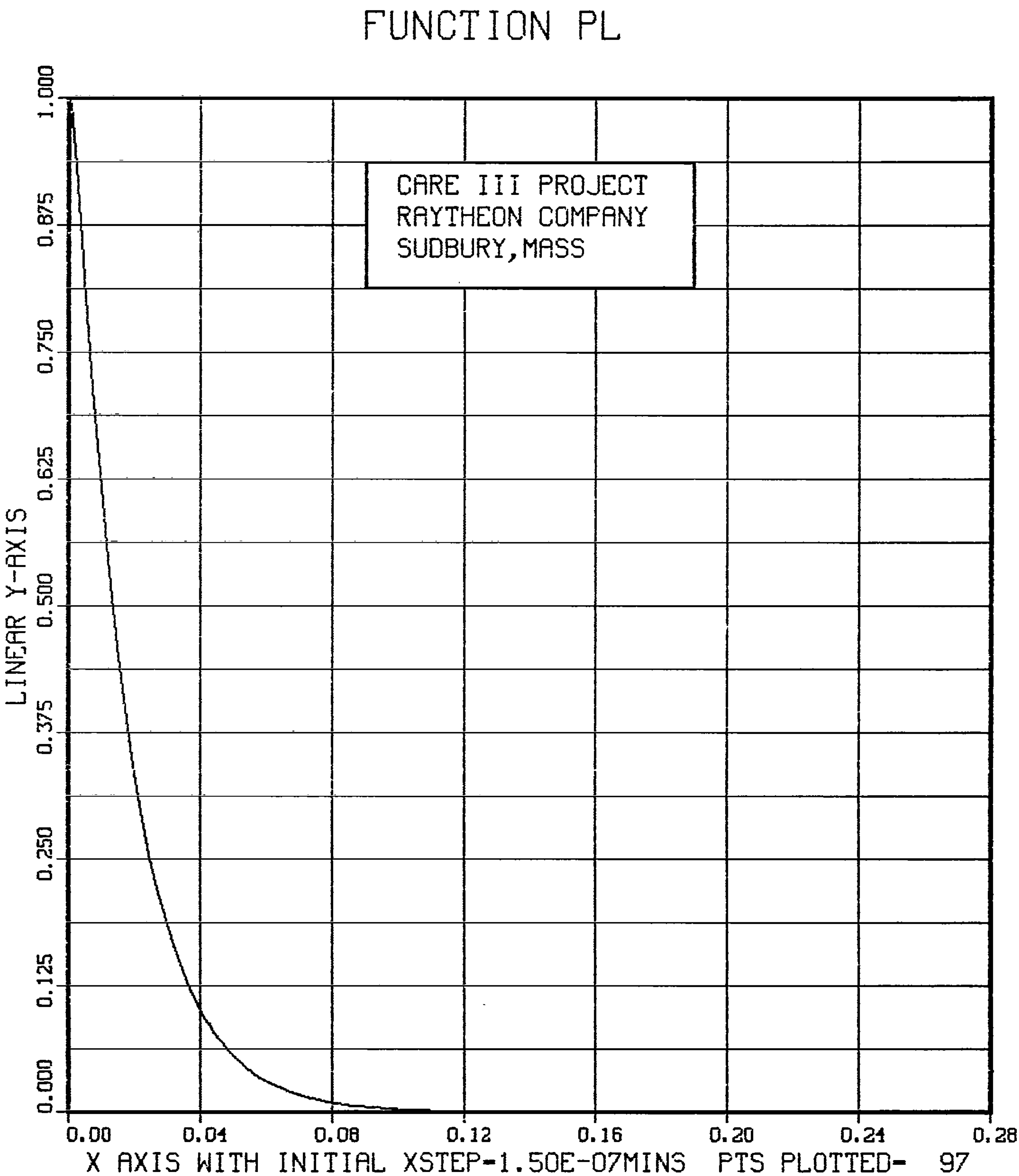

FIGURE 4e'
TEST CASE 3C

(Function PL vs. Time-Mins., CARE III Model)

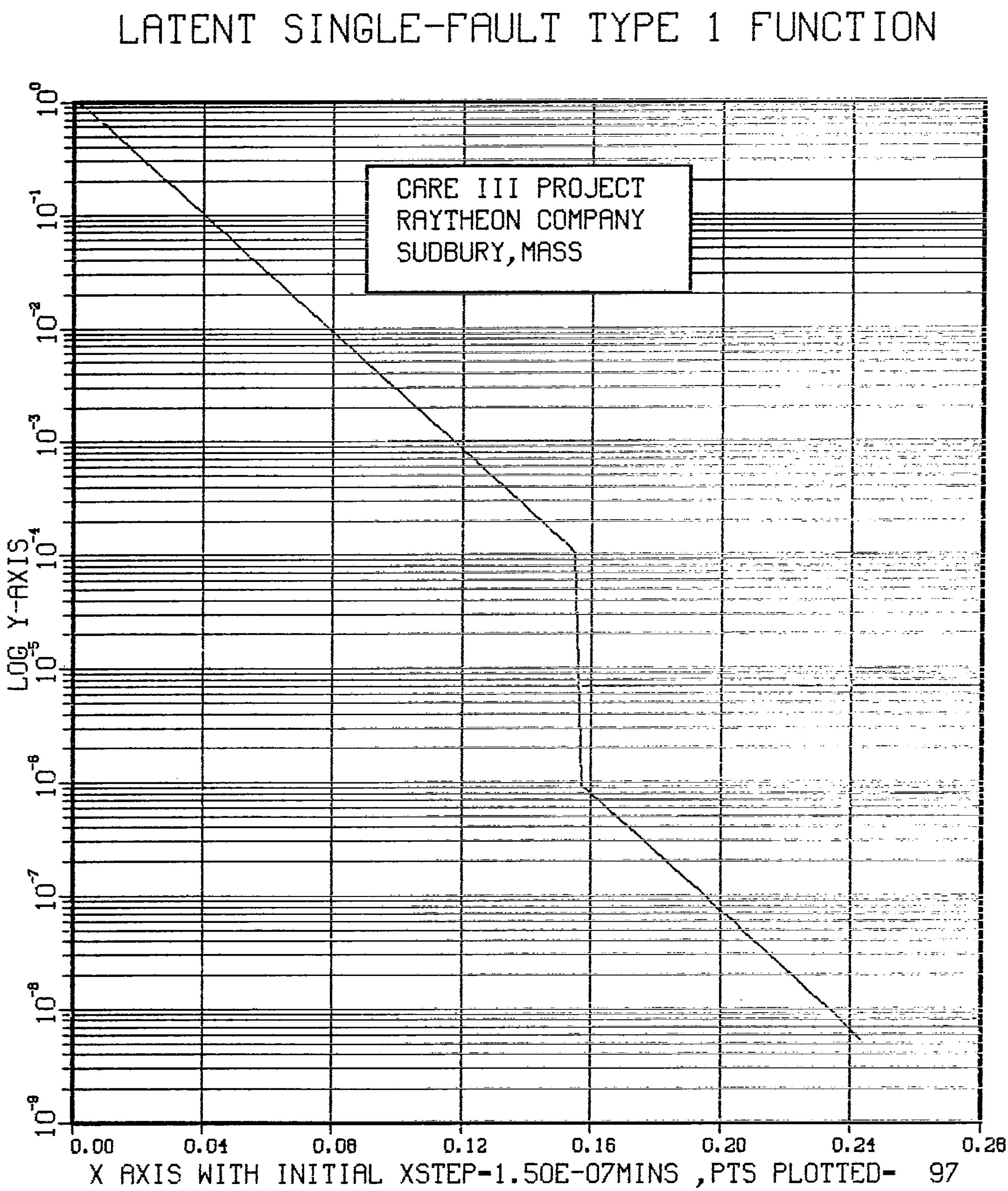

FIGURE 4f
TEST CASE 3C
TRUNC = 10-4

(Function PL vs. Time-Mins., Markov Model)

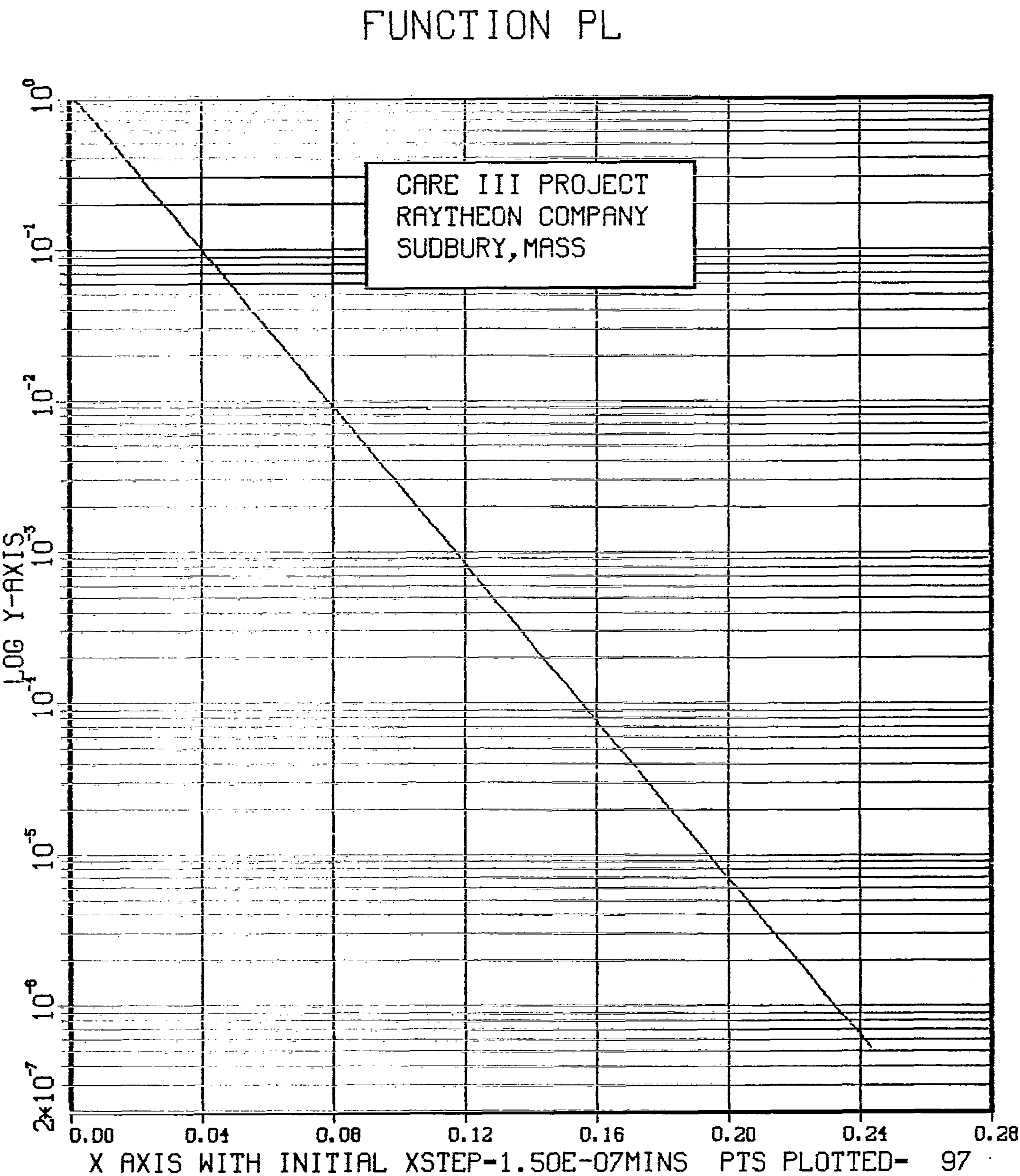

FIGURE 4f'
TEST CASE 3C

(Function PDF vs. Time-Mins., CARE III Model)

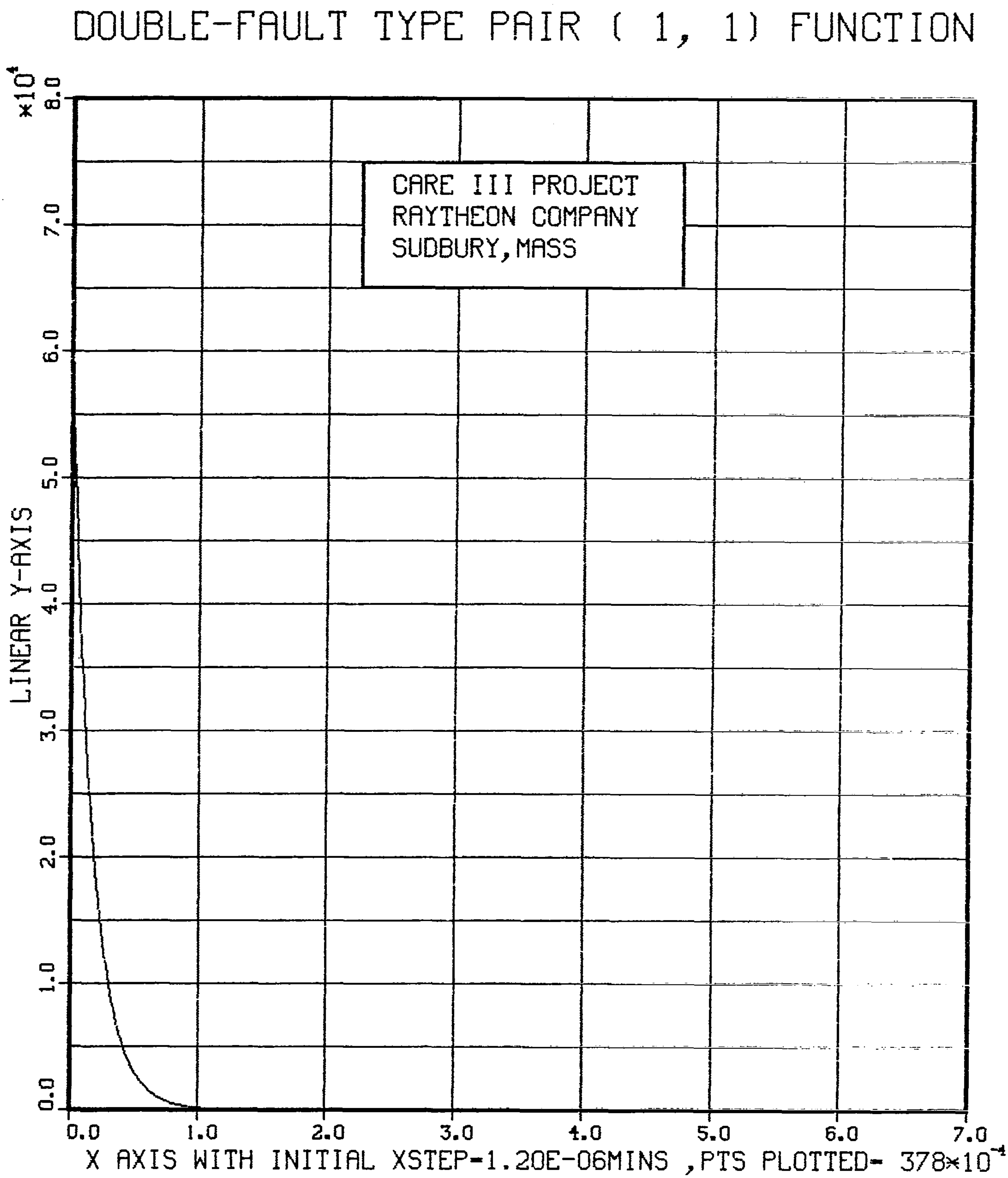

FIGURE 4g
TEST CASE 3C
TRUNC = 10^{-4}

(Function PDF vs. Time-Mins., CARE III Model)

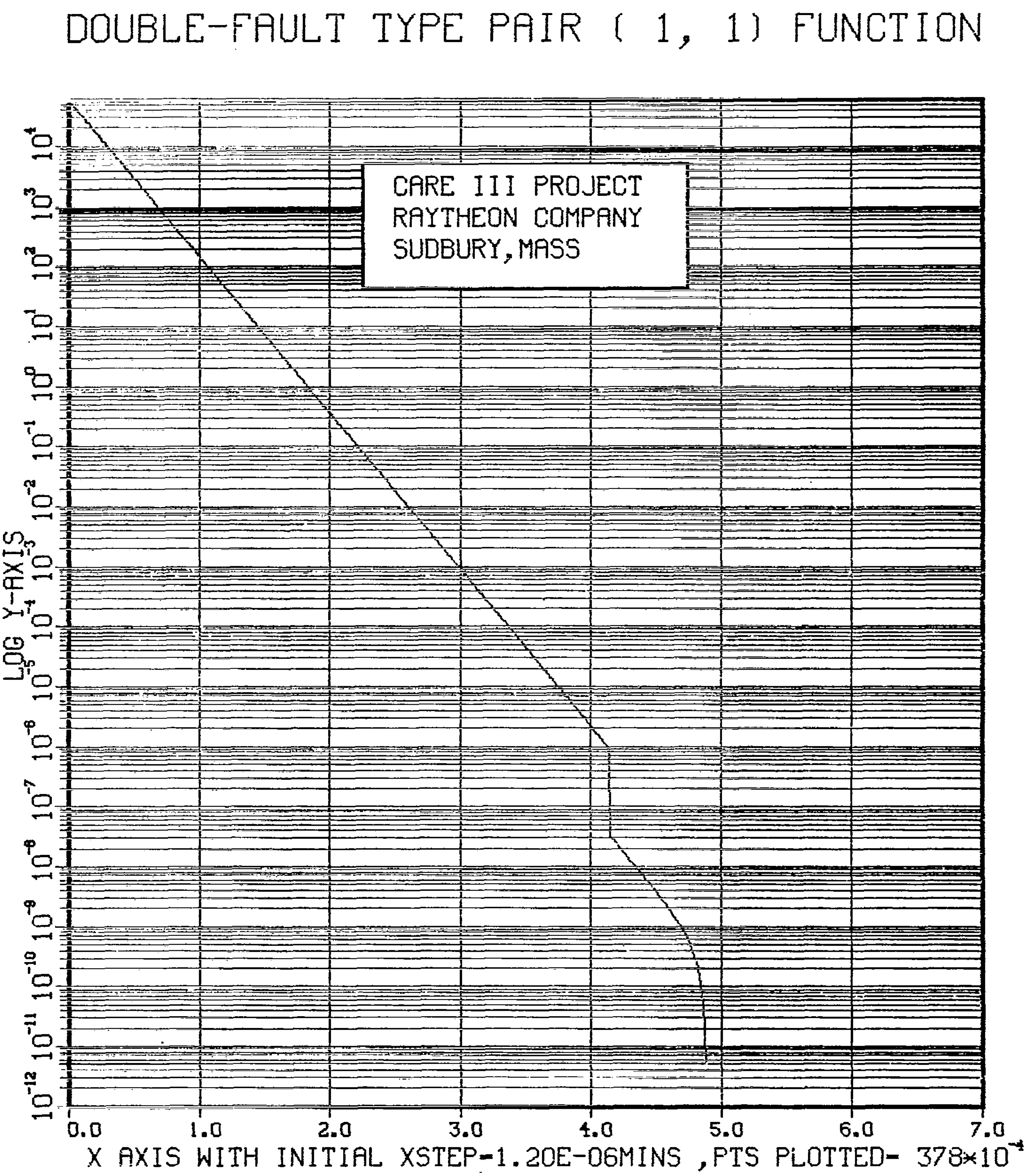

FIGURE 4h
TEST CASE 3C
TRUNC = 10^{-4}

(Function Q SUM vs. Time-Mins., CARE III Model)

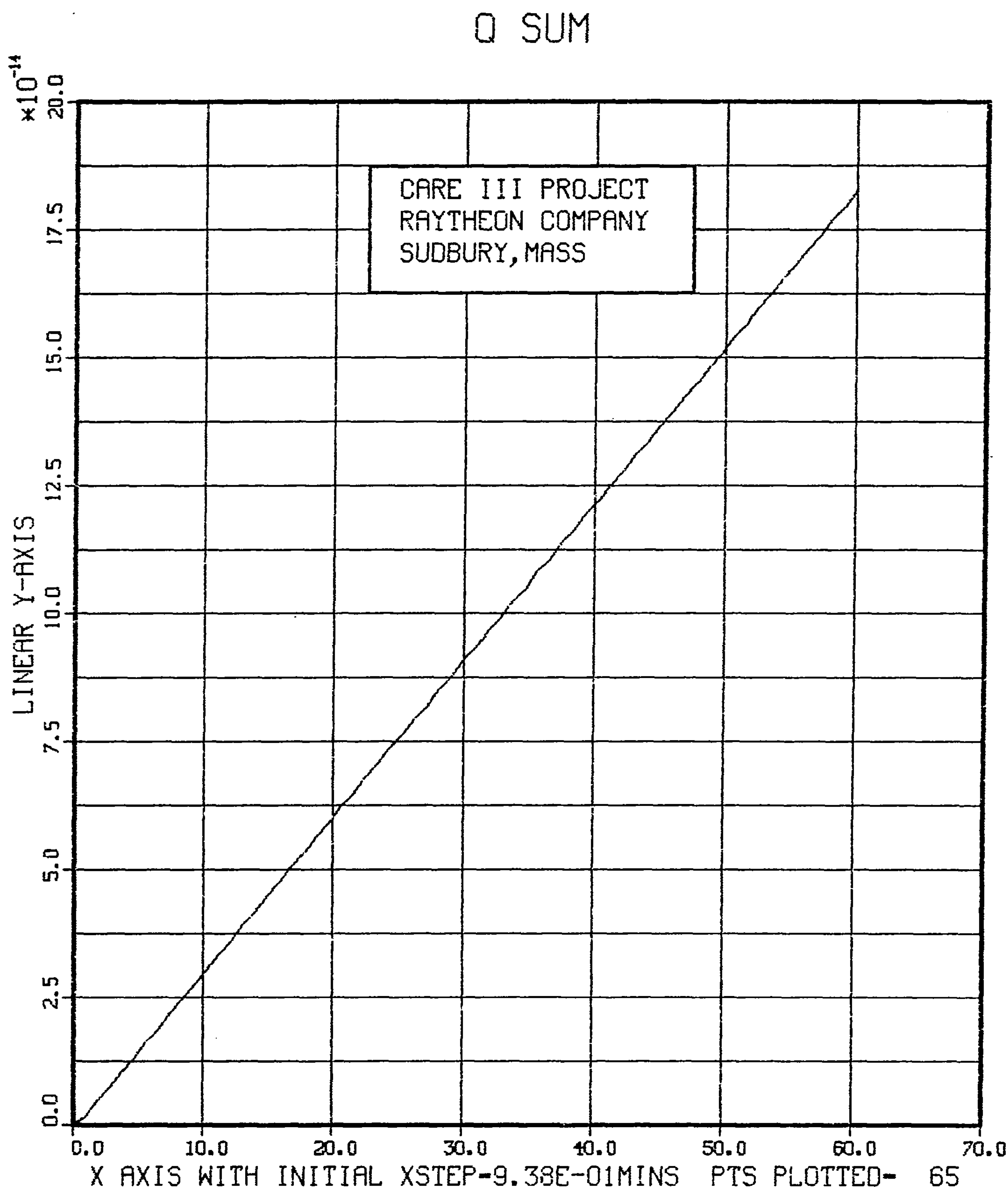

FIGURE 4i
TEST CASE 3C
TRUNC = 10^{-4}

(Function Q SUM vs. Time-Mins., CARE III Model)

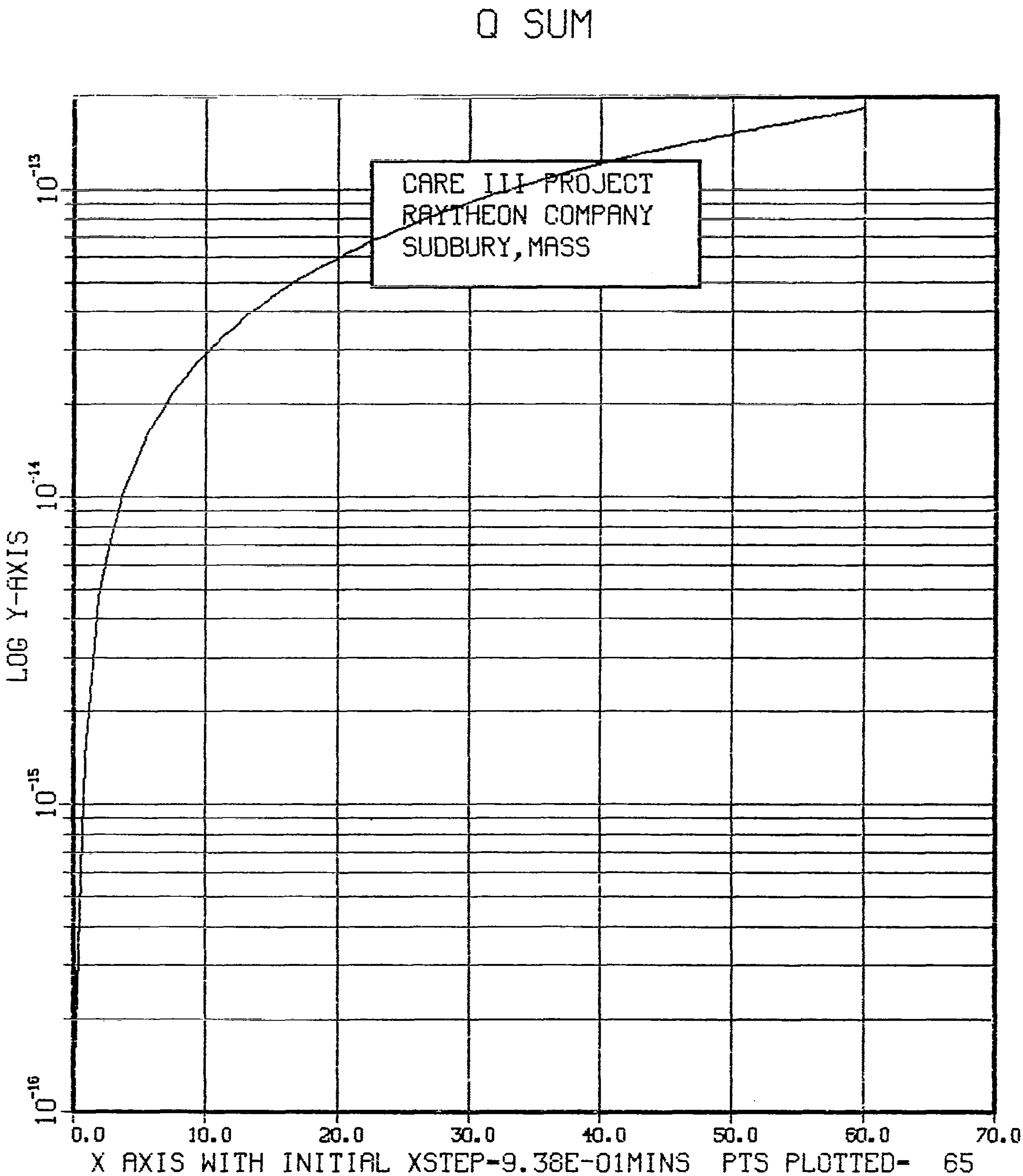

FIGURE 4j
TEST CASE 3C
TRUNC = 10^{-4}

(Function P* SUM vs. Time-Mins., CARE III Model)

P* SUM

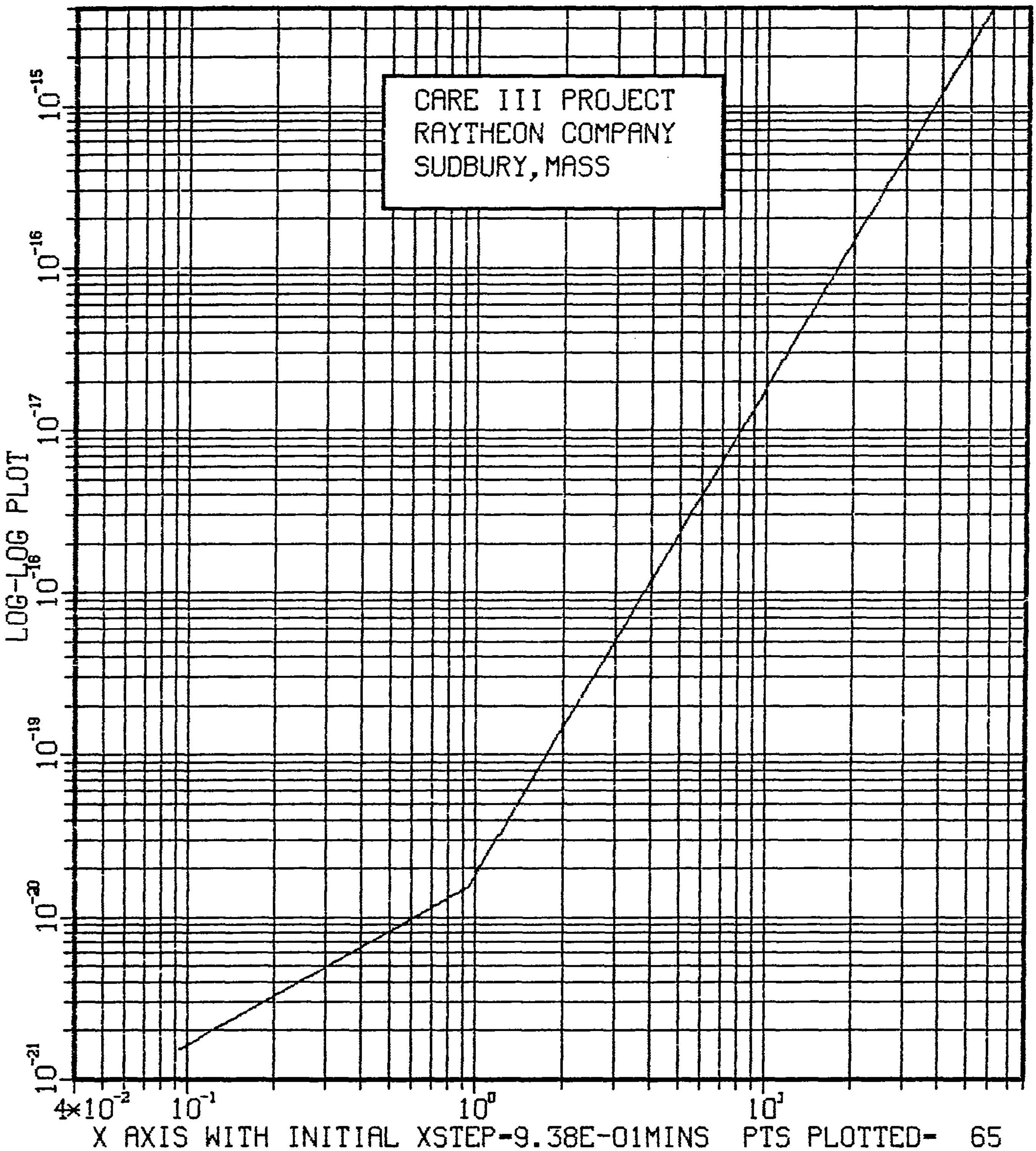

FIGURE 4k
TEST CASE 3C
TRUNC = 10^{-4}

(Function Q + P* SUM vs. Time-Mins., CARE III Model)

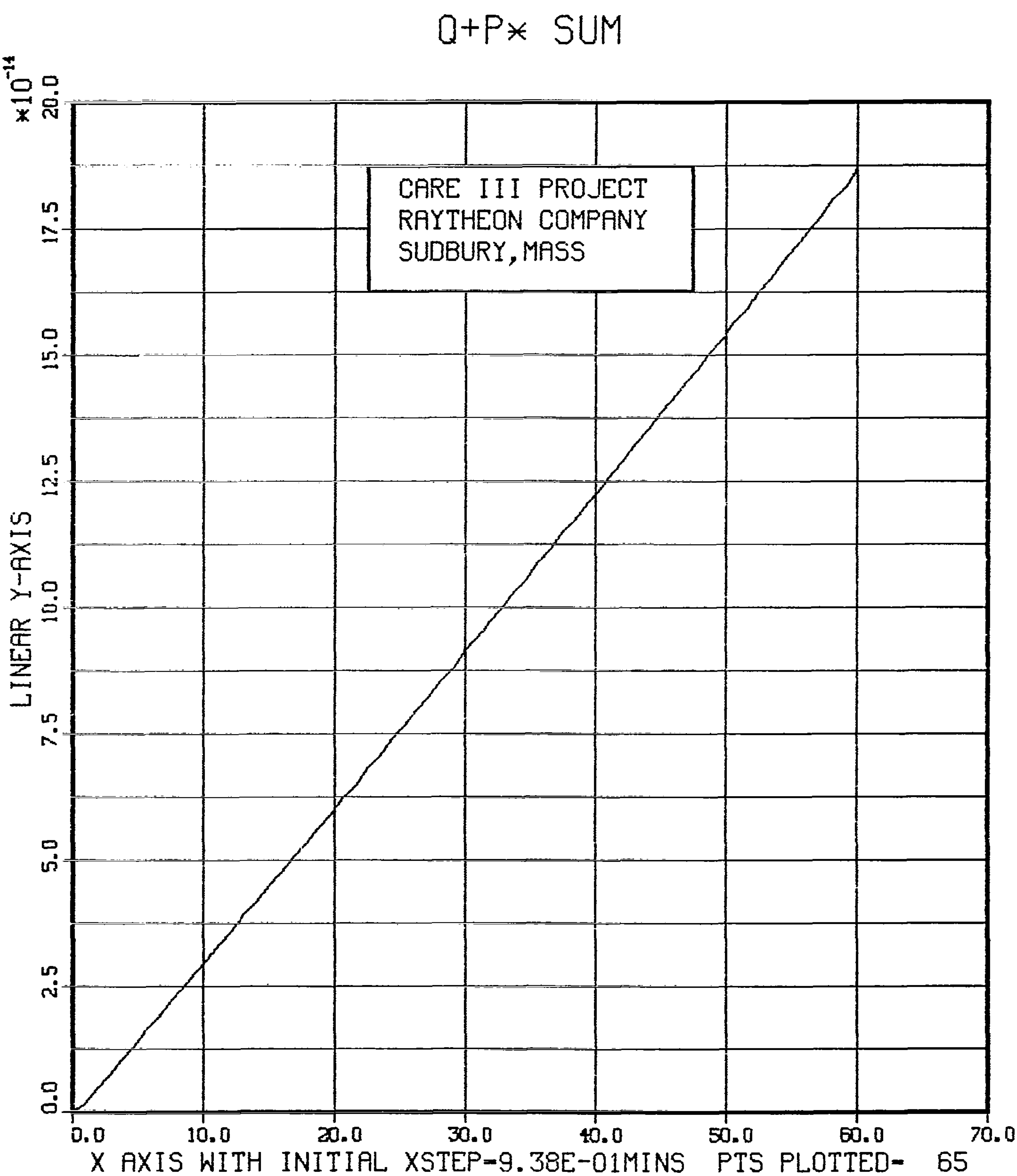

FIGURE 41
TEST CASE 3C
TRUNC = 10^{-4}

(Function Q + P* SUM vs. Time-Mins., CARE III Model)

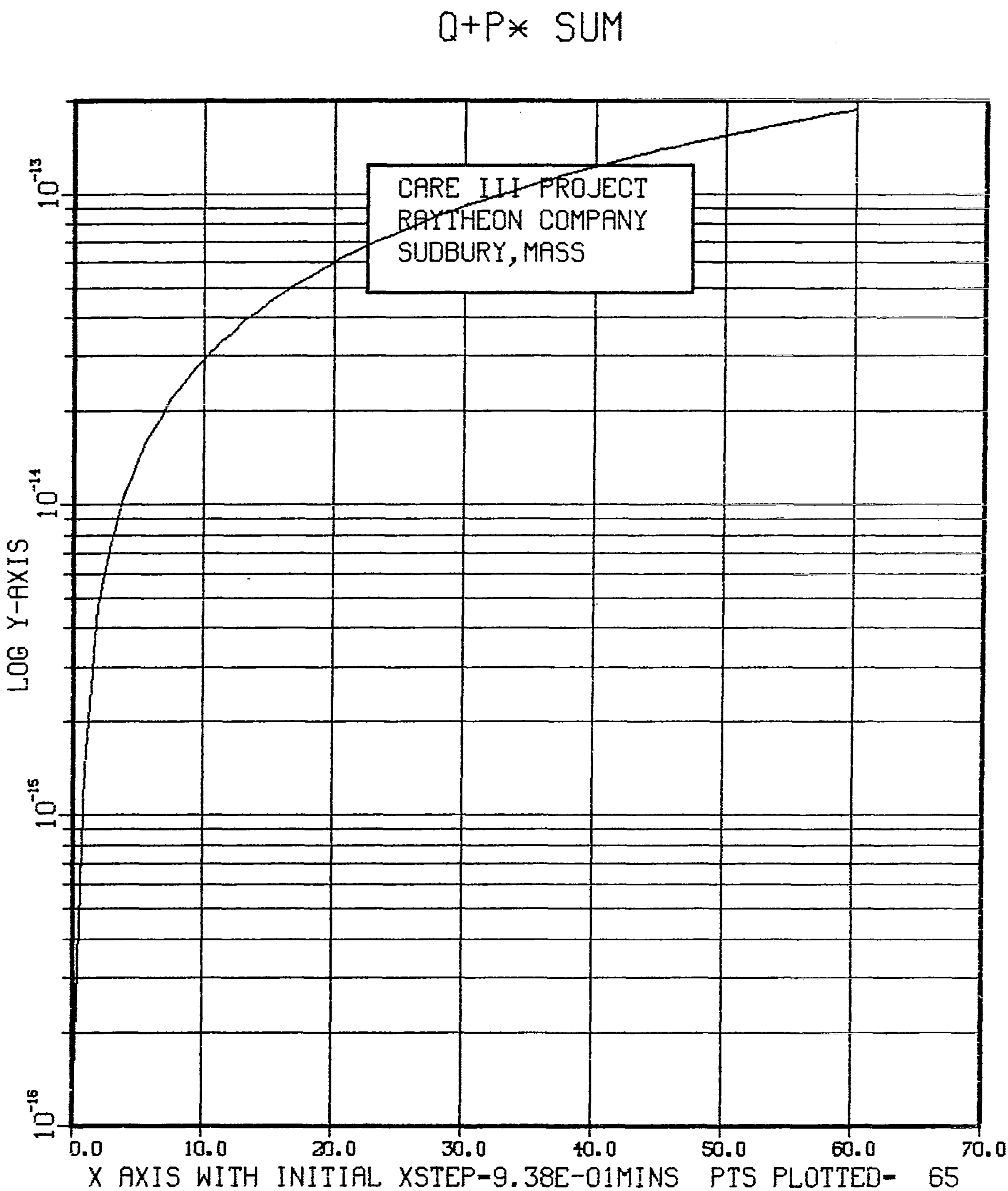

FIGURE 4m
TEST CASE 3C
TRUNC = 10^{-4}

(Function PB vs. Time-Mins., CARE III Model)

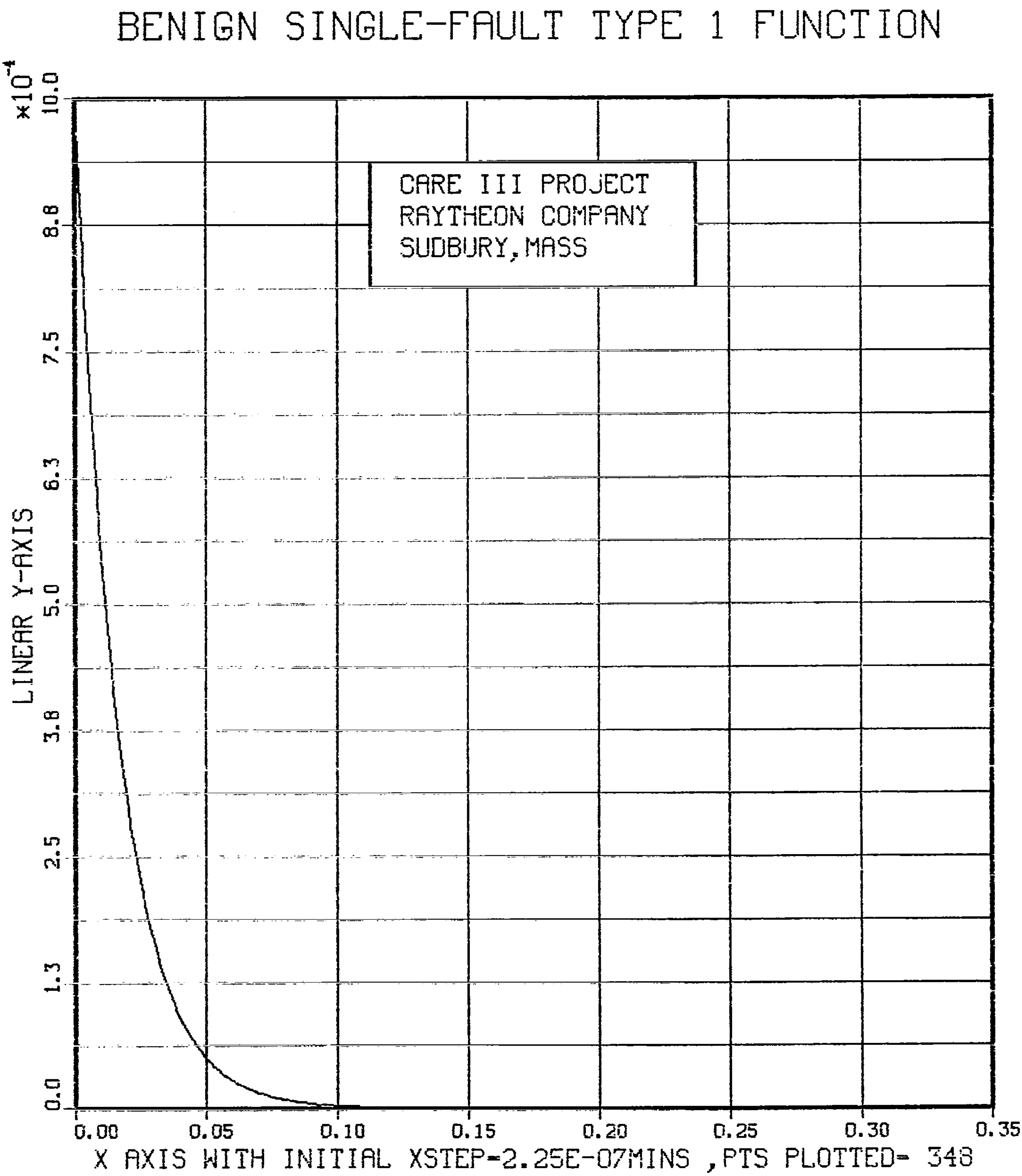

FIGURE 4'a
TEST CASE 3C
TRUNC = 10^{-6}

(Function PB vs. Time-Mins., CARE III Model)

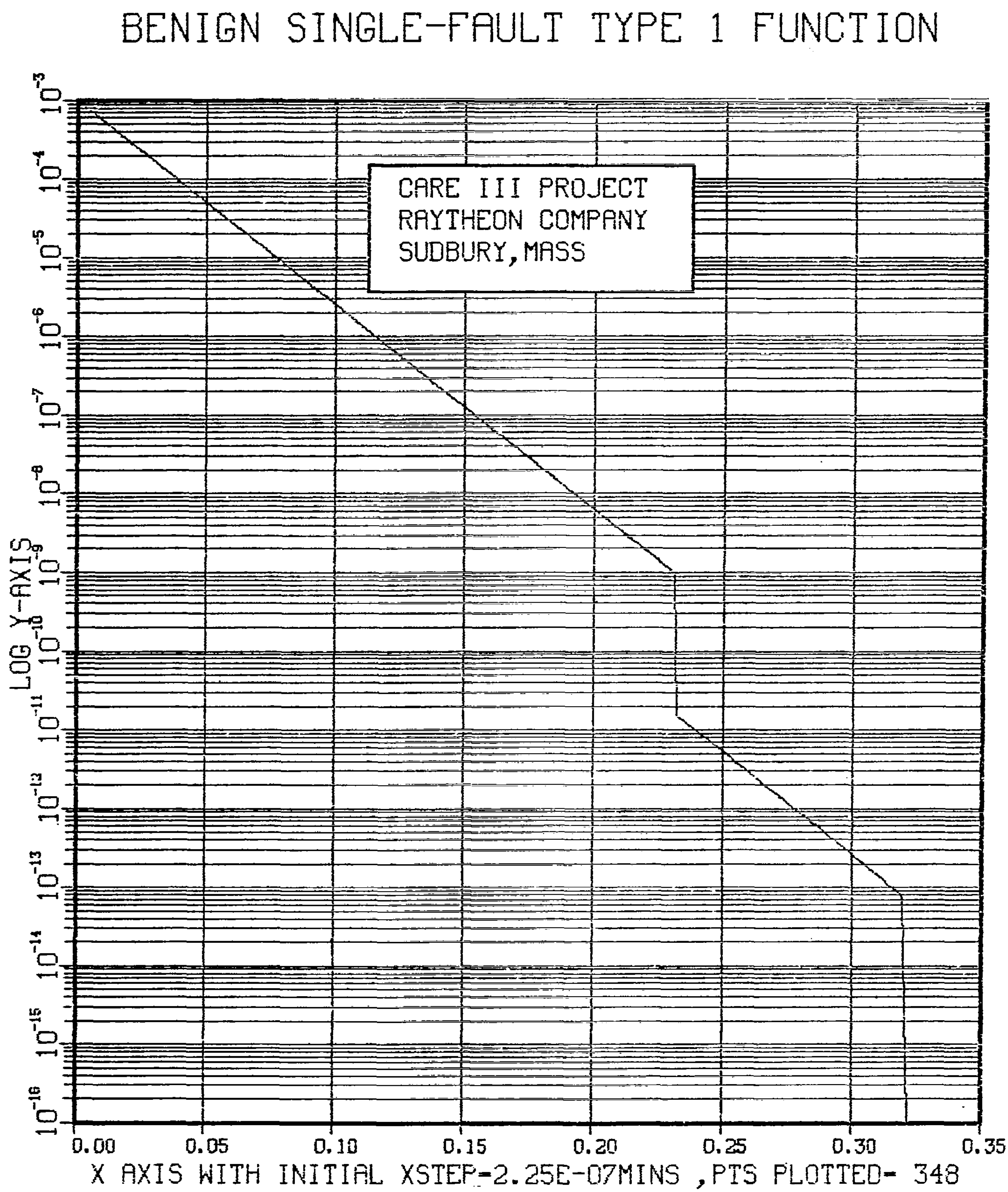

FIGURE 4'b
TEST CASE 3C
TRUNC = 10^{-6}

(Function PNB vs. Time-Mins., CARE III Model)

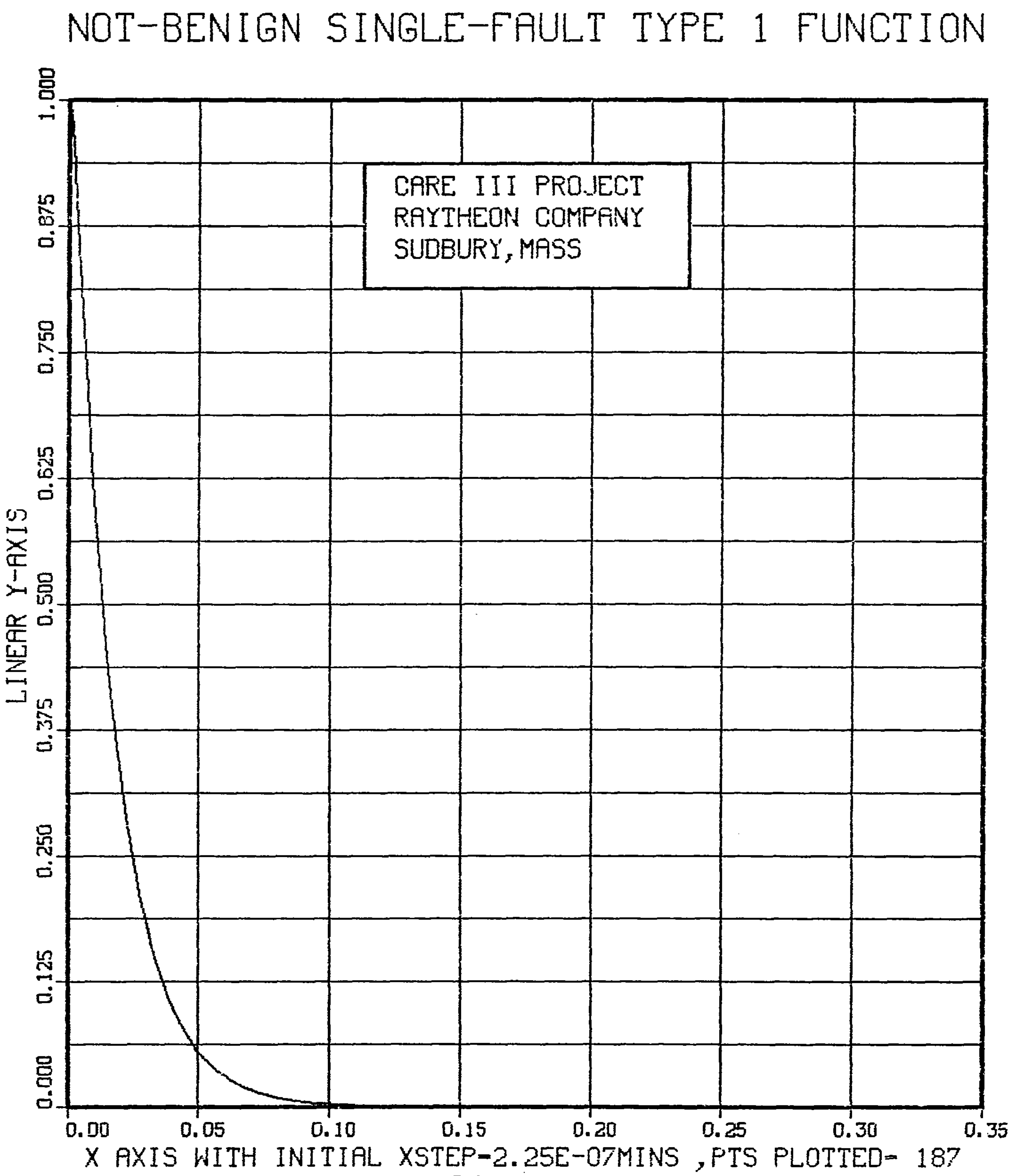

FIGURE 4'c
TEST CASE 3C
TRUNC = 10^{-6}

(Function PNB vs. Time-Mins., CARE III Model)

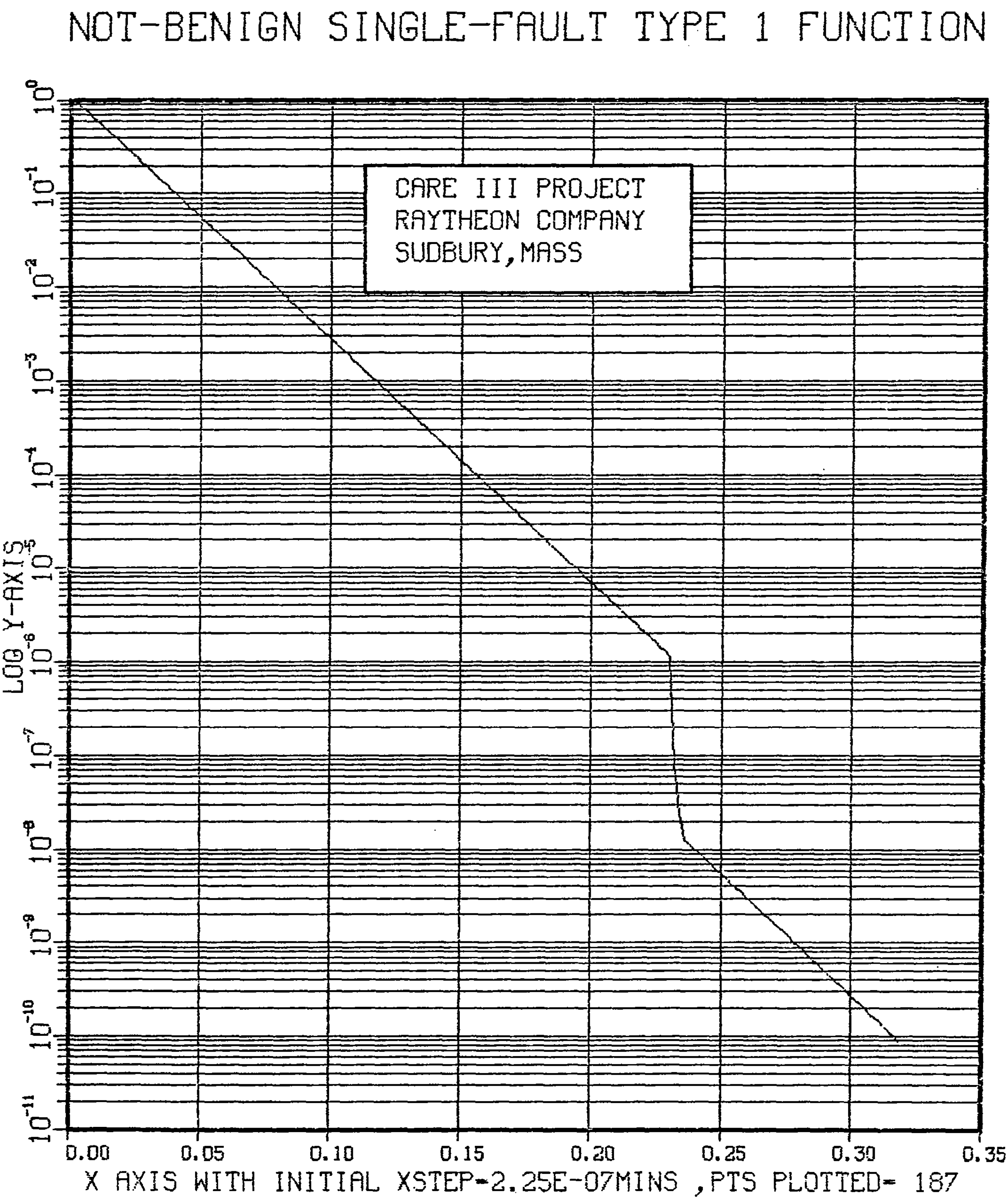

FIGURE 4'd
TEST CASE 3C
TRUNC = 10^{-6}

(Function PL vs. Time-Mins., CARE III Model)

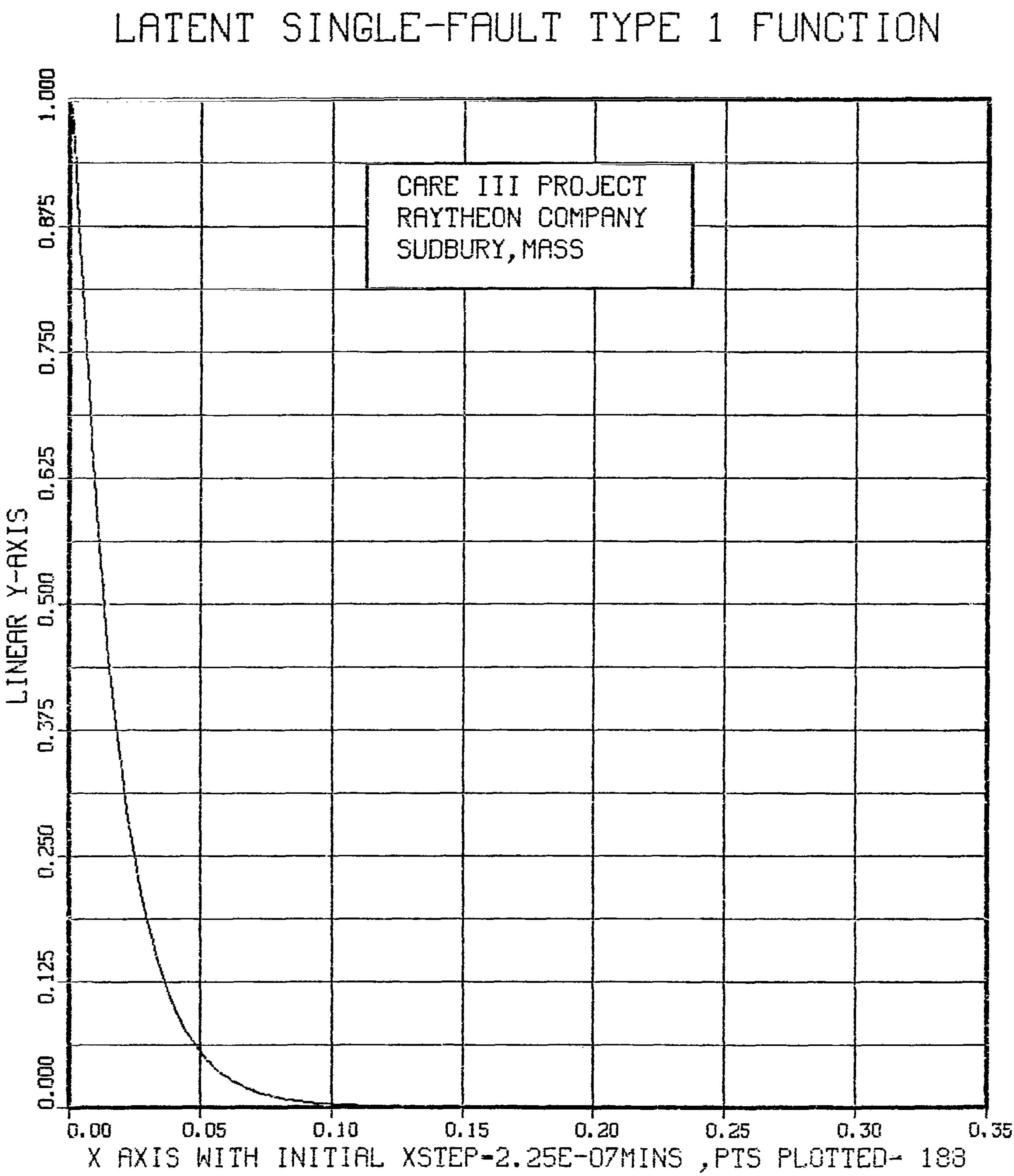

FIGURE 4'e
TEST CASE 3C
TRUNC = 10^{-6}

(Function PL vs. Time-Mins., CARE III Model)

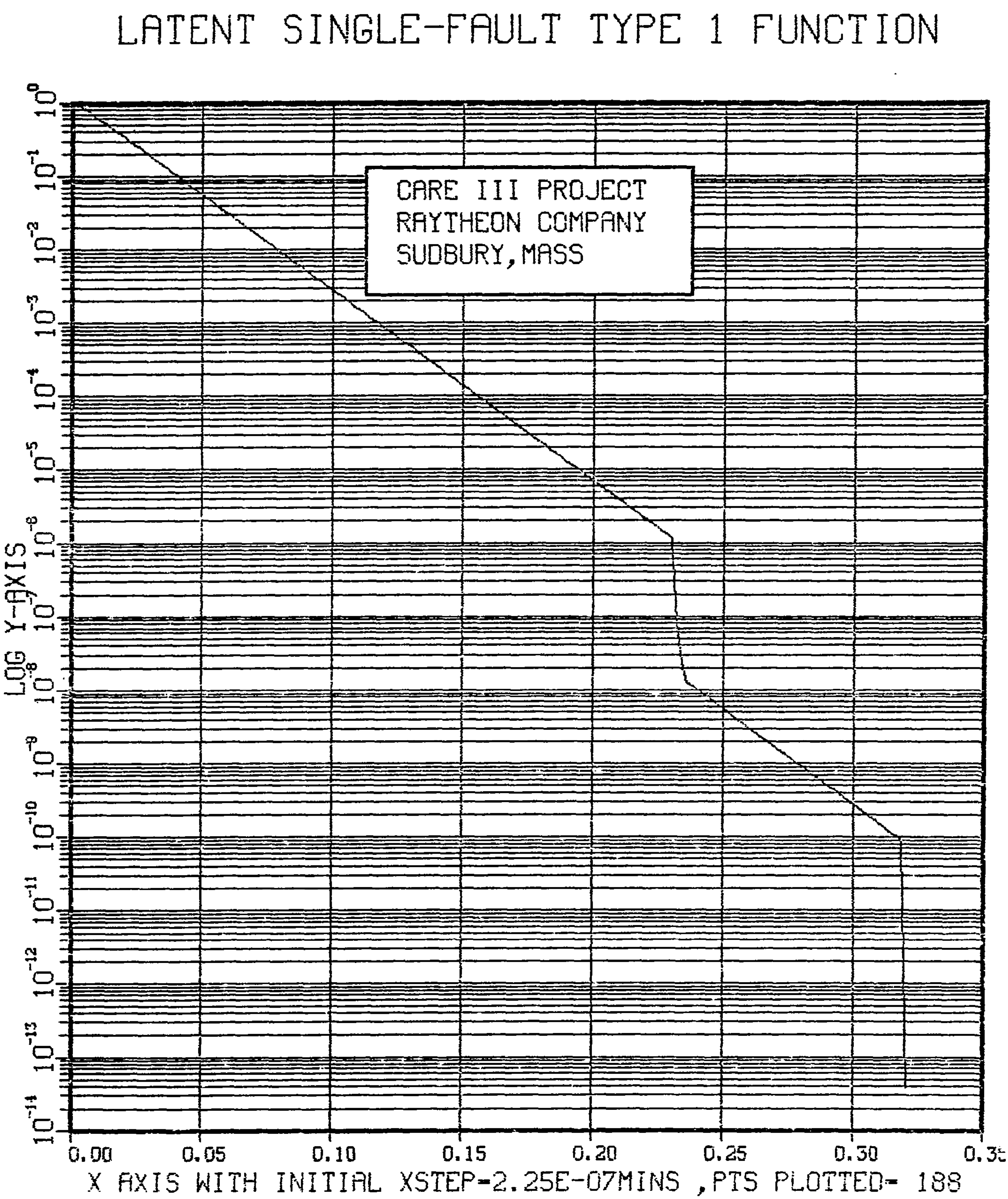

FIGURE 4'f
TEST CASE 3C
TRUNC = 10^{-6}

(Function PB vs. Time-Mins., CARE III Model)

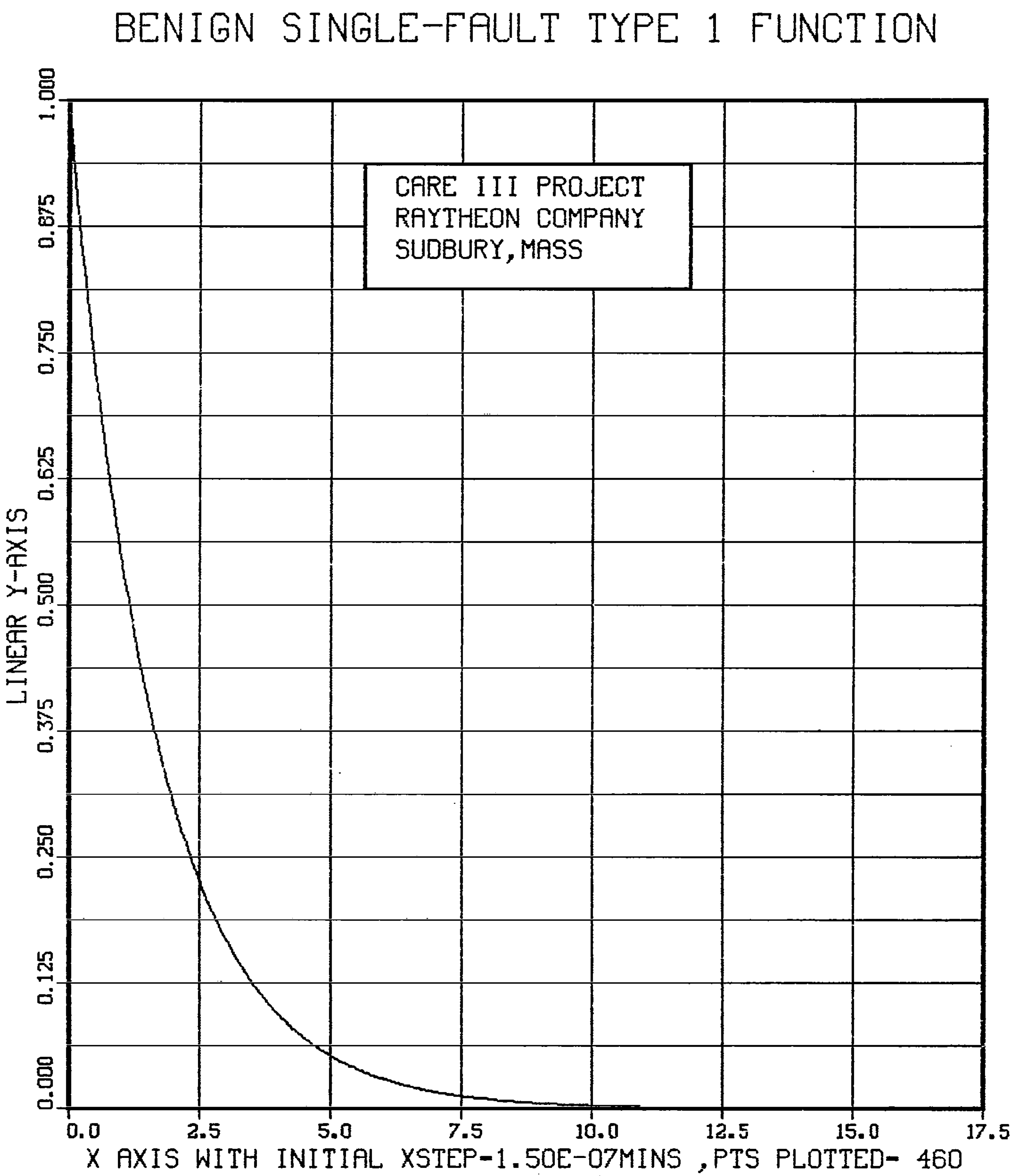

FIGURE 5a
TEST CASE 3D'

(Function PB vs. Time-Mins., CARE III Model)

BENIGN SINGLE-FAULT TYPE 1 FUNCTION

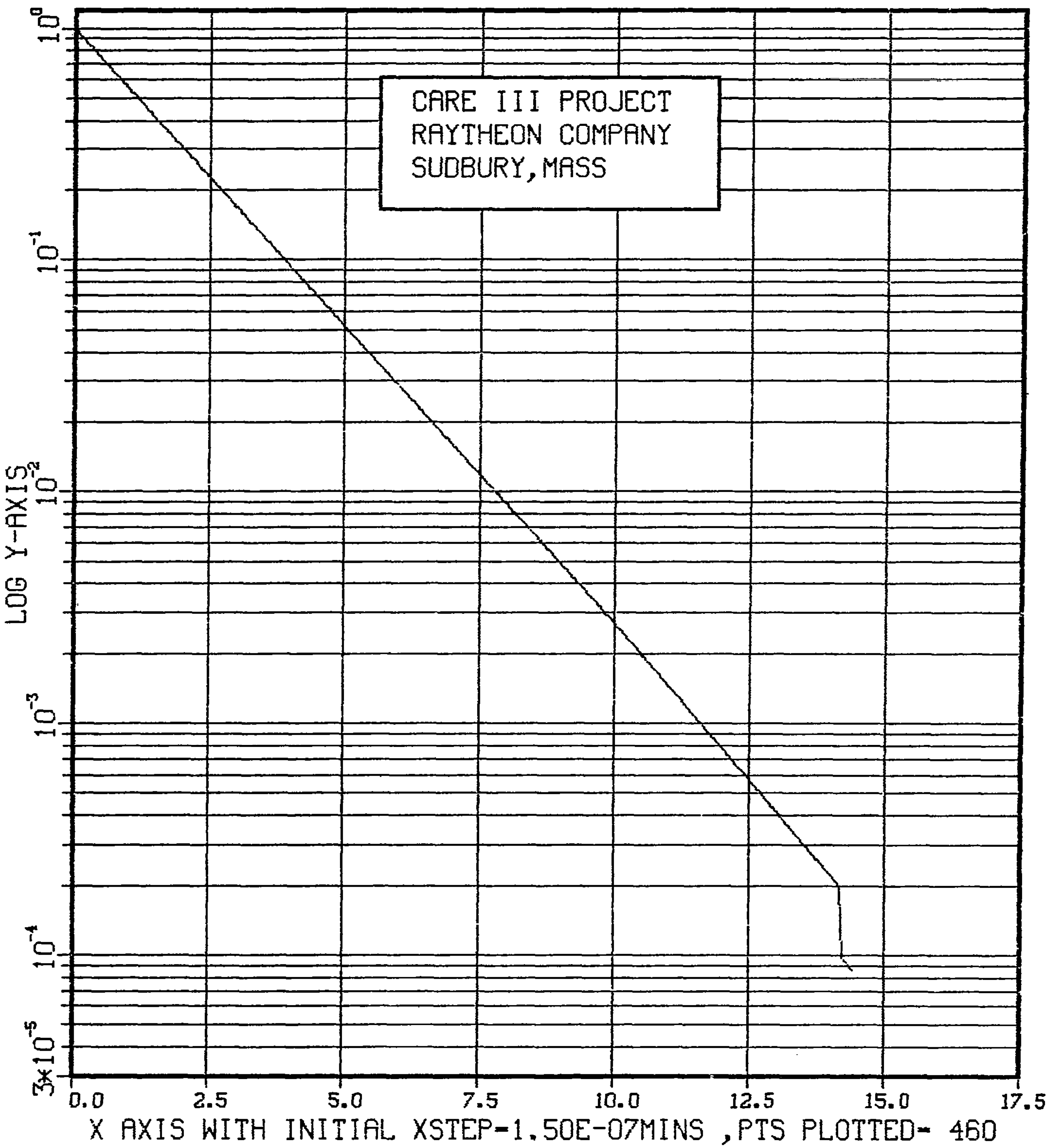

FIGURE 5a'
TEST CASE 3D'

(Function PB vs. Time-Mins., CARE III Model)

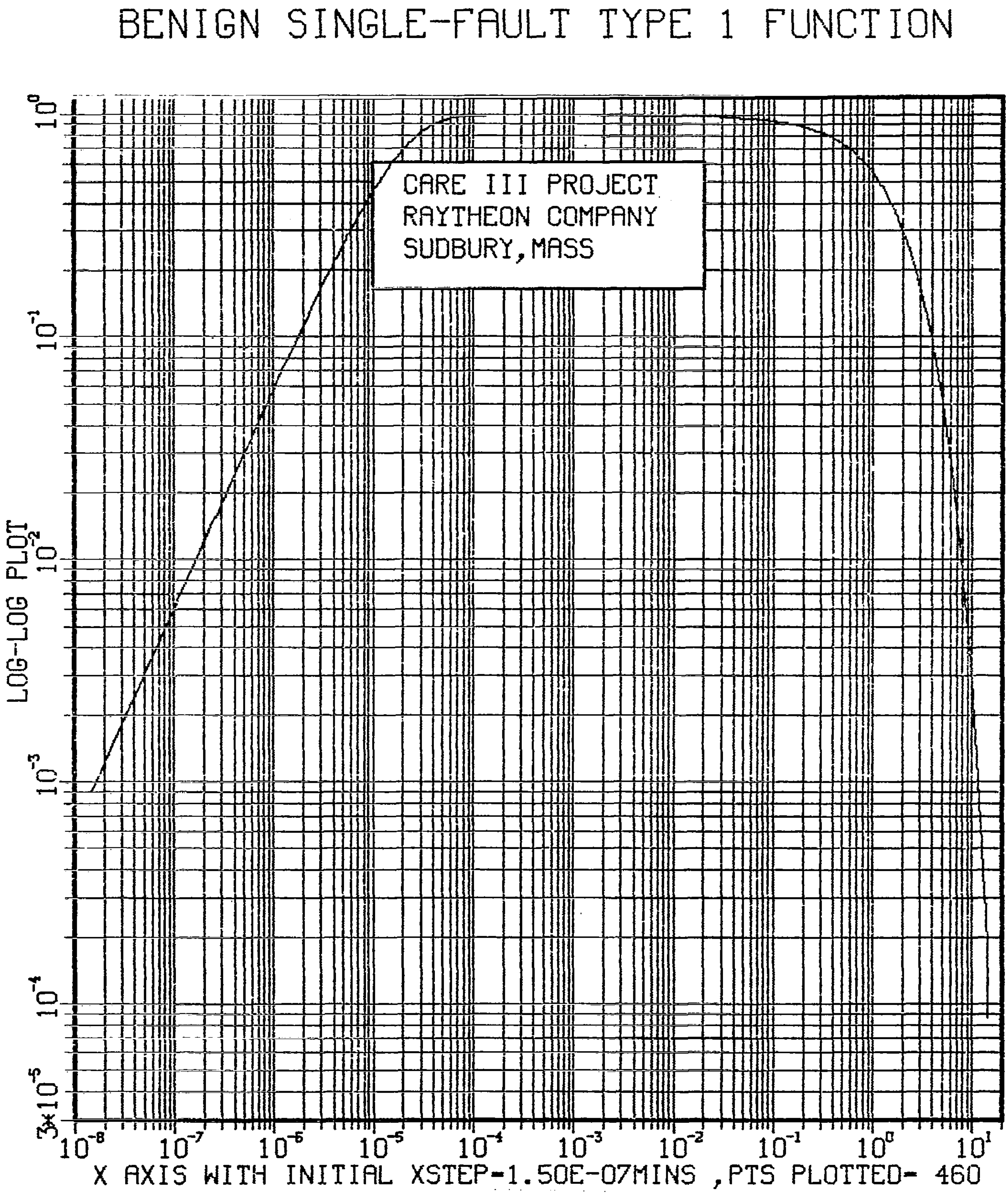

FIGURE 5a''
TEST CASE 3D'

(Function PB vs. Time-Mins., Markov Model)

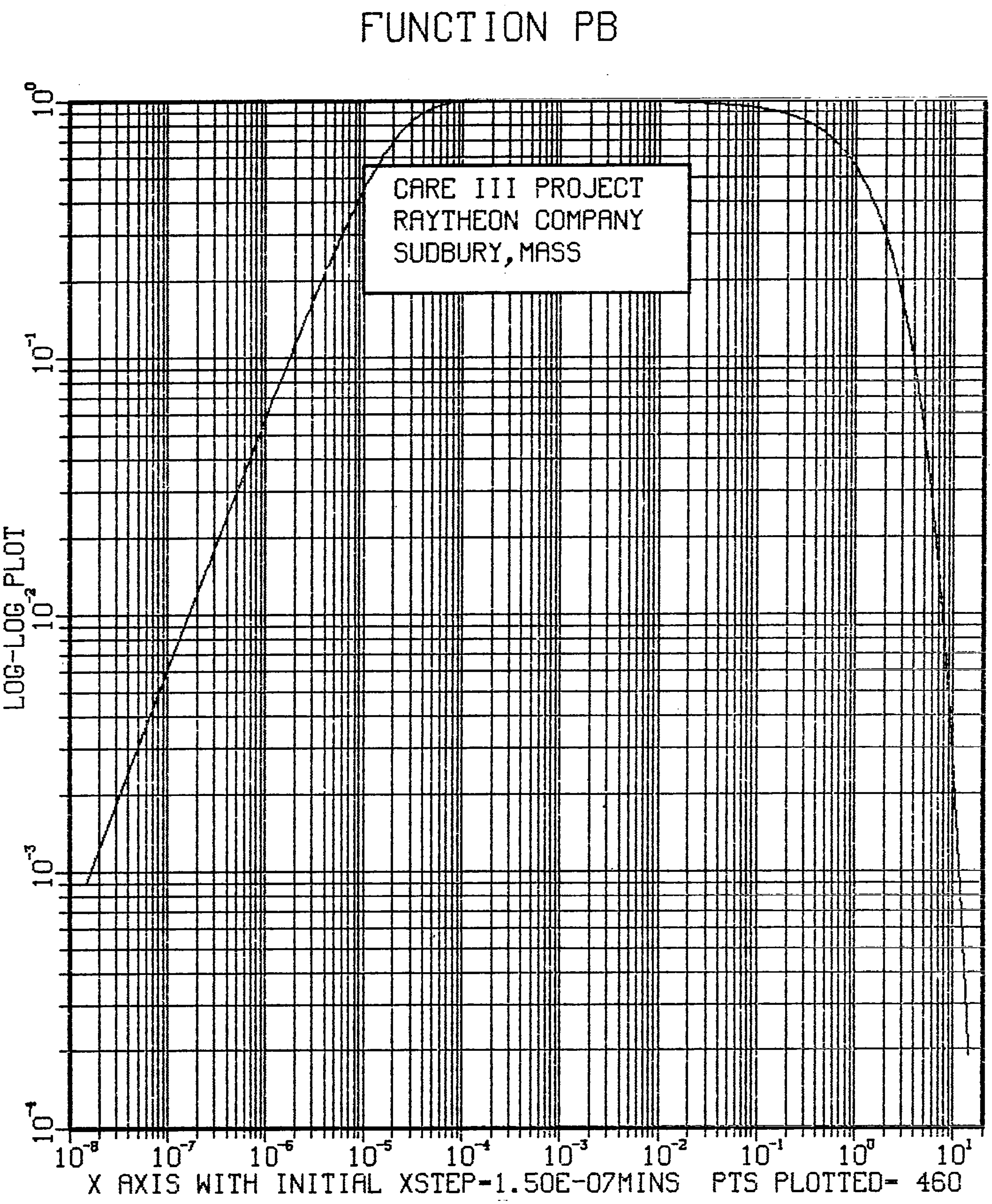

FIGURE 5a'''
TEST CASE 3D'

(Function PNB vs. Time-Mins., CARE III Model)

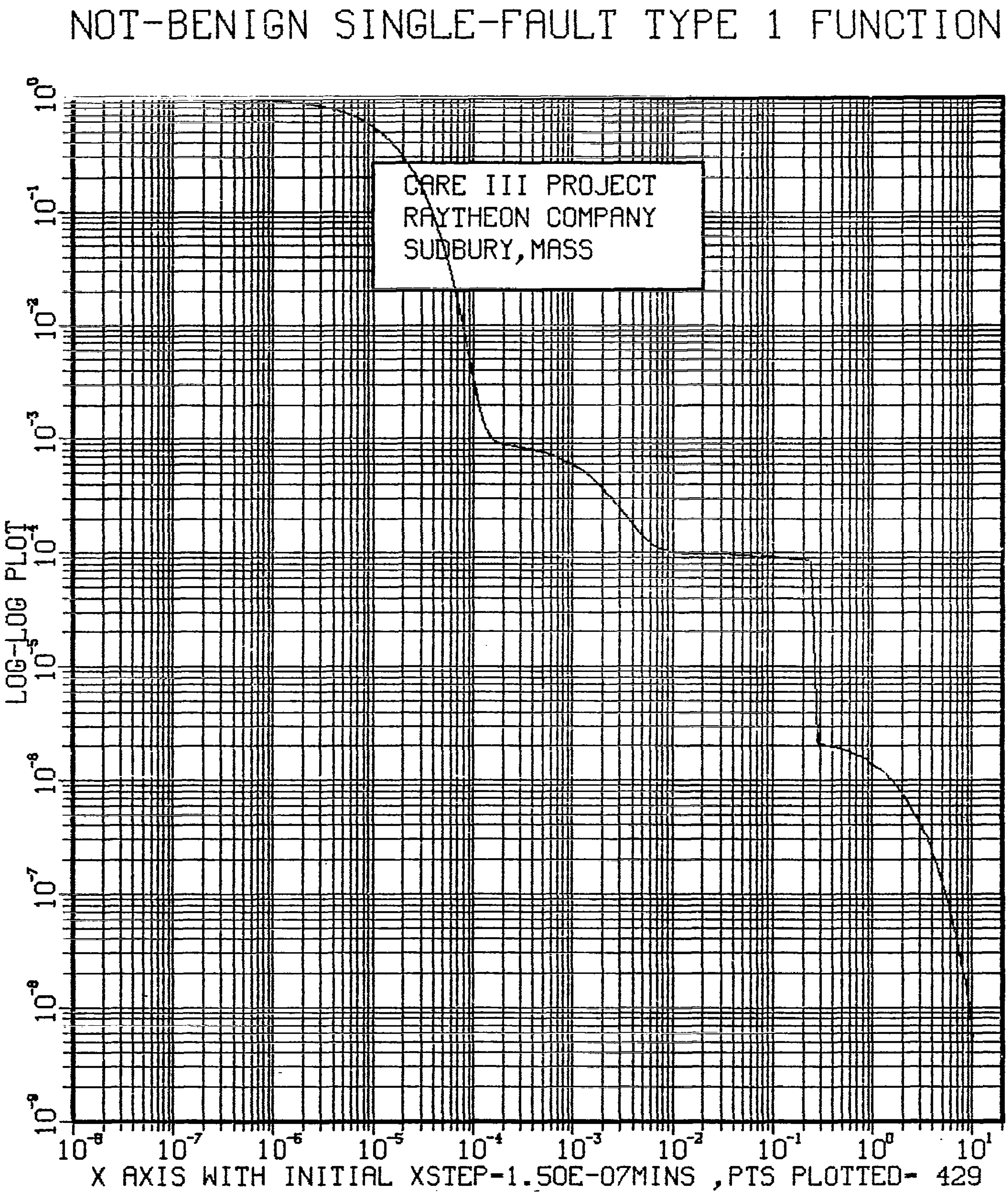

FIGURE 5b
TEST CASE 3D'

(Function PNB vs. Time-Mins., Markov Model)

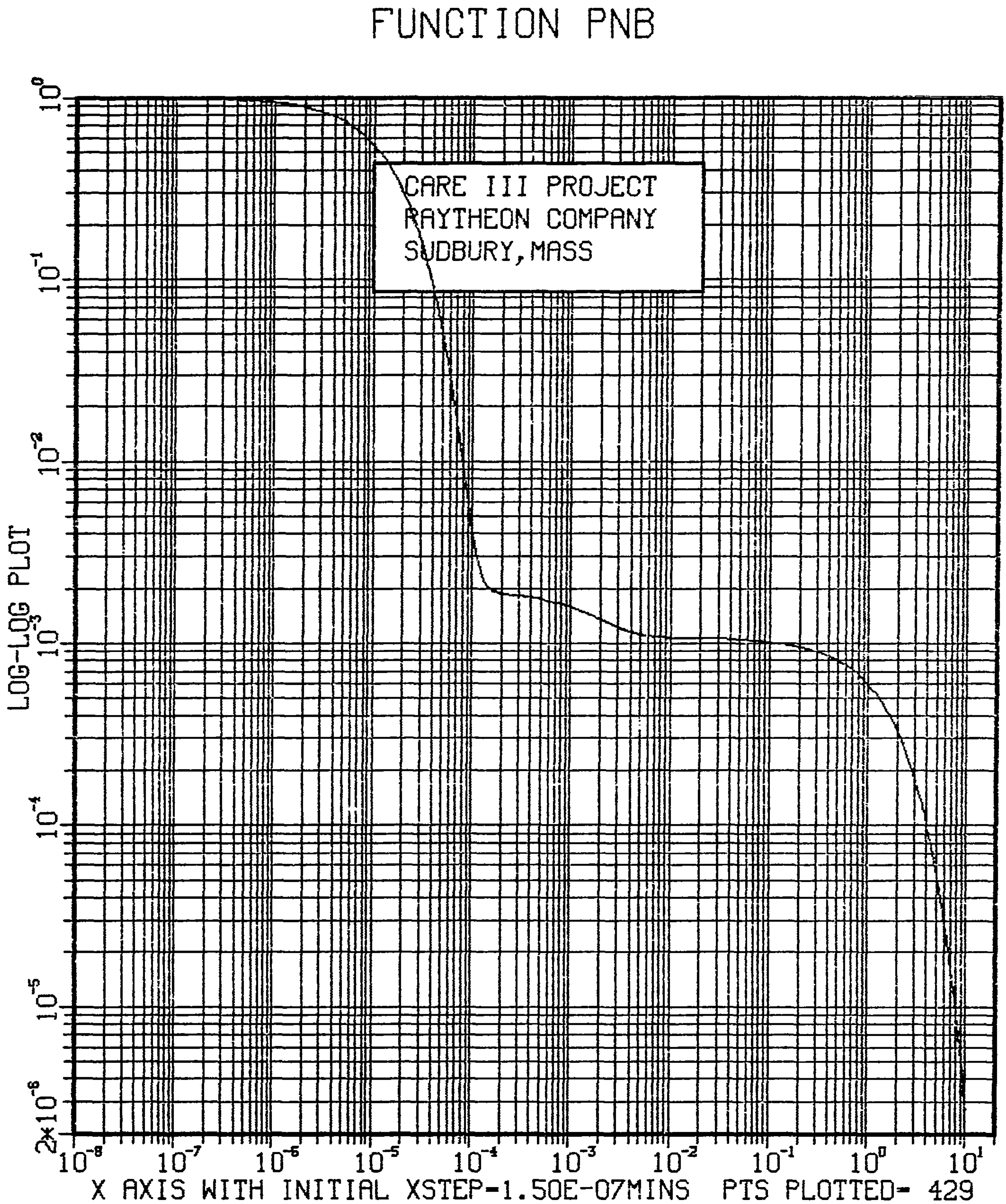

FIGURE 5b'
TEST CASE 3D'

(Function PL vs. Time-Mins., CARE III Model)

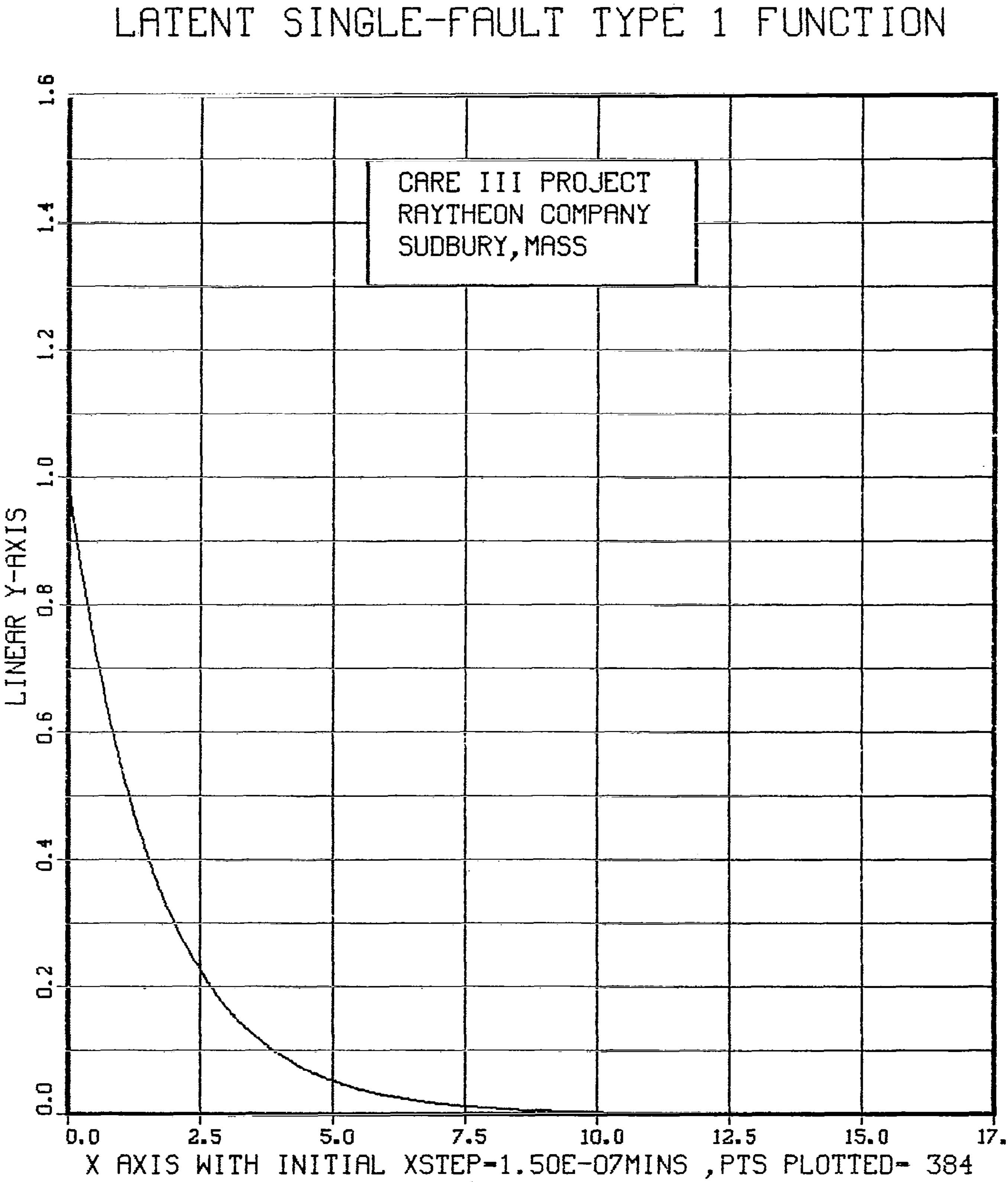

FIGURE 5c
TEST CASE 3D'

(Function PL vs. Time-Mins., CARE III Model)

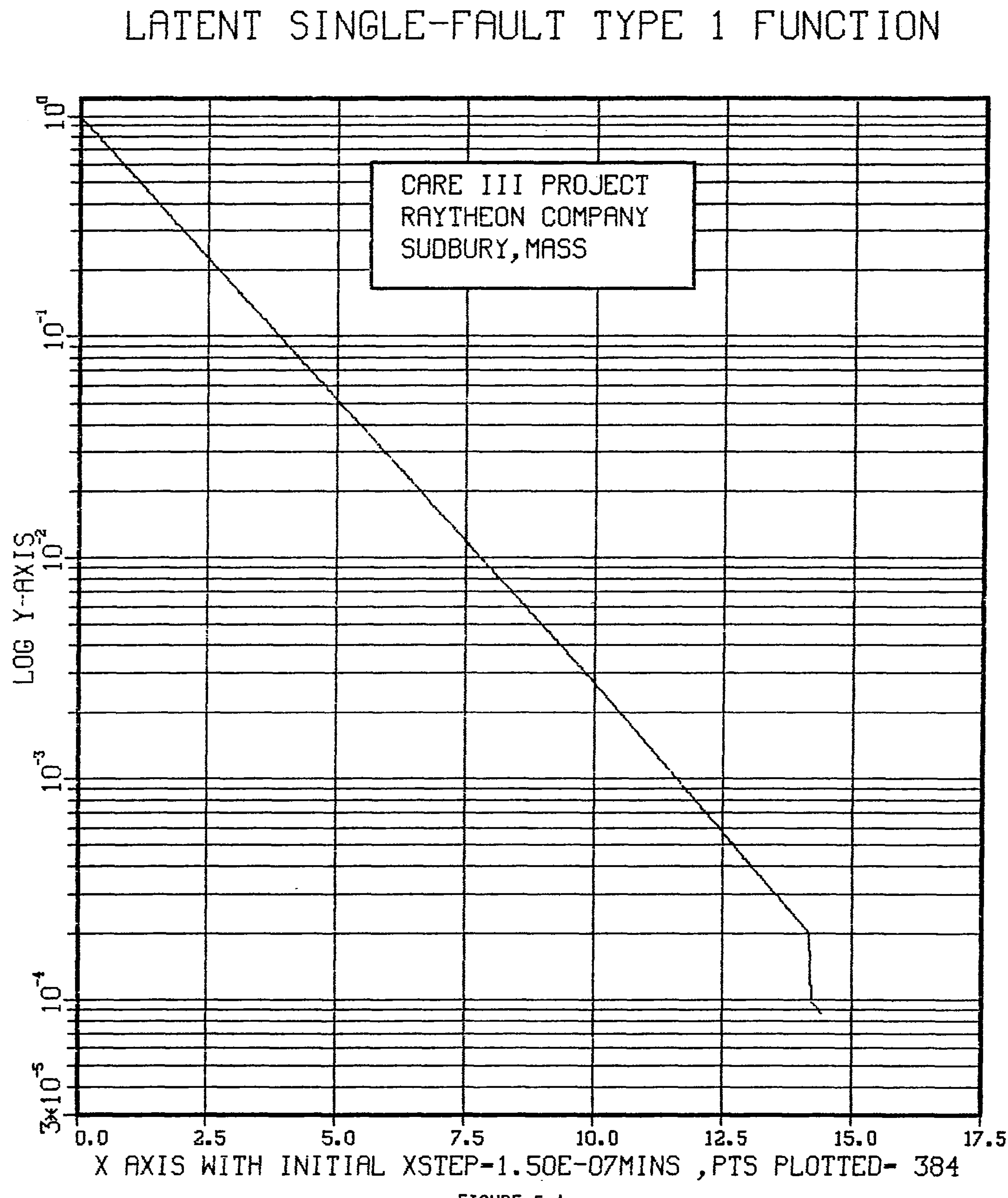

FIGURE 5c'
TEST CASE 3D'

(Function PL vs. Time-Mins., CARE III Model)

LATENT SINGLE-FAULT TYPE 1 FUNCTION

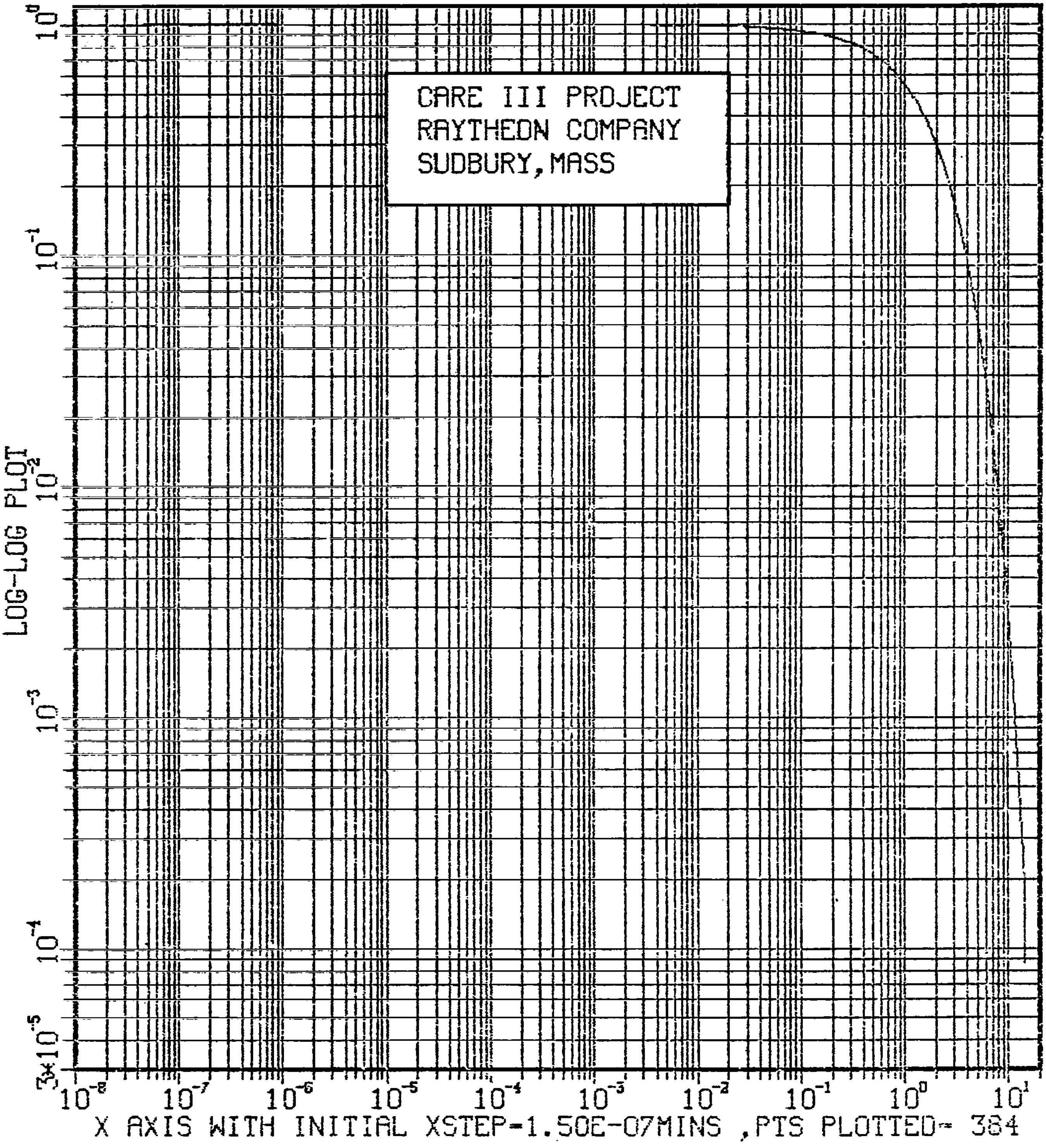

FIGURE 5c''
TEST CASE 3D'

(Function PL vs. Time-Mins., Markov Model)

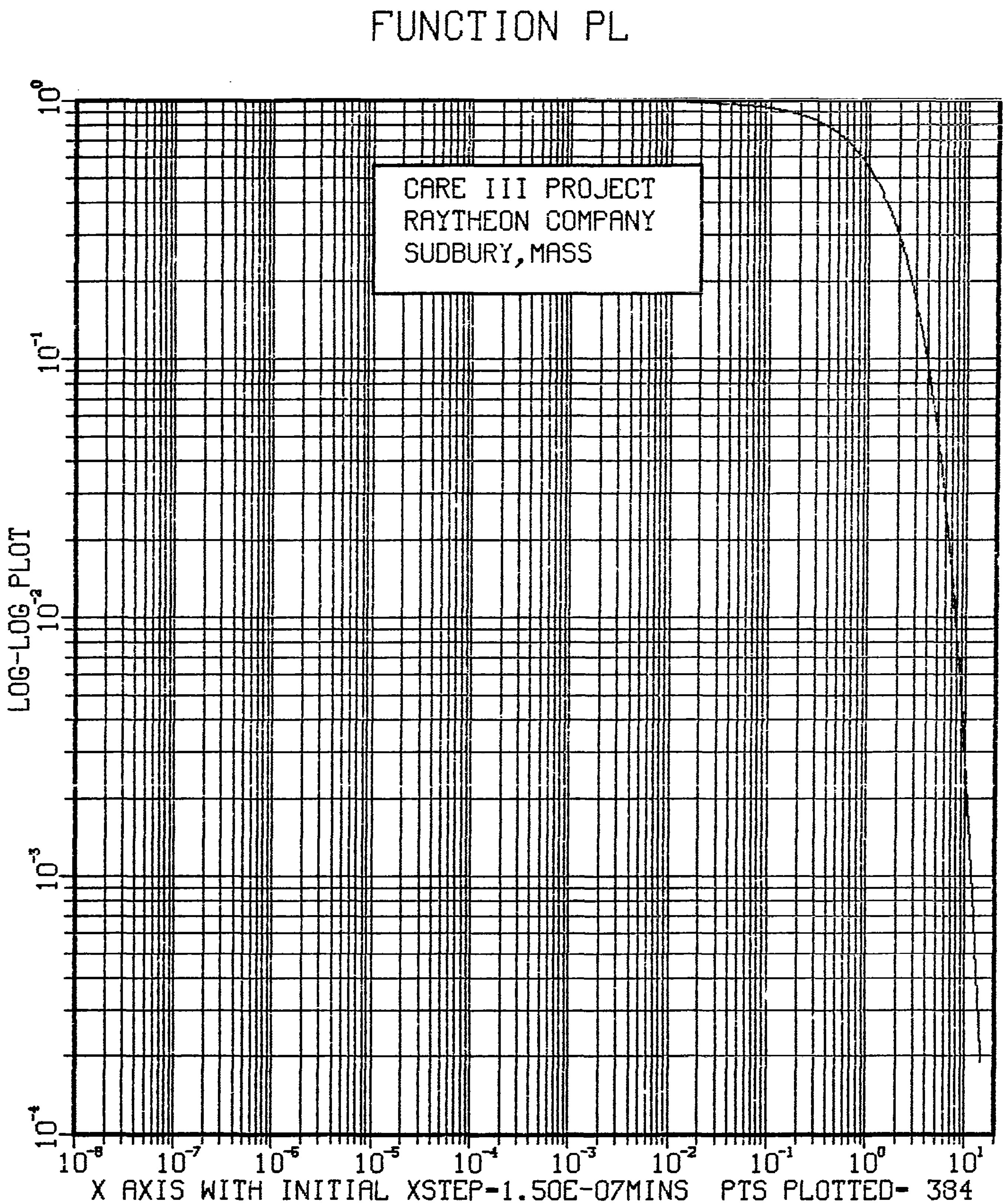

FIGURE 5c'''
TEST CASE 3D'

(Function PDF vs. Time-Mins., CARE III Model)

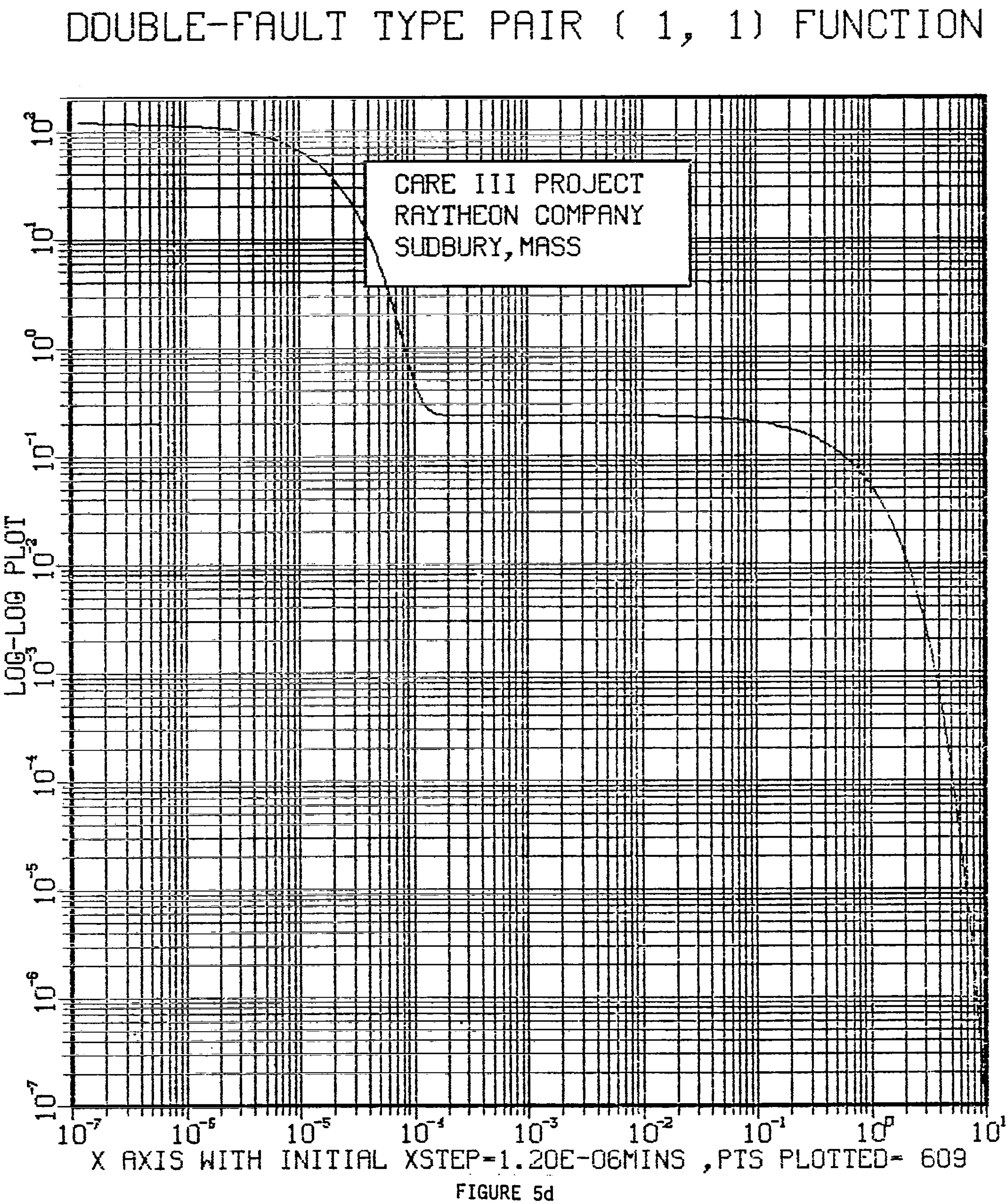

FIGURE 5d
TEST CASE 3D'

(Function Q SUM vs. Time-Mins., CARE III Model)

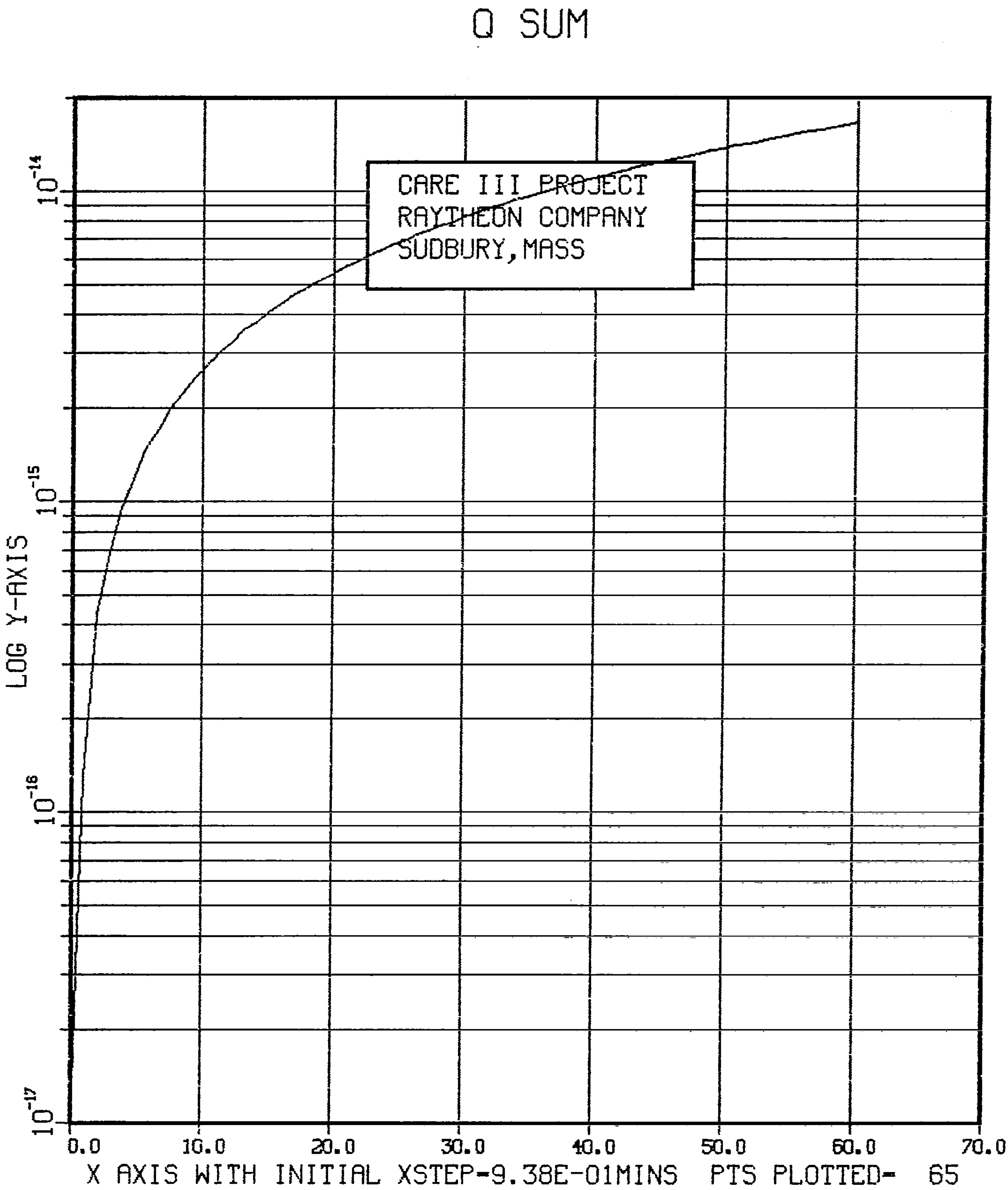

FIGURE 5e
TEST CASE 3D'

(Function P* SUM vs. Time-Mins., CARE III Model)

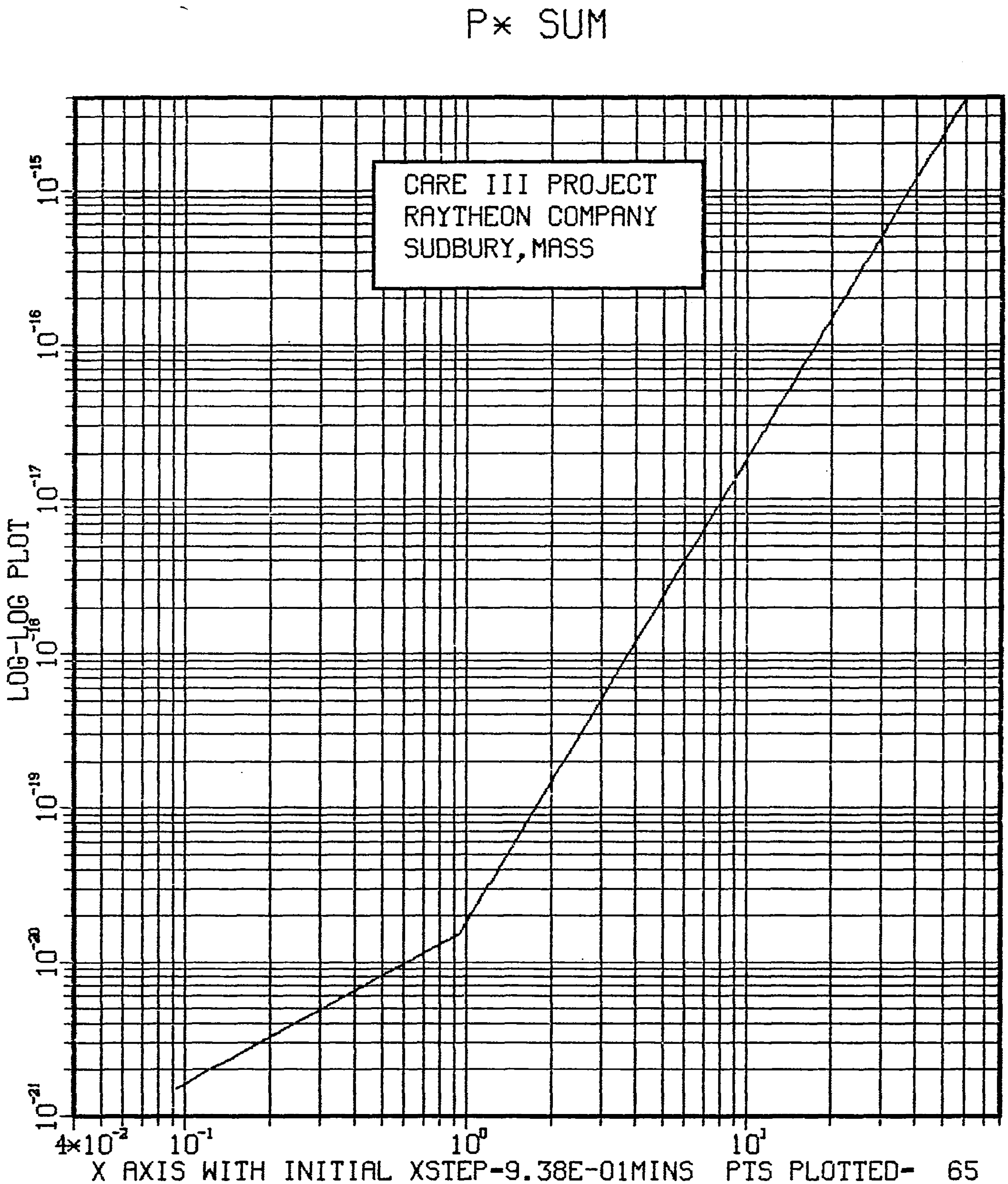

FIGURE 5f
TEST CASE 3D'

(Function Q + P* SUM vs. Time-Mins., CARE III Model)

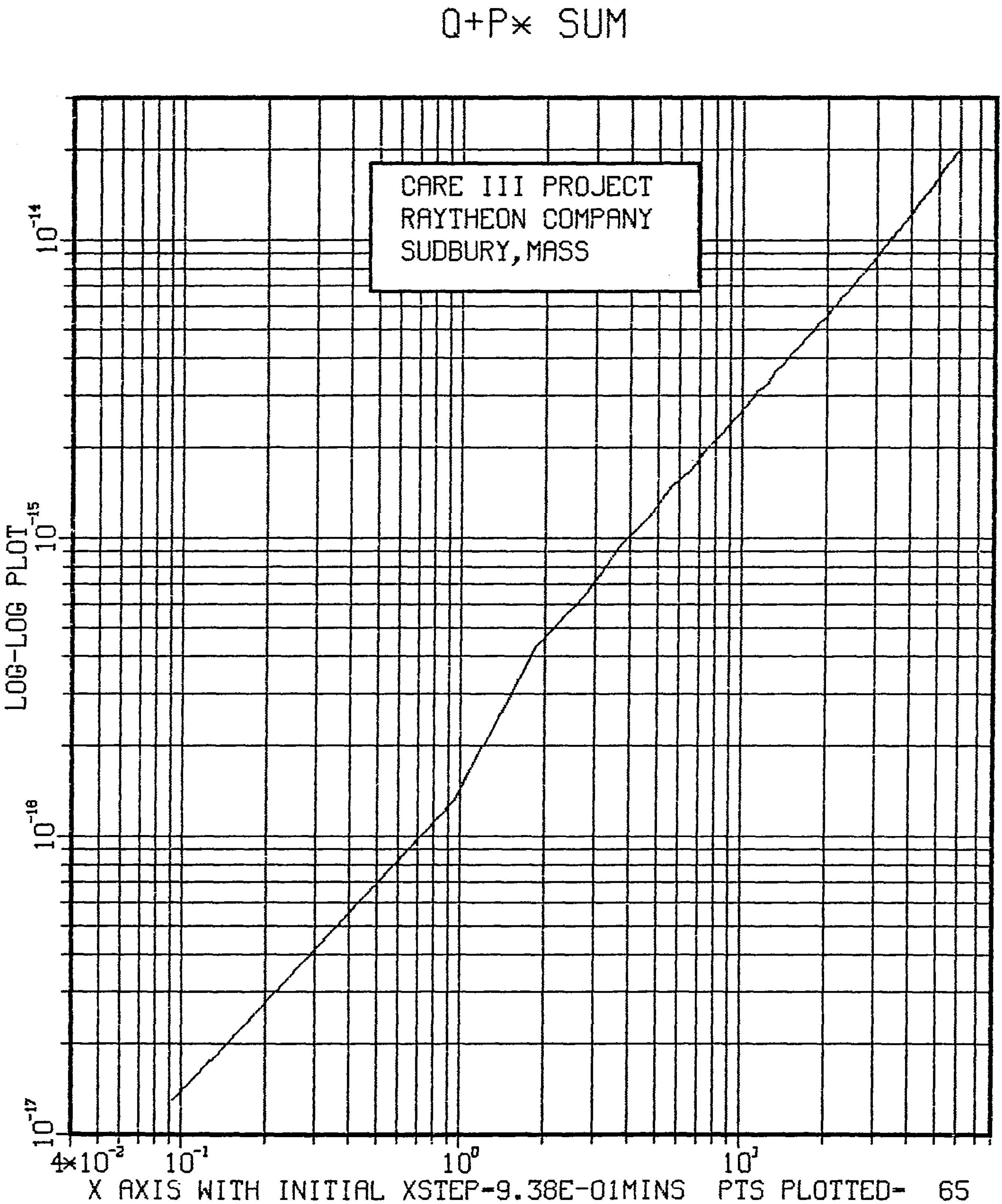

FIGURE 5g
TEST CASE 3D'

(Function PDF vs. Time-Mins., CARE III Model)

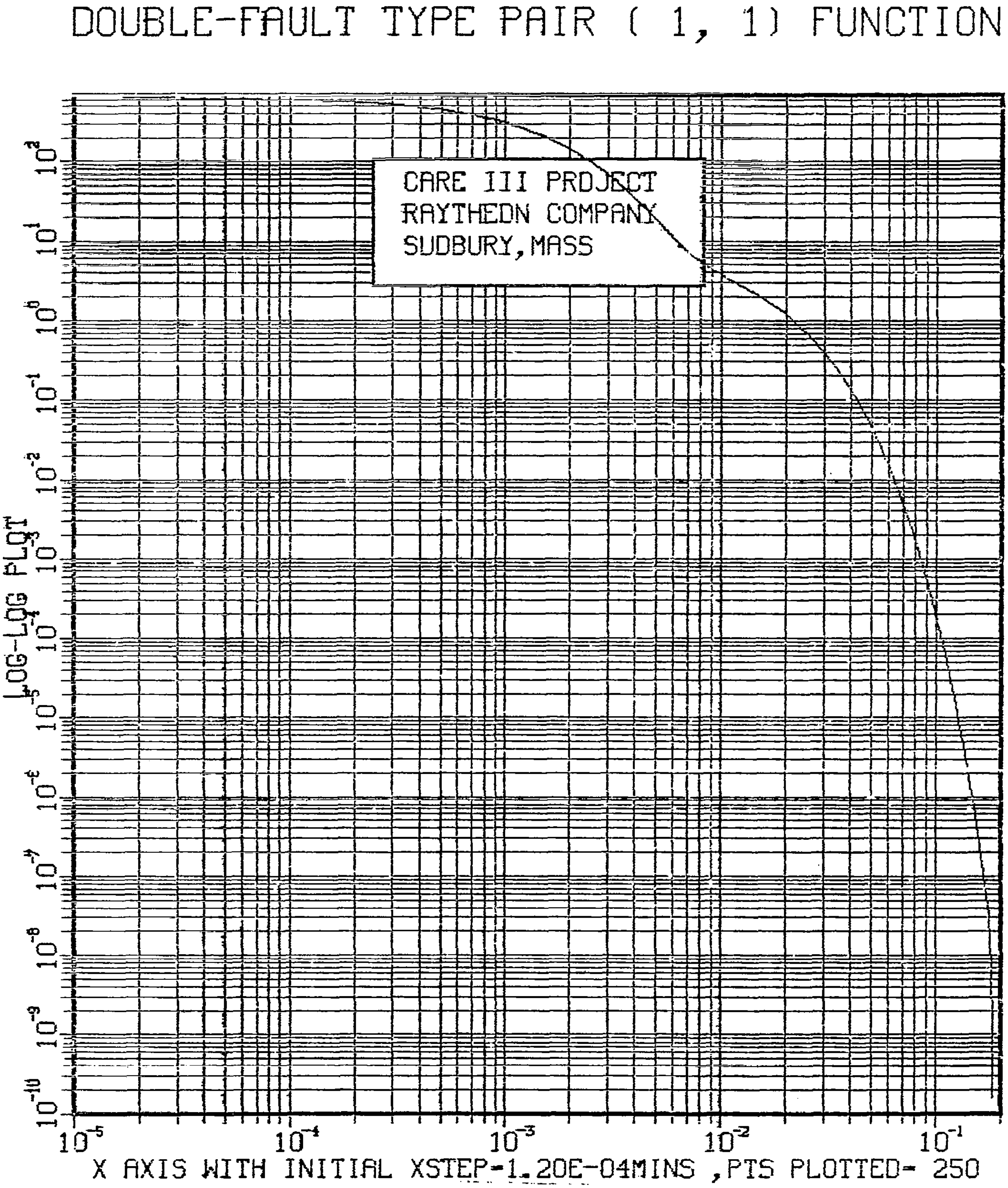

FIGURE 6a
TEST CASE 4A

(Function PDF vs. Time-Mins., Markov Model)

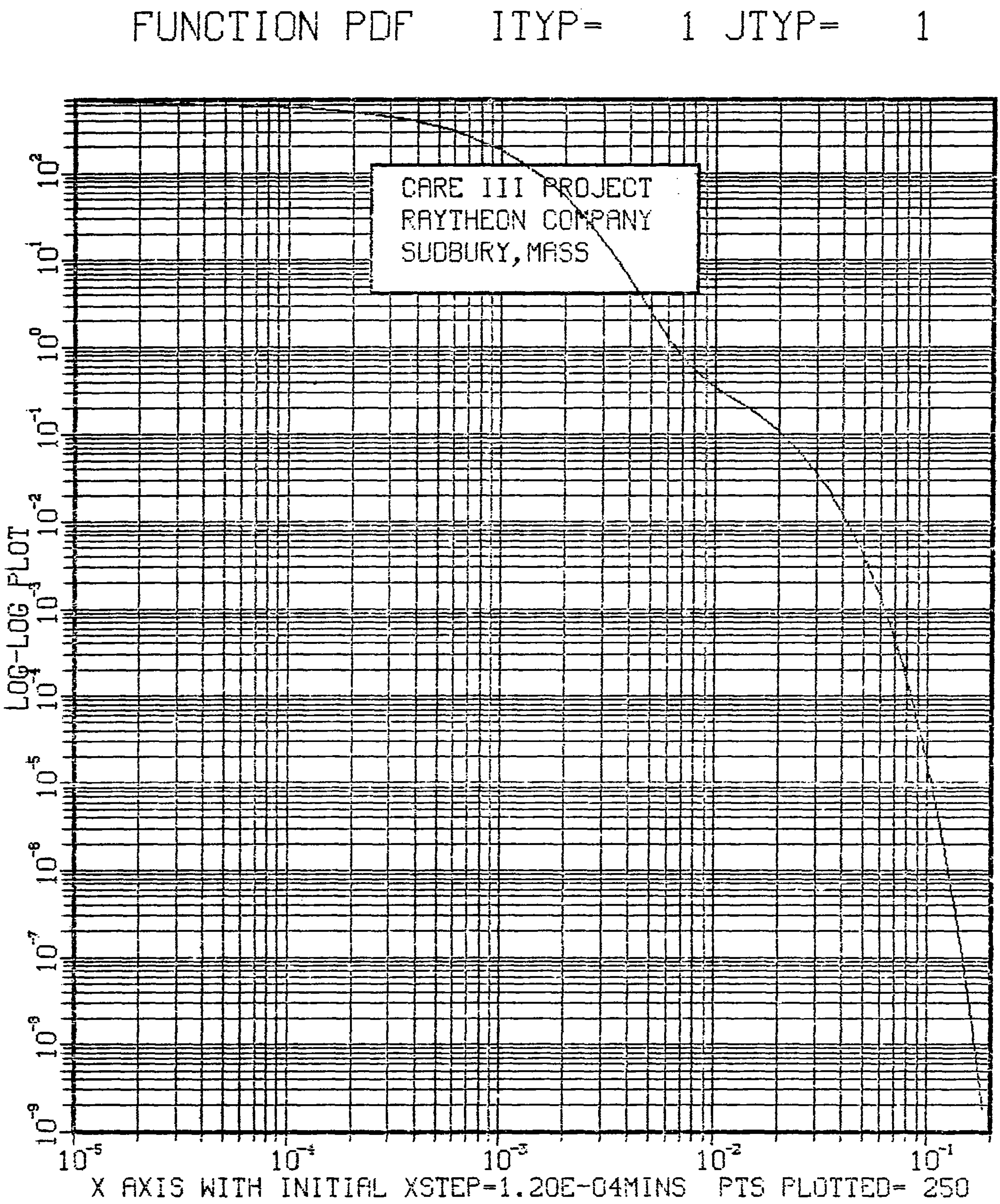

FIGURE 6a'
TEST CASE 4A

(Function PDF vs. Time-Mins., CARE III Model)

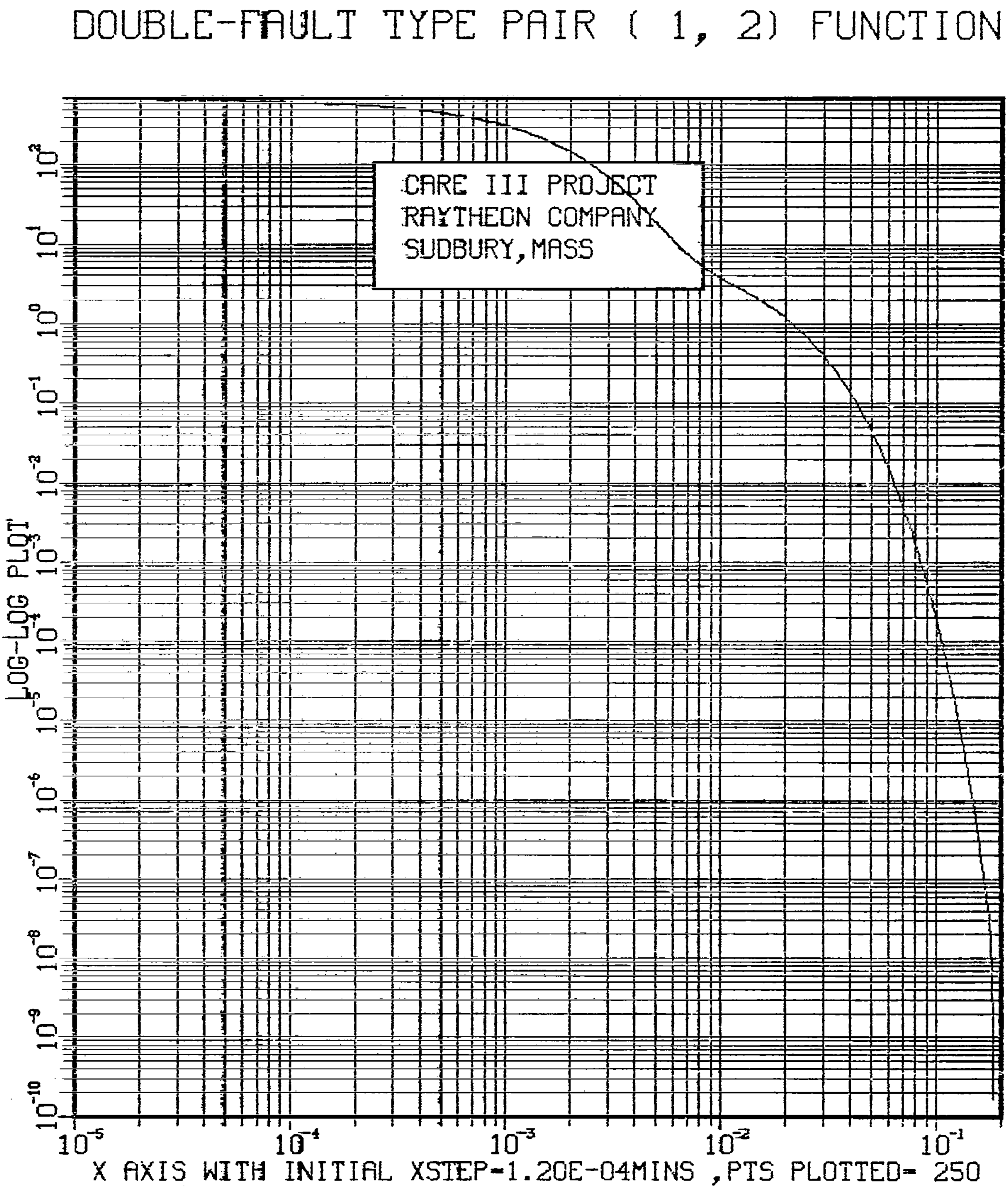

FIGURE 6b
TEST CASE 4A

(Function PDF vs. Time-Mins., Markov Model)

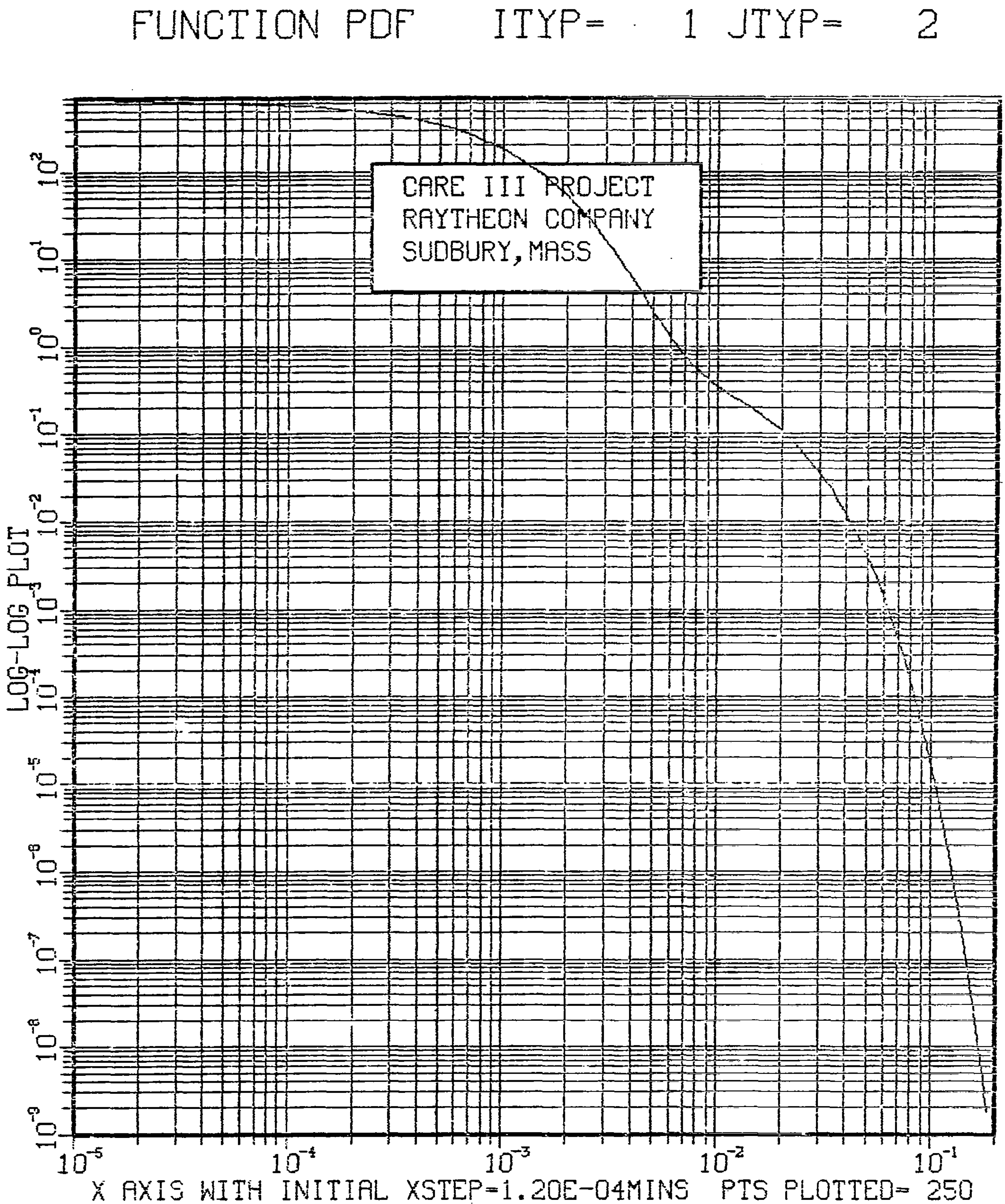

FIGURE 6b'
TEST CASE 4A

(Function PDF vs. Time-Mins., CARE III Model)

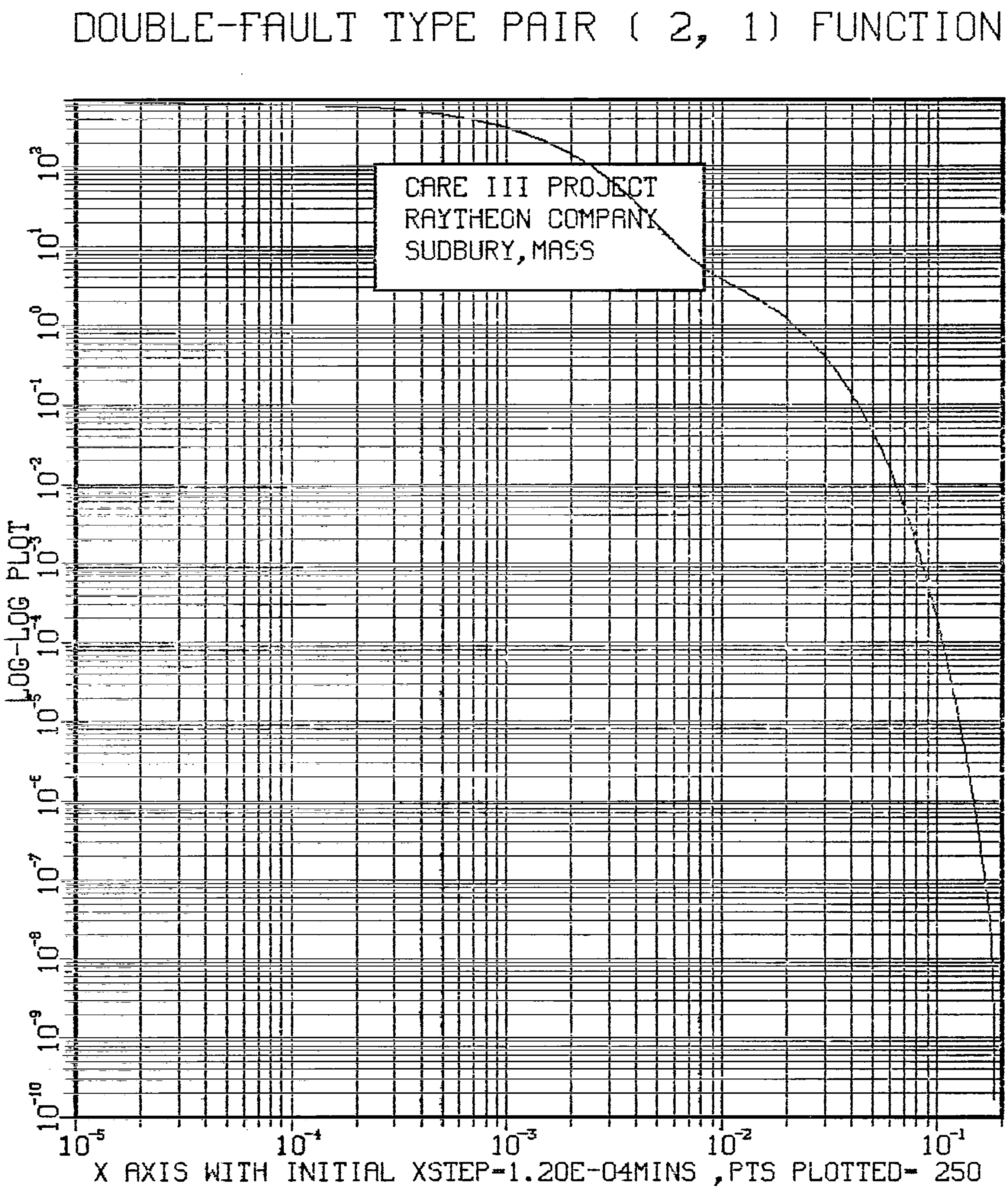

FIGURE 6c
TEST CASE 4A

(Function PDF vs. Time-Mins., Markov Model)

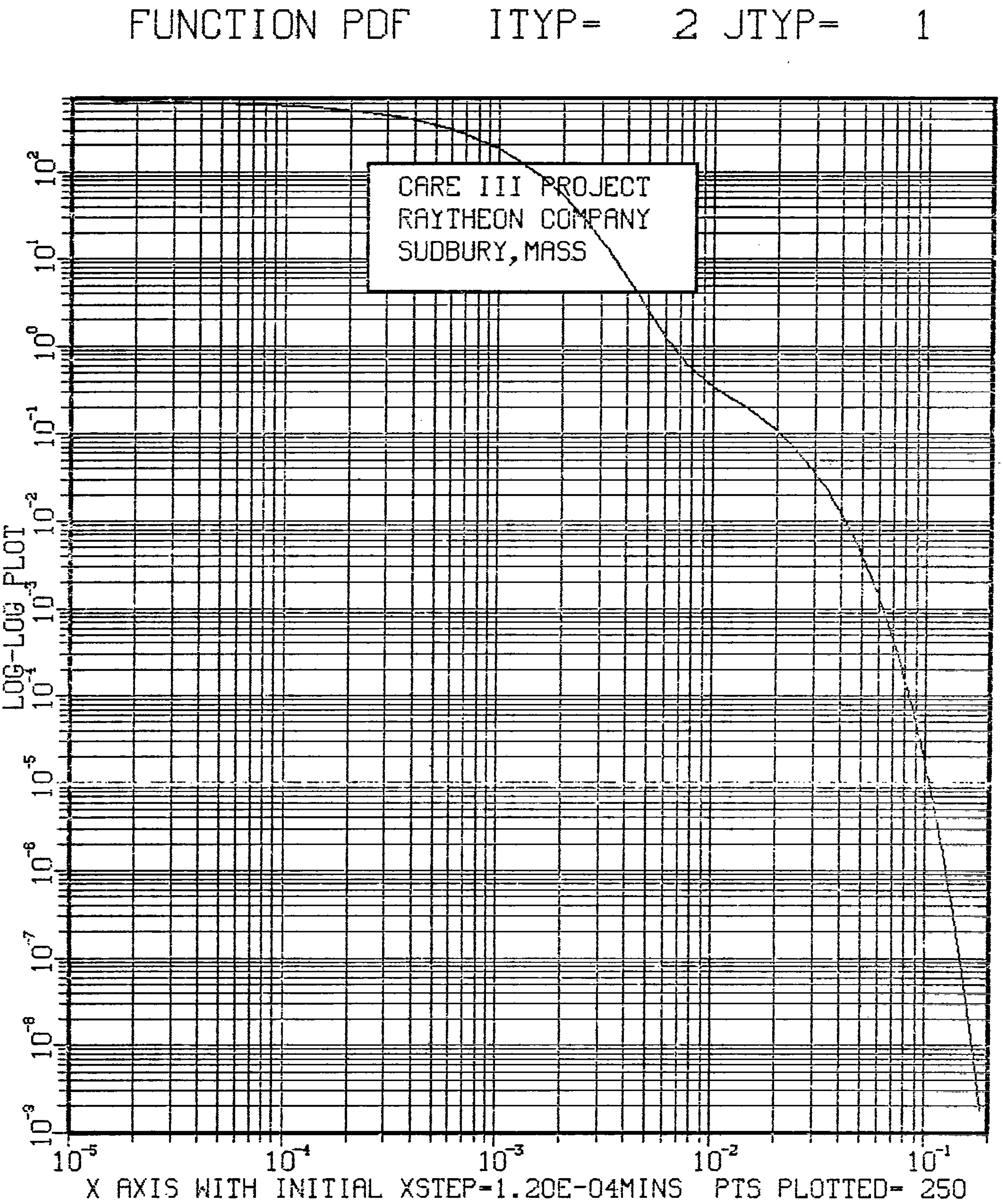

FIGURE 6c'
TEST CASE 4A

(Function PDF vs. Time-Mins., CARE III Model)

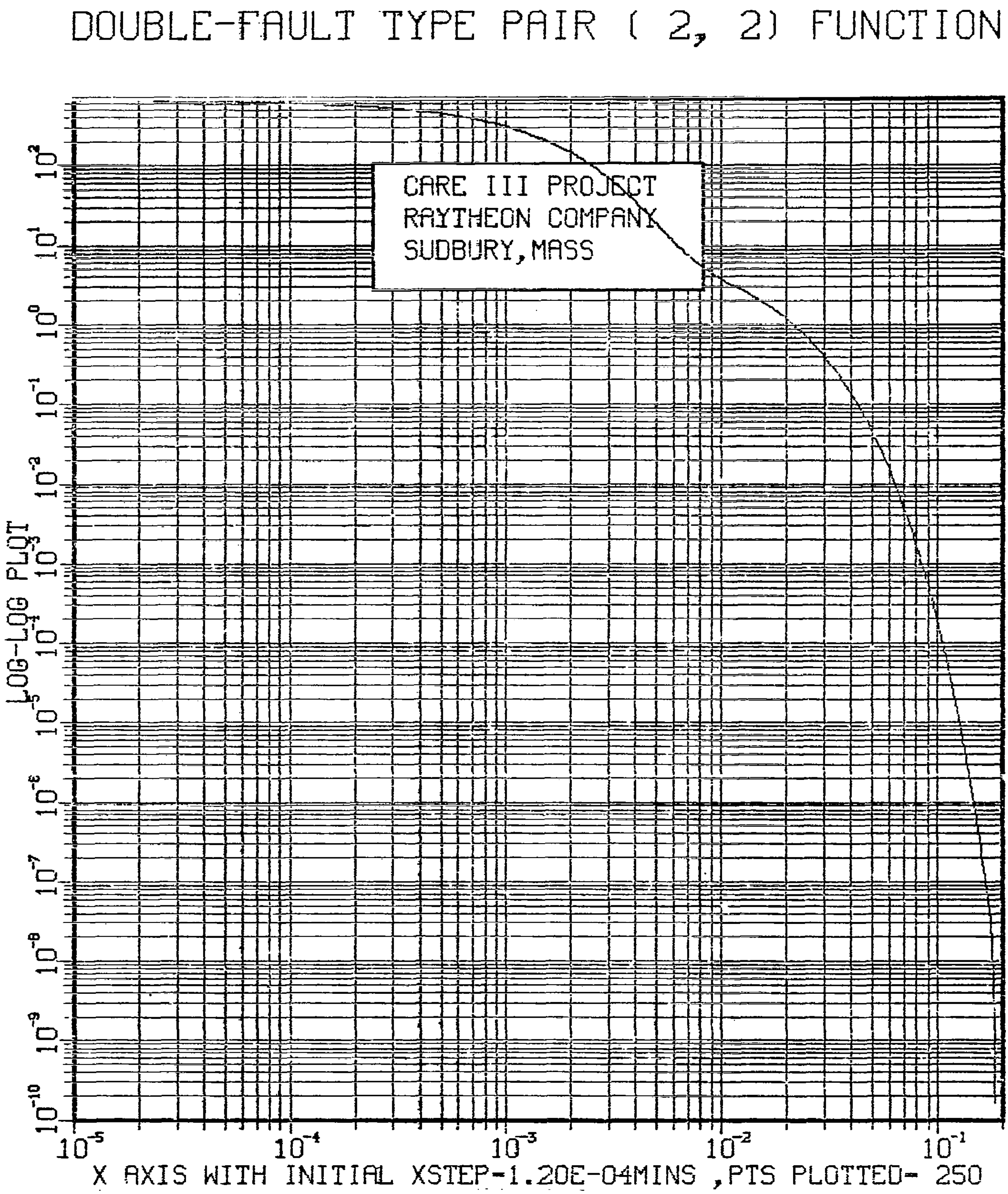

FIGURE 6d
TEST CASE 4A

(Function PDF vs. Time-Mins., Markov Model)

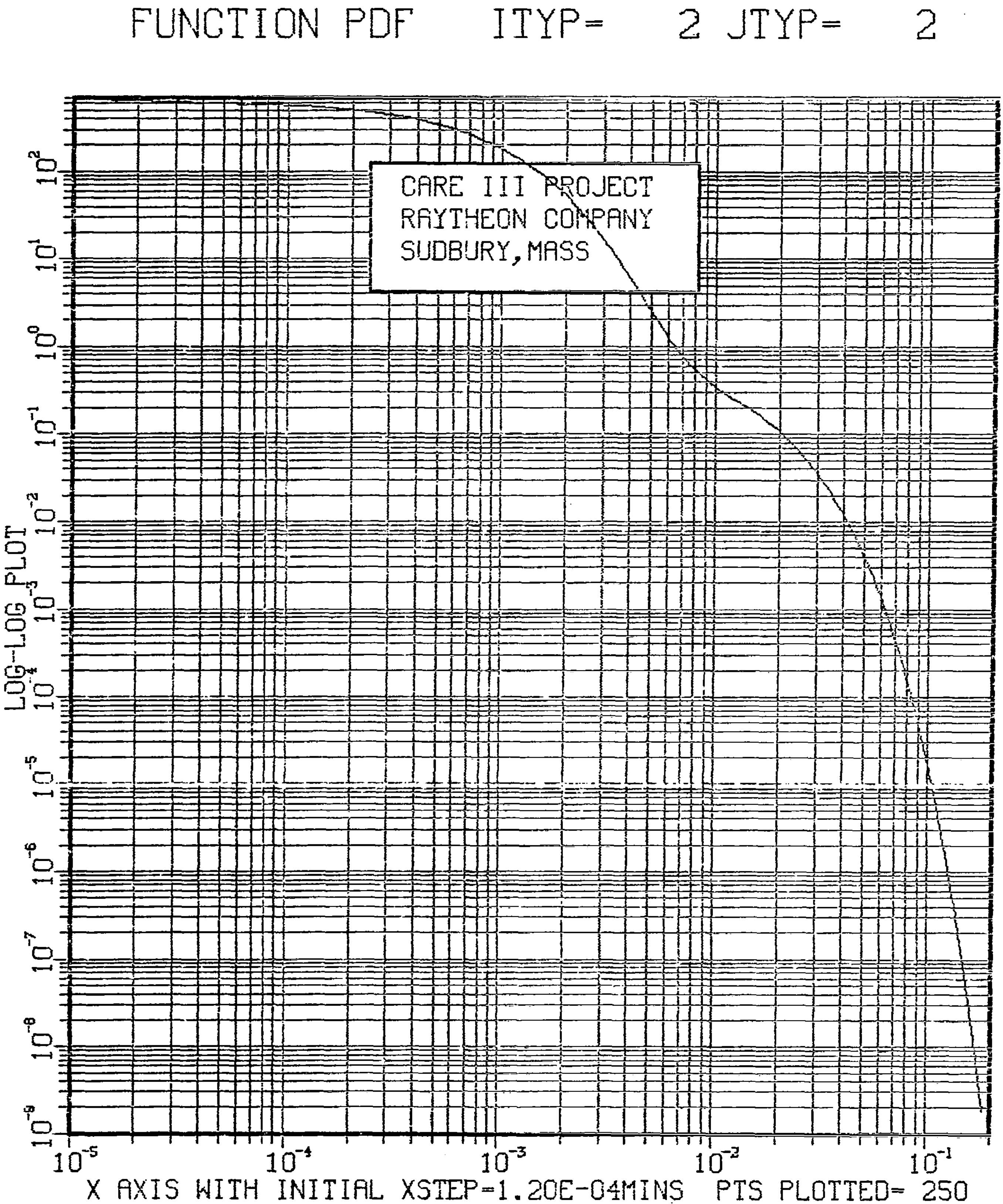

FIGURE 6d'
TEST CASE 4A

(Function PDF vs. Time-Mins., CARE III Model)

DOUBLE-FAULT TYPE PAIR (1, 1) FUNCTION

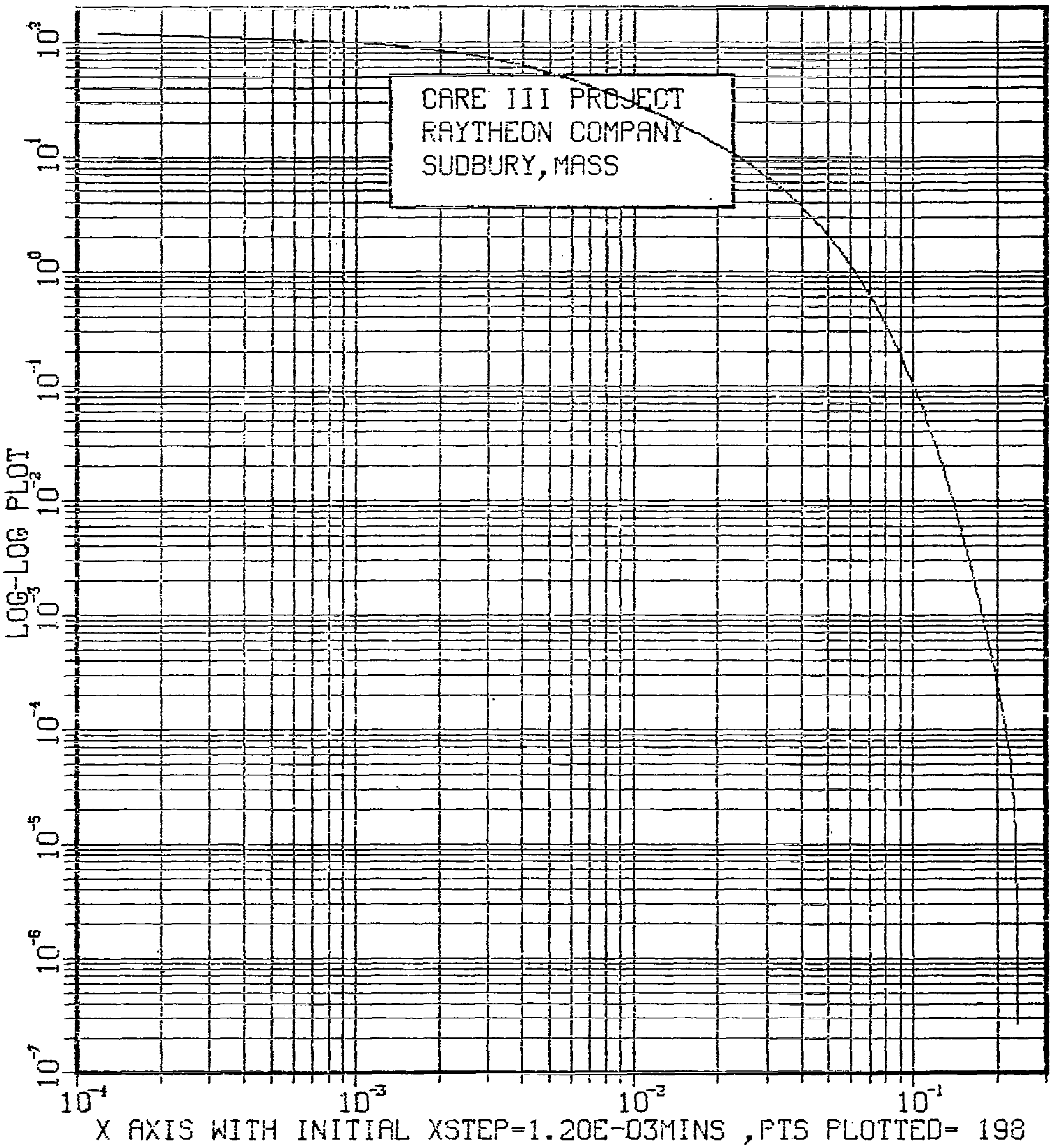

FIGURE 7a
TEST CASE 4B

(Function PDF vs. Time-Mins., Markov Model)

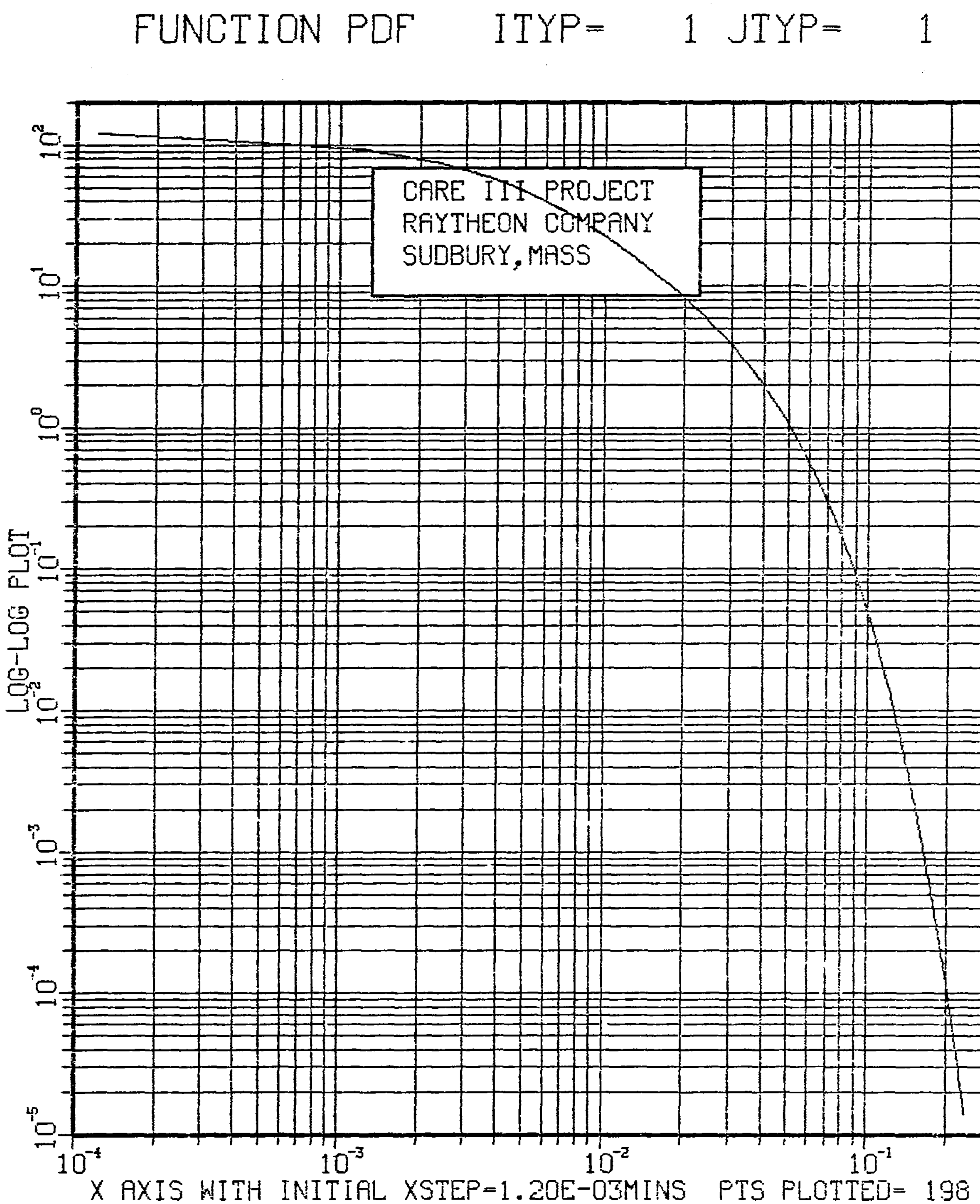

FIGURE 7a'
TEST CASE 4B

(Function PDF vs. Time-Mins., CARE III Model

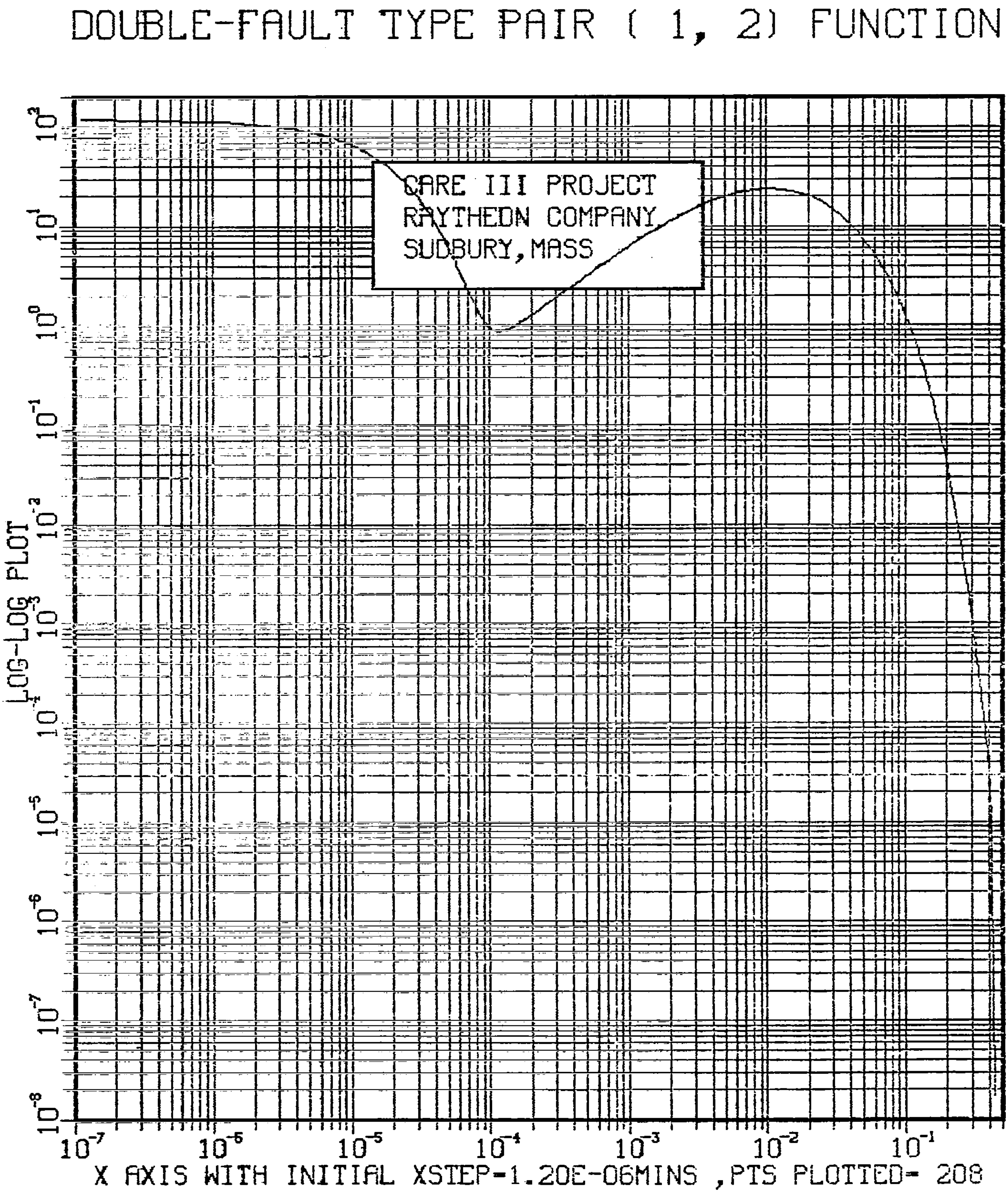

FIGURE 7b
TEST CASE 4B

(Function PDF vs. Time-Mins., CARE III Model)

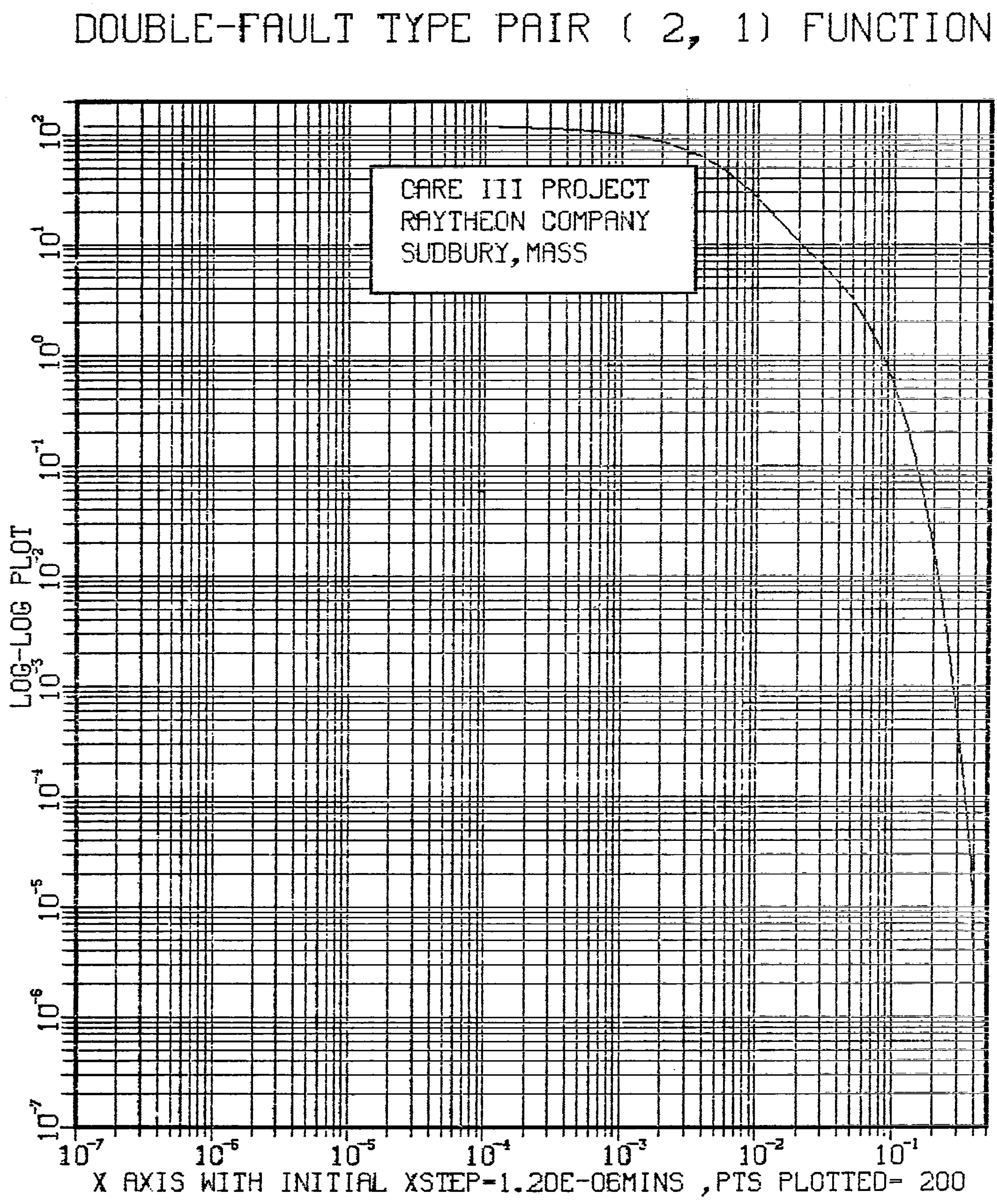

FIGURE 7c
TEST CASE 4B

(Function PDF vs. Time-Mins., Markov Model)

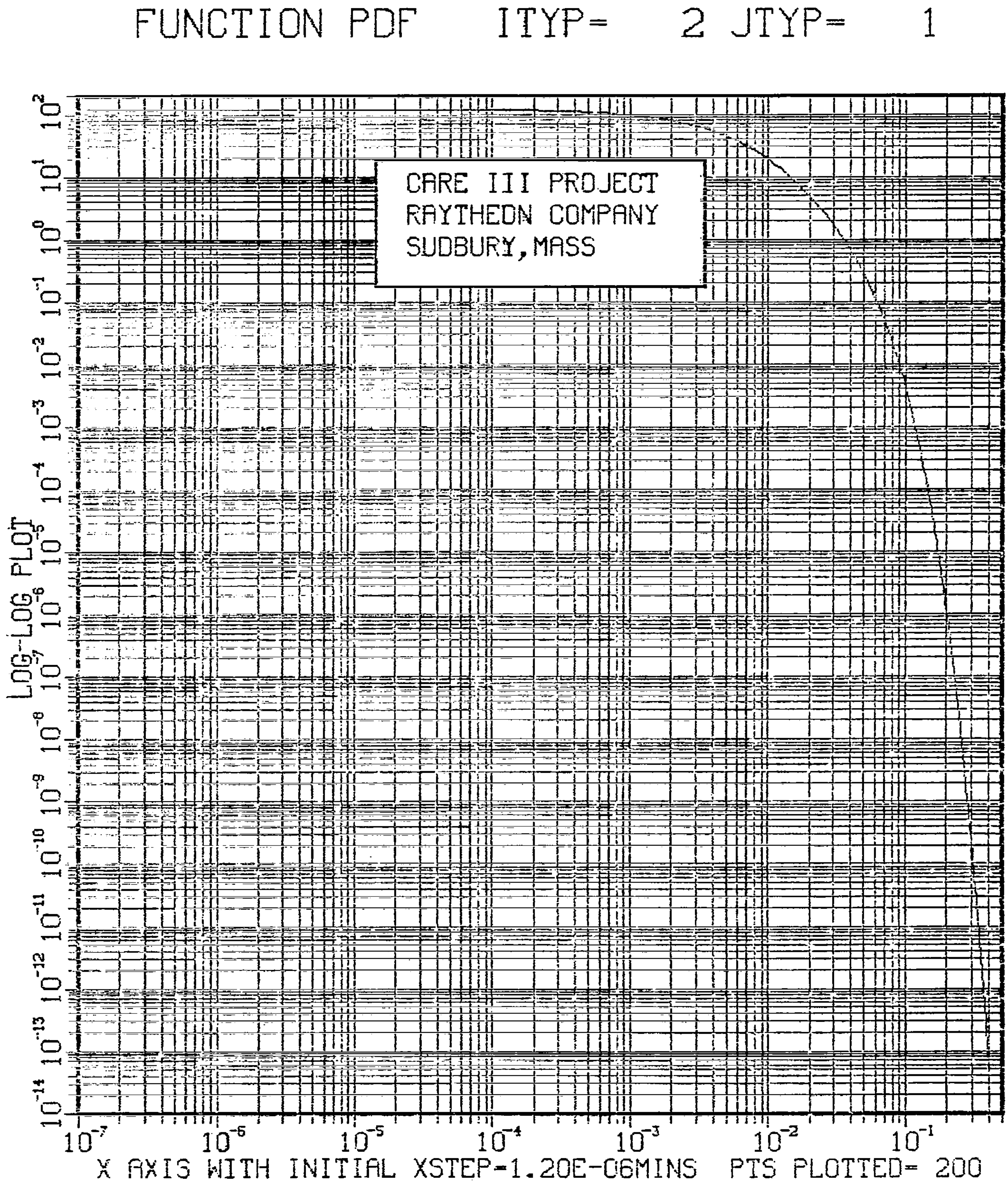

FIGURE 7c'
TEST CASE 4B

(Function PDF vs. Time-Mins., CARE III Model)

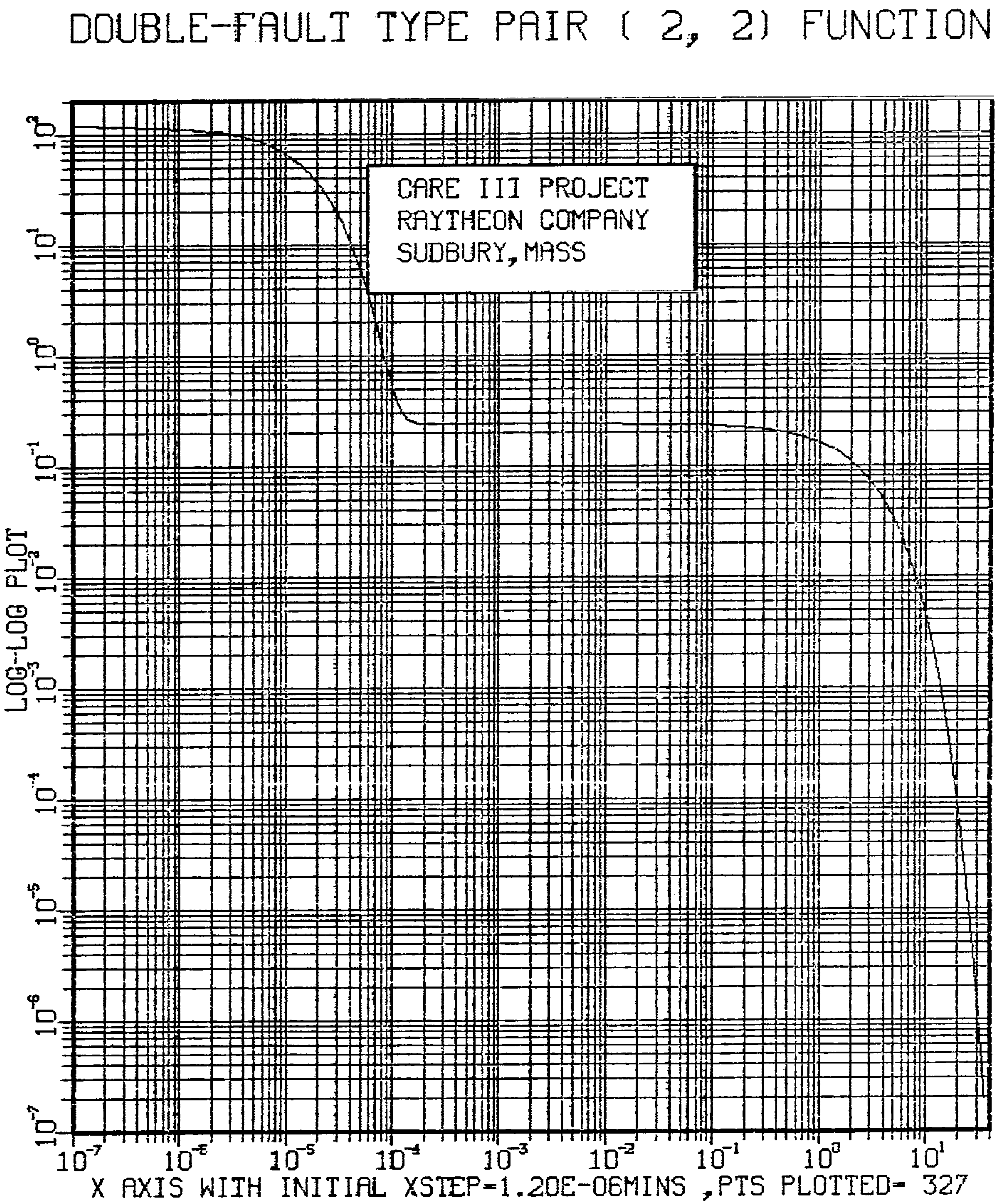

FIGURE 7d
TEST CASE 4B

(Function PDF vs. Time-Mins., Markov Model)

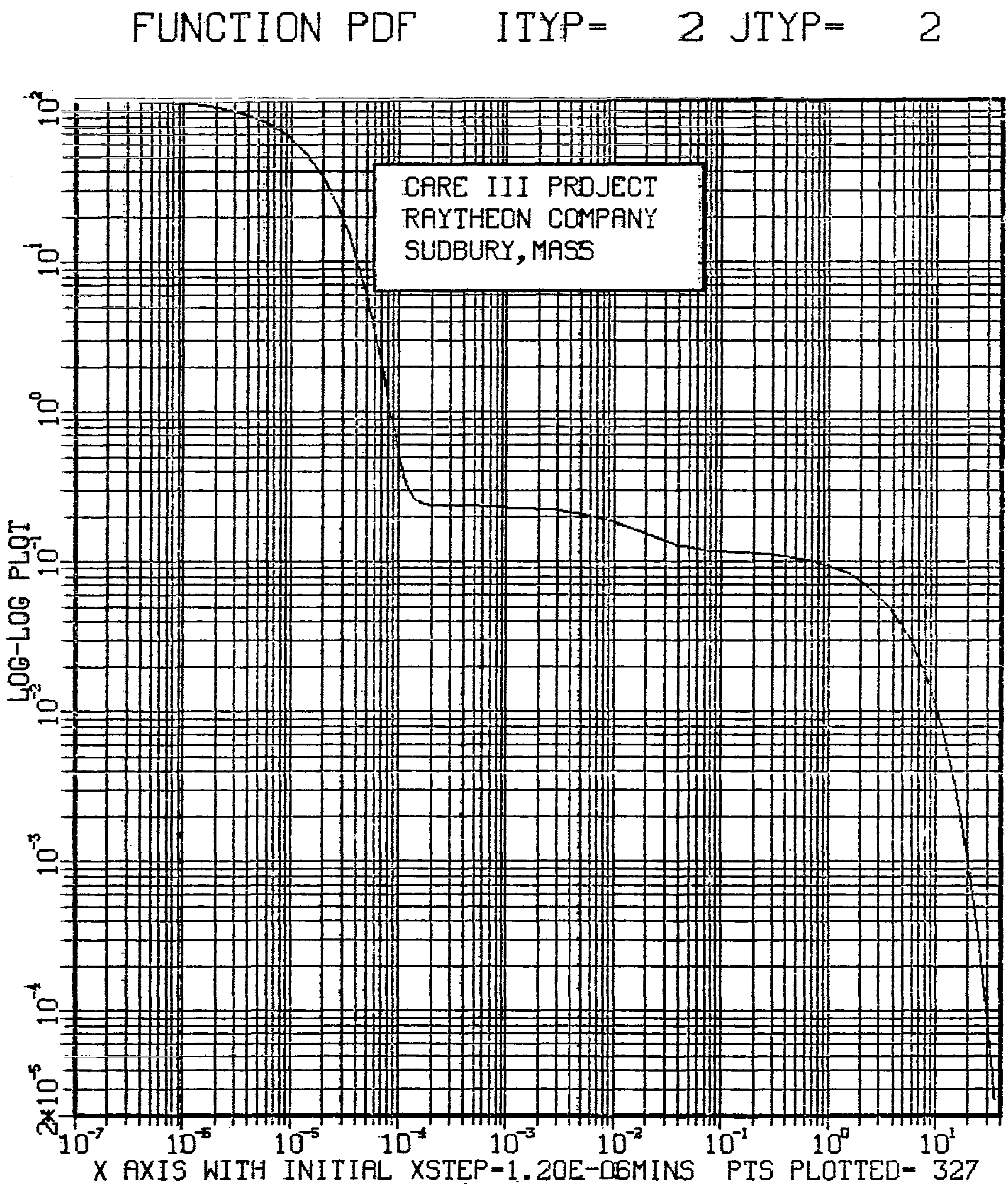

FIGURE 7d'
TEST CASE 4B

PERMANENT FAULT TEST CASES

Table 1

TEST CASE	α	β	δ	ρ	ε	P_B	P_A	C	tmax	λ	ω	RESULTS
* 1a	0	0	3.6E3	3.6E4	3.6E5	0	1	1	60 min.	10^{-5}	1	2.0572878738E-14 +
1b	0	0	3.6E3	3.6E4	3.6E5	0	1	.99	"	10^{-5}	1	3.5777939190E-07
1b'	0	0	3.6E3	3.6E4	3.6E5	0	1	.999	"	10^{-5}	1	3.5777957557E-08
* 1c	0	0	0	3.6E4	3.6E5	0	1	1	"	10^{-5}	1	2.2233692679E-14
1d	0	0	3.6E5	3.6E4	3.6E5	0	1	1	"	10^{-5}	1	4.8809076939E-15
* 1e	0	0	0	3.6E4	3.6E4	0	1	1	"	10^{-10}	0.5	2.3100883426E-13
* 1f	0	0	0	3.6E3	3.6E4	0	1	1	"	3.162 E-3	2	2.4820526862E-13
1g	0	0	3.6E5	3.6E3	3.6E4	0	1	1	"	10^{-5}	1	4.9061433149E-15
1h	0	0	3.6E3	3.6E3	3.6E4	0	1	1	"	10^{-5}	1	7.1097745431E-14
* 1h'	0	0	7.2E3	3.6E3	3.6E4	0	1	1	"	10^{-5}	1	6.4784031556E-14
* 1i	0	0	0	3.6E3	3.6E4	0	1	1	"	10^{-5}	1	1.8629094420E-13

NOTE: ρ and ε use constant rate functions; δ uses constant density for all but cases with *'s.

+ Test cases with corresponding plots.

TRANSIENT FAULT TEST CASES

Table 1 cont.

TEST CASE	α	β	δ	ρ	ε	P_A	P_B	C	tmax	λ		RESULTS
* 2a	3.6E4	0	0	3.6E4	3.6E5	1	0	1	60	10^{-5}	1	9.4311701023E-17
* 2a'	3.6E4	0	0	3.6E3	3.6E4	1	0	1	60	10^{-5}	1	2.1241235519E-19
* 2b	3.6E3	0	0	3.6E4	3.6E5	1	0	1	60	10^{-5}	1	7.3331992621E-16
* 2b'	3.6E3	0	0	3.6E3	3.6E5	1	0	1	60	10^{-5}	1	1.9607740169E-16
* 2c	3.6E2	0	0	3.6E4	3.6E5	1	0	1	60	10^{-5}	1	9.7825070511E-16 +
* 2c'	3.6E2	0	0	3.6E3	3.6E4	1	0	1	60	10^{-5}	1	1.5202140048E-15

INTERMITTANT FAULT TEST CASES

Table 1 cont.

	TEST CASE	α	β	δ	ρ	ε	P_A	P_B	C	tmax (min)	λ	ω	RESULTS	
	3a	3.6E3	3.6E3	0	3.6E3	3.6E4	1	0	1	60	10^{-5}	1	3.9885881529E-13	
	3a'	3.6E3	3.6E3	3.6E4	3.6E3	3.6E4	1	0	1	60	10^{-5}	1	4.9337584406E-14	
*	3a"	3.6E3	3.6E3	3.6E4	3.6E3	3.6E4	1	0	1	60	10^{-5}	1	5.2490555488E-14	
	3b	3.6E6	3.6E6	0	3.6E3	3.6E4	1	0	1	60	10^{-5}	1	4.6277452759E-13	
	3b'	3.6E6	3.6E6	3.6E4	3.6E3	3.6E4	1	0	1	60	10^{-5}	1	3.5945484343E-14	+
*	3b"	3.6E6	3.6E6	3.6E4	3.6E3	3.6E4	1	0	1	60	10^{-5}	1	3.6243046680E-14	
	3c	3.6E3	3.6E6	0	3.6E3	3.6E4	1	0	1	60	10^{-5}	1	1.8671106774E-13	+
	3c'	3.6E3	3.6E6	3.6E4	3.6E3	3.6E4	1	0	1	60	10^{-5}	1	1.2962063686E-14	
*	3c"	3.6E3	3.6E6	3.6E4	3.6E3	3.6E4	1	0	1	60	10^{-5}	1	2.0598920303E-14	
	3d	3.6E6	3.6E3	0	3.6E3	3.6E4	1	0	1	60	10^{-5}	1		
	3d'	3.6E6	3.6E3	3.6E4	3.6E3	3.6E4	1	0	1	60	10^{-3}	1	2.6004620396E-12	+
*	3d"	3.6E6	3.6E3	3.6E4	3.6E3	3.6E4	1	0	1	60	10^{-5}	1	2.6467251658E-12	

NOTE: Case 3d did not run to completion, due to an unacceptable amount of accumulated error.

DOUBLE FAULT TEST CASES

Table 2

TEST CASE	α	β	δ	ρ	ε	P_A	P_B	C	tmax (min)	λ	ω	FAULT TYPE
* 4A	3.6E3	3.6E3	3.6E3	3.6E4	3.6E5	1.0	0.0	1.0	60	10^{-5}	1	1 +
	3.6E3	3.6E3	3.6E3	3.6E4	3.6E5	0.9	0.1	1.0	60	10^{-5}	1	2
* 4B	3.6E3	3.6E3	0	3.6E3	3.6E4	1.0	0.0	1.0	60	10^{-5}	1	1 +
	3.6E6	3.6E3	0	3.6E3	3.6E4	1.0	0.0	1.0	60	10^{-5}	1	2
* 4C	3.6E3	3.6E3	0	3.6E3	3.6E4	1.0	0.0	1.0	60	10^{-5}	1	1
	3.6E3	3.6E6	0	3.6E3	3.6E4	1.0	0.0	1.0	60	10^{-5}	1	2
* 4D	3.6E3	3.6E3	0	3.6E3	3.6E4	1.0	0.0	1.0	60	10^{-5}	1	1
	3.6E6	3.6E6	0	3.6E3	3.6E4	1.0	0.0	1.0	60	10^{-5}	1	2
* 4E	3.6E3	0	0	3.6E3	3.6E4	1.0	0.0	1.0	60	10^{-5}	1	1
	3.6E3	0	0	3.6E3	3.6E4	1.0	0.0	1.0	60	10^{-5}	1	2
* 4F	3.6E3	0	0	3.6E3	3.6E4	1.0	0.0	1.0	60	10^{-5}	1	1
	3.6E3	3.6E3	0	3.6E3	3.6E4	1.0	0.0	1.0	60	10^{-5}	1	2

FTMP CASES

Table 3

α	β	CARE III PHASE III	CARE III PHASE II	CARE III PHASE I	DRAPER MODEL
10	1	1.1149		1.1161	1.124
10	10	1.2196		1.2041	1.207
10	100	1.1725	1.1682	1.1718	1.174
10	1000	1.123		1.1274	1.129
100	1	0.9331		1.0054	1.2073
100	10	1.9527		1.9072	1.924
100	100	1.664		1.6585	1.661
100	1000	1.215	1.2142	1.2181	1.220
1000	1	0.263	0.2847	0.4239	1.46
1000	10	3.387	3.3973	3.7975	4.22
1000	100	6.3614		6.1513	6.17
1000	1000	2.156		2.1198	2.12

Failure Probability (x 10-8)

NOTE: Not all cases were run for Phase II.

SIFT CASES

Table 4

TEST	N_1	N_2	λ_1	λ_2	δ	CARE III PHASE III	CARE III PHASE I	SRI
1	10	5	1.0E-4	1.0E-5	180	2.4858873E-8	2.4861769E-8	2.50 E-8
2	9	4	1.0E-4	1.0E-5	180	1.9880060E-8	1.9882421E-8	2.00 E-8
3	8	3	1.0E-4	1.0E-5	180	4.5428416E-8	4.5400324E-8	4.56 E-8
4	10	5	1.01E-4	1.1E-5	18000	2.5373536E-10	2.5115101E-10	2.55 E-10
5	9	4	1.01E-4	1.1E-5	18000	2.08176805E-10	2.0611656E-10	2.10 E-10
6	8	3	1.01E-4	1.1E-5	18000	3.6450809E-8	3.6412602E-8	3.65 E-8

APPENDIX 1

Source Listings
SFMODL
DFMODL

```
      PROGRAM SFMODL(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,PLFILE,
     1                 TAPE4=PLFILE)
C**********************************************************************
C  THIS PROGRAM CALCULATES INTERMEDIATE FUNCTIONS USED IN THE
C  THE SINGLE FAULT MODEL IN COVRGE.
C**********************************************************************
      COMMON// SFAR(1800),IDUBAR(40),EIGA(4,4),TIME(1800),
     1              ITITLE(6),JTITLE(6),PAR(1800,4)
      COMMON/FLTPM/ ALPHA,BETA,DELTA,RHO,EPSILON,PA,PB,C
      DIMENSION TBASR(4)
      LOGICAL PFFLAG,GENXPTS,ANOTHER,PDPFLG,PLTFLG,QFLAG
      COMMON/EIGCOM/ EIGSD(4,4,4),EIGWR(4)
      DATA ANOTHER/.FALSE./,TBASR/5HHRS  ,5HMINS ,5HSECS ,5HMSECS/
      REWIND 4
      PLTFLG = .FALSE.
      WRITE(6,9)
    9 FORMAT(3X,"TYPE 8 REAL NUMBERS SEPARATED BY COMMAS FOR ",
     1        /"        ALPHA,BETA,DELTA,RHO,EPSILON,PA,PB,C")
      READ(5,*) ALPHA,BETA,DELTA,RHO,EPSILON,PA,PB,C
      WRITE(6,59)
   59 FORMAT(/,3X,"TYPE AN INTEGER .LE.4 FOR TIME BASE AS IN CARE3")
      READ(5,*) ITBASE
   30 WRITE(6,31)
      QFLAG = .FALSE.
   31 FORMAT(/,3X,"ENTER FUNCTION TO BE CALCULATED AS: 'PA','PB','PNB'"
     1         ",'PL','PF','PDP',OR 'Q'")
      READ(5,32) FUNCTYP
   32 FORMAT(A3)
      IF(FUNCTYP.NE.3HQ  ) GO TO 33
      WRITE(6,34)
   34 FORMAT(/,3X,"ENTER FLIGHT TIME, NUMBER OF 'Q' STEPS(.LE.64)")
      READ(5,*) FT,NQSTPS
      QFLAG = .TRUE.
      PFFLAG = .FALSE.
      PDPFLG = .FALSE.
      GO TO 35
   33 WRITE(6,40)
   40 FORMAT(/,3X,"ENTER INITIAL STEP SIZE")
      READ(5,*) STEP
      STEPIN = STEP
      WRITE(6,50)
   50 FORMAT(/,3X,"ENTER DOUBLING ARRAY; DELIMIT WITH COMMAS.",/)
      READ(5,*) (IDUBAR(N),N=1,40)
      ITSTPS = 0
      DO 51 N = 1,40
   51 ITSTPS = IDUBAR(N) + ITSTPS
      ITSTPS = ITSTPS + 1
      ICOEF = 0
      PFFLAG = .FALSE.
      PDPFLG = .FALSE.
      PF = 0.0
      GENXPTS = .FALSE.
```

```
      IF(FUNCTYP.EQ.3HPA ) IFUNCTN = 1
      IF(FUNCTYP.EQ.3HPB ) IFUNCTN = 2
      IF(FUNCTYP.EQ.3HPNB) IFUNCTN = 3
      IF(FUNCTYP.EQ.3HPL ) IFUNCTN = 4
      IF(FUNCTYP.EQ.3HPF ) PFFLAG = .TRUE.
      IF(FUNCTYP.EQ.3HPDP) PDPFLG = .TRUE.
      INDEX=1
      IF(IFUNCTN.NE.0 .OR. PFFLAG .OR. PDPFLG) GO TO 56
      WRITE(6,55)
   55 FORMAT(/,3X,"UNRECOGNIZED FUNCTION")
      GO TO 30
   56 NSTPST=1
      NSTPF=IDUBAR(1)
      T=0.0
      IF(ANOTHER) GO TO 25
C
C  CHOOSE TIME BASE CONVERSION FACTOR
C
   35 TBCF=0.0
      TBASE=TBASR(ITBASE)
      IF(TBASE.EQ.5HHRS  ) TBCF=1.0
      IF(TBASE.EQ.5HMINS ) TBCF=60.0
      IF(TBASE.EQ.5HSECS ) TBCF=3600.0
      IF(TBASE.EQ.5HMSECS) TBCF=3.6E6
      IF(TBCF.NE.0.0) GO TO 17
      WRITE(6,23)  TBASE
   23 FORMAT(/"** ERROR INCORRECT TIME BASE = ",A5)
      STOP
   17 ALPHA=ALPHA/TBCF
      BETA=BETA/TBCF
      DELTA=DELTA/TBCF
      EPSILON=EPSILON/TBCF
      RHO=RHO/TBCF
C
C
C    EIGA(I,J) ARE THE ELEMENTS OF THE MATRIX WHICH REPRESENTS
C    THE CAREIII SINGLE FAULT MODEL
C
   18 EIGA(1,1) = -(ALPHA+RHO+PA*DELTA)
      EIGA(1,2) = BETA
      EIGA(1,3) = (1-PA)*C*EPSILON
      EIGA(1,4) = 0.0
      EIGA(2,1) = ALPHA
      EIGA(2,2) = -BETA
      EIGA(2,3) = 0.0
      EIGA(2,4) = (1.0-PB)*C*EPSILON
      EIGA(3,1) = RHO
      EIGA(3,2) = 0.0
      EIGA(3,3) = -(ALPHA+EPSILON)
      EIGA(3,4) = BETA
      EIGA(4,1) = 0.0
      EIGA(4,2) = 0.0
      EIGA(4,3) = ALPHA
      EIGA(4,4) = -(BETA+EPSILON)
      CALL EIGEN(EIGA)
      DO 20 I=1,4
      EIGWR(I) = - EIGWR(I)
   20 CONTINUE
```

```
25 IF(QFLAG) CALL SFQ(SFAR,TIME,FT,NQSTPS)
   IF(.NOT.PDPFLG) GO TO 22
   CALL PDP(SFAR,TIME,STEP,ITSTPS,IDUBAR)
   GO TO 75
22 IF(.NOT. PFFLAG) GO TO 60
   DO 58 I=NSTPST,NSTPF
          DO 57 L=1,4
                IF(EIGWR(L).EQ.0.0) GO TO 57
26               PF = (((EIGSD(3,L,1)+EIGSD(4,L,1))/EIGWR(L))
  1                   *(1.0-PREEXP(-EIGWR(L)*T)))+PF
57         CONTINUE
                TIME(I) = T
                T=T + STEP
                PF = (1.0-C)*EPSILON*PF
          SFAR(I) = PF
          PF = 0.0
58 CONTINUE
   GO TO 71
60 IF(QFLAG) GO TO 74
   DO 70 I = NSTPST,NSTPF
      DO 65 ICOEF = 1,4
         TRM1=EIGSD(ICOEF,1,1)*PREEXP(-(EIGWR(1)*T))
         TRM2=EIGSD(ICOEF,2,1)*PREEXP(-(EIGWR(2)*T))
         TRM3=EIGSD(ICOEF,3,1)*PREEXP(-(EIGWR(3)*T))
         TRM4=EIGSD(ICOEF,4,1)*PREEXP(-(EIGWR(4)*T))
         PAR(I,ICOEF)=TRM1+TRM2+TRM3+TRM4
         IF(PAR(I,ICOEF).LT.0.0) PAR(I,ICOEF) = 0.0
65    CONTINUE
         TIME(I)=T
         T=T+STEP
70 CONTINUE
71 CONTINUE
   INDEX=INDEX+1
   NSTPST=NSTPF+1
   NSTPF=NSTPST-1+IDUBAR(INDEX)
   STEP=STEP*2.
   IF(I.LE.ITSTPS) GO TO 25
   IF(PFFLAG) GO TO 74
   DO 73 I = 1,ITSTPS
      PAR(I,3) = PAR(I,1) + PAR(I,3) + PAR(I,4)
      PAR(I,4) = PAR(I,2) + PAR(I,3)
      SFAR(I)  = PAR(I,IFUNCTN)
73 CONTINUE
74 CONTINUE
75 WRITE(6,76)
76 FORMAT(/,3X,"WOULD YOU LIKE THIS ARRAY PRINTED AT THE TERMINAL ?")
   READ(5,101) IANSWER
   IF(IANSWER.NE.1HY) GO TO 91
   DO 90 I=1,ITSTPS
         WRITE(6,80) I,SFAR(I),TIME(I)
80       FORMAT(/,3X,I4,5X,E16.10,5X,1PE16.10)
90 CONTINUE
91 CONTINUE
```

```
      WRITE(6,95)
   95 FORMAT(/,3X,"WOULD YOU LIKE THIS FUNCTION PLOTTED ?")
      READ(5,101) IANSWER
      IF(IANSWER.EQ.1HY) PLTFLG = .TRUE.
      IF(IANSWER.NE.1HY) GO TO 102
      WRITE(6,96)
   96 FORMAT(/,3X,"ENTER PLOT TYPE: 1=LINEAR, 2=LOG, 3=BOTH")
      READ(5,*) LNORLG
      ENCODE(55,97,ITITLE) FUNCTYP
   97 FORMAT(9HFUNCTION ,A5,1H$)
      ENCODE(60,98,JTITLE) STEPIN,TBASE,ITSTPS
   98 FORMAT(26HX AXIS WITH INITIAL XSTEP= ,1PE8.2,A5,
     1       13H PTS PLOTTED=,I4,1H$)
      CALL CPLOT(SFAR,TIME,STEPIN,GENXPTS,ITSTPS,LNORLG,ITITLE,JTITLE)
  102 WRITE(6,100)
  100 FORMAT(/,3X,"DO YOU WISH TO COMPUTE ANOTHER FUNCTION?")
      READ(5,101) IANSWER
  101 FORMAT(A1)
      ANOTHER = .TRUE.
      IF(IANSWER.EQ.1HY) GO TO 30
      IF(PLTFLG) CALL DONEPL
      STOP
      END
      FUNCTION PREEXP(X)
C
      DATA REALMAX/1.0E+322/,REALMIN/1.0E-293/,EXPMAX/741.67/,
     1     EXPMIN/-675.82/
C
      IF (X.GT.EXPMIN .AND. X.LT.EXPMAX) GO TO 100
C  SET FUNCTION TO A VALUE VERY CLOSE TO 0.0 BUT NOT EQUAL TO 0.0
      IF (X.LE.EXPMIN) PREEXP = REALMIN
C  SET FUNCTION TO THE MAXIMUM VALUE THE CDC CAN HANDLE
      IF (X.GE.EXPMAX) PREEXP = REALMAX
      GO TO 200
  100 PREEXP = EXP(X)
  200 RETURN
      END
```

```
      SUBROUTINE PDP(PDPAR,TIME,STEP,ITSTPS,IDUBAR)
      COMMON/FLTPM/ ALPHA,BETA,DELTA,RHO,EPSILON,PA,PB,C
      DIMENSION PDPAR(1),TIME(1),IDUBAR(1)
      INDEX = 1
      NSTPST = 1
      NSTPF = IDUBAR(INDEX)
      T = 0.0
      CN1 = (2.0*ALPHA+RHO+EPSILON+PA*DELTA)/2.0
      CN2 = SQRT((RHO-EPSILON+PA*DELTA)**2.0
     1            +4.0*RHO*(1-PA)*C*EPSILON)/2.0
      RLAM1 = CN1 + CN2
      RLAM2 = CN1 - CN2
      RLAM3=EPSILON
      IF(PA.EQ.1.0) GO TO 20
      A1=(RLAM2-(ALPHA+RHO+PA*DELTA))/(RLAM2-RLAM1)
      A2=(RLAM1-(ALPHA+RHO+PA*DELTA))/(RLAM1-RLAM2)
      B1=((ALPHA+RHO+PA*DELTA-RLAM1)/((1.0-PA)*C*EPSILON))*A1
      B2=((ALPHA+RHO+PA*DELTA-RLAM2)/((1.0-PA)*C*EPSILON))*A2
      GO TO 40
   20 IF((RHO-EPSILON+DELTA).LT.0.0) GO TO 30
      A1 = 1.0
      A2 = 0.0
      B1 = -RHO/(RHO-EPSILON+DELTA)
      B2 = -B1
      GO TO 40
   30 A1 = 0.0
      A2 = 1.0
      B1 = RHO/(RHO-EPSILON+DELTA)
      B2 = -B1
   40 C1=(ALPHA/(EPSILON-RLAM1))*B1
      C2=(ALPHA/(EPSILON-RLAM2))*B2
      CONST1=PA*DELTA*A1+PA*C*EPSILON*B1+PB*C*EPSILON*C1
      CONST2=PA*DELTA*A2+PA*C*EPSILON*B2+PB*C*EPSILON*C2
      CONST3=PB*C*EPSILON*(C1+C2)
C
C    MAIN CALCULATIONS
C
   50 DO 100 I=NSTPST,NSTPF
      PDPAR(I)=CONST1*((1.0-PREEXP(-RLAM1*T))/RLAM1) + CONST2*((1.0-
     1       PREEXP(-RLAM2*T))/RLAM2) - CONST3*((1.0-PREEXP(-RLAM3*T))/
     2       RLAM3)
      TIME(I)=T
      T=T+STEP
  100 CONTINUE
      INDEX = INDEX + 1
      NSTPST=NSTPF+1
      NSTPF = NSTPST + IDUBAR(INDEX) - 1
      STEP = STEP * 2
      IF(I.LE.ITSTPS) GO TO 50
      RETURN
      END
```

```
      SUBROUTINE SFQ(QAR,TIME,FT,NSTEPS)
      COMMON/FLTPM/ ALPHA,BETA,DELTA,RHO,EPSILON,PA,PB,C
      COMMON/EIGCOM/EIGSD(4,4,4),EIGWR(4)
      DIMENSION QAR(1),TIME(1),PFAR(65),PFLAR(65)
      DATA RLAMBD1/1.0E-02/
C
C  THIS SUBROUTINE GENERATES FINAL 'Q' VALUES.
C
      STEP = FT/NSTEPS
      ITSTPS = NSTEPS + 1
C
C  SET INITIAL KNOWN CONDITIONS
C
      PFAR(1)=0.0
      PFLAR(1)=0.0
      QAR(1)=0.0
      T=STEP
      TIME(1)=0.0
C
C   CALCULATE COEFFICIENTS AND CONSTANTS
C
      CONST=(1.0-C)*EPSILON
      CD1= EIGSD(3,1,1)+EIGSD(4,1,1)
      CD2= EIGSD(3,2,1)+EIGSD(4,2,1)
      CD3= EIGSD(3,3,1)+EIGSD(4,3,1)
      CD4= EIGSD(3,4,1)+EIGSD(4,4,1)
      Z1 = EIGWR(1)-RLAMBD1
      Z2 = EIGWR(2)-RLAMBD1
      Z3 = EIGWR(3)-RLAMBD1
      Z4 = EIGWR(4)-RLAMBD1
C
C    MAIN CALCULATIONS
C
      DO 150 ITAU=2,ITSTPS
      X1 = -(EIGWR(1)*T)
      EIGL1T = 1.0 - PREEXP(X1)
      X2 = -(EIGWP(2)*T)
      EIGL2T = 1.0 - PREEXP(X2)
      X3 = -(EIGWR(3)*T)
      EIGL3T = 1.0 - PREEXP(X3)
      X4 = -(EIGWR(4)*T)
      EIGL4T = 1.0 - PREEXP(X4)
      X5 = -(Z1*T)
      EIGL5T = 1.0 - PREEXP(X5)
      X6 = -(Z2*T)
      EIGL6T = 1.0 - PREEXP(X6)
      X7 = -(Z3*T)
      EIGL7T = 1.0 - PREEXP(X7)
      X8 = -(Z4*T)
      EIGL8T = 1.0 - PREEXP(X8)
      PFAR(ITAU)=CONST*(((CD1*EIGL1T)/EIGWR(1)) + ((CD2*EIGL2T)/
     1           EIGWR(2))+ ((CD3*EIGL3T)/EIGWR(3)) + ((CD4*EIGL4T)/
     2            EIGWR(4)))
      PFLAR(ITAU)=CONST*(((CD1*EIGL5T)/Z1) + ((CD2*EIGL6T)/Z2) +
     1                  ((CD3*EIGL7T)/Z3) + ((CD4*EIGL8T)/Z4))
      QAR(ITAU)= PFAR(ITAU)-PFLAR(ITAU)*PREEXP(-RLAMBD1*T)
      TIME(ITAU)=T
      T=T+STEP
  150 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE EIGEN(EIGA)
C     ***********************************************
C       EIGEN CALLS TWO IMSL SUBROUTINES--EIGRF & LEQT2F
C       EIGRF COMPUTES EIGENVALES AND EIGENVECTORS OF
C       THE MATRIX EIGA(I,J)
C       LEQT2F SOLVES THE LINEAR SYSTEM AX=Y, WHERE
C       THE COLUMNS OF A ARE THE EIGENVECTORS OF EIGA
C       AND Y IS AN M BY N MATRIX WHOSE COLUMNS ARE THE INDIVI-
C       DUAL RIGHT HAND SIDES(NON-HOMOGENOUS TERMS)
C     ************************************************
      COMMON/EIGCOM/ EIGSD(4,4,4),EIGWR(4)
      DIMENSION WK(24),WKAREA(28),A(4,4),EIGC(4,4),EIGA(4,4)
      COMPLEX W(4),Z(4,4)
C       NEIG IS THE SIZE OF THE MATRIX WHOSE EIGENVALUES
C       WE ARE FINDING
C       24(WK DIMENSION) IS OBTAINED BY MULT. NEIG+2 BY NEIG
C       28(WKAREA DIMENSION) IS OBTAINED BY EVALUATING (NEIG**2)+3*NEIG
      NEIG=4
      M=4
      IDGT=4
      IJOB=2
      DO 50 J=1,4
      DO 50 I=1,4
      EIGC(I,J)=0.
   50 IF(I.EQ.J) EIGC(I,J)=1.
      CALL EIGRF(EIGA,NEIG,NEIG,IJOB,W,Z,NEIG,WK,IER)
      IF(WK(1).LT.1.0)PRINT(6,*) "EIGRF PERFORMED WELL,IER =",IER
      IF(WK(1).GE.1.0)PRINT(6,*)"EIGRF PERFORMED SATISFACTORY,IER =",EIR
      IF(WK(1).GT.100.0)PRINT(6,*) "EIGRF PERFORMED POORLY,IER =",EIR
      DO 60 I=1,NEIG
      EIGWR(I)=W(I)
      DO 70 J=1,NEIG
      EIGA(I,J)=Z(I,J)
      IF(ABS(EIGA(I,J)).GT.1.0E-10) GO TO 65
      EIGA(I,J)=0.0
 65   A(I,J)=EIGA(I,J)
 70   CONTINUE
 60   CONTINUE
      CALL LEQT2F(A,M,NEIG,NEIG,EIGC,IDGT,WKAREA,IER)
      DO 80 K=1,NEIG
      DO 80 I=1,NEIG
      DO 80 J=1,NEIG
C  EIGSD(I,J,K) ARE THE CONSTANTS USED TO CONSTRUCT THE
C  PROBABILITY FUNCTIONS P(I/K(T)) - PROBABILITY THAT THE SYSTEM
C  IS IN STATE I AFTER STARTING IN STATE K.
      EIGSD(I,J,K)=EIGA(I,J)*EIGC(J,K)
C
 80   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CPLOT(ARTOPLT,X,STEP,GENXPTS,NPTS,LNORLG,ITITLE,
     1     JTITLE)
C *********************************************************************
C  THIS ROUTINE WORKS WITH THE DISSPLA PLOTTING PACKAGE.
C *********************************************************************
C  NOTE - ARRAYS 'ARTOPLT' AND 'X' MUST BE OF THE SAME DIMENSION
C  IN THE CALLING ROUTINE.
      DIMENSION ARTOPLT(1),X(1),ITITLE(1),JTITLE(1)
      LOGICAL GENXPTS,AZROFLG,XZROFLG,AEPZFLG
      DATA XAXISMX/7.0/,YAXISMX/8.0/,XLEFT/2.25/,XRIGHT/4.75/,
     1      YLOWER/6.50/,YUPPER/7.50/,IFRAME/2/,XPOS/2.50/,YPOS1/7.25/,
     2      YPOS2/7.0/,YPOS3/6.75/,AZROFLG/.FALSE./,XZROFLG/.FALSE./,
     3      AEPZFLG/.FALSE./
      IF(NPTS.EQ.1) GO TO 25
      K = 0
      ITEST = 1
      IAXSTYP = LNORLG
      IF(.NOT.GENXPTS) GO TO 15
      DO 10 I=1,NPTS
   10 X(I)=(I-1)*STEP
   15 IF(X(1).EQ.0.0) XZROFLG = .TRUE.
      IF(XZROFLG) X(1) = STEP/10.0
      XMIN=X(1)
      XMAX=X(NPTS)
      YMIN=ARTOPLT(1)
      YMAX=ARTOPLT(1)
C
C*************TEST FOR ARRAY - PLOT TYPE COMPATABILITY*************
C
      DO 20 J=1,NPTS
         IF(X(J).LT.XMIN) XMIN = X(J)
         IF(ARTOPLT(J).EQ.0.0) K = K+1
         IF(ARTOPLT(J).EQ.0.0) AZROFLG = .TRUE.
         IF(.NOT.(AZROFLG.AND.((J.EQ.1).OR.(J.EQ.NPTS)))) GO TO 19
            IF(J.EQ.1) ARTOPLT(J) = ARTOPLT(J+1)/10.0
            IF(J.EQ.NPTS) ARTOPLT(J) = ARTOPLT(J-1)/10.0
            YMIN = ARTOPLT(J)
            AZROFLG = .FALSE.
   19    IF(ARTOPLT(J).LT.YMIN) YMIN = ARTOPLT(J)
         IF(ARTOPLT(J).GT.YMAX) YMAX = ARTOPLT(J)
   20 CONTINUE
      IF(K.NE.NPTS) GO TO 31
   25    WRITE(6,30)
   30    FORMAT(/4X,"ARRAY TO BE PLOTTED IS IDENTICALLY ZERO; NO PLOT",
     1   "GENERATED.")
      RETURN

   31 IF(.NOT.AZROFLG) GO TO 33
         WRITE(6,32)
   32    FORMAT(/4X,"DATA POINT ON LOG AXIS .LE. ZERO; LINEAR",
     1  " AXIS PLOT GENERATED.")
      IAXSTYP = 1
   33 YTEST = (0.05*YMAX) + (0.95*YMIN)
      XTEST = (0.05*XMAX) + (0.95*XMIN)
      NP = NPTS - 1
      DO 34 I = 1,NP
         IF((XTEST.GE.X(I)).AND.(XTEST.LE.X(I+1))) ITEST = I+1
   34 CONTINUE
```

```
      IF(ARTOPLT(ITEST).LE.YTEST) IAXSTYP = 4
      IF(IAXSTYP .EQ. 4) WRITE(6,35)
   35 FORMAT(/4X,"LINEAR PLOT INADEQUATE; LOG-LOG PLOT USED")
      IF(XZROFLG) WRITE(6,36) XMIN
   36 FORMAT(/4X,"ZERO VALUE IN TIME ARRAY; XMIN = ",E10.5," USED")
      IF(AEPZFLG) WRITE(6,37) YMIN
   37 FORMAT(/4X,"ZERO VALUED END-POINT IN DATA; YMIN = ",E10.5,
     1       " USED FOR LOG PLOTS.")
C
C***********DISSPLA INITIALIZATION***********************************
C
      CALL COMPRS
      CALL BGNPL(-1)
      CALL NOCHEK
      IF(AZROFLG) GO TO 38
      CALL ALGPLT(YMIN,YMAX,YAXISMX,YORIGL,YCYCL)
      CALL ALGPLT(XMIN,XMAX,XAXISMX,XORIGL,XCYCL)
   38 CALL AXSPLT(YMIN,YMAX,YAXISMX,YORIG,YSTEP,YAXIS)
      CALL AXSPLT(XMIN,XMAX,XAXISMX,XORIG,XSTEP,XAXIS)
C
C***********LINEAR PLOT ROUTINE**************************************
C
      IF(IAXSTYP.NE.1.AND.IAXSTYP.NE.3) GO TO 40
         CALL TITLE(ITITLE,100,JTITLE,100,"LINEAR Y-AXIS$",100,
     1               XAXISMX,YAXISMX)
         CALL FRAME
         CALL BLNK1(XLEFT,XRIGHT,YLOWER,YUPPER,IFRAME)
         CALL GRAPH(XORIG,XSTEP,YORIG,YSTEP)
         CALL GRID(1,2)
      GO TO 50
C
C***********SEMI-LOG PLOT ROUTINE************************************
C
   40 IF(IAXSTYP.NE.2.AND.IAXSTYP.NE.3) GO TO 45
         CALL TITLE(ITITLE,100,JTITLE,100,"LOG Y-AXIS$",100,
     1               XAXISMX,YAXISMX)
         CALL FRAME
         CALL BLNK1(XLEFT,XRIGHT,YLOWER,YUPPER,IFRAME)
         CALL YLOG(XORIG,XSTEP,YORIGL,YCYCL)
         CALL GRID(1,1)
      GO TO 50
C
C***********LOG-LOG PLOT ROUTINE*************************************
C
   45 IF(IAXSTYP.NE.4) GO TO 50
         CALL TITLE(ITITLE,100,JTITLE,100,"LOG-LOG PLOT$",100,
     1               XAXISMX,YAXISMX)
         CALL FRAME
         CALL BLNK1(XLEFT,XRIGHT,YLOWER,YUPPER,IFRAME)
         CALL LOGLOG(XORIGL,XCYCL,YORIGL,YCYCL)
         CALL GRID(1,1)
C
C*******************************************************************
C
```

```
   50 CONTINUE
C     CALL DASH
      CALL RESET("BLNK1")
      CALL MESSAG("CARE III PROJECT$",100,XPOS,YPOS1)
      CALL MESSAG("RAYTHEON COMPANY$",100,XPOS,YPOS2)
      CALL MESSAG("SUDBURY,MASS$",100,XPOS,YPOS3)
C     CALL RESET("DASH")
      CALL CURVE(X,ARTOPLT,NPTS,0)
      CALL ENDPL(0)
      IF(IAXSTYP.NE.3) GO TO 100
      IAXSTYP = 2
      GO TO 40
  100 CONTINUE
      RETURN
      END
```

```
      PROGRAM DFMODL (INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,
     1                PLFILE,TAPE4=PLFILE)
C
C*********************************************************************
C*                                                                   *
C*   THIS PROGRAM ANALYTICALLY CALCULATES THE INTERMEDIATE           *
C*   DOUBLE FAULT FUNCTION PDF, FOR COMPARISON WITH CARE III.        *
C*                                                                   *
C*********************************************************************
C
      COMMON// DFAR(1800),IDUBAR(40),TIME(1800),ITITLE(6),JTITLE(6)
      LOGICAL GENXPTS,ANOTHER
      COMMON/DEIGCOM/DEIGSD(3,3,3),DEIGWR(3),DEIGA(3,3),NEIGD,
     1               DEIGIC(3)
      DIMENSION FLTPM(9,2),EVALDF(4,2),COEFDF(4,4,2),TBASR(4)
      DATA TBASR/5HHRS  ,5HMINS ,5HSECS ,5HMSECS/,ANOTHER/.F./
C
C                      INPUT PARAMETERS
C
C    THE FAULT TYPE PARAMETERS ARE STORED IN THE ARRAY: FLTPM(I,L).
C
C          I   PARAMETER
C
C          -------------
C          1 * ALPHA
C          2 * BETA
C          3 * DELTA
C          4 * RHO
C          5 * EPSILON
C          6 * PA
C          7 * PB
C          8 * C
C          9 * LAMBDA
C
C          L   PARAMETER
C
C          -------------
C          1 * FIRST FAULT TYPE
C          2 * SECOND FAULT TYPE
C
       REWIND 4
      WRITE(6,10)
   10 FORMAT(3X,/,"ENTER FIRST FAULT TYPE PARAMETERS AS FOLLOWS: ",/,
     16X,"ALPHA1,BETA1,DELTA1,RHO1,EPSILON1,PA1,PB1,C1")
      READ(5,*) (FLTPM(I,1),I=1,8)
      WRITE(6,15)
   15 FORMAT(3X,/,"ENTER SECOND FAULT TYPE PARAMETERS AS FOLLOWS: ",/,
     16X,"ALPHA2,BETA2,DELTA2,RHO2,EPSILON2,PA2,PB2,C2")
      READ(5,*) (FLTPM(I,2),I=1,8)
      WRITE(6,20)
   20 FORMAT(/,3X,"TYPE AN INTEGER .LE.4 FOR TIME BASE AS IN CARE3")
      READ(5,*) ITBASE
```

```
   25 WRITE(6,30)
   30 FORMAT(/,3X,"ENTER ITYP,JTYP.")
      READ(5,*) ITP,JTP
      WRITE(6,40)
   40 FORMAT(/,3X,"TYPE AN INTEGER .LE. 1799 FOR NUMBER OF STEPS")
      READ(5,*) ITSTPS
      WRITE(6,45)
   45 FORMAT(/,3X,"ENTER INITIAL STEP SIZE")
      READ(5,*) STEP
      STEPIN = STEP
      WRITE(6,50)
   50 FORMAT(/,3X,"ENTER DOUBLING ARRAY; DELIMIT WITH COMMAS.",/)
      READ(5,*) (IDUBAR(N),N=1,40)
      ITSTPS = ITSTPS + 1
      GENXPTS = .FALSE.
      INDEX=1
   60 NSTPST=1
      NSTPF=IDUBAR(1)
      T=0.0
C
C  CHOOSE TIME BASE CONVERSION FACTOR
C
      IF(ANOTHER) GO TO 76
      TBCF=0.0
      TBASE=TBASR(ITBASE)
      IF(TBASE.EQ.5HHRS   ) TBCF=1.0
      IF(TBASE.EQ.5HMINS  ) TBCF=60.0
      IF(TBASE.EQ.5HSECS  ) TBCF=3600.0
      IF(TBASE.EQ.5HMSECS) TBCF=3.6E6
      IF(TBCF.NE.0.0) GO TO 69
      WRITE(6,65)  TBASE
   65 FORMAT(/"** ERROR INCORRECT TIME BASE = ",A5)
      STOP
   69 DO 75 L=1,2
         DO 70 I=1,5
            FLTPM(I,L) = FLTPM(I,L)/TBCF
   70    CONTINUE
   75 CONTINUE
   76 CONTINUE
```

```
C
C-------------DOUBLE FAULT MATRIX CALCULATIONS-------------------------
C
C         EVALDF(I,L)=EIGENVALUES (LAMBDA1...LAMBDA2)
C         COEFDF(I,J,L)=CHARACTERISTIC EQUASION COEFFICIENTS.
C           L-INDEX=INITIAL CONDITION (1 OR 2)
C           J-INDEX=EQUASION A,B,C,OR D (1,2,3,OR 4)
C           I-INDEX=COEFFICIENT I.E. A1,A2,A3,OR A4
C
C          EIGA(I,J) ARE THE ELEMENTS OF THE MATRIX WHOSE
C          EIGENVALUES WE ARE FINDING.
C
      NEIGD=3
C
C     SET INITIAL CONDITIONS
C
      DEIGIC(1)=1
      DEIGIC(2)=0
      DEIGIC(3)=0
      DO 90 L=1,2
      DEIGA(1,1)=-(FLTPM(1,JTP)+FLTPM(2,ITP)+FLTPM(4,JTP)+FLTPM(3,JTP))
      DEIGA(1,2)=FLTPM(2,JTP)
      DEIGA(1,3)=0.0
      DEIGA(2,1)=FLTPM(1,JTP)
      DEIGA(2,2)=-(FLTPM(2,ITP)+FLTPM(2,JTP))
      DEIGA(2,3)=FLTPM(1,JTP)
      DEIGA(3,1)=0.0
      DEIGA(3,2)=FLTPM(2,ITP)
      DEIGA(3,3)=-(FLTPM(1,ITP)+FLTPM(2,JTP)+FLTPM(4,ITP)+FLTPM(3,ITP))
         CALL DEIGEN
         DO 85 J=1,3
            EVALDF(J,L)=-DEIGWR(J)
            DO 80 I=1,3
               IF(L.EQ.1) K = 1
               IF(L.EQ.2) K = 3
               COEFDF(I,J,L)=DEIGSD(J,I,K)
 80         CONTINUE
 85      CONTINUE
         DEIGIC(1)=0
         DEIGIC(3)=1
 90   CONTINUE
      FUNCTYP = 3HPDF
120   DO 135 I = NSTPST,NSTPF
         DFAR(I) = 0.0
         RHO1  = FLTPM(4,ITP) * PREEXP(-FLTPM(4,ITP)*T)
         RHO2  = FLTPM(4,JTP) * PREEXP(-FLTPM(4,JTP)*T)
         BETA1 = FLTPM(2,ITP)
         BETA2 = FLTPM(2,JTP)
            PA1B2 = 0.0
            PA2B1 = 0.0
            DO 125 N = 1,3
               PA1B2 = (COEFDF(N,3,1)*PREEXP(-EVALDF(N,1)*T))+PA1B2
               PA2B1 = (COEFDF(N,1,1)*PREEXP(-EVALDF(N,1)*T))+PA2B1
125         CONTINUE
            DFAR(I) = ((BETA2+RHO1)*PA1B2)
     1                + ((BETA1+RHO2)*PA2B1)
            TIME(I) = T
            T = T + STEP
135   CONTINUE
```

```
      INDEX = INDEX + 1
      NSTPST = NSTPF + 1
      NSTPF = NSTPST + IDUBAR(INDEX) - 1
      STEP = STEP * 2
      IF(I .LE. ITSTPS) GO TO 120
      WRITE(6,140)
  140 FORMAT(/,3X,"WOULD YOU LIKE THIS ARRAY PRINTED AT THE TERMINAL ?")
      READ(5,101) IANSWER
      IF(IANSWER.NE.1HY) GO TO 155
      WRITE(6,142) FUNCTYP,ITP,JTP
  142 FORMAT(1H1,/,3X,"FUNCTION ",A3,"  ITYP= ",I2,"  JTYP= ",I2)
      DO 150 I=1,ITSTPS
            WRITE(6,145) I,DFAR(I),TIME(I)
  145       FORMAT(/,3X,I4,5X,E16.10,5X,1PE16.10)
  150 CONTINUE
  155 CONTINUE
      WRITE(6,160)
  160 FORMAT(/,3X,"WOULD YOU LIKE THIS FUNCTION PLOTTED ?")
      READ(5,101) IANSWER
      IF(IANSWER.NE.1HY) GO TO 176
      WRITE(6,165)
  165 FORMAT(/,3X,"ENTER PLOT TYPE: 1=LINEAR, 2=LOG, 3=BOTH")
      READ(5,*) LNORLG
      ENCODE(55,170,ITITLE) FUNCTYP,ITP,JTP
  170 FORMAT(9HFUNCTION ,A5,7H ITYP= ,I3,7H JTYP= ,I3,1H$)
      ENCODE(60,175,JTITLE) STEPIN,TBASE,ITSTPS
  175 FORMAT(26HX AXIS WITH INITIAL XSTEP= ,1PE8.2,A5,
     1       13H PTS PLOTTED=,I4,1H$)
      CALL CPLOT(DFAR,TIME,STEPIN,GENXPTS,ITSTPS,LNORLG,ITITLE,JTITLE)
  176 WRITE(6,180)
  180 FORMAT(/,3X,"WOULD YOU LIKE TO COMPUTE ANOTHER FUNCTION?")
      READ(5,101) IANSWER
      IF(IANSWER.EQ.1HY) ANOTHER = .TRUE.
      IF(IANSWER.EQ.1HY) GO TO 25
  101 FORMAT(A1)
  190 CONTINUE
      CALL DONEPL
      STOP
      END
      FUNCTION PREEXP(X)
C
      DATA REALMAX/1.0E+322/,REALMIN/1.0E-293/,EXPMAX/741.67/,
     1     EXPMIN/-675.82/
C
      IF (X.GT.EXPMIN .AND. X.LT.EXPMAX) GO TO 100
C  SET FUNCTION TO A VALUE VERY CLOSE TO 0.0 BUT NOT EQUAL TO 0.0
      IF (X.LE.EXPMIN) PREEXP = REALMIN
C  SET FUNCTION TO THE MAXIMUM VALUE THE CDC CAN HANDLE
      IF (X.GE.EXPMAX) PREEXP = REALMAX
      GO TO 200
  100 PREEXP = EXP(X)
  200 RETURN
      END
```

```
      SUBROUTINE  DEIGEN
C     ************************************************
C       DEIGEN CALLS TWO IMSL SUBROUTINES--EIGRF & LEQT2F
C       EIGRF COMPUTES EIGENVALES AND EIGENVECTORS OF
C       THE MATRIX EIGA(I,J)
C       LEQT2F SOLVES THE LINEAR SYSTEM AX=Y, WHERE
C       THE COLUMNS OF A ARE THE EIGENVECTORS OF EIGA
C       AND Y IS AN M BY N MATRIX WHOSE COLUMNS ARE THE INDIVI-
C       DUAL RIGHT HAND SIDES(NON-HOMOGENOUS TERMS)
C     ************************************************
      COMMON/DEIGCOM/ EIGSD(3,3,3),EIGWR(3),EIGA(3,3),NEIG,EIGIC(3)
      DIMENSION WK(15),WKAREA(18),A(3,3),EIGC(3,3)
      COMPLEX W(3),Z(3,3)
C       NEIG IS THE SIZE OF THE MATRIX WHOSE EIGENVALUES
C       WE ARE FINDING
C       15(WK DIMENSION) IS OBTAINED BY MULT. NEIG+2 BY NEIG
C       18(WKAREA DIMENSION) IS OBTAINED BY EVALUATING (NEIG**2)+3*NEIG
      M=NEIG
      IDGT=4
      IJOB=2
      DO 50 J=1,NEIG
      DO 50 I=1,NEIG
      EIGC(I,J)=0.
   50 IF(I.EQ.J) EIGC(I,J)=1.
      DO 55 I=1,NEIG
         EIGC(I,1)=EIGIC(I)
   55 CONTINUE
      CALL EIGRF(EIGA,NEIG,NEIG,IJOB,W,Z,NEIG,WK,IER)
      IF(WK(1).LT.1.0)PRINT(6,*) "EIGRF PERFORMED WELL,IER =",IER
      IF(WK(1).GE.1.0)PRINT(6,*)"EIGRF PERFORMED SATISFACTORILY, =",
     1IER
      IF(WK(1).GT.100.0)PRINT(6,*) "EIGRF PERFORMED POORLY,IER =",IER
      DO 60 I=1,NEIG
      EIGWR(I)=W(I)
      DO 70 J=1,NEIG
      EIGA(I,J)=Z(I,J)
      IF(ABS(EIGA(I,J)).GT.1.0E-10) GO TO 65
      EIGA(I,J)=0.0
 65   A(I,J)=EIGA(I,J)
 70   CONTINUE
 60   CONTINUE
      CALL LEQT2F(A,M,NEIG,NEIG,EIGC,IDGT,WKAREA,IER)
      DO 80 K=1,NEIG
      DO 80 I=1,NEIG
      DO 80 J=1,NEIG
C  EIGSD(I,J,K) ARE THE CONSTANTS USED TO CONSTRUCT THE
C  PROBABILITY FUNCTIONS P(I/K(T)) - PROBABILITY THAT THE SYSTEM
C  IS IN STATE I AFTER STARTING IN STATE K.
      EIGSD(I,J,K)=EIGA(I,J)*EIGC(J,K)
C
C
 80   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE CPLOT(ARTOPLT,X,STEP,GENXPTS,NPTS,LNORLG,ITITLE,
     1     JTITLE)
C *******************************************************************
C  THIS ROUTINE WORKS WITH THE DISSPLA PLOTTING PACKAGE.
C *******************************************************************
C  NOTE - ARRAYS 'ARTOPLT' AND 'X' MUST BE OF THE SAME DIMENSION
C  IN THE CALLING ROUTINE.
      DIMENSION ARTOPLT(1),X(1),ITITLE(1),JTITLE(1)
      LOGICAL GENXPTS,AZROFLG,XZROFLG,AEPZFLG
      DATA XAXISMX/7.0/,YAXISMX/8.0/,XLEFT/2.25/,XRIGHT/4.75/,
     1      YLOWER/6.50/,YUPPER/7.50/,IFRAME/2/,XPOS/2.50/,YPOS1/7.25/,
     2      YPOS2/7.0/,YPOS3/6.75/,AZROFLG/.FALSE./,XZROFLG/.FALSE./,
     3      AEPZFLG/.FALSE./
      IF(NPTS.EQ.1) GO TO 25
      K = 0
      ITEST = 1
      IAXSTYP = LNORLG
      IF(.NOT.GENXPTS) GO TO 15
      DO 10 I=1,NPTS
   10 X(I)=(I-1)*STEP
   15 IF(X(1).EQ.0.0) XZROFLG = .TRUE.
      IF(XZROFLG) X(1) = STEP/10.0
      XMIN=X(1)
      XMAX=X(NPTS)
      YMIN=ARTOPLT(1)
      YMAX=ARTOPLT(1)
C
C*************TEST FOR ARRAY - PLOT TYPE COMPATABILITY*************
C
      DO 20 J=1,NPTS
         IF(X(J).LT.XMIN) XMIN = X(J)
         IF(ARTOPLT(J).EQ.0.0) K = K+1
         IF(ARTOPLT(J).EQ.0.0) AZROFLG = .TRUE.
         IF(.NOT.(AZROFLG.AND.((J.EQ.1).OR.(J.EQ.NPTS)))) GO TO 19
            IF(J.EQ.1) ARTOPLT(J) = ARTOPLT(J+1)/10.0
            IF(J.EQ.NPTS) ARTOPLT(J) = ARTOPLT(J-1)/10.0
            YMIN = ARTOPLT(J)
            AZROFLG = .FALSE.
   19    IF(ARTOPLT(J).LT.YMIN) YMIN = ARTOPLT(J)
         IF(ARTOPLT(J).GT.YMAX) YMAX = ARTOPLT(J)
   20 CONTINUE
      IF(K.NE.NPTS) GO TO 31
   25    WRITE(6,30)
   30    FORMAT(/4X,"ARRAY TO BE PLOTTED IS IDENTICALLY ZERO; NO PLOT",
     1   "GENERATED.")
      RETURN
```

```
   31 IF(.NOT.AZROFLG) GO TO 33
          WRITE(6,32)
   32     FORMAT(/4X,"DATA POINT ON LOG AXIS .LE. ZERO; LINEAR",
     1    " AXIS PLOT GENERATED.")
      IAXSTYP = 1
   33 YTEST = (0.05*YMAX) + (0.95*YMIN)
      XTEST = (0.05*XMAX) + (0.95*XMIN)
      NP = NPTS - 1
      DO 34 I = 1,NP
         IF((XTEST.GE.X(I)).AND.(XTEST.LE.X(I+1))) ITEST = I+1
   34 CONTINUE
      IF(ARTOPLT(ITEST).LE.YTEST) IAXSTYP = 4
      IF(IAXSTYP .EQ. 4) WRITE(6,35)
   35 FORMAT(/4X,"LINEAR PLOT INADEQUATE; LOG-LOG PLOT USED")
      IF(XZROFLG) WRITE(6,36) XMIN
   36 FORMAT(/4X,"ZERO VALUE IN TIME ARRAY; XMIN = ",E10.5," USED")
      IF(AEPZFLG) WRITE(6,37) YMIN
   37 FORMAT(/4X,"ZERO VALUED END-POINT IN DATA; YMIN = ",E10.5,
     1       " USED FOR LOG PLOTS.")
C
C***********DISSPLA INITIALIZATION***********************************
C
      CALL COMPRS
      CALL BGNPL(-1)
      CALL NOCHEK
      IF(AZROFLG) GO TO 38
      CALL ALGPLT(YMIN,YMAX,YAXISMX,YORIGL,YCYCL)
      CALL ALGPLT(XMIN,XMAX,XAXISMX,XORIGL,XCYCL)
   38 CALL AXSPLT(YMIN,YMAX,YAXISMX,YORIG,YSTEP,YAXIS)
      CALL AXSPLT(XMIN,XMAX,XAXISMX,XORIG,XSTEP,XAXIS)
C
C***********LINEAR PLOT ROUTINE**************************************
C
      IF(IAXSTYP.NE.1.AND.IAXSTYP.NE.3) GO TO 40
         CALL TITLE(ITITLE,100,JTITLE,100,"LINEAR Y-AXIS$",100,
     1              XAXISMX,YAXISMX)
         CALL FRAME
         CALL BLNK1(XLEFT,XRIGHT,YLOWER,YUPPER,IFRAME)
         CALL GRAPH(XORIG,XSTEP,YORIG,YSTEP)
         CALL GRID(1,2)
      GO TO 50
C
C***********SEMI-LOG PLOT ROUTINE************************************
C
   40 IF(IAXSTYP.NE.2.AND.IAXSTYP.NE.3) GO TO 45
         CALL TITLE(ITITLE,100,JTITLE,100,"LOG Y-AXIS$",100,
     1              XAXISMX,YAXISMX)
         CALL FRAME
         CALL BLNK1(XLEFT,XRIGHT,YLOWER,YUPPER,IFRAME)
         CALL YLOG(XORIG,XSTEP,YORIGL,YCYCL)
         CALL GRID(1,1)
      GO TO 50
```

```
C
C***********LOG-LOG PLOT ROUTINE***********************************
C
   45 IF(IAXSTYP.NE.4) GO TO 50
         CALL TITLE(ITITLE,100,JTITLE,100,"LOG-LOG PLOT$",100,
     1              XAXISMX,YAXISMX)
         CALL FRAME
         CALL BLNK1(XLEFT,XRIGHT,YLOWER,YUPPER,IFRAME)
         CALL LOGLOG(XORIGL,XCYCL,YORIGL,YCYCL)
         CALL GRID(1,1)
C
C******************************************************************
C
   50 CONTINUE
C     CALL DASH
      CALL RESET("BLNK1")
      CALL MESSAG("CARE III PROJECT$",100,XPOS,YPOS1)
      CALL MESSAG("RAYTHEON COMPANY$",100,XPOS,YPOS2)
      CALL MESSAG("SUDBURY,MASS$",100,XPOS,YPOS3)
C     CALL RESET("DASH")
      CALL CURVE(X,ARTOPLT,NPTS,0)
      CALL ENDPL(0)
      IF(IAXSTYP.NE.3) GO TO 100
      IAXSTYP = 2
      GO TO 40
  100 CONTINUE
      RETURN
      END
```

APPENDIX 2

Execution Field Lengths

PROGRAM	FIELD LENGTH (Octal)
CAREIN	154000
COVRGE	163700
CARE3	157200
CVGPLT	127600
RELPLT	076000

APPENDIX 3

Selected Test Cases

THE FOLLOWING ARE THE CAREIII INPUT FILES
FOR THE SELECT TEST CASES REPORTED ON.

```
--------------------------------------------------------------------------------
 $FLTTYP ALP=0.,BET=0.,DEL=3.6E3,RHO=3.6E4,EPS=3.6E5,IEPSF=1,IDELF=1,
  CVPLOT=.T.,IAXSCV=3,C=1.0,CVPRNT=.T.$
 $STAGES NOP=3,2,N=4,M=2,IRLPCD=3,RLPLOT=.T.,IAXSRL=3$
 $FLTCAT OMG=1.0,RLM=1.0E-5 $
 $RNTIME FT=60.,ITBASE=2,SYSFLG=.T.,CPLFLG=.T.,NSTEPS=64$
***TEST CASE-T1A S.NEUMANN 23FEB82***
 1 1 2 2
 2 0 1
 CRITICAL-FAULT TREE
 1 4 5 5
 1 1 4
 5 2 1 2 3 4
--------------------------------------------------------------------------------
 $FLTTYP ALP=3.6E2,BET=0.,DEL=0.0,RHO=3.6E4,EPS=3.6E5,IEPSF=1,IDELF=1,
  CVPLOT=.T.,IAXSCV=4,C=1.0,CVPRNT=.T.,DBLDF=.06$
 $STAGES NOP=3,2,N=4,M=2,IRLPCD=3,RLPLOT=.T.,IAXSRL=4$
 $FLTCAT OMG=1.0,RLM=1.0E-5 $
 $RNTIME FT=60.,ITBASE=2,SYSFLG=.T.,CPLFLG=.T.,NSTEPS=64$
***TEST CASE-T2C S.NEUMANN 24FEB82***
 1 1 2 2
 2 0 1
 CRITICAL-FAULT TREE
 1 4 5 5
 1 1 4
 5 2 1 2 3 4
--------------------------------------------------------------------------------
 $FLTTYP ALP=3.6E6,BET=3.6E6,DEL=3.6E4,RHO=3.6E3,EPS=3.6E4,IEPSF=1,
  IDELF=2,CVPLOT=.T.,IAXSCV=4,C=1.0,CVPRNT=.T.,DBLDF=.02$
 $STAGES NOP=3,2,N=4,M=2,IRLPCD=3,RLPLOT=.T.,IAXSRL=4$
 $FLTCAT OMG=1.0,RLM=1.0E-5 $
 $RNTIME FT=60.,ITBASE=2,SYSFLG=.T.,CPLFLG=.T.,NSTEPS=64$
***TEST CASE-T3B' S.NEUMANN 24FEB82***
 1 1 2 2
 2 0 1
 CRITICAL-FAULT TREE
 1 4 5 5
 1 1 4
 5 2 1 2 3 4
--------------------------------------------------------------------------------
 $FLTTYP ALP=3.6E3,BET=3.6E6,DEL=0.0,RHO=3.6E3,EPS=3.6E4,IEPSF=1,IDELF=1,
  CVPLOT=.T.,IAXSCV=3,C=1.0,CVPRNT=.T.,TRUNC=1.0E-6$
 $STAGES NOP=3,2,N=4,M=2,IRLPCD=3,RLPLOT=.T.,IAXSRL=3$
 $FLTCAT OMG=1.0,RLM=1.0E-5 $
 $RNTIME FT=60.,ITBASE=2,SYSFLG=.T.,CPLFLG=.T.,NSTEPS=64$
***TEST CASE-T3C S.NEUMANN 02MAR82***
 1 1 2 2
 2 0 1
 CRITICAL-FAULT TREE
 1 4 5 5
 1 1 4
 5 2 1 2 3 4
--------------------------------------------------------------------------------
```

```
 $FLTTYP ALP=3.6E6,BET=3.6E3,DEL=3.6E4,RHO=3.6E3,EPS=3.6E4,IEPSF=1,
  IDELF=2,CVPLOT=.T.,IAXSCV=4,C=1.0,CVPRNT=.T.,DBLDF=0.02$
 $STAGES NOP=3,2,N=4,M=2,IRLPCD=3,RLPLOT=.T.,IAXSRL=4$
 $FLTCAT OMG=1.0,RLM=1.0E-5 $
 $RNTIME FT=60.,ITBASE=2,SYSFLG=.T.,CPLFLG=.T.,NSTEPS=64$
***TEST CASE-T3D' S.NEUMANN 1MAR82***
 1 1 2 2
 2 0 1
 CRITICAL-FAULT TREE
 1 4 5 5
 1 1 4
 5 2 1 2 3 4
--------------------------------------------------------------------------------
 $FLTTYP ALP=2*3.6E3,BET=2*3.6E3,DEL=2*3.6E3,RHO=2*3.6E4,EPS=2*3.6E5,
 NFTYPS=2,IEPSF=2*1,IDELF=2*1,CVPLOT=.T.,IAXSCV=3,C=1.0,CVPRNT=.T.,
     PA=1.0,0.9,PB=0.0,0.1 $
 $STAGES NOP=3,2,N=4,M=2,IRLPCD=3,RLPLOT=.T.,IAXSRL=3$
 $FLTCAT OMG=1.0,RLM=1.0E-5 $
 $RNTIME FT=60.,ITBASE=2,SYSFLG=.T.,CPLFLG=.T.,NSTEPS=64$
***TEST CASE-T4A S.NEUMANN 03MAR82***
 1 1 2 2
 2 0 1
 CRITICAL-FAULT TREE
 1 4 5 5
 1 1 4
 5 2 1 2 3 4
--------------------------------------------------------------------------------
 $FLTTYP ALP=3.6E3,3.6E6,BET=2*3.6E3,DEL=2*0.0,RHO=2*3.6E3,EPS=2*3.6E4,
 NFTYPS=2,IEPSF=2*1,IDELF=2*1,CVPLOT=.T.,IAXSCV=3,C=1.0,CVPRNT=.T.,
 PA=2*1.0,PB=2*0.0 $
 $STAGES NOP=3,2,N=4,M=2,IRLPCD=3,RLPLOT=.T.,IAXSRL=3$
 $FLTCAT OMG=1.0,RLM=1.0E-5 $
 $RNTIME FT=60.,ITBASE=2,SYSFLG=.T.,CPLFLG=.T.,NSTEPS=64$
***TEST CASE-T4B S.NEUMANN 02MAR82***
 1 1 2 2
 2 0 1
 CRITICAL-FAULT TREE
 1 4 5 5
 1 1 4
 5 2 1 2 3 4
--------------------------------------------------------------------------------
```

APPENDIX 4

Coverage Functions

Single-Fault Model Equations

FUNCTION	MATHEMATICAL EXPRESSION*	DEFINITION
$\phi(t)$	$\alpha e^{-\beta t}\int_0^t e^{-(\alpha-\beta)\tau} r(\tau)d(\tau)d\tau$	β^{-1} TIMES THE PROBABILITY INTENSITY OF RE-ENTERING STATE A EXACTLY t TIME UNITS AFTER THE PREVIOUS ENTRY
$P_a(t)$	$e^{-\alpha t}r(t)d(t) + \beta\int_0^t \phi(t-\tau)P_a(\tau)d\tau$	PROBABILITY OF BEING IN STATE A AT TIME t WHEN $P_A = P_B = 1$
$P_b(t)$	$\phi(t) + \beta\int_0^t \phi(t-\tau)P_b(\tau)d\tau$	PROBABILITY OF BEING IN STATE B AT TIME t WHEN $P_A = P_B = 1$
$P_e(t)$	$\int_0^t e^{-\alpha\tau}\rho(\tau)d(\tau)e(t-\tau)d\tau + \beta\int_0^t \phi(t-\tau)P_e(\tau)d\tau$	PROBABILITY OF BEING IN STATE A_E OR B_E AT TIME t WHEN $P_A = P_B = 1$

* t HERE IS A MEASURE OF THE TIME SINCE THE ENTRY INTO STATE A.

Single-Fault Model Equations(Continued)

FUNCTION	MATHEMATICAL EXPRESSION*	DEFINITION
$p_e(t)$	$e^{-\alpha t}\rho(t)d(t) + \beta\int_0^t \phi(t-\tau)p_e(\tau)d\tau$	INTENSITY OF ENTRY INTO STATE A_E AT TIME t WHEN $P_A = P_B = 1$
$p_{\bar{e}}(t)$	$e^{-\alpha t}\delta(t)r(t) + \beta\int_0^t \phi(t-\tau)p_{\bar{e}}(\tau)d\tau$	INTENSITY OF ENTRY INTO STATE A_D FROM STATE A AT TIME t WHEN $P_A = P_B = 1$
$p_f(t)$	$(1-C)\int_0^t p_e(\tau)\varepsilon(t-\tau)d\tau$	INTENSITY OF ENTRY INTO STATE F AT TIME t WHEN $P_A = P_B = 1$
$\Psi_A(t)$	$C\int_0^t p_e(\tau)\varepsilon(t-\tau)\left(\frac{\beta + \alpha e^{-(\alpha+\beta)(t-\tau)}}{\alpha+\beta}\right)d\tau + p_{\bar{e}}(t)$	INTENSITY OF ENTRY INTO STATE A_D AT TIME t FOR THE FIRST TIME

* t HERE IS A MEASURE OF THE TIME SINCE THE ENTRY INTO STATE A.

Single-Fault Model Equations (Continued)

FUNCTION	MATHEMATICAL EXPRESSION*	DEFINITION
$\Psi_B(t)$	$\frac{\alpha C}{\alpha+\beta}\int_0^t P_e(\tau)(1 - e^{-(\alpha+\beta)(t-\tau)})\varepsilon(t-\tau)d\tau$	INTENSITY OF ENTRY INTO STATE B_0 AT TIME t FOR THE FIRST TIME
$\chi_B(t)$	$\int_0^t \Psi_B(\tau)e^{-\beta(t-\tau)}d\tau$	PROBABILITY OF HAVING ENTERED STATE B_D FOR THE FIRST TIME AND THEN REMAINING IN THE BENIGN STATE UNTIL TIME t
$P_{dp}(t)$	$P_A\int_0^t \Psi_A(\tau)d\tau + P_B\int_0^t \Psi_B(\tau)d\tau$	PROBABILITY THAT A FAULT HAS BEEN DIAGNOSED AS PERMANENT BY TIME t
$F_X(t)$	$F_X(t) + \int_0^t [(1-P_A)\Psi_A(t-\tau) + (1-P_B)\beta\chi_B(t-\tau)]F_X(\tau)d\tau$	FUNCTION RELATING PROBABILITIES AND INTENSITIES DERIVED WHEN $P_A = P_B = 1$ TO THOSE SAME QUANTITIES WHEN P_A & P_B ARE ARBITRARY

* t HERE IS A MEASURE OF THE TIME SINCE THE ENTRY INTO STATE A.

Single-Fault Model Equations (Continued

FUNCTION	MATHEMATICAL EXPRESSION*	DEFINITION
$P_B(t)$	$F_X(t)$ with $F_x(t) = P_b(t) + X_B(t)$	PROBABILITY OF BEING IN STATE B AT TIME t
$P_{\overline{B}}(t)$	$F_X(t)$ with $F_x(t) = P_a(t) + P_e(t)$	PROBABILITY OF BEING IN A NON-BENIGN STATE AT TIME t
$P_L(t)$	$F_X(t)$ with $F_x(t) = \begin{cases} P_b(t) + X_B(t) + P_a(t) + P_e(t) & \text{PERMANENT FAULTS} \\ P_a(t) + P_e(t) & \text{TRANSIENT FAULTS} \end{cases}$	PROBABILITY OF A LATENT FAULT OR UNDETECTED ERROR AT TIME t
$P_{DP}(t)$	$F_X(t)$ with $F_x(t) = P_{dp}(t)$	PROBABILITY THAT A FAULT HAS BEEN DIAGNOSED AS PERMANENT BY TIME t

* t HERE IS A MEASURE OF THE TIME SINCE THE ENTRY INTO STATE A.

Double-Fault Model Equations

FUNCTION	MATHEMATICAL EXPRESSION	DEFINITION
$c_i(t)$ $i = 1,2$ $j = 3-i$	$\beta_i(t)d_j(t)r_j(t)a_j(t)$ + $(1-P_{A_j})b_i(t)\,\delta_j(t)r_j(t)a_j(t)$ + $b_i(t)d_j(t)\rho_j(t)a_j(t)$	TRANSITION RATE FROM STATE A_jB_i TO STATE F
$f_i(t)$ $i = 1,2$ $j = 3-i$	$\alpha_j(t)b_i(t)d_i(t)r_i(t)$	TRANSITION RATE FROM STATE A_jB_i TO STATE B_1B_2
$c_4(t)$	$\int_0^t [c_1(t-\tau)\beta_2(\tau)b_1(\tau)$ + $c_2(t-\tau)\beta_1(\tau)b_2(\tau)]d\tau$	INTENSITY OF ENTRY INTO STATE F t TIMEUNITS AFTER ENTRY INTO STATE B_1B_2
$c_3(t)$	$\int_0^t [f_1(t-\tau)\beta_2(\tau)b_1(\tau)$ + $f_2(t-\tau)\beta_1(\tau)b_2(\tau)]d\tau$	INTENSITY OF RE-ENTRY INTO STATE B_1B_2 t TIME UNITS AFTER A PREVIOUS ENTRY

Double-Fault Model Equations (Continued)

FUNCTION	MATHEMATICAL EXPRESSION	DEFINITION
$p_3(t)$	$f_1(t) + \int_0^t c_3(t-\tau)p_3(\tau)d\tau$	INTENSITY OF ENTRY INTO STATE B_1B_2 t TIME UNITS AFTER ENTRY INTO STATE A_2B_1
$p_{DF}(t)$	$c_1(t) + \int_0^t c_4(t-\tau)p_3(\tau)d\tau$	INTENSITY OF ENTRY INTO STATE F t TIME UNITS AFTER ENTRY INTO STATE A_2B_1

REFERENCES

1. Bryant, L. A.; and Stiffler, J. J.: CARE III PHASE II Report, User's Manual. NASA CR-165864, 1982.

2. Smith, T. B., et al.: A Fault Tolerant Microprocessor Architecture for Aircraft, Interim Report. NASA CR-165911, 1978.

3. Stiffler, J. J.; Bryant, L. A.; and Guccione, L.: CARE III FINAL REPORT, PHASE I. NASA CR-159122, 1979.

4. Stiffler, J. J.; and Bryant, L. A.: CARE III PHASE II REPORT - Mathematical Description. NASA CR-3566, 1982.

5. Wensley, J. H., et al.: Design Study of Software-Implemented Fault-Tolerance (SIFT) Computer. NASA CR-3011, 1982.

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA CR-3631		

4. Title and Subtitle	5. Report Date
CARE III PHASE III REPORT - TEST AND EVALUATION	November 1982
	6. Performing Organization Code
7. Author(s) J. J. Stiffler, J. S. Neumann, and L. A. Bryant	8. Performing Organization Report No.
9. Performing Organization Name and Address Raytheon Company 528 Boston Post Road Sudbury, Massachusetts 01776	10. Work Unit No.
	11. Contract or Grant No. NAS1-15072
	13. Type of Report and Period Covered Contractor Report
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, DC 20546	14. Sponsoring Agency Code 505-34-43-05

15. Supplementary Notes

NASA Project Engineer: Salvatore J. Bavuso

16. Abstract

CARE III (Computer-Aided Reliability Estimation, version three) is a computer program designed to help estimate the reliability of complex, redundant systems; although the program can model a wide variety of redundant structures, it was developed specifically for fault-tolerant avionics systems - systems distinguished by the need for extremely reliable performance since a system failure could well result in the loss of human life.

The first CARE program, developed at the Jet Propulsion Laboratory in 1971, provided an aid for estimating the reliability of systems consisting of a combination of any of several standard configurations (e.g., standby-replacement configurations, triple-modular redundant configurations, etc.). CARE II was subsequently developed by Raytheon, under contract to the NASA Langley Research Center, in 1974. It substantially generalized the class of redundant configurations that could be accommodated, and included a coverage model to determine the various coverage probabilities as a function of the applicable fault recovery mechanisms (detection delay, diagnostic scheduling interval, isolation and recovery delay, etc.).

CARE III further generalizes the class of system structures that can be modeled and greatly expands the coverage model to take into account such effects as intermittent and transient faults, latent faults, error propagation, etc.

This study reports on the initial test and evaluation of CARE III. Further efforts to validate CARE III are required.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Reliability modeling Fault coverage Fault models Fault tolerant avionics	Unclassified - Unlimited Subject Category 59

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	166	A08

*For sale by the National Technical Information Service, Springfield, Virginia 22161