AD-A090 849 ADVISORY GROUP FOR AEROSPACE RESEARCH AND DEVELOPMENT--ETC F/G 9/3
FAULT TOLERANCE DESIGN AND REDUNDANCY MANAGEMENT TECHNIQUES.(U)
SEP 80
UNCLASSIFIED AGARD-LS-109 NL

1 OF 2
AD A
090849

LEVEL II

AGARD-LS-109

AGARD-LS-109

AD A090849

AGARD

ADVISORY GROUP FOR AEROSPACE RESEARCH & DEVELOPMENT

7 RUE ANCELLE 92200 NEUILLY SUR SEINE FRANCE

AGARD LECTURE SERIES No. 109

Fault Tolerance Design and Redundancy Management Techniques

NORTH ATLANTIC TREATY ORGANIZATION

DISTRIBUTION AND AVAILABILITY

(14) AGARD-LS-109

NORTH ATLANTIC TREATY ORGANIZATION

ADVISORY GROUP FOR AEROSPACE RESEARCH AND DEVELOPMENT

(ORGANISATION DU TRAITE DE L'ATLANTIQUE NORD)

(11) Sep 80 (12) 175 p

AGARD Lecture Series No.109

FAULT TOLERANCE DESIGN AND REDUNDANCY MANAGEMENT TECHNIQUES

The material in this publication was assembled to support a Lecture Series under the sponsorship of the Guidance and Control Panel and the Consultant and Exchange Programme of AGARD, presented on: 13–14 October 1980 in Athens, Greece; 16–17 October 1980 in Rome, Italy and 20–21 October 1980 in London, UK.

THE MISSION OF AGARD

The mission of AGARD is to bring together the leading personalities of the NATO nations in the fields of science and technology relating to aerospace for the following purposes:

- Exchanging of scientific and technical information;
- Continuously stimulating advances in the aerospace sciences relevant to strengthening the common defence posture;
- Improving the co-operation among member nations in aerospace research and development;
- Providing scientific and technical advice and assistance to the North Atlantic Military Committee in the field of aerospace research and development;
- Rendering scientific and technical assistance, as requested, to other NATO bodies and to member nations in connection with research and development problems in the aerospace field;
- Providing assistance to member nations for the purpose of increasing their scientific and technical potential;
- Recommending effective ways for the member nations to use their research and development capabilities for the common benefit of the NATO community.

The highest authority within AGARD is the National Delegates Board consisting of officially appointed senior representatives from each member nation. The mission of AGARD is carried out through the Panels which are composed of experts appointed by the National Delegates, the Consultant and Exchange Programme and the Aerospace Applications Studies Programme. The results of AGARD work are reported to the member nations and the NATO Authorities through the AGARD series of publications of which this is one.

Participation in AGARD activities is by invitation only and is normally limited to citizens of the NATO nations.

Published September 1980

ISBN 92-835-0274-4

Printed by Technical Editing and Reproduction Ltd
Harford House, 7–9 Charlotte St, London, W1P 1HD

FOREWORD

This Lecture Series No.109 is sponsored by the Guidance and Control Panel of AGARD and implemented by the Consultant and Exchange Programme.

These lectures are intended to provide the basic theory on concepts involved in the application of advanced software, state estimation, and implementation techniques involved in redundancy management, and to give a review covering the necessary background and state-of-the-art involved in the application of advancing technologies.

T.B.CUNNINGHAM
Lecture Series Director

LIST OF SPEAKERS

Lecture Series Director: Mr T.B.Cunningham
Honeywell Systems Research Center
MN 17-2367
2600 Ridgway Parkway
Minneapolis, Minn. 53413
USA

SPEAKERS

Professor J.Ackermann
DFVLR Institute for Dynamics
of Flight Systems
8031 Oberpfaffenhofen
Germany

Mr K.J.Folkesson
Flight Control Systems
Saab-Scania Aerospace
Linköping
Sweden

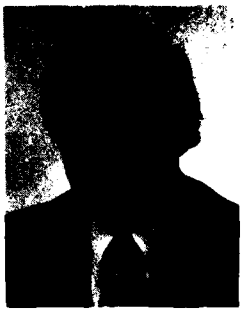
Mr M.Labarrère
Centre d'Etudes et de Recherches
de Toulouse
2 Avenue Edouard Belin
31055 Toulouse Cedex
France

Dr K.Levitt
Computer Science Laboratory
Stanford Research Institute International
330 Ravenswood Avenue
Menlo Park, California 94025
USA

Mr K.Szalai
NASA-Dryden Flight Research Center
P.O. Box 273
Edwards, California
USA

Professor A.S.Willsky
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
Cambridge, Mass. 02139
USA

LIST OF SPEAKERS



LECTURE SERIES DIRECTOR:

T. B. Cunningham

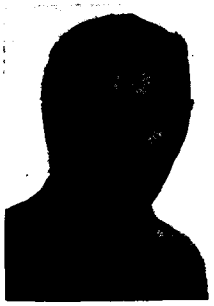
Senior Principal Research Engineer
Systems and Research Center
Honeywell, Inc.
Minneapolis, Minnesota U.S.A

J. Ackermann

Director, DFVLR
Institute for Dynamics
of Flight Systems
Oberphaffenhofen, GERMANY

K. N. Levitt

Computer Science Laboratory
Stanford Research Institute
International
Menlo Park, California U.S.A.



K. Folkesson

Manager, Flight Control Systems
Saab Scania Aerospace
Linkoping, SWEDEN

K. J. Szalai

Chief, Control Branch
National Aeronautics and Space
Administration
Hugh L. Dryden Flight Research
Center
Edwards, California U.S.A.

M. Labarrere

Centre d'Etudes et de Recherches
Toulouse, FRANCE

A. S. Willsky

Laboratory for Information and
Decision Systems and
Department of Electrical Engineering
and Computer Sciences
Massachusetts Institute of Technology
Cambridge, Massachusetts U.S.A.

CONTENTS

INTRODUCTION AND OVERVIEW

by
Thomas B. Cunningham
Honeywell Systems and Research Center
Minneapolis, Minnesota
UNITED STATES OF AMERICA

SUMMARY

The pursuit of fault tolerance in avionics systems can have a significant impact on costs and performance of future high performance aircraft. This introduction to Lecture Series No. 109 gives a brief discussion of the motivation for fault tolerance through failure management. The technical scope of the lecture series is also bounded.

1. AVIONICS - PERFORMANCE AND COSTS

The successful design of high performance Avionics Systems for aircraft blends a number of hardware and software technologies together to meet a clear (hopefully) set of requirements.

Performance - Hardware performance goals can be such things as noise, bias and bandwidth specifications for sensors to meet navigation accuracy or flight control stability requirements. Specifications on software for such tasks might include strapdown algorithms or control law compensation. These in turn dictate digital computer memory and throughput requirements.

Reliability - Reliability specifications are evloving which form the basis for determining the minimum hardware and software complement to attack performance requirements. The most prevalent type of specification is the probability of catastrophic failure which is currently 10^{-9} or better for commercial aircraft and 10^{-4} to 10^{-5} for military applications. These are supplemented by other constraints which dictate a minimum "number of channels of things" directly. Specifications pertaining to aircraft dispatch and aircraft availability with failed components from areas remote to maintenance facilities are examples.

Survivability - The survival of an aircraft and crew after part of the vehicle has been damaged has significant relevance to military operations due to combat encounters and also civil flight (i.e., engines sometimes fly apart or fall off). Solutions to survivability issues vary from stuffing every single point failure under the pilot seat to fully dispersed "brick wall" redundant channels.

These three areas: performance, reliability and survivability can provide sufficient constraint boundaries to dictate an avionics suite for an aircraft. Performance dictates sensors, surface, computer requirements and placements. Reliability can be obtained by a sufficiently large number of redundant channels (dispatch adds one more). Survivability can take the easy way out and stuffs it all under the pilot seat.

System fault tolerance is easily handled with redundancy and failure management techniques consisting of well placed comparitors. End of design and end of lecture series 109.

This oversimplification has some obvious flaws.

- Extra weight of redundant avionics impact performance,
- Survivability consisting of an expensive pilot seat cushion ignores others on board
- Electrical wires (proliferated by redundant avionics) make terrific antennas

The major missing element, however, is cost. Avionics costs are high.

The added element of minimum costs or more formally "life cycle costs" and "cost of ownership" drive systems technology to look at alternatives to blind redundancy and separate function hardware.

Sharing of hardware, most exemplified by navigation - weapon delivery - attitude reference - flight control sensor sharing is an emerging technique for cost reductions. This, however, causes some survivability problems which must be dealt with. Other hardware (and cost) reducing techniques encompass one of the main thrusts of this lecture series. These have to do with the replacement of hardware with analytical techniques. Nowhere are the requirements of performance, reliability and survivability higher and the cost reduction potential greater than the flight control problem.

2. FLIGHT CONTROL - THE CRUCIAL FAULT TOLERANCE TEST

The use of advanced digital flight control design techniques with fly-by-wire implementation offers great performance benefits for future aircraft. Almost all of these performance enhancing techniques, such as active control for implementing related static stability, flutter and structural mode control, maneuver load control, and gust suppression, result in higher risks for mission completion and flight safety due to flight control component failure.

In recognition of these higher risks a proliferation of redundant components has evolved; namely sensors, computers, and servo-actuation systems. In parallel with this a technology has been developed to manage this redundancy. Redundancy and appropriate management are deemed sufficient to give mission and safety assurance as predicted by reliability analysis but very few such systems have flown to date.

More recent developments in failure management have resulted from analytical techniques aimed at

- Reducing the costs and logistics of redundant hardware.
- Improving aircraft survivability by allowing more dispersion of components.

Such techniques are currently in the development and flight test stages.

The objective of the lecture series is to present the current "state-of-the-art" and future directions in fault tolerance through failure management. This will include:

1. The currently available engineering techniques and mathematical tools which can be brought to bear
   - Flight proven redundancy management techniques
   - Reliability and survivability considerations
   - Modern estimation and failure monitor design techniques
2. Results of proven failure management systems from flight test results and production flight systems.
3. Descriptions of "analytical redundancy" design efforts from conceptual design through flight test results.
4. Flight control design techniques to:
   1. Prevent instability due to undetected failures or failure recovery transients,
   2. Provide reversion modes to enhance performance with failed hardware components.

One theme which is consistent throughout the evolution of failure management is increased reliance on on-board digital computers for implementing failure management systems as well as other flight management systems.

The digital computer, therefore, is more than another hardware component "in the loop," because many of the failure risks are now imposed on the computer software. For this reason a special lecture on software validation and verification techniques will be included in the series.

ACKNOWLEDGEMENTS

The author would like to express his gratitude to Ms Sue Barton and Ms Cynthie Stary for their diligent and patient secretarial support of this lecture series.

FAILURE DETECTION IN DYNAMIC SYSTEMS

by

Alan S. Willsky
Department of Electrical Engineering and Computer Science
and
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

SUMMARY

In this paper we present an introduction to the basic concepts behind the design of algorithms for the detection of failures in dynamic systems. We will focus our attention on two very important methods: the multiple model (MM) technique and the generalized likelihood ratio (GLR) method. In the context of these two methods we will explore many of the fundamental issues that arise in failure detection, including the structure of failure detection algorithms, the computational complexity of different approaches, and the different modeling and system configuration assumptions on which different algorithms are based. Following these discussions we will focus our attention on two issues: the robust use of analytical redundancy in practical applications and the design of decision rules that reflect the system tradeoffs that must be made.

I. Introduction

In recent years a wide variety of techniques have been proposed for the detection of failures in dynamic systems. Some of these methods have been developed starting from general, abstract dynamic models, while other have been produced in the context of particular applications. While the general methods provide the basis for a widely applicable failure detection methodology, their very generality may obscure or at least fail to accentuate the important concepts that must be considered in the practical implementation of failure detection systems. On the other hand, while the methods that have been developed for specific applications may directly address these basic concepts, this is often done in a very problem-specific manner which can make it difficult to separate out those aspects of the design that can be generalized and those that cannot.

In this paper we will focus our attention on two general failure detection techniques in order to provide an introduction to the techniques that have been developed in this area. We will, however, try to provide a fair amount of insight into the structure of these methods and in this way we will uncover the important issues that must be considered in the design of failure detection systems. Throughout this paper we will draw heavily from results and practical experience documented in references [1-7].

In the next two sections we develop the multiple model (MM) and generalized likelihood (GLR) ratio techniques and discuss and contrast the models on which they are based, their structure, their computational complexity, and other of their properties. With this as background, in Section IV we discuss some of the basic issues involved in failure detection, how these are addressed by the MM and GLR methods, and how these methods relate to other failure detection methods. We also provide a brief discussion of the problem of practical, robust failure detection. Finally, in Section V we provide an introduction to the use of techniques in statistical decision theory for the design of decision rules for failure detection systems.

II. The Multiple Model Method

The multiple model method deals with a problem of the following type: we observe a sequence of inputs, $u(k)$, $k=0,1,2,\ldots$, to and outputs $y(k)$, $k=1,2,\ldots$, from a system and we wish to choose one out of a given finite set of possible models that we feel is most likely to have responded in the observed fashion. This type of problem is, of course, not of interest solely in the context of failure detection, but in fact it also arises in system identification and in adaptive control. Indeed, the initial development of the MM technique was performed in these contexts, and in the initial part of this section we will follow some of these early treatments (see references [8-13, 19-21, 24] for more on the development of MM). Later in the section we will look a bit more closely at the adaptation and use of MM for failure detection (see [1,4,6,20,22] for other treatments and applications of MM to problems of detecting failures and other abrupt events.)

The specific problem of the type described in the preceding paragraph that is addressed by the MM technique involves assuming that the actual system corresponds exactly to one of a finite set of linear stochastic systems, indexed by $i=1,\ldots,N$:

$$x_i(k+1) = A_i(k)x_i(k) + B_i(k)u(k) + w_i(k) + g_i(k) \tag{2.1}$$

$$y(k) = C_i(k)x_i(k) + v_i(k) + b_i(k) \tag{2.2}$$

where $w_i(k)$ and $v_i(k)$ are independent, zero-mean, Gaussian white noise processes, with

$$E[w_i(k)w_i(j)'] = Q_i(k)\delta_{jk} \tag{2.3}$$

$$E[v_i(k)v_i(j)'] = R_i(k)\delta_{jk} \tag{2.4}$$

Also, $b_i(k)$ and $q_i(k)$ are deterministic functions of time (corresponding to biases, linearizations about different operating points, etc.). In addition, the state vectors $x_i(k)$ may be of different dimensions for different values of i (corresponding to assuming that the different hypothesized models represent different orders for the dynamics of the real system). There are a number of issues that can be raised concerning this formulation, and we defer our critique of the MM method until after we have developed its basic structure. We note here only one technical point which is that we will focus on a discrete-time formulation of the MM method. Continuous-time versions can be found in the literature (see [24]), and they differ from their discrete-time counterparts only in a technical and not in a conceptual or structural manner.

Assuming that one of these N models is correct, we now have a standard multiple hypothesis testing problem. That is, let H_i denote the hypothesis that the real system corresponds to the ith model, and let $p_i(0)$ denote the a priori probability that H_i is true. Similarly, let $p_i(k)$ denote the probability that H_i is true based on measurements through the kth measurement, i.e. given $I_k = \{u(0),\ldots,u(k-1), y(1),\ldots,y(k)\}$. Then Bayes' rule yields the following recursive formula for the $p_i(k)$

$$p_i(k+1) = \frac{p(y(k+1)\,|\,H_i, I_k, u(k))\,p_i(k)}{\sum_{j=1}^{N} p(y(k+1)\,|\,H_j, I_k, u(k))\,p_j(k)} \tag{2.5}$$

Thus, the quantities that must be produced at each time are the conditional probability densities $p(y(k+1)\,|\,H_i, I_k, u(k))$ for i=1,...,N. However, conditioned on H_i, this probability density is precisely the one step prediction densities produced by a Kalman filter based on the ith model.

That is, let $\hat{x}_i(k+1|k)$ be the one-step predicted estimate of $x_i(k+1)$ based on I_k and u(k), assuming that H_i is true. Also let $\hat{x}_i(k+1|k+1)$ denote the filtered estimate of $x_i(k+1)$ based on $I_{k+1}=\{I_k, u(k), y(k+1)\}$ and the ith model. Then these quantities are computed sequentially from the following equations:

$$\hat{x}_i(k+1|k) = A_i(k)\hat{x}_i(k|k) + B_i(k)u(k) + g_i(k) \tag{2.6}$$

$$\hat{x}_i(k+1|k+1) = \hat{x}_i(k+1|k) + K_i(k+1)\gamma_i(k+1) \tag{2.7}$$

where $\gamma_i(k+1)$ is the measurement innovations process

$$\gamma_i(k+1) = y(k+1) - C_i(k)\hat{x}_i(k+1|k) \tag{2.8}$$

and K(k+1) is calculated off-line from the following set of equations:

$$P_i(k+1|k) = A_i(k)P_i(k|k)A_i'(k) + Q_i(k) \tag{2.9}$$

$$V_i(k+1) = C_i(k)P_i(k+1|k)C_i'(k) + R_i(k) \tag{2.10}$$

$$K_i(k+1) = P_i(k+1|k)C_i'(k)V_i^{-1}(k+1) \tag{2.11}$$

$$P_i(k+1|k+1) = P_i(k+1|k) - K_i(k+1)C_i(k)P_i(k+1|k) \tag{2.12}$$

Here $P_i(k+1|k)$ denotes the estimation error covariance in the estimate $\hat{x}_i(k+1|k)$ (assuming H_i to be true), and $P_i(k+1|k+1)$ is the covariance of the error $x_i(k+1) - \hat{x}_i(k+1|k+1)$, again based on H_i. Also under hypothesis H_i, $\gamma_i(k+1)$ is zero mean with covariance $V_i(k+1)$, and it is normally distributed (since we have assumed that all noises are Gaussian). Furthermore, conditioned on H_i, I_k, and u(k), y(k+1) is Gaussian, has mean $C_i(k)\hat{x}_i(k+1|k)$ and covariance $V_i(k+1)$. Thus, from (2.8) we deduce that

$$p(y(k+1)\,|\,H_i, I_k, u(k)) = \frac{1}{(2\pi)^{m/2}[\det V_i(k+1)]^{1/2}} \exp\{-\frac{1}{2}\gamma_i'(k+1)V_i^{-1}(k+1)\gamma_i(k+1)\} \tag{2.13}$$

where m is the dimension of y.

Equations (2.5)-(2.8) and (2.13) define the MM algorithm. The inputs to the procedure are the y(k) and u(k) and the outputs are the $p_i(k)$. The implementation of the algorithm can be viewed as consisting of a bank of N Kalman filters, one based on each of the N possible models. The outputs of these Kalman filters are the innovations sequences $\gamma_i(k+1)$, which effectively measure how well each of the filters can track and predict the behavior of the observed data. Specifically, if the ith model is correct, then the one-step prediction error $\gamma_i(k)$ should be a white sequence, resulting only from the intrinsic uncertainty in the ith model. However if the ith model is not correct, then $\gamma_i(k)$ will not be white and will include errors due to the fact that the prediction is based on an erroneous model. Thus the probability calculation (2.5), (2.13) basically provides a quantitative way in which to assess which model is most likely to be correct by comparing the performances of predictors based on these models.

In the remainder of this section we will address some of the most important questions that arise in understanding how the MM algorithm should be used. Some of these questions we will consider in some detail, while others we will simply raise.

Question 1: What can be done in using MM in problems in which the real system is nonlinear and/or the noises are non-Gaussian?

This is a very problem-dependent question. The Gaussian assumption is basically used in one place--i.e. in the evaluation of $p(y(k+1)|H_i,I_k,u(k))$ in (2.13). It has been our experience that using this formula, even when $\gamma_i(k+1)$ is non-Gaussian, causes essentially no performance degradation. As we have pointed out, what MM really attempts to do is to calculate a measure of how well each of the Kalman filters is tracking by looking at the prediction errors $\gamma_i(k+1)$, and the $p_i(k)$ are simply measure of how well each of the models are tracking relative to each other and to how well we would expect them to be tracking. The critical term in (2.13) is

$$\gamma_i(k+1)V_i^{-1}(k+1)\gamma_i(k+1) \tag{2.14}$$

which is the square of the tracking error normalized by the predicted covariance of these errors assuming H_i is true. Thus if this quantity is large, we would tend to disregard the ith model, while if this is small, the ith filter is tracking well. The $p_i(k)$ exhibit exactly this type of behavior, and thus we can expect MM to be reasonably robust to non-Gaussian statistics. Of course this depends upon the application, but we have had good success in several applications [3,4,6] in which the noises were decidedly non-Gaussian.

As far as the nonlinearity of the real system is concerned, an obvious approach is to linearize the system about a number of operating points for each possible model and use these linearized models to design extended Kalman filters which would be used in place of Kalman filters in the MM algorithm. Again the utility of this approach depends very much on the particular application. Essentially the issue is whether the tracking error from the extended Kalman filter corresponding to the linearized model "closest to" the true, nonlinear system is markedly smaller than the errors from filters based on "more distant" models. This is basically a signal-to-noise ratio problem, similar to that seen in the idealized MM algorithm in which everything is linear. In that case the noise is measured by the $V_i(k+1)$. The larger these are, the harder it will be to distinguish the models (the quantity in (2.14) becomes smaller as V_i is increased, and this in turn tends to flatten out (as a function of i) the probabilities in (2.13)). In the nonlinear case, the inaccuracies of the extended Kalman filters effectively increase the $V_i(k+1)$ thus reducing their tracking capabilities and making it more difficult to distinguish among them. Therefore, the performance of MM in this case will depend upon how "far apart" the different models are, as compared to how well each of the trackers tracks. The farther apart the models are, the more signal we have; the poorer the tracking performance is, the more noise is present.

Clearly the issue of robustness of MM to discrepancies between the hypthesized models and true system is very problem-dependent. For some further insight into this we refer the reader to [6,15,17,18], in which several experiences in applying MM to nonlinear problems are reported, and to the following question.

Question 2: Even if the true system is linear, what can we do when it does not correspond exactly to one of the hypothesized models?

Again this is a question of signal-to-noise ratio, but in the linear case a nuber of results and approaches have been developed for dealing with this problem. For example, Baram [14] has developed a precise mathematical procedure for calculating the distance between different linear models and he has shown that the MM procedure will converge to the model closest to the real model (i.e. $p_i(k)\longrightarrow 1$ for the model nearest the true system). This can be viewed as a technique for testing the robustness of MM or as a tool that enables us to decide what models to choose. That is, if the real system is in some set of models that may be infinite or may in fact represent a continuum of models (corresponding to the precise values of certain parameters), then Baram's results can be used to decide upon a finite set of these models that span the original set and that are far enough apart so that MM can distinguish among them. For example, in adaptive flight control (reference [15]) we may be interested in determining the flight condition (operating point) of an aircraft, and we can think of using MM by hypothesizing a set of linearized models that span the flight envelope. Morevoer, as is done in [16], we can also consider adaptively changing the set of hypothesized models in order to obtain a finer estimate of the true model once we have determined which of a coarse set of linear models is nearest to the true one. This is one of many potential applications of MM, which is the topic of the next question.

Question 3: To what problems can MM be applied?

As we have just discussed MM can be used as the basis for a maximum likelihood system identification procedure in which we begin with a coarse discretization of the set of possible models which is replaced by successively finer sets as MM pinpoints the true system. Clearly the signal-to-noise ratio issue will determine how finer this identification can be made. A second area of application of MM is in adaptive control [9,12,13,15,16,24]. Specifically, one can imagine a control system that consists of a set of possible control laws corresponding to several possible models of the plant to be controlled. Having such a set we can use the probabilities from the MM algorithm to decide on the appropriate mix of the various control strategies. For example, suppose that we have a set of linear control laws

$$u_i(k) = G_i(k)\hat{x}_i(k|k) \qquad i=1,\ldots,N \tag{2.15}$$

Then we can consider two possible control strategies -- a maximum probability control law

$$u(k) = u_{i_0}(k) \tag{2.16}$$

where

$$p_{i_0}(k) \geq p_i(k) \qquad \text{for all } i \neq i_0$$

or a probabilistically weighted control law

$$u(k) = \sum_{i=1}^{N} p_i(k) u_i(k) \tag{2.17}$$

The latter of these has been referred to as the Multiple Model Adaptive Control (MMAC) algorithm [9,12,15,24]. If one of these models accurately represents the true plant, either of these two control laws works reasonably well in the sense that eventually the probability will lock onto the closest model and the system will be stabilized. However, the transient behavior of this system can be quite erratic, especially if some of the control laws in (2.15) are destabilizing. The development of MMAC into a usable design methodology is an area of current research [24].

In addition to identification and adaptive control, there are, of course many applications in which MM is used to distinguish among truly distinct hypotheses (as opposed to hypotheses linked via the values of some dynamic parameters). For example, MM has been successfully used in detecting arrhythmias in electrocardiograms [4] in which the models represent completely different rhythm patterns, in deducing muscle movement from electromyogram signals [25] in which the hypotheses represent different movements of the human arm, and in detecting capacity-reducing incidents on a freeway [6], in which the models describe traffic dynamics with and without lane blockages. Finally, of course, there is the application to the detection of failures in dynamic system [1,17,18,20,22]. While this is the focus of this paper, it should be emphasized that the references on other applications of MM ideas are of potentially great value to the engineer interested in applying MM to failure detection, since issues such as robustness and MM response characteristics are raised in all of these applications.

In order to give some insight into how MM can be used in failure detection, let us give a number of the most often used models for failures. As we will see these fall naturally into the framework of the MM method as given by (2.1)-(2.4). Assume that we have a normal model of the form

$$x(k+1) = A(k)x(k) + B(k)u(k) + w(k) \tag{2.18}$$

$$y(k) = C(k)x(k) + v(k) \tag{2.19}$$

Actuator failures or dynamic disturbances can be modeled in one of three ways:

- A model identical to (2.18) except for a different B-matrix. For example, setting one column of B to zero can be used to model failure to zero of an actuator.
- The addition of a driving term in eq. (2.18), i.e.

$$x(k+1) = A(k)x(k) + B(k)u(k) + w(k) + g(k) \tag{2.20}$$

- Such a model can be used to model stuck actuators. In [26] it was used to model a leak in a reaction control system.
- A model identical to (2.18), (2.19) except that the covariance of w(k) is increased markedly. This may be used to model erratic disturbances.

Sensor failures can also be modeled in three ways:

- A model identical to (2.18), (2.19) except for a different C-matrix. For example, setting one row of C to zero can be used to model failure to zero of a sensor.
- The addition of a bias term in (2.19), i.e.

$$y(k) = C(k)x(k) + v(k) + b(k) \tag{2.21}$$

- A model identical to (2.18), (2.19) except that the covariance of v(k) is increased in order to model degraded and erratic sensor performance. Specific elements of the covariance may be selectively increased in order to model degradations in particular sensors.

A number of successful applications of MM methods to failure detection problems have appeared in the literature, and all of them have had to deal with one issue that we have not yet discussed. This is that in failure detection we are not simply attempting to determine which of the models given in (2.1)-(2.4) is the correct one, but rather we are trying to detect a shift from one model to another. While this is not directly taken into account in the MM model as described to this point, the MM algorithm often does work for this problem without any major modifications. The important issue in this is the adaptability of MM -- i.e. if a model switch occurs, MM will, theoretically, eventually indicate this. Two things must be taken into account, however

(1) From the probability evolution equation (2.5) we see that if $p_i(k)$ is small, then $p_i(k+1)$ will grow only slowly at best. In practice we have found that numerical roundoff often leads to a $p_i(k)$ being set to zero. In this case $p_i(j)$

will be zero for all j>k. In order to avoid this drastic effect and also the extreme sluggish response of MM to a change in models, a lower bound is usually set on the $p_i(k)$. In different applications we have found bounds from 10^{-3} down to 10^{-15} to be satisfactory, with very little sensitivity to the precise value of of the bound.

(2) If a particular model is not correct up untiltime k the Kalman filter based on this model may develop large errors. If then this model becomes correct at time k, it may take a long time before the prediction errors (2.8) decrease to reflect the validity of the model. From (2.13) and (2.5) we see that this in turn means that MM may not respond to this change for some time. In practice we have found that this is not a particularly bad problem if the errors in all of the Kalman filters remain bounded even when the model on which they are based is incorrect. If a particular real system-mismatched Kalman filter combination is unstable, then there may be problems if the system switches to the model corresponding to this filter. What we have found is a workable solution to this problem is to reset the estimates of potentially divergent Kalman filters to the estimate of the most probable model, and this is done whenever the probability of possibly diverging filters falls below a threshold (such as 10^{-2}).

With these modification MM will respond more quickly to model changes. Whether it responds fast enough is a question that depends on the application. If fast response is needed for control purposes or because additional model shifts are possible, then one may wish to consider a problem formulation that explicitly includes model switches. In the next section we describe one such formulation, and in the remainder of this section we indicate how the MM formulation can be modified to incorporate model changes and what the cost is for this modification.

Specifically assume now that at every time k, the real system corresponds to one of the models in (2.1)-(2.4) but that the model may change from time to time. Clearly there are several different constraints that we can play on the possible sequences of models. For example, if there are no constraints, then there are N^{k+1} possible sequences of models over the first k time steps (any of N at t=0, any of N at t=1,...). On the other hand, a much more reasonable model (assuming that we are looking for single failures which are sufficiently separated in time so that they can be detected and accounted for separately), would be to allow only those sequences that start with one particular model (the "normal" model) and have a single shift to any of the other failure models. In this case there are (kN-k+1) possible sequences up to time k -- essentially we must account for all possible failure times.

The MM solution for any such set of possible sequences of models is conceptually identical to that discussed previously, except here in principle we must design a Kalman filter for each allowable sequence of models. The residuals from these filters are then used exactly as described earlier to compute the probabilities for all hypothesized sequences. Since the number of possible sequences and thus filters grows in time, some method for pruning the tree of hypotheses is needed. For example, we can think of throwing away very unlikely models. A variety of techniques for handling such MM trees have been considered in the literature [19-22, 27]. While this may at first glance appear to be a hopelessly complex solution to the failure detection problem, this approach is not without merit. Specifically, as in [20] this approach often provides a great deal of insight into the structure of a failure detection problem. Also, the implementation of Kalman filter trees is not only within the realm of feasibility for implementation using high speed digital hardware, but it is also unavoidable in problems such as multi-object tracking and tracking maneuvering objects [22,23,27] in which keeping track of large number of possibilities is crucial.

III. The Generalized Likelihood Ratio Method

The generalized likelihood ratio (GLR) method deals with a formulation similar to that for MM, but different enough so that the structure of the solution is quite different. The starting point for GLR is a model describing normal operation of the observed signals or of the system from which they come. Abrupt changes are then modeled as additive disturbances to this model that begin at unknown times. As just discussed for MM, we will look at the case of a single such change, the assumption being that abrupt changes are sufficiently separated to allow for individual detection and compensation. The solution to the problem just described and applications of the method can be found in [1,2,4,6,23,28,29,30]. In this section we outline the basic ideas behind the technique and discuss some of its properties.

We assume that the system under consideration can be modeled as

$$x(k+1) = A(k)x(k) + B(k)u(k) + w(k) + f_i(k,\theta)\nu \tag{3.1}$$

$$y(k) = C(k)x(k) + v(k) + g_i(k,\theta)\nu \tag{3.2}$$

where the normal model consists of these equations without the f_i and g_i terms. These terms, $f_i(k,\theta)\nu$ and $g_i(k,\theta)\nu$, represent the presence of the ith type of failure mode, i=1,...,N. Here θ is the unknown time at which the failure occurs (so $f_i(k,\theta) = g_i(k,\theta)=0$ for $k<\theta$), and f_i and g_i are the specified dynamic profiles of the ith failure mode. For example, if $f_i=0$ and g_i=a vector whose components are all zero except for the jth one which equals 1 for $k\geq\theta$, then this failure mode corresponds to the onset of a bias in the jth component of y. Finally, the scalar ν denotes the magnitude of the failure (e.g. the size of the bias) which we can model as known (as in MM and as in what is called simplified GLR (SGLR)) or unknown.

Assume that we design a Kalman filter based on normal operation, i.e. by neglecting f_i and g_i. From the previous section we have that this filter is given by

$$\hat{x}(k+1|k) = A(k)\hat{x}(k|k) + B(k)u(k) \tag{3.3}$$

$$\hat{x}(k+1|k+1) = \hat{x}(k+1|k) + K(k+1)\gamma(k+1) \tag{3.4}$$

$$\gamma(k+1) = y(k+1) - C(k)\hat{x}(k+1|k) \tag{3.5}$$

where K, P, and V are calculated as in (2.9)-(2.12). Suppose now that a type i failure of size ν occurs at time θ. Then, because of the linearity of (3.1)-(3.5) we can write

$$x(k) = x_N(k) + \alpha_i(k,\theta)\nu \tag{3.6}$$

$$\hat{x}(k|k) = \hat{x}_N(k|k) + \beta_i(k,\theta)\nu \tag{3.7}$$

$$\hat{x}(k+1|k) = \hat{x}_N(k+1|k) + \mu_i(k+1,\theta)\nu \tag{3.8}$$

$$\gamma(k) = \gamma_N(k) + \rho_i(k,\theta)\nu \tag{3.9}$$

where x_N, $\hat{x}_N$, and γ_N are the responses if no failure is present and the other terms are the responses due solely to the failure. Straightforward calculations yield recursive equations for these quantities:

$$\alpha_i(k+1,\theta) = A(k)\alpha_i(k,\theta) + f_i(k,\theta), \quad \alpha_i(\theta,\theta)=0 \tag{3.10}$$

$$\beta_i(k+1,\theta) = [I-K(k+1)C(k+1)]\mu_i(k+1,\theta) + K(k+1)[C(k+1)\alpha_i(k+1,\theta) + g_i(k+1,\theta)\nu] \tag{3.11}$$

$$\mu_i(k+1,\theta) = A(k)\beta_i(k,\theta), \quad \beta_i(\theta-1,\theta) = 0 \tag{3.12}$$

$$\rho_i(k,\theta) = C(k)[\alpha_i(k,\theta)-\mu_i(k,\theta)] + g_i(k,\theta) \tag{3.13}$$

The important point about these quantities is that they can be precomputed. Furthermore, by its definition, $\gamma_N(k)$ is the innovations under normal conditions, i.e. it is zero-mean, white, Gaussian with covariance V(k). Thus we now have a standard detection problem in white noise: we observe the filter residuals $\gamma(k)$, which can be modeled as in (3.9), and we want to detect the presence of a failure (i.e. that $k>\theta$) and perhaps determine its identity (i) and estimate its time of occurrence θ and size ν, if this is modeled as being unknown. The solution to this problem involves matched filtering operations. First, define the precomputable quantities

$$a(k,\theta,i) = \sum_{j=\theta}^{k} \rho_i'(j,\theta)V^{-1}(j)\rho_i(j,\theta) \tag{3.14}$$

This has the interpretation as the amount of information present in $y(\theta),..,y(k)$ about a type i failure occurring at time θ.

The on-line GLR calculations consist of the calculation of

$$d(k,\theta,i) = \sum_{j=\theta}^{k} \rho_i'(j,\theta)V^{-1}(j)\gamma(j) \tag{3.15}$$

which are essentially correlations of the observed residuals with the failure signatures $\rho_i(j,\theta)$ for different hypothesized failure types, i, and times, θ. If ν is known (the SGLR case), then the likelihood of a type i failure having occurred at time θ given data $y(1),...,y(k)$ is

$$\ell_s(k,\theta,i) = 2\nu d(k,\theta,i) - \nu^2 a(k,\theta,i) \tag{3.16}$$

If ν is unknown, then the generalized likelihood of this failure is

$$\ell(k,\theta,i) = \frac{d^2(k,\theta,i)}{a(k,\theta,i)} \tag{3.17}$$

and the maximum likelihood estimate of ν assuming a failure of type i at time θ is

$$\hat{\nu}(k,\theta,i) = \frac{d(k,\theta,i)}{a(k,\theta,i)}$$

Thus the GLR algorithm consists of the single Kalman filter (3.3)-(3.5), the matched filter operations of (3.15), and the likelihood calculation of (3.16) or (3.17). The outputs of the method are these likelihoods and the estimates of eq. (3.18) if ν is modeled as unknown. The basic idea behind GLR is that different types of abrupt changes produce different kinds of effects on the filter innovations -- i.e. different signatures -- and GLR calculates the likelihood of each possible event by correlating the innovations with the corresponding signature.

As with the MM method a number of issues can be raised about GLR. Some of these, such as the effect of nonlinearities (Question 1) and robustness to model errors (Question 2), are very similar to the MM case. Essentially it still can be viewed as a signal-to-noise ratio problem: in the nonlinear case the additive decomposition of (3.9) is not precisely valid, but it may be approximately correct. Also,

different failure modes can be distinguished even in the presence of modelling errors if their signatures are different enough. Again these issues depend very much on the particular application. We refer the reader to [3,5-7,23,29,30] for discussions of several applications of GLR to applications in which these issues had to be addressed.

The third question considered in Section II was the possible applications of MM. Since GLR is directly aimed at detecting abrupt changes, its applications are restricted to problems involving such changes, such as failure detection [1-3,30], detecting arrhythmias in electrocardiograms [5,28], maneuver detection [23], etc. Note that the model used in (3.1), (3.2) for such changes is an additive model. Thus it appears on the surface that the types of abrupt changes that can be detected by GLR are a special subset of those that can be detected by MM, since (2.1), (2.2) allow parametric changes (in A,B,C,Q,R) as well as additive ones. There are several points, however, that must be taken into account in assessing and comparing MM and GLR:

(1) The price one pays for allowing parametric changes in MM is the necessity of implementing banks of Kalman filters, and actually trees of such filters to account for switches between models. GLR, on the other hand, requires a single Kalman filter and a growing number of correlation calculations as in (3.15), which in principle must be calculated for i=1,...,N and θ=1,...,k. We will comment shortly on the computational issues concerned with these correlations, but for now we simply point out that they are typically far less involved than the calculations inherent in Kalman filters (see [3,5] for examples of how simple these calculations can be). Also, because it operates on the outputs of a normal mode filter, GLR can be easily implemented and attached as a monitor to an already existing system.

(2) Extensions to the GLR method can be developed for the detection of parametric changes [31]. This extended GLR bears some similarity to extended Kalman filtering and iterated extended Kalman filtering.

(3) It has been our experience that a GLR system based on the detection of additive effects can often also detect parametric failures. For example, a gain change in a sensor does look like a sensor bias, albeit one that is modulated by the value of the variable being sensed. That is, any detectable failure will exhibit a systematic deviation between what is observed and what is predicted to be observed. Obviously, the ability of GLR to detect a parametric failure when it is looking for additives ones is again a question of robustness. If the effect of the parametric failure is "close enough" to that of the additive one, the system will work. This has been the case in all of our experience. In particular we refer the reader to [3] for a an additive-failure-based design that has done extremely well in detecting gain changes in sensors. Note of course that in this mode GLR is essentially only indicating an alarm -- i.e. the estimate $\hat{\nu}$ of the "bias" is meaningless, but in most failure detection problems our primary interest is in simply identifying which of several instruments has failed.

There are two final issues that should be mentioned in discussing GLR. The first concerns the calculation of statistical measures of performance of GLR. As mentioned in the preceding section, Baram [14] has developed a method for measuring the distance between models and hence a measure of the detectability and distinguishability of different failure modes. Similar calculations can be performed for GLR, but in this case it is actually simpler to do and interpret, as we can use standard detection-theoretic ideas. Specifically, a direct measure of the detectability of a failure mode is the information $a(k,\theta,i)$ defined in (3.14). This quantity can be viewed as the correlation of $\rho_i(j,\theta)$ with itself at zero lag. Similarly, we can determine the relative distinguishability of a type i failure at two times θ_1 and θ_2 as the correlation of the corresponding signatures

$$a(k,\theta_1,\theta_2,i) = \sum_{j=\max(\theta_1,\theta_2)}^{k} \rho_i'(j,\theta_1)V^{-1}(j)\rho_i(j,\theta_2) \tag{3.19}$$

and the relative distinguishability of type i and m failures at times θ_1 and θ_2 similarly:

$$a(k,\theta_1,\theta_2,i,m) = \sum_{j=\max(\theta_1,\theta_2)}^{k} \rho_i'(j,\theta_1)V^{-1}(j)\rho_m(j,\theta_2) \tag{3.20}$$

These quantities directly provide us with information about how system redundancy is used to detect and distinguish failures and can be used in deciding whether additional redundancy (e.g. more sensors) are needed. Also, the quantities in (3.14), (3.19), and (3.20) directly give the statistics of the likelihood measures (3.16), (3.17). For the SGLR case of (3.16), ℓ_s is Gaussian, and its mean under no failure is $-\nu^2 a(k,\theta,i)$, while if a type m failure occurs at time ϕ, its mean is

$$E[\ell_s(k,\theta,i)\,|\,(m,\phi)] = \nu^2[2a(k,\theta,\phi,i,m)-a(k,\theta,i)] \tag{3.21}$$

For example if $(m,\ell) = (i,\theta)$ -- i.e. if the precise failure and time assumed in the calculation of $\ell_s(k,\theta,i)$ are true, then its mean is $+\nu^2 a(k,\theta,i)$. In the case of (3.17), under no failure $\ell(k,\theta,i)$ is a chi-squared random variable with 1 degree of freedom, while if a failure (m,ϕ) of size ν occurs $\ell(k,\theta,i)$ is non-central chi-squared with mean

$$E[\ell(k,\theta,i)\,|\,(m,\phi)] = 1 + \frac{\nu^2 a(k,\theta,\phi,i,m)^2}{a(k,\theta,i)} \tag{3.22}$$

Clearly these quantities can be very useful in evaluating the performance of GLR failure detection algorithms and for determining decision rules based on the GLR outputs. We will say a bit more about both topics in Section V and refer the reader to [30,32] for more details.

The other issue to be mentioned is the pruning of the tree of possibilities. As in the MM case in principle we have a growing number of calculations to perform, as $d(k,\theta,i)$ must be calculated for i=1,..,N and all possible failure times up to the present, i.e. θ=1,...,k. What is usually done is to look only over a sliding window of possible times:

$$k-M_1 \leq \theta \leq k-M_2 \tag{3.23}$$

where M_1 and M_2 are chosen based on the a's -- i.e. on detectability and distinguishability considerations. Basically after M_2 time steps from the onset of failure we have collected enough information so that we may make a detection with a reasonable amount of accuracy. Further, after M_1 time steps we will have collected a sufficient amount of information so that detection performance is as good as it can be (i.e. there is no point in waiting any longer). Clearly we want M_1, M_2 large to allow for maximum information collection, but we want them small for fast response. This is the typical probability of error - decision delay tradeoff that arises in all failure detection problems.

IV. Robust Comparison Residual Generation

Underlying both the GLR and MM methods is the issue of using system redundancy to generate comparison signals that can be used for the detection of failures. For MM these signals are the several innovations processes, while for GLR the single normal filter innovations can be viewed as a set of comparison signals. Moreover, all failure detection systems (including, for example, all those discussed in [1]) can be viewed as consisting of a comparison signal generation system, followed by a decision mechanism based on the comparison signals. In the next section we will say something about the decision mechanism aspect of failure detection. In this section we briefly examine the issue of generating comparison signals. Both of these topics are considered in far greater detail in [32].

The fundamental idea involved in finding comparison signals is to use system redundancy, i.e. known relationships among measured variables to generate signals which are small under normal operation and which display predictable patterns when particular anomalies develop. For example, if we have three identical sensors, $y_1(k)$, $y_2(k)$, $y_3(k)$, we can vote, i.e. we can examine the pair of comparison signals

$$\begin{aligned} &y_1(k) - y_2(k) \\ &y_2(k) - y_3(k) \end{aligned} \tag{4.1}$$

If both are systematically large and of opposite sign, we know y_2 has failed, if the first is large and the second small, y_1 has failed, etc.

As a second example suppose $y_1(k)$ and $y_2(k)$ measure the same quantity and that $y_3(k)$ measures α_1 times the value of this quantity at time k plus α_2 times this quantity at time k-1. Then two possible comparison signals are

$$\begin{aligned} &y_1(k) - y_2(k) \\ &\alpha_1 y_2(k) + \alpha_2 y_2(k-1) - y_3(k) \end{aligned} \tag{4.2}$$

As a third example, suppose $y_1(k)$ measures a variable that is supposed to be constant. Then, a comparison signal for the detection of failures in $y_1(k)$ is

$$y_1(k) - \frac{1}{k-1} \sum_{j=1}^{k-1} y_1(j) \tag{4.3}$$

These three examples illustrate the basic kinds of redundancy. The first case is an example of direct redundancy, while the other two are examples of functional redundancy. In the first of these we are comparing disimilar instruments by using relationships among the variables they measure. In the second of these we use a description of the temporal evolution of a variable that is measured. Most failure detection systems that have been used in practice have been based on a careful exploitation of one or more of these types of redundancy. GLR and MM represent two general approaches to using all available redundancy. This has some benefits, but it also raises some questions that must be considered in the design process.

Suppose we have three instruments that are supposed to be measuring the same variable. In the GLR and MM contests this would be equivalent to a model of the form

$$x(k+1) = A(k)x(k) + B(k)u(k) + w(k)$$

$$\begin{pmatrix} y_1(k) \\ y_2(k) \\ y_3(k) \end{pmatrix} = \begin{pmatrix} -c'- \\ -c'- \\ -c' \end{pmatrix} x(k) + \begin{pmatrix} v_1(k) \\ v_2(k) \\ v_3(k) \end{pmatrix} \tag{4.4}$$

The MM and GLR algorithms then compare the relative tracking accuracy of this model with models based on

assuming that each of the three components of y develops a fault. These models may involve failure to zero (for MM) or the onset of a bias (for GLR) or some other, similar model. The important point to note, however, can be seen by comparing (4.1) and (4.4). The comparison signals (4.1) do measure relative "tracking" accuracy in a relatively primitive sense in that they use no information about the evolution of the variable being measured. Specifically, if the model (4.4) is considered with A=B=0, i.e. so that x(k) is white noise, then MM and GLR closely resemble straightforward voting systems in that they also use no information about the evolution of x(k).

Thus voting systems that use comparison signals in (4.1) are essentially degenerate versions of MM or GLR systems. Does this mean that MM and GLR are better? The answer, of course, is maybe, but what <u>is</u> true is that MM and GLR allow for the examination of a variety of methods for generating comparison signals, based on whatever type of redundancy is presented and is used. To gain some insight into this issue, let us consider the question of when and why voting systems may or may not be useful. The first point is that voting systems require at least triplex redundancy. Furthermore, if we want a system that is n fail-operational and <u>still</u> uses only voting, we need n+3 of each instrument type. The reason for this is, of course that voting systems neglect <u>all</u> functional redundancy. Another problem with voting systems is that subtle instrument degradations may not be detected by the coarse comparison signals of (4.1), and neither will effects (such as those caused by temperature change) that affect all instruments. Detection of such changes will require the use of some analytic information. On the other hand, voting systems are: (a) simple to implement, and (b) robust. It is this latter point that is the key one for our discussion.

Specifically, all failure detection is based on analytical relationships between sensed variables, including voting methods, which assume that sensors measure precisely the same variable. However, in order to develop a reliable failure detection technique, we must assess the uncertainty in our knowledge of any analytical relationship. For example, if three identical accelerometers are mounted on a vehicle but one of them is slightly misaligned with respect to the other two, then this misalignment will cause the comparison signals in (4.1) to have a component due solely to this misalignment. Similarly, if the variable being sensed by $y_1(k)$ is not exactly constant then these will be spurious components to the comparison signal (4.3). Also, if α_1 and α_2 are not known precisely, then the same can be said for (4.2).

While this can be a problem for voting systems, it can be even more of one for failure detection methods based on analytical redundancy. Specifically, compare the type of error in the comparison signals of a voting system to that in a comparison signal such as

$$(y_1(k)-y_1(k_0)) + \alpha \sum_{j=k_0}^{k-1} y_2(j) \tag{4.5}$$

Here y_1 might be the output of a velocity sensor and y_2 would then be the output of an accelerometer. The constant of proportionality α might arise, for example, if y_1 is really a mach meter, in which case α would be the inverse of the speed of sound (see [3] for a more involved version of this example). If this constant is not known precisely, then the effect of this accumulates in time in the comparison signal (4.5) because of the integration (summation) to take into account the functional relationship between y_1 and y_2. This is in direct contrast to the memoryless voting-like signal

$$y_1(k) - \alpha y_2(k) \tag{4.6}$$

in which errors in α don't lead to growing errors.

Thus we see the tradeoff in using analytical relationships. On the one hand, if we can use such relationships we can reduce hardware redundancy and maintain the same level of fail-operability. In addition analytical redundancy allows us to extract more information from the data, and thus we can detect subtler changes in system component characteristics. On the other hand, the use of this information can cause problems if there are large uncertainties in the parameters specifying the analytical relationship. Thus, in applying MM and GLR care must be taken in deciding what analytical information can be used and how it should be used. What this in fact implies is that for practical implementation of MM- or GLR-based detection systems one should <u>not</u> use one, absolutely complete dynamical model to describe the entire system, all possible system redundancies and all possible failure modes. Rather, one should attempt to isolate the most reliable types of redundancy available for <u>each</u> hypothesized failure mode or for groups of such modes, and should design a separate reduced order GLR or MM system based on each of these sets of redundant information and corresponding failure set. An excellent example of this is given in [3], in which sensor failure detection for an aircraft is performed. For each sensor different analytical relationships are found and tests are designed to use this redundancy and also to reflect our uncertainty in the parameters that specify the redundancy relationships. For example, mach meter angle of attack, and sideslip angle measurements are compared to accelerometer outputs through basically a vector version of (4.5), i.e. through the kinematical relationship

$$\dot{v} = a \tag{4.7}$$

Although there are uncertainties in this relationship (due to wind, misalignment errors, uncertainties in our knowledge of our orientation with respect to the earth), they are far less severe than in the dynamical relationship

$$a = f(M,\alpha,\beta,h,\delta,T) \tag{4.8}$$

relating acceleration to mach, angle of attach, sideslip, altitude, control surface positions, and thrust. Thus (4.7) was used in [3] for detection of all types and sizes of accelerometer failures, while (4.8) was used in a supplementary fashion only to detect large accelerometer failures quickly.

Based on the approach in [3] we have been working on a general methodology for breaking a system into smaller systems that can be used for reliable failure detection. This is described in detail in [32], and we simply outline the basic ideas here. We will restrict our attention to the detection of sensor failures. Suppose we have a linear system (under normal conditions) parametrized by a vector η of unknown parameters

$$x(k+1) = A(\eta)x(k) + w(k) \tag{4.9}$$

$$y(k) = C(\eta)x(k) + v(k) \tag{4.10}$$

Let the vector of sensed quantities be denoted by

$$z(k) = C(\eta)x(k) = \begin{bmatrix} z_1(k) \\ \vdots \\ z_m(k) \end{bmatrix} \tag{4.11}$$

and let the components of $y(k)$ be denoted by $y_1(k),\ldots,y_m(k)$. Let S denote the set of possible values of η.

Suppose now that we are interested in determining a redundant relationship that can be used for detecting failures in $y_1(k)$. Suppose further that we know the general nature of this relationship but not its details. Specifically, suppose that we know that a relationship of the form

$$\alpha_0'z(k)+\alpha_1'z(k-1) +\ldots+ \alpha_M'z(k-M) = 0 \tag{4.12}$$

holds when $w=0$, where the first component of α_0 equals one and the remaining coefficients in $\alpha_0,\ldots,\alpha_M$ are unknown and in general depend upon the unknown vector η. Note that (4.12) essentially says that

$$z_1(k) = -\sum_{j=2}^{m} \alpha_{0j}z_j(k) - \sum_{\ell=1}^{M}\sum_{j=1}^{m} \alpha_{\ell j}z_j(k-\ell) \tag{4.13}$$

Here

$$\alpha_\ell = \begin{pmatrix} \alpha_{\ell 1} \\ \vdots \\ \alpha_{\ell m} \end{pmatrix}$$

In this case, we have an auto-regressive, moving average (ARMA) model for $z_1(k)$.

Note that if we know the α's, we can think of replacing the z's by the y's and thus will obtain a comparison signal that generalizes those considered earlier in this section. Moreover, we can also consider using GLR or MM, which generate essentially closed-loop versions of these comparison signals (by using a Kalman filter we reduce the accumulative effect of parameter errors that can occur directly from the use of (4.13) and also the effects of w.) Specifically, assuming that we know the α's, we can use (4.10), (4.11) to write

$$z_1(k) = -\sum_{j=2}^{m} \alpha_{0j}y_j(k) - \sum_{\ell=1}^{M}\sum_{j=1}^{m} \alpha_{\ell j}y_j(k-\ell) + N(k) \tag{4.14}$$

$$y_1(k) = z_1(k) + v(k) \tag{4.15}$$

where the process noise is

$$N(k) = \sum_{j=2}^{m} \alpha_{0j}v_j(k) + \sum_{\ell=1}^{M}\sum_{j=1}^{m} \alpha_{\ell j}v_j(k-\ell) \tag{4.16}$$

Note that this model can be put into state space form. It does, however, have colored process noise which is also correlated with the measurement noise. These issues notwithstanding, one can determine a state model, which can then be used as the basis for a MM or GLR failure detection algorithm for failures in y_1. Note from (4.14) and (4.15) that in this model such a failure affects <u>both</u> the dynamics and the measurement equations.

Let us now return to the question of the choice of the α's. Clearly one would like to choose them to minimize the component in the left-hand side of (4.12) due to modeling errors, i.e. to uncertainty in our knowledge of η. This leads directly to the following min-max optimization problem. Let $K(\tau;\eta)$ denote the steady-state covariance of z assuming a particular value of η. This is given by the following equations

$$M(\eta) = A(\eta)M(\eta)A'(\eta) + Q \tag{4.17}$$

where Q is the variance of w in (4.9). Then

$$K(\tau;\eta) = C(\eta)A^{\tau}(\eta)M(\eta)C'(\eta), \quad \tau\geq 0 \tag{4.18}$$

$$K(\tau,\eta) = K'(-\tau,\eta) \quad \tau\leq 0 \tag{4.19}$$

Suppose now we look at the second moment of the left-hand side of (4.12), with the α's fixed and assuming a particular value of η:

$$E\left[\left(\sum_{j=0}^{M} \alpha_j^2 z(k-j)\right)^2\right] \triangleq J(\alpha,\eta) = \sum_{\ell,j=0}^{m} \alpha_j' K(\ell-j;\eta)\alpha_\ell \tag{4.20}$$

Then a measure of how much error might arise in (4.12), (4.13), or (4.14)-(4.15) from model errors for a particular set of α's is

$$\max_{\eta \in S} J(\alpha,\eta) \tag{4.21}$$

which leads directly to a natural criterion for the choice of the α's. They should be chosen to be the solution of the min-max problem

$$\min_{\alpha} \max_{\eta \in S} J(\alpha,\eta) \tag{4.22}$$

A detailed discussion of this approach is given in [32].

V. Sequential Decision Rules

As mentioned in the preceding section, the second part of a failure detection algorithm is the decision rule that uses the available comparison signals to make decisions on the interruption of normal operation by the declaration of failures. The MM and GLR methods as described in Sections II and III go part of the way toward making these decisions by generating "maximally informative statistics," i.e. probabilities and likelihood ratios, but we have yet to indicate how these numbers are used in a detection mechanism. Clearly a number of issues must be considered in designing such a mechanism. Performance characteristics such as expected delay until detection, false alarm rate, and probabilities of incorrect failure identification must be considered. This can often be done by establishing a structure for a rule and then optimizing the parameters in this structure with respect to a performance index consisting of a weighted combination of measures such as those mentioned above. For example, MM failure detection is often performed by declaring an alarm the first time the probability of a failure model exceeds a specified threshold. GLR detection is often done by choosing $\hat{\mathfrak{i}}(k)$, $\hat{\theta}(k)$ to maximize $\ell_s(k,\theta,i)$ or $\ell(k,\theta,i)$ and then by declaring a failure of type $\hat{\mathfrak{i}}(k)$ at time $\hat{\theta}(k)$ if

$$\ell(k,\hat{\theta}(k),\hat{\mathfrak{i}}(k)) > \text{Threshold} \tag{5.1}$$

Also, as discussed in several papers [6,25,30], one may include a persistence requirement -- i.e. a probability or likelihood must stay above a threshold for a period of time -- to reduce false alarms and to achieve better performance in general.

All of these approaches are aimed at incorporating the tradeoffs inherent in failure detection, but the methods described above do this in an ad hoc manner. This is not to say that they aren't useful, but rather that they must be applied carefully in each application in order to obtain the desired performance tradeoffs. One advantage of these methods is that the decision rules -- maximize and compare to a threshold -- are simple, while the main disadvantage is that the rule does not explicitly reflect the desired tradeoffs. There is an alternative approach that has exactly the opposite properties -- i.e. it allows for a direct incorporation of performance tradeoffs but is extremely complex. This is the Bayesian Sequential Decision approach described in [32], in which an algorithm for the calculation of approximate Bayes decision rules is described. In this section we briefly outline the basic ideas behind this formulation and then describe the one special case that leads to great simplifications and which has been used in many applications including failure detection [3].

We assume that we have a vector comparison signal of the SGLR type (where for simplicity we absorb ν into the signature):

$$\gamma(k) = \rho_i(k,\theta) + \tilde{\gamma}(k) \tag{5.2}$$

where $\tilde{\gamma}(k)$ is zero-mean, white Gaussian noise with covariance $V(k)$. Here for $i=0$, $\rho=0$, corresponding to normal operation, and $i=1,\ldots,N$ denote the possible failure modes. As before, θ is the time of failure. The problem is to detect a failure as quickly as possible but with reasonable false alarm rates. The way this tradeoff is included in the Bayesian formulation is as follows. Our decision process consists of two parts. We first must decide if it is time to stop and make a declaration or if we should continue monitoring. If we decide to stop, we then must decide what failure to declare. The first part of this is called the stopping rule, while the second is the terminal decision rule.

Corresponding to the two parts of the decision rule are two parts to the performance measure. Let

$$L(k,(j,\phi),(i,\theta)) \tag{5.3}$$

denote the cost incurred if we stop at time k and make the declaration that a type j failure occurred at time ϕ when in reality a type i failure occurred at time θ. Here, if $i=0$, then θ is irrelevant, but we use the general notation of (5.3) for simplicity. Note that this cost allows us to include a wide variety of performance measures. For example, $L(k,(j,\phi),(0,-))$ is the cost of a false alarm and $L(k,(j,\phi),(i,\theta))$ for $j \neq i$ is the cost of an incorrect identification. Note that if we are really not interested in determining the failure time, we can consider a special case of the cost function of equation (5.3) in which

the cost essentially doesn't depend on ϕ and θ, but only on i and j. Note, however that for $\theta>k$, $L(k,(j,\phi),(i,\theta))$ is essentially the cost of a false alarm, since we are declaring a failure before it occurs. See [32] for a detailed discussion of this cost function.

The second part of the performance measure for the Bayes Sequential Decision problem is the cost of deferring a decision. Let

$$b(k,(i,\theta)) \tag{5.4}$$

denote the cost of deferring a decision at least until time k, when a type i failure has occurred at time θ. Clearly we will take

$$b(k,(0,-))=0 \tag{5.5}$$

and

$$b(k,(i,\theta))=0 \qquad \text{for} \quad k<\theta \tag{5.6}$$

since there should be no cost for waiting if nothing has actually happened as yet.

Given this framework, the Bayes Sequential Decision Problem is to choose a stopping rule and terminal decision rule to minimize the total expected cost, i.e. the sum of the expected terminal decision cost (5.3) and the expected cost (5.6) that is accrued before we stop. These rules operate on the observed process $\gamma(k)$ of equation (5.2) and at time k one can view the Bayes rule as partitioning the space of possible observation sequences $(\gamma(1),\ldots,\gamma(k))$ into a number of regions. For example, if we are not interested in determining failure time but only failure type, there are N+1 regions. If the actually observed set of observations falls into a particular one of these we will defer decision and continue monitoring, while the other N regions correspond to regions in which we stop and declare one of the N failure types.

Therefore the solution to the Bayes problem requires determining these regions. As discussed in [32], one can alternately, and more profitably, view the problem as one of determining the regions in the space of possible likelihood ratio sequences $(\ell_s(k,\theta,i))$. In this formulation the GLR algorithm can be viewed as a system that generates a sequence of statistics which can be used directly for decision-making. The problem, of course is to determine the regions, and in general this is very complex. The reason for this is that in order to make a decision to stop at time k, we must compare the expected cost of stopping and deciding now, based on the data in hand, i.e. $\gamma(1),\ldots,\gamma(k)$, versus the additional cost of waiting at least one more time step. This second cost involves estimating the decision cost we will incur in the future based on the present data. Thus the stopping decision is intuitively one of deciding if we have enough information already or if we can afford to (if b is small and we expect a great deal of information in the future) or should (if the expected cost of stopping now is large) defer decision. Performing the expectations required in making this comparison of options is a complex procedure. An algorithm is described in [32] for the approximate solution of the Bayes problem for an important class of problems. In the remainder of this section we describe one special case in which the solution can be found explicitly.

This special case is known as the sequential probability ratio test (SPRT), and it deals with the problem when N=1, i.e. when only a single failure mode is being looked for, and when $\theta=0$, i.e., if a failure is present it was present from the initial point of the observation interval. While this is a somewhat restrictive example, it does have many practical applications to failure detection problems. For example, if we follow the methodology outlined in the preceding section, we will isolate different types of analytical redundancy for different possible failure modes. Therefore, we may have several possible comparison signals, each of which is used for the detection of a single failure type. For example, this is precisely what was done in [3]. Therefore, the restriction that N=1, while eliminating some applications, does include many important problems. The restriction that $\theta=0$ eliminates one of the principal sources of complexity in the Bayesian formulation, and that is the growing number of possibilities. In some problems this restriction may cause problems, while in other applications it won't. For example in [3] a system with dual redundancy was considered. By comparing each pair of identical sensors we can determine <u>when</u> a failure has occurred but cannot isolate <u>which</u> instrument has failed without analytical redundancy. Thus we can use the direct comparison to <u>trigger</u> a failure isolation algorithm, based on some form of analytic redundancy that leads to a comparison signal as in (5.2) with $\theta=0$, since we know when a failure may have occurred. In addition to this case, there are approaches [33,34] to restarting a SPRT periodically to take into account the possibility of an unknown failure onset time. Therefore, while the SPRT formulation is restrictive, it does have many applications, and it also has a structure that can be used to suggest modifications (such as the reset just mentioned) that can be used to overcome some of its limitations.

The SPRT can actually be developed for a fairly general formulation of the statistical description of $\gamma(k)$ under the no failure hypothesis H_0 and the failure hypothesis H_1 (where we have now dropped θ since we are assuming that it equals 0). For a general treatment we refer the reader to Chapter 7 of [35] and to [36]. We will restrict our description to that given in [3]. Specifically, we suppose that under H_0 the scalar observation is given by

$$\gamma(k) = \tilde{\gamma}(k) \tag{5.7}$$

where $\gamma(k)$ is a zero-mean, white Gaussian process with variance σ^2. Under H_1 we assume that

$$\gamma(k) = m + \tilde{\gamma}(k) \tag{5.8}$$

where m is a known, constant bias. Then, the SPRT consists of the calculation of the SGLR statistic (only for $\theta=0$) for this special case

$$\ell_s(k) = \frac{1}{\sigma^2} \sum_{j=1}^{k} (\gamma(k) - \frac{m}{2})m \tag{5.9}$$

The decision rule -- i.e. the decision regions for this statistic consists then of comparing $\ell_s(k)$ to two thresholds a and b (where a<b):

$$\begin{aligned} &\ell_s(k) \leq a \quad \text{declare } H_0 \text{ (no failure)} \\ &a < \ell_s(k) < b \quad \text{defer decision} \\ &b \leq \ell_s(k) \quad \text{declare } H_1 \text{ (failure)} \end{aligned} \tag{5.10}$$

This rule yields the shortest expected time to decision with specified probabilities of false alarm P_F (declaring H_1 when really H_0) and missed alarm P_M (declaring H_0 when really H_1), and the thresholds a and b can be computed directly in terms of P_F and P_M (see [36]).

REFERENCES

1. Willsky, A.S., "A Survey of Several Failure Detection Methods," Automatica, Nov. 1976, pp. 601-611; also in AGARDograph No. 224 on Integrity in Electronic Flight Control Systems.

2. Willsky, A.S. and Jones, H.L., "A Generalized Likelihood Ratio Approach to the Detection and Estimation of Jumps in Linear Systems," IEEE Trans. Automatic Control, Vol. AC-21, Feb. 1976, pp. 108-112.

3. Deckert, J.C., Desai, M.N., Deyst, J.J. and Willsky, A.S., "F8-DFBW Sensor Failure Identification Using Analytic Redundancy," IEEE Trans. on Automatic Control, Vol. AC-22, No. 5, Oct. 1977, pp. 795-803.

4. Gustafson, D.E., Willsky, A.S., Wang, J.-Y., Lancaster, M.C., and Triebwasser, J.H., "ECG/VCG Rhythm Diagnosis Using Statistical Signal Analysis I: Identification of Persistent Rhythmas," IEEE Trans. Biomed. Eng., Vol. BME-25, No. 4, July 1978, pp. 344-353.

5. Gustafson, D.E., Willsky, A.S., Wang, J.-Y., Lancaster, M.C., and Triebwasser, J.H., "ECG/VCG Rhythm Diagnosis Using Statistical Signal Analysis II: Identification of Transient Rhythms," IEEE Trans. Biomed. Eng., Vol. BME-25, No. 4, July 1978, pp. 353-361.

6. Willsky, A.S., Chow, E.Y., Gershwin, S.B., Greene, C.S., Houpt, P.K., and Kurkjian, A.L., "Dynamic Model-Based Techniques for the Detection of Incidents on Freeways," IEEE Trans. Automatic Control, to appear Vol. AC-25, June 1980.

7. Kurkjian, A.L., Gershwin, S.B., Houpt, P.K., Willsky, A.S., Greene, C.S., and Chow, E.Y., "Estimation of Roadway Traffic Density on Freeways Using Presence Detector Data," Transportation Sciences, to appear in 1980.

8. Lainiotis, D.G., "Joint Detection, Estimation, and System Identification," Information and Control, Vol. 19, Aug. 1971, pp. 75-92.

9. Willner, D., Observation and Control of Partially Unknown Systems, Ph.D., Thesis, MIT, May 1973.

10. Magill, D.T., "Optimal Adaptive Estimation of Sampled Processes," IEEE Trans. on Automatic Control, Vol. AC-10, pp. 434-439, Oct. 1965.

11. Lainiotis, D.G., "Partitioning: A Unifying Framework for Adaptive Systems, I: Estimation," Proc. of IEEE, Vol. 64, No. 8, pp. 1126-1142, 1976.

12. Stein, G., "An Approach to the Parameter Adaptive Control Problem," Ph.D. Thesis, Purdue University.

13. Saridis, G.N. and Dao, T.K., "A Learning Approach to the Parameter Self Organizing Control Problem," Automatica, Vol. 8, pp. 587-592.

14. Baram, Y., Information, Consistent Estimation and Dynamic System Identification, Ph.D. Thesis, MIT, Nov. 1976

15. Athans, M., et.al., "The Stochastic Control of the F-8C Aircraft Using a Multiple Model Adaptive Control (MMAC) Method Part I: Equilibrium Flight", IEEE Trans. on Automatic Control, Vol. AC-22, No. 5, pp. 768-780, 1977.

16. Stein, G., et.al., "Adaptive Control Laws for the F-8 Flight Test," IEEE Trans. on Automatic Control, Vol. AC-22, No. 5, pp. 758-768, 1977.

17. Montgomery, R.C. and Caglayan, A.K., "A Self-Reorganizing Digital Flight Control System for Aircraft," AIAA 12th Aerospace Sciences Meeting, Washington, D.C., Jan. 30-Feb. 1, 1974.

18. Montgomery, R.C. and Price, D.B., "Management of Analytical Redundancy in Digital Flight Control Systems for Aircraft," AIAA Mechanics and Control of Flight Conference, Anaheim, Calif., Aug. 5-9, 1974.

19. Buxbaum, P.J. and Haddad, R.A., "Recursive Optimal Estimation for a Class of Nongaussian Processes," Proc. Symp. on Computer Processing in Communications, Polytech. Inst. of Brooklyn, April 8-10, 1969.

20. Willsky, A.S., Deyst, J.J., and Crawford, B.S., "Two Self-Test Methods Applied to an Inertial System Problem," J. Spacecr. Rockets, Vol. 12, No. 7, July 1975, pp. 434-437.

21. Newbold, P.M. and Ho, Y.C., "Detection of Changes in the Characteristics of a Gauss-Markov Process," IEEE Trans. Aerospace Elec. Sys., Vol. AES-4, No. 5, Sept. 1968, pp. 707-718.

22. M. Athans, R.H. Whiting, and M. Gruber, "A Suboptimal Estimation Algorithm with Probabilistic Editing for False Measurements with Applications to Target Tracking with Wake Phenomena," IEEE Trans. Aut. Control, Vol. AC-22, No. 3, June 1977, pp. 372-384.

23. Tenney, R.R., Hebbert, R.S., and Sandell, N.R., "A Tracking Filter for Maneuvering Sources," IEEE Trans. Aut. Control, Vol. AC-22, April 1977, pp. 246-251.

24. Greene, C.S., "An Analysis of the Multiple Model Adaptive Control Algorithm," Ph.D. Dissertation, Report No. ESL-TH-843, M.I.T., Elec. Sys. Lab., Cambridge, Mass., August 1978.

25. Doerschuk, P.C., Gustafson, D.E., and Willsky, A.S., "Multifunctional Upper-Extremity Prosthesis Control Signal Generation Using EMG Signal Processing," Proc. 1979 Joint Aut. Contr. Conf., Denver, Colorado, June 1979.

26. Deyst, J.J. and Deckert, J.C., "RCS Jet Failure Identification for the Space Shuttle," Proc. IFAC 1975, Camb., Mass., Aug. 1975.

27. Keverian, K. and Sandell, N.R., "Multiobject Tracking by Adaptive Hypothesis Testing," Rept. No. LIDS-R-959, Lab. for Inf. and Dec. Systems, M.I.T., Cambridge, Mass. Dec. 1979.

28. Gustafson, D.E., Willsky, A.S., Wang, J.Y., Lancaster, M.C., and Triebwasser, J.H., "A Statistical Approach to Rhythm Diagnosis in Cardiograms," Proc. IEEE, Vol. 65, No. 5, May 1977, pp. 802-804.

29. Mier-Muth, A.M. and Willsky, A.S., "A Sequential Method for Spline Approximation with Variable Knots," International J. of System Science, Vol. 9, No. 9, 1978, pp. 1055-1067.

30. Bueno, R., Chow, E.Y., Dunn, K.-P., Gershwin, S.B., and Willsky, A.S., "Status Report on the Generalized Likelihood Ratio Failure Detection Technique, with Application to the F-8 Aircraft," Proc. 1976 IEEE Conf. on Dec. and Control, Dec. 1976.

31. Willsky, A.S., *"Status Report Number One, on the Development of a Methodology* for the Detection of System Failures and for the Design of Fault-Tolerant Control Systems," Rept. No. ESL-SR-781, Electronic Systems Lab., M.I.T., Cambridge, Mass., Nov. 1977.

32. Chow, E.Y., "A Failure Detection System Design Methodology," Ph.D. Dissertation, M.I.T. Dept. of Elec. Eng. and Comp. Sci., Cambridge, Mass., July 1980.

33. Walker, B.R., "A Semi-Markov Approach to Quantifying the Performance of Fault-Tolerant Systems," Ph.D. Dissertation, M.I.T. Dept. of Aero. and Astro, Cambridge, Mass., July 1980.

34. Chien, T.T., "An Adaptive Technique for a Redundant Sensor Navigation Systems," Ph.D. Dissertation, M.I.T. Dept. of Aero. and Astro., Cambridge, Mass., February 1972.

35. Bertsekas, D.P., Dynamic Programming and Stochastic Control, Academic Press, New York, 1976.

36. Fukunaga, K., Introduction to Statistical Pattern Recognition, Academic Press, New York, 1972.

COMPUTER BASED IN-FLIGHT MONITORING

by
Kjell Folkesson
SAAB-Scania, Aerospace Division
Linkoping, Sweden

SUMMARY

Digital computers are widely used today in flight control systems (FCS). They offer versatile computational capability, and the computer programs are easy to modify. Digital computers are generally very capable in nonlinear operations , which makes possible software mechanization of sophisticated sensor and servo monitor algorithms, as well as extensive computer self test.

This paper describes various computer techniques to monitor flight safety critical FCS components such as sensors, servos and the FCS computer itself.

Flight safety critical FCS sensors are usually redundant. The degree of redundancy is a function of the control authority of the sensors, the stability of the aircraft, and existing back-up arrangements. The redundant sensor outputs are compared in the digital computer. The difference between the signals is checked against a fixed or variable threshold. The sophistication of the sensor monitor algorithm varies depending upon the number of redundant sensors, probability of false failure indication, etc.

The digital FCS computer can be used for servo monitoring in many different ways. The servo configuration usually determines the best monitor solution. In redundant servo configurations, various signals, such as electrical current, differential pressure, velocity, or servo position, can be provided to the digital computer and monitored for failure detection. A cost- efficient solution is to compare the output of a single channel servo to the output of a software model of the servo and use the difference for failure detection.

Because the FCS digital computer is usually a flight safety critical element, it has to be closely monitored. Failures must be detected and isolated with very high confidence. In redundant digital FCS computers, both computer self test and monitoring of the computer outputs are used to detect computer failures. The monitoring can be realized in software or in external hardware. An efficient computer self test, detecting all critical hardware failures, is very desirable, since it makes it possible to build a fail-safe, single-computer flight control system, or a dual- fail operational fly-by-wire system, using only three computers. The digital computer self test consists of a number of independent tests which exercise and check the digital computer. Most of the self test is part of the in-flight resident software. Some of the tests, however, require external hardware.

1. INTRODUCTION

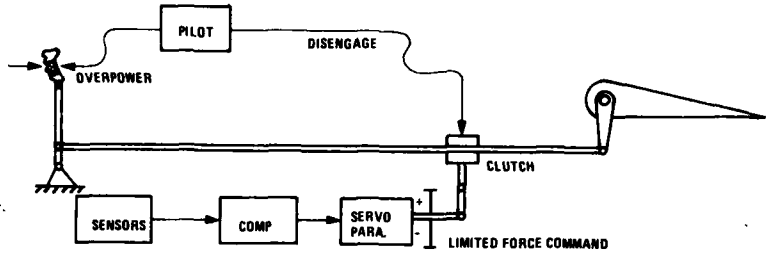
The evolution of automatic flight control systems (AFCS) has come a long way since the first autopilots became available more than fifty years ago. At that time sensors, servos and computers were mostly electromechanical devices. The control laws were simple and the authority of the system generally limited to a level which the pilot could overpower. The pilot thus performed the monitoring function and handled the redundancy management himself. Since the aircraft were fairly stable and well damped the autopilot relief mode with its low bandwidth and parallel servo arrangement was most common. The redundancy management the pilot handled usually was to disengage the parallel servo clutch as indicated in Figure 1A.

As the flight envelope was expanded the aircraft configuration had to be adapted to low speed, subsonic, and supersonic speed requirements and the performance of the unaugmented vehicle was really not satisfactory in any part of the flight envelope. Artificial pitch, roll, and yaw damping as well as pilot command agumentation had to be added to the autopilot modes and the bandwidth and authority of the AFCS was increased far beyond what the pilot could monitor and counteract in case of a failure.

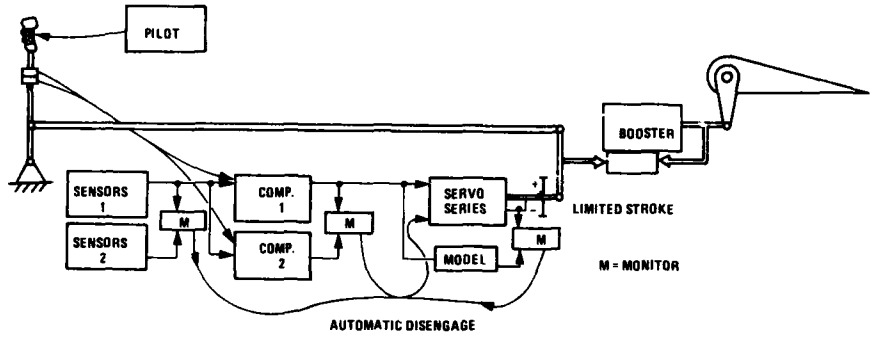
The sensors were still electromechanical. The computer included basically analog components and the high authority flight control servos were arranged in series with the pilot commands. The aerodynamic forces on the control surfaces became so big at high speed that fully hydraulic servo actuators were required to move the control surfaces.

The flight safety risks related to a potential failure in the AFCS made it necessary to design the AFCS with built in safety. This was accomplished by comparing the output of critical system components to the output of similar redundant components or models. Figure 1B illustrates this principle. The result of an AFCS failure means a monitor miscompare, disengagement of the servocommand and centering of the series servo. The pilot may then safely return and land using the mechanical control system although with degraded control performance.

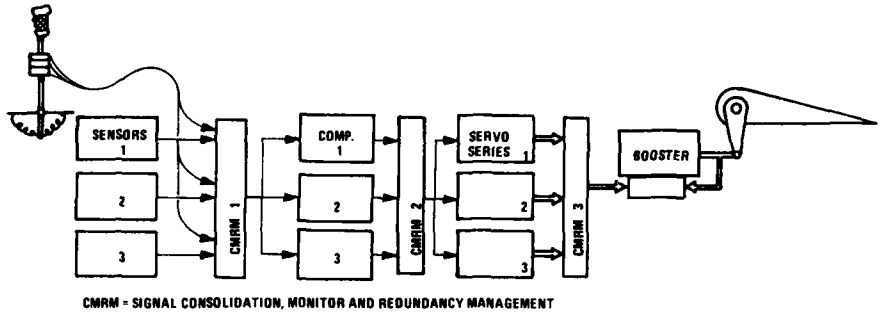
Today aircraft are being developed with active control technology (ACT) designed into the basic airframe. Reduced static stability (RSS) and flutter mode control (FMC) are examples of two ACT areas which make it possible to reduce the wing and tail area and improve the dynamic performance. In the RSS aircraft the stability has to be kept at specified levels by artificial stabilization provided by aircraft motion sensor



a. Autopilot Parallel Servo Mechanization, Pilot Monitored



b. Monitored AFCS with Series Servo



c. Full Authority Triplex EFCS

Figure 1. The History of Failure Management for Flight Control

feedbacks, suitable control laws and full authority series servos. The RSS aircraft may not be flyable without artificial stabilization. This requires very high availability of the stabilizing control functions, usually leading to triplex or quadruplex redundancy. It is then a logical step to replace the old mechanical FCS with an all electrical fly-by-wire (FBW) system, which provides the required artificial stabilization and other ACT modes as well as pilot command augmentation and desirable autopilot modes. The enhanced availability of the pilot flight control system is due to the redundancy. Current requirements are stricter than they used to be for the mechanical FCS.

The all electrical redundant flight control system (EFCS) is generally complex due to the monitoring and redundancy management required to detect and isolate failed system components. There are many ways to mechanize the EFCS. Significant factors to be considered are the requirements on and the qualities of the aircraft itself, the mission time and the risks versus cost the buyer is willing to accept.

The EFCS redundancy and monitoring concepts usually show the system divided in three basic areas

- o sensors
- o computers
- o servos

Each section has its own monitor and redundancy management handling. Figure 1C shows the basic structure of a redundant EFCS. Three blocks perform the signal Consolidation, the Monitoring and the failure Redundancy Management (CMRM). The CMRM 1 and 2 in Figure 1C may include solid state logic or be integrated in computer 1, 2, and 3, especially favorable if 1, 2, and 3 are digital computers. CMRM 3 is very likely a mechanical device using force or position criteria to eliminate failed series servo and select the correct servo command for the booster servo.

In the following sections, this paper will deal with methods for signal consolidation, failure monitoring and redundancy management when failures have occurred in any of the single or redundant EFCS components.

The discussions begin in section 2 with a hypothetical flight control specification. This illustration will serve as the focal point for subsequent sections covering details of individual failure management techniques.

The author recognizes the emergence of new software techniques for hardware failure detection such as analytical redundancy. These techniques will be discussed for sensor monitoring in other papers, however, a similar analytical technique for servo monitoring and loop closure performance enhancement is discussed herein.

2. COMPUTER BASED IN-FLIGHT MONITORING

2.1 Assumptions

Computer based in-flight monitoring will be discussed with reference to the electrical flight control system defined below. It is assumed to be an integral part of a fighter aircraft with reduced static pitch stability (RSS). The EFCS pitch inner loop channel must, therefore, be operational for safe flight.

The EFCS pitch inner loop shall provide artificial stabilization (SAS) and pilot command augmentation (CAS).

The SAS/CAS mode flight qualities will meet level 2 requirements with total and static pressure available and level 3 without pressure information. The assumed sensor, computer and servo redundancy is determined by a desire to achieve a probability of catastrophe due to failure in the EFCS pitch inner loop of about 10^{-8} during one flight hour.

DESCRIPTION OF THE EFCS, PITCH CHANNEL

The reference EFCS pitch channel includes the following system elements.

Quantity	Element
2	Ptot and Pstat sensors (total and static pressures)
3	Stickposition sensors
4	Rate Gyros
3	Digital Computers
2	EFCS Servos
1	Tandem Booster servo

The EFCS pitch SAS/CAS channel functional principle and redundancy levels are shown in Figure 2.

Ptot and Pstat Sensors - The total and static pressure sensors are dual. Sensor set 1 is normally used. They are monitored during flight by comparison and reasonableness tests performed within the digital computers. After a failure the pressure information is disregarded and the loop gain is set constant.

Stick Position Sensors - The pitch stick position sensors are triplex. During flight a mid value select (MVS) algorithm selects which one of the three signals shall be used in the CAS-computations. The sensor signals are monitored during flight by cross comparison. The pilot is informed after a first failure. After a second failure the EFCS and booster servos are centered to a preselected trim position.

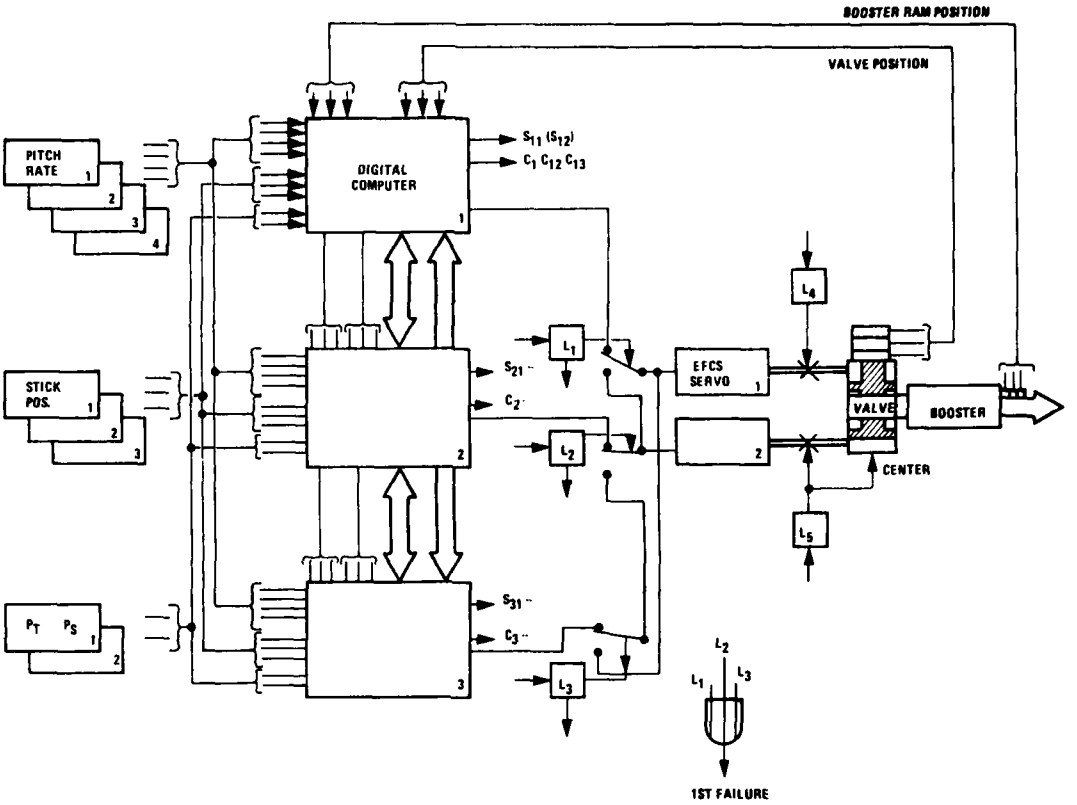
Rate Gyros - The pitch rate gyros are quadruplex because they have relatively high failure probabilities and are difficult to test during flight. Three of the rate gyro signals are used in a fault-free system. Their midvalue is used in the CAS-computations. The fourth rate gyro is active standby, its signal replaces the first failed gyro signal. The three active rate gyros are monitored by cross comparison. The pilot is informed after a second rate gyro failure. The servos are centered after a third gyro failure.

Digital Computers - The three digital computers perform identically:

- o Control Law Computations
- o Signal Consolidation, Monitoring and Redundancy Management for all sensors
- o Monitoring and redundancy Management for the servos
- o Monitoring and Built in test (BIT) for the digital computers themselves.

The computers are synchronized to a degree necessary for data exchange of servo commands etc. between the computers. The computer servo commands are distributed to the two EFCS-servos via a switching network L_1, L_2, L_3 as shown in Figure 2.

In a normally fault free system computer 1 is commanding servo 1 with computer 2 and 3 and EFCS servo 2 as active stand bys. The three computers are monitored by internal cross comparison for a first failure. After the first failure the cross compare is deleted and replace by each of the fault free computer's autonomous executed built in test, BIT.



Digital Computer Failures — Failure Flags

	1st Failure	2nd Failure	Red by
Computer 1	$C_{21} \cdot C_{31}$	C_1	L_1
Computer 2	$C_{12} \cdot C_{32}$	C_2	L_2
Computer 3	$C_{13} \cdot C_{23}$	C_3	L_3

Failure detected by: Compare, Bit.

L_1, L_2, L_3 changes states of the switches as indicated by the arrows in the above figures. L_1, L_2, L_3 also indicates a first failure.

Servo Failures

	Failure Flags	Red by
Servo 1	$S_{11} \cdot S_{21} + S_{21} \cdot S_{31} + S_{31} \cdot S_{11}$	L_4 L_5
Servo 2	$S_{12} \cdot S_{22} + S_{22} \cdot S_{32} + S_{32} \cdot S_{12}$	L_5

L_4 L_5 manipulates the two flowswitches.
L_4 closes flow 1 and opens a flow bypass.
L_5 closes flow 2 and initiates servo centering.

Figure 2. EFCS - Pitch Channel

The first failed digital computer is switched off by the L_1 - L_3 logic and the switching network according to Figure 2 and the logic statements tabled at the bottom of the figure 2. The second failed computer will also be switched off according to the logic statements. This assumes, however, that the failure is detected by the computer BIT. Because BIT fault detection coverage is less than 100 percent, a second computer failure not detected may cause a catastrophe. If the failure is detected the servos are centered.

Servos - The dual servos have full rate and range authority. The EFCS servo is an element within the integrated servo loop, which includes the EFCS servo and the booster and is digitally closed through the computers. Servo 1 is normally operational and servo 2 is active stand by. In a fault free system servo 1 is controlled by computer 1 and servo 2 by computer 2. The operational servo is monitored within the digital computers. Each computer may include a model of the EFCS servo. The EFCS servo position, which is equal to tandem valve position, is measured and the signal sent to the digital computer where it is compared to the output of the model. If the difference exceeds a given threshold a failure flag is set to be read by the L_4, L_5 logic, which controls the two flow switches shown in figure 2. A failed servo 1 is replaced by servo 2. Servo 2 is, as mentioned previously, only monitored when active. The booster servo is centered after a servo 2 failure. All the redundant EFCS elements are automatically tested on ground before each flight. Particular attention is made to check that all monitors, logic circuits and switches engaged in the redundancy management are fault free.

In summary, the EFCS pitch SAS/CAS redundant channels are shown in figure 2 and described above. The failure logic and redundancy management is defined on the bottom of figure 2. Monitor principles, monitor coverage and how the monitoring functions are mechanized are listed in Table 1. Table 1 also shows the consequences of 1st, 2nd and 3rd failures.

TABLE 1. EXTENT AND PRINCIPLES OF THE COMPUTER-BASED IN-FLIGHT MONITORING

	SENSORS			COMPUTERS	SERVOS
	DUPLEX	TRIPLEX	QUADRUPLEX		
MONITOR PRINCIPLES	Compare, Reasonable.	Crosscomp.	Crosscomp. Active STD By	Internal Crosscomp. BIT	Model and Compare
MONITOR COVERAGE	Comp. 100% Reas. 80%	100%	100%	Crosscomp. 100% BIT 99%	100%
MECHANIZATION	← Software →			(BIT Requires Minor HW)	→
CONSEQUENCES					
1st Failure	Level 2[1] or 3	Inform.	——	——	Inform.
2nd Failure	Level 3	Castastrophe	Inform	Inform[2] Catastrophe	Catastrophe
3rd Failure	——	——	Catastrophe	Catastrophe	

1) If Detected by Reasonableness Test
2) If Detected by BIT

It should be noted that almost all monitoring functions are realized by software. The degree of system redundancy is related to the failure probabilities of the system elements.

The following sections describe the individual techniques used to meet the assumed requirements for this illustration.

2.2. Sensor Monitors

2.2.1 Dual Sensors

The EFCS pitch SAS/CAS channel includes four air pressure sensors, two measuring total pressure P_T and two measuring static pressure P_S. The pressures are transformed to electrical signals which are used in the redundant digital computers for gain scheduling and logic statements. According to Table 1 the pressure signals are not flight safety critical assuming that a failure is detected and the gain in the control loop is set to a fixed predetermined value. Dual signals can be monitored during flight by:

- o Self test and
- o Comparison

The digital computer is well suited to perform both types of monitoring. Almost any kind of stimuli signals can be provided for sensor tests and appropriate algorithms can be used for evaluation of the response of the tested element. The problem is mostly the tested element itself, which shall be active in the control loop all the time and at the same time be tested with additional stimuli inputs.

Self Test

Two solutions to the self test problem are:

- o Reasonableness test of the regular output
- o Limited sensor hardware monitoring

Reasonableness Tests - May include rate and range checks as well as evaluation of the frequency content of the output signal. The reasonableness test of the pressure sensors is recommended to include rate and range checks.

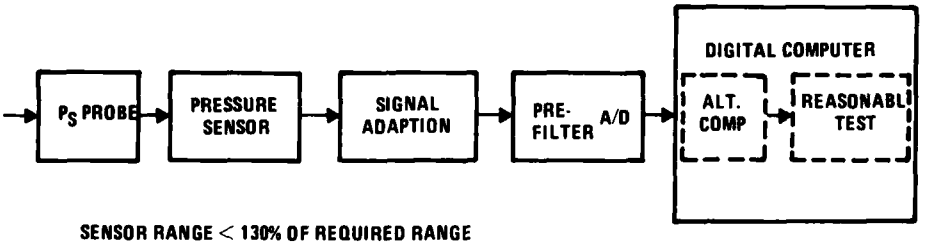


Figure 3. Reasonableness Test for the Altitude Signal

The main message of figure 3 is that the installed system or the signal path from the P probe to the reasonableness test algorithm in the computer shall be oversized. The measurable and transferable range shown by the elements shall be at least 130 percent of the operational altitude of the aircraft. Similarily the measurable and transferable altitude rate (range change/time) should be at least 200% of the real operational aircraft rate. The sample rate must also be properly chosen. The reasonableness test in the digital computer shall indicate a failure whenever the computed:

altitude is more than 125% of operational altitude
altitude rate is more than 150% of operational rate.

The altitude test will detect a majority of the hardover failures in the pressure sensors and signal path elements.

The altitude rate test will detect abrupt changes in the altitude information indicating both a hardover failure to full scale or zero output. The number of instructions required to do the reasonableness test are about ten and the computational rate can be as low as 5 - 10 times per second. The added computational load of the reasonableness test is thus very low. The coverage of this reasonableness test is about 70%.

A similar reasonableness test can be applied to any sensor and signal path. It leads to a sensor system with somewhat larger dynamic ranges which somewhat reduces the accuracy of the signals. The benefit of failure isolation to one of two dual sensors may be worth the price.

Limited Sensor Hardware Monitoring - includes dedicated tests such as sensor voltage level tests, check of the error signals in internal sensor servo loops, added internally

compensated bias signals, etc. The pressure sensor hardware test consists of voltage checks within the pressure sensors.

The pressure sensor includes a pressure sensing element in a bridge circuit, amplifiers, and temperature compensation electronics. The voltage level at critical points are picked off and added together. The resulting signal is provided to the digital computers. If the voltage level exceeds specified limits a failure indication is given. The added digital computer load is negligible. This test increases the pressure sensor reasonableness test coverage from 70 to about 80%.

Sensor hardware BIT of the above mentioned type is most important in more complex sensor elements such as analog air data computers and gyro platforms. The test methods are now changing following the trend of transfer from analog to digital mechanization.

Comparison Monitoring of dual flight safety critical sensors is a frequently used method, which usually detects all significant failures in the signal paths. The comparison monitor is included in the reference FBW-pitch channel for monitoring of the dual pressure sensors. The probability of detecting a failure is very close to 100 percent. Comparison of pressure sensor 1 and 2 is performed within the digital computers, identical in all three. When a failure causing a miscompare occurs the pressure information is disregarded and not used thereafter. The principles of the total pressure comparison monitor is shown in figure 4.

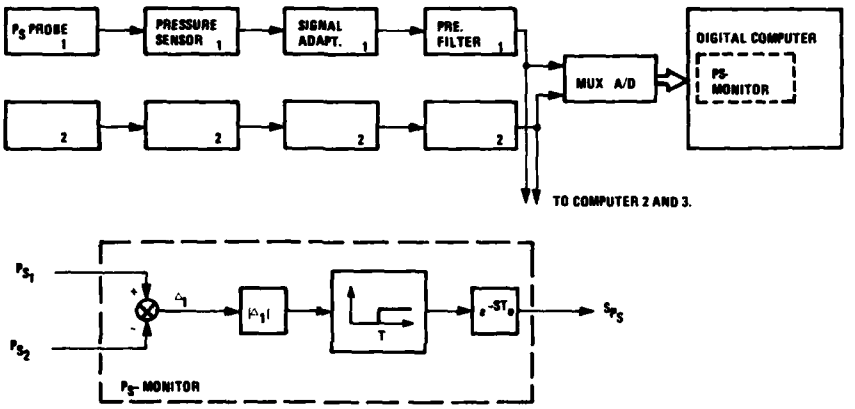


Figure 4. P_S Comparison Monitor

Figure 4 shows the principle of the P_S comparison monitor. It should be noted that the P_{S1} and P_{S2} signals are split off to digital computer 2 and 3 before they go into the single channel MUX. The P_S-monitor is mechanized in software. The main problem associated with designing the monitor is to determine the threshold T, and the time delay To. The always existing conflict is how to design a monitor that detects all failures within minimum time and without failure indications for extreme but accepted component tolerances. This problem will be discussed assuming redundant signal paths. Each path consisting of n element, the gain of each element R_n and the tolerance E_n as shown in Figure 5.

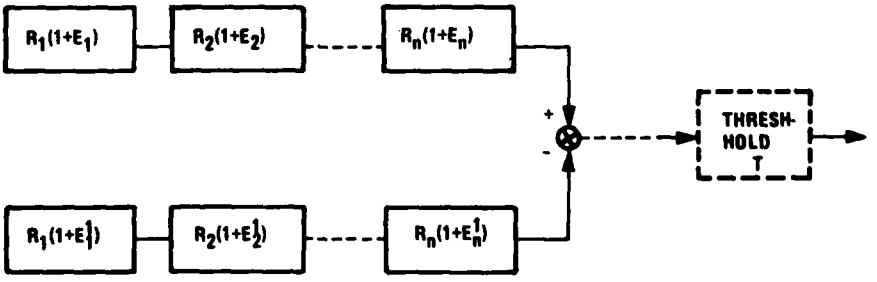


Figure 5. Tolerances of Cascaded Elements

Tolerances of cascaded elements can be written as:

$$R_1(1\pm E_1).R_2(1\pm E_2)\ldots R_n(1\pm E_n) =$$

$$R_1.R_2\ldots R_n(1\pm E_1\pm E_2\pm\ldots E_n\pm E_1E_2\pm E_2E_3\pm\ldots)$$

The tolerance E is about 1% in this application. Double and higher order products can be disregarded. Left is:

$$R_1.R_2.R_3...R_n(1\pm E_1\pm E_2\pm...\pm E_n)$$

Two principles can (at least) be used when adding the tolerances.

1. "Arithmetic addition, worst case, same sign".

$$E_A = E_1 + E_2 + ... + E_n$$

2. The RSS-method

$$E_R = (E_1^2 + E_2^2 + ... + E_n^2)^{\frac{1}{2}}$$

It can be shown that E_R goes towards the standard deviation (1σ-value) of the tolerance population assuming that E_1 to E_n are properly defined for the actual tolerance distribution.

According to Figure 5, the two methods regarding suitable setting of monitor threshold T considering given hardware-tolerances $E_1...E_n$ gives:

$$E_A = E_1 + E_2 + ...+ E_n + E_1^1 + E_2^1 + ... + E_n^1$$

$$E_R = (E_1^2 + E_2^2 + ... + E_n^2 + E_1^{1^2} + E_2^{1^2} + ... + E_n^{1^2})^{\frac{1}{2}}$$

Extreme tolerances are not allowed to cause a false failure indication. The probability of false failure indications must be kept at a minimum. Therefore it is important to define proper threshold margins.

Estimated hardware failure probability of one of the two pressure channels is in the order of $5x10^{-5}$ per flt hr. It is in reasonable to assume that the probability of false failure indication must be a factor to ten less or in the order of $5x10^{-6}$ false failure indications per flt hr.

For normal distributed tolerances a probability for false failure indication less than $5x10^{-6}$ corresponds to a threshold setting of $4.4\sigma = T$. To avoid false failure indications T must be

$$T_A > E_A$$

$$T_R > E_R$$

Each of the pressure signal paths contains 4 cascaded elements each element having a 1σ tolerance of 1%. The two methods then give the following threshold T value.

$$T_A > E_A = 2 \times 4 \times 1 \times 4.4 = 35\%$$

$$T_R > 4.4\ E_R = 4.4\ (1^2 \times 4 + 1^2 \times 4)^{\frac{1}{2}} = 13\%$$

Conclusions from this are:

1. The RSS-method is the most realistic method to determine the threshold T when the hardware tolerances are normal distributed. This is however not always the case especially when the system elements included parts that were screened more carefully.

2. The pressure sensor monitor threshold shall be set at 13 percent of full scale to avoid false failure indications.

The Time Delay To - the signals from the pressure sensors contain air pressure noise and voltage transients which in spite of the digital computer prefilter may propagate into the computer and the comparison monitor and be interpreted as failures. The time delay To must be long enough to prohibit temporary failure indications from reaching the outside world. On the other hand To must be short enough to not block a real failure signal.

In the digital computer To is realized by requesting several consecutive failure indications before the failure is announced. The time between the failure indications is dependent on how frequent the monitor functions is calculated. The pressure monitor calculations shall in this example be performed 5 times per second, which offers a To of n x 200 m sec. It is reasonable to choose n = 3 and To = 600 m sec.

To can be chosen relatively long for the pressure sensor comparison monitor since the pressure information only controls the SAS/CAS loop gain and in itself cannot command a servo hardover. The computer is programmed to inhibit any new gain schedule outputs after a failure is detected. The feature to inhibit a faulty signal or output shall generally be used in software monitors.

The comparison monitor software includes about 30 instructions and the computational rate is 5 times per second. The computer load is thus very low.

How to determine the comparison monitor threshold and time constant is often a delicate matter, especially for rate gyros and accelerometers. These sensors generally are given high command authority which means that a hardover failure in the sensor may cause full servo hardover within .5 second. With an aircraft response of 1 G per degree control surface and a pitch short period time constant of .5 sec the aircraft load factor will reach 10 G within about 1 second, as shown in Figure 6.

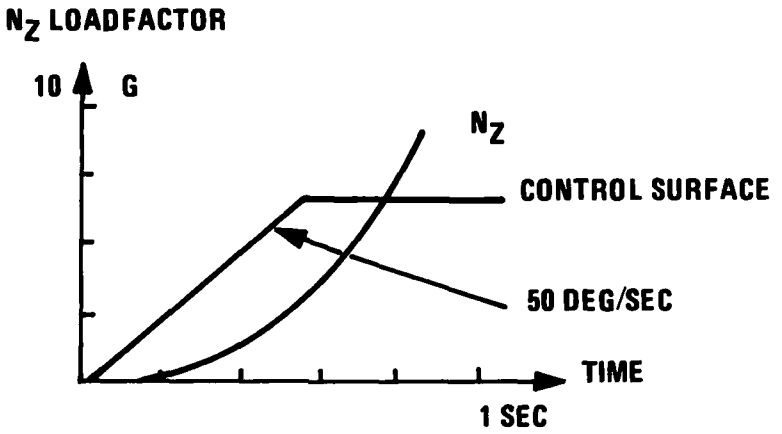


Figure 6. Load Factor Following A Hardover Failure

It is obvious from Figure 6 that the type of sensor hardover failure must be detected and disconnected within less than .1 second in order to limit the aircraft load factor to reasonable 1-2 G. This usually means a conflict between the requirement on failure transients and the risk of false failure detection when determining T and To. The conflict can in many cases be resolved by some additional program instructions used to make T or To or both functions of altitude and/or speed. The SAS/CAS loop signal level is usually lower at high speed which makes it possible to reduce the threshold T and get a tighter monitor.

2.2.2. Triplex Sensors

The EFCS pitch channel includes triplex stick position sensors. The stick displacement is a function of the pilot pitch command force. The sensors are of the differential transformer type. They must be treated as high authority sensors since they may cause full control surface deflecif they fail.

The three sensors are monitored by identical monitor software in all three computers. The sensors are checked by cross channel comparison. A midvalue algorithm selects the position value to be used in the control law calculations. No reasonableness test is assumed to be needed because the position sensors are reliable and felt to contribute fairly little to the total risk for catastrophe due to EFCS failures.

Cross Comparison

Cross comparison of three redundant sensor signals herein called sensor A, B and C involves the following setps:

- Compute the three cross differences A, B, C
- Check the differences against the threshold T
- Delay a failure indication the time To
- Interpret the combination of failure indications
- Announce failure
- Reset the monitor after temporary (false) failure

The cross channel monitor shall be able to:

- o Identify first failed channel
- o Detect a second failure.

The cross channel monitor can typically identify a first failure with little difficulty. Detection of a second failure is in some cases a problem however.

One mechanization example of a cross channel monitor is shown in Figure 7a . The block diagram and logic symbols are used to explain the monitor function.

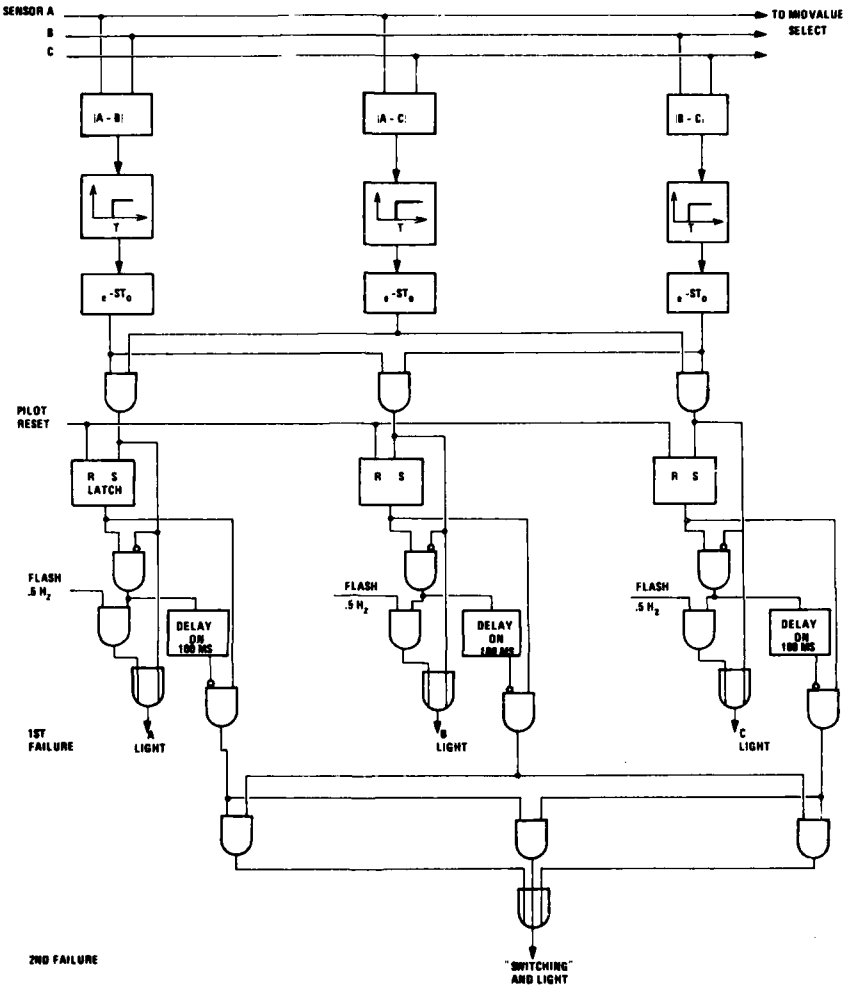
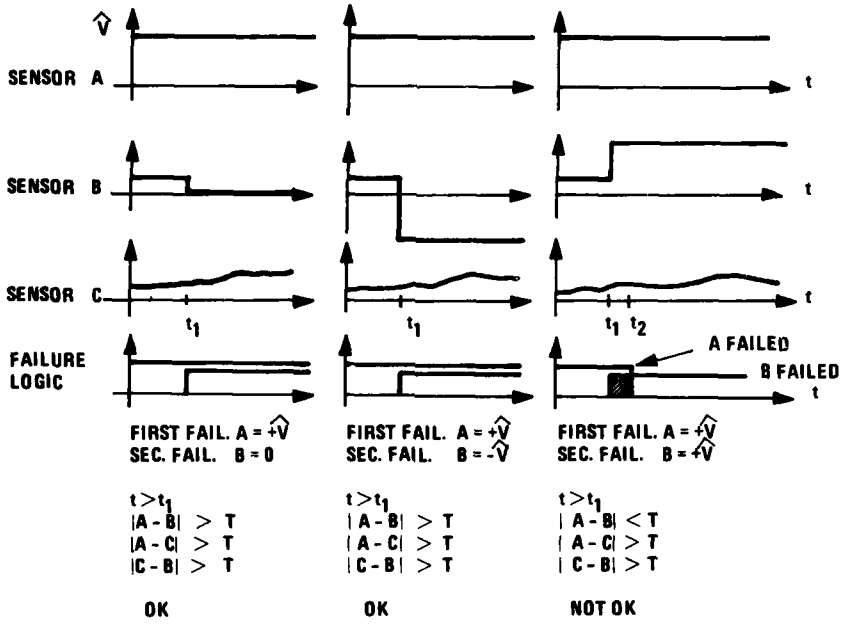


Figure 7a.Cross Channel Monitor

Computation of the cross channel differences, the check against the threshold T and the time delay To is realized with similar software as used in the dual channel comparison monitors. The three and - gates located downstream are provided such inputs that their high output identifies a first failed channel A, B or C. The logic subsequently triggers a warning light and sets a latch. If the first failure indication was false the and-gate returns to low state. The latch is still set and the logic statement for flashing warning lights (.5 Hz) is fulfilled. The flashing light is an information to the pilot to reset the sensor warning light.

Extra logic is needed to detect a second failure of similar sign and size as the first failure. The problem is illustrated in Figure 7b Of the three second failure possibilities shown in Figure 7b, the last one will not be indicated as a second failure since both sensor A and B have similar failures the logic will interpret C as the faulty sensor and A and B as fault free. This problem can be resolved in various ways. In this example a delay of 100 milliseconds is applied on the first failure logic signal before it is allowed to change back to low state. The second failure is recognized by the monitor during this delay. Figure 7a shows how the delay is located sharing part of the "flash logic".

When the cross channel monitor is mechanized in software the configuration of Figure 7a transformed to a flow structure. Several modifications and improvements are possible, especially for second failure detection. A cross channel monitor subroutine in the digital computer will require less than 100 instructions memory space.



PROBLEM: IN THE LAST COLUMN THE SECOND FAIL. SITUATION IS SIMILAR TO THE FIRST FAIL. SITUATION

	$t<t_1$
THE LOGIC INDICATES	$\|A-B\|>T$
$t<t_1$ ONLY A FAILED	$\|A-C\|>T$
$t>t_1$ ONLY C FAILED	$\|C-B\|<T$

SOLUTION: DELAY THE FIRST FAILURE (A) ~100 MSEC ($t=t_2-t_1$) BEFORE IT IS ALLOWED TO AUTOMATICALLY RESET. THIS ALLOWS THE DOWNSTREAM SWITCHING LOGIC TO REACT.

Figure 7b.Cross Channel Monitor - The Second Failure Problem

The stick position sensors are flight safety critical and the cross channel monitor calculations must be done 40 times per second. One cross channel monitor will thus request 1-2 percent of the computer throughput.

2.2.3. Quadruplex Sensors

The pitch rate gyros are quadruplex in the discussed EFCS pitch channel. The reasons for the selected redundancy are:

- o The rate gyros are high authority sensors
- o Pitch rate feedback is required for safe slight
- o The rate gyros have fairly high failure probability
- o The coverage of gyro self test is far below 100 percent

It may be possible to replace one or two of the rate gyros with sensors shared with the inertial reference system or by analytical redundancy. The monitoring principles may remain the same anyway.

The four rate gyro signals are available for all three digital computers as shown in Figure 2. The monitor software is identical for the computers. The monitor principle is to use:

- o Triplex cross channel monitoring,
- o Fourth sensor active standby engaged after first triplex sensor failure.

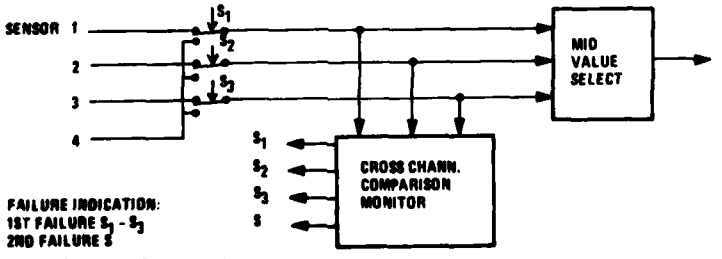
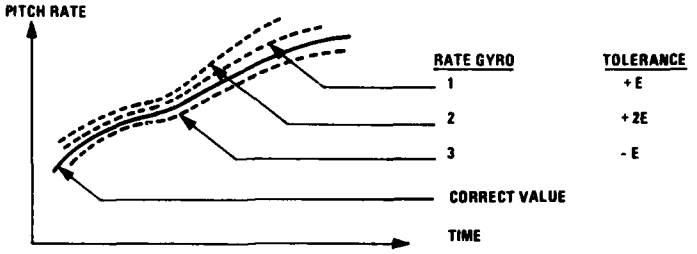


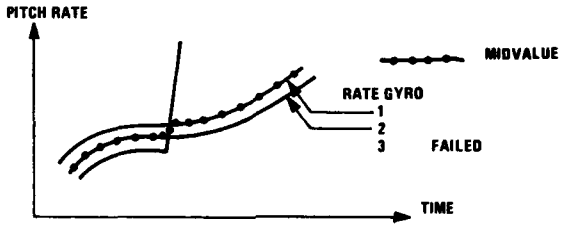
Figure 8. Monitoring Of Quadruplex Signals

Figure 8 shows the principle in block diagram form. The cross channel comparison monitor is described in figure 7 and above. The midvalue select algorithm has valuable qualities which makes it desirable to also use it in the quadruplex monitor.

The MVS algorithm selects the middle of three signals. Since tolerances can have both positive and negative sign and are normally distributed it is likely that one of three signals has a tolerance sign different from the two others. Therefore the MVS may frequently select the best signal and it will never pick the worse signal as the midvalue, which is illustrated in Figure 9A. The MVS quality to never select the worse signal out of three is extremely valuable when one sensor fails. The failed sensor will be the extreme or worse one and therefore immediately detected and isolated as shown in Figure 9B.



a. Mid Value Select Principle



b. MVS Failure Isolation

Figure 9. Mid Value Select

The MVS algorithm can be extracted from Table 2.

TABLE 2. MID VALUE SELECT TRUTH TABLE

LARGEST	A	A	C	C	B	B
MIDDLE	B	C	A	B	C	A
DIFFERENCES	C	B	B	A	A	C
1. |A| - |B|	+ / + / -	+ / - / -	+ / - / +	- / - / +	- / + / +	- / + / -
2. |B| - |C|						
3. |C| - |A|						

The truth table shows that:

If differences 1 and 2 have same sign then B is midvalue

If differences 1 and 3 have same sign then A is midvalue

If differences 2 and 3 have same sign then C is midvalue

The MVS algorithm defined above requires about 20 program instructions and can be stored as a subroutine. The computational rate for the pitch rate gyro MVS algorithm shall be 40 times per second, which is fast enough not to degrade the pitch innerloop stability.

2.3. SERVO MONITORS

The EFCS pitch channel shown in figure 2 contains 1 dual integrated hydraulic servo for control of the elevator control surface. The integrated servo has full rate and range authority. Loss of servo control will result in a catastrophe because no artificial pitch stabilization is available. The servo must be carefully monitored and a first servo failure quickly detected and isolated.

Two different servo monitor mechanizations will be discussed in the following subsections:

- o External software servo model
- o Internal software servo model

Both alternates strive for a minimum of servo hardware through maximum utilization of digital computer software.

2.3.1. Servo Monitor With External Software Model

A dual integrated EFCS servo with external software model is shown in Figure 10.

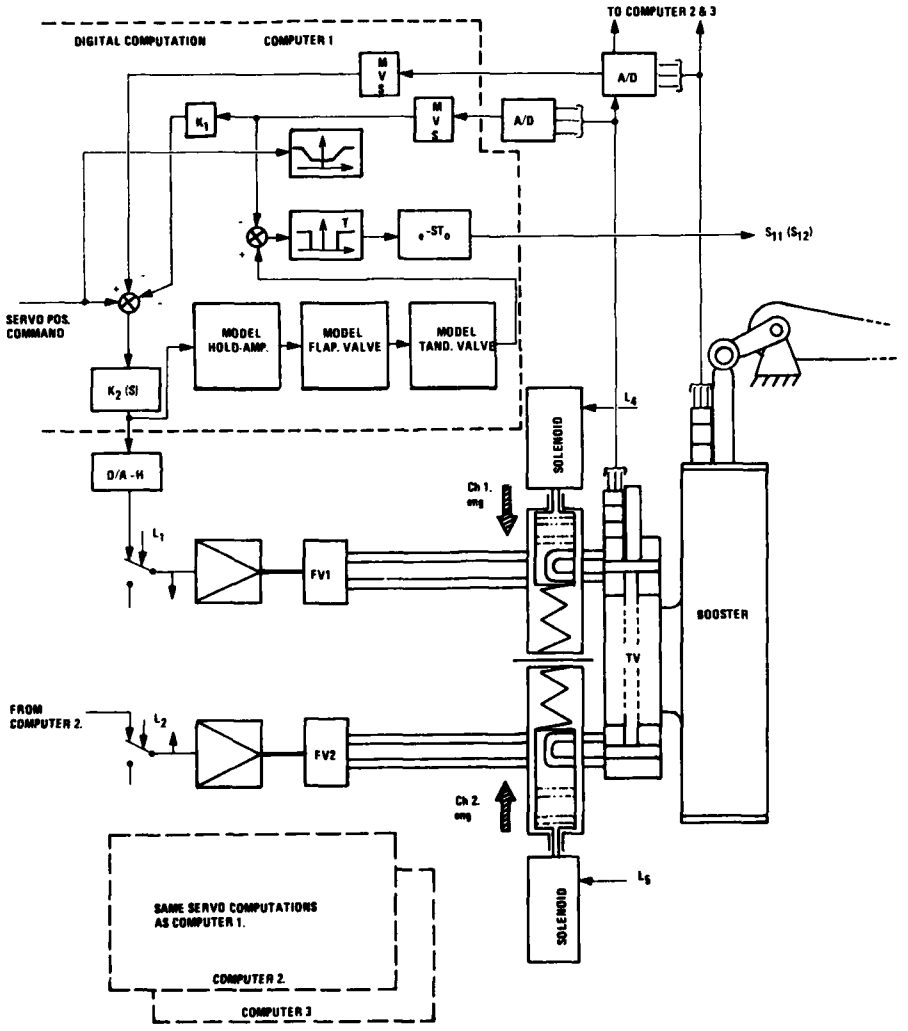


Figure 10. Dual EFCS Servo With External Software Model

The redundant servo amplifiers, flapper valves (FV1 and FV2), the two hydraulic flow switches and the tandem valve (TV) and booster actuator are mechanized in hardware. Other hardware includes Tandem valve and booster actuator triplex position sensors as well as the A/D and D/A convertors. The sensor signal consolidation using midvalue select, and the filtering and the loop closure are done digitally. Also the servo models and the servo monitor are realized in software. All three digital computers perform identical servo computations based on the same midvalues. The digital servo computations have to be performed at least 80 times per second, otherwise the servo performance will be degraded.

As can be seen from Figure 10 only one of the dual EFCS-servos controlling the tandem valve can be active at the same time. In this case, for a fault free system, only flapper valve FV1 is engaged. This also means that only FV1 is monitored.

When a first failure has occurred and the second flapper valve FV2 has been engaged and is controlling the tandem valve then the monitor automatically starts to monitor FV2. A first servo failure indication S_{11} is interpreted differently in the failure logic L_4 and L_5 than a second failure indication S_{12}.

The external software model monitor is a digital mechanization of the traditional external analog servo model. The servo loop feedbacks are control surface rate (which is proportional to tandem valve position) and control surface or booster actuator position. Only tandem position is used in the servo monitor. The booster actuator has a very low failure probability and is treated here as safe.

The difference between the servo position command signal and a proper mix of tandem valve and booster position feedback is computed, D/A converted and fed to FV1 windings. The same signal is the input to the software servo model.

The model of the hold circuit and the amplifier is a simple transport delay corresponding to the characteristics of the hold circuitry and the output frequency. The time delay is thus about 12 m sec for an output rate of 80 times per second.

The flapper valve model is a first order lag characterized by a gain and time constant. The flapper valve has high bandwidth and the time constant is 1-2 m sec, which considering the sampling rate may be negligible. The tandem valve is modelled by a fixed gain and an integration.

The servo model is therefore summarized by:

Model	Transfer Function
Hold/Ampl	$K_1 e^{-sT}$
Flapper valve	$K_2 \frac{1}{1+\tau s}$
Tandem valve	$K_3 \frac{1}{s}$

The difference between the servo model output and the actual tandem valve position is calculated and checked against the threshold T. If the threshold is exceded more than 5 consecutive computational cycles (T = 50 m sec) a failure is announced, S_{11} (S_{12}) high.

The Figure 10 mechanization includes a minimum of hardware and consequently also less hardware tolerances, which is very helpful when determining the threshold T. The conflict between severe failure transients and the risk for false failure indications is very pronounced for a full authority servo. This problem is for the here discussed EFCS servo resolved by:

- o Minimum Hardware
- o Variable Threshold

The variable threshold mechanization is shown in Figure 10. The threshold is a function of the servo command with a minimum threshold for small commands. The rationale for the threshold schedule is the fact that high control surface response (G's per degree control surface) only requires small control surface deflections and small tandem values signals compared to low speed or landing conditions where the product of large signals and tolerances requires large thresholds to avoid false failure indications.

Digital servo loop closure and servo monitoring with external software model requires about 100 instructions all together. The computational rate has to be high and in the order of 80 to 100. The requested throughput is in the order of 10 KOPS or roughly 5 percent of what is available in currently marketed flight computers. The required throughput may be reduced if the time to detect a failure allows the monitor computations to be reduced to 40 times per second.

2.3.2. Servo Monitor With Internal Software Model

Figure 11 shows an integrated EFCS servo with internal model. Only channel 1 is shown. In a fault free system channel 1 is controlled by computer 1 and channel 2 by computer 2. The servo monitor in all three computers is monitoring the active servo channel 1, because they all use the same servo command and servo feedbacks. The hardware mechanization is identical to the servo configuration described in Section 2.3.1. The servo loop closure, filtering and monitoring are mechanized in software. All three digital computers perform identical computations. The failure logic and redundancy management is similar to what has been described for the dual EFCS servo with external model.

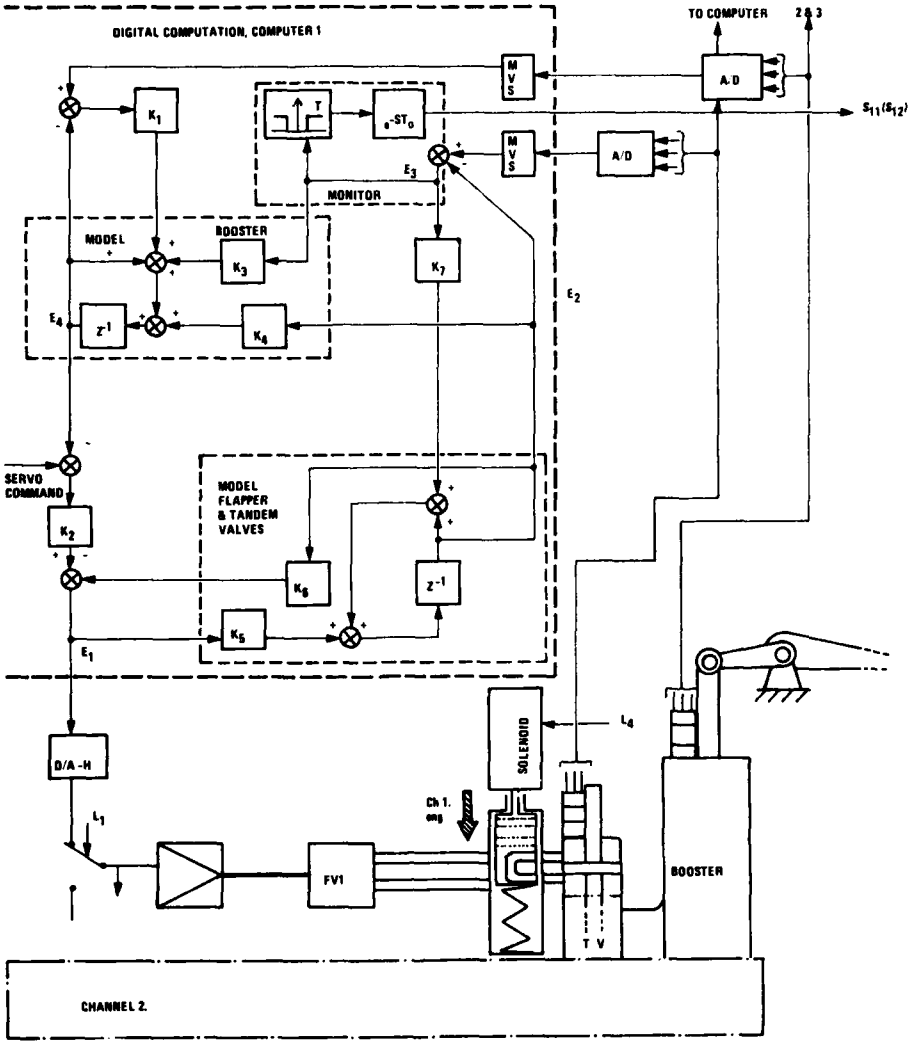


Figure 11. Channel One of Dual EFCS Servo With Internal Model

The EFCS servo with internal software model is characterized by the fact that the two approximate models are active in the closed servo loop. This technique uses a Kalman filter for estimating the servo status and providing the servo measurement error signals. It is of great value in this servo application because of the following reasons.

- o Efficient software programs for synthesis of servo control laws are available. The programs account for computational delays and sample and hold effects.
- o The dynamic performance of the servo loop can be greatly manipulated and easily modified.
- o The difference signal to be monitored is available as part ot the control law.
- o The internal servo model is automatically adjusted to reduce the effect of changing environment and tolerances.

The block diagram in Figure 11 contains three dashed boxes which can be interpreted as the flapper and tandem valve models, the booster model and the servo monitor.

A servo command multiplied with the fixed gain K_2 causes an error signal which commands the tandem valve to a certain position. At the same time E_1 drives the flapper and tandem valves model and an estimated valve position E_2 is predicted. If the tandem valve position and the prediction differ the error signal is fed back and allowed to correct the valve model in order to minimize the error.

The predicted tandem valve E_2 position is also multiplied with the Gain K_6 and fed back to the servo command error signal E_1. The signal's third purpose is to serve as input to the booster model. The booster model response is compared to the booster position. The difference is fed to the booster model with a proper sign to reduce the difference. The predicted booster response is fed back and summed with servo command.

It can thus be noted that the servo loop feedbacks are the predicted tandem valve and booster positions which correspond to filtered position sensor signals. The cross feed

of the servo valve position error E_3 with the gain K_3 to the booster model corresponds to conventional shaping filtering and is done to improve the dynamics of the integrated servo.

The servo monitor is conventional. The difference between predicted (model) and real tandem valve position E_3 is already available as part of the servo control law. E_3 is compared to the threshold T and a failure indication is delayed the time To before it is announced. The key issue when designing the servo models, control laws, and monitor is to find a good compromise between the desire to adapt the flapper and tandem valves model for zero difference between the predicted and measured valve positions E_3 and the risk of washing out softover servo failures. If the gain K_7 is increased the difference E_3 will be reduced. Reduced E_3 may mean that a servo softover (slowly growing) failure will be washed out and not detected.

The problem can usually be resolved since the K_7 gain can be selected low enough to make it possible to detect all servo failures causing an increasing load factor of less than .1 G per second and still keep the risk for false failure indications at an acceptable level by properly setting the threshold and time delay.

The digital implementation of the described modified Kalman filter requires about 100 instructions including the MVS signal consolidation and the monitor. The computational rate should be kept at 80 times per second. The filter design features may make it possible to go even lower, however.

2.4. Digital Computer Monitors

2.4.1. Background

Past analog systems have relied primarily on the use of redundant hardware channels and comparison monitors to meet system safety requirements. A fail operative/fail-safe system would consist of three identical channels of hardware with comparison monitors to isolate and disengage a faulty channel. These system configurations typically used dedicated built-in-test (BIT) circuits to perform automatic preflight tests. These preflight tests insured the proper operation of the monitors and critical system functions. As much as twenty to thirty percent of the total system was comprised of the monitor and test function circuits.

Modern digital systems using a general purpose computer have two inherent characteristics that can be exploited to vastly improve the testability and reliability of present systems. These characteristics are time quantizations and the stored program concept.

The stored program concept simply means that the computing system is directed by and under the control of a program (software). The system can, therefore, be programmed to perform tests as well as its primary task of controlling the aircraft. Time quantization means that system computations are not simultaneous, and a significant amount of hardware is time shared. As a result of these characteristics, the system can be performing test functions time shared with its primary control functions. Also, hardware circuits need only be tested once, regardless of the number of functional uses that they have.

System reliability can be improved by using self test. Self test can isolate failures to the bad channel when less than three channels are operative and comparison voting is impossible.

The effect of testing on a three-channel redundant system can be demonstrated by considering the following reliability expression:

$$P_{SF} = P^3 + 3P^2RD_T + 3PR^2D_V$$

Where

- P_{SF} = Probability of system failure
- P = Probability of failure of a single channel
- R = Reliability of a single channel = (1 - P)
- D_T = Self test deficiency
- D_V = Comparison voter deficiency

D_V approaches zero for the first failure because voting is possible. This reduces the equation to:

$$P_{SF} = P^3 + 3P^2(1-P)D_T$$

For a 1 hour mission and a probability of single channel failure $P = 200 \times 10^{-6}$ per operation hour the probability of total system failure becomes:

P_{SF} = 1.2×10^{-7} for D_T = 1 , 0% BIT coverage

P_{SF} = 1.2×10^{-9} for D_T = 0.01, 99% BIT coverage

BIT with 99 percent coverage thus reduces the risk for loss of EFCS computation due to digital computer failures by a factor of 100 during a one hour mission.

The above calculation of mission reliability assumes that the system was fault- free at the start of the mission. It is reasonable to assume that comparators will detect failures (D_V approaches zero) as long as the comparators are operational, and inputs are large enough to cause comparator trips. The comparators, therefore, must be exercised in preflight test to insure that they are capable of properly reacting to a miscompare. In general, all EFCS monitoring and fault reaction switching must be exercised in preflight to preclude latent failures. This is not a trivial task because a considerable amount of the system is involved in monitoring. This includes; processor, memory, analog inputs/outputs, discrete inputs/outputs, disengage circuits, etc. Preflight test coverage for critical monitoring and fault reaction switching functions must approach 100 percent to ensure flight safety over the life of the system.

2.4.2. Comparison Voting

As described in section 2.1. and Figure 2 a first computer failure is detected by the software voters identically mechanized in the three digital computers.

The voters include cross channel comparison similar to the triplex sensor monitor described in section 2.2.2. and Figure 7. The inputs to the voter are the three servo commands, which are transmitted on the busses and available in all three computers. In the computers the cross channel differences are computed and the differences checked against the threshold. A failure indication is delayed a suitable time (50- 100 m seconds) before the computer failure is announced.

Figure 12 shows the three digital computers, the exchange of servo commands, and the software voter monitor.

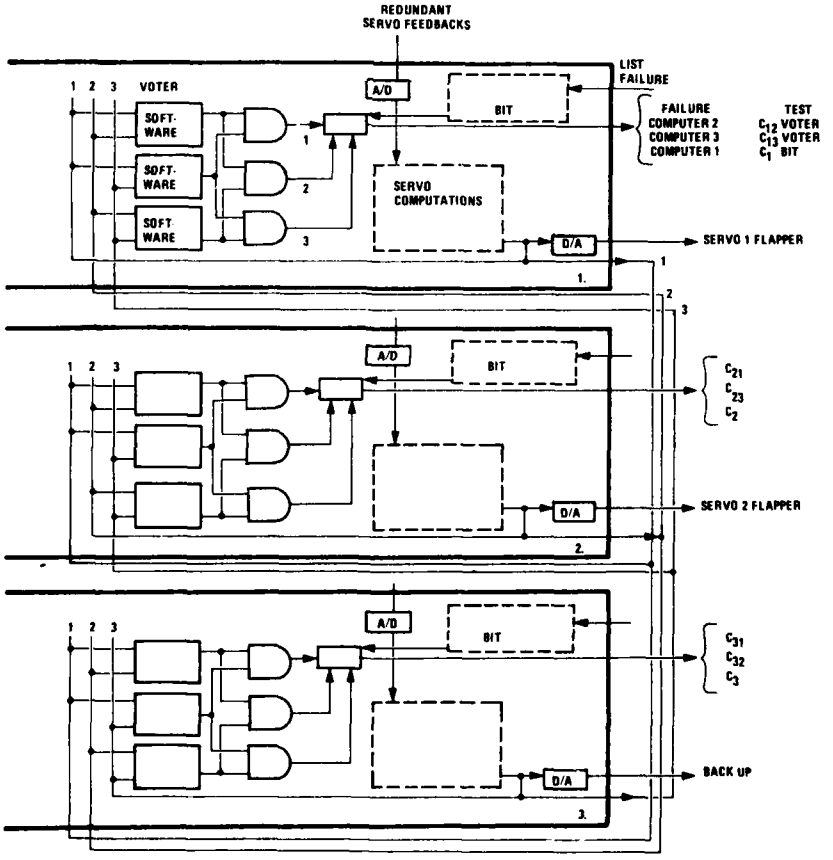


Figure 12. Computer Monitoring - Voting and BIT

The failure logic L_1, L_2, L_3 defined in Figure 2 requests failure indications from two computers before any computer disengagement takes place. The reasons for this are two fold:

- o One faulty computer shall not be able to disengage a fault free computer
- o The probability of two or three failures occurring at the same time is negligible.

The first failed digital computer is disengaged and the servo commands are provided by one of the two fault free computers according to the computer redundancy management scheme shown in Figure 2 and in Table 3.

The three computers are provided signals from the L_1, L_2, L_3 logic when a first failure has occurred. The first failure signal is used to disable the cross channel voting program and engage the inflight BIT.

TABLE 3. DIGITAL COMPUTER REDUNDANCY MANAGEMENT

DIGITAL COMPUTER FAILURE		DIGITAL COMPUTER CONTROLLING	
1ST FAILURE	2ND FAILURE	SERVO 1	SERVO 2
Computer 1	—	Computer 2	Computer 2
2	—	1	3
3	—	1	2
1	Computer 2	3	3
1	3	2	2
2	1	3	3
2	3	1	1
3	1	2	2
3	2	1	1

1ST FAILURE DETECTED BY CROSSCHANNEL VOTING

2ND FAILURE DETECTED BY IN-FLIGHT BIT

It should be noticed that the degree of computer syncronization will affect the threshold level, T.

- o If the computers are fully synchronized and uses identical inputs the threshold, T, can be set at the least significant BIT level.
- o If the computers are asynchronous the threshold must be high enough to allow for channel differences due to inputs and outputs at different times.

2.4.3. In Flight BIT

The in flight BIT engages after a first failure. Its function is to detect and isolate a second failed digital computer. How well this is accomplished depends on the BIT self test program efficiency, the BIT coverage. The inflight BIT is developed as an interactive process where the digital computer failure modes are identified and the test methods required to detect the different failure modes established. The BIT coverage is to a great extent a function of the effort spent on the failure mode effect analysis (FMEA) and the BIT program development.

To achieve a bite coverage of better than 99 percent the following computational elements must be tested at a minimum:

- o Power Supplies
- o Central Processor (CPU)
- o Memory
- o Multiplexer
- o A/D & D/A Converters
- o S/H Amplifiers
- o Real Time Clock
- o I/O Control

The above listed digital computer elements are tested by a series of dedicated in flight test programs as shown in Figure 13. Table 4 shows how the different computer elements are tested by the various BIT monitors. All BIT monitors are mechanized in software except the watch dog timer, real time clock monitor and the dynamic computation monitor that requires additional hardware.

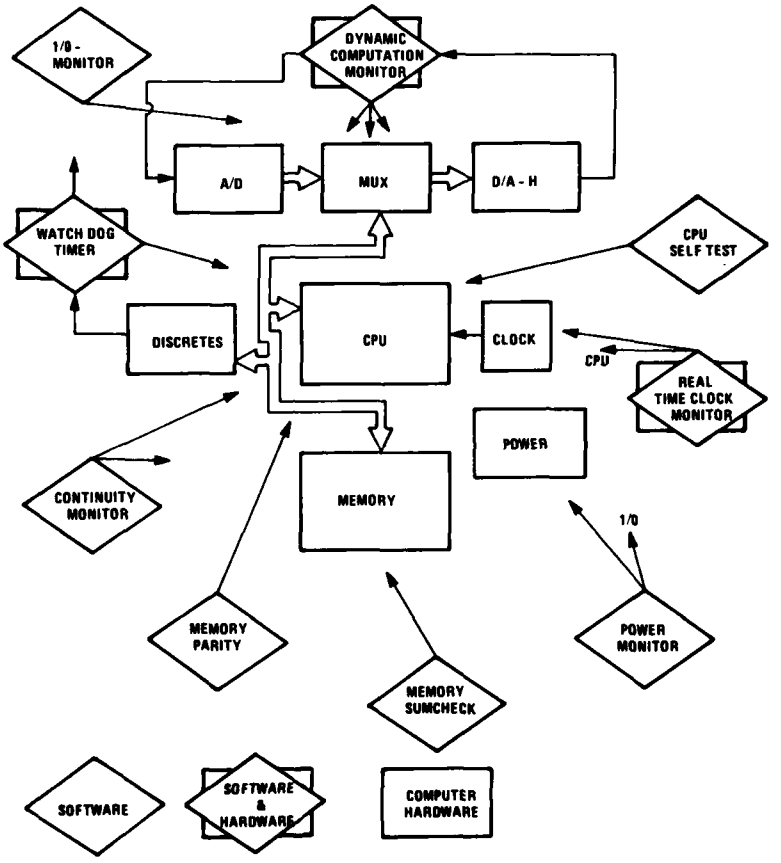


Figure 13. EFCS Digital Computer In Flight BIT

The different BIT monitors are described in the following paragraphs.

Power Supply Monitor - The CPU power supplies which are critical to EFCS operation are monitored via several comparisons against prestored nominal voltage values. Two to six power supply tests may be performed,dependent on the number of voltages required by the digital computer. The purpose of these tests is two-fold:

- o To compare the supply outputs against predetermined nominals and tolerances.
- o Since these are known-value analog inputs, a test of the analog I/O and Analog-to-Digital converter is also performed.

The monitor computations are performed 40 times a second. Each power supply output is compared against a prestored constant. If the difference exceeds the trip level a counter, dedicated to that supply monitor, is incremented. If failures are counted on two consecutive comparisons, the program indicates a power supply failure by setting the index register to a unique value and jumping to the "Fail Loop" and then subsequent system disengagement due to the time-out of the Watchdog Timer.

TABLE 4. COMPUTER IN FLIGHT BIT

COMPUTATIONAL ELEMENTS	COMPUTER MONITORS								
	Power Supply Monitor	CPU Self Test	Memory Parity	Memory Sumcheck	I/O Monitor	Continuity Monitor	Real Time Clock Monitor	Watch Dog Timer	Dynamic Computation Monitor
Power Supply	x							x	x
Central Processor (CPU)		x				x	x	x	x
Memory and Interface			x	x		x			x
Multiplexer	x								x
A/D and D/A Converter	x				x				x
Sample and Hold Amplifiers									x
Real Time Clock							x	x	x
I/O Control	x				x			x	x

CPU Self Test - Correct operation of the central processor can be verified by a self-test program that is designed to thoroughly exercise every instruction within its repertoire. The object of such a program is to test all logic elements within the processor that are critical to successful on-line operation. The effectiveness of such a test program is measured by the number of logic elements verified compared to the total number of elements in the CPU.

Functionally, the CPU test includes exercise of the following components.

- o Addressing capability between CPU and memory
- o Op code decoding
- o Information transfer between register: between memory/register
- o Logic instruction execution
- o Shift and rotate instruction execution
- o Load and store instruction execution
- o Arithmetic instruction execution
- o I/O instruction execution
- o Indexing
- o Indirect addressing
- o Branch and return
- o Conditional and unconditional jump
- o Arithmetic Logic Unit (ALU)
- o Micro program

In-flight processor self-test program of approximately 200 words of memory and 1 millisecond computation time can provide the necessary coverage.

Memory Sum Check and Parity - Instruction and data (constants) memory can be verified by memory sum checks. Memory sum checks add the contents of blocks of memory and then compare these results against predetermined constants.

For semiconductor memories, a parity concept can be employed for in-line monitoring. By including parity into the memory organization, continuous and complete failure monitoring can be obtained. This in-line monitoring technique is mechanized almost exclusively with hardware. The only software required is that to check out the parity detection circuits. This can be accomplished simply by transmitting a discrete from the central processor which forces "even" instead of "odd" parity.

Parity checks are also performed as each memory location is accessed and the combination of these two self-test techniques will detect every predictable sequence or combination of instruction or constant memory failure.

Parity error detection requires inclusion an additional (parity) BIT appended to each word of N bits. This bit is always set so that the total number of one bits is odd (even). This is called an odd (even) parity scheme and provides for single bit error

detection. If a single or odd number of errors occur, e.g., a bit changes from a 0 to a 1, then the number of one bits is changed from odd to even. Therefore, no single or odd number of failures in a word can escape detection. However, with integrated circuit (IC) memories and multiple bits of a word per IC, the probability of multiple (even numbered) bit failures is no longer insignificant. But with the advent of large scale IC's, and complete decoding of the address within the IC's, the memory can be partitioned such that complete failure detection can be provided with multiple parity bits.

I/O Monitor - The analog-to-digital (A/D) converter and power supply can be performance tested by requiring the A/D converter to convert all of the regulated power supply voltages. These results are then compared against predetermined constants within the processor.

In addition to checking the A/D converter and power supply, this test checks a majority of the analog multiplexer and I/O control circuitry. The I/O and multiplex circuitry associated with the analog and discrete inputs will also automatically be self-tested as part of the comparison or in-line monitoring of the sensors and discrete inputs.

All outputs, analog and digital, can be tested via a "wraparound" check. This means taking the outputs and multiplexing them into the processor as though they were inputs. Since the analog outputs are obtained by digital-to-analog conversions, these outputs can be verified by comparing the data sent out with that coming in to see if they agree within the tolerances of the analog hardware. This test provides not only a check on the D/A converter and any multiplexing and sample holds that may be required, but is also an additional check on the input circuitry. The discrete outputs can be tested in exactly the same manner (i.e. the discrete sent out is compared against the discrete signal multiplexed back in).

Continuity Monitor - The primary function of the continuity monitor is to ensure the proper program flow through the critical instructions. This monitor checks for proper exit and sequence through the computations. If a failure is detected, the processor jumps to the "fail loop" and subsequent system disengagement, after that the watch dog timer has timed out.

Real Time Clock Monitor - The purpose of the Real Time Clock (RTC) monitor is to ensure that computational time between RTC restarts has not exceeded the designed computational period. RTC restarts are controlled by the RTC circuits. If the computational time were allowed to drift, all filter computations performed would have an effective frequency shift.

The RTC is a resetable hardware counter that counts down the processor clock. When the prescribed number of counts have elapsed since the update, the RTC provides a halt release signal. This discrete is checked to make sure the RTC has not timed out prior to executing the hold mode.

During real time computations, real time clock updates, are strategically placed in software to effectively detect CPU control and transfer failures. This is accomplished by using the real time clock updates in the software in a manner such that if the CPU updates it, it is an extremely reliable indication that the CPU is performing properly.

Watch Dog Timer - The purpose of the Watchdog Timer, WDT, is to protect against central processor or memory failures which prevent execution of a computation cycle in the prescribed period of time. It is designed to provide this protection without dependence on processor or memory functions. The essential element is a monostable single-shot flip-flop which has a high output state for about 1.5 times the expected computational cycle after receiving an update pulse. A low output state disengages the computer directly through the L_1, L_2, L_3 hardware logic. To maintain system engagement, the computer program checks that the WDT is not failed and then issues an output control pulse to update the flip-flop once every computation cycle.

Because the WDT does not require a functioning processor or memory to cause a disengagement, it is relied upon for that function when failures are detected by other processor or memory monitors. In those cases, failure detection causes the computation flow to jump to a "Fail Loop" which prevents update of the WDT.

Dynamic Computation Monitor - Table 4 shows that the above described monitors check most of the computational elements. Many of them are tested by more than one monitor. The sample and hold amplifiers are an exception however. The dynamic computation monitor (DCM) is added to test of sample and hold circuits. In addition it provides a redundant test of many other computer elements.

The DCM provides an independent and continuous test of CPU capability to perform control functions. It also checks that critical instructions take the correct time to execute.

The concept assumed for the EFCS digital computers and shown in Figure 14 defines a relatively precise control function which must be performed on an analog element by the CPU and I/O. The analog element to be controlled is an operational amplifier integrator. The objective of the control law contained in the software is to produce a stable $\pm$ 5 VDC, 20 Hz triangular integrator output. While performing the basic control function, the control law also maintains certain similarities to the control functions.

1. Exercise A/D and D/A conversion.
2. Samples input at 40 Hz rate and outputs a command at 40 Hz.

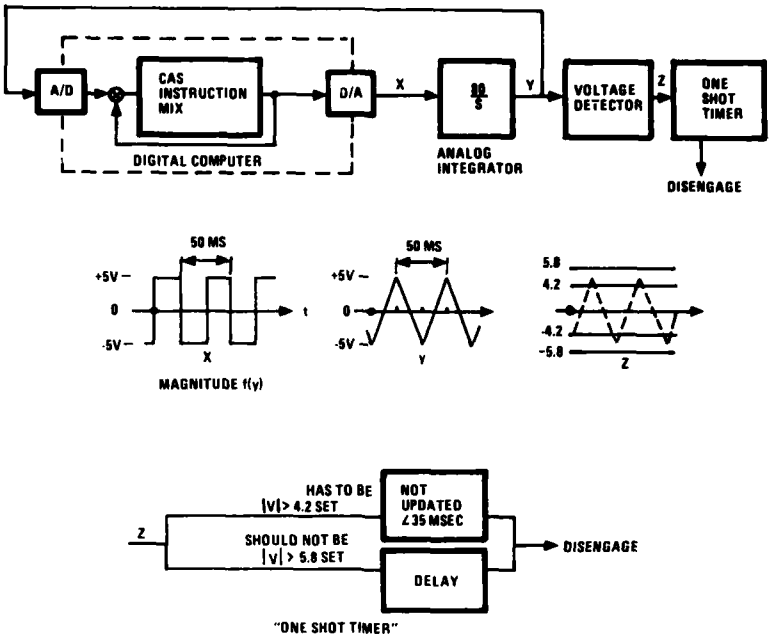


Figure 14. Dynamic Computation Monitor

3. Exercises much of the instruction repertoire used by the control law computations.
4. Relies on the Real Time Clock for maintenance of a precise computational interval.
5. Uses CONSTANT and SPAD memory.

The integrator output is monitored for peak values both above and below nominal. Failure to exceed a 4.2 VDC magnitude at least every 35 ms causes disengage. Exceeding 5.8 VDC magnitude at any time also causes disengagement.

Since the two DCM related subroutines straddle critical control law instructions this monitor checks that these instructions take the correct amount of time to execute. If they take too long a time, or too short a time, (indicating a wrong instruction sequence) the DCM monitor will trip.

BIT Computer Load - Each digital computer is exercising identical BIT-programs. Estimated amount of instructions, computational rates and computer load are listed in Table 5.

In addition to the number of instructions listed in Table 4 some of the software monitors include limited subroutines and the memory sum checks uses indexed summation. The total time to compute this 19000 instructions are about 100 milliseconds assuming an "average EFCS computer." When the subroutines and indexed summation is included the total time to perform the in flight BIT corresponds to about 200 m seconds.

In summary the in flight BIT envisioned for the EFCS Digital Computers is:

- o 9 Different tests of monitors are required
- o BIT includes about 500 instructions executed at 80, 40 or 20 Hz.
- o The BIT computational load including subroutines etc., is about 40 KOPS.
- o The BIT coverage is based on experience judged to be 99 percent.

TABLE 5. IN FLIGHT BIT INSTRUCTIONS AND COMPUTATIONAL RATE

COMPUTER MONITORS	NUMBER OF INSTRUCTIONS	COMPUTER	
		RATE HZ	LOAD
Power Supply Monitor	60	40	2400
CPU Self Test	200	40	8000
Memory Parity	15	40	600
Memory Sumcheck	40/40	40/20	2400
I/O Monitor	50	40	2000
Continuity Monitor	25	40	1000
Real Time Clock Monitor	10	80	800
Watch Dog Timer	5	80	400
Dynamic Computation Monitor	10/30	80/20	1400
TOTAL	495		19000

DETECTION DE PANNE DE CAPTEURS D'AVION PAR UTILISATION DE LA REDONDANCE ANALYTIQUE

Marc LABARRERE
CERT - DERA
BP. 40 25
31055 TOULOUSE Cedex (France)

SUMMARY

Among the different failure detection techniques using analytical redundancy, the authors look for those the most suitable to difficult flight conditions. The goal is to replace triplex vital systems by duplex systems associated with analytical redundancy. Thence the problem is mainly an isolation problem of the failed sensor for which estimation techniques seem well adapted because of the turbulence which cannot be ignored. To be implemented on board those techniques must be simple and robust.

Natural on-board redundancy may have different aspects : stochastics or deterministics, statics or dynamics and according to the case various estimation algorithms have been used :
- blender operating on various measurements
- estimation by observers or Kalman filters using one or several equations and one or several measurements
- autoadaptive techniques requesting the flight configuration identification are referred to.

According to these criteria the authors will present the solution which has been considered (choice and use of deterministic redundancy relations which are independent of atmospheric disturbances). This procedure has been applied to records of real flights.

RESUME

Parmi les différentes techniques de détection de pannes par utilisation de la redondance analytique, les auteurs présentent celles qui semblent le mieux convenir à l'aéronautique. L'objectif est de remplacer les systèmes triplex par des systèmes duplex associés à un calcul de redondance. De ce fait, ce problème est avant tout un problème d'isolation du capteur défaillant pour lequel les techniques d'estimation semblent bien adaptées en raison de la turbulence atmosphérique. Pour pouvoir être implantées sur le calculateur de bord, ces techniques doivent de plus être simples et robustes.

Suivant la nature de la redondance analytique, aléatoire ou déterministe, statique ou dynamique, des techniques différentes ont été utilisées :
- le mixage des observations
- l'estimation par des observateurs ou des filtres de Kalman utilisant une ou plusieurs équations et une ou plusieurs mesures.
- les procedures autoadaptatives par identification de la configuration de vol.

La décision capteur en panne est prise à partir de test de forme logique ou test séquentiel du rapport de probabilité.

Les auteurs présenteront la solution qu'ils ont retenue en fonction de différents critères : choix et utilisation de relations de redondance déterministes indépendantes de la turbulence atmosphérique. Cette procédure a été appliquée sur des enregistrementsde vol.

I - INTRODUCTION

L'exigence de performances dans l'aéronautique et les progrès récents de l'électronique et en particulier de l'électronique numérique ont conduit progressivement à utiliser dès le stade de la conception les théories modernes de la commande déjà en application dans le domaine spatial. Elles ont permis de mettre en oeuvre les principaux concepts de la commande automatique généralisée (CAG) : la stabilité artificielle,le contrôle direct des forces, la limitation des charges en manoeuvre, l'antiturbulence, l'antiflottement structural... Pour ces applications, la sécurité et la fiabilité deviennent fondamentales. Ces systèmes de commande de vol très sophistiqués et souvent tout électrique devront être hautement fiables et tolérants aux pannes afin d'assurer une fiabilité au moins comparable à celle des avions classiques à commandes mécaniques.

Etant donné la difficulté d'obtenir la fiabilité requise par une simple chaîne de commande, l'approche traditionnelle classique a consisté à installer en parallèle plusieurs chaînes de commande associées à une logique de choix plus ou moins complexe. Suivant la fiabilité désirée on utilise des chaînes duplex, triplex et même quadruplex. De tels systèmes sont très pénalisants en ce qui concerne le coût, le poids, le volume, la puissance consommée. T.F.Westermeier /20/ donne les variations de ces grandeurs lorsqu'on passe d'une configuration triplex à une configuration quadruplex :

Redondance	Maintenance	Poids	Coût
Triplex	1	1	1
Quadruplex	1.18	1.18	1.42

La sécurité obtenue par la juxtaposition de systèmes identiques pose de plus le problème de l'indépendance de leurs pannes. Des systèmes identiques de même conception fabriqués simultanément, travaillant dans la même ambiance ne vont-ils pas tomber simultanément en panne ? On aboutit à l'utilisation de chaînes de principe et de construction différents ; c'est la redondance dissemblable qui entraîne un nouvel accroissement des coûts de conception et de maintenance.

En ce qui concerne plus particulièrement les capteurs, la situation est encore plus critique. Les systèmes CAG nécessitent un grand nombre de capteurs qui interviennent pour une part importante dans la sécurité de l'ensemble. Les simples amortisseurs deviennent des systèmes sophistiqués avec beaucoup de capteurs. Outre les inconvénients précédemment cités la multiplication des capteurs pose le problème de leur implantation sur la structure, de la qualité des mesures obtenues et de leur environnement. Ainsi les emplacements de prises de données aérodynamiques sont limités et conduisent à des mesures différentes difficiles à comparer. La réduction du nombre de capteurs apparaît alors fondamentale. Elle peut être obtenue par différentes approches :

- intégration des différentes fonctions -en particulier les capteurs utilisés en navigation tels que les centrales à inertie peuvent être utilisés pour la stabilisation et le pilotage
- autotest des capteurs - certaines pannes de capteurs peuvent être détectées par des tests effectués directement sur le capteur lui même.
- modification des lois de commande. D'après les théories de l'observabilité et de la controllabilité l'état d'un système peut être reconstruit et commandé à partir d'un nombre limité de capteurs, ce nombre pouvant même se réduire à un seul capteur. Mais pratiquement, les performances du système de commande diminuent avec le nombre de capteurs. Cette reconstruction de signaux peut être obtenue au moyen d'observateurs ou de simples filtres. Ainsi, dans de nombreux pilotes automatiques la vitesse de tangage est obtenue par une fausse dérivation de l'assiette
- redondance analytique. Toutes les mesures accessibles à bord d'un avion ne sont pas incohérentes, il existe entre elles des relations issues de la mécanique du vol. Cette redondance fonctionnelle entre les différents signaux peut être exploitée pour obtenir la fiabilité requise en évitant une multiplication systématique des capteurs.

C'est cette dernière approche très prometteuse qui fait l'objet de ce papier. En effet, l'apparition des calculateurs numériques de bord pour la commande permet d'envisager l'utilisation de ces techniques plus sophistiquées pour la détection de pannes de capteurs à bord des avions et un certain nombre d'études de faisabilité ont été effectuées, l'objectif étant non pas de supprimer toute la redondance parallèle mais de la réduire sans altérer les performances et la sécurité.

Les techniques mises en oeuvre sont fondées sur les théories de l'estimation et de la décision et sur la connaissance d'un modèle de référence de l'avion qui constitue la redondance analytique.

II - EXIGENCES DE LA SECURITE

Actuellement, on considère en aéronautique que la probabilité de panne catastrophique ne doit pas dépasser 10^{-7} par heure et la probabilité d'abandon de mission 10^{-3} par heure. Ces exigences se traduisent sur les mesures par une probabilité de l'ordre de 10^{-9} par heure par mesure dont la perte est catastrophique et de 10^{-5} par heure pour les mesures dont la perte n'entraîne qu'un abandon de mission. La probabilité de panne d'un capteur pouvant atteindre 10^{-4} par heure cela conduit à définir des architectures redondantes spécifiques.

De manière classique, on distingue différents niveaux de tolérance aux pannes auxquels sont associées des configurations de redondance directe :

. "Fail Safe": lorsqu'une panne apparait elle est détectée mais non isolée. C'est le cas des systèmes duplex. Le système incriminé est déconnecté et la mission se poursuit en mode dégradé. Cela implique que le système considéré n'est pas essentiel pour la sécurité de l'avion et que le pilote pourra rétablir la situation avant la catastrophe.

. "Single fail operative": une panne peut être détectée, isolée et corrigée, le système devient alors "fail safe". Pour un système essentiel, cela se traduit par une mission dégradée après la première panne. C'est le cas des chaînes triplex.

. "dual fail operative": le système doit être capable de fonctionner normalement malgré la présence de la panne de deux éléments identiques. C'est le cas des chaînes quadruplex tout électrique . Bien que les performances ne soient pas dégradées après la deuxième panne, la mission est abandonnée car une troisième pourrait être catastrophique.

Les mesures seront "dual fail operative" si elles sont indispensables à la sécurité de l'avion ou simplement "single" si la mission peut être poursuivie dans des conditions "fail safe" après l'apparition d'une panne.

Dans le cas d'une réalisation multiplex idéale et dans l'hypothèse de pannes indépendantes, la probabilité de panne d'une mesure est suivant la redondance utilisée (duplex, triplex ou quadruplex) : $2Q_c$ $3Q_c^2$ ou $4Q_c^3$ où Q_c est la probabilité de panne par heure d'un capteur

La redondance analytique devrait permettre de diminuer ce niveau de redondance parallèle tout en satisfaisant les normes de sécurité.

Ainsi pour un système "single fail operative" puis "fail safe" constitué par un système triplex la probabilité d'abandon de mission $3Q_c^2$ est bien plus faible que celle désirée même pour un gyromètre dont la probabilité de panne peut atteindre 10^{-4} par heure ($3Q_c^2 = 3.10^{-8} << 10^{-5}$) Par contre, si on se contente d'un système duplex dont la panne est détectée par une simple logique et isolée par la redondance analyti-

que la probabilité d'abandon de mission devient $Q_c + 2Q_c Q_D$ ou Q_D est la probabilité de panne de la détection par la redondance analytique. Dans le cas du gyromètre cela conduit à une probabilité de panne de l'ordre de 5.10^{-2}, le diagnostic par la redondance analytique doit permettre dans 95% des cas de distinguer un bon capteur d'un mauvais. Cette performance semble envisageable mais elle sera obtenue au prix d'une plus grande complexité de calculs. Pour aboutir à une diminution du poids et du coût, les algorithmes devront cependant être suffisamment simples pour pouvoir être implantés sur le calculateur de bord de l'avion.

III - ANALYSE DE LA REDONDANCE ANALYTIQUE

La redondance analytique à bord d'un avion est constituée par les équations de la mécanique du vol qui régissent le mouvement de l'avion, le modèle des perturbations atmosphériques et les mesures.

III.1 - Equations générales du mouvement

Les équations générales du mouvement d'un avion s'obtiennent en projetant sur un repère les équations d'équilibre des forces et des moments.

D'où les équations de la dynamique de translation en projetant l'équation des forces sur le repère aérodynamique :

$$\begin{aligned} m\dot{V} &= F_X - mg\sin\gamma + T\cos\alpha\cos\beta \\ mV(-p\sin\alpha + r\cos\alpha + \dot{\beta}) &= F_Y + mg\cos\gamma\sin\mu - T\cos\alpha\sin\beta \\ mV(p\cos\alpha\sin\beta - (q-\dot{\alpha})\cos\beta + r\sin\alpha\sin\beta) &= F_Z + mg\cos\gamma\cos\mu - T\sin\alpha \end{aligned} \qquad (1)$$

où V représente la vitesse sol de l'avion, m la masse, F_X F_Y F_Z les composantes des forces aérodynamiques p q r les vitesses de roulis, tangage et lacet, α l'incidence, β le dérapage, γ la pente, μ le gite aérodynamique, g l'accélération de la pesanteur et T la poussée.

Et la dynamique de rotation obtenue par la projection de l'équation des moments sur le repère avion :

$$\begin{aligned} A\dot{p} + (C - B)\,q\,r &= L \\ B\dot{q} + (A - C)\,r\,p &= M \\ C\dot{r} + (B - A)\,p\,q &= N \end{aligned} \qquad (2)$$

où A, B, C représentent les inerties et L, M, N les composantes des moments.

A ces équations de la dynamique s'ajoutent les équations de changement de repères et en particulier les relations entre les angles du trièdre aérodynamique et du trièdre terrestre :

$$\begin{aligned} \sin\gamma &= \sin\theta\cos\alpha\cos\beta - \sin\beta\cos\theta\sin\phi - \cos\beta\sin\alpha\cos\theta\cos\phi \\ \cos\gamma\sin\mu &= \sin\theta\cos\alpha\sin\beta + \cos\beta\cos\theta\sin\phi - \sin\alpha\sin\beta\cos\theta\cos\phi \\ \sin\chi\cos\gamma &= \cos\alpha\cos\beta\sin\psi\cos\theta + \sin\beta(\cos\psi\cos\phi + \sin\psi\sin\theta\sin\phi) + \sin\alpha\cos\beta(-\cos\psi\sin\phi + \sin\psi\sin\theta\cos\phi) \end{aligned} \qquad (3)$$

où θ représente l'assiette, ψ l'azimut et ϕ le gite.

Les relations entre les vitesses de rotation :

$$\begin{aligned} p &= -\dot{\psi}\sin\theta + \dot{\phi} \\ q &= \cos\theta\sin\phi\,\dot{\psi} + \cos\phi\,\dot{\theta} \\ r &= \cos\theta\cos\phi\,\dot{\psi} - \sin\phi\,\dot{\theta} \end{aligned} \qquad (4)$$

ou les relations inverses :

$$\begin{aligned} \dot{\phi} &= p + \mathrm{tg}\theta\,(q\sin\phi + r\cos\phi) \\ \dot{\theta} &= q\cos\phi - r\sin\phi \\ \dot{\psi} &= \frac{q\sin\phi + r\cos\phi}{\cos\theta} \end{aligned} \qquad (5)$$

Et les équations des mesures des accélérations liées à l'avion :

$$\begin{aligned} \gamma_x &= \dot{V}\cos\alpha\cos\beta - V\sin\beta(r - \dot{\beta}\cos\alpha) + V\sin\alpha\cos\beta(q - \dot{\alpha}) + g\sin\theta \\ \gamma_y &= \dot{V}\sin\beta + V\cos\beta(-p\sin\alpha + r\cos\alpha + \dot{\beta}) - g\cos\theta\sin\phi \\ \gamma_z &= \dot{V}\sin\alpha\cos\beta + V\sin\beta(p - \dot{\beta}\sin\alpha) + V\cos\alpha\cos\beta(\dot{\alpha} - q) + g(1 - \cos\theta\cos\phi) \end{aligned} \qquad (6)$$

III.2 - Influence des perturbations

Les perturbations atmosphériques qui agissent sur un avion sont la turbulence et les gradients de vent. L'approche classique consiste à introduire l'effet des perturbations sur la mécanique du vol par la considération des composantes de la vitesse du vent dans un repère inertiel R_o ramené au centre de gravité de l'avion G.

Considérons les repères suivants liés au centre de gravité :

- R(G, X, Y, Z) le repère avion
- $R_a(G, X_a, Y_a, Z_a)$ le repère aérodynamique lié à la vitesse aérodynamique
- $R_v(G, X_v, Y_v, Z_v)$ le repère lié à la vitesse sol

Entre les différents repères, on définit les rotations :

$R \rightarrow R_a$ $\quad -\alpha_a \vec{GY} + \beta_a \vec{GZ}_a$

$R \rightarrow R_v$ $\quad -\alpha \vec{GY} + \beta \vec{GZ}_v$

$R_a \rightarrow R_v$ $\quad -\alpha_v \vec{GY}_a + \beta_v \vec{GZ}_v$

L'équivalence des rotations $R \to R_a \to R_v$ et $R \to R_v$ permet d'obtenir la relation aux petits angles

$$\alpha = \alpha_v + \alpha_a$$
$$\beta = \beta_v + \beta_a$$

La vitesse de l'avion par rapport au sol V est la somme de la vitesse aérodynamique et de la vitesse du vent :

$$V = V_a + V_v$$

ce qui donne dans l'approximation des petits angles :

$$\beta_v = + \frac{v}{V_a}$$

$$\alpha_v = + \frac{w}{V_a}$$

$$V_a = V - u$$

Comme v et w ont une distrubtion sur la longueur de l'avion, à cause des vitesses de rotation, il en est de même pour α_v et β_v.

La variation de la vitesse verticale du vent w sur la longueur de l'avion peut être considérée équivalente en ce qui concerne l'aérodynamique à une vitesse de tangage :

$$q_v = - \frac{\partial w}{\partial x}$$

En raisonnant de même avec les autres variations, on obtient les relations :

$$p_v = \frac{\partial w}{\partial y} \qquad q_v = - \frac{\partial w}{\partial x} \quad \text{et} \quad r_v = \frac{\partial v}{\partial x}$$

Ces variations de vitesse dues à la turbulence n'interviennent dans les équations de la mécanique du vol que par les termes aérodynamiques, les expressions inertielles ne sont pas modifiées. Ainsi dans le calcul des forces et des moments aérodynamiques, les composantes de la vitesse angulaire sont simplement remplacées par la différence entre la vitesse angulaire "inertielle" et la vitesse angulaire due au vent :

$$p_a = p - p_v$$
$$q_a = q - q_v$$
$$r_a = r - r_v$$

Les perturbations atmosphériques interviennent aussi sur les capteurs aérodynamiques, ainsi la girouette mesure l'incidence aérodynamique $\alpha_a = \alpha - \alpha_v$ et le Badin mesure la vitesse par rapport à l'air. Par contre, les capteurs inertiels ne sont pas affectés par la turbulence.

Les perturbations atmosphériques en général considérées sont de deux types :

- des rafales isolées de la forme :

$$u = \frac{u_M}{2} \left(1 - \cos \frac{2\pi V t}{\lambda}\right) \tag{7}$$

- de la turbulence correspondant au modèle classique de Dryden dans lequel les vitesses de la turbulence dans le repère inertiel R_o sont caractérisées par leur représentation spectrale :

$$\Phi_{uu}(\omega) = \sigma_u^2 \frac{L_u}{\pi V} \frac{1}{1+\left(\frac{L_u}{2\pi V}\omega\right)^2}$$

$$\Phi_{vv}(\omega) = \sigma_v^2 \frac{L_v}{2\pi V} \frac{1+3\left(\frac{L_v}{2\pi V}\omega\right)^2}{\left(1+\left(\frac{L_v}{2\pi V}\omega\right)^2\right)^2} \tag{8}$$

$$\Phi_{ww}(\omega) = \sigma_w^2 \frac{L_w}{2\pi V} \frac{1+3\left(\frac{L_w}{2\pi V}\omega\right)^2}{\left(1+\left(\frac{L_w}{2\pi V}\omega\right)^2\right)^2}$$

où les intensités σ_u, σ_v, σ_w et les paramètres L_u, L_v, L_w sont fonctions de l'altitude.

III.3 - Le modèle linéarisé d'un avion

Dans le cas d'un vol stabilisé et dans l'hypothèse de petits mouvements on peut se contenter d'un modèle linéarisé autour d'un point de fonctionnement. Les équations d'évolution et de mesure prennent la forme :

$$\dot{X} = A\,X + B\,U + E\,V \tag{9}$$
$$Z = C\,X + D\,U + F\,V + W$$

où X est le vecteur d'état, U le vecteur des commandes, V le vecteur des perturbations atmosphériques, Z le vecteur des mesures et W le vecteur des bruits de mesure .

Les matrices A, B, C, D, E, F sont fonctions du point de fonctionnement considéré et en particulier de la pression dynamique.

III.4 - Les capteurs

A bord d'un avion, on dispose de nombreuses informations. L'équipement de base est constitué par des mesures accélérométriques (γ_x ,γ_y ,γ_z), des mesures angulaires (α, β, ϕ, θ, ψ), desmesures de vitesse angulaire (p, q, r), les mesures de la vitesse aérodynamique,du Mach et de l'altitude. A coté de ces capteurs il faut noter des capteurs plus complexes comme les centrales à inertie qui fournissent simultanément plusieurs informations. Les grandeurs mesurées s'expriment en fonction des variables d'état de la dynamique de l'avion.

A ces relations, il faut ajouter les modèles des capteurs, leurs caractéristiques nominales et leurs statistiques de panne qui font partie intégrale de la redondance analytique.

Du point de vue de leur dynamique, les capteurs peuvent être modélisés par un second ordre bien amorti de bande passantesupérieure à 5 hertz. Cette dynamique peut donc être négligée dans l'étude d'une détection de pannes par redondance analytique qui ne prend en compte que les modes rigides dont la fréquence est inférieure à 2 hert². Par contre, il faut tenir compte des erreurs tolérées sur les capteurs qui permettent de distinguer le fonctionnement nominal d'un fonctionnement défectueux. La fiche technique fournie par le constructeur et les normes donnent les performances et en particulier les erreurs de zéro,de linéarité et de facteur d'échelle. Le bruit de mesure qui englobe les bruits électriques et des évolutions haute fréquence non modélisées telles que les vibrations de la structure peut être évalué à partir d'enregistrements de vol. L'ensemble de ces erreurs est en général modélisé par un bruit blanc gaussien passé dans un filtre dont la largeur de bande est égale à celle du capteur, auquel on ajoute un biais constant dont la taille maximum permet d'englober les autres erreurs.

Les statistiques de pannes de ces capteurs sont mal connues. Parmi les différentes pannes rencontrées on peut citer :

- la rupture du capteur qui délivre soit zéro, soit la valeur maximum, soit une évolution aléatoire
- le biais ou faux zéro
- la dérive du zéro
- l'hystérésis
- l'erreur de gain
- le blocage à une valeur atteinte
- l'augmentation du bruit
- la modification des caractéristiques dynamiques

Ce sont la rupture du capteur et les biais qui apparaissent le plus souvent dans l'analyse des défaillances de capteur. Ces différents types de panne peuvent être utilisés directement dans la procédure de détection ou pour tester les procédures de détection.

Toute cette connaissance a priori constitue la redondance analytique, elle peut être classée en trois catégories :

. la redondance directe où l'on dispose de plusieurs capteurs identiques en parallèle fournissant idéalement des sorties identiques. C'est cette redondance qui est utilisée dans les voteurs logiques

. la redondance statique dans laquelle les mesures sont liées à tout instant par des relations algébriques dépendant ou non du temps

. la redondance dynamique qui fait appel à l'évolution des signaux et qui se traduit par des relations intégro-différentielles entre les mesures

Elle peut être aussi stochastique ou déterministe suivant qu'elle prenne en compte ou non la modélisation stochastique des perturbations et des pannes.

A bord d'un avion, on rencontre les différents types de redondance et si la plupart des techniques de détection de panne peuvent être appliquées, elles ont été adaptées au problème aéronautique de la détection de panne de capteurs. Les mesures sont nombreuses et faiblement bruitées, le modèle de l'avion est assez bien identifié mais fonction de la configuration, par contre les perturbations atmosphériques qui affectent ce modèle sont importantes et mal connues. L'objectif n'est pas de supprimer les chaînes parallèles pour les remplacer par la redondance analytique mais pour l'instant simplement de les réduire, de passer d'un système triplex (ou quadruplex) à un système duplex (ou triplex) associé à un calcul de vraisemblance, sans altérer la sûreté de fonctionnement de l'ensemble. De ce fait, c'est plus un problème d'isolation de panne que de détection, la détermination de l'instant d'apparition de la panne étant obtenu par comparaison logique entre les informations identiques,ce qui simplifie notablement le problème en permettant d'employer les tests d'hypothèses.

Les procédures de détection développées sont constituées par un ou plusieurs filtres de diagnostic (observateurs de Luenberger ou filtre de Kalman) qui génèrent des signaux caractéristiques des pannes, à partir desquels est effectuée la décision.

IV - LES METHODES DE DETECTION

Suivant la connaissance du système utilisé, différentes techniques ont été développées et testées sur des simulations d'avions et même sur des enregistrements de vol.

IV.1 - Les méthodes statiques

La redondance statique est représentée par les équations de mesure qui s'écrivent dans le cas linéaire :

$$Z = H X + \varepsilon$$

où z est le vecteur de mesure à m composantes
X est un vecteur de grandeurs inconnues de dimension $n < m$

ε est le vecteur des bruits de mesure de statistique connue; on suppose que ces bruits sont blancs de moyenne nulle et de matrice de covariance connue R.

L'ensemble des relations de redondance est donné par la matrice C telle que :

$$C\,H = 0$$

ce qui entraîne en l'absence de panne :

$$C\,Z = C\,\varepsilon$$

Le vecteur $b = C\,Z$ obtenu est, en l'absence de panne, de moyenne nulle et de matrice de covariance CRC^T
En présence de panne les caractéristiques de b sont modifiées et différentes procédures ont été développées :

- test sur l'erreur d'estimation $\hat{\varepsilon}$ des observations :

$$\hat{\varepsilon} = Z - H\hat{X} \text{ avec } \hat{X} = (H^T H)^{-1} H^T Z$$

- seuil sur l'erreur entre la mesure Z_i et la mesure estimée par les moindres carrés à partir des autres mesures
- test sur la matrice de covariance du bruit R estimée par le maximum de vraisemblance ou de manière itérative par des moindres carrés pondérés.

Ces méthodes /21//11/ ont surtout été appliquées aux centrales inertielles strap down et ont débouché sur la notion de senseur oblique. La réalisation de ce système quadruplex "dual fail operative" nécessite 12 accéléromètres alors que 6 accéléromètres orientés convenablement offrent les mêmes possibilités.

Cette notion de capteur oblique peut se généraliser au cas où les capteurs sont différents mais liés par la relation statique :

$$Z = H\,X + \varepsilon$$

IV.2 - Comparaison dynamique des mesures - Les observateurs

Ces méthodes utilisent la connaissance dynamique du système mais font abstraction des perturbations atmosphériques qui n'interviennent que dans le choix des seuils. Cette connaissance a priori est représentée par le système d'équations :

$$\dot{X}(t) = f(X(t), u(t))$$

$$Z(t) = h(X(t), u(t))$$

où X(t) représente le vecteur d'état, Z le vecteur de mesure et u(t) celui des entrées connues.

Et dans le cas linéaire :

$$\begin{aligned} \dot{X}(t) &= F\,X(t) + G\,u(t) \\ Z(t) &= H\,X(t) \end{aligned} \qquad (10)$$

Les relations de redondance sont dynamiques, ce sont des relations intégro-différentielles qui relient les sorties en l'absence de bruits et qui s'écrivent :

$$C(p)\,Z(p) = 0$$

où le filtre C(p) est obtenu directement à partir des équations du système ou par la théorie des observateurs.

Les équations d'un avion se prêtent bien à la détermination de relations du premier ordre entre les sorties de la forme :

$$\dot{Z}(t) = f(Z(t), u(t))$$

La différence $Z(t) - f(Z(t), u(t))$ caractérisée statiquement à partir de la connaissance des bruits et des perturbations atmosphériques en l'absence de panne permet de détecter la panne d'une des mesures.

Ainsi, les relations entre les vitesses de rotation (4) donnent des relations de redondance indépendantes des perturbations atmosphériques qui permettent de détecter une panne sur les capteurs mesurant les angles d'Euler (ϕ, θ, ψ) ou les vitesses de rotation (p, q, r). Les signaux d'erreur sont obtenus par des combinaisons non linéaires des mesures filtrées par des constantes de temps choisies pour prendre en compte la dynamique de l'avion rigide, c'est à dire de l'ordre de 0.1 seconde, les dérivations sont alors remplacées par des filtres passe-haut.

L'exemple de la figure 1 représente le calcul du signal d'erreur correspondant à l'équation de roulis

$$\dot{p} = -\dot{\psi}\sin\theta + \dot{\phi}$$

qui est vérifiée quelles que soient les perturbations atmosphériques.

Pour ce signal, qui permet de détecter une panne sur les mesures de p et ϕ et éventuellement sur ψ et θ le seuil de détection ne dépend que des bruits de mesure.

Par contre, les autres équations (1) et (2) font intervenir les perturbations, les seuils de détection seront donc plus grands pour les prendre en compte.

T.B.Cunningham et R.D. Poyneer /7//8/, ont testé ce concept sur une simulation de l'avion A7D. A partir de cinq relations ils ont effectué un diagnostic sur les mesures de p, q, r, ϕ, θ, ψ , α, de la vitesse aérodynamique U et de l'altitude.

Cette procédure très simple à mettre en oeuvre donne de très bons résultats pour la détection des pannes des capteurs p, q, r, ϕ, θ, ψ en utilisant des relations entre les vitesses de rotation, mais des résultats bien plus pauvres en ce qui concerne les mesures aérodynamiques α et U qui nécessitent l'emploi de relations faisant intervenir les perturbations atmosphériques. En outre de faibles dérives sur les grandeurs qui sont dérivées, ne seront pas détectées si elles se traduisent par des biais sur les signaux d'erreur inférieurs aux seuils de détection.

La théorie des observateurs fournit aussi des filtres diagnostics et des signaux tests. Pour le système déterministe :

$$\dot{X} = F\,X + G\,u$$
$$Z = H\,x$$

un observateur

$$\dot{\hat{X}} = F\,\hat{X} + G\,u + K\,(Z - H\,\hat{X})$$
$$\hat{Z} = H\,\hat{X}$$

délivre une estimation exacte de la sortie en l'absence de perturbations et de panne. Un test sur l'innovation $(Z-\hat{Z})$ permet de détecter la présence d'une panne dans l'ensemble des mesures considérées.

N.Stückenberg /19/ applique ce concept pour l'isolation des capteurs défaillants de la chaîne duplex de stabilisation et de commande latérale de l'avion HF B 320. La détection est effectuée au moyen de deux observateurs déterministes de Luenberger, chacun opérant sur un jeu de capteurs avec une dynamique égale ou voisine du système pilote. Les seuils sur les innovations sont définis par l'influence maximum des perturbations (figure 2).

Pour la détection de panne des capteurs d'un hydroptère et en l'absence de redondance parallèle, R.N.Clark /4/5/6/, propose différentes structures à partir d'observateurs permettant d'estimer l'état à partir d'une seule mesure.

IV.3 - Diagnostic par filtrage de Kalman

Par rapport aux observateurs déterministes, le filtre de Kalman rajoute l'influence des perturbations et des bruits de mesure dans la détermination du filtre diagnostic. L'innovation, c'est à dire l'écart entre les sorties mesurées et les sorties estimées par un filtre de Kalman, possède des propriétés particulièrement intéressantes : c'est un bruit blanc de moyenne nulle, de variance connue, très sensible à l'apparition d'une panne qui se traduit par une modification de ces propriétés statistiques. Les diverses méthodes consistent à tester ces caractéristiques par des tests d'amplitude, de moyenne, de blancheur ou de variance.

Théoriquement, deux filtres de Kalman, chacun opérant sur un jeu de mesures remplissent la fonction isolation mais sur le plan pratique le filtre de Kalman présente des inconvénients; sa réalisation est relativement complexe surtout lorsqu'on envisage le modèle non linéaire de l'avion, les erreurs de modélisation peuvent entraîner des divergences du filtre et la signature de la panne dans l'innovation se prête plus ou moins bien à la détection de panne.

Pour pallier à ces inconvénients différentes procédures ont été développées à partir de réalisations stationnaires ou de batterie de filtres de Kalman étendus simplifiés, chacun n'utilisant qu'une ou plusieurs relations de redondance.

Ainsi Maybeck /15/ dans son étude sur le F4 considère les relations entre vitesses de rotation /5/ et estime les angles ϕ, θ, ψ à partir d'un filtre de Kalman étendu.

N'ayant pas de capteurs doublés, la détection et l'isolation de la panne sont obtenues par un test sur la fonction de vraisemblance d'une panne $L_N(i)$ portant sur les N dernières valeurs de chaque innovation :

$$L_N(i) = \sum_{j=i-N+1}^{i} \log p\left[e(j) \,|e(j-1),\ldots e(1)\right]$$

Tandis que T.B.Cunningham et R.D.Poyneer /7//8/ utilisent les mêmes relations, mais séparément. A l'estimation de l'angle, ils rajoutent celle d'un biais et d'un facteur d'échelle. Pour l'estimation de l'incidence et de l'altitude, ils considèrent des équations simplifiées reliant l'accélération normale à la dérivée de l'incidence et la dérivée de l'altitude à l'incidence et un modèle du premier ordre des perturbations atmosphériques. L'isolation est obtenue par un simple test logique ou par un test séquentiel de probabilité portant soit sur les innovations soit sur les fonctions de vraisemblance. La figure 3 représente la structure retenue pour rendre les vitesses angulaires "fail operative".

Cette solution avec des filtres de Kalman donne des résultats équivalents à la comparaison dynamique de mesures pour les capteurs de ϕ, θ, ψ, p, q et r mais des performances supérieures pour les autres et en particulier pour le capteur d'incidence.

J.C.Deckert, M.N.Desai, J.J.Deyst et A.S.Willsky /9//10/ utilisent séparément toutes les relations de redondance pour la détection de pannes des capteurs doublés de l'avion de la NASA F8C DFBW. Ils considèrent que les pannes ne sont que des biais; outre le fait que ce type de panne est courant, la procédure mise au point sur les biais doit permettre de détecter pratiquement toutes les pannes.

A chaque capteur sont associées une ou plusieurs relations de récurrence qui fournissent des résidus dans lesquels on recherche une signature de la panne par des tests du type SPRT. Le résidu est en général calculé par un filtre de Kalman simplifié intégré par Euler dont le gain est constant pour réduire l'influence des perturbations et nul dès que la panne apparaît, pour avoir de meilleures signatures.

Par exemple, l'isolation d'un capteur de Φ est obtenue à partir de l'équation cinématique de rotation :

$$\dot{\phi} = p + \dot{\psi} \sin \theta$$

par le calcul du résidu γ donné par les équations :

$$\hat{\Phi}'(t_n) = \hat{\Phi}(t_{n-1}) + \bar{p}_m\, T + \left[\psi_m(t_n) - \psi_m(t_{n-1})\right] \sin \bar{\theta}_m$$

$$\gamma_\phi(t_n) = \phi_m(t_n) - \hat{\phi}'(t_n)$$

$$\hat{\phi}(t_n) = \hat{\phi}'(t_n) + k\,\gamma_\phi(t_n)$$

où l'indice m indique les valeurs mesurées et $\bar{x}_m$ la moyenne de x_m sur l'intervalle $(t_{n-1}.t_n)$. Le gain k est choisi pour avoir une constante de temps d'environ 1/2 seconde.

Les tests et les seuils de détection sont définis en fonction des bruits de mesure, des perturbations et des biais tolérés pour les capteurs.

A ces tests séquentiels sur les résidus initialisés par la logique entre les deux jeux de capteurs sont rajoutés des tests séquentiels initialisés périodiquement et des tests de qualité des tests.

La procédure développée a donné d'excellents résultats en simulation et sur des enregistrements de vol.

IV.4 - Autoadaptation et détection

La plupart des relations de redondance sur lesquelles repose la redondance analytique, évoluent en fonction des conditions de vol. Dans le cadre des études sur l'autoadaptation menées à la NASA sur l'avion F8C, G.Stein et G.L.Hartman /18/ ont associé à l'identification en ligne du point de fonctionnement une procédure de détection de pannes de capteurs.

Le domaine de vol est quantifié en sous domaines $\mathcal{D}_i$ auxquels sont affectés des modèles M_i. Le modèle de l'avion est déterminé par la théorie des filtres multiples hypothèses qui nécessite un filtre de Kalman pour chaque hypothèse. Le modèle retenu est celui qui correspond à la fonction de vraisemblance maximum.

L'isolation de la panne est obtenue par l'addition d'un filtre de Kalman correspondant au modèle retenu opérant sur un deuxième jeu de capteurs et par un test séquentiel sur la différence des fonctions de vraisemblance (cf figure 4).

Bien que cette approche semble intéressante pour une détection de panne dans tout le domaine de vol et qu'elle ait donné de bons résultats en simulation, sa réalisation est complexe et délicate en particulier en ce qui concerne l'autoadaptation, l'isolation étant simplement effectuée par un filtre de Kalman.

Si sur le principe, l'approche par filtre de Kalman diffère fondamentalement de la simple combinaison des mesures, les procédures expérimentées sont très comparables, le filtre de Kalman fournissant surtout un guide pour la détermination des filtres délivrant les signaux caractéristiques des pannes.

V - LES MONITEURS

Dans les procédures, la logique de détection est aussi importante que la détermination du filtre, c'est elle qui conditionne les performances finales de l'ensemble de la chaîne. On distingue des logiques à seuils et des tests séquentiels.

V.1 - La logique à seuil

C'est une comparaison du signal d'erreur à des seuils choisis pour avoir des taux de fausse alarme et de non-détection convenables.

Si les signaux en l'absence de panne sont supposés blancs gaussiens de moyenne nulle et de variance connue, pour satisfaire les taux de fausse alarme désirés par un simple test d'amplitude il faut choisir des seuils très grands et la probabilité de détection est faible.

Pour satisfaire les performances plusieurs solutions sont proposées :

- la déclaration d'une panne si le seuil est dépassé plusieurs fois consécutives. Pour avoir un taux de fausse alarme inférieur à 1 pour 1000 heures de vol, T.B.Cunningham et R.G.Poyneer /7//8/ utilisent trois tests consécutifs et un seuil égal à 3,5 σ. L'écart type σ est calculé à partir des bruits de mesure et des perturbations atmosphériques

- la comparaison d'une moyenne du signal sur un intervalle ou du signal filtré à un seuil, cela revient à faire un test binaire à partir des mesures considérés dans la fenêtre sur les hypothèses.

- H_o : pas de panne, le signal d'erreur est de moyenne nulle et de variance connue
- H_1 : panne, le signal a même variance mais une moyenne +m ou -m.

Le seuil et l'intervalle (ou la constante de temps) sont choisis pour avoir des taux de fausse alarme et de non détection donnés.

Ces paramètres résultent aussi d'un compromis entre ces performances et le retard pour déclarer une panne.

Deckert et al. /9/ /10/ utilisent ce test pour initialiser un test séquentiel de validation de la panne plus performant.

V.2 - Test séquentiel du rapport de probabilité (SPRT)

A partir de n observations x_1, $x_2 \ldots x_n$ le test d'hypothèse binaire décide quelle hypothèse H_o ou H_1 doit être retenue suivant la valeur du rapport de probabilité :

$$\Lambda_n = \frac{p(x_1, x_2 \ldots x_n \mid H_o)}{p(x_1, x_2 \ldots x_n \mid H_1)}$$

Le test séquentiel du rapport de probabilité (SPRT) ne fixe pas a priori la taille de l'échantillon utilisé pour la décision, à tout instant il se réserve la possibilité de reporter la décision si l'information est insuffisante.

Suivant la valeur du rapport de probabilité, il y a trois possibilités :

- accepter H_o si $\Lambda_n \leq A$
- accepter H_1 si $\Lambda_n \geq B$
- ne pas choisir et attendre une information supplémentaire si $A < \Lambda_n < B$

Les bornes A et B sont fixées en fonction des taux de fausse alarme et de non-détection et le SPRT minimise le nombre moyen d'observations nécessaire pour effectuer la décision.

Pour une séquence blanche gaussienne de variance connue σ, de moyenne nulle dans l'hypothèse H_o et de moyenne m dans l'hypothèse H_1, cela s'écrit :

$$\text{Log } A + k\,\frac{m^2}{2\sigma^2} \underset{H_o}{<} m\,\frac{\sum_{i=1}^{k} x_i}{\sigma^2} \underset{H_1}{<} \text{Log } B + \frac{km^2}{2\sigma^2}$$

Ce test permet de détecter dans la séquence la présence d'un biais m donc d'une panne. Si le signe du biais n'est pas connu, on peut rechercher soit un biais +m soit un biais -m ce qui se ramène finalement au test :

$$\text{Log } A + \frac{km^2}{2\sigma^2} \underset{H_o}{<} m\,\frac{\left|\sum_{i=1}^{k} x_i\right|}{\sigma^2} \underset{H_1}{<} \text{Log } B + \frac{km^2}{2\sigma^2}$$

qui est représenté par la figure (5).

Notons que si le biais est en moyenne inférieur à m/2, on choisira l'hypothèse H_o, sinon l'hypothèse H_1.

Bien que le SPRT doive conclure par une des deux hypothèses, il est préférable de limiter le temps de détection et de considérer pour le SPRT une quatrième possibilité "pas de panne" si le temps de décision est supérieur à une certaine valeur T.

A partir des résidus des deux filtres de Kalman correspondant à deux jeux de capteurs, un SPRT sur la variation de la différence de fonctions de vraisemblance réalise la fonction d'isolation . Ce signal

$$l_n = \Delta L_n^{(2)} - \Delta L_n^{(1)} = \frac{1}{2}\left[\gamma_i^{(1)}\ \Lambda^{-1}\ \gamma_i^{(1)T} - \gamma_i^{(2)}\ \Lambda^{-1}\ \gamma_i^{(2)T}\right]$$

où Λ est la matrice de covariance des innovations γ_i, est supposé blanc et gaussien.

Dans l'hypothèse H_o, le jeu de capteurs 1 est en panne, l_n a une moyenne +m alors que dans l'hypothèse H_1 c'est le jeu de capteurs 2 et l_n a une moyenne -m.

D'où la décision :

- capteur 1 en panne si $\Sigma l_n \geqslant \sigma^2 \frac{\text{Log } B}{m}$
- capteur 2 en panne si $\Sigma l_n \leqslant \sigma^2 \frac{\text{Log } A}{m}$
- Pas de décision si $\sigma^2 \frac{\text{Log } A}{m} < \Sigma l_n < \sigma^2 \frac{\text{Log } B}{m}$

Ce dernier test a été utilisé par T.B.Cunningham et R.D.Poyneer et par G.Stein et G.L.Hartman dans les procédures représentées aux figures (3) et (4). Notons que ce test ne peut conclure que par une hypothèse de panne, le taux de fausse alarme ne dépend donc que de la logique de détection d'apparition d'une panne.

Pour tester une hypothèse, le SPRT est particulièrement bien adapté, il délivre la décision dans un temps minimum. Les performances apparaissent supérieures à celles des simples logiques à seuil. Dans les "monitoring" de systèmes duplex avec redondance analytique, les logiques à seuils seront principalement utilisées pour la détection de l'instant d'apparition de la panne et les SPRT pour l'isolation du capteur défaillant

VI - EXEMPLE D'APPLICATION /14/

A partir de ces filtres diagnostics et de ces moniteurs une procédure de détection a été étudiée pour l'avion NORD 262 et ses capteurs longitudinaux.

L'objet de cette étude était l'utilisation de la redondance analytique, l'analyse des performances "fail operative" d'une chaîne duplex associée à un calcul de vraisemblance, la procédure développée devant être suffisamment simple pour être implantée sur le calculateur de bord.

VI.1 - Modèle du Nord 262

Les évolutions longitudinales de l'avion sont représentées par un modèle linéaire obtenu par linéarisation des équations générales du mouvement autour d'un point de fonctionnement. Elles ont la forme classique :

$$\dot{X} = A\,X + B\,U + E\,V$$
$$Z = C\,X + D\,U + F\,V$$

où X est le vecteur d'état (V, α, θ, q), U le vecteur des commandes, V le vecteur des perturbations verticales et longitudinales et Z le vecteur des mesures.

Cet avion est équipé d'une centrale qui délivre la vitesse sol V, l'assiette θ, l'accélération verticale γ_{z_o} d'une girouette (α), d'une mesure de vitesse air U, d'accéléromètres (γ_x, γ_z) et d'un gyromètre (q).

A ces mesures, il faut ajouter le braquage de la gouverne de profondeur δ_m. L'autre grandeur de commande, la poussée, qui n'était pas accessible a été supposée inconnue donc traitée comme une perturbation. L'ensemble des erreurs admissibles nominales ont été regroupées en deux types d'erreurs : un biais et un bruit caractérisé par son écart-type dont les amplitudes sont données par le tableau ci-après.

Mesures	Biais max.	Bruit (σ)	Variations maximales
V m/s	0.75	0.02	20
α °	0.2	0.05	10
θ °	0.2	0.01	15
q °/S	0.1	0.05	10
u m/S	1	0.25	20
γ_x m/S^2	0.1	0.03	3
γ_z m/S^2	0.1	0.03	10
γ_{z_o} m/S^2	0.1	0.03	10

VI.2 - Choix d'un filtre diagnostic

Le modèle des perturbations étant mal connu, nous nous sommes limités à l'utilisation de la redondance déterministe, c'est à dire aux relations integro-differentielles entre les mesures indépendantes des perturbations atmosphériques.

Ces relations de redondance : $\sum_{i=o}^{p} (P_i\, z^{(i)} + Q_i\, u^{(i)}) = 0$

s'obtiennent par élimination de l'état X, des entrées inconnues V et de leurs dérivées $X^{(i)}$ et $V^{(i)}$ entre les différentes équations du modèle et leurs dérivées.

Pour le premier ordre, on a :

$$\begin{vmatrix} Z \\ U \\ \dot{Z} \\ \dot{U} \end{vmatrix} = \begin{vmatrix} C & D & F & 0 & 0 \\ 0 & I & 0 & 0 & 0 \\ CA & CB & CE & D & F \\ 0 & 0 & 0 & 0 & I \end{vmatrix} \begin{vmatrix} X \\ U \\ V \\ \dot{U} \\ \dot{V} \end{vmatrix} \qquad \text{ou} \quad \mathcal{Z} = \mathcal{C}\mathcal{X}$$

Les relations de redondance expriment la dépendance linéaire entre les lignes de la matrice $\mathcal{C}$, elles sont données par la matrice $\mathcal{R}$ telle que $\mathcal{R}\mathcal{Z} = 0$ c'est à dire telle que $\mathcal{R}\mathcal{C} = 0$

Cette équation matricielle ne suffit pas à définir la matrice $\mathcal{R}$, il faut fixer des contraintes supplémentaires sur la matrice $\mathcal{R}$, c'est à dire sur les relations de redondance.

L'ensemble des relations indépendantes est obtenue par une triangularisation de la matrice $\mathcal{C}$ qui a été effectuée par un algorithme de calcul dérivé de la méthode de Gauss Jordan.

Pour rechercher d'une façon exhaustive toutes les relations de redondance entre p composantes de $\mathcal{Z}$ toutes les combinaisons de p composantes de $\mathcal{Z}$ ont été calculées, puis on a cherché une relation entre ces p composantes.

Cette approche permet d'obtenir toutes les relations de redondance entre les sorties et leurs dérivées indépendantes des entrées inconnues. En particulier, on retrouve toutes les relations statiques et l'équation $\dot{\theta} = q$.

Ces relations de redondance indépendantes des perturbations atmosphériques peuvent être utilisées pour estimer les sorties et générer des signaux d'erreur. Chaque relation permet de calculer au bruit de mesure près une grandeur en fonction des autres et de leurs dérivées.

Pour les relations statiques, l'expression est évidente, elle s'écrit :

$$z_i = \sum_{j=1}^{m} \alpha_j\, z_j + \sum_{j=1}^{l} \beta_j\, u_j \qquad \text{avec} \quad \alpha_i = 0$$

Par contre, les relations dynamiques du premier ordre nécessitent un préfiltrage par exemple par un filtre du premier ordre.

Les relations prennent la forme :

$$\frac{z_i}{1+Tp} = \sum_{j=1}^{m} \left(\frac{\alpha_j + \gamma_j\, p}{1 + Tp}\right) z_j + \sum_{j=1}^{l} \left(\frac{\beta_j + \delta_j p}{1 + Tp}\right) u_j$$

ce qui revient à remplacer les dérivations par de fausses dérivations sans introduire d'erreur sur la relation.

Finalement, que l'on soit en statique ou en dynamique, si on considère des mesures filtrées, les relations s'écrivent :

$$z_{if} = \sum_{j=1}^{m} (\alpha_j\, z_{jf} + \gamma_j\, z_{jf}) + \sum_{j=1}^{l} (\beta_j\, u_{jf} + \delta_j\, u_{jf}) \qquad \text{avec} \quad \alpha_i = 0$$

où l'indice f indique que la grandeur est filtrée.

La constante de temps T du filtre est choisie en fonction d'un compromis, grande pour réduire l'influence du bruit mais faible pour ne pas trop filtrer les pannes. Nous avons retenu T = 0,2 sec.

Dans la mesure où l'on n'utilise pas l'ensemble des relations de redondance la qualité de l'estimation et de la détection dépend donc du choix de ces relations.

Ce choix est effectué en fonction des objectifs recherchés de qualité, de simplicité et de robustesse de la détection, c'est à dire en fonction d'un multicritère :

- bruit et biais minimal sur la reconstitution qui sont significatifs de la qualité de l'estimation. Pour chaque relation, ces grandeurs sont calculées en fonction du bruit et des biais admissibles sur les capteurs.
- nombre minimal de mesures intervenant dans une relation pour avoir des relations aussi simples que possible. En outre pour simplifier les relations la participation maximum de chaque mesure $|\alpha_{ji}\ z_{imax}|$ est comparée à l'erreur de reconstitution ; si elle est inférieure à la moitié de l'écart type, la mesure est éliminée dans la relation.
- découplage maximal des relations pour réduire les conséquences d'une fausse détection. Ce critère se traduit par un nombre maximal de zéros sur une colonne de la matrice des relations, alors que le précédent portait sur les lignes.
- insensibilité au point de fonctionnement. Le modèle de l'avion et les relations de redondance qui en sont déduites dépendent de la vitesse, du centrage, de l'altitude ; par un calcul correspondant à différents points de fonctionnement, on recherchera les relations les plus insensibles.
- si de plus les probabilités de panne des différents capteurs sont connues on peut calculer la probabilité de panne de la reconstitution. En particulier si une panne peut affecter deux mesures, on recherchera si possible pour reconstituer l'une de ces grandeurs une relation qui ne fasse pas intervenir l'autre.

Ainsi pour toutes les mesures à l'exception de la vitesse inertielle V des relations ont été retenues En effet, seule la dérivée de cette vitesse peut être calculée à partir des mesures considérées, le test ne pourra porter que sur la dérivée de la vitesse.

VI.3 - Organisation de la procédure

L'ensemble de la procédure correspondant à une mesure est décrit par la figure 6. A chaque instant $k\Delta t$ toutes les grandeurs mesurées sont filtrées et on calcule leurs dérivées :

$$z'_f(k)\ ,\ z''_f(k),\ \dot{z}'_f(k),\ \dot{z}''_f(k)$$

Un test logique sur l'écart entre les mesures correspondant à une même grandeur fournit l'instant d'apparition de la panne :

$$\Delta(k) = \left|z'_f(k) - z''_f(k)\right|$$

Une panne est déclarée sur la mesure i lorsque la composante $\Delta_i(k)$ est supérieure à un seuil de détection ε_i calculé pour tenir compte des bruits de mesure et des biais admissibles.

Lorsqu'une panne est détectée, trois tests d'hypothèses séquentiels (SPRT) sont initialisés :

- l'un, le SPRT de détection portant sur la différence entre les mesures filtrées $\Delta_i(k)$ pour corroborer l'hypothèse panne sur la ième mesure détectée par la logique à seuil
- et deux SPRT d'isolation portant sur les écarts entre les mesures filtrées z'_{fi} et z''_{fi} et leur estimation $\hat{z}_{fi}$ pour isoler le capteur défaillant

Dans l'hypothèse H_o le signal testé est supposé être un processus stochastique blanc gaussien de moyenne nulle et d'écart type σ alors que dans l'hypothèse H_1 qui correspond à un fonctionnement anormal, il est de moyenne $\pm$ m. Cela revient à tester si le biais du signal d'erreur est compris entre -m/2 et +m/2 où m/2 est le biais admissible sur le signal testé calculé à partir de l'ensemble des biais admissibles, et l'écart type σ à partir des bruits.

Les taux de fausse alarme et d'alarme manquée qui interviennent dans les seuils de décision du SPRT ont été fixés à 10^{-4}. De plus, le temps de décision des SPRT est limité à 10 secondes.

La reconstitution $\hat{z}_{fi}$ nécessaire au lever du doute est calculée à partir des mesures validées par la logique à seuil. Les capteurs retenus sont les plus proches des prédictions effectuées à partir des capteurs retenus à l'étape précédente :

$$z(k) = z_f(k-1) + \dot{z}_f(k-1)\Delta t$$

Ce choix constitue un filtrage logique sur les mesures utilisées.

L'ensemble de la logique qui gère les informations de la logique à seuil et des différents SPRT est représenté sur la figure 7 . Le SPRT de détection qui travaille directement sur les mesures filtrées est prioritaire : si la panne détectée par la logique à seuil n'est pas confirmée, l'hypothèse panne est abandonnée et l'ensemble de la procédure est réinitialisée ; par contre, si l'hypothèse panne est confirmée, suivant les SPRT d'isolation un capteur est déclaré en panne et éliminé des mesures ou bien une alarme est fournie.

Pour les SPRT d'isolation, un capteur est déclaré en panne si le SPRT correspondant retient l'hypothèse panne. Une alarme est donnée et la procédure de détection est réinitialisée lorsqu'il apparait certaines configurations correspondant à des incompatibilités entre les différentes hypothèses retenues par les SPRT alors que l'hypothèse panne a été confirmée. Ces alarmes sont dues aux taux de fausse alarme et d'alarme manquée retenus pour les seuils des SPRT, aux seuils d'erreurs admissibles sur les différentes mesures mais aussi à des pannes que la procédure ne permet pas de détecter telles qu' une faible dérive sur la vitesse inertielle. Dans ce cas, le SPRT de détection qui porte sur les mesures de vitesse filtrées détecte la panne lorsque la dérive dépasse le seuil de détection ; par contre les SPRT d'isolation qui portent sur les dérivées des mesures et sur leur reconstitution ne détecteront pas de panne si la dérive est trop faible.

Pendant la procédure de détection, que le pilote automatique soit enclenché ou non, pour éviter de

fournir des informations erronées ou retardées telles que la reconstitution on retiendra pour le pilotage la sortie capteur la plus proche de l'estimée fournie par les relations de redondance.

VI.4 - Résultats

La procédure développée a fait l'objet de nombreux tests, d'une part sur des mesures fournies par une simulation non linéaire à six degrés de liberté du NORD 262 piloté en automatique ou en manuel au moyen d'un minimanche et d'autre part sur des enregistrements de vol. Différents types de pannes ont été introduits: des blocages à valeur atteinte, des ruptures de capteurs, des biais et des dérives.

Les enregistrements ont été recueillis lors de vols de démonstration du NORD 262 qui ne correspondent pas exactement à un vol longitudinal stationnaire, le point de fonctionnement retenu correspond à une configuration moyenne.

Malgré les perturbations atmosphériques et les variations de poussée qui ne sont pas accessibles à la mesure, les estimations permettent de recalculer les mesures avec une bonne précision comme le montre la figure 8. La reconstitution est peu sensible à des variations relativement importantes d'altitude ou de vitesse. Par contre, l'hypothèse de vol longitudinal est fondamentale, les erreurs d'estimation deviennent prohibitives lors de virage important.

Un grand nombre de détection de pannes a été effectué en vol longitudinal, les différentes pannes introduites ont été isolées à l'exception bien sûr des faibles dérives sur la vitesse que la procédure ne peut traiter. Les figures 9 et 10 représentent les mesures, la reconstitution et les instants de détection et d'isolation pour des pannes différentes.

Cette procédure dont l'implantation sur le calculateur de bord est très simple à mettre en oeuvre s'est, pour cette application, avérée plus performante et surtout moins sensible à des erreurs de modèle qu'une précédente méthode utilisant un filtre de Kalman.

Cette approche semble intéressantepour le vol stabilisé que ce soit la croisière ou les phases critiques de décollage ou d'atterrissage. Son extension à tout le domaine de vol, qui n'a pas été étudiée, nécessiterait la prise en compte de phénomènes non linéaires et en particulier l'utilisation des équations de la cinématique de rotation.

VII - CONCLUSION

Après avoir étudié le problème de la sécurité des capteurs à bord des avions, nous avons présenté les principales méthodes développées en aéronautique pour la détection de panne de capteurs par utilisation de la redondance analytique, que ce soit pour la détermination des filtres diagnostics ou pour celle des moniteurs. A notre connaissance, il n'existe pas d'avion équipé d'un tel système et les procédures sont encore au stade de l'expérimentation en simulation et sur des données réelles de vol. Dans les études menées on note un souci général de robustesse aux imperfections du modèle, de simplicité des procédures pour pouvoir les implanter sur le calculateur de bord et malgré tout de performances pour assurer une sécurité comparable à celles des systèmes multiplex. Pour ce faire, les techniques sont adaptées à la forme particulière des équations de la mécanique du vol et à la structure de capteurs.

Assurer le caractère "fail operative" à des systèmes duplex apparait comme une limitation des possibilités de la redondance analytique mais c'est un problème réel et une étape fondamentale dans le développement de ces techniques.

En ce qui concerne les performances : pour les mesures des angles ϕ, θ, ψ et des vitesses angulaires p, q, r, les performances obtenues sont excellentes mais les relations (4) qui les lient sont exactes, valables pour tout le domaine de vol et indépendantes des perturbations atmosphériques. Par contre, pour les mesures aérodynamiques qui font intervenir le modèle de l'avion et surtout les perturbations atmosphériques, les techniques sont moins performantes. Pour ces grandeurs l'emploi de filtres de Kalman est plus satisfaisant bien qu'il nécessite un modèle des perturbations. La généralisation à tout le domaine de vol et pour tous les capteurs des relations de redondance développées au paragraphe VI pour le modèle linéaire semble intéressante.

Dès à present outre le fait qu'elle fournit une redondance dissemblable, la redondance analytique offre pour de nombreux capteurs des performances comparables sinon supérieures à celles des structures parallèles. Son implantation limitée à ces capteurs peut être envisagée pour en réduire le nombre ou pour accroitre la sécurité de l'avion.

BIBLIOGRAPHIE

/1/ A.S.WILLSKY. A survey of design methods for failure detection in dynamic systems. Automatica, vol.12 pp 601-611, 1976

/2/ A.V.MOZGALEVSKII. Technical diagnostics: continuous systems(survey). Automatika i telemekhanika, janvier 1978

/3/ M.LABARRERE, M.PIRCHER, S.TRABULSI. Détection de pannes de capteurs par utilisation de la redondance analytique(étude bibliographique). Rapport intermédiaire 1/7170 DERA/DRET, Convention DRET n°77408

/4/ R.N.CLARK. A simplified instrument failure detection scheme IEEE Transaction on Aerospace, Vol.AES 14 N°/4 - Juil. 1978.

/5/ R.N.CLARK. Instrument fault detection. IEEE Transaction Aérospace Vol.AES 14, N°3, mai 1978

/6/ R.N.CLARK. The dedicated observer approach to instrument failure detection. IEEE 1979 Decision & Control - Décembre 1979

/7/ T.B.CUNNINGHAM, R.D.POYNEER. Fault tolerant digital flight control using analytical redundancy. IEEE NAECON 77 Record 111

/8/ T.B.CUNNINGHAM, R.D.POYNEER. Sensor failure detection using analytical redundancy. Joint Automatic Control conference 1977, proceedings 278-287

/9/ J.C.DECKERT, M.N.DESAI, J.J.DEYST, A.S.WILLSKY. F8 DFBW sensor failure identification using analytic redundancy. IEEE Trans. Automatic Control, vol AC 22, n°5, octobre 1977

/10/ J.C.DECKERT, M.N.DESAI, J.J.DEYST, A.S.WILLSKY. Reliable dual Redundant sensor failure detection and identification for the NASA F-8 DFBW aircraft. NASA CR 2944, février 1978

/11/ E.GAI, J.V.HARRISON, K.C.DALY. "FDI performance of two redundant sensor configurations" IEEE Aerospace Vol AES 15, n°13, Mai 1979

/12/ R.C.HENDRICK, C.D.HILL. Self testing digital flight control applications. AIAA Paper 75 568

/13/ M.LABARRERE, B.GIMONET, A.BUCHARLES. An estimation technique applied to failure detection. Congrès IFAC, Helsinki, Juin 1978

/14/ M.LABARRERE, M.PIRCHER, B.GIMONET, A.BUCHARLES, F.ESCAFFRE. Recherche de méthodes de détection de pannes de capteurs. Rapport DRET/DERA n°3/7170, Toulouse, Décembre 1979

/15/ P. MAYBECK. Failure detection through functional redundancy. Technical report AFFDL TR 76 93

/16/ R.C.MONTGOMERY, AK. CAGLAYAN. A self reorganizing digital flight control system for aircraft. AIAA 12th Aerospace Sci. Meeting, Washington, Feb. 1974

/17/ M.J.PELEGRIN, M.LABARRERE, M.PIRCHER. Automatic recovery after sensor failure on board. SYmposium AGARD on Advances in guidance and control systems using digital techniques. OTTAWA, Mai 1979

/18/ G.STEIN, G.L.HARTMAN. F8C adaptive flight control extensions. Rapport NASA. CR 2881

/19/ N.STUCKENBERG. An observer system for sensor failure detection and isolation in digital flight control system. AGARD CP 272, OTTAWA, Mai 1979

/20/ T.F.WESTERMEIER. Redundancy management of digital FBW systems. Proceeding of the Joint Automatic control conference 1977, pp 272-277

/21/ J.C.WILCOX. Competitive evaluation of failure detection algorithms for strapdown redundant inertial instuments. AIAA Guidance and control conference, Key Biscayne, Florida, Août 1973

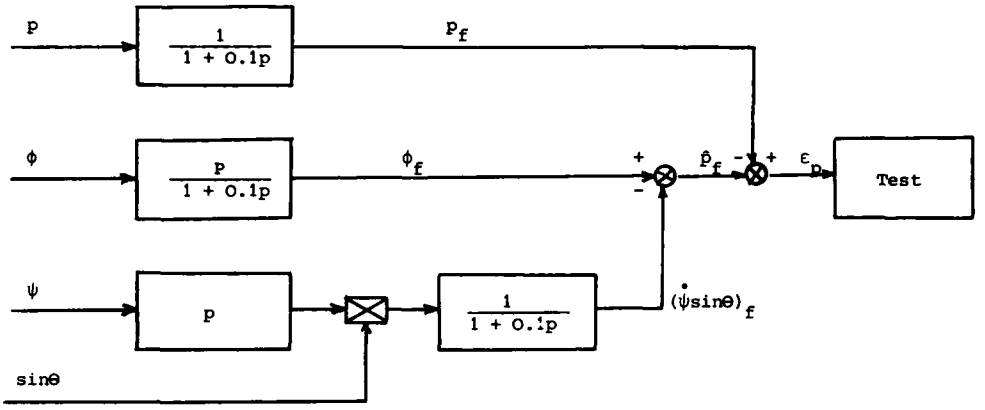


FIGURE 1 : Test sur l'équation de roulis

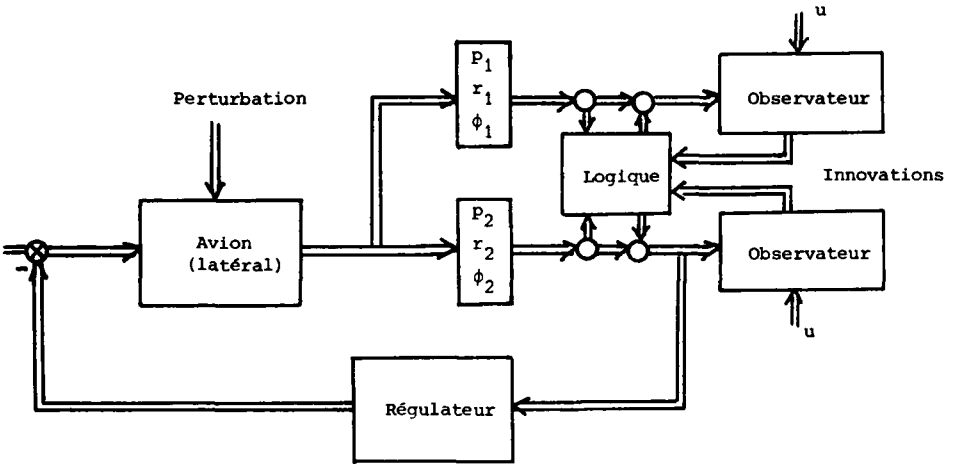
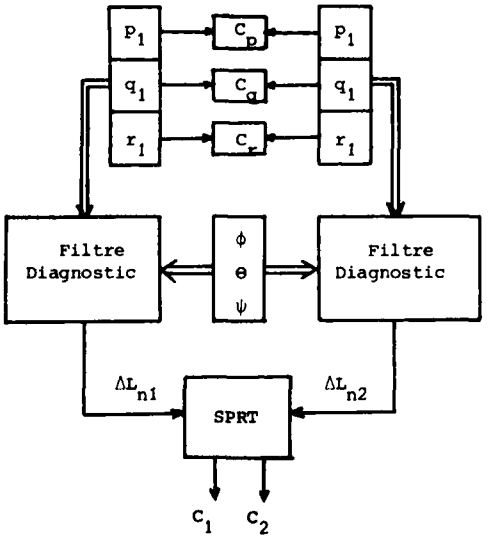
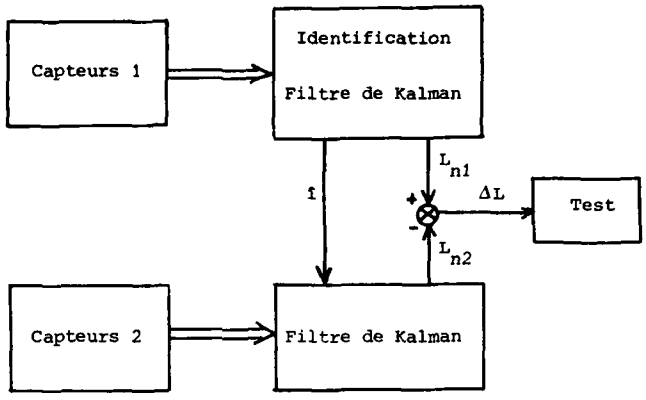


FIGURE 2 : CSAS avec détection de panne (réf.19)



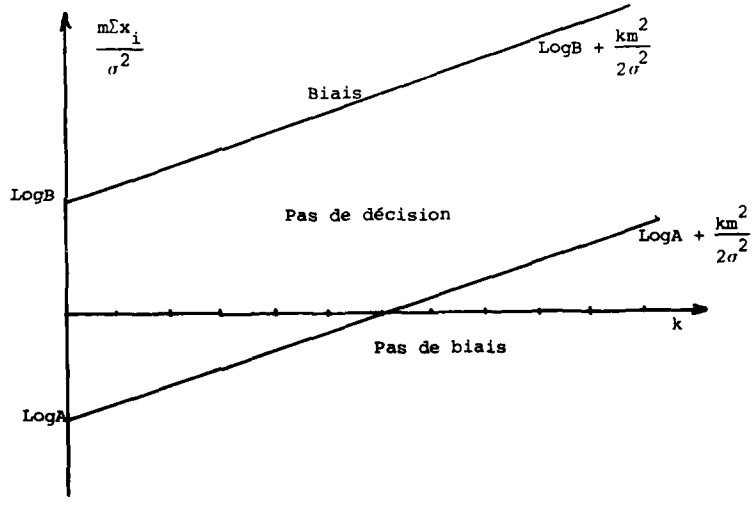
PANNE			p_1	q_1	r_1	p_2	q_2	r_2
COMPARATEUR	C_p		1			1		
	C_q			1			1	
	C_r				1			1
SPRT	C_1		1	1	1			
	C_2					1	1	1

FIGURE 3 : Détection de panne sur les vitesses angulaires (Ref. 8)

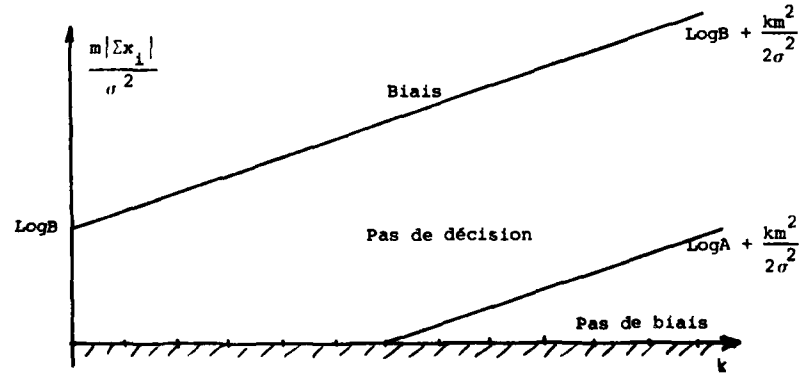


f : configuration retenue par l'identification

FIGURE 4 : Autoadaptation et détection de panne (réf. 18)



a) Biais de signe connu



b) Biais de signe inconnu .

FIGURE 5 : Recherche d'un biais par le SPRT

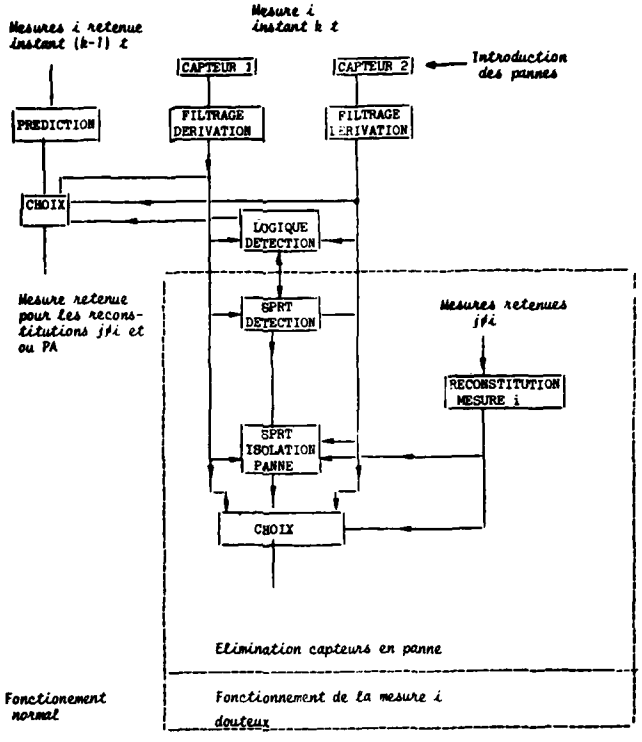


FIGURE 6 : Procédure de détection

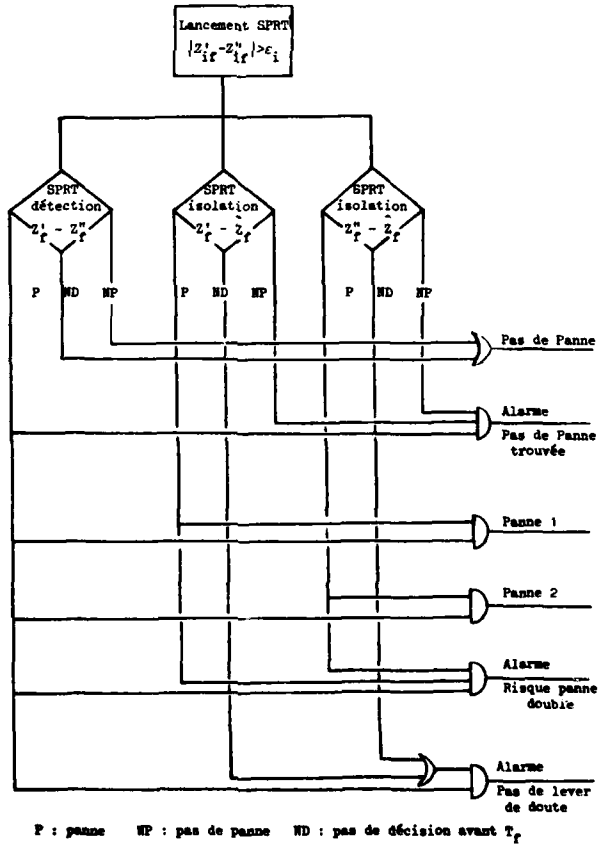


FIGURE 7 : Logique de détection

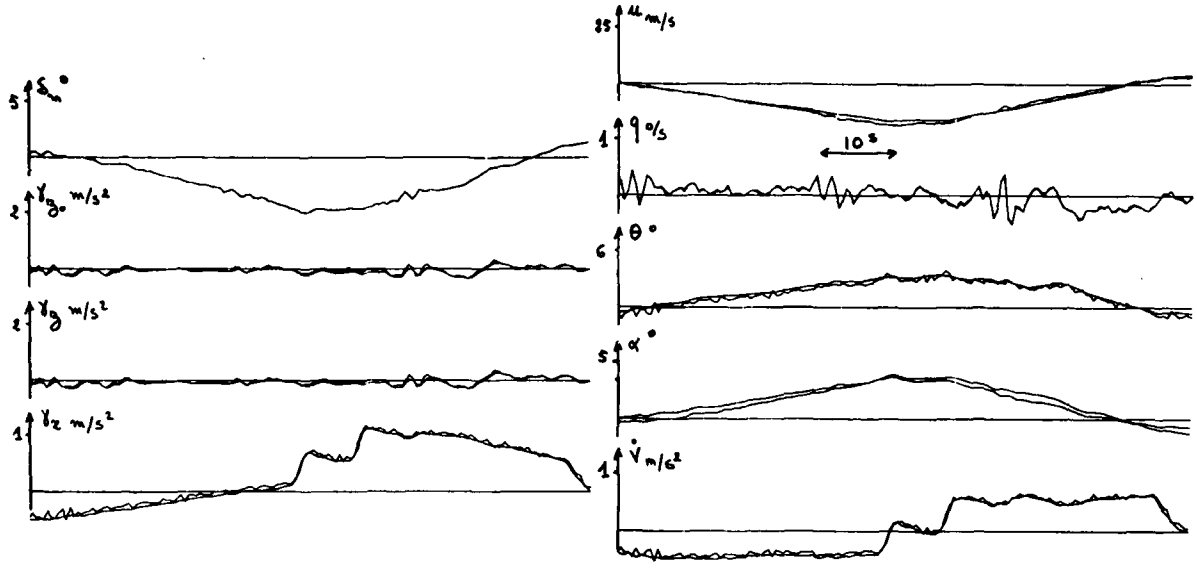
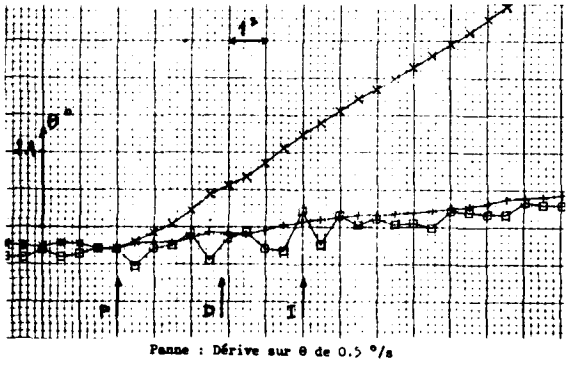
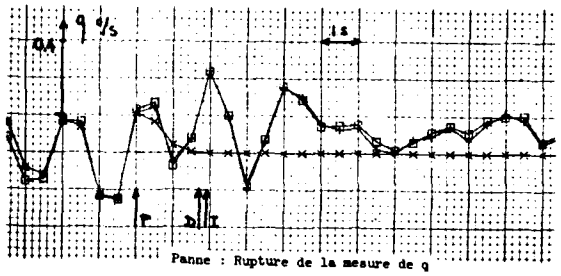


FIGURE 8 : Comparaison estimation-enregistrements de vol



Panne : Dérive sur θ de 0.5 °/s



Panne : Rupture de la mesure de q

P : Introduction de la panne
D : Panne détectée
I : Panne isolée

\+ capteur 1
x capteur 2
● estimation

FIGURES 9-10 : Tests de la procédure sur des enregistrements de vol.

SOFTWARE VALIDATION AND VERIFICATION TECHNIQUES[1]

Karl N. Levitt, Associate Director

Computer Science Laboratory
SRI International
333 Ravenswood Avenue
Menlo Park, CA 94025

ABSTRACT

Computers are being increasingly used as a critical component in nuclear reactors, aircraft, and other applications, where a failure can be life-threatening. Recent studies have revealed that a centralized computer controlling a commercial aircraft should have a mean time to failure of at least 10,000 years. By the judicious application of hardware redundancy, such a reliability can be approached -- but assuming perfect software. We believe that the current approach to software validation -- extensive testing with respect to informal specifications -- cannot be relied on to achieve the needed reliability. The newly emerging technique of program verification gives promise of leading to vastly more reliable programs. Program verification is, at least conceptually, a very simple idea: By mathematical reasoning a program, for all input values, is shown to yield output values defined by an independently supplied formal specification. This paper summarizes the state-of-the-art in program verification, addressing the following issues: available techniques and on-line tools, the cost of verifying a system, possible sources of unreliability in an alleged proof, and outstanding technical problems. A brief description is given of the SIFT (Software Implemented Fault Tolerance) computer, currently being developed by SRI and Bendix, and our effort to prove the correctness of its operating system.

1. INTRODUCTION

Computers are being increasingly used in situations where a failure could be life-threatening, e.g., in controlling a nuclear reactor, aircraft, or missile launching system. Computers are, of course, used in many situations where a failure can cause inconvenience and financial loss. For example, the media has widely reported on the recent failures in numerous air traffic control systems and the impact in terms of delayed flights.

What reliability is required for a computer, for example, having total responsibility for flight critical computations of a commercial aircraft? It has been reported [17] that a rate of 10^{-10} failures/hour over a 10 hour flight (equivalent to a MTBF of 10,000 years) is needed for the computer. With this reliability, the probability of the computer not generating a correct result is significantly less than the pilot suffering a fatal heart attack during the flight or the airframe suffering physical damage that would impede its ability to remain airborne. No single computer can achieve this reliability in the presence of hardware failures (i.e., spontaneous failure of the computer logic), but with suitably applied redundancy it is achievable. For example, the SIFT computer (Software Implemented Fault Tolerance) [23] is a multicomputer configuration in which failures in one (or more) constituent computers are masked by voting on the the results computed by tasks running redundantly on 3 (or more) computers. Subsequent to its being identified as faulty, an offending computer is logically reconfigured out of the system. The voting, identification of faulty computers, and reconfiguration are all accomplished by software -- hence the name SIFT. Peripherals, a common cause of computer reliability can be made more reliable in an aircraft environment by the technique of analytic redundancy [12]. Briefly, the technique calls for the replication of sensors and actuators in a way that likely failures are masked. With sufficient redundancy it appears that a computer like SIFT can maintain operation in the presence of hardware failures.

Can the required reliability be achieved for the software? Obviously one cannot measure with any assurance a MTBF of 10,000 years in a program. Thus what we are seeking is a technique for assuring that programs are completely free of errors.

To better understand the source of software errors, it is convenient to view software as the product of the work of three actors.

- a **buyer** who poses **requirements** on the system,
- a **designer** who creates an organization for the system, often as a set of individually **specified** units,
- an **implementor** who produces executable code (**implementations**) for the units.

---

[1]Some of the research reported in this paper was sponsored by the National Aeronautics and Space Administration, Langley Research Center under Contract NAS1-15528. All of the results presented here have appeared in the open literature; the author assumes total responsibility for the conclusions based on these results.

In most current system efforts, the requirements and design are described using informal (i.e., imprecise) media, typically a combination of prose, pictures, diagrams, etc. Testing of a system is usually accomplished in two stages: (1) individual units are tested with respect to their specifications, and (2) the integrated system is tested with respect to the system requirements. There are two major deficiencies of this approach.

- Given that only a small fraction of the possible inputs to a program can be applied as tests, the testing of a program cannot give assurance that the program is perfect.

- Since the specifications and requirements (against which the system is being tested) are informal, it is difficult to determine if the output for a given test input is as desired. In general, it is this informality that leads to poor communication among the actors and, ultimately and usually, to the buyer not getting the system he thought he wanted.

Although, as briefly discussed in Chapter 2, there is significant research in progress on developing new testing techniques, testing seems to be inherently inadequate for achieving guaranteed reliability.

The main message of this paper is that by the technique of formal verification, it will be possible to obtain reliable systems. Verification is based on the observation that the properties of a system are subject to mathematical proof. Verification, as initially proposed by Floyd [10], is defined as proving that a program satisfies its specification for all values of input variables. In our three-actor model of software production, this form of the designer's specifications; we call this process **implementation verification**. A separate process, which we call **design verification**, is used to demonstrate that the designer's specifications satisfy the buyer's requirements. An additional advance that we describe is a technique for decomposing a large implementation into units that can be independently verified. Our technique is based on an approach called HDM (Hierarchical Development Methodology) [19, 20, 14], but there are other, closely related techniques.

An important aspect of the research work in verification has been the production of programs, commonly called **verification environments**, to assist in the verification process. Mechanization is absolutely essential in the light of the detail associated with the verification of even small programs; this detail would preclude any real confidence in a system proved manually. We, and several other research institutes, are developing such programs to: analyze the specifications and system code, produce mathematical conjectures to prove, and then to attempt to prove these conjectures using symbolic manipulation techniques. The programs (called **mechanical theorem provers**) that carry out the proofs of conjectures are still rather primitive in their ability to do complex reasoning -- certainly as compared with a mathematician. However, despite this and other limitations discussed in the paper, the existing verification environments can be used to verify a large class of useful systems.

The skeptical reader is undoubtedly asking himself many questions about the feasibility of verification, among which are likely to be the following.

- Is it indeed possible to produce requirements and specifications for real complex systems?

- What programming languages can be accommodated by current and proposed verification environments?

- How reliable is the verification process itself?

- When, if ever, will verification be routinely applied to real systems?

- What special training will be required of practioners of verification?

The paper will attempt to provide answers, or at least best guesses, to these questions.

Section 2 briefly reviews some recent (and promising) work in program testing. Section 3 gives an overview of program verification, in the process giving desiderata for a formal specification of a program. Our approach to specification and verification of systems is discussed in Section 4, followed by a description of the current SIFT project in Section 5. A description of the programs that constitute a verification environment appears in Section 6. The paper concludes with a discussion on the prospects for verification -- essentially given as answers to the above questions.

2. NEW RESULTS IN PROGRAM TESTING

A user in testing a program provides input data, runs the program with this data, and determines if the output values are as he expected. Obviously, the behavior of most programs cannot be observed for all combinations of input data. For example, consider a program that multiplies two 32 bit numbers. The number of possible combinations of input data to the multiplier is 2^{72}. Assuming that each combination can be tested in 1 microsecond, the total testing time would be in excess of 100,000 years. In lieu of exhaustive testing, the practice is to select a few instances of input data for each "part" of the program. This strategy indeed is sensible, since each part usually does correspond to an equivalence class of input values. For example, the multiplier program is likely to have parts that handle overflow and underflow.

To assist a user in identifying such program parts and in distinguishing input data that is likely to reveal errors in the program, several experimental testing environments have been developed. They roughly fall into two classes: (1) based on the notion of path testing, and (2) based on program mutants. Below we briefly describe the basic techniques, present the experimental results of the use of the environments, and give our conclusions on the effectiveness of the techniques.

2.1. Path Testing

Path testing is based on the observation that each path (or, better, collections of paths) corresponds to a significant part of the program. Thus, confidence in the program can be claimed by testing each path in the program. Obviously, all paths in a program containing loops cannot be tested. The approach followed, then, is to be selective; this usually reduces to ensuring the following: (1) each statement is

executed at least once, (2) each outcome of each branch is executed at least once, and (3) each loop is executed at least once. Run-time systems have been developed that determine if a set of user-supplied input tests for a program exercise the program to satisfy the above three criteria. More advanced systems [3], [13] automatically analyze a program. In particular, they identify a sufficient set of paths that satisfy the above three criteria and then attempt to generate input data that would cause each of these paths to be executed. The attempt to generate such input data can fail for two reasons: (a) the generation of such data would involve proving a theorem that is beyond the capability of the system (see Section 6 for a discussion of the limitations of current theorem proving programs), or (b) there is no data that would cause execution of the path. The system, in discovering and reporting instances of (b) can alert the user to possible problems with his program. However, it has been found that the test data automatically generated by these systems, corresponding to the limited number of paths considered, is usually inadequate for revealing errors. The systems usually generate degenerate data, e.g., an array of equal elements for a program that sorts an array.

These systems can be used in a mode, called **symbolic evaluation** that appears to be more effective in alerting the user to possible errors. Here, the system treats ~~input~~ variables as ~~symbolic~~ entities -- rather than assigning concrete values to the variables. Corresponding to each considered path, the system generates a **predicate** (i.e., a boolean-valued expression) that evaluates to true for all input values that would cause execution of the path in question. (These predicates are generated in a manner almost identical to that described in the Section 3 for verification conditions; hence we omit detailed discussion of the process in this section.) Typically, these predicates are simplified by the system before giving them to the user. Howden [21], as a result of extensive experimentation with an interactive symbolic evaluator, has reported that 10-20 percent of the errors in a program were naturally detected by observation of the predicates. Significantly, most of these errors were missed by the programmer in manually selecting test data; none of these errors were detected by automatically tested data.

2.2. Program Mutants

The underlying assumption driving this work is that an incorrect program is likely to be "almost" correct. For example, a loop is programmed to terminate when a indexing variable reaches N instead of N+1. A system developed by Budd, et al. [5] generates for a given program P a class of programs that are close to P; such programs are called **mutants**. For example, a mutant could be generated by replacing the terminating variable of a Fortran DO loop by the starting condition. In testing P, the user executes both P and each of the mutants with a set of test data T. By analysis of the results, the user attempts to determine the "adequacy" of the test data, i.e., can he be confident that P is correct. If all of the mutants yield output values different from P, then it is likely that T is adequate. Otherwise, the user casts his attention to the mutants that yield the same result as P on T. This system, when applied to the same class of programs tested by symbolic evaluation with Howden's system, revealed a superset of the errors identified by Howden's system.

2.3. Assessment

We believe that both the symbolic evaluation and *mutant systems represent an advance to the current* testing practice. They seem to be headed towards to mechanizing the aspects of testing that are routine, but leaving critical analyses to the user. Given that the systems have only been applied to relatively simple programs, it is difficult to predict their performance on large systems. For example, it is likely that the user, being forced to examine too many paths (in the symbolic evaluation method) or too many *mutants (in the mutant method)* will simply not correctly analyze the results of tests. The detail given to the user at any given time could be reduced by the technique of **abstraction**, i.e., replacing blocks of code with specifications that describe the blocks' behavior.

However, we do not believe that either system can be relied on to achieve the reliability required by, for example, the SIFT operating system. In the absence of precise requirements for a statement against which the system's behavior is compared, it will be impossible for the user to correctly determine if the result of a test is what the buyer really wanted.

3. PROGRAM VERIFICATION

In this section we briefly describe the method that is commonly used to prove that a program satisfies a specification. In the following section, we extend this notion in several directions, primarily to prove that a system satisfies a requirement. What we present here is, by necessity, barely an introduction, but should at least give the reader some intuition and a pointer to the ever increasing literature on program verification.

It is safe to assume that all of the readers are familiar with the notion of mathematical proof. For example, we have all proved many of the theorems of plane geometry. The correctness of these theorems have been established using rational logical arguments.

How are similar mathematical techniques applied to programs? Consider a simple sequence of instructions in some programming language. Suppose you are given a precise mathematical description of some input state. The description, denoted as the **input assertion** and written in some precisely described language called a **specification language**, usually constrains the input variables to values meaningful to the program. Further, suppose you are asked to show that some statement -- called the **output assertion** -- holds at the output of the program. The process is to step through the computation, at each step deriving a symbolic description of the current state from the previous step and the effects of each instruction. When this process reaches the output of the program, the derived symbolic description is shown to imply the given output assertion. This demonstration is achieved using the usual laws of logic, algebra, etc.

What if the program contains loops? The process described above would seem to entail endless repetition over the statements of the loop. In Floyd's method [10], this problem is alleviated by the placement of programmer-supplied assertions (called **intermediate assertions** at points in the program such that each loop is cut by at least one assertion. The symbolic analysis process then is applied to **paths** in the program, where a path is a sequence of program statements surrounded by assertions. That is, the program

is proved by proving each of its paths. (To be precise, if each path is proved, then the program will conform with its specification only when it halts. A separate proof process is used to ensure that the program halts for all inputs satisfying the input assertion.)

An interesting property of Floyd's method is that the verification of the program cannot be imperiled by an incorrect intermediate assertion, Unless all of the intermediate assertions are adequate for proving the program, the process will simply indicate that the program is not proved, although it is not immediately clear whether the problem is with the code, the input assertion, the output assertion, or the intermediate assertions.

It should be clear that a "proved" program will behave according to its input and output assertions, i.e., the program's specifications. Of course, the specifications might not capture what the buyer expected of the program. There is no resolution to this dilemma, except for the specifications to be so clear that they can be determined adequate by inspection. The clarity of specifications (and requirements) is a constant theme of the remainder of this paper.

To illustrate the technique let us consider the following simple program, FINDMAX, written in a very simple programming language, that finds the maximum element in an array.

```
   FINDMAX

START:
       I := 0
       MAX := A[0]
       LOC := 0
L:
       I := I + 1
       IF N < 1 GOTO HALT
       IF MAX >= A[I] GOTO L
       LOC := I
       MAX := A[I]
       GOTO L
HALT:
```

By inspection, it should be clear that MAX holds the current value of the maximum, and LOC holds the location of the current maximum. Each time around the loop MAX and LOC are changed if the current value of MAX is less than the new array element, A[I], being read.

What are appropriate input (and output) assertions for this program, to be viewed as describing the states at the START (and HALT) labels. An appropriate input assertion is 0 <= N, expressing the obvious property that arrays of negative length are not to be considered. An appropriate output assertion is the following

```
A[0] <= MAX ... A[N] <= MAX
       AND
MAX = A[LOC]
       AND
0 <= LOC <= N
```

In the first conjunct we have made use of the ellipsis notation assuming, for the moment, it is present in our specification language. The first conjunct is obviously needed. The necessity of the second conjunct is almost as obvious, especially if the programmer wants to use the final values of MAX and LOC in some other program that calls FINDMAX. However, the third conjunct could easily be omitted, possibly leading to an unintended specification if the array A has values outside of the range 0:N. It cannot be overemphasized that careful review of specifications is absolutely necessary.

By placing an intermediate assertion at the label L, it can be easily determined that there are no loops in the program that are not cut by this assertion. As mentioned above, it is the user's job to provide the intermediate assertion. The commonly used heuristic is to identify some boolean-valued expression that captures the values of the program variables every time control reaches the point in question -- in this case label L. After a bit of analysis, you might come up with the following assertion.

```
MAX = A[LOC]
   AND
A[0] <= MAX, A[1] <= MAX ... A[I] <= MAX
   AND
0 <= LOC <= I <= N
```

Let us consider a particular path to prove, namely the path from START to L. After processing of the first three assignments, it should be clear that the program variables have the following values:

```
I = 0                                   (1)
MAX = A[0]                              (2)
LOC = 0                                 (3)
```

From the input assertion, given that N has not been modified, it is also known that

```
0 <= N                                  (4)
```

From these facts it is possible to prove that the intermediate assertion is true. For example, looking at the third conjunct of the intermediate assertion, it should be clear by simple algebra and logic that

```
(I = 0 AND LOC = 0 AND 0 <= N)
    IMPLIES
(0 <= LOC <= I <= N)
```

is a theorem.

4. VERIFYING REAL SYSTEMS

Although particular clever programs can cause problems, it is safe to say that most of the technical problems attendant to proving small sequential programs (i.e., algorithms) are solved. This is certainly not the case for systems. What distinguishes a system from a simple program of the kind illustrated in Section 3?

- It is very difficult to give precise specifications for a system.

- Systems are large.

- Systems usually consist of simultaneously executing programs.

- Systems are implemented with real programming languages, sometimes with several languages.

- A system is subject to many modifications in its lifetime.

Each of these problems as they relate to verification can be (and, to a degree, have been) addressed in isolation of the others. However, recent work has attempted to seek a general method that embraces all of the significant problems associated with large system development and verification. The "Holy Grail" for systems -- now called a development/verification **methodology** -- is yet to be discovered, but there is clear progress. Our discussion here will be concerned with HDM (Hierarchical Development Methodology) being studied at SRI, but it should be clear that there is other work in progress with similar goals.

Briefly, HDM is an approach for decomposing a large system into modules that are independently specified, implemented and verified. The particularly unique feature of HDM is the language SPECIAL, used for specifying modules. Below we indicate how HDM addresses the problems attendant to large systems.

4.1. Describing the Behavior of a System

This is probably the most challenging problem associated with the verification of real systems, simply because most systems are implemented with a only a fuzzy view of what they are supposed to do. For the perceivable future, this will remain a primary hindrance to verification, although new techniques and experience should certainly help.

First let us focus attention on the specification of a system; later we will shift attention to a system's requirements. A system **specification** is a formal (i.e., precise) statement of the system's behavior under all possible circumstances. Any questions about the behavior should be answerable from the specification. HDM, in particular SPECIAL, provides the constructs for writing a specification. At this time, it should be pointed out that the programs that constitute a system are also a specification. What distinguishes a SPECIAL specification from a specification derived from the implementation is conciseness and readability. Using SPECIAL it is possible to produce a specification that only contains necessary detail; extraneous facts of the system are **abstracted** out. We will not present the syntax of SPECIAL here; it is too cumbersome for this paper and is described in an available report. Rather we will illustrate the technique of specification with reference to a very simple system.

Consider the problem of keeping word counts in a simple database. The user of the system is to be provided with the ability to

1. Query the count for a word

2. Insert a "new" word. If the word was previously inserted, its count is incremented by one; if not, its count is set to one

3. Delete the occurrence of a word.

The fundamental problem in specifying any system is deciding on how to view the system's data. In using SPECIAL, one is encouraged to view the system as containing abstract data structures, abstract in the sense that they apply only to the specification -- not to the implementation.

For our simple database example, one could view the database as two arrays: (1) an array that holds the words of the data base in sorted order, and (2) an array that holds the counts of the words. We wish to indicate that this is a poor representation from the viewpoint of specification. For example the specification of "Insert" must speak to (a) searching array (1) for occurences of a word, (b) finding the corresponding count in array (2), and (c) if the "inserted" word does not appear in array (1), inserting it so as to maintain the array in sorted order. As the reader has undoubtedly guessed, this representation is too close to being the way data could be handled by the implementation.

A better way of viewing data for the specification is in terms of an "infinite" array, the domain of which is all words in the universe. Of course, this representation is only for purposes of specification and clearly unsuitable for implementation. A possible state of the array for a subset of the words in the universe is indicated below.

counts	0	3	0	12	325	176	...
words	a	b	c	aa	bar	foo	...

The SPECIAL specification of the "Insert" function is then as follows.

```
OVFUN insert(word w)
  EFFECTS
    'word_store(w) =
       word_store(w) + 1;
```

Here "word_store" is a function that can be viewed as the array. The appearance of 'word_store(w) indicates the value of "word_store" after the insert operation concludes. We have simplified the specification by omitting any mention of the finite size of any real data base; however, the reader should be assured that this issue is handled in SPECIAL.

At this time is convenient to distinguish between a "requirement" and a "specification". Recall that a specification provides complete information about the functional behavior of a system, i.e., the externally visible effect of any sequence of calls to the system. A requirement, on the other hand, is only intended to address some of the questions about a system. For example, we have worked on the problem of expressing what it means for an system to maintain classified documents. The particular requirement statement we proposed [8] constrains information to flow only from a document at classification L to one at level L or higher. Note that this requirement says very little about the functional behavior of the system; it does not even require the system to do anything! Currently, SPECIAL cannot adequately express requirements that involve information flow; work is in progress on this problem.

4.2. Verifying Large Systems

As indicated above, HDM addresses this problem by providing constructs for hierarchically decomposing a system into modules, each of which is independently proved. If module M1 is situated above M2 in the hierarchy, then the implementation of M1 is proved with respect to M1's specifications, using the specifications of M2. The challenge here is to create a decomposition for a system. As is the case for system specification, the availability of techniques (like HDM) can provide only limited assistance. Significant creativity will always be required to come up with a clean decomposition. However, it is possible that as experience in decomposing large system accumulates, libraries of decompositions will become available. Even if a given available decomposition does not exactly fit a new system design, it could serve as a model. For example, we have developed [9] a decomposition for an operating system that is intended to maintain data security; however, closely related decompositions could well apply to operating systems with different goals.

4.3. Proving Programs in Contemporary Programming languages

The toy language used in the example of Section 3 employs a few primitive instruction types: assignment, test, and transfer. All contemporary high-level languages contain these simple instructions (although some languages eschew the "goto"), but many more. The additional instructions serve the following purposes:

- Provide higher level constructs so programs can be less verbose
- Provide redundancy (e.g., through type declarations) to permit the detection of many obvious, but troublesome, errors by the compiler
- Provide the means for composing a large program out of smaller ones.

In order to prove programs written in high level language,s it is necessary to define the effect of each instruction on the state of program variables. In effect, we proposed such a definition for the assignment statement of our toy language in describing the symbolic evaluation process. Fortunately, such definitions have been developed for several popular languages, e.g., Pascal [11] and ANSI FORTRAN [2], and are required for all new languages, e.g., Ada. It is not implied here that it is simple to produce the definition for all language constructs. Particular constructs that are difficult to formally define are concerned with parallel processing and aliasing -- an object being referred to by more than one name; research is still in progress in this area.

In summary, the verification of systems is still a research area. It is likely that each new system verification effort will suggest new research problems.

5. SIFT -- AN EXAMPLE OF A SYSTEM BEING VERIFIED

The following is a very brief sketch of SIFT and the current effort to verify it. Additional details can be found in [23], although the description in the cited paper is somewhat obsolete; it has been necessary to modify the design as the verification effort has revealed errors.

As briefly described in Section 1, SIFT can be viewed as a multicomputer, i.e., an interconnection of identical units, each consisting of a processor and a full complement of memory. It is the goal of SIFT to provide reliable computation of tasks, a task typically corresponding to some aircraft function. A task is executed according to a regular schedule, each execution denoted as an iteration. Each task is assigned to 3 or more computers, the actual degree of replication possibly depending on the criticality of the task. The execution of tasks on different computers need not all be in locked-step, although there is the requirement that all replicas of a task commence execution within some window, say 50 microseconds. When a task T concludes the execution of an iteration, it's computer C broadcasts the output data of the task to all other processors. This data is stored in a special buffer area corresponding to C in each of the computers. Another task needing data from T will read data from each of the buffer areas. By voting on the contents of these buffer areas, errors (single errors in the case of three-fold replication) can be masked. In addition, any discrepancies in the input to a voter are noted for future processing. An executive program, itself a task, processes the discrepencies to identify possibly failed computers. If an error occurs for only a few iterations, it is viewed as transient and no action is taken. In the other

hand, if a computer persists in causing errors, it is viewed as permanently failed. The executive program reports such permanently failed computers to all computers, causing the failed computer to be ignored.

Let us now consider a requirement statement for SIFT. This statement has two aspects: (1) the reliability of the system in the presence of permanent and transient faults, and (2) the scheduling of tasks in a timely manner. Item (1) is handled by a **reliability model** and (2) by an **input/output model**. The reliability model simply states that the system will generate correct results if the number of good computers processing each task exceeds the number of computers that have failed, but whose failure has not been noted and handled by the executive. The input/output model can be viewed as maintaining a set of tables that indicate the following for each task T: (1) the tasks supplying inputs to T, (2) the output values that T computes during each execution window, (3) the window in which T is to perform iteration i, for all i. The major property expressed is that if SIFT has enough good computers, all tasks will read data from the right input tasks and produce the expected output values, all in the specified window. Obviously, this a very abstract statement of the behavior of SIFT, but appears to capture the essential aspects of reliability and scheduling. We have produced formal mathematical statements of these properties in a form close to SPECIAL.

We have also produced a hierarchical decomposition for SIFT, SPECIAL specifications for each module, and an implementation for each module in Pascal [22]. Work is in progress on proving the specifications with respect to the requirements (design proof) and on proving the Pascal implementation with respect to the module specifications (implementation proof). In order to simplify the design proof we have found it desirable to introduce additional models between the topmost pair (reliability and input/output) and the specifications of the executive.

As a by-product of the verification effort to date, we have noted a fundamental error in the original design of SIFT. The error related to synchronizating computers and reading data from a sensor whose value is not staticized -- see [18]. Briefly, in synchronizing the computers by the exchange of clock values, three computers are not adequate for ensuring that the synchronization is maintained in the presence of all single faults. Four computers are required.

6. MECHANIZING THE VERIFICATION OF SYSTEMS

There has been extensive work on developing programs that will automatically, or with user assistance, verify a program. Given the work on development methodologies, there now exist some tools for verifying systems.

Why is mechanization needed? Considering the detail associated with proving any nontrivial system, it would be impossible for a manually generated proof to be reliable. In addition, it is expected that the elapsed time required for proving a system can be significantly reduced by mechanization.

A "traditional" verification system consists of basically two components: a **verification condition generator** and a **theorem prover**. The verification condition generator performs the symbolic evaluation associated with processing paths in the program, producing a set of conjectures to be proved. The theorem prover attempts to show the conjectures are theorems, carrying out the required logic and algebraic manipulations. Based on the existing formal definitions, verification condition generators have been developed for the defined subsets of a few high-level languages. The languages included herein are Fortran, Pascal and Jovial [7]. Work is in progress on a verification condition generator for Pascal as the implementation language of HDM.

How effective are mechanical theorem provers at proving the conjectures associated with program verification? Although most of the typical conjectures can be handled automatically (for example all of those for the simple program of Section 3), many are beyond the abilities of current theorem provers. A particular theorem prover currently under development at SRI [4] seems to be as powerful as any existing theorem prover. Its particularly important and, in some cases, unique features are the following.

- a small core of assumed definitions, primitive types, and axioms
- the ability for the use to add new types and definitions in a manner that ensures the consistency of the theory. In contrast, some systems allow a user to insert new axioms that are inconsistent with the current axiom set, thus permitting nontheorems to be incorrectly judged as theorems
- a reasonably fast simplifier for propositional logic and linear inequalities
- the ability for the user to suggest lemmas that might achieve a proof. The theorem prover will not use such lemmas until it judges them to be theorems.

Given that is impossible to exhibit a computer program that will judge whether an arbitrary conjecture is a theorem, it is clear that the theorem prover is doomed to fail on many conjectures. However, it appears that skilled users of the Boyer-Moore theorem prover can use it to prove almost any verification condition associated with a realistic program. Work is in progress on extending the theorem prover's capability to prove more theorems automatically.

It is appropriate to briefly mention other issues in the development of a powerful environment for verifying systems. One major drawback to verifying a system is that the verification has to be redone subsequent to every modification. Given the current cost of verifying a system and the large number of modifications expected for a system in its lifetime, it is clear that the verification should only be redone for those verification conditions affected by the modification. Moriconi [16] has produced a system that can reason about the impact of modifications to a program or its specifications. Work is in progress on applying the Moriconi technique to the HDM/Pascal verification system.

Another recent advance is a program that will automatically produce a verification condition generator for a high-level language. This program, called a meta-verification condition generator [15], accepts as input the syntax and formal definition of a high-level language.

7. DISCUSSION

It is well-established that current testing techniques are inadequate for revealing all errors in a system. However, verification does appear to be an attractive alternative. Just as mathematicians for thousands of years have attempted to prove important conjectures, we believe that important systems will be verified.

There are numerous highly-respected researchers who are skeptical of the feasibility of verification. The most vocal of these are DeMillo, Lipton and Perlis [6], who claim that an alleged proof cannot be accepted until it goes through what they call the "social process", i.e., careful scrutiny by many mathematicians. Given that no mathematician is likely to scrutinize all of the alleged proofs for any large system, these "proofs" will remain suspect. We do not claim that an incorrect "proof" will never be generated for a system. There are many sources of unreliability in the proof process, including

1. specifications or requirements for the system that do not capture the intent of the buyer

2. errors in the verification environment

However, we believe that the recent work on developing specification languages and applying them to real systems indicates that it is indeed possible to produce specifications that can be carefully scrutinized. Verification environments are, of course, very large programs and subject to error. However, recent work by Boyer and Moore [1] is aimed at producing a theorem prover that has a primitive core extendible by new proof procedures that are subject to verification. We believe that the "social process" need not be applied to the entire lengthy proof of a system, but only to its requirements statement and the primitive core of the verification system.

Despite the recent progress that leads us to claim that certain important systems can be verified today, much fundamental work remains to be done. We have already mentioned the need to produce theorem provers that need less human guidance. Other areas where further work is essential relate to

- Creating formally definable specification languages that can lead to more readable specifications

- Understanding the problem attendant to specifying performance properties of systems

- Proving properties of numerical programs

- Gaining more experience in specifying and proving real systems.

REFERENCES

[1]
R.Boyer and J Strother Moore.
Metafunctions: Proving them Correct and Using Them Efficiently as New Proof Procedures.
Technical Report CSL-108, Computer Science Laboratory, SRI International, December, 1979.

[2]
R. Boyer and J Strother Moore.
A Verification Condition Generator for Fortran.
Technical Report CSL-103, Computer Science Laboratory, SRI International, June, 1980.

[3]
R.S. Boyer, B. Elspas, and K.N. Levitt.
SELECT--A formal system for testing and debugging programs by symbolic execution.
In Proc. Int. Conf. Reliable Software, pages 234-244. IEEE, April, 1975.

[4]
Robert Boyer and J Strother Moore.
A Computational Logic.
Academic Press, 1979.

[5]
T.A. Budd, R. J. Lipton, and F. G. Sayward.
The design of a prototype mutation system for program testing.
In Conference Proceedings. AFIPS, 1978.
Proceedings of National Computer Conference.

[6]
R.A. DeMillo, R.J. Lipton, and A.J. Perlis.
Social Processes and Proofs of Theorems and Programs.
Communications of the ACM 22(5):271-280, May, 1979.

[7]
B. Elspas.
Rugged Jovial Environment.
Interim Report 2 for Rome Air Development Center, Contract F30602-78-C-0031, SRI International, Computer Science Laboratory, January, 1980.

[8]
R. Feiertag, K. Levitt, and L. Robinson.
Proving Multilevel Security of a System Design.
In Proceedings of Sixth ACM Symposium on Operating Systems Principles, pages 57-65. ACM, November, 1977.

[9]
R. Feiertag and P.G. Neumann.
The foundations of a Provably Secure Operating System (PSOS).
In Conference Proceedings of the National Computer Conference. National Computer Conference, 1979. pp. 329-334.

[10]
R.W. Floyd.
Assigning meaning to programs.
Mathematical Aspects to Computer Science 19:19-32, 1968.

[11]
C.A.R. Hoare and N. Wirth.
An Axiomatic Definition of the Programming Language PASCAL.
ACTA Informatica 2:335-355, 1973.

[12]
A.L. Hopkins,Jr., T.B.Smith,III, and J.H. Lala.
FTMP--A Highly Reliable Fault-Tolerant Multiprocessor for Aircraft.
In Proceedings of the IEEE, pages 1221-1239. The Institute of Electrical and Electronics Engineers, Inc., October, 1978.

[13]
J.C. King.
A New Approach to Program Testing.
In International Conference on Reliable Software, pages 228-233. IEEE, April, 1975.

[14]
K. Levitt, L. Robinson, and B. Silverberg.
HDM Handbook,Volume III: A Detailed Example in the Use of HDM.
CSL Report 95 for Project 4828, SRI International, June, 1979.

[15]
M. Moriconi, L. Flon and R. Schwartz.
Automatic Construction of Verification Condition Generator.
1980.
Technical report from Computer Science Laboratory of SRI International.

[16]
M.S. Moriconi.
A designer/verifier's assistant.
To be published in IEEE Transactions on Software Engineering, Vol. SE-5, No. 4, pp. 387-401, July 1979.

[17]
N.D. Murray, A.L. Hopkins, and J.H. Wensley.
Highly Reliable Multiprocessors.
In P.R. Kurzhals, editor, Integrity in Electronic Flight Control Systems, pages 17.1-17.16. Advisory Group for Aerospace Research and Development, 1977.
AGARDograph No. 224.

[18]
M.Pease, R. Shostak, and L. Lamport.
Reaching Agreement in the Presence of Faults.
JACM 27(2):228-234, April, 1980.

[19]
L.Robinson.
HDM Handbook, Volume 1: The Foundations of HDM.
CSL Report 93 for Project 4828, SRI International, June, 1979.

[20]
B.Silverberg, L.Robinson, and K. Levitt.
HDM Handbook, Volume II: The Languages and Tools of HDM.
CSL Report 94 for Project 4828, SRI International, June, 1979.

[21]
W. E. Howden.
DISSECT--A Symbolic Evaluation and Program Testing System.
IEEE Transactions on Software Engineering , January, 1978.

[22]
C. Weinstock.
SIFT:System Design and Implementation.
October,
1980.
The Conference will be held in Japan in October.

[23]
J.H. Wensley, L. Lamport, J. Goldberg, M. Green, K.N. Levitt, P.M. Melliar-Smith, R.E. Shostak, and C.B. Weinstock.
SIFT: Design and Analysis of a Fault-tolerant Computer for Aircraft Control.
Proceedings of the IEEE 66(10), October, 1978.
pp.1240-1255.

ACKNOWLEDGMENT

Most of the ideas presented in this paper originated with colleagues of the author at SRI. SRI's verification environment and its application to several systems are the product of the following teams of individuals: theorem proving (Robert Boyer and J Moore), the HDM methodology for system development and verification (Larry Robinson, Brad Silverberg, Peter Neumann, and David Elliott), verification environment development (Dwight Hare, Mark Moriconi and Mike Green), and the development and verification of SIFT (Jack Goldberg, Leslie Lamport, Michael Melliar-Smith, Richard Schwartz and Chuck Weinstock). The author also acknowledges the support of several U.S. Government agencies; in particular, Nick Murray of NASA-Langley Research Center has been a long term supporter of fault-tolerant computing and very early recognized the importance of verification in the context of building reliable flight control systems.

FAILURE MANAGEMENT TECHNIQUES FOR HIGH SURVIVABILITY

by
Thomas B. Cunningham
Honeywell Systems and Research Center
Minneapolis, Minnesota
UNITED STATES OF AMERICA

SUMMARY

Survivability of aircraft can be greatly enhanced by employing a number of considerations and techniques in design and placement of avionics components. Summarizing these:

1. The initial sizing and location of surfaces should include the impact of survivability.

   - Due to the increased number of surfaces for primary performance, highly vulnerable control axes can usually be provided with suitable reversion modes by reconfiguring existing surfaces.
   - On board digital computers easily accommodate a number of reversion mode control systems.
   - Modifications to existing surfaces to provide adequate reversion mode capability should be minor (if needed at all). These benefits however should be explored in preliminary design.

2. Avionics hardware sharing offers cost reductions and can provide high performance if reliability and survivability issues are successfully addressed.

   - Dispersion of sensors for survivability can cause reduction in reliability fault detection coverage and possible flight control performance. Both problems can be addressed with modern estimation observer technology.
   - Navigation performance suffers with shared dispersed sensors. A minimum navigation complement, i.e., strapdown gyros (3), and accelerometers (3) is required in a common rigid location.

3. Observers offer a structure for seeking solutions to survivability problems.

   - Normalization of dispersed sensors for use in flight control systems
   - Construction of blended error signals for analytical fault detection.
   - Replacement signals for sensors lost due to component random failure.

4. Observers for "in-the-loop" sensor reconstruction often require stability margin enhancement. Techniques for examining this problem and improving stability now exist.

The above considerations are discussed in detail in sections 1. and 2.. Finally, in section 3., these techniques are combined with some trends in sensor and computer technology to formulate a candidate for a flutter mode control implementation.

1. PRELIMINARY DESIGN CONSIDERATIONS FOR HIGH SURVIVABILITY

In the preliminary design of aircraft the goals of the mission are foremost in the designers priorities. The static performance requirements, such as, range, speed, weapon carrying capability, ceiling, along with dynamic requirements such as, maneuverability, ride quantity, and handling quantities combined to dictate the aircraft configuration.

Aircraft survivability to small weapons becomes a matter of critical hardware location and protection. Such considerations are typically handled after the fact.

Survivability through digital computer management is enhanced by incorporating some extra considerations in the baseline system. Two areas for discussion are:

- Surfaces and Actuators
- Sensors

1.1 Surfaces and Actuators

New aircraft designs are evolving with more and dispersed surfaces; figure 1 for example. These surfaces are proliferating due to their ability to enhance the primary mission and flight goals. The following is a partial list of such surfaces.

- Elevators
- Stabilators
- Flaps
- Speed or Dive Brakes
- Rudder(s)

- Ailerons
- Flaperons
- Horizontal Canards
- Vertical Canards
- Close Coupled Canards
- Thrust Vectoring Surfaces
- Glove Vanes
- Inlet Doors
- Swing Wing

These surfaces are used in an ever-increasing list of flight modes for normal operation and combat:

- Direct force modes (vertical and side)
- Constant attitude, direct translation modes
- Ride quality
- Terrain following/avoidance
- Path following
- Numerous hold modes
- Precision maneuver coordination modes

Beyond an inherent survivability improvement credited to improved maneuverability and low level flight capabilities, these surfaces offer the opportunity to reconfigure after a surface or partial FCS loss to enemy fire. Section 2.2 contains some design procedures for this, however, some attention to this problem during preliminary design can greatly enhance the ultimate survivability.

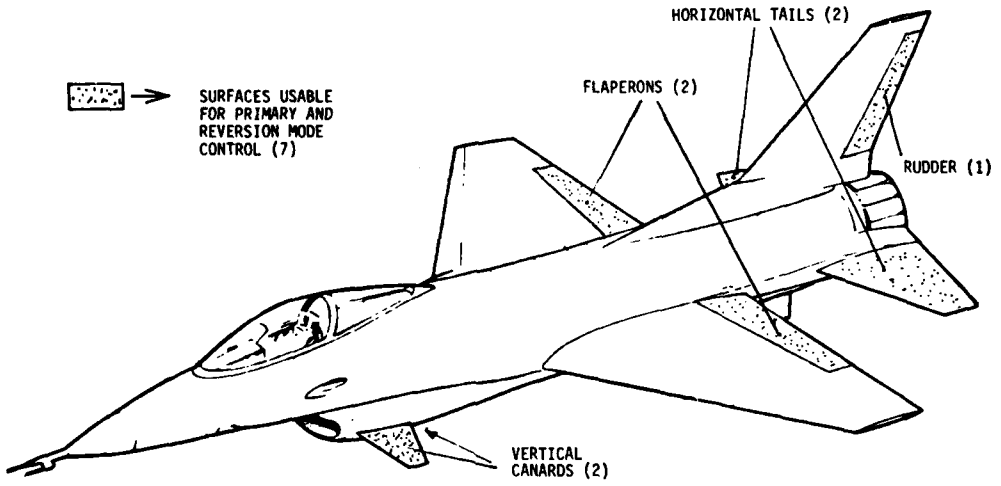


Figure 1. YF-16 Control Surfaces

Survivable FCS Hierarchy -- FCS survivable modes of flight can be classified according to the level of flight retained after reconfiguration of undamaged components.

1. Mission Continuation -- Although a loss of operational capability is implicit with the loss of a surface, many damage modes still leave enough control authority in remaining surfaces (and other force and moment generators) to continue some portion of the mission. Loss of a vertical canard, for example, while limiting the ability to employ side modes, probably allows many ground attack and air-to-air missions to still be conducted.

2. Mission Abort, Fly Away, and Land -- The ability to fly to a safe landing after combat damage requires sufficient surface deflection to break off the mission, trim to level flight, and sufficient control rate to maneuver during landing approach.

3. Mission Abort and Combat Area Egress -- Some combat damage modes do not allow continued flight for more than a short period of time. Loss of propulsion is the prime example. A glide away capability is provided by surface retrim and some small extra deflection and rate capability to maneuver.

Single Hit Survival -- Perhaps the easiest guideline to use for design of flight control systems is to install sufficient dispersed redundancy, reversion mode capability and component failure isolation to withstand a single hit of small arms hostile fire (up to 37 mm high explosive incendiary for example). Along with the flight control changes discussed herein this dictates some other requirements upon the system.

- Surfaces that are inoperative must not go hard over in the "free" position. Unstable hinge moments cause surfaces to go to stop positions when released, say due to actuator damage. FCS reversion modes using remaining surfaces are not likely to have enough authority to recover in the face of such a mistrim. Unstable hinge moments, while attractive to actuator designers, are not survivable, and should be eliminated from the design flight envelop.

- Hydraulic systems must be made survivable via check valves which isolate a damaged hydraulic line from removing the use of the numerous surfaces serviced by the line.

Surface Size and Actuator Modifications-- Considerations discussed above yield some basic design requirements that make survivable reversion mode control feasible. Some preliminary analysis as to what reversion capabilities exist to meet failed systems will perhaps dictate some changes in the baseline surface sizes and actuation concepts.

Grumman Aircraft Corporation studied a reversion mode control design for a hypothetical statically unstable derivative of the F-14 aircraft [2]. Table 1 lists some alternatives for providing control using the available surfaces shown in figure 2.

TABLE 1. GRUMMAN STUDY VEHICLE* ALTERNATE CONTROL SURFACES

CONTROL FUNCTION	PITCH CONTROL	ROLL CONTROL	YAW CONTROL	TRIM
Primary	Collective Tail	Differential Tail Plus Spoilers	Dual Rudders	Horizontal Tail, Glove Vane
Alternate Surface Configuration	Wing Flaps Plus Spoilers Body Flaps Speed Brakes Half of horizontal Tail Plus Spoilers Combinations of Above	Differential Tail Spoilers Rudder in Combinations with Above	Single Rudder Differential Wing Flaps Plus Spoilers Combinations of Above	Alternatives include Wing Sweep, Flaps and Glove Vane Control

* Statically Unstable Derivative of the F-14

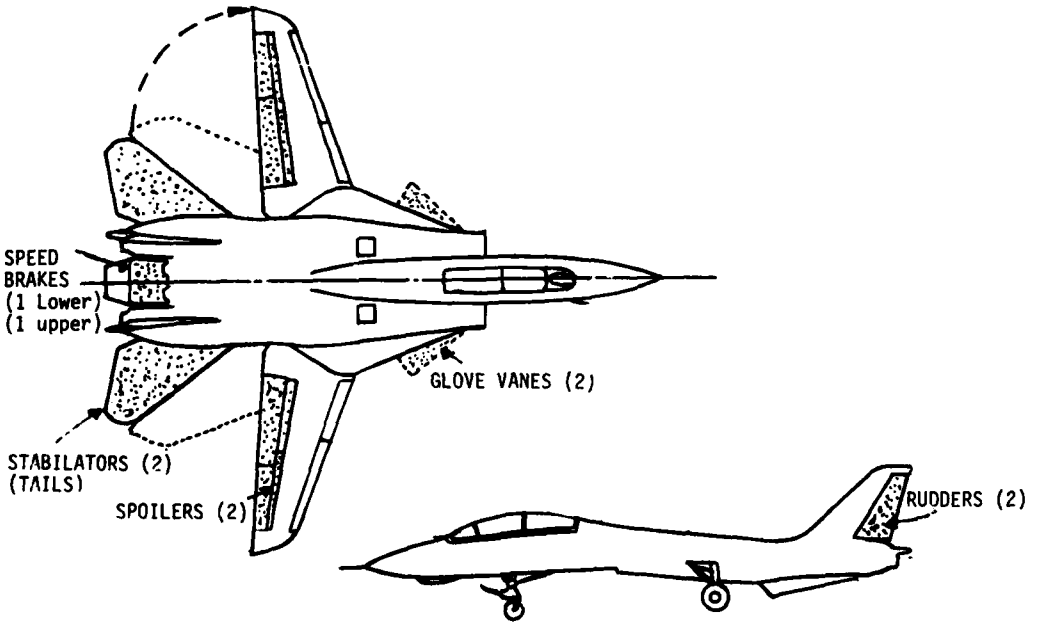


Figure 2. Grumman Study Vehicle and Control Surfaces

The loss of one horizontal tail (of two) provides an interesting design study. The maximum pitch authority solution for this failure mode is to use the remaining horizontal tail. This, however, creates a corresponding rolling moment which must be rebalanced. The spoilers are the only other roll surfaces available and these proved insufficient.

A solution results from redesigning the speed brakes for pitch control. Specifically, the lower speed brake must be doubled in size. Some other details of surface out reversion mode design are discussed in Section 2.2.

1.2 Sensor Location

Sensors which measure the dynamic state of an aircraft are used for numerous tasks related to normal and combat operations. In addition to all the flight control functions, other subsystems such as weapon delivery, navigation, and attitude reference systems need similar information with varying degrees of accuracy and reliability. Each function dictates desired locations. Table 2 exemplifies some of these desires.

TABLE 2. FLIGHT FUNCTION SENSOR NEEDS

Function	Sensors Needed	Requirements	Location Requirements
1. Flight Control System			
a. SAS, CAS or RSS	p, q, r, n_z, n_y, α, and β	Low accuracy, moderate bandwidth and high reliability	Placed to minimize bending, ahead of c.g.
b. GLA, FMC	Accelerations	Low accuracy high bandwidth and high reliability	Numerous sensors placed on body and wings to pick up structural deflections
c. Autopilot	M, H, α	Low accuracy and bandwidth	N/A
2. Inertial Navigation	p, q, r, n_x, n_y, n_z (strapdown complement)	High accuracy	Colocated, rigidly mounted
3. Attitude Reference	p, q, r (strapdown) θ, ϕ, ψ (platform)	Moderate accuracy and high reliability	Colocated
4. Weapon Delivery	$\ddot{x}$, $\ddot{y}$, $\ddot{z}$, $\dot{\phi}$, $\dot{\theta}$, $\dot{\psi}$	Moderate accuracy	Colocated

Emerging concepts involve sensor sharing based upon strapdown accelerations, and body rates to derive all the measurements needed for SAS and CAS flight control functions, inertial navigation, attitude reference, and weapon delivery.

Four variations for survivable placements encompass most of the location issues for shared sensors. Figure 3 displays these variations for a hypothetical fail- operational (based on reliability considerations) sensor system. Although other configurations exist (skewed accelerometers for example) these four are sufficient to exercise the applicable software technology available to mechanize numerous options.

Configuration 1: Orthogonal Colocated Accelerometers and Angular Rate Sensors -- This configuration contains sufficient hardware to meet accuracy performance goals for navigation, weapon delivery, and attitude reference .

Flight control performance requirements are dictated less by sensor accuracy than by relative stability assurance for a given closed loop design. Ideal sensor locations for a rigid body augmentation system are exemplified in figure 4. Colocation does pose some conflict as unwanted bending mode respones might be included.

A search for a compromise location can be performed based on maximizing stability margins. Results of such an analysis [6] indicate that some stability enhancement and structural mode isolation will improve the situation. A lead-lag compensation with notching improves the stability margins, however, a state reconstruction filter which can perform the same function, is discussed in section 2.0.

Reliability of this configuration is very good as full fail-operational failure logic can be provided with comparison monitoring. High coverage is guaranteed by colocation of sensors.

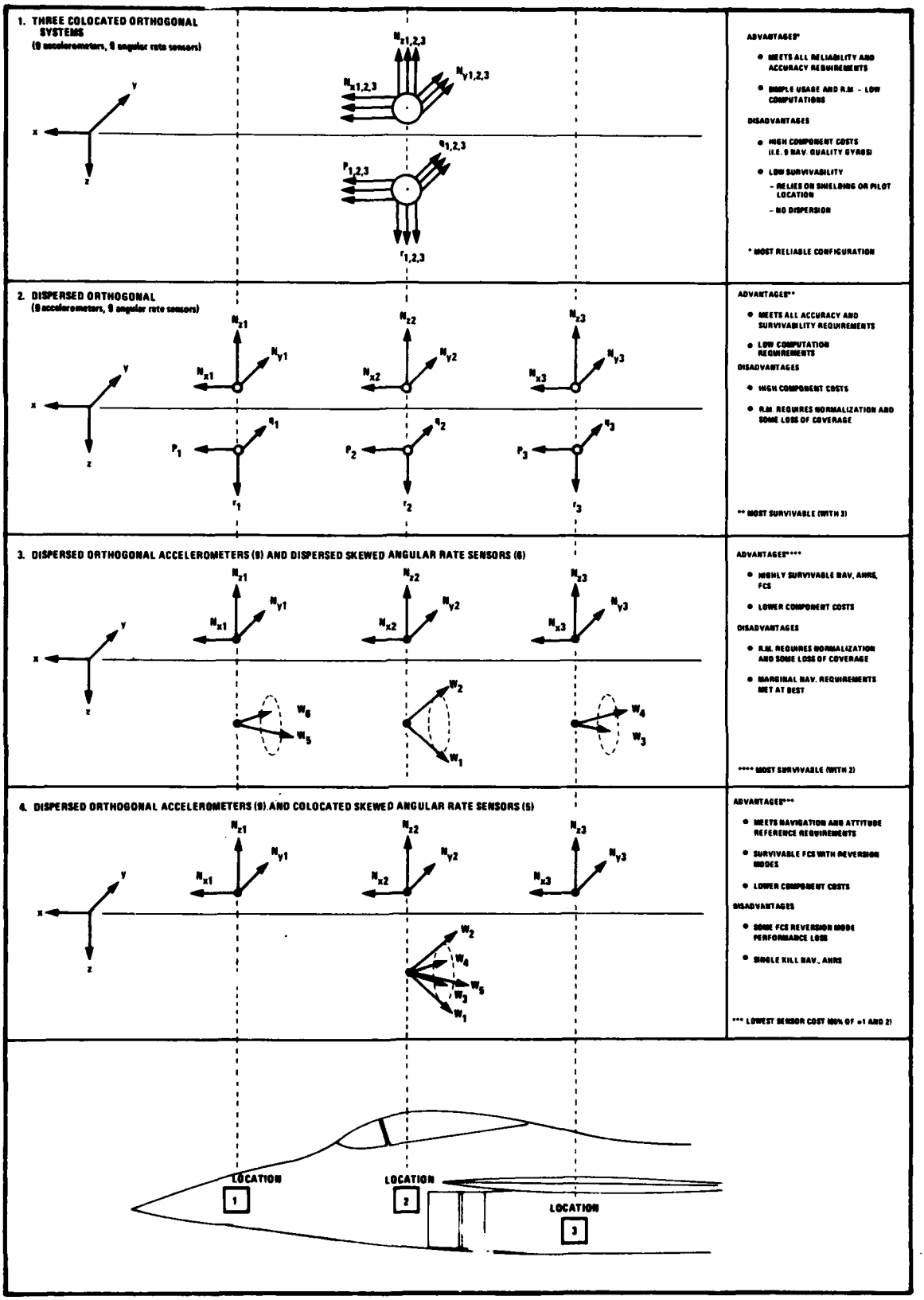


Figure 3. Shared Avionics Configurations

Survivability is low for a single hit. One can counter this by placing all single kill items near the pilot station. This solution has a couple of flaws, however.

1. Such a solution is obviously unacceptable for multiseat aircraft.
2. The loss of a single kill item located near the pilot station not only correlates highly with the loss of the pilot but virtually guarantees it.

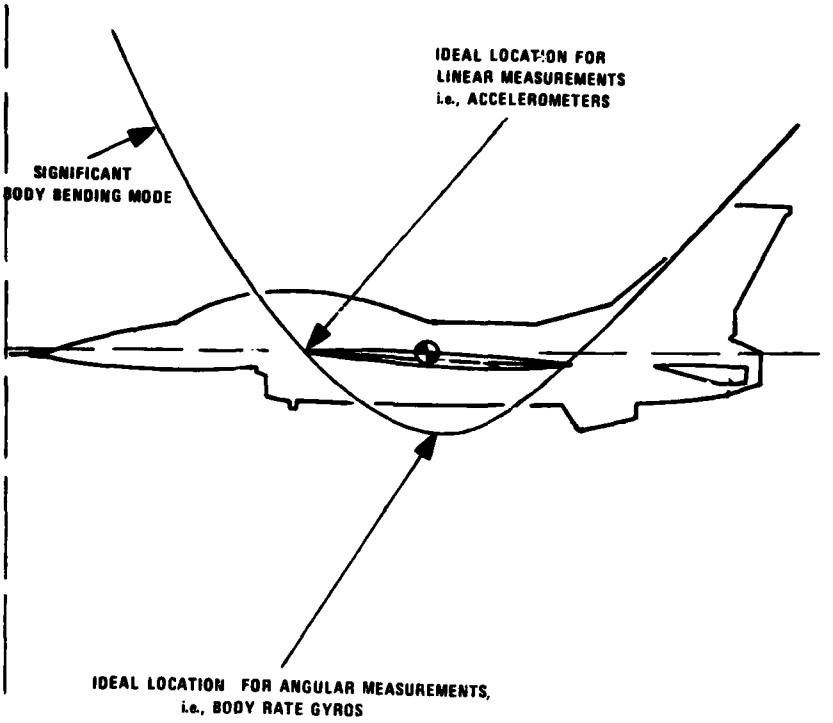


Figure 4. Ideal Sensor Locations For Flight Control

Hardware costs of this configuration are high because the high cost of 9 accelerometers and 9 body rate sensors, all of inertial quality. Using a figure of $6,000 for an inertial grade gyro and $1,500 for an inertial grade accelerometer (both prices are conservatively low) a sensor price tag of 67.5 K$ for this complement results.

Software costs are low due to the use of standard failure management techniques. Some extra flight control software to enhance stabilty would probably be required.

Configuration 2: Dispersed Orthogonal Sensors -- Figure 3 shows configuration 2 containing three dispersed orthogonal accelerometer triads and three dispersed orthogonal body rate sensor triads. Sensor costs are identical with configuration 1. The sole goal of the dispersion is improved survivability, which is greatly enhanced.

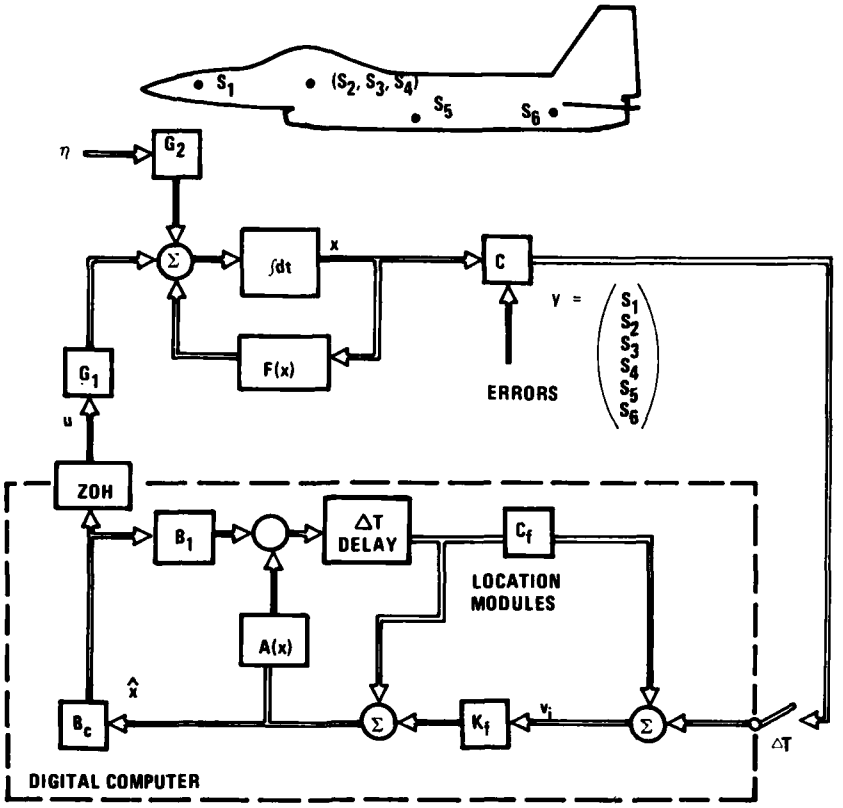
The major drawback to dispersed sensors is performance. Flight control and redundancy management can be attacked with a state reconstruct filter (figure 5). Design techniques for such a filter are discussed in section 2.1.2, however, the features used for the present application are:

- Implementation of a state feedback control law; allowing the use of advanced design techniques such as optimal control, etc. [7].
- Filter structure includes models for sensor location and structural modes (optional). This provides the framework of properly normalizing the sensor data for flight control.
- Redundancy management is conducted on reconstructed error signals. This is equivalent to model compensated comparison monitoring. Performance is not as good as colocated sensors and thorough analysis of failures must be conducted. Grumman has successfully evaluated such a design [2].
- Failure reconfiguration is optimized with the reconstruction filter by changing the filter gain matrix after a sensor failure.

In summary, configuration 2 sacrifices some failure detection coverage for survivability. This assumes appropriate flight control performance for all three locations can be achieved. Also, the overall software costs have risen due to the reconstruction filter.

Configuration 3: Dispersion with Skewed Body Rate Sensors -- Skewing can reduce sensor cost by eliminating some redundancy. The attempt here is to reduce costs over configuration 2 and retain the same basic performance overall comparisons are as follows:

- Reliability -- 6 skewed gyros provide potential 2 fail-operational redundancy for three axis rate measurements. Dispersion of sensors reduces the coverage of parity equations, however.



PLANT CHARACTERISTIC	DIGITAL COMPUTER FUNCTIONS
• PLANT DYNAMICS $\dot{x} = F(x)\,x + G_1\,u + G_2\,\eta$ • MEASUREMENTS (DISPURSED) $y = Cx$	• STATE OBSERVER UPDATE $\bar{x}_{i+1} = A(\hat{x}_i)\,\hat{x}_i + B_1\,u_i$ • ERROR VECTOR $v_i = y_i - C_f\,\hat{x}_i$ • STATE ESTIMATE $\hat{x}_i = \bar{x}_c + K_f v_i$ • FAILURE MANAGEMENT –BASED ON ERRORS, v_i –SERVO SENSORS • CONTROL LAW $u_i = G_2\,\hat{x}_i$

Figure 5. Dispersed Sensor Normalization

- Survivability -- This concept has a high degree of survivability. Loss of one colocated gyro pair to combat fire leaves 4 rate sensors which can be fault analyzed with parity equations.

- Performance

  - FCS -- normalization of sensors with a state reconstruction filter (figure 5) potentially meets flight control goals.

  - Navigation -- tight navigation requirements of modern strapdown systems (< 1 nmph drift) can only be met with current state-of-the-art sensors if they are rigidly mounted and in very close proximity. Minimum performance with dispersion, therefore, would require dual skewed triads.

Configuration 4: Dispersed Orthogonal Accelerometers and Colocated Skewed Rate Sensors -- This last configuration is the lowest cost design relying on 9 accelerometers and 5 angular rate sensors to perform the shared functions. In addition to the cost benefits this design combines features of the most advanced software concepts with other proven techniques.

- Reliability -- The use of parity equations on the pentad cluster provides high coverage fail operational performance. Primary FCS signals typically include the three body rate signals Pitch rate, q, is most crucial for feedback control of statically unstable vehicles. As demonstrated by General Dynamics [21] a normal acceleration feedback can be used to replace a failed pitch rate signal when the redundancy level drops below the safe level. This type of reconfiguration (discussed in more detail in section 2.1.2) provides extra redundancy levels for safe flight.

  Comparison monitors on accelerometers have to deal with the dispersion issue, however, the case of a normalizing filter (figure 5 again) aides in this process.

- Survivability -- FCS survivability relies on using accelerometers at locations 1 and 3 to stabilize the system should the hardware at location 2 become lost to enemy fire. The pitch rate signal has been discussed. Section 2.1.2 demonstrates the observer design theory applicable to this problem.

  Table 3 summarizes a failure scenario for reversion and reconfiguration.

- Performance:

  - FCS -- Some performance will be sacrificed if a reversion to accelerometer feedback is performed, however, this possibility is remote for normal failures and should be quite adequate for safe flight after combat damage.

  - Navigation -- Because 3 accelerometers are colocated with the gyro package, the primary strapdown navigation is performed with colocated sensors. Reversion to another dispersed set of 3 accelerometers in conjunction with the rate sensor package should result in minimal performance degradation. Despite the dispersion of accelerometers from attitude reference this situation is easier to correct than full dispersion.

TABLE 3. CONFIGURATION 4 FAILURE MANAGEMENT*

FAILURE (cause)	DETECTION AND ISOLATION SCHEME	REVERSION MODES		REMARKS
		NAV & AHRS	FCS	
Accelerometer (random)	Comparison and voting	Use location 2 cluster if available, switch to another if not	N/A	1. NAV accuracy is reduced if cluster 2 is lost 2. Normalization of sensors necessary, some loss of fault detection coverage
Rate sensor (random)	Skewed parity equations	Continue on remaining sensors	Continue on remaining sensors	Consider Accelerometer for FCS reversion mode
Accelerometer cluster (combat loss)	Comparison and voting	Use location 2 cluster if available	Note for further failure action	Same as 1. above
Rate sensor cluster (combat loss)	Analytical Red. monitors with accelerometers	Loss of functions	Switch to rate observers with accelerometers	See section 2.1.2.

* First failures considered. Second random failures are safe.

2. SURVIVABLE RECONFIGURATION MANAGEMENT

The reconfiguration of a given complement of computers, sensors, and surfaces will maximize the survivability of the aircraft. Whether one chooses to add, disperse, and-or reconfigure depends upon the survivability goals. The techniques discussed here are the same, with the main theme of utilizing computer software for analytical techniques and management rather than increase hardware in sensors and surfaces. In the case of sensors the reduction is mostly one of costs. The constraints are harder with surface failures as it is unlikely that redundant surfaces will be added to an aircraft solely to improve survivability.

2.1. Sensor Reconfiguration With Observers

The most common failure management technique for sensor reliability is colocated redundancy and failure detection by comparison. When more than two sensors are present the mid value selection will provide a valid signal and fault isolation can be performed by voting [3]. Two system goals make these techniques undesirable by themselves.

- Sensor cost reduction by eliminating redundant hardware
- Survivability enhancement through location separation

Both of these goals can be attacked with software observers. Two types of observers can be examined.

1. Full state Kalman filters
2. Reduced order observers, i.e., Luenberger

Both observers offer the capability of meeting the system goals stated above. Kalman filtering, however, provides a design which is more suitably tailored to the fault detection problem because the error signals produced match the goals of monitor logic.

- o Minimum residual (error signal) RMS
- o Uncorrelated residual - important for hypothesis testing monitors

2.1.1 Observer Design For Fault Detection And Isolation

The use of Kalman observers for fault detection and isolation involves tailoring the filter design to the various monitors designs available.

Filter Design

The discrete time dynamics are modeled by

$$x_{i+1} = Ax_i + B_1u_i + B_2\omega_i$$

$$y_i = Cx_i + D_1u_i + D_2\omega_i$$

where

$$E\left[\omega_i\omega_j^T\right] = Q\delta_{ij}$$

x is the system state, n-dimensional, u is the control input, m-diemnsional, and y is the measurement, p-dimensional.

The Kalman observer structure can be written

$$\bar{x}_{i+1} = A_F\hat{x}_i + B_{1F}u_i$$

$$\hat{x}_i = \bar{x}_i + K_F(y_i - C_F\bar{x}_i - D_{1F}u_i)$$

where

$\bar{x}_i$, $\hat{x}_i$ are the estimates of x before and after update, respectively.
A_F , B_{1F} , C_F , and D_{1F} are filter representations of A, B_1, C, and D_1 respectively.

The degree with which a Kalman filter design varies in fault detection performance versus parameter variations is analyzed as shown in figure 6. The basic evaluation indices are

$R_T \triangleq$ Residual (error signal) Covariance
$\Phi_T \triangleq$ Residual Autocorrelation Matrix

If the parameters of the filter are identical to the plant at all deisgn conditions

$R_T = R^* =$ Residual Covariance for the Optimal filter design
$\Phi_T = 0$ (p by p null matrix)

For assessing the impact of various gain schedules on filter designs a useful combination of R_T and R^* is the following

Residual Index

$$RI = \text{trace}(R^{*-1}R_T)/(\text{No. of measurements})$$

This index is useful because

$$\lim_{R_T \to R^*}(RI) = 1$$

Plant	Filter
$x_{i+1} = Ax_i + B_1u_i + B_2w_i$ (order n_x)	$\hat{x}_{i+1} = A_F\hat{x}_i + B_{1F}u_i$
$y_i = Cx_i + D_1u_i + D_2w_i$ (order n_r)	$\hat{y}_i = \hat{x}_i + K(y_i - C_F\hat{x}_i - D_{1F}u_i)$
$E[w_iw_j^T] = Q\delta_{ij}$	

Let

$$z_i^T = (x_i^T, e_i^T) \text{ (order } 2n_x\text{) where } e_i = \hat{x}_i - x_i$$

$$z_{i+1} = A_Tz_i + B_{T1}u_i + B_{T2}w_i$$

$$\Delta A = A - A_F$$

$$\Delta B_1 = B_1 - B_{1F}$$

$$\Delta C = C - C_F$$

$$\Delta D_1 = D_1 - D_{1F}$$

and

$$y_i = C_Tz_i + D_{1T}u_i + D_{2T}w_i$$

Then

$$A_T = \begin{bmatrix} A & 0 \\ A_FK\Delta C - \Delta A & A_F(I - KC_F) \end{bmatrix}$$

$$B_{1T} = \begin{bmatrix} B_1 \\ A_{KF}K\Delta D_1 - \Delta B_1 \end{bmatrix} \quad ; \quad B_{2T} = \begin{bmatrix} B_2 \\ A_FKD_2 - B_2 \end{bmatrix}$$

$$C_T = \begin{bmatrix} \Delta C & , & -C_F \end{bmatrix} \quad ; \quad D_{1T} = \Delta D_1 \quad ; \quad D_{2T} = D_2$$

Therefore if

$$Z = E[z_iz_i^T]$$

then

$$R_T = E[v_iv_i^T] = C_TZC_T^T + D_{2T}QD_{2T}^T$$

$$E[v_{i+1}v_i^T] = C_TA_TAC_T^T + C_TB_{2T}QD_{2T}^T$$

and $\theta_T = R_T^{-1}E[v_{i+1}v_i^T]$

Figure 6. Estimation Filter Performance With Known Parameter Variations

An example of typical results using the residual index is shown in figure 7. These results were obtained for an analytical redundancy filter designed using lateral directional dynamic equations for an A-7 aircraft.

Reduced order filters are also applicable to the failure detection and isolation problem, however, these do not match up with the designed monitor characteristics of minimal error signal RMS or low error signal correlation.

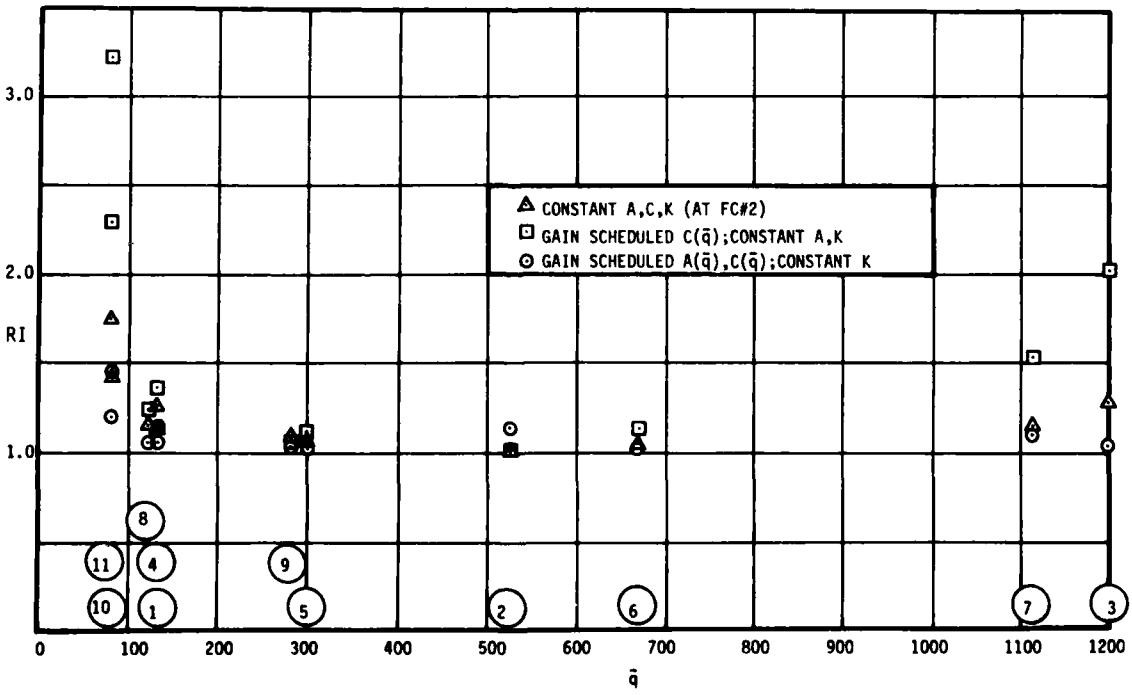


Figure 7. A-7 Reconstruction Filter Performance

Monitor Design

Fault detection monitors are designed to maximize the detection probability with minimum false alarms. Two types have evolved:

1. Multiple trip error signal boundary
2. Hypothesis tests

The first type is illustrated in figure 8. It operates by declaring a fault when the boundary is exceeded a number of times consecutively. It is commnly used with comparison monitors and is described in reference 3. It has two basic characteristics.

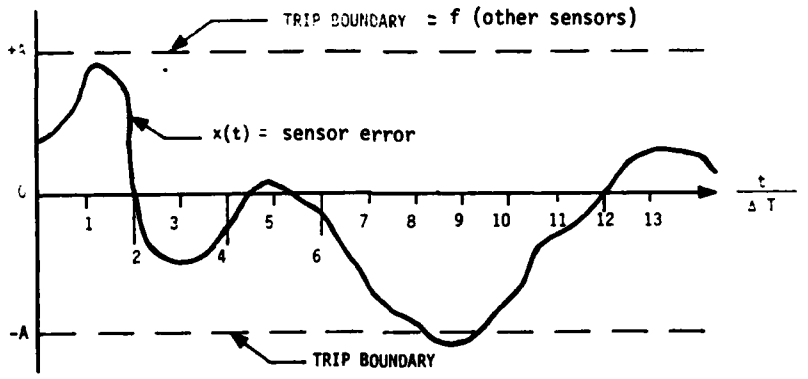


Figure 8. Multiple Trip Boundary Monitor

1. It is tolerant to single frame error anomalies, most commonly caused by computer hickups.
2. It is insensitive to correlated error signals. This is both good and bad; good because it is less sensitive to modeling errors and bad because it is less sensitive to some soft failures, e.g., bias growth.

The hypothesis test monitors are described in greater detail elsewhere [8], [9]. One variation is the following:

Given a ratio of probability density functions

$$\Lambda_n = \frac{f_0(x_1, x_2, \ldots, x_n)}{f_1(x_1, x_2, \ldots, x_n)}$$

where

x_i $(i = 1,2,\ldots,n)$ is a sequence of random variables

f_0 is a hypothesized (H_0) density function of the x_i's

f_1 is another hypothesized (H_1) density function for the x_i's

A sequential likelihood Ratio Test (SLRT) is

Accept H_0 if $\Lambda_n \leq a$

Accept H_1 if $\Lambda_n \geq b$

No decision if $a < \Lambda_n < b$

The assumptions are used for application in analytical redundancy:

- The sequence of residuals; $\nu_i : i = 1, 2, \ldots, n$ are independent and guassian
- The density function f_k hypothesizes a mean value μ_k i.e.,

$$f_k = \text{const} * \exp\left[-\sum_{i=1}^{n}(\nu_i - \mu_k)^2/(2\sigma_k^2)\right]$$

for all cases.

CASE 1

- no fault (H_0); $\sigma_0 = \sigma$ and $\mu_0 = 0$

  (σ is the unfaulted residual RMS)

- fault has occurred (H_1); $\sigma_1 = \sigma$ and $\mu_1 = m\sigma$

CASE II

- no fault (H_0); $\sigma_0 = \sigma$ and $\mu_0 = 0$
- fault has occurred (H_1'); $\sigma_1' = \sigma$ $\mu_1' = m\sigma$

The combination monitor is shown in figure 9. In application the residual sum is started and proceeds until H_0 is declared. The sum is then restarted.

This variation results in low software costs. More elaborate SLRT's (or SPRT's) are described in reference 10 with recent flight test conducted at NASA Dryden [11]. Sequential likelihood ratio tests are sensitive to correlated error signals. This is good for detecting soft failures, a dead sensor for example. On the other hand, more precise modeling of sensor anomalies such as biases, scale factors, bandwidth, etc., is demanded to avoid false alarms.

2.1.2 Observer Design For Feedback Control

In the last section an observer design for optimum fault detection was discussed. Many occasions arise in failure management when it is desired to use an observer for a missing sensor.

- o Loss of last remaining sensor type either through random failure or combat damage.
- o Observers in the loop in lieu of analytical redundnacy avoids some coverage difficulties.
- o Observers as a part of the baseline control system can enhance performance of dispersed sensors (as discussed earlier).

Robust Observers For Feedback Control

The idea behind any observer is to construct designated dynamic states from a limited set of output measurement. The traditional use of an observer is for artificially creating an output which is used by the feedback control system but not measured. The increasing popularity with Linear-Quadratic- Gaussian (LQG) optimal control theory, for example, produces a feedback control law which typically requires many more outputs (or system states) than are measured. An observer supplies the state estimates needed. The failure management feedback observer situation is similar. Here the use of the feedback observer can take two basic forms

1. An observer is used to replace a missing measurement. The feedback control law is not altered unless the reconfigured system cannot meet basic performance requirements. This type of observer is illustrated in figure 10.
2. When an observer is used in the primary flight control system, (an increasing trend). The loss of a sensor is handled by reconfiguring the observer measurement input to reconstruct with remaining measurements. This is illustrated in figure 11.

Observer design theory provides for maintaining closed loop roots for a system designed without an observer through the separation theorem [12]. This theorem does not provide for retaining stability margins however [13]. The following example illustrates the use of a Kalman observer in-the-loop. The design incorporates a method of improving the stability margins of the reconfigured system.

A-7 Lateral-Directional Axes - Sensor Output Reconstruction [14]

The lateral-directional axes of the A-7 rigid body dynamics and gust environment are modeled in continuous time as shown in figure 12. The model differs from the more conventional presentation in that the roll angle, , is treated as an input rather than a dynamic state. This eliminates the spiral root which, because of its low frequency, exhibits stability problems in filter gain scheduling over the flight envelop.

Sensor models used in filter design and simulation are shown in Table 4. High frequency rate gyro and accelerometer noises intensities were calculated based on inclusion of all sensor outputs above 2 hertz frequency. This would include structural vibrations and engine noise corruption as well as internal sensor noise. This approach better represents the total noise environment for rigid body filter designs.

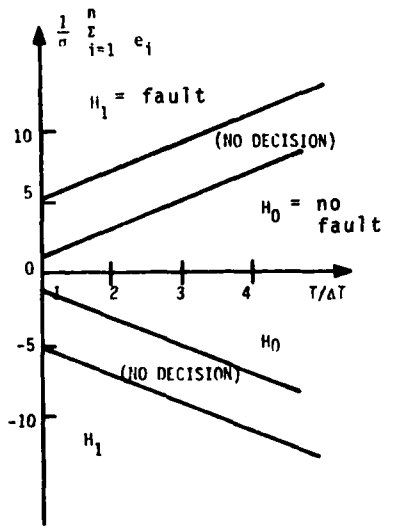


Figure 9. SLRT Hypothsis Test Monitor

TABLE 4. A-7 NOMINAL SENSOR CHARACTERISTICS

Basic Sensor	Noise Level (1σ)	Nulls (1σ)	Sensor Limit	Misalignment (deg)	Cross Axis Variable	Scale Factor Error (% Full Scale)
P (o/s)	1.1	.5	± 200	.43	Neg	5.5
R (o/s)	.4	.08	± 80	.43	P	5.5
n_y (g's)	.03	.0025	± 5	.16	n_z	5.4
ϕ (deg)	.75	1.6	± 180	--	--	--
V (fps)	2.5	7.9	Upper 1000 Lower 197	--	--	--
H (ft)	25	25	Upper 50,000 Lower -1,000	--	--	25

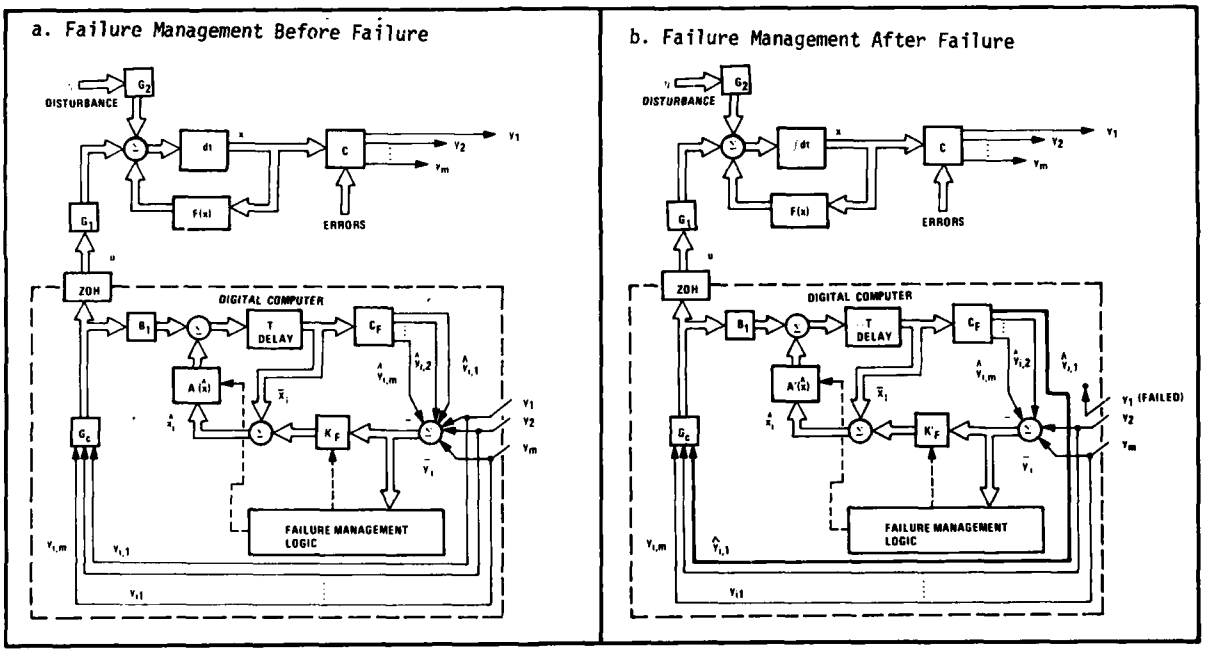


Figure 10. Reconstructed Sensor Replacing Failed Component

The basic A-7 CAS is shown in figure 13. The reconstruction design involves providing an observer for a lost yaw rate gyro, R , from remaining measurements of lateral acceleration, n , roll rate, P , and roll attitude, . The reconstructed yaw rate, R, is then used in the flight control system of figure 13 in place of R . This replacement is represented generically in figure 10.

The Kalman filter was designed and implemented in discrete form. Notation changes from those of figure 12 are as follows:

Continuous (F, G, H, E) → Discrete (A, B, C, D)

The original filter was designed over 11 flight conditions for fault detection through residual monitoring [9]. The use of the filter in the flight control loop not only proved to have poor stability margins - it was unstable as shown in figure 14.

Obviously, a change in the filter bandwidth would solve the problem, but, attempts to adjust filter bandwidth by 'juggling' the existing measurement and process noise intensities proved unsatisfactory. A new technique [15] did work, however.

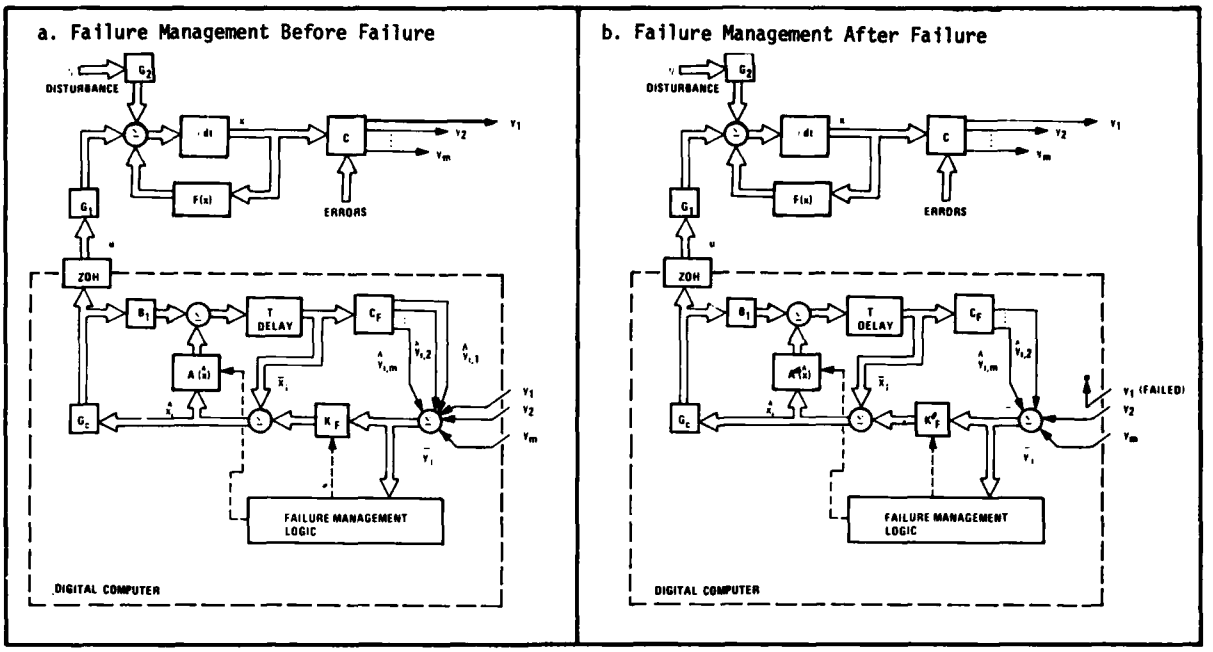


Figure 11. State Estimation With Failed Sensors

Open-Loop System

$$\dot{x} = Fx + G_1 u + G_2 \eta$$
$$y = Hx + E_1 u + E_2 \eta$$

where

$$x^T = (\beta,\ p,\ r,\ \phi_g)$$
$$y^T = (n_y,\ p_m,\ r_m)$$
$$u^T = (\delta_r,\ \delta_a,\ \phi_m)$$
$$\eta^T = (\eta_g,\ \eta_{n_y},\ \eta_p,\ \eta_r,\ \eta_\phi)$$

Matrix Definitions

$$F \triangleq \begin{bmatrix} \frac{Y_\beta}{V_o} & \frac{Y_p}{V_o} & \frac{Y_r}{V_o} - 1.0 & \frac{Y_\beta}{V_o} \\ L'_\beta & L'_p & L'_r & L'_\beta \\ N'_\beta & N'_p & N'_r & N'_\beta \\ 0 & 0 & 0 & -a_g \end{bmatrix} ; \quad G_1 \triangleq \begin{bmatrix} \frac{Y_{\delta_r}}{V_o} & \frac{Y_{\delta_a}}{V_o} & \frac{g}{V_o} \\ L'_{\delta_r} & L'_{\delta_a} & 0 \\ N'_{\delta_r} & N'_{\delta_a} & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$G_2 \triangleq \begin{bmatrix} 0 & 0 & 0 & 0 & \frac{g}{V_o} \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \sigma_g\sqrt{\frac{2}{\Delta t}} & 0 & 0 & 0 & 0 \end{bmatrix} \quad (\alpha_g = 6.0 \text{ ft/sec})$$

$$H \triangleq \begin{bmatrix} Y_\beta & Y_p & Y_r & Y_\beta \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} ; \quad E_1 \triangleq \begin{bmatrix} Y_{\delta_r} & Y_{\delta_a} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$E_2 \triangleq \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Figure 12. A-7 Lateral-Directional Filter Design Plant

As described in [15] one can directly tradeoff system stability margins with filter noise rejection performance. The procedure was applied by appending the noise input matrices, B_2 and D_2, with the control input matrices, B_1 and D_1, respectively.

$$x_{i+1} = Ax_i + B_1 u_i + \begin{bmatrix} B_2, B_1' \end{bmatrix} \begin{bmatrix} \omega_i \\ \pi_i \end{bmatrix}$$

$$y_i = Cx_i + D_1 u_i + \begin{bmatrix} D_2, D_1' \end{bmatrix} \begin{bmatrix} \omega_i \\ \pi_i \end{bmatrix}$$

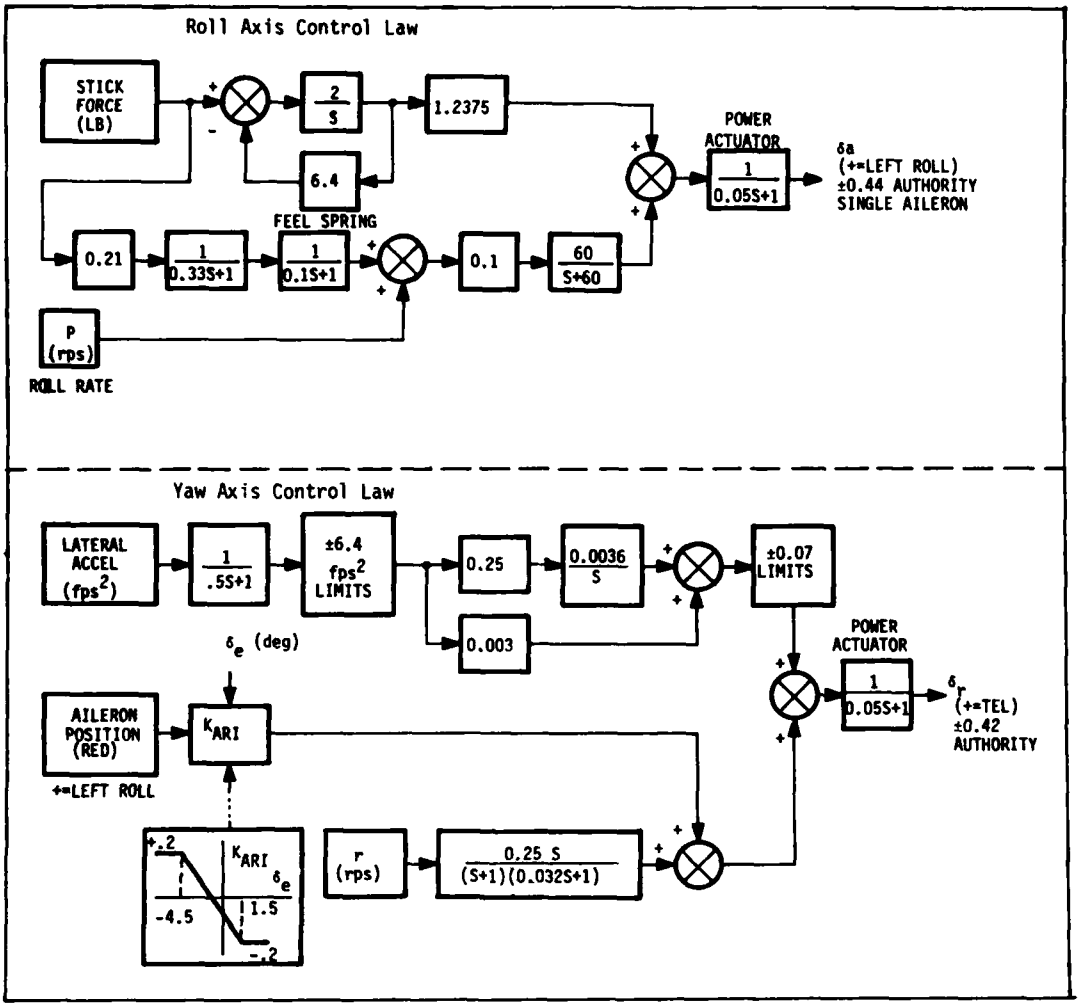


Figure 13. A-7D Roll and Yaw Axes Control Laws

where π_i is ficticious noise.

The prime indicates that only selected inputs are included, i.e., and matrix columns of B_1 and D_1 are not all used. Also, D_1' was set to zero allowing no input from the fictitious noise input, π_1, to influence the acceleration equation.

The fictitious noise input, π_i , is Gaussian

$$E[\pi_i] = 0$$

$$E[\pi_i \pi_j^T] = \rho^2 I \delta_{ij} \qquad (I = 2x2 \text{ identity matrix})$$

When $\rho = 0$ the optimum Kalman filter results. When the stability characteristics approach those of the original system, i.e., with R_m in the loop. The impact of free parameter, ρ, on continuous equivalent filter eigenvalues is seen in figure 15. The plot demonstrates Butterworth root exursion patterns typical of optimal systems.

The R reconstruction design was evaluated on Honeywell's hybrid computer facility. Aircraft dynamics were simulated over the complete A-7 flight envelope on a PACER 100 hybrid computer. On board computer functions for analytical redundancy and state reconstruction were installed on a SIGMA 5 computer. Primary control functions were implemented on the analog portion of the PACER system.

R reconstruction is demonstrated in figure 16 with a response to a step β -gust input. The (a) column represents the unfailed closed loop system, (b) and (d) illustrate the system performance without R . Column (c) curves show the reconstruction filter in the loop. Close comparison of (a) and (c) demonstrate the design performance. Comparison of (c) and figure 14 show the increased robustness of the system.

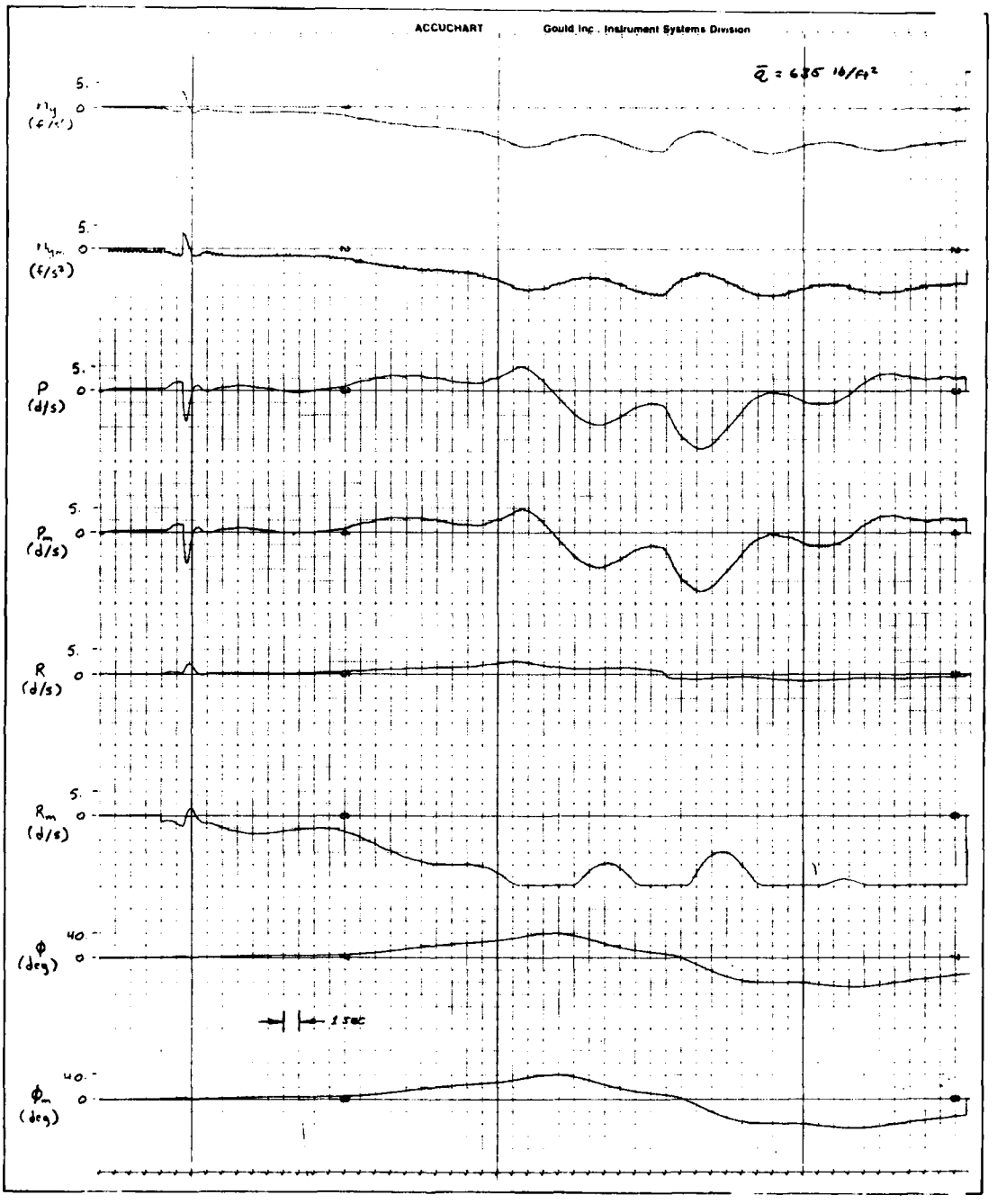


Figure 14. β Gust Response, R Reconstruction Using Optimal Kalman Gains

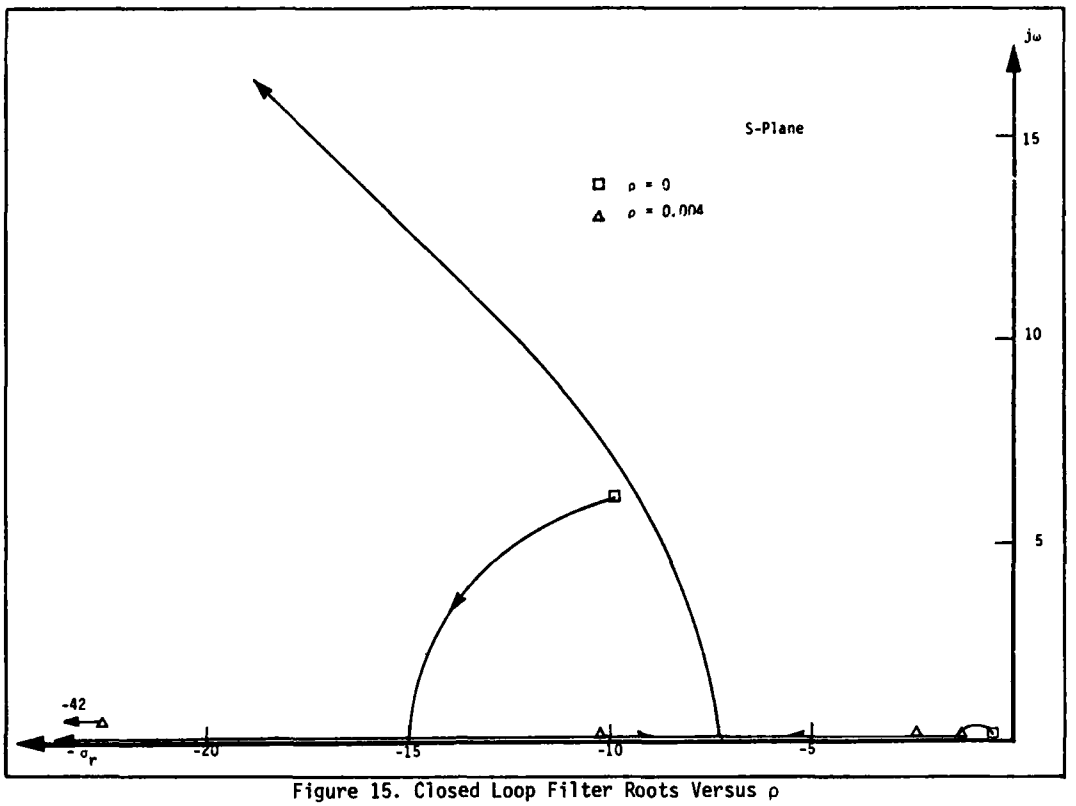


Figure 15. Closed Loop Filter Roots Versus ρ

Reduced Order Observers for Feedback Control

The use of full state observers as exemplified by the A-7 example carries with it a computational load and complexity which is not always necessary. For various reasons one might wish to examine reduced order observers to reconstruct sensors missing due to random failure or combat loss.

- Complexity and size of full state observers might be beyond the acceptable limits of simple software verification.
- Kalman Filter design techniques are based on balancing noise responses. Some sensors are virtually noise free (i.e., those shared with an INS). The predominant design conditions are, therefore, not noise rejection.
- Some reduced order observer design techniques attack robustness problems directly, as demonstrated in [16] and applied in [17], and therefore can be used with success.

The following design example illustrates a reduced order observer design for a commercial transport. Although a combat survivability issue does not exist here, the design technique would be identical regardless of the source of failure.

A Pitch Rate Observer for IAAC [18] -- IAAC, which stands for Integrated Application of Active Control, is being investigated by Boeing for NASA to determine fuel saving benefits. Part of the control design goals involve restabilizing a Related Static Stability (RSS) pitch axis with active control. A preliminary control law containing two feedback loops is shown in figure 17. The airspeed loop controls the phygoid mode while the pitch rate loop stabilizes the RSS mode and is therefore flight safety critical.

The pitch rate signals are obtained from the inertial reference system. Assuming a triple system reveals a high degree of reliability. If, however, this were a military aircraft with a shared sensor set as shown in figure 3a a single shot would eliminate the entire source of pitch rate signals. Short of adding additional dispersed pitch rate sensors the possibility of using a normal accelerometer to observe the vehicle pitch rate for feedback into the existing control loop was explored.

A Luenberger observer design technique which is based upon eigenstructure placement rules (eigenvalue-eigenvector placement) is briefly outlined in figure 18. Using this technique, a second order observer was designed for pitch rate using normal acceleration at one flight condition. This is shown in figure 19.

The design was based upon a third order mode for the vehicle short period dynamics plus a single actuator root at 20 rad/sec. This plus the single measurement of n produces a theoretical minimum order observer of two states.

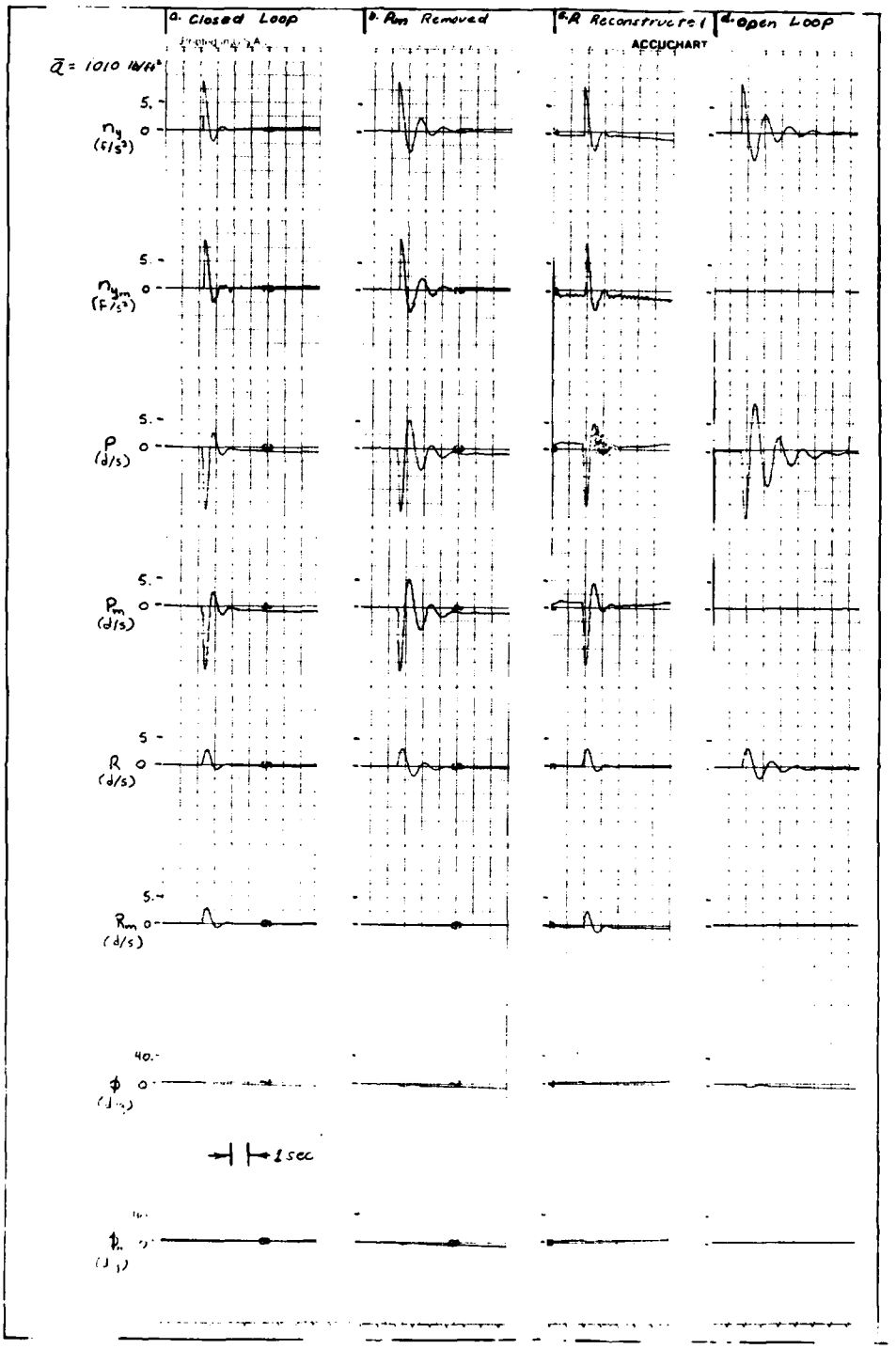


Figure 16. β Step Gust Response, R Reconstruction

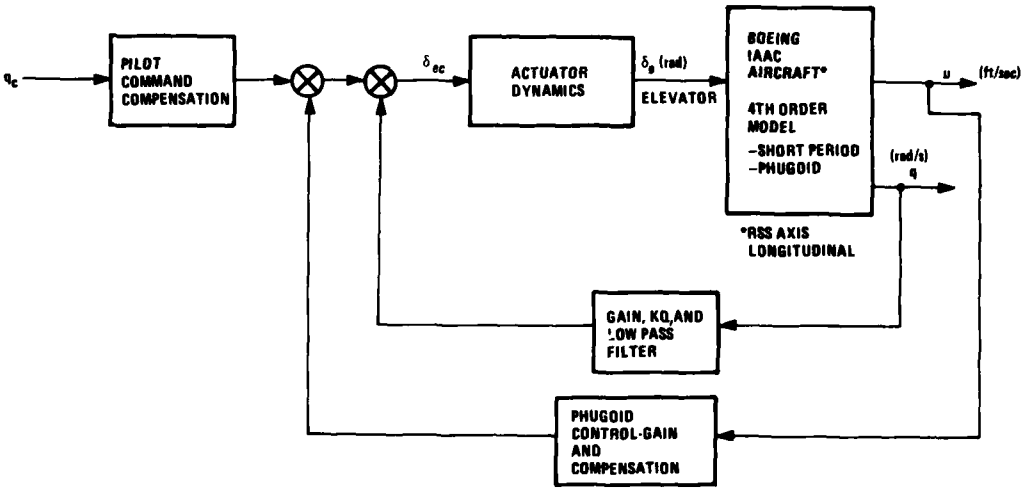


Figure 17. IAAC Pitch Axis Control Structure

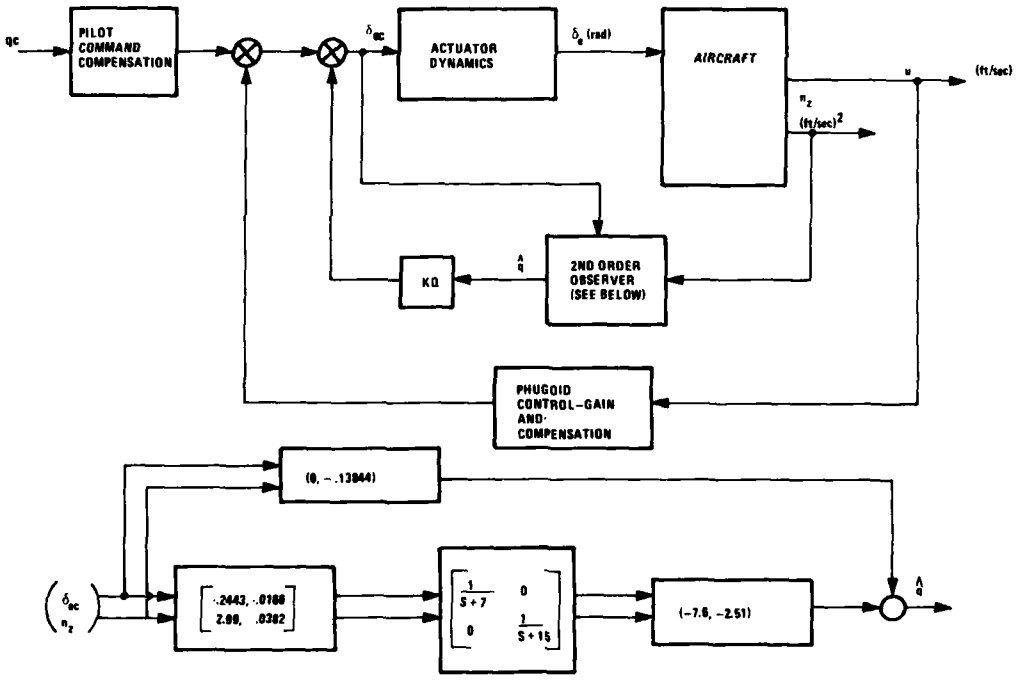
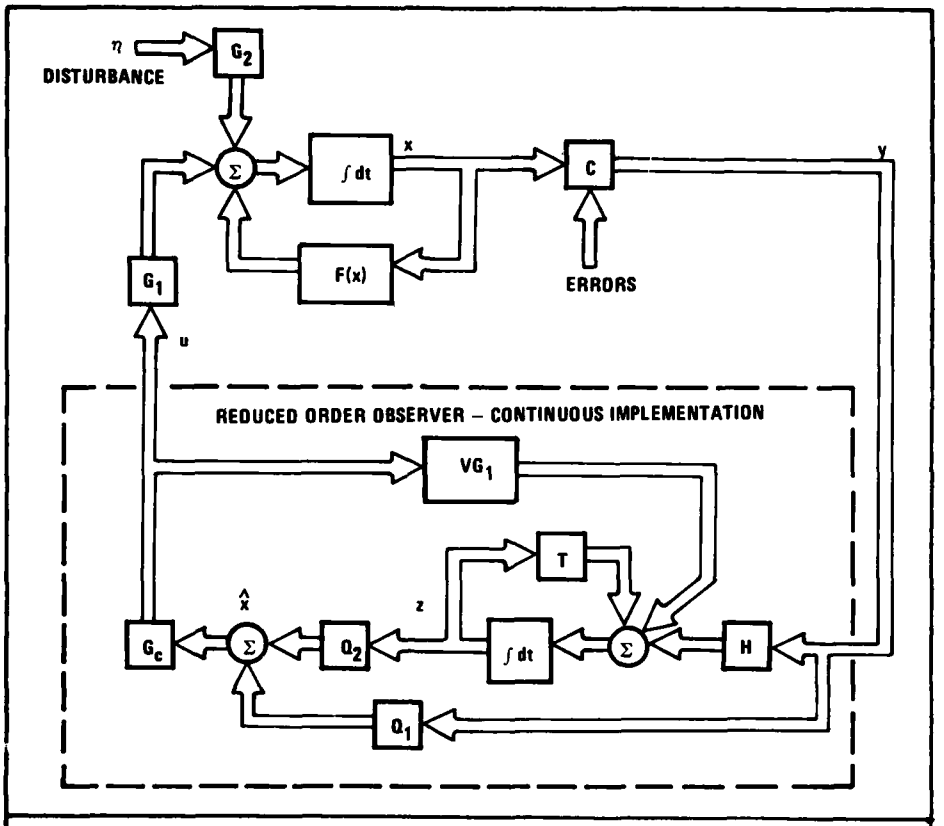


Figure 19. Pitch Rate Observer In the Loop



PLANT

$\dot{x} = Fx + G_1u + G_2\eta$ (n^{th} order)

$y = Cx +$ errors (p^{th} order)

OBSERVER (Order = n-p)

$\dot{z} = Tz + Hy + VG_1u$

$\hat{x} = \begin{bmatrix} Q_1, Q_2 \end{bmatrix} \begin{bmatrix} y \\ z \end{bmatrix}$

DEFINING CONDITIONS

$VF - HC = TV$

$\begin{bmatrix} Q_1, Q_2 \end{bmatrix} = \begin{bmatrix} C \\ V \end{bmatrix}^{-1}$

define V, H, and T

Q_1 and Q_2

MODAL OBSERVER DESIGN TECHNIQUES [17]

1. Choose T
   = diagonal matrix for good transient response
2. Choose admissable elements of V to
   a. Make $\begin{bmatrix} C \\ V \end{bmatrix}$ well conditioned for inversion
   b. Make observer robust
      - V → orthonormal [17]
      - VG_1 → 0 (null) [16]
3. Find remainder of V and H by solving [19]

   $V(F-KC) = TV$

   for V and K where

   $H = VK$

Figure 18. Reduced Order Observer Design

The performance of the observer is summarized in figure 20. 20a contains the transient responses for q and to a step elevator command, plus the original stability margins. figure 20b contains results for the observer in the loop. Key results are:

1. Gain margin is reduced from 15 db to 8 db. The specification is 6 db or greater.

2. Pitch rate transients are very close for the two designs. The pitch rate response differs from the observed pitch rate response at low frequency (the transient response simulation includes the phygoid mode). This is expected because the observer design was performed only on the short period mode.

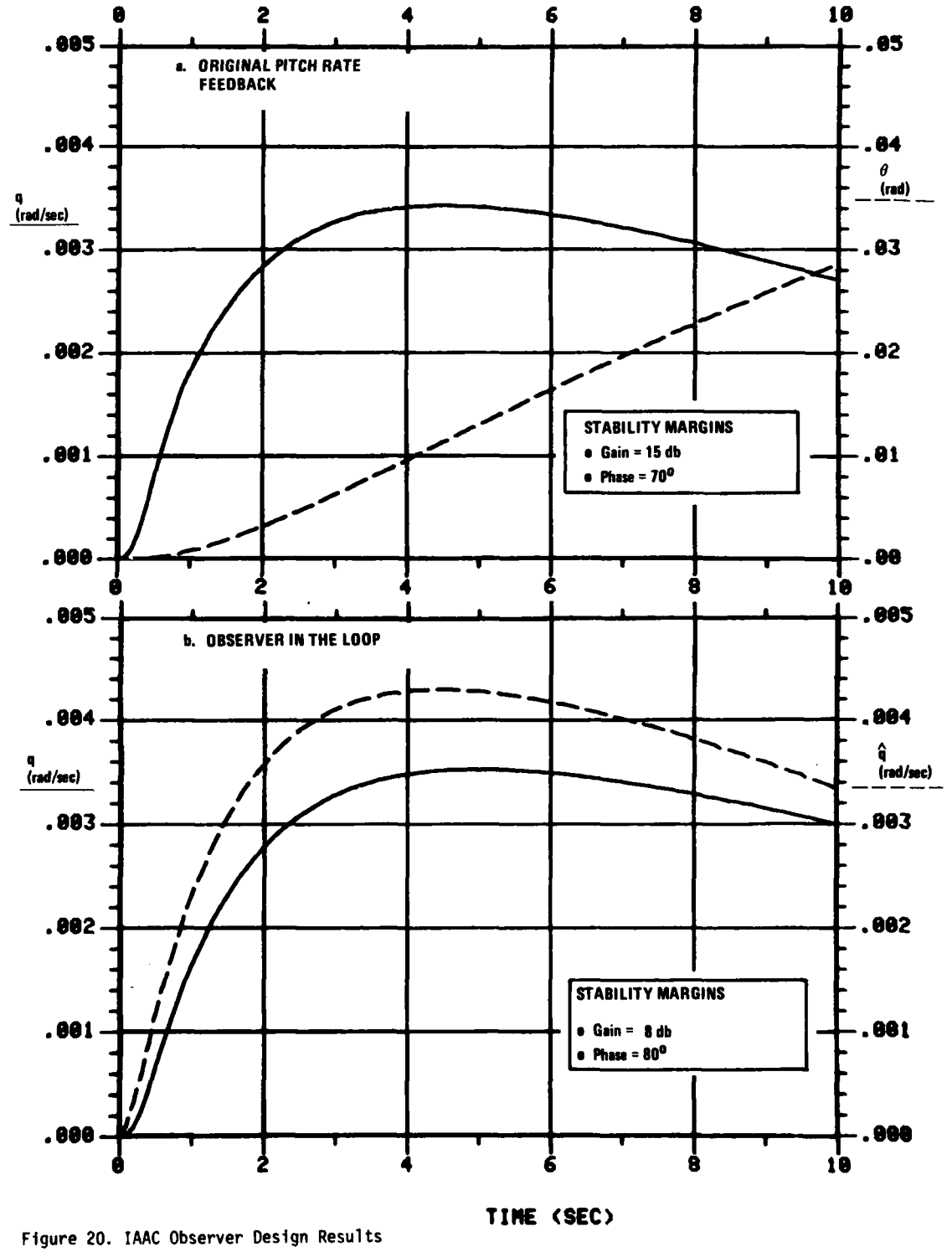


Figure 20. IAAC Observer Design Results

2.2 Reconfiguration of Surfaces

The number of control surfaces on modern aircraft has increased in order to implement advanced combat modes. The use of prestored reversion modes for individual surface out situations can greatly improve aircrafts survivability. The stored control laws require little extra memory and the designs can be made with straight forward control synthesis techniques.

One area of interest for safety assurance is the guarantee of flight control integrity with respect to failures. This idea means that if a high probability failure occurs then the remaining control system and vehicle will remain stable. The statically unstable F16, for example, has two stabilators each with the same pitch authority. The control law in this case has integrity to an individual stabilator failures by making the control law maintain a negative gain margin of at least 6 db. Other integrity design concepts are discussed by Ackerman [20].

A Grumman study [2] for reversion mode control design was discussed earlier. Another study was conducted for an YF-16 aircraft, i.e., F-16 with canards (figure 1) at a single flight condition to assess the reversion mode design feasibility of that vehicle. Results are described in reference 21 and summarized in Table 5.

TABLE 5. YF-16 REVERSION MODES

Surface Configuration	Pitch Control	Roll	Yaw	Status of Direct Force Modes
All intact (primary)	Collective Stabilators	Differential stabilators and flaperons	Rudder	All surfaces used
One stabilator out	• New feedback law on remaining stabilator • Crossfeed to flaperons and rudder	Flaperons only	Rudder (primary)	• Retain side force modes • Drop lift modes
One flaperon out	Primary mode	Differential stabilators only	Rudder (primary)	• Retain side force modes • Drop lift modes
Rudder out	Primary mode	Combination feedback and feedforward control using stabilators, flaperons, and commands	Forward differential	• Retain lift modes • Drop side modes

3. A 1990 SURVIVABLE FLUTTER MODE CONTROL SYSTEM

The concepts and ideas discussed in previous sections can be used for evaluating the various hardware and software tradoffs one can postulate for a given design goal. The following is an example of a combination of emerging hardware technologies and how they might be blended with our failure management techniques to result in a survivable flutter mode control (FMC) system.

The FMC is optimized for performance with sensors at four locations. The system can be made stable (for all operational speeds) with any two of the sensor locations working, although some control law modifications are necessary.

System Description

The system is described as follows.

Hardware:

- Dispersed remote "Smart" accelerometers each consisting of
  1. 2 piezoresistive element stain gauge type accelerometers
  2. 1 microcomputer element with 4 12 bit A/D converters and enough logic capability to perform high rate
     - comparison monitoring (for sensor and A/D fault detection)
     - anti aliasing for lower sample rate usage
     - smoothing and bandpass filtering (if needed)
     - interface to the data BUS
     - analytical redundancy with incoming sensor data
- Total data transmission via a fiber optic data BUS.
- Redundant central processors which blend all incoming sensor information to:
  - perform control command generation
  - reconfiguration after failures

Software:

- Advanced control law design implementation via Kalman filter state reconstruction
  - estimate states for optimal feedback from numerous sensor inputs
  - reconfiguration of filter after sensor failure (i.e., sensor pair drops out).
- Control surface and actuation failure management
- Software stored failure status for later maintenance diagnosis

"Smart" Accleelerometer -- Figure 21 shows the "smart" accelerometer proposed. Use of two low cost piezoresistive accelerometers plus an advanced microporcessor chip provides a very reliable (20,000 hr MTBF) and low cost ($200 goal) component.

The basic failure management begins with a simple comparison monitor for fault detection. Isolation with analytical redundancy is perforemd, however, the basic concept does not rely on this occuring instantaneously because the logic in the main computer will act upon the failure flag, i.e., g set to zero, by ignoring the information from sensor pair i.

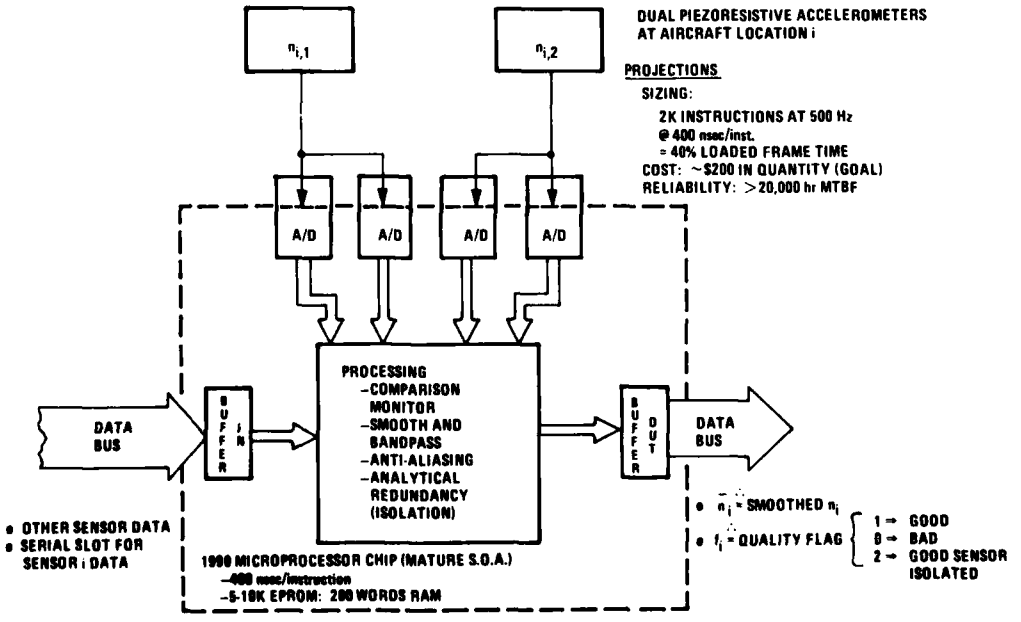


Figure 21. "Smart" Accelerometer For Flutter Mode Control

The failure isolation will, therefore, be continued until a high confidence hypothesis of an isolated failure is obtained. This will be transmitted to the main computer for two subsequent actions.

1. Storage of isolation information for later use by maintenance for LRU servicing.

2. Possible use of remaining "Good" sensor in control computations if redundancy levels of remaining sensor locations is reduced to a safety of flight hazard.

Main Computer Functions -- The appropriate use of the "smart" accelerometer data in the main control computer improves the overall performance for the FMC control system.

Figure 22 shows an implementation using four sensor locations. These locations are representative of a system which attempts to measure and isolate certain wing bending and torsional modes.

A Kalman belnder is used to isolate the appropriate states for control feedback. Based upon the "smart" sensor flag logic both the Kalman filter and control law gain matrices can be changed to reflect a level of sensor degredation or control law reversion mode.

Further failure detection can be performed on the Kalman blender residuals. Because the sensors faults are detected independent of the filter operation, failures due to surface failures can be attacked with the appropriate hypothesis monitor logic.

4. CONCLUDING REMARKS

The use of the software techniques for high survivability discussed in this paper requires additional analysis to assess vulnerability under various threats conditions and scenarios. This aspect of the problem is important but can be examined conceptually independantly of the techaniques discussed here.

Survivability analysis computer packages can become extremely involved and expensive to use. A "Quick look" survivability analysis tool has been developed for low cost assessment of dispersion effectiveness and sensitivity analysis [22]. This has proven effective for preliminary design of aircraft avionic systems.

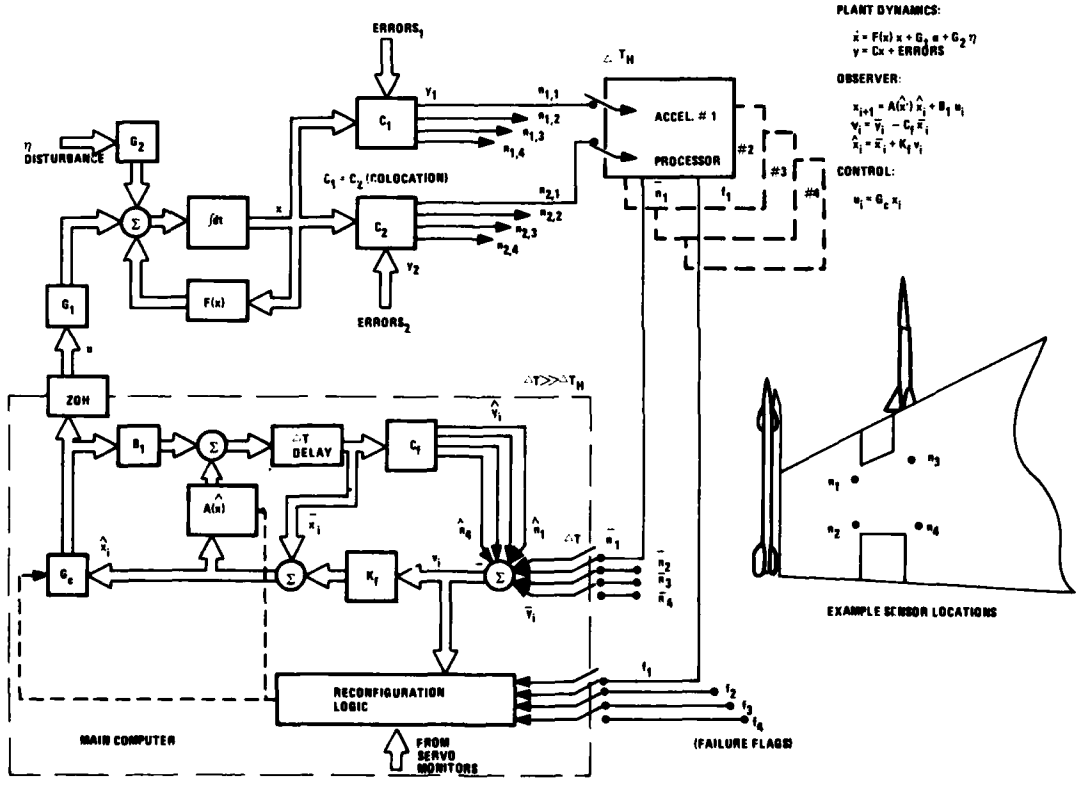
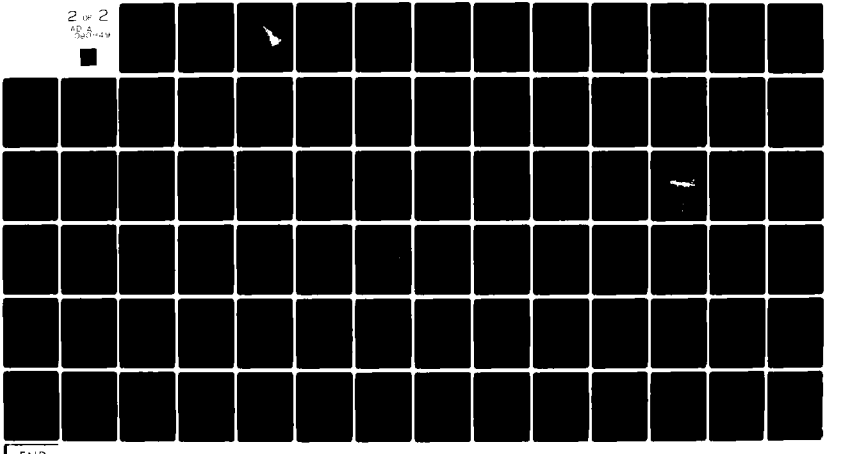


Figure 22. 1990 FMC Implementation

AD-A090 849 ADVISORY GROUP FOR AEROSPACE RESEARCH AND DEVELOPMENT--ETC F/G 9/3
FAULT TOLERANCE DESIGN AND REDUNDANCY MANAGEMENT TECHNIQUES.(U)
SEP 80
UNCLASSIFIED AGARD-LS-109 NL

2 OF 2
AD A 090849



END
DATE FILMED
11-80
DTIC

5. REFERENCES

1. Deyst, J.J., and Hopkings, "Highly Survivable Integrated Avionics", Aeronautics and Astronautics, Vol. 16, No. 9, September 1978.

2. Boudreau, J.A., and Berman, H.C., Dispersed and Reconfigurable Digital Flight Control System Study, Air Force Flight Dynamics Laboratory, Final Report No. AFFDL-TR-79-3125, December 1979.

3. Folkesson, K., "Computer Base In-Line Monitoring," AGARD Lecture Series No. 109 - Fault Tolerance Design and Redundancy Management Techniques, AGARD-LS-109, October 1980.

4. Cunningham, T.B., et.al., Fault Tolerant Digital Flight Control with Analytical Redundancy, Air Force Flight Dynamics Laboratory, Final Report No. AFFDL-TR-77-25, May 1977.

5. Cunningham, T.B., and Poyneer, R.D., "Sensor Failure Detection Using Analytical Redundancy," 1977 Joint Automatic Control Conference, proceedings June 1977.

6. Burns, R.C., Multi-Function Reference Assembly (MIRA), Air Force Flight Dynamics Laboratory, Final Report No. AFFDL-TR-78-105, September 1978.

7. Harvey, C.A., and Pope, R.E., "Design Techniques for Multivariable Flight Control Systems," Theory and Application of Optimal Control In Aerospace Systems, AGARDOGRAPH GCP-56, June 1980.

8. Lindgren, B.W., Statistical Theory, MacMillan Co., London, 1968.

9. Willsky, A.S., "Failure Detection is Dynamic Systems," AGARD Lecture Series No. 109 - Fault Tolerance Design and Redundancy Management Techniques, AGARD-LS-109, October 1980.

10. Deckart, J.C., et.al., Reliable Dual-Redundant Sensor Failure Detection and Identification for the NASA F-8 DFBW Aircraft, NASA CR 2944, February 1978.

11. Szalai, K.J., "Flight Experience with Flight Control Redundancy," AGARD Lecture Series 109 - Fault Tolerance Design and Redundancy Management Techniques, AGARD-LS-109, October 1980.

12. Bryson, A.E., and Ho, Y.C., Applied Optimal Control, Ginn and Co., London, 1969.

13. Doyle, J.C., "Guaranteed Margins for LQG Regulators," IEEE Transactions on Automatic Control, Vol. AC-23, No. 4, August 1978.

14. Cunningham, T.B., Doyle, J.C., and Shaner, D.A., "State Reconstruction for Flight Control Reversion Modes," 1977 Conference on Decision and Control, December 1977.

15. Doyle, J.C., and Stein, G., "Robustness with Observers," IEEE Transactions on Automatic Control, Vol. AC-24-No. 4, August 1979.

16. Stein, G. and Athens, M.,Asymptotic Weight Selection For Optimal Regulators and Filters, MIT Report LIDS-FR-921, June 1974. See also proceedings of the IEEE Conference on Decision and Control, December 1979.

17. Ostroff, J.S., and Moore, B.C., "A Guide to Reduced Order Observer Design," IEEE Conference on Decision and Control, December 1979.

18. Shomber, H.A., "Application of Integrated Active Controls to Future Transports," AIAA Atmospheric Flight Mechanics Conference and Special Session: Energy Efficient Aircraft Design, AIAA paper 79-1654, August 1979.

19. Cunningham, T.B., "Eigenspace Selection Procedures For Closed Loop Response Shaping with Modal Control", IEEE Conference on Decision and Control, November 1980.

20. Ackerman, J., "Robust Control System Design," AGARD Lecture Series No. 109 - Fault Tolerance Design and Redundancy Management Techniques, AGARD-LS-109, October 1980.

21. Watson, J.H., Yousley, W.J., and Railey, J.M., "Redundancy Management Considerations for a Control Configured Fighter Aircraft Triplex Digital Fly-By-Wire Flight Control System," AGARD Symposium on Advances in Guidance and Control Systems Using Digital Techniques, Ottawa, Canada, May 1979.

22. Graham, K.D., et.al., Aircraft Flight Control Survivability Through the Use of On-Board Digital Computers, Joint Technical Coordinating Group For Aircraft Survivability, JTCG/AS-77-T-002, May 1980.

FAILURE MANAGEMENT FOR THE SAAB VIGGEN JA37 AIRCRAFT

by
Kjell Folkesson
SAAB-Scania, Aerospace Division
Linkoping, Sweden

SUMMARY

The JA-37 Viggen is the first military aircraft in series production and field-service equipped with a digital automatic flight control system. The JA-37 Digital Automatic Flight Control System (DAFCS) has high control authority and is a flight safety critical system. It has duplex sensors, a single channel digital computer, and simplex secondary servos. The digital computer performs control-law calculation and sensor and servo monitoring, as well as extensive self test on ground and during flight. The sensors are monitored by comparison. The servos are monitored by comparing the output from a software model with the servo output.

The digital computer is self tested by various built-in-test (BIT) programs and checks. The tests are mainly parts of the resident computer program, though some tests involve external hardware. The in-flight BIT must test all critical computer components with a very high degree of confidence. The design phase included extensive effort to develop the DAFCS safety system and verify its efficiency. The DAFCS software had to be organized for ease of understanding, avoidance of errors, and effectiveness of testing. It had to be clearly documented and strict routines for changes had to be followed.

The DAFCS hardware required special attention. A detailed failure mode and effects analysis was performed to identify all flight safety critical failures. Computer tests were developed to detect all failures before severe aircraft motion transients can occur. The efficiency of the safety system was documented by calculation and verified in rig (ironbird) simulation and flight test.

The results of the JA-37 DAFCS design phase proved that the single channel DAFCS met the stated safety requirements. Later field experience has verified that the JA-37 production DAFCS is a reliable and safe system.

1. INTRODUCTION

1.1 Aircraft 37 Viggen Program

The aircraft 37 Viggen program started in the early sixties. The design goal was to define and develop a multi purpose aircraft which could perform ground attack, reconnaissance, and fighter missions. The differences between the versions were supposed to be different weapons and electronics only. The airframe and the Flight Control System (FCS) should remain unchanged.

The flight envelope of the multi-purpose aircraft was large including both high and low speeds at high and low altitudes. Especially good performance at the low speed landing configuration was desired. As a result of these requirements the double delta wing configuration with a front canard was selected and the aircraft was named Viggen (thunderbolt).

The various SAAB 37 Viggen versions are listed in Table 1.

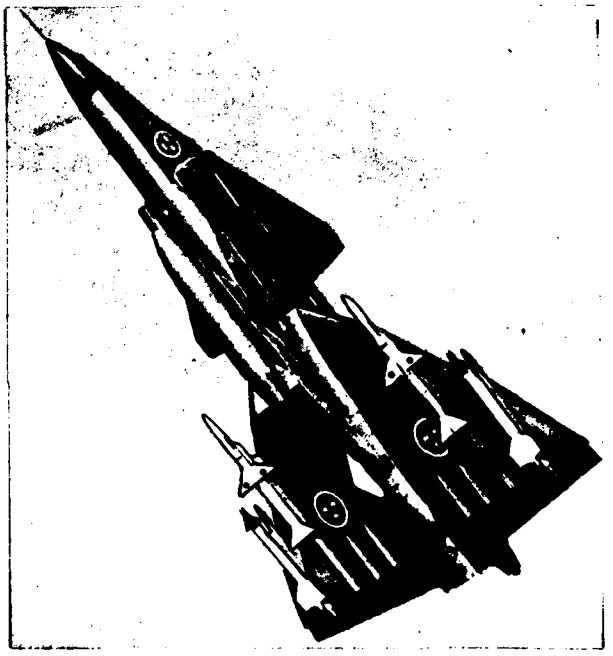
The multipurpose goal was met for all versions except for the fighter version JA37 Viggen. The JA37 fighter required improved performance and maneuverability to meet the stated mission requirements. To accomplish this, significant changes had to be made to the FCS.

Figure 1 shows the JA37 Viggen fighter double delta wing configuration. The canard itself produces lift and in addition the airstream from the canard over the main wing causes a favorable additional lift effect. The four bulges under each wing houses four tandem servo actuators. Some basic data for the JA37 Viggen is also given in Figure 1.

TABLE 1.

THE SAAB 37 VIGGEN PROGRAM

GO AHEAD	1962
37-1 1:ST TEST A/C FLIGHT	1967
DELIVERIES TO SWEDISH AIR FORCE	
AJ37 GROUND ATTACK/FIGHTER	1971
SK37 TRAINER	1972
SH37 SEA SURVEILLANCE	1975
SF37 RECONNAISSANCE	1977
JA37 FIGHTER/GROUND ATTACK	1979



Version

JA 37 Fighter/Ground Attack

Main wing span 10.60 metres
Normal take-off weight 17.000 kg
Max. level speed Mach 2

Figure 1. Saab 37 Viggen

1.2. Assumptions for the JA37 FCS

The JA37 requirements affecting the flight control system are listed in Table 2.

Good maneuverability is obviously a demanding requirement for a fighter aircraft. Transonic trim tranisents had to be minimized as passages through transonic are frequently expected. Improved pilot precision control was desired for direct aiming weapons such as the fix mounted gun.

The low cost requirement was very important. The cost of the design and development (D&D), series production, and maintenance phases had to be considered.

TABLE 2.

REQUIREMENTS ON THE JA37 FCS/AFCS

FUNCTION

INCREASED MANUEVERABILITY
MORE G's, MORE TURN RATE

SMOTHER TRANSONIC PASSAGE
SUPPRESSED TRIM TRANSIENTS

IMPROVED PILOT PRECISION CONTROL
IMPLEMENTATION OF AN AIMING MODE

COST

MINIMIZED LIFETIME COST
EASY TO MODIFY DURING DEVELOPMENT
LOW PRODUCTION COST
LOW MAINTENANCE COST

FLIGHT SAFETY

PROBABILITY OF 07 DAFCS "CATASTROPHIC FAILURE" $< 10^{-6}/1.5$ HR
FAILURE TRANSIENTS $-2G < \Delta N_Z < 3G$
$|N_Y| < .5G$
LATERAL DISPLACEMENT $< 3.5M$
PROBABILITY OF NUISANCE DISENGAGEMENT $< 10^{-3}/1$ HR

Finally the flight safety aspects were crucial. It was known at this time that the Automatic Flight Control System (AFCS) had to be a high authority system, from a flight safety point of view. This meant that the AFCS must be given authority enough to command high control surface rates and large surface deflections. The high authority could cause dangerous aircraft responses if a failure occurred and was not quickly eliminated.

The probability of catastrophe was required to be less than 10^{-6} per 1.5 flight hour due to failure in the AFCS. A catastrophe was conservatively defined as a situation where an AFCS failure caused an aircraft transient response exceeding the specified failure transient requirements. Some of the more significant failure transient requirements are listed in Table 2.

It can be noted that an AFCS failure had to be detected and eliminated within a very short time. Failure elimination was allowed to be done by disengagement of the AFCS since the aircraft is stable and a mechanical Primary Flight Control System (PFCS) provides acceptable flight control qualities. The conflict between a very tight and efficient safety system and the risk for false, or nuisance, AFCS disengagements was investigated. The investigation resulted in a requirement on probability for nuisance disengagement of less than 10^{-3} during one hour flight. This requirement corresponded to the expected probability of hardware failure since the AFCS MTBF was assumed to be around 1000 hours.

1.3. JA37 FCS/DAFCS Program

As a consequence of the requirements discussed above, modifications were decided upon for implementation in the JA37 Viggen flight control system. These are listed in Table 3 and discussed below.

The control surface actuator power was increased to allow more control surface deflection at high G-flight conditions. An electromechanical series trim controlled by the flight control computer was integrated in the PFCS. The inputs to the series trim control laws are pilot command, load factor and pitch rate. The series trim reduces the transonic

TABLE 3.

SELECTED CONFIGURATION JA37 FCS/AFCS

PFCS

INCREASED SERVO-RAM POWER
ADDED PITCH SERIES TRIM

AFCS

SINGLE CHANNEL DIGITAL AFCS (07 DAFCS)
EASY TO MODIFY BY SW CHANGES
MINIMUM PRODUCTION HARDWARE
HONEYWELL SELECTED FOR THE D & D PHASE

QUESTION:

THE 07 DAFCS IS A HIGH AUTHORITY SYSTEM. WILL A SINGLE CHANNEL DAFCS WITH BUILT IN SELF TEST BE SAFE TO FLY?

THE DESIGN AND DEVELOPMENT PHASE HAD TO PROVIDE THAT ANSWER.

pitch trim changes to a minor fraction of what it otherwise would be. The decision to go digital for the AFCS, renamed 07DAFCS, was based on cost requirements plus available airworthy digital minicomputers in the early 1970's. It was cost effective because control laws and logic in software could easily be modified and tested. Maintenance could be greatly simplified since the digital computer could be loaded on ground with a separate automated test program.

The big question mark was whether flight safety required dual digital computers and comparison monitoring or if in flight Built-In-Test (BIT) had sufficient failure detection coverage to justify a decision on a single digital computer. The single computer alternate was preferred because it meant minimum hardware minimum cost and minimum maintenance. It was decided to start rig and flight test with dual comparison monitored digital computers and fund an extensive safety verification program which should verify that a single computer DAFCS was safe to fly.

The JA37 FCS and DAFCS program was from the go ahead a joint effort between Honeywell and SAAB closely coordinated with the Swedish Air Force technical personnel. The major facilities involved in the JA37 FCS/DAFCS program as well as the major activities are listed in Table 4.

TABLE 4.

JA37 FCS/DAFCS PROGRAM

JOINT EFFORTS SAAB/HONEYWELL
FACILITIES
1 FCS - R1G
1 A/C - SYSTEMS - R1G
1 TEST A/C , FCS
1 TEST A/C , A/C SYSTEMS
1 TEST R1G FOR BIT DEVELOPMENT
ALL INCLUDING 07 DAFCS PTs
ACTIVITIES
TO DEVELOP AND VERIFY
CONTROL LAWS AND LOGIC
SAFETY SYSTEM
PRE FLIGHT BIT
MAINTENANCE TESTS

2. DESCRIPTION OF THE JA37 FCS and DAFCS

The 07 DAFCS is a high authority, failsafe, single processor digital control system that provides the following functions:

- o Control Augmentation System (CAS), normal mode
  - Stability augmentation
  - Transonic trim change compensation
- o Attitude Hold
  - Pitch attitude hold
  - Roll attitude hold
  - Heading hold
  - Control stick steering
- o Altitude Hold
- o Automatic Airspeed Control

These control modes are realized with a combination of sensors, computer and servos. The single computer DAFCS will be described in the following section.

2.1. Hardware Components

The JA37 FCS mechanization including the PFCS and DAFCS is shown in Figure 2.

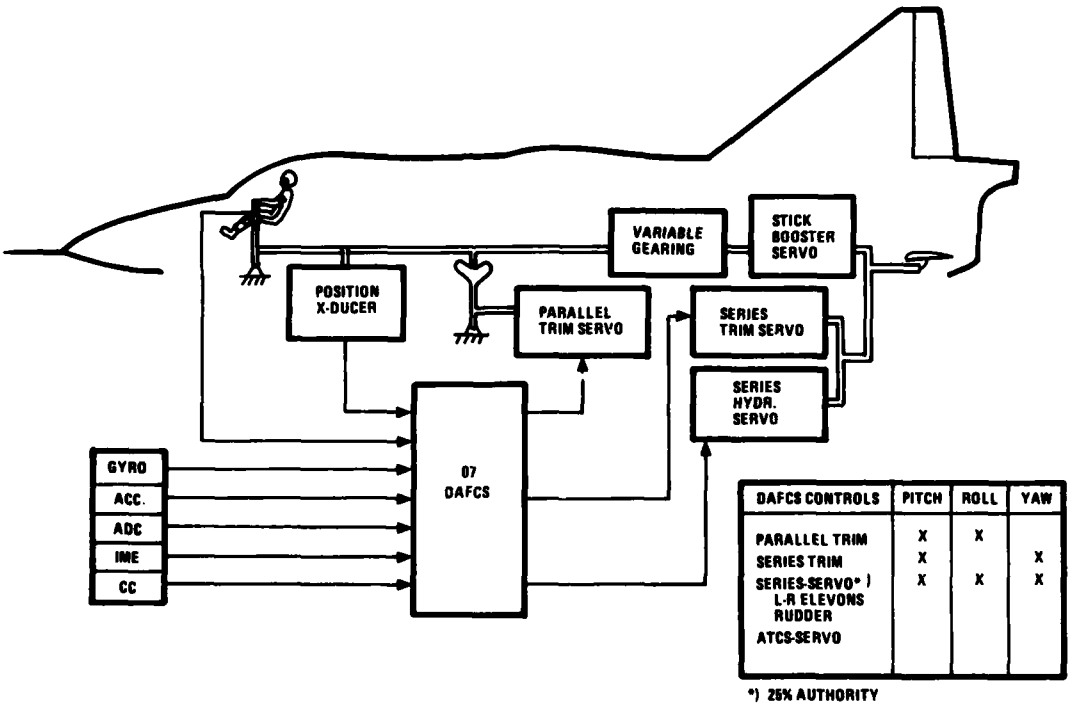


Figure 2. Flight Control System

The JA37 aircraft has only three primary control surfaces, right and left elevon and the rudder. The control surfaces are controlled by the pilot via the mechanical PFCS, by the 07 DAFCS via secondary series servos and via automatic or manual parallel and series trim actuators. The high bandwidth secondary series servos are most critical regarding flight safety. They can command 5 degrees control surface deflection within 0.2 seconds if they fail hardover. This corresponds to about 10G nose up or down in the most critical flight conditions. The table included in Figure 2 shows the servos which are controlled by the 07 DAFCS computer.

The following inputs are provided to the 07DAFCS computer.

- o Pilot commands pitch and roll, duplex position sensors
- o Duplex rate gyros for aircraft angular rates (pitch, roll, yaw)
- o Dual accelerometers for n_z and n_y acceleration
- o Single angle of attack sensor
- o Speed and altitude information from single ADC

- o Euler angles ϕ , θ from a single IMU
- o Miscellaneous signals from the central computer
- o Miscellaneous logic information for mode selection, etc.

The 07 DAFCS system concept is shown in block diagram form in Figure 3.

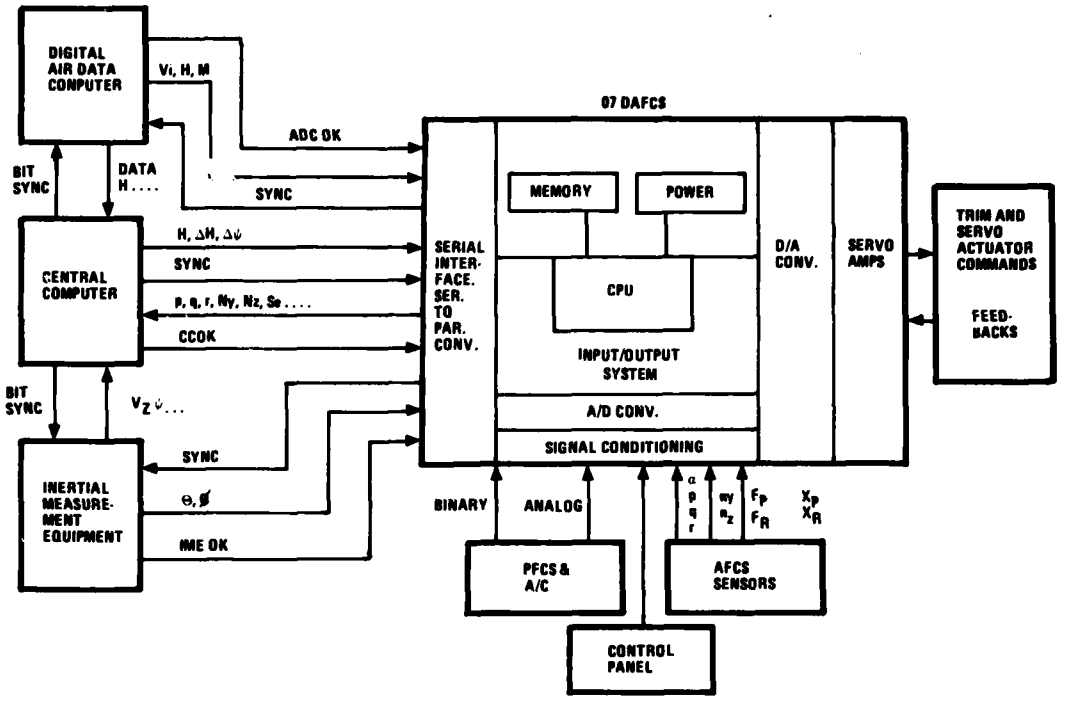


Figure 3. 07 Digital AFCS

The digital computer is a one box unit including the following basic components:

- o Central Processor Unit (CPU)
- o Memory and memory interface
- o Signal conditioning, prefilters
- o A/D and D/A conversion circuits
- o In/out control system
- o Serial interface for serial binary communication
- o All servo summing and power amplifiers
- o Electrical power unit

Also shown in Figure 3 is how the ADC, CC and IMU serial communication are arranged. The central computer provides H and signals for the 07 DAFCS autopilot modes. The central computer receives most of the DAFCS sensor signals from the DAFCS. These signals are used in the CC calculations, they are also stored for maintenance purpose. Significant characteristics of the DAFCS computer are summarized in Table 5.

All flight safety critical 07 DAFCS sensors are duplex. They are monitored by comparison within the DAFCS. All single sensor inputs are limited to a safe magnitude. A servo software model is provided the same inputs as the real servo. The output from the model and the servo response are compared. If they differ more than a given value during a given time the servo is disengaged and locked in a preselected trim position. The digital computer itself is monitored by in flight BIT, which will be described in the following sections.

2.2. DAFCS Computer Program

The DAFCS computer program and the program structure are described to a fairly detailed level in the following. The reason for this is the flight safety criticality of the software. The software program must be structured in a way which simplifies programming, verification, and modification.

Software verification and validation are fundamental activities in the overall flight safety verification program. The DAFCS software is organized for ease of understanding, avoidance of errors, and effectiveness of verification testing. The following is a description of software organization, including rules imposed to ensure uniform, orderly programming.

TABLE 5.

07 DAFCS COMPUTER

HONEYWELL HDC-301 CPU
16 BIT
47 INSTRUCTIONS
FIX POINT
SEMS-8 PLANAR CORE MEMORY
18 BIT WORDS
8 K WORDS
PARITY CHECKING
WRITE PROTECTION
INPUT/OUTPUT SYSTEM
PROGRAM CONTROLLED
DMA
MAX THROUGHPUT
~ 140 KOPS "AFCS MIX"

Rate Structure

The DAFCS Operational Program is organized into the rate structure presented in Figure 4. The rate structure indicates --

- Computer program component (CPU) computation rates
- CPC computation order
- CPC functional allocation
- CPC computation time.

The program starts at START for power ON. The Rate Structure Computation, CPC80A directs the computation consecutively to one of 16 possible computation paths. CPC80A is computed every computation cycle; i.e., 80 per second. CPC40A is computed every other cycle; i.e., 40 per second.

The program HALTS at the start of CPC40A and CPC40B. HALT RELEASE occurs each 12.5 milliseconds. As a result, the CPC40A computation begins precisely each 25 milliseconds, as does CPC40B. Control laws such as inner loop control laws, or servo monitors, are located immediately after a program HALT RELEASE because periodicity is important.

The CPC's are organized by function; i.e., each CPC has functional integrity to the greatest possible extent. Such organization facilitates assignment of the programming task and simplifies verification testing. Each CPC is composed of a set of adjacent instructions with a single entry and single exit for normal operation. CPC's have dedicated subroutines located at the end of the CPC instruction set.

Data Organization

Data organization is based on the following principles:

- o Variables "stored into" by only one CPC
- o Each computer output signal generated from only one CPC
- o Limited number of constants used indiscriminately throughout the program identified as "common constants." Remaining constants are identified as "unique" and used in only one CPC.

The limiting of storage variables and outputting of variables to one CPC results in an orderly and systematic program, and simplifies the analysis of program operation. Use of "common" and "unique" constants restricts the impact of changes in constant values.

Variables "stored into" by a specific CPC are listed together in alphabetical order. The various CPC variable lists are ordered, as are the CPC instruction sets. Indexed variables have separate labels in the variable lists.

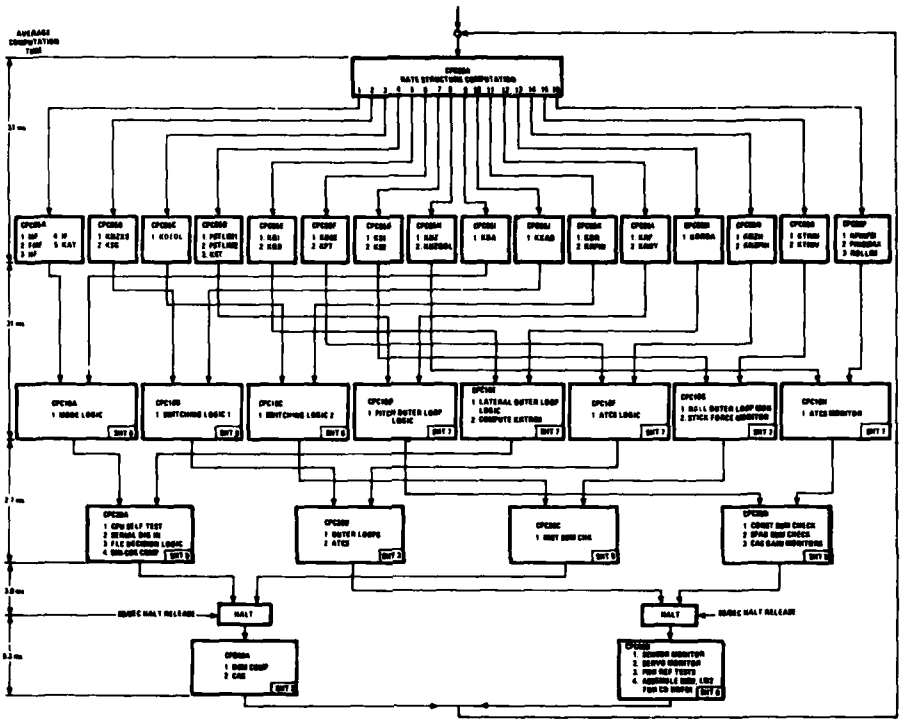


Figure 4. 07 DAFCS Software Structure

CPC Functional Allocation Criteria

Control Laws - Control laws are assigned to CPC by function. A particular function is completely contained within a CPC. Inputs to the control law CPC can include sensor outputs, logic variables, and scheduled gains which are processed or computed in other CPC's. Control laws with tight tolerances on dynamic characteristics are located immediately after HALT RELEASE to ensure periodicity.

Input/Output - Input signal processing is performed within the used CPC if the signal is used in only one CPC. Input signals from a single source (e.g., digital air data signals) are all received within one CPC. Each output signal is generated from only one CPC. Output signals to a single destination (e.g., instrumentation data recorder) are sent within one CPC.

Monitoring - Fault detection monitoring is assigned to CPC's by function. Computation rates for monitoring functions are dependent on --

- o Fault reaction time requirements
- o Number of fault indications prior to fault reaction required to minimize nuisance disengage probability.

Monitoring of analog devices e.g., dual sensor comparison monitoring and servo to servo model comparison monitoring allows at least three miscompares prior to fault reaction (disengagement). Digital hardware monitoring functions such as CPU self-test, memory sum check, activate fault reaction after a single fault indication.

Monitor function computation rate must be sufficiently high such that time required for fault identification, such as three computation cycles for analog devices, plus time required for fault reaction, e.g., recentering servo actuators, is small enough to restrict failure transients to allowable limits.

Logic - DAFCS logic is organized into functional blocks

- o Mode Logic - Logic which interrogates state of engage switches and monitors and sets the DAFCS control mode.
- o Switching Logic - Logic which switches external devices (engage solenoids, engage lights, warning lights, etc.)
- o Outer Loop Logic - Logic which controls outer loop synchronizer states based on DAFCS mode, etc.

The logic is computed at sufficiently high rate to avoid noticeable delays in mode changing or turn-on of annunciator lights. Critical fault reaction logic such as servo disengagement for a sensor or a servo hardover, is integrated into the monitor function, while warning light activation is performed within the Switching Logic CPC.

Gain Scheduling - Gain schedule computation are separated from control law computation because gain schedule computation rate requirements are an order of magnitude lower.

Software Macros - Macros for specific DAFCS functions e.g., lag, and limiter, are selected from the Honeywell Modular Software Catalog. The modules have been thoroughly analyzed and tested, and are thus dependable.

3. FLIGHT SAEFTY VERIFICATION OF THE JA37 FCS

The JA37 PFCS consists of mechanical, electromechanical or hydraulic mechanical components, which either have limited authority or are designed to be safe, i.e., the tandem actuators. Most of the PFCS components were used in earlier versions of 37 Viggen.

The problem for JA37 thus was to verify safe functioning of the 07 DAFCS. To do that a DAFCS Safety Group was organized. The group included members from the Air Force, Honeywell, and SAAB. The responsibility of the safety group was to continuously perform a detailed review of all DAFCS safety related activities and whenever a weak area was found, request improvements. The expected final output from the Safety Group was a written statement declaring that the 07 DAFCS had been reviewed and found safe to fly.

The total 07 DAFCS safety system included:

- o definition and verification of all monitors and BIT
- o required activities to verify stated probability figures.

TAble 6 summarizes the DAFCS monitors and flight safety related activities.

TABLE 6.
07 DAFCS SAFETY SYSTEM

SENSORS
- SINGLE SENSORS, SAFE
- DUAL SENSORS, COMPARISON MONITORS

SERVOS
- SERVO MODELS IN SOFTWARE
- COMPARISON MONITORS

COMPUTER
- SOFTWARE CONTROL
  - STRUCTURE
  - DOCUMENTATION
  - CHANGE PROCEDURES
- HARDWARE FAILURE MONITORING
  - FMEA
  - IN FLIGHT BIT
  - MONITORING EFFECTIVENESS DEMO

CATASTROPHE PROBABILITY CALCULATIONS

NUISANCE DISENGAGEMENT PROBABILITY CALCULATION

VERIFICATION IN RIGSIMULATION AND FLIGHT TEST

The prinicples of the 07 DAFCS sensor and servo monitoring are described in Section 2.1. The verification of correct monitor operation was performed in simulation, rig tests and flight test. After some modifications of the servo models the results were acceptable and the transient failure specidication met.

Development and verification of the 07 DAFCS single channel digital computer flight safety system became the most demanding task. This procedure is described in the following sections.

3.1. Software Documentation, Test, Validation And Change Procedures

The first condition to be met for successful software verification and validation is a well organized program structure. A good way to do this is to divide the program into blocks of suitable size. Each block shall contain its own complete functions and have its own dedicated inputs and outputs as described in section 2.2. of this paper. Each block can then be programmed and checked out independent of other blocks. Finally the blocks are linked together in the assembling procedure.

Software Documentation

The software documentation must be accurate, understandable and maintainable for effective software control and verification. The 07 DAFCS software documentation is summarized in Table 7 and described below.

TABLE 7.

07 DAFCS SOFTWARE DOCUMENTATION

JA37 DAFCS SOFTWARE SPECIFICATION
- MACHROS, SUBROUTINES
- GAIN SCHEDULES
- FUNCTIONAL PROGRAM LISTINGS
- PRE FLIGHT BIT LISTINGS
- MEMORY MAPS

SYSTEM SCHEMATICS
- HW $\longrightarrow$ SW(Z^{-1}) $\longrightarrow$ HW

DAFCS SYSTEM TEST PROCEDURES
- HARDWARE ACCEPTANCE TEST
  - HDC-301 SELF TEST
- SOFTWARE ACCEPTANCE TEST
  - FREQUENCY RESPONSE
  - NONLINEARITIES TEST
  - LOGIC REASONABLENESS TEST
  - MONITOR TRIP LEVELS ETC
  - PROGRAM EXECUTION TIME
  - OVER FLOW CONTROL
- POST INSTALLATION TESTS
  - END TO END A/C - FCS
- PRE FLIGHT BIT

The DAFCS documentation includes:

- o Software specification, including source program listing
- o System schematics
- o System test procedures

The DAFCS software specification includes a directory of system software documentation as well as software description. The DAFCS specification is computer edited and printed for ease of maintenance.

The source program listing is obtained as output from the assembly process during generation of the sequence of program instructions and data.

Annotations included from part of the program include

- o Tabulation of program constants and variables including definition and scaling
- o Documentation of computation flow where graphical flow charting is inappropriate

The source program format is thus important to make the program listing as clear, concise, meaningful, and informative as possible, and to provide portions of basic documentation.

The data and variable lists in the program listing serves as the documentation for program variables; therefore, the list is well organized and labeled. Variables and constants are organized by CPC in alphabetical order. The various CPC variable (and constant) lists are ordered, as are the instruction sets and the software specification CPC descriptions. Indexed variables have separate labels. The description of data items as appropriate, include:

- o Label
- o Definition
- o Numerical value
- o Scaling
- o Dimensions

Variable subscripts are related to CPC to avoid duplication and to promote ease of identification.

System Schematics

A graphical description of DAFCS computer program computations are incorporated into the system schematics. Control laws are depicted in block diagram format, as shown in Figure 5. Logic is represented in flow chart format. Combined flow chart/block diagram depiction is used where appropriate, e.g., monitoring representation.

System schematics are organized functionally, as are the CPC's. Heading and comments included on system schematics are consistent with the software specification and source program listing comments.

Flow charts must depict the system functional operation, as well as serve as guides to follow program operations in the listing.

Control Law Representation - A sample control law schematic is presented in Figure 5. The control law schematic contains the end-to-end computation; e.g., sensor input to servo command.

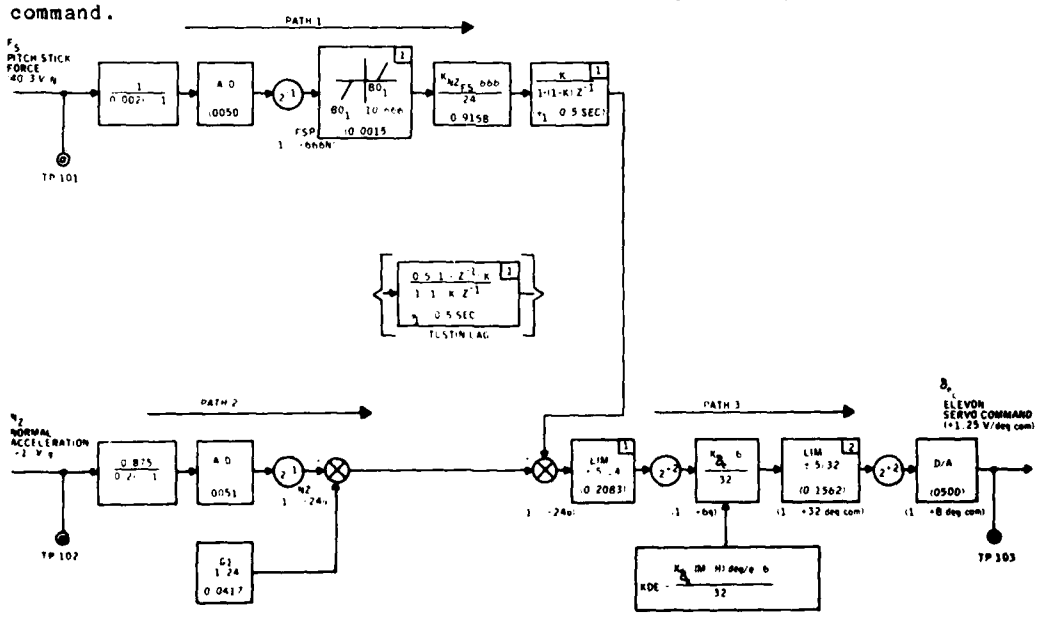


Figure 5. Control Law Schematics

DAFCS Test Procedures include the following:

Hardware Acceptance Test Procedures (ATP) - Hardware Acceptance Test Procedures are automated procedures which check each device included in the system. Hardware ATP is performed on each deliverable unit, and redone after hardware modification.

Software Acceptance Test Procedures - Software Acceptance Test Procedures are manual test procedures that theoretically test every functional characteristic of the DAFCS software that is important for satisfactory performance and safety. There tests are performed in a test station with computer hardware and simulated input/output. Characteristics that are tested include:

- o Control Laws
  - End-end frequency response characteristics for each signal path (gain, phase versus frequency)
  - Designed non-linearities (break-outs, limits)
  - Critical signal deadband, resolution
- o Logic. Each logic decision is exercised by the procedures. "Reasonable" combinations of logic inputs states are checked for proper output.
- o Monitoring
  - Monitor trip levels
  - Monitor trip time (time from fault introduction until completion of critical fault reaction).
  - Proper fault reaction (e.g., disengage, light warning light).
- o Gain Scheduling. Gain variations with flight condition (e.g., do gain versus Mach, altitude).
- o Program Timing. Programming timing is controlled by analysis and test. The DAFCS program has been analyzed to establish computa tion time as a function of logic input variable states:
  - Control law computation time as function of mode
  - Logic computation time as a function of mode, monitor states, etc.
  - Monitor computation time as a function of monitored device state

  The analysis ensures that allowed computation time limits are not exceeded with combinations of computation time maximums that can be reasonably expected to occur.
- o Overflow - Overflow is controlled by analysis and test. The program was analyzed and the maximum attainable signal level at each sum point, shift, division operation, lag or limiter input determined. These values were compared to overflow level at the various points. At any point where overflow is physically possible, the conditions for overflow are noted i.e., indicate combination of input signals required to produce overflow, and why this combination will not occur. The analysis is documented as part of the software specification and periodically updated to reflect software changes.

Procedures are organized by CPC.

The Software ATP was performed in its entirety on a prototype DAFCS computer unit. The complete ATP need not be repeated on subsequent units that can be shown to have identical memory via memory sum check, and have passed hardware ATP. After a program change, the Software ATP is performed on modified CPC's to the extent necessary to maintain quality assurance.

Post Installation Test (PIT) Procedures - Post Installation Test Procedures are procedures used to verify the installed DAFCS. These procedures test all system interfaces. Each sensor and discrete signal source is exercised and proper system response verified by measurements. Each driven device is also exercised.

System functional characteristics not previously verified in ATP are verified in Post Installation Test. Each developmental unit is subjected to PIT after initial installation. An abbreviated PIT is performed after re-installation following system modification.

Preflight Test Procedures - Preflight procedures comprise execution of DAFCS flight line BIT.

The above defined software documentation was modified several times during the 07 DAFCS D&D phase. Today it exists in its final form defining the 07 DAFCS series production system.

Software Verification and Validation

Verification of the DAFCS software was a continuously on going procedure that started with debugging of the program later followed by the software acceptance test procedures and the postinstallation and preflight tests. Applicable parts of these software tests were performed after any software modification. Extensive closed loop rig and flight tests were performed in addition to the dedicated software tests. The rig and flight tests included a variety of tasks from simple transient responce checks to performing complete missions. Several tasks included exercising the functional logic and the redundancy management.

The final phaseof the software verification included complete and several partial software acceptance test procedures, all performed successfully. It also included may post installation and pre flight tests in rigs and in prototype and series aircraft. These tests and several hundred successful rig and flight hours together formed the documentation on which the software validation finally was constituted.

Software Change Procedures

The D&D Phase - The change procedure was implemented after initial ATP of the DAFCS operational computer program.

A DAFCS functional change requirement was identified, and formulated in an informal manner (e.g., memo, mod request, etc.). A Working Program (WP), or uncontrolled version of the DAFCS Master Program (MP) was modified and assembled to produce a paper tape for loading into the DAFCS Software Development Unit. The modification was handpatched into the memory of the Software Development Unit if feasible. Informal request, and use of Working Program was employed to minimize paper work in the early stage of development and better exploit the flexibility of digital mechanization.

The modification was developed by the programmer in the Software Development Unit until hs was satisfied that the modification will satisfy the requirement. The resulting paper tape was loaded into a prototype system. Proper loas was verified by sum check specificaion.

Appropriate Software ATP tests were performed by the systems engineer on one prototype system, and limited safety-oriented PIT and Preflight tests performed on each system after installation in its respective aircraft.

The test results were reviewed by those responsible. After performance of any automatic preflight test, each changed system flew.

The Series Production Phase - After the software validation, during the 07 DAFCS series production pahse the software change procedures had to be more stringent. There were a couple of reasons for this:

1. No errors can be tolerated in the series production software.
2. More systems were around and more people ivolved in handling them.

The 07 DAFCS software change procedures during the 07 DAFCS and JA37 aircraft production phase are shown in flow diagram form in Figure 6.

When a problem is defined during the 07 DAFCS series production or when installing the systems in the JA37 aircraft at SAAB or during field service at an air base the following questions have to answered and actions have to be taken according to Figure 6.

- Is the problem caused by wrong handling or probably by maintenance error? If not, there is a functional error in the aircraft.
- Is there a preflight test error?
- If not, there is an 07 DAFCS problem and a preliminary engineering change notice (ECN) that has to be written and approved by the Air Force and SAAB. This decision also provides funds for further action.
- A working program (WP) is issued and desired software changes are implemented. Hardware modifications are implemented in 07 DAFCS rig or systems if required.
- After post installation tests, rig and/or flight tests are performed to verify that the software (and hardware) modification eliminated the problem.
- When the problem is resolved a final FCN is issued and approved. The approval also defines how to fund the series production software (and hardware) changes.
- The series production 07 DAFCS software documentation and test procedures are updated and the final issue of the software working program is becoming the new master program. Hardware production is changed if required.

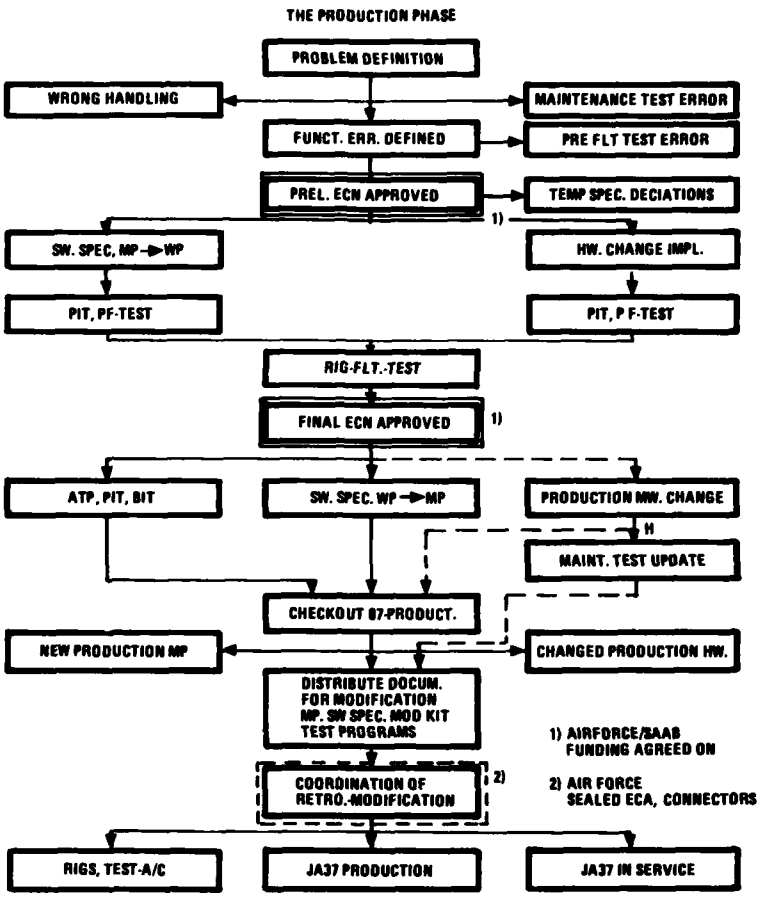


Figure 6. 07 DAFCS Software Change Procedures

- o The updated series production documentation is now ready for distribution. It includes the new master program and modification kits if hardware changes are involved.
- o Coordination and control of all retroactive modifications at SAAB and at the airbases is executed by the Air Force personnel. The 07 DAFCS computer box and its connectors are sealed by the Air Force after installation and test in the series production aircrafts. No seals can thereafter be broken without approval of the Air Force 07 DAFCS maintenance group.

3.2. Hardware FMEA, In Flight BIT And Safety Verification

The single digital computer approach required extensive safety effort in order to verify that the in flight computer selftest, BIT, was sufficient enough to detect all flight safety critical failures.

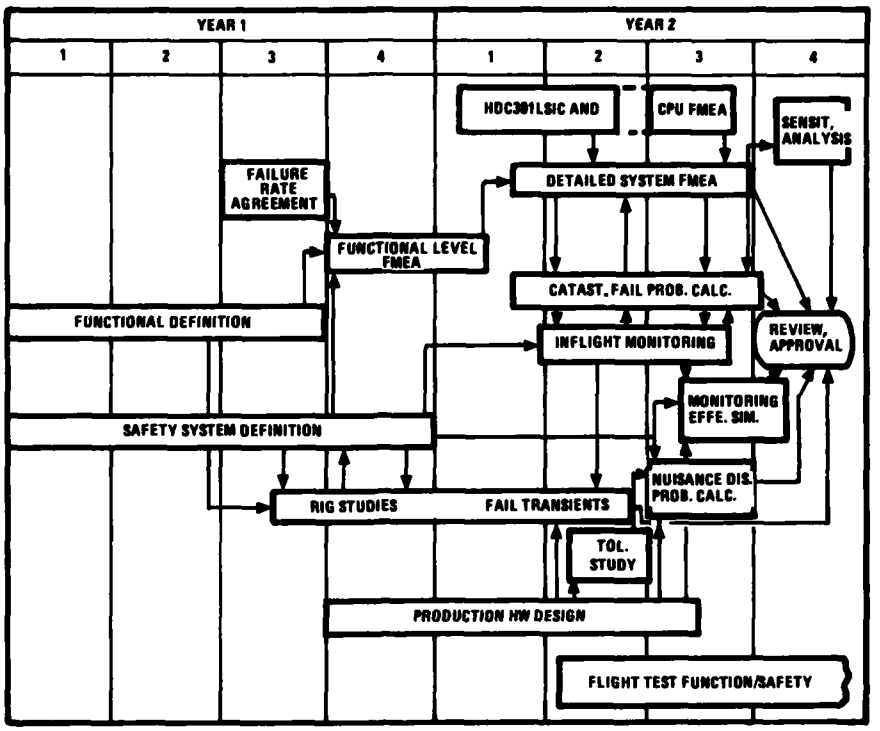
An 07 DAFCS hardware safety verification plan was establised and followed. The plan is shown in Figure 7.

The plan covers two calendar years and starts with functional and safety system definition.

Hardware FMEA. When the 07 DAFCS was defined and system schematics and card layouts became available the Failure Mode Effect Analysis (FMEA) was started. The FMEA was performed at functional and detailed level and involved all 07 DAFCS components. The main effort was spent on the digital computer, especially the HDC 301 CPU and its LSI circuits were anlayzed in detail.

The objecitves of the 07 DAFCS FMEA are listed in TAble 8.

During the first part of the FMEA the identified failure modes were divided into flight safety critical and not critical failure modes. A majority of the failure modes were found to be flight safety critical. The CPU in particular has a great number and



FMEA AND PROBABILITY CALCULATIONS BY HONEYWELL.
RIG - AND FLIGHT TEST BY SAAB.
ALL EFFORTS CHECKED BY A SAFETY GROUP.
SAFETY GROUP: 1 AIRFORCE, 2 SAAB, 1 HONEYWELL.
SAFETY GROUP MUST APPROVE OR DISAPPROVE!

Figure 7. 07 DAFCS Hardware Safety Verification Plan

TABLE 8.

07 DAFCS FMEA OBJECTIVES

- o ESTABLISH THE DAFCS FAILURE MODES AND DETERMINE THE FAILURE RATES ASSOCIATED WITH THE IDENTIFIED FAILURES
- o IDENTIFY THE INFLIGHT MONITOR THAT WILL DETECT EACH IDENTIFIED DAFCS FAILURE
- o DETERMINE IF PRE-FLT TEST WILL DETECT EACH IDENTIFIED FAILURE
- o INFLUENCE THE BASIC DESIGN BY STRIVING TO OPTIMIZE MARGINS IN SAFETY, RELIABILITY, FAULT ISOLATION/ DETECTION, AND NUISANCE DISENGAGEMENTS.

HDC 301 CPU FMEA

- o DETERMINE FAULT DETECTION EFFECTIVENESS OF CPU SELF TEST PROGRAM
- o SELF TEST ROUTINE OPTIMIZATION
- o IDENTIFICATION OF FAILURE MODES UNDETECTED·BY CPU SELF TEST
- o DETERMINE FAULT DETECTION EFFECTIVENESS OF OTHER MONITOR FUNCTIONS AGAINST MODES NOT DETECTED BY SELF TEST.

contributes about 90 percent of all critical failure modes. When all critical failure modes were identified the next step was to verify that all failures were detected by either a software monitor or by some of the dedicated digital computer selftests run as part of the in flight BIT. This effort led to development of new computer self tests that had to be included in the BIT. Several hardware modifications were also introduced.

The results of the FMEA are summarized in Table 9.

TABLE 9.

07 DAFCS FMEA

RESULTS

- CONDUCTED IN TWO STAGES - THE 1ST AT FUNCTIONAL LEVEL, THE 2ND A DETAILED LEVEL FMEA.
- DAFCS DIVIDED INTO 56 FUNCTIONS - 42 WITHIN THE ECA AND 14 EXTERNAL.
- 1775 ECA PEICE PARTS ANALYZED AND 3584 TOTAL ECA FAILURE MODES INVESTIGATED.
- 13 HARDWARE CHANGES AND 15 SOFTWARE CHANGES MADE TO SYSTEM
- ALL FAILURE MODES THAT WOULD RESULT IN A CATASTROPHIC FAILURE WILL BE IDENTIFIED BY INFLIGHT MONITORS

HDC 301 CPU FMEA

- SELF TEST PROGRAM OF 301 PROCESSOR OPTIMIZED
- 1230 FAILURE MODES TESTED WITH 100% EFFECTIVENESS
- NO KNOWN POTENTIALLY CATASTROPHIC 301 FAILURES EXIST THAT ARE NOT DETECTED

In Flight BIT - As mentioned above the in flight BIT was continuously improved during the FMEA. When the 07 DAFCS safety verification program was finished the BIT included 13 dedicated digital computer tests. The BIT efficiency, or coverage, had to be about 99.95 percent in order to meet the stated requirement on probability of catastrophe.

Table 10 illustrates why the high BIT coverage is required, it also lists the 13 parts of the in flight BIT.

Most of the tests or monitors included in the 07 DAFCS in flight BIT are described in the paper entitled "Computer based in flight monitoring" presented as part of this lecture series. Some of the selftests are more or less tailored for the 07 DAFCS digital computer architecture. These tests are described below.

1. The Dynamic Computation Monitor (DCM) is monitoring the entire digital computer including the A/D and D/A converters. The philosophy is that the (DCM) shall provide a simplified analog model of the A/C dynamics. That model shall then be controlled by control laws and logic located in the digital computer in a similar way as the aircraft is controlled. The output of the model is checked regarding magnitude and frequency (time) and a failure announced if the requirements are not met. Figure 8 shows the 07 DAFCS DCM.

   The analog dynamic model is simply an integrator provided a voltage input x from the digital computer. X is a 20 Hz square wave of 5 volts magnitude. The part of a cycle the square ware is positive (and negative) is controlled by the computer software. The output Y from the integrator is a triangular wave. Y is fed back to the digital computer and compared to the computer output X. The difference signal is manipulated by a chain of instructions aimed for exercise most of the CPU. The part of a cycle the output X shall be positive is determined in the final step before X is D/A converted. The output Y of the integrator is thus controlled by the computer program. The output peak voltage must be between 4.2 and 5.8 volt otherwise a computer (or DCM) failure will be announced after a slight delay and the 07 DAFCS shut down.

2. The memory sum checks are separated in 3 different blocks. This is more a matter of practicality than an answer to a technical problem. The so called scratch pad memory (SPAD), where some data and temporarily results are stored, are split into two redundant areas. The same data are stored in each area. The

TABLE 10.

07 DAFCS COMPUTER MONITORING

REQUIRED COMPUTER IN FLIGHT BIT COVERAGE P_C%

$$P_C \sim \left(1 - \frac{10\text{-}7}{200 \times 10\text{-}6}\right) \sim 99.95\% \text{ OF CRITICAL FAILURES}$$

PREFLIGHT TEST OF ALL MONITORS AND IN FLT BIT.

REQUIRED IN FLIGHT BIT

1 CPU SELF TEST
EXERCISE INSTRUCTION REPERTOIRE, (~ 200 WRDS)

2 REAL TIME CLOCK MONITOR
RESETABLE 12.5 MSEC HW. COUNTER

3 DYNAMIC COMPUTATION MONITOR
COMPARES DIG. AND AN. ARITHM. & INTEGRATION

4 CONTINUITY MONITOR
SW. CHECK PROGRAM STRUCTURE FOLLOWED

5 WATCH DOG TIMER
HW. TIMER, DISENGAGE AFTER 1.5 x 12.5 MSEC

6 MEMORY PARITY ERROR DETECTION

7 CRITICAL INSTRUCTION MEMORY SUM CHECKS

8 CRITICAL CONSTANT MEMORY SUM CHECKS

9 CRITICAL SPAD (DUAL) MEMORY SUM CHECKS

10 DAFCS GAIN MONITORS
SW. CHECK OF SIGN, MAGNITUDE ETC.

11 I/O READY MONITOR
SW. CHECK IF I/O STUCK BUSY

12 I/O ANALOG AND DISCRETE WRAPAROUNDS

13 ELECTRICAL VOLTAGE CHECKS (6)

SPAD contents are then added together for each area and the results compared contents and the memory sums change and are never known outside the computer.

3. The 07 DAFCS gain schedules are fairly complex and large gains may drive some loops unstable. Some computed gains therefore have to be monitored. The tests are of reasonableness type such as check of gain sign and magnitude.

4. The 07 DAFCS computer in/out systems is tested by the DCM and also by a couple of additional tests. The I/O ready monitor checks that the data transferred by the I/O circuits cages as a function of time. If it does not a failure is announced.

The analog and discrete wrap around test also checks the I/O system. In this test analog and discrete digital computer outputs are stored in the computer memory. The outputs, for analog signals after D/A conversion, are fed back or "wrapped around" to the computer input circuits. The analog signals are A/D converted before they are compared to the original outputs stored in the computer memory. If the comparison doesn't agree a failure exists.

Hardware safety verification - The 07 DAFCS hardware safety program shown in the figure 7 verification plan included several other activities in addition to the FMEA and optimization of the monitors and inflight BIT.

One major effort was to verify and demonstrate that all 07 DAFCS critical hardware failures were detected by the monitors and the inflight BIT. The demonstration was performed by anaysis, rigs, simulations and flight test.

The anaysis included failure probability calculations of catastrophe and nuisance

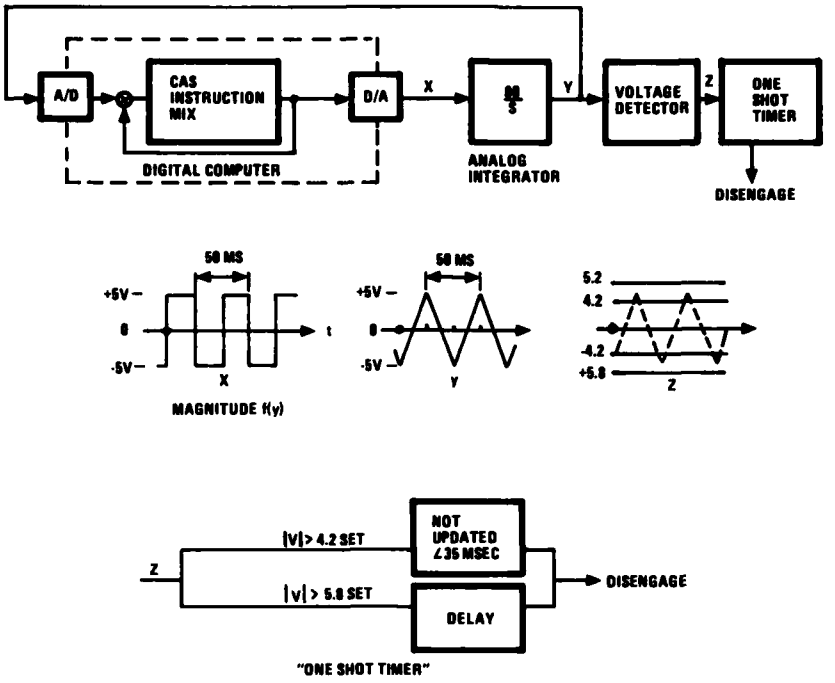


Figure 8. Dynamic Computation Monitor

disengagement of the 07 DAFCS. In addition to the probability of a catastrophe calculations a senstivity study was performed some of the more significant failure rates were varied. The nuisance probability calculations considered the combined effect of hardware tolerances and full scale signals.

Open and closed loop rig simulations were performed to verify that the 07 DAFCS monitors and BIT detected all simulated failures. The rigs contained hardware prototypes of the 07 DAFCS digital computer and PFCS components. Failure transients were recorded in the closed loop rig simulations.

The most critical failure transients were later repeated in flight test to insure that the failure transient specification was met.

3.3. DAFCS Monitoring System

The FMEA, monitor and inflight BIT optimization performed within the safety verification program resulted in the 07 DAFCS safely system, which was further rig and fight tested. Some minor modfications were introduced during that period but basically the 07 DAFCS safety system remained unchaged. The 07 DAFCS safety system now in series production is shown in the Figure 9 block diagram.

Figure 9 shows that the Auto-Throttle Control System (ATCS) sensrs, miscellaneous logic inputs, the IME anc CC are single and thus not flight safety critical. The ATCS servo is also single and safe. Figure 9 further illustrates that:

- o Critical sensors are dual.
- o The ADC is monitored by comparison to approximate altitude and speed information from the separate pressure controller PFCS pitch gearing.
- o The digital computer is monitored by BIT programs stored in the computer memory.
- o The external analog DCM hardware and the DCM computer software also monitors the digital computer.
- o The servos are monitored by compairson to servo models in software.
- o The watch dog timer (WDT) both checks that the computer is running and is used to disengage the servos if the computer fails (goes into "fail loop").
- o Sensor and servo failures disengage the servos by discrete logic signals.

3.4. Safety Verification Results

At the end of the 07 DAFCS safety verification program a major safety design review was held. The joint Safety Group assisted by several specialists reviewed the performed safety tasks and summarized the results. At that time preliminary rig and flight test results were available.

The results are in condensed form restated in Table 11.

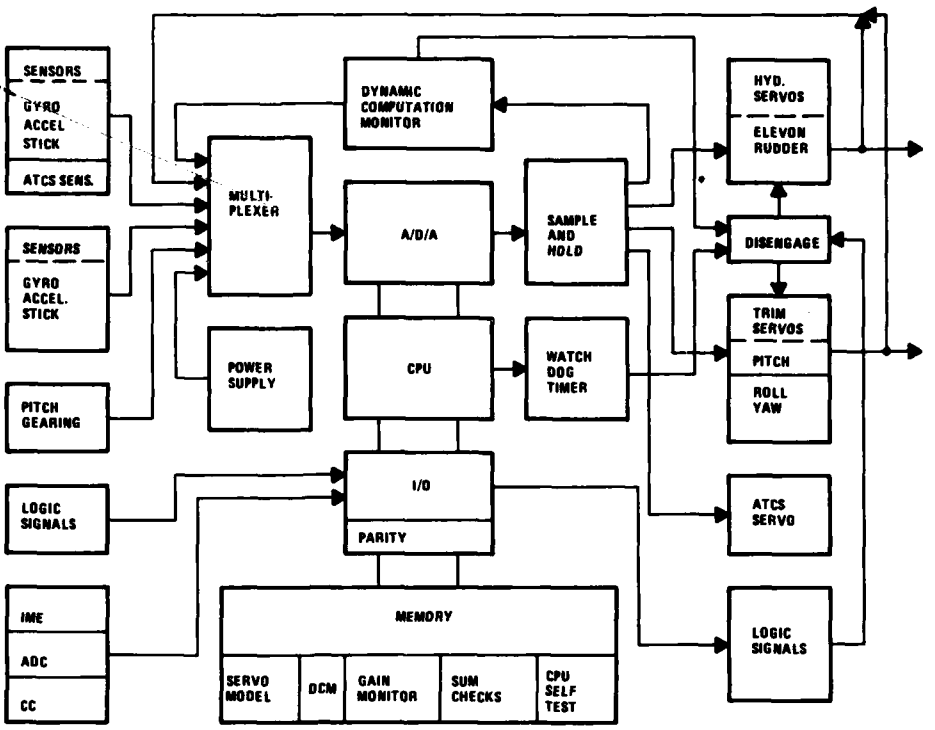


Figure 9. DAFCS System Block Diagram

TABLE 11.

07 DAFCS SAFETY SYSTEM RESULTS

SENSORS, SERVOS, COMPUTER —— OPTIMIZED MONITORING

FMEA COMPLETED ACCORDING TO PLAN
INFLIGHT BIT OPTIMIZED

FAILURE TRANSIENTS

SIMULATION AND FLIGHT TEST VERIFIED ALL FAILURE TRANSIENTS WITHIN SPEC.

PROBABILITY OF CATASTROPHE

1. PROBABILITY OF FAILURE REQ. DISENG. $474 \cdot 10^{-6}$
2. PROBABILITY OF NO DISENGAGEMENT $24 \cdot 10^{-6}$

1x2. PROBABILITY OF FAILURE & NO DISENG. $0.011 \cdot 10^{-6}$

3. PROBABILITY OF UNDETECTED FAILURE $0.068 \cdot 10^{-6}$

PROBABILITY OF CATASTROPHE

$0.079 \cdot 10^{-6}$ PER FLT HR

PROBABILITY OF NUISANCE DISENGAGEMENT

OPTIMIZED MONITOR THRESHOLDS AND LAGS

THRESHOLDS AT $\sim 4.5\,\sigma$ TOL. AND MAX SIGNALS

PROBABILITY OF NUISANCE

$0.39 \cdot 10^{-3}$ PER FLT HR

Based on the effort performed and achieved results the Safety Group concluded that "the single channel dgital computer 07 DAFCS was safe to fly."

Up to this point the FCS-rig and the FCS test aircraft 07 DAFCS prototypes had included dual channel dgital computers and comparison monitors. The series production documentation had also been prepared for dual computers. After the decision to go single computer one of the redundant computers in the two prototypes was disabled and the series production documentation adapted to the single computer configuration.

Results of the D&D Phase-D&D phase is shown in Table 12. Suitable marginals exists as can be seen.

TABLE 12.
07 DAFCS MEMORY - AND COMPUTATIONAL - LOAD

THE SERIES PRODUCTION 07 DAFCS AFTER THE D & D PHASE.

	MEMORY K-WORDS	IN FLIGHT COMP.-LOAD
CONTROL LAWS, LOGIC	2.8	40%
IN FLIGHT MONITORING	0.9	32%
PRE FLIGHT BIT	1.7	
EXECUTIVES & UTILITY	0.8	1%
	6.2	
SPARE	25%	27%

The results of many hundred fight hours with 3 tests aircraft equipped wth 07 DAFCS prototypes have verified the safety verification program results. The results from the D&D phase can be commented as follows:

- o The software control procedures worked fine.
- o It was possible to meet both the failure transient and the nuisance requirements.
- o The FMEA became extensive and time consuming. Several tests had to be added to the BIT as a result of the FMEA.
- o The monitoring effectiveness demonstration for the 07 computer and HDC 301 became a major task.
- o The probability calculations caused little trouble once the presumptions were agreed on.

Experience from the series production - Table 13 summarizes the results from about 1000 flight hours with 07 DAFCS series production systems.

4. CONCLUDING REMARKS

The 07 DAFCS development program verified that the single digital computer configuration is safe to fly. Flight test results from prototype and series production aircraft confirms the results from the safety verification program.

Extensive maintenance BIT has been developed and is operational in the series production systems. Thus the expected benefits of the digital mechanization have been encuntered. The effort spend on the FMEA and development of computer BIT was justified considering the cost of arborne digital coputers at that time.

5. REFERENCES

1. Software Control Procedures for the JA 37 Digital Automatic Flight Control Systems, by D. G. Bailey, Honeywel Inc. and K. Folkesson SAAB-SCANIA.

TABLE 13.

EXPERIENCE FROM THE JA37/07 DAFCS PROGRAM

SERIES PRODUCTION

07 DAFCS UNITS	NUMBER OF FAILURES	TYPE OF FAILURE
ECA	1	LOW VOLTAGE
RATE GYRO PITCH	1	NULL OFF. OUT OF TOL.
N_Z-ACCELEROMETER	1	TEMPORARY HANG UP
RUDDER SERIES SERVO	1	TIGHT MONIT.THRESH,ON GROUND
UNDEFINED	1	TEMPORARY ROLL ATT DISTURB.

CONCLUSION

SO FAR NO DIGITAL COMPUTER FAILURE HAS OCCURRED

THE INFLIGHT DIGITAL COMPUTER SELF TEST HAS NOT CAUSED ANY NUISANCE DISENGAGEMENT

THE CONTROL ROUTINES ESTABLISHED FOR IMPLEMENTATION OF NEW SOFTWARE EDITIONS INTO THE SERIES PRODUCTION SYSTEMS ARE IN OPERATION AND APPRECIATED

FLIGHT EXPERIENCE WITH FLIGHT CONTROL REDUNDANCY MANAGEMENT

Kenneth J. Szalai, Richard R. Larson, and Richard D. Glover
NASA Dryden Flight Research Center
P.O. Box 273
Edwards, California 93523
U. S. A.

SUMMARY

Flight experience with both current and advanced redundancy management schemes has been gained in recent NASA flight research programs using the F-8 digital fly-by-wire aircraft. This paper reviews the flight performance of fault detection, isolation, and reconfiguration (FDIR) methods for sensors, computers, and actuators. Results of induced failures as well as of actual random failures are discussed. Deficiencies in modeling and implementation techniques are also discussed. The paper also presents comparisons of multisensor tracking in smooth air, in turbulence, during large maneuvers, and during maneuvers typical of those of large commercial transport aircraft. The results of flight tests of an advanced analytic redundancy management algorithm are compared with the performance of a contemporary algorithm in terms of time to detection, false alarms, and missed alarms. The performance of computer redundancy management in both iron bird and flight tests is also presented.

1.0 INTRODUCTION

Multichannel digital flight control has evolved from research and demonstration (Refs. 1 to 6) to application in the space shuttle (Ref. 7), in commercial transports (Ref. 8), in prototype aircraft (Ref. 9), and in production military fighters (Ref. 10). Even full authority digital fly-by-wire (DFBW) control without mechanical reversion has been achieved: this approach has been used on the F-8 digital fly-by-wire aircraft and the space shuttle orbiter.

Management of a multicomputer/sensor/actuator/power system is markedly different and significantly more complex than managing a single channel system with a number of backup channels. This is because the multichannel system must perform functionally as a single channel system and as completely independent channels in order to accommodate faults. This fundamental conflict is the basis of much analytical and experimental research for the derivation of software and hardware methods that provide superior performance and high levels of reliability.

NASA has been conducting research in aircraft digital fly-by-wire control at the Dryden, Langley, Ames, Lewis, and Johnson centers. The primary objective of this work is the development of highly reliable flight control systems that permit the implementation of active control functions, which contribute to energy efficiency and high maneuverability. A major flight experiment at the Dryden Flight Research Center involved the use of a Navy F-8C testbed aircraft that was modified to incorporate full authority digital fly-by-wire control systems. The vehicle has a triplex fail-operational digital fly-by-wire primary flight control system and an analog emergency backup control system (Refs. 1 to 3).

Flight experience has been gained with both current and advanced redundancy management schemes during the 80-flight program. This paper reviews the actual flight performance of digital fly-by-wire fault detection, isolation, and reconfiguration (FDIR) methods and compares the results with design goals, models, and expectations. The implications for future designs are also discussed.

2.0 NASA F-8 DIGITAL FLY-BY-WIRE PROGRAM

The specific objective of the F-8 DFBW program was to generate a validated data base for the design of future fly-by-wire flight control systems. From 1972 to 1973, the Dryden Flight Research Center flight tested a single channel DFBW control system in an F-8C aircraft (Fig. 1). The airplane was equipped with hardware developed for the Apollo lunar module control system and, from its first flight, was flown with the basic mechanical linkage control system removed. The program successfully demonstrated the feasibility of DFBW control for aircraft (Ref. 3), but it did not address multichannel redundancy management issues.

The second phase of the program, which has been flown since 1976, involves an experimental triplex DFBW control system that was designed to be both a research tool and the primary flight control system of the aircraft. Key elements of the research program were hardware and software redundancy management as well as the implementation of advanced control laws. In terms of flight control, the system was designed to be fail operational, full time, and full authority. Thus, the overall design and operational requirements for the system were inherently realistic as well as being representative of the ultimate test of the system—manned flight without the capability of reverting to a mechanical control system. The generic requirements for the system (which are listed in Table 1) were made taxing, yet realistic. Specific tests were performed to validate various system concepts and the design methodology. In 4 years of operation, incidental and implicit results have also been derived. Both are included in this paper.

3.0 F-8 DIGITAL FLY-BY-WIRE SYSTEM DESCRIPTION

3.1 Overall Mechanization

The overall mechanization of the F-8 DFBW control system is shown in Figure 2. A triplex digital computer set containing control law and system redundancy management software communicates with a specially designed interface unit (IFU). The IFU processes input data, which consist of pilot commands and aircraft sensor signals, and output data, which consist of surface commands, cockpit displays, and telemetry data. Surface commands

are routed through a switching mechanism to the servodrive electronics and then to the force-summed secondary actuators, which are installed in series with the existing F-8C power actuators. There are five actuator sets: one for each aileron and horizontal stabilizer surface and one for the rudder.

The triplex analog computer bypass system provides the pilot with an emergency unaugmented command path to the control surfaces in the event of a total primary digital system failure. This path was provided primarily to protect against a common-mode software failure in the infant stages of flight test. The switching mechanism allows either the primary system or the bypass system to drive the secondary actuators based on pilot selection or automatically, according to fault status.

Electrical power is provided to three independent flight control buses by an engine-driven dc generator. Each bus is protected by a 40-ampere-hour battery, which would allow approximately 90 minutes of operation in the event of a loss of generator power. Secondary actuator hydraulic power is provided by the aircraft's three hydraulic systems, each of which supplies one of the triple chambers of each actuator.

3.2 Primary Digital System Mechanization

A functional block diagram of the primary digital system is shown in Figure 3. The three channels (A, B, C) are identical. A variety of motion sensors and switch discretes is used. Analog sensors are converted through a 12-bit analog-to-digital converter. Each channel conditions and converts sensor data associated with that channel. The data are placed in the buffer memory of that channel and in the buffer memories of the other two channels by way of the serial data buses. Thus, each computer contains an identical set of redundant sensor data. Sensor redundancy management is performed in software, and the selected sensor signals are used in the control laws to compute surface command signals. The serial data buses are also used by the computers to transmit and receive status information for computer redundancy management. The intercomputer discretes are used to synchronize the computers.

The interface unit output consists of surface position commands to the horizontal stabilizer, ailerons, flaps (symmetric ailerons), and rudder; discretes to the cockpit panels; and telemetry data. The panel discretes are transmitted serially to the encoder/decoder for distribution within the cockpit. The serial telemetry data, which consist of internal digital computer data, are routed to an onboard tape recorder. Surface commands are generated through a 12-bit digital-to-analog converter to an electronic switch. When the primary digital system is operational, control surface commands are passed directly through the switch to midvalue select logic that is mechanized in hardware. There is a midvalue select circuit for each of the five secondary actuators. Comparators placed around these select circuits are used to detect and isolate servo channel faults.

The secondary actuators use high-gain two-stage servovalves to control hydraulic pressure of 2.07×10^7 N/m^2 (3000 lb/in^2) across each of three pistons. Force summing occurs along the common output shaft, which is mechanically linked to the metering valve of the existing dual-tandem F-8C power actuators.

3.3 Digital Computer Characteristics

The computers used in the digital flight control system are general purpose, stored-program machines. They contain two sets of eight fixed-point general registers and eight registers for hardware floating point operations. Detailed computer characteristics are listed in Table 2.

3.4 Flight Control Sensors

Table 3 lists the flight control sensors used by the primary DFBW system, their redundancy level, and the signal type. Rate gyros and accelerometers are nearly collocated on a rigid base approximately 0.3 meter on a side.

3.5 Software Sequence

The time-sequential operation of the system is illustrated in Figure 4(a). The accompanying memory allocation is shown in Figure 4(b). One complete cycle represents the minor cycle, or shortest iteration period, which is 20 milliseconds. A computer clock interrupt starts the minor cycle. Computer synchronization occurs first. An exchange of status information occurs in the crosslink routine, which is part of the computer redundancy management process. The executive scheduler sets the interrupt time to initiate the next 20-millisecond cycle.

Sensor and discrete data are read in the input routine. A complete set of redundant sensor information is available in each computer at this time. The sensor selection routine produces a single sensor value for each set for use in the control law program. The first portion of the control law program contains only those computations necessary to produce the actuator command outputs. These commands are sent to the digital-to-analog converter hardware approximately 5.5 milliseconds after the sensor data are read. The second part of the control law program contains filter equation updates, gain scheduling, moding and display logic, and other computations insensitive to sequence. Sensor fault detection, isolation, and reconfiguration are performed on the basis of comparisons between sensor differences and predetermined fault threshold values.

A data telemetry program sends 1000 32-bit words per second to an onboard tape recorder for postflight processing. The remainder of the minor cycle is used for executing the computer self-test routine, which includes both central processor and memory validity tests.

4.0 COMPUTER REDUNDANCY MANAGEMENT

4.1 Computer Fault Detection, Isolation, and Reconfiguration

The computer fault detection, isolation, and reconfiguration hierarchy is shown in Figure 5. The heart of the approach is the restart process. A restart is an online reinitialization of a flight control system channel that has

suffered a serious fault. A restart is performed in an attempt to clear the problem so that normal operation may be resumed.

The restart process is initiated either indirectly, by an abnormal condition resulting from a hardware or software fault in the interface unit or computer, or directly, by the built-in test equipment (BITE) in the computer. Indirectly caused restarts result from the detection of a manifestation of an element fault rather than from the detection of the fault itself. This is a key factor in the design. It is based on the premise that serious problems will eventually manifest themselves in abnormalities in the synchronization and data exchange (crosslink) processes.

Two outcomes are possible from a restart attempt. Should normal operation be achieved, the program continues. Should several restarts fail to restore the system to normal operation, the offending channel is declared permanently failed, and is removed from the voting set. Certain BITE outputs from the interface unit and computer are known to produce unrecoverable or no-confidence states. These BITE signals are routed directly to the channel fail logic, which is a hybrid hardware/software system.

Table 4 lists recoverable and nonrecoverable conditions and the corresponding detection mechanisms. System status miscompare software tests are performed on data that are exchanged on the serial crosslink. These data include such items as flight control system mode and cycle count, which indicates synchronization status.

The implementation of the channel fault designation is shown in Figure 6. Self-detected faults or agreement on a fault by the other two partners leads to a channel failure. A failed channel cannot participate in a fail vote for another channel. There is overlap in the fault detection tests: a given fault may trigger detection mechanisms in several tests.

4.2 Computer Synchronization

To ensure that each processor handles input and crosslink data at the same two points in the computation cycle, the computers are software synchronized to begin each 20-millisecond cycle within 50 microseconds of each other. No other synchronization points exist within the computation cycle. Output commands are sent independently by each channel when they are available. Because of the close tolerance on the synchronization point at the beginning of each cycle and the particular minor cycle structure used, output synchronization is unnecessary. The close synchronization also provides an excellent measure of channel health and hence becomes an important element in computer fault detection.

Every 20 milliseconds an internal computer clock interrupt occurs and the computer issues a discrete high signal to each of the other computers. The computer then reads the discretes it has received from the other computers. If discretes are present from both computers, the discrete is reset, and a second read is performed to ensure that the other computers have also reset their discretes. This process accomplishes synchronization. The computer clock is then reset to interrupt at the next cycle time. If, after a short wait to allow for skew between processors, one computer fails to synchronize with the other two, the two remaining computers exit the synchronization program and continue normal processing.

This synchronization scheme results in the reading of sensor data with less than 50 microseconds of skew between computers. The exchange of sensor data following synchronization results in bit-identical data in each computer. This means that under unfailed conditions, the output commands of the three computers are bit identical as well.

4.3 Fault Tolerance Predictions

Theoretical reliability analyses (Ref. 11) were used in the initial trade studies to determine the overall configuration. At this stage of the design, mechanization details were unknown, so a number of assumptions had to be made: (a) a digital channel would have a failure rate of from 10^{-3} to 10^{-4} per hour; (b) a triplex analog backup system would have a failure rate of 4.5×10^{-4} per hour; (c) the failure rate of the aircraft's hydraulic system would not be affected by the DFBW system; (d) mission time would be 1 hour; (e) the probability of correct fault detection, isolation, and reconfiguration would be 1.0 with three channels operating; (f) at least two analog channels would have to be operating for controlled flight; and (g) the digital and analog systems would be independent.

The form of the analysis performed prior to detailed system design is shown in Figure 7. The results of this analysis are summarized in Figure 8, which shows the range of probabilities for total system failure per hour. At the time this analysis was performed it was known that the results would be optimistic because of assumption (g), that the digital and analog systems would be completely independent. Approximating the failure rates of the shared circuitry, the probability of total system loss for the selected F-8 DFBW configuration was computed to be 10^{-9} in a 1 hour flight ($D_C^3 B^3$).

Reliability analyses were also used during the preliminary design phase of the program to provide a basis for the selection of alternate design approaches. For example, three approaches to the sensor interface were considered. In two of them, hardwire links were provided between a sensor and more than one computer of the DFBW system. In the third approach, the one actually implemented, each computer collected data from its own set of sensors. The digitized result was exchanged among computers via the serial data bus. Although other approaches showed an improvement of up to 26 percent in sensor subsystem reliability, the theoretical contribution to an overall improvement in system reliability was small.

A detailed analysis of F-8 DFBW system reliability (Ref. 12) was undertaken after the system was implemented and flight tested. The method followed the MIL-HDBK-217 procedure. During the construction of the reliability equations, a new way to visualize the equations was conceived of which gave the system analyst additional insight into subsystem dependencies. The loss of control and abort probabilities were then computed from the best available component and subsystem failure rates and from actual experience during ground tests. The results, which are shown in Figure 9, are similar to the predesign analysis but indicate higher loss of control and abort figures. The shortcomings of the MIL-HDBK-217 method were as follows: (a) the MIL-HDBK-217 format

did not permit the direct expression of overall DFBW system complexity; (b) some failure modes had to be ignored because of the complexity of the interaction between the software and the hardware; (c) the probability of occurrence of many software/system unique failure modes were not estimable or measurable.

The best approach to these problems would be to include all of these effects and compute the sensitivity of system reliability as a function of various occurrence probabilities. It is apparent that the complexity of digital systems can easily outstrip the ability of reliability models to assess the integrity of a system. This is true not only for the MIL-HDBK-217 approach, but also for more sophisticated state modeling and other computer-aided reliability assessment tools. Designers will probably continue to use algorithms and other techniques that are difficult to model, however, so more effort should be devoted to the development of adequate assessment methods.

The general purpose digital computers used in the F-8 DFBW program suffered from substantial failure rates during the first part of the flight test program. Actual failure rates at one time were more than five times the predicted values. The hardware design did not allow the acquisition of sensor data from a failed channel. The probability of the loss of a complete triplex sensor set due to sensor and computer faults in different channels was substantially higher than could be tolerated in a system requiring the operation of two sensors. For the F-8, loss of a complete sensor set merely led to degraded operation. This example indicates the need to make architectural selections based on some of the more extreme failure rate possibilities.

4.4 Actual Synchronization Performance

Because computer synchronization was such a vital part of system performance, special tests were conducted to determine the performance of the synchronization algorithm. Actual measurements of the time skew between computers upon exiting from the synchronization routine ranged from 4.5 to 9.5 microseconds. The synchronization of the three computers is illustrated in Figure 10. The distribution of "look" loops shows that the triplex set acquired synchronization the first time through the algorithm more than 99 percent of the time. The distribution was nearly identical for a different set of three computers. The design allows for as many as 10 "look" loops within each half of the synchronization algorithm before a synchronization miss is noted. Ten consecutive misses are required before partners stop looking for the missing partner.

The synchronization method used ensures that the computer-generated actuator commands will be identical bit for bit. Telemetry data from actual flight test maneuvers confirms the close tracking of the digital-to-analog converter outputs (Fig. 11). Digital telemetry data showed the commands to be bit identical. Synchronization integrity has also been extremely good. In 4000 hours of operation on the flight software, no loss of synchronization due to other than known hardware faults has occurred. Resynchronization following induced transient faults in hardware, software, and power has been totally reliable. Alarm tables and fault logs implemented in the flight software show that not one synchronization cycle has been missed during any flight or during all monitored ground operations, except for cases of bonafide hardware faults. This result, combined with the fact that temporary loss of synchronization is recoverable, indicates that channel synchronization is not a significant or unique contributor to system unreliability.

4.5 Experience With Computer Redundancy Management

Table 5 summarizes the in-flight computer and interface unit failure experience. Computer faults during the second and third flights validated the fail-operational capability of the system earlier than expected. For these and all other hard failures experienced in flight, the system FDIR response was identical to that observed during ground testing. In both cases, the two remaining channels declared the faulty channel hard failed (nonrecoverable), and operation continued with no degradation in flying qualities. Precautionary, routine landings were made on the remaining two channels.

An actual in-flight computer failure is shown in Figure 12. Multiple restarts occurred in an attempt to correct what was an unrecoverable fault. No control surface deviations occurred during these restarts. The channel was declared failed by both partners; it also declared itself failed within 350 milliseconds of the failure. The remaining two channels operated normally.

The input-output fault on flight 17 was the only transient fault experienced during the flight program. A restart restored the channel to normal operation. Had restart capability not been present, the channel would have failed permanently. This case illustrated the advantage of automatic restart and resynchronization capability. The ability of the system to tolerate severe transient problems is illustrated in Figure 13. In this test, 28 volt source power was interrupted on one channel. Restarts in one channel had no effect on the other two channels. Once power was restored to the interrupted channel, the channel was immediately restarted and returned to normal synchronized operation. The improvement in system reliability due to restart capability could easily be computed if the probability of the occurrence of recoverable transient faults were known, along with the conditional probability of successful recovery given the occurrence of a recoverable fault. The latter probability can be estimated from ground tests, and is in excess of 0.99. The probability of occurrence of a transient fault cannot be predicted, and only a crude estimate can be made even after a substantial amount of operating experience.

Transient fault accommodation increases the complexity of the software, but it provides protection against unknown and unforeseen circumstances. This is one example of good systems engineering practice based on a poorly defined model.

4.6 Computer Channel Independence

The assumption of channel independence is fundamental to parallel multichannel flight control design. Experience has been gained during the course of the F-8 DFBW program which bears on this subject.

The control system performed very well. All takeoffs and landings were made using the primary digital system. Reversion to analog backup system was not required on any flight. No channel fault ever propagated across channel boundaries. No software anomaly occurred on any flight. And finally, no hazardous, or even anomalous, surface commands were ever generated.

Further, detailed analyses carried out during the course of the flight program, usually following hardware faults, did not reveal any single fault that would lead to the loss of the entire triplex digital flight control system. No exception to this finding has been observed in any of the ground and flight testing performed on the system.

However, the analyses did uncover some dual, successive faults which could lead to the loss of the triplex system. Although the general design specifications would not appear to be violated by a two fault condition, the system was supposed to accommodate these classes of faults.

To be specific, the first fault would be a partial failure in which one channel did not receive serial data from one partner. Thus, this channel would have a differing opinion as to the health of its partners. A second permanent or transient failure serious enough to cause a restart could then prevent the system from restoring normal operation. Relatively minor software logic modifications corrected this potential problem.

Although the sequence of events described did not occur during actual operation, the general characteristics of the problem should be noted in future applications. These may be stated as follows: (a) existence of a latent fault which appears benign at the three channel level of operation; (b) existence of a situation in which the multiple channels do not have an identical image of the state of the system; (c) the combination of a benign latent fault with a normal system reaction to a noncatastrophic second fault or anomaly.

Experience during the F-8 DFBW program has led project engineers to conclude that even though present technology can produce systems that have small probabilities of common mode failures, a means of surviving such failures is necessary, either by an independent backup system or by built-in recovery modes. This conclusion is based on the fact that the potentially troublesome combination of circumstances described above escaped detection until well into the flight program. Further, a detailed analysis of all actual system faults is highly recommended; it was this type of analysis that led to the identification of the additional failure modes.

5.0 BASELINE SENSOR REDUNDANCY MANAGEMENT

5.1 Sensor Fault Detection, Isolation, and Reconfiguration Algorithms

Analog and discrete signal fault detection, isolation, and reconfiguration are accomplished in software with relatively simple algorithms. The fail-operational fail-safe analog sensor FDIR algorithm is depicted in Figure 14. With three good sensors, the signal selector chooses the midvalue for use in the control laws. Only the selected value is converted to a floating point number to reduce computation overhead. The selected value is compared with each of the three input signals for fault detection. When the difference between the selected value and an input signal exceeds a stored tolerance level for five consecutive minor cycle tests, or 100 milliseconds, the fault detection and isolation module declares the offending sensor hard failed. Table 6 lists the fault threshold values used in the flight program for each flight control sensor. The failure status monitor module reconfigures the signal selector to average the calculations of the remaining two good sensors.

When operating on dual sensors, the fault detection and isolation module compares the difference of two signals against the threshold prior to the time of signal selection. If the threshold is exceeded, the failure status monitor commands the signal selector to send the last good value until either the sensor set is determined to be unusable or the two sensors again agree. Agreement must recur within 100 milliseconds to prevent the set from being declared permanently failed.

Figure 15 illustrates the operation of the FDIR algorithm when it is operating with two sensors. A hardover fault is illustrated in Figure 15(a). The signal selector holds the last good value until the persistence count increases to 5, at which time a sensor set is declared failed.

In Figure 15(b), sensor A suffers a transient fault. The disturbance in the selected average value is again suppressed by the signal selector that holds the last good value. In this example, the two sensors return to agreement before the persistence count reaches 5, and the signal selector reverts to averaging.

Figure 16 shows the simulation response of the aircraft to two successive hardover failures of pitch rate gyros. No vehicle motion transient occurs. The second fault causes an automatic downmode to the direct mode, a pitch control law which does not use pitch rate feedback.

5.2 Sensor Noise Statistics

Data were obtained for several sensors to establish the actual noise content of the signals that would be processed by the sensor FDIR algorithms. Sensor data from ground operation of the aircraft with the engine off, at idle, and at 80 percent rpm were analyzed. An 82-second segment of sampled rate gyro and accelerometer outputs for the three conditions was analyzed. The estimated standard deviations of the results are shown in Table 7. With the engine off, the root mean square (rms) output levels are approximately equivalent to one-third of each sensor's least significant quantization bit. The analysis also showed that the noise characteristics are dominated by nearly white, random variations of the least significant one or two bits. The analog output of the sensors suggests that the random bit variations are due to system noise rather than to internally generated sensor noise.

The rms sensor outputs increase substantially when the engine is operating. These results are also shown in Table 7. Power spectral density analysis indicates that most of the increase can be attributed to resonances between 3 and 20 hertz. While these resonances are, in fact, legitimate sensor measurements, they were treated as effective sensor noise in this analysis. The power spectral density results for a pitch rate gyro at 80 percent engine rpm is shown in Figure 17. The results of these tests showed that quiescent system noise, whether correlated or not, was small relative to the failure thresholds which for the pitch rate gyro, for example, was 4 degrees per second.

5.3 In-Flight Multisensor Statistics

Single sensor statistics are not as germane to multisensor FDIR as sensor differences which can result from uncorrelated noise, scale factor and bias errors, and measurement deviations due to sensor placement. Detailed analyses of sensor pair data were conducted to determine actual sensor differences in the flight environment. Telemetry data from the flight control sensors were examined once per second for several flight situations. Various statistics were computed for individual sensor outputs and sensor pair differences, such as mean, rms value, standard deviation, and maximum absolute differences.

Table 8 summarizes the results of analyses of rate gyros and accelerometers for a flight in which the F-8 DFBW aircraft was simulating final approaches to Edwards Dry Lake by the shuttle orbiter. These approaches were characterized by sensor A rms values during the 55 minutes of data analyzed. The rms values of the sensor pair differences are generally less than half the maximum difference recorded. The maximum differences are all less than half the fault threshold itself. Table 8 also summarizes the sensor FDIR performance on that particular flight. Except for the miscompares in the longitudinal accelerometer set, the FDIR results are entirely consistent with the sensor analysis. It should be noted that miscompares are counted for a deviation above the threshold for any of three pairs of sensors ((A, B), (A, C), and (B, C)). The data in Table 8 were for pair A minus C only. Also, the computer FDIR operates at 50 samples per second, compared with the one-per-second sample rate used in the sensor analysis.

Table 9 summarizes the rms and maximum absolute differences for the (A, C) sensor pair during a takeoff roll. Except for the rms longitudinal accelerometer difference, all values are less than those observed during the flight.

Table 10 summarizes the rms and maximum differences for the A minus C sensor pair during a 9-minute period of light-to-moderate turbulence. In general, the values are most similar to those of the takeoff roll.

Another, more graphic illustration of sensor pair differences is presented in the series of crossplots in Figure 18. These crossplots represent the sensor pair with the largest deviations during moderate air combat maneuvering. The ideal tracking relationships, along with the software fault tolerance values, are superimposed on each crossplot. These data were acquired by sampling sensor responses at a rate of 50 per second. The crossplots in Figure 18 are presented at 5 samples per second and are representative of the results for the high sample rate. These data show comfortable margins for nuisance fault rejection. Simulation results of induced failure tests show that maximum transients of approximately 0.2g for accelerations and 5 degrees per second for body rates result from sensor reconfiguration.

Analyses are currently directed at establishing models for the distribution of the sensor pair differences. χ^2 analysis confirms the observations that the distributions are not normal. Figure 19 shows the actual pair difference histogram for the yaw rate gyro (A, C) pair difference. This particular histogram, besides showing a nonzero mean, illustrates the nature of most of the sensor histograms. The distribution falls abruptly to zero, with an insufficient number of events of high magnitude to approximate a normal distribution.

5.4 Overall Sensor Fault Detection, Isolation, and Reconfiguration Experience

The instrumentation software built into the flight control system software program allowed sensor FDIR performance to be examined and analyzed after each flight. Counters in the FDIR program logged each miscompare of any sensor pair of a triplex or duplex set and the number of times the fail count reached within 1 or 2 of 5, the point at which the sensor was declared permanently failed.

Table 11 summarizes all flight experience with the analog sensor FDIR algorithm. Permanent failures were declared for a lateral accelerometer, angle-of-attack vane, altimeter, pitch attitude gyro, and heading gyro. Of these, the angle-of-attack vane, pitch attitude gyro, and altimeter failures could be attributed to actual hardware faults. The lateral accelerometer fault declaration was a false alarm. Because of this single event, the lateral acceleration fault threshold was increased to 0.2g. The two heading gyros are not slaved to any north-seeking system, because they are used only in the autopilot heading hold mode. The gyro spin axes are initialized to be offset 90°, resulting in fairly large errors during high rate turns. Because of the unique manner in which the heading gyros are used in the F-8 DFBW system, these results are not significant.

Table 11 also shows that during the flight program the sensor FDIR failure count reached 4 for a lateral accelerometer, altimeter, and pitch attitude gyro sensor pair. The number of times the count reached 4 is indicated in the chart. Miscompares tallied for every flight are summarized in the table by the maximum number of miscompares on any single flight. The tally for the lateral accelerometer and altimeter are associated with deteriorating performance prior to the actual failure declaration. In both cases, the fault trend was observed prior to the fault declaration.

Experience with control stick position linear variable differential transducers (LVDT's) is also of interest. As indicated in Table 10, no actual or nuisance faults occurred in any LVDT sensor set. Special ground tests were conducted to determine the actual tracking performance of these transducers. Table 12 summarizes the tracking data for the center stick, side stick, and rudder pedals. The side stick is a limited displacement type and uses LVDT transducers. Maximum position displacement for the side stick is approximately 0.16 centimeter for maximum force input. Flight deviations are well under the fault threshold values. Moderate frequency stick motion tests conducted during ground tests show increases in deviations, the highest occurring in the pitch center stick. With very high stick motion frequency the deviations are between 27 and 80 percent of the fault threshold value. The differences are attributable to the frequency-sensitive nature of the alternating current transducers and the associated demodulator characteristics. These results indicate the need for special care in the selection of LVDT transducer tolerances.

Fault detection, isolation, and reconfiguration on redundant discretes is carried out in a manner similar to that for analog sensors, with discrete voting and comparison operations used to perform the selection and fault isolation functions. Ground experience showed that the design value of 100 milliseconds to accommodate for skew tolerance was too small. The skew tolerance was increased to 300 milliseconds before the first flight. Except for

one occurrence, no nuisance discrete faults have been declared. Because of insufficient mechanical overtravel on the switches, unmodeled wing flexure during the takeoff run of the first flight caused the entire triplex discrete set to be declared failed. A mechanical adjustment corrected the problem.

In general, the flight experience with sensor fault detection, isolation, and reconfiguration is consistent with design expectations. Fewer nuisance faults occurred than were expected because the actual sensor pair difference distributions were not known prior to flight. The experience with this relatively simple FDIR algorithm indicates that for many applications more complex methods may not be required. Current work is directed at developing design nomographs for the selection of fault thresholds and failure count values based on specifications for missed failure, false alarm, and sensor reliability.

6.0 ANALYTIC REDUNDANCY MANAGEMENT

Current approaches to sensor redundancy management utilize cross channel comparison to detect faults at the three or four sensor level. Sensor miscompares at the two level may be treated in one of three ways: (a) degrade the control mode to permit operation without the sensor set; (b) synthesize the signal provided by the sensor set with information from independent sources; or (c) isolate the good sensor and use it.

The baseline F-8 DFBW system illustrates the first strategy. For example, a second pitch rate gyro failure will cause an automatic transfer from the command or stability augmentation mode to the direct mode, which provides unaugmented control of the aircraft. The second approach would use state reconstruction techniques to provide an estimated sensor signal for use in the control laws instead of relying on further fault isolation. The third approach would use an in-line algorithm to isolate the remaining good sensor for use in the control laws. Some sensors have built-in test circuits to provide an indication of sensor health. Spin motor rotation detector circuits are commonly found in gyro assemblies, for example. Another technique often proposed is the use of torque signals to isolate the operational sensor. However, the failure coverage of such built-in test equipment is generally inadequate to provide the high integrity fault isolation needed for a flight-critical system.

A more general approach to both approaches (b) and (c) is termed analytic redundancy management (ARM). By this is meant the use of relationships among dissimilar sensors to achieve fault detection and isolation. A substantial amount of theoretical and simulation work has been conducted to establish the feasibility of applying analytic redundancy techniques to flight control sensors (Refs. 13 to 18). An analytic redundancy management algorithm was implemented and flight tested on the F-8 DFBW aircraft to assess the failure coverage and false-alarm characteristics of the analytic redundancy approach as well as to gain insight into the implementation and practicality issues.

6.1 F-8 Digital Fly-By-Wire Analytic Redundancy Management Algorithm

An experimental ARM algorithm (Ref. 13) was implemented in the onboard DFBW computer software so as to execute in parallel with the baseline sensor FDIR. The ARM algorithm operated on the (A, B) sensor pair only, simulating a dual sensor configuration. Results of ARM processing were recorded for postflight analysis only. There was no actual sensor reconfiguration.

The F-8 DFBW ARM algorithm relies on a direct comparison of sensor pair data to trigger the fault isolation process. The fault detection algorithm compares the difference of the average value of each sensor over a moving window with a threshold tolerance. When this moving window difference exceeds the fault threshold, the fault isolation process is initiated. A bias failure magnitude (BFM) value is chosen for each sensor type based on *a priori* sensor statistics and predicted error sources. A threshold magnitude of 0.75 BFM and window length is stipulated for each sensor type to yield a predicted false-alarm or missed failure probability of 10^{-4} per event. The BFM and window values for the 12 sensors are given in Table 13.

The fault isolation process involves a comparison between each sensor of a suspect pair and an estimate of the correct sensor value based on independent, dissimilar, and unfailed sensors. The ARM algorithm was designed to detect and isolate failures in 12 sensor types; the longitudinal accelerometer, normal accelerometer, lateral accelerometer, roll rate gyro, pitch rate gyro, yaw rate gyro, pitch attitude gyro, roll attitude gyro, directional gyro, altimeter, Mach meter, and angle-of-attack vane. The output of the single sideslip vane was used for some cases.

Four types of analytic redundancy are utilized by the algorithm. Rotational kinematics (RK) relates the integrated outputs of the rate gyros and the outputs of the attitude gyros. Altitude kinematics (AK) relationships exist between the altimeter output and the second integral of the resolved accelerometer outputs. Translational kinematics (TK) relates the integrated output of the accelerometers, vertical gyros (pitch and roll), and rate gyros with the outputs of the air data sensors—Mach meter, altimeter, and angle-of-attack and -sideslip vanes. Translational dynamics (TD) relates the aerodynamic forces on the aircraft as measured by accelerometers and the calculated aerodynamic forces based on air data sensor outputs and stored estimates of aerodynamic coefficients.

The residual process, γ, which represents the comparison between the suspect sensors and the other, unfailed instrument types, is then processed in a sequential probability ratio test (SPRT). The SPRT gathers enough information to choose between two hypotheses, assuming a Gaussian process: The first, hypothesis 1 (H_1), states that at time t_K, the residual process, γ, is Gaussian with mean m_K and variance σ^2. The second, hypothesis 2 (H_2), states that at time t_K, the residual process, γ, is Gaussian with a mean of zero and variance σ^2. The first represents the failure hypothesis, where the mean is calculated assuming a bias failure magnitude of BFM. The log likelihood ratio z_K for the K^{th} sample is defined in terms of conditional probabilities as follows:

$$z_K = -\ell n \frac{p(\gamma_K|H_1)}{p(\gamma_K|H_2)} \qquad (1)$$

and after n samples have been taken (assuming the independence of the residual processes), the log likelihood ratio of the n samples, u_n, is given by the expression

$$u_n = -\ell n \frac{p(\gamma_1, \ldots, \gamma_n | H_1)}{p(\gamma_1, \ldots, \gamma_n | H_2)} = \sum_{K=1}^{n} z_K \tag{2}$$

For the case of the two hypotheses given above, the form of z_K is

$$z_K = \frac{m_K}{\sigma^2}\left(\frac{m_K}{2} - \gamma_K\right) \tag{3}$$

and the log likelihood ratio for n samples given by Equation (2) becomes

$$u_n = \sum_{K=1}^{n} \frac{m_K}{\sigma^2}\left(\frac{m_K}{2} - \gamma_K\right) \tag{4}$$

If either H_1 or H_2 is assumed to be true, the stipulation of incorrect classification probabilities directly yields the thresholds $a < 0$ and $b > 0$ and the following decision rule: If

$$\left.\begin{array}{l} u_n \le a \\ \text{accept } H_1. \text{ If} \\ a < u_n < b \\ \text{take another sample. If} \\ b \le u_n \end{array}\right\} \tag{5}$$

accept H_2. If the log likelihood ratio is between the thresholds, a choice of hypotheses that meets the specified incorrect classification probabilities cannot yet be made, and another sample must be taken.

One attractive property of the SPRT is that it minimizes the number of observations necessary to meet the probabilities. In addition, the SPRT is independent of the *a priori* probabilities of the two hypotheses, and its performance in the presence of a mean or variance other than that hypothesized is readily analyzed. It is because of these properties and because of the inherent simplicity of the SPRT (see Eqs. (4) and (5)) that the SPRT was chosen as the basic identification mechanism for the algorithm.

The direct SPRT approach was modified in two ways for reasons related to the practical application of the method. First, a time limit was set for the test. If a failure identification could not be made within this time, an unidentifiable fault was declared and the algorithm was reinitiated. This situation reflects a breakdown of the algorithm. The second modification attempts to account for unmodeled errors in the residual process γ. For a given instrument type, j, the modified likelihood ratio becomes:

$$u_n^j = \sum_{K=1}^{n} \left[\frac{m_K^j}{\sigma^2}\left(\frac{m_K^j}{2} - \gamma_K^j\right) + \frac{|m_K^j|}{\sigma^2}|E_K|\right] \tag{6}$$

where E_K is the worst case error from all sources in the analytic redundancy test at time t_K. The effect of this modification is to add a moving threshold which is conservative by an amount equal to the worst case contribution of the noise terms.

Using the Euler relationships for body axis and inertial quantities, the residual γ_K^j for the roll rate gyro j at the time t_K is defined as follows

$$\gamma_n^j = \sum_{K=1}^{n} \left\{p_K^j T - \left[\left(\varphi_K - \varphi_{K-1}\right) - \left(\psi_K - \psi_{K-1}\right)\sin\theta_K\right]\right\} \tag{7}$$

where T is the sample period, φ_K, θ_K, and x_K are the measurements of roll, pitch, and heading attitude at time t_K, and p_K^j is the output roll rate gyro j at time t_K. The values for p_K^j and θ_K are averaged over the sample period. For the calculation of the modified SPRT, the added error term contained the following terms: attitude measurement noise, 1.15° alignment error, 5 percent scale factor error, attitude measurement bias, and heading gyro error. The equations describing all of the RK, AK, TK, and TD relationships are described in Reference 13.

6.2 ARM Flight Results for Induced Faults

The F-8 DFBW ARM software included an extensive in-flight simulated fault inducement capability. Sensor faults which could be simulated included bias shift, drift, scale factor difference, hardover, transient pulse, and

open line. Other sources of sensor error which could be included were uncertainties in sensor alignment, center of gravity, weight, and stored aerodynamic coefficients to be used in the translational dynamics algorithms.

The first phase of the flight program involved the inducement of simulated sensor faults during cruise flight and the operation of the ARM algorithm without induced faults during moderate maneuvering flight. Table 14 contains the sensor fault values for the first phase of the flight program. Table 15 summarizes the test results for the aircraft in cruise flight. The average time for fault isolation is shown for each sensor type and fault source. All sensor bias and drift faults were isolated correctly. Because faults were induced during cruise flight, fault identifications did not always occur for other faults. This was due to insufficient sensor response in cruise flight. Fault isolation times agreed well with predictions, as shown by Figure 20. Faults induced during moderate maneuvering flight produced results similar to those in Table 15.

6.3 Overall ARM Flight Performance

The induced fault tests confirmed the validity and capability of the ARM concept. The analysis of the flight data led to a more detailed knowledge of the performance of the ARM algorithm in the flight environment.

Most of the problems encountered during the flight tests were related to the fact that the sensor systems were incompletely modeled. The synchro-to-digital conversion scheme used for the attitude gyros can produce large transient deviations at attitudes near ±180°. This caused false rate gyro and attitude gyro fault declarations during large rolling maneuvers. Several false alarms were associated with the lack of alignment in the directional gyros, which produced large average heading angle errors. Analysis of these and similar cases led in most cases to increases in the modified SPRT worst case error terms. Increases from 50 to 300 percent were made in the acceleration terms for the TD, TK, and AK modes of ARM. These changes did not substantially degrade ARM failure detection or isolation performance, and they significantly reduced false-alarm susceptibility.

In some cases, worst case errors were accounted for in the SPRT calculations rather than in the sensor system model. Such was the case for the altimeter and Mach instruments in the transonic region, where smooth, repeatable behavior was observed for both acceleration and deceleration maneuvers (Fig. 21). These characteristics could easily be modeled if necessary.

Other problems that occurred in this phase of the flight testing were associated with operational logic that was not included in the software: a false alarm occurred for the angle-of-attack vanes after touchdown as rollout velocity decreased to taxi speeds.

During one flight, an actual hardware fault occurred in one angle-of-attack vane during ARM operation. The fault was detected and isolated correctly by the ARM algorithm. Figure 22 shows the two vane signals and the modified SPRT results for each vane. Angle-of-attack residuals are computed as the difference between normal acceleration as computed from aerodynamic forces using stored aircraft lift and drag coefficients, and acceleration as measured by the normal accelerometers. The baseline F-8 DFBW sensor FDIR algorithm detected the vane fault several seconds later, when an absolute error of 4° persisted for 100 milliseconds.

Another positive result was noted when no permanent fault was declared during flight operations in moderate buffet during maneuvers at angles of attack above 18°, which produced pitch rate gyro oscillations of 4 degrees per second in the 6 to 10 hertz frequency range.

6.4 Observations on Analytic Redundancy

The first phase of the flight tests showed ARM performance to be very good. Modifications made to the software for the second phase of the flight program should further improve the integrity of the algorithms. Several observations regarding the ARM concept may be made on the basis of the development and flight test experience, however.

First, the performance of the ARM algorithms was substantially more sensitive to the degree of modeling of normal sensor operation than to the degree of modeling of sensor failure modes. The degree of modeling necessary to represent the sensor system (including signal conditioning processing) and deviations due to natural phenomena is significantly greater than is required for contemporary flight control system design. Sensor biases affect ARM performance to a greater degree than they do contemporary FDIR approaches. This may require the incorporation of bias removal terms to account for long term bias and bias variations. The complexity of the software implementation, although greater than is necessary for contemporary FDIR algorithms, is similar to that for guidance and navigation algorithms, and should not pose unusual problems. The F-8 DFBW ARM algorithm required approximately 5.5 milliseconds of computation time every 60 milliseconds, and was implemented in approximately 8000 16-bit words of software, excluding modules unique to flight testing. And finally, the algorithm lends itself to logical and convenient segmentation suitable for distributed processing.

7.0 FLIGHT QUALIFICATION

The successful flight test experience with the F-8 DFBW aircraft was due to both the quality of the system and the extensive and rigorous ground testing that preceded the first flight. The ground test facilities included a digital computer emulator, single and triple channel breadboard systems, a flight system, and a complete iron bird which was interfaced with a six-degree-of-freedom aerodynamic simulation. The iron bird was capable of operation with either the breadboard systems or the actual flight hardware. Approximately 2000 hours of system operation preceded the first flight.

The failure modes and effects tests were critical to the successful qualification of the digital fly-by-wire control system. These tests were conducted on the final version of the flight software incorporated in the iron bird. Because the iron bird contained all the electrical, hydraulic, mechanical, and electronic subsystems installed in the flight vehicle, the resulting environment was realistic.

The selection of the fault matrix is difficult for a system as complex as this one. A component-by-component or wire-by-wire open/short fault test matrix is clearly impossible. Therefore, the strategy adopted was to

induce faults at the functional level and to judge the integrity of FDIR on the basis of its response. It is imperative that the fault detection system be designed at the outset to be testable. If the FDIR scheme is sophisticated, involves operating on a large amount of data, and proceeds through a large number of intricate steps, the system will quite possibly be unverifiable. That is, the number of tests required to establish correct response, even for those faults that can be envisioned, will be extremely large.

The F-8 DFBW FDIR design and failure modes and effects testing approach is based on the following assumptions: (a) The ability of a channel to maintain synchronous operation and to communicate with its partners is a primary indicator of channel health. (b) A relatively small amount of information exchanged between channels can be used as the basis for checking channel health. This information can be thought of as vital signs. (c) Fault detection can be based on the manifestation of a fault, rather than on the identification of the faulty component. (d) The restart mechanism will provide automatic transient fault protection whether or not the source of the fault is known.

These assumptions dramatically reduce the test matrix and make it possible to approach the failure modes and effects testing in the following manner: (a) Verify that FDIR response to abnormalities in the small number of vital signs is correct. (b) Verify that all built-in test hardware produces warning signals to the FDIR for the types of faults it is designed to detect. (c) Verify that the restart process restores normal operation for transient abnormalities in the vital signs or for the built-in test warning signals. (d) Verify that the restart mechanism suspends its attempts to restore operation for unrecoverable faults. (e) Verify fail-operational fail-safe performance after all unrecoverable faults.

This relatively modest test matrix provides a thorough functional validation of the FDIR process. The test matrix is summarized in Table 16. Faults were induced for all combinations of two channels, even though channels were identical.

The successful flight qualification of the DFBW system was due in large part to the realistic test environment. Virtually all of the critical software was tested in the presence of all other software and subsystems. This was considered to be vital to the success of the testing.

Computations of flight reliability are based on the assumption that the system is fault free before flight. An extensive built-in automatic preflight test sequence was performed to ensure that no latent faults existed in the system before takeoff. No known fault went undetected by the preflight test program.

8.0 CONCLUSIONS

Flight experience with the F-8 digital fly-by-wire (DFBW) aircraft confirms the ability of multichannel digital flight control systems to perform critical and complex primary flight control functions satisfactorily. The decision to use multichannel digital systems on commercial and military aircraft implies a high degree of confidence in the state of the art of digital flight control system design, manufacturing, and flight qualification procedures. The following specific conclusions and observations are drawn from this experience.

(1) Actual flight experience has confirmed and validated design expectations for contemporary fault detection, isolation, and reconfiguration approaches for multiflight computers and sensors.

(2) Practice has overtaken modeling and prediction methods. MIL-HDBK-217 was applied to the digital fly-by-wire system, but the approach was found to have serious deficiencies. It is not currently possible to assess the reliability of a digital fly-by-wire system satisfactorily with contemporary methods. Designers will continue to use advanced redundancy management techniques even though they are difficult to model, however, so more research should be done to advance assessment techniques.

(3) The computations of loss of control and abort probabilities assume channel independence. Experience acquired during the F-8 DFBW program indicates that existing technology is capable of producing systems with extremely small probabilities of common mode system failures. However, means of surviving the most likely faults must be provided, because validated models for determining the probability of such faults do not exist.

(4) Adjustments were required to the sensor and computer fault threshold constants during the flight test program. Detailed sensor pair difference models based on flight performance will permit the systematic selection of fault threshold values and persistence count as a function of required missed fault and false-alarm probabilities. False alarms were rare in the flight program.

(5) The assumed characteristics of transient faults or system noise have a major impact on the overall integrity of FDIR algorithms, yet such models are not generally available before flight.

(6) Synchronization of a multicomputer set was found to have no significant or unique contribution to system unreliability.

(7) Analytic redundancy management (ARM) performance was very good in detecting and isolating a large set of induced sensor faults. High quality sensor system modeling, more than failure mode analysis, is required to provide a high integrity fault isolation capability. No unusual computational requirements were encountered in the implementation of the ARM algorithm.

(8) Extensive ground testing was believed to be responsible for the successful flight experience with the DFBW system. The control system design enhanced verifiability.

(9) Further research is necessary to enable pure DFBW systems to tolerate and survive common mode software faults without the need for an independent dissimilar backup system.

Safe flight operations have been conducted with the fault tolerant, full authority digital fly-by-wire flight control system during 80 F-8 DFBW research flights. This experience has confirmed the feasibility, practicality, and vi-

ability of digital fly-by-wire control. Future applications will confirm the benefits of such systems in achieving greater aircraft performance at lower cost.

REFERENCES

1. Kenneth J. Szalai, Philip G. Felleman, Joseph Gera, and Richard D. Glover, Design and Test Experience With a Triply Redundant Digital Fly-by-Wire Control System, Aug. 1976, AIAA Paper 76-1911.

2. Samuel R. Brown and Kenneth J. Szalai, Flight Experience With a Fail-Operational Digital Fly-by-Wire Control System, Nov. 1977, AIAA Paper 77-1507.

3. Kenneth J. Szalai, Calvin R. Jarvis, Gary E. Krier, Vincent A. Megna, Larry D. Brock, and Robert N. O'Donnell, NASA, Digital Fly-by-Wire Flight Control Validation Experience, 1978, NASA TM-72860.

4. L. E. Fairbanks and J. E. Templeman, Flight-Critical Digital Control System Evaluation, Apr. 1975, AIAA Paper 75-566.

5. Air Force Flight Dynamics Lab., Flight Test Evaluation of a Digital Multimode Flight Control System for the A-7D Aircraft, 1975, AFFDL-TR-75-97, *Vols. I and II.*

6. S. S. Osder, D. C. Massman, and B. J. Devlin, Flight Test of a Digital Guidance and Control System in a DC-10 Aircraft, Apr. 1975, AIAA Paper 75-567.

7. Eugene A. O'Hern, Space Shuttle Avionics Redundancy Management, Apr. 1975, AIAA Paper 75-571.

8. John D. McDonnell, Douglas Aircraft Co., Advanced Digital Avionics for the DC Super 80, 1979, DP-6843. (Available from Douglas Aircraft Co. Tech. Library, MC 36-84, 3855 Lakewood Blvd., Long Beach, CA 90846).

9. Richard E. Kestek, YC-14 *Digital Flight Control Data Management*, Aug. 1975, AIAA Paper 75-1087.

10. John E. Krings, F/A-18 Status Report, Soc. Experimental Test Pilots 1979 Report to the Aerospace Profession, c. 1980, pp. 1-9.

11. C. L. Seacord and D. K. Vaughn, NASA, Preliminary System Design Study for a Digital Fly-by-Wire Flight Control System for an F-8C Aircraft, 1976, NASA CR-2609.

12. L. D. Brock and H. A. Goodman, Reliability/Safety Analysis of a Fly-by-Wire System, Aug. 1980, AIAA 80-1760-CP.

13. Mukund N. Desai, James C. Deckert, and John J. Deyst, Jr., Dual-Sensor Failure Identification Using Analytic Redundancy, *J. Guidance and Control, Vol. 2, No. 2, May-June 1979, pp. 213-220.*

14. R. C. Montgomery and D. B. Price, Management of Analytical Redundancy in Digital Flight Control Systems for Aircraft, Aug. 1974, AIAA Paper 74-887.

15. J. J. Deyst and J. C. Deckert, RCS Jet Failure Identification for the Space Shuttle, Proc. of the Internat. Fed. Automatic Control, Part 4, 1975, pp. 28.21-28.28.

16. J. C. Deckert, M. N. Desai, and J. J. Deyst, Evaluation of the F-8 DFBW Analytic Redundancy Sensor FDI Algorithm Using Telemetry Data, Nov. 1977, AIAA Paper 77-1506.

17. Thomas B. Cunningham and Robert D. Poyneer, Sensor Failure Detection Using Analytical Redundancy, Proceedings of the Joint Automatic Control Conf., San Francisco, Calif., June 1977, Vol. 1, pp. 278-287.

18. A. S. Willsky and H. L. Jones, A Generalized Likelihood Ratio Approach to State Estimation in Linear Systems Subject to Abrupt Changes, Proceedings of the Conference on Decision and Control, Nov. 1974, pp. 846-853.

TABLE 1.—GENERIC CONTROL SYSTEM DESIGN REQUIREMENTS

Area	Requirement
Fault tolerance	Fail-operational/fail-safe Second fail to backup
Turn-on/off	Automatic initialization from arbitrary turn-on/off sequence
Fault detection	Hard failures to be detected within 200 msec Hard failure declarations to be irreversible
Reconfiguration	Automatic
Transient fault recovery	Fully restartable in any configuration/mode
Immunity to false alarms	Design to be heavily weighted to avoid false alarms
Output command voting	Analog voting of surface commands
Computer intercommunication	Minimum possible
Synchronization	Frame or minor cycle only
Control-law interface	Multicomputer structure to be transparent to control laws
System integrity	Full time Full control surface authority Flight critical control No mechanical reversion

TABLE 2.—DIGITAL CONTROL COMPUTER CHARACTERISTICS

Central processor unit—	
Number system	Binary, fixed point, two's complement, fractional hexadecimal floating point
Operation	Full parallel
Fixed-point data	16 and 32 bits, including sign
Floating-point data	32 bits (24-bit mantissa) and 64 bits (56-bit mantissa)
Typical execution times, register to register, μsec —	
Fixed point:	
Addition	1.0
Multiplication	4.8
Division	8.4
Floating point:	
Addition	2.4
Multiplication	5.0
Division	10.5
Average instruction rate, per sec	480,000
Main storage —	
Type	Random access, destructive readout, ferrite magnetic core, nonvolatile
Cycle time, nsec	900 read/write
Addressable unit	16-bit halfword
Features	Parity and store protect on halfword
Capacity	32,000 full words
Input-output —	
Type	Parallel halfword plus parity, multiplex, half duplex
Maximum data rate, per sec	225,000 full words
Discretes	Four input, four output
External interrupts	Five levels
Physical characteristics —	
Weight, kg	24
Volume, m^3	0.025
Power, W	375
Environment	MIL-E-5400 Class 2X

TABLE 3.—INPUT SENSOR TYPE AND REDUNDANCY LEVEL

Sensor	Redundancy level	Signal type
Pitch rate	3	dc
Roll rate	3	dc
Yaw rate	3	dc
Axial acceleration	3	dc
Lateral acceleration	3	dc
Normal acceleration	3	dc
Pitch center stick position	3	ac LVDT[1]
Roll center stick position	3	ac LVDT
Pitch side stick force	3	ac LVDT
Roll side stick force	3	ac LVDT
Rudder pedal position	3	ac LVDT
Angle of attack	2	dc
Angle of sideslip	1	dc
Horizontal stabilizer actuator position	3	dc potentiometer
Surface positions (5)	1	dc potentiometer
Pitch attitude	2	Synchro
Roll attitude	2	Synchro
Heading angle	2	Synchro
Wing position	3	dc potentiometer
Mach	2	dc potentiometer
Altitude	2	Serial digital
Computer temperature	1	dc

[1]Linear variable differential transducer.

TABLE 4.—SYSTEM FAULT CATEGORIES

Detection mechanism	Potentially recoverable faults	Nonrecoverable faults
Computer built-in test hardware	Privileged instruction Illegal operation Store protect violation Instruction monitor Arithmetic error Power disruption	CPU/main store parity error Input/output main store parity error Microstore parity error Go/no go counter timeout Excessive restart rate after a potentially recoverable fault
Interface unit built-in test hardware	Temporary loss of computer input/output activity	Interface unit power or clock fault Loss of computer input/output activity for >530 msec
Computer software	Computer sync loss Input/output error Crosslink status miscompare	Excessive restart rate after a potentially recoverable fault

TABLE 5.—IN-FLIGHT COMPUTER AND INTERFACE UNIT FAILURE EXPERIENCE

Flight	Failure	Comments
2, 3, 53, 69	Computer memory fail	Parity error was unrecoverable Channel fail declared by partners and self No surface transient No change in flying qualities Routine landing on two channels
17	Transient interface unit component fault	Restart restored normal operation No surface transient Flight mission continued Component failed permanently in later ground test
19, 60	Computer fault stopped program execution	Partners assumed channel was off No restarts No surface transients No change in flying qualities Routine landing on two channels
22	Interface unit component permanent fault	Channel fail declared by partners and self Restarts failed to restore normal operation No surface transients No change in flying qualities Routine landing on two channels

TABLE 6.—FLIGHT CONTROL SENSOR FAULT THRESHOLDS

Sensor set	Tolerance
Pitch rate	4 deg/sec
Roll rate	10 deg/sec
Yaw rate	4 deg/sec
Axial accelerometer	0.1 g
Lateral accelerometer	0.2 g
Normal accelerometer	0.5 g
Pitch center stick	1.0 cm
Roll center stick	1.0 cm
Rudder pedals	0.5 cm
Angle of attack	2.0 deg
Left secondary actuator	3.5 cm
Right secondary actuator	3.5 cm
Pitch attitude	15 deg
Roll attitude	15 deg
Heading angle	15 deg
Mach	0.05
Altitude	150 m

TABLE 7.—ESTIMATE OF STANDARD DEVIATION OF SENSOR NOISE

Sensor	Ground test data		
	Engine off	Engine idle	80 percent rpm
Normal accelerometer	0.0018	0.0042	0.016
Pitch rate gyro	0.011	0.019	0.057

TABLE 8.—SENSOR PAIR STATISTICS

[55 Minute Flight Sequence]

Parameter	Pitch rate gyro, deg/sec	Roll rate gyro, deg/sec	Yaw rate gyro, deg/sec	Normal acceleration, g	Lateral acceleration, g	Longitudinal acceleration, g
Sensor A, rms	1.03	2.67	1.00	1.09	0.03	0.17
Sensor A - C, rms	0.53	0.77	0.43	0.04	0.01	0.01
Maximum |A - C|	1.13	1.47	1.62	0.15	0.05	0.04
Fault threshold	4.0	10.0	4.0	0.5	0.2	0.1
Total number of miscompares	0	0	0	0	0	3

TABLE 9.—SENSOR PAIR STATISTICS FOR TAKEOFF ROLL

Parameter	Pitch rate gyro, deg/sec	Roll rate gyro, deg/sec	Yaw rate gyro, deg/sec	Normal acceleration, g	Lateral acceleration, g	Longitudinal acceleration, g
Sensor A - C, rms	0.26	0.37	0.16	0.02	0.005	0.02
Maximum |A - C|	0.65	0.99	0.47	0.04	0.01	0.05

TABLE 10.—SENSOR PAIR STATISTICS FOR 9.5 MINUTES OF LIGHT TURBULENCE

Parameter	Pitch rate gyro, deg/sec	Roll rate gyro, deg/sec	Yaw rate gyro, deg/sec	Normal acceleration, g	Lateral acceleration, g	Longitudinal acceleration, g
Sensor A - C, rms	0.28	0.27	0.27	0.02	0.01	0.01
Maximum |A - C|	0.55	0.83	0.83	0.05	0.01	0.01

TABLE 11.—SENSOR FDIR FLIGHT EXPERIENCE

Sensor	Number of actual hardware faults	Number of false fault declarations	Number of times fail count reached 4	Maximum number of miscompares on any flight
Pitch rate gyro	0	0	0	0
Roll rate gyro	0	0	0	0
Yaw rate	0	0	0	0
Longitudinal accelerometer	0	0	0	0
Lateral accelerometer	0	1	2	226
Normal accelerometer	0	0	0	6
Altitude	1	0	5	267
Mach	0	0	0	0
Pitch attitude	1	0	2	10
Roll attitude	0	0	0	72
Heading angle	0	4	0	9
Pitch center stick	0	0	0	0
Roll center stick	0	0	0	0
Rudder pedals	0	0	0	0
Pitch side stick	0	0	0	0
Roll side stick	0	0	0	0
Angle of attack	1	0	0	7

TABLE 12.—POSITION SENSOR TRANSDUCER EXPERIENCE

Position sensor	Range	Fault threshold	Worst sensor pair deviation		
			Flight deviation	Moderate frequency ground test	High frequency ground test
Pitch center stick, cm	15	1	0.07	0.20	0.46
Roll center stick, cm	10	1	0.08	0.11	0.27
Rudder pedal, cm	7	0.5	0.04	0.05	0.28
Pitch side stick, N	110	2.7	----	0.12	1.80
Roll side stick, N	70	4.5	----	0.76	3.60

TABLE 13.—NOMINAL ARM SENSOR PARAMETERS

Sensor	Bias failure magnitude, BFM	Window size for fault detection, sec
Mach	0.045	0.18
Altitude, m	30.5	0.30
Angle of attack, deg	2.0	0.54
Longitudinal accelerometer, g	0.15	1.02
Lateral accelerometer, g	0.15	1.02
Normal accelerometer, g	0.15	1.02
Roll rate, deg/sec	3.55	0.18
Pitch rate, deg/sec	2.3	0.12
Yaw rate, deg/sec	2.3	0.12
Roll attitude, deg	4.6	0.36
Pitch attitude, deg	2.12	0.36
Heading angle, deg	1.7	0.96

TABLE 14.—TYPE AND MAGNITUDE OF INDUCED FAULTS

Sensor	Bias	Drift rate, per sec	Scale factor	Hardover amplitude	Pulse amplitude (0.18 sec)
Mach	0.0675	0.045	0.75	2	0.096
Altitude	45	30	----	27,000	90
Angle of attack, deg	3	2	0.75	25	6.88
Longitudinal acceleration, g	0.23	0.15	----	2	0.3
Lateral acceleration, g	0.23	0.15	----	2	0.3
Normal acceleration, g	0.23	0.15	0.75	10	0.3
Roll rate, deg/sec	5.32	3.55	0.75	240	69
Pitch rate, deg/sec	3.44	2.3	0.5	70	17
Yaw rate, deg/sec	3.44	2.3	----	70	17
Roll attitude, deg	6.88	4.6	----	180	41
Pitch attitude, deg	3.18	2.1	----	90	10
Heading angle, deg	22.5	15	----	180	36

TABLE 15.—AVERAGE TIMES TO FAULT ISOLATION

Sensor	Bias	Drift	Scale factor	Hardover	Pulse	Open line
	Average time to fault isolation, sec					
Mach	0.08	Not tested	0.02	0.02	0.06	0.02
Altitude	0.14	Not tested	Not tested	0.02	0.06	0.02
Angle of attack	0.86	2.04	Not isolated	0.18	Not isolated	0.24
Longitudinal accelerometer	8.64	4.44	Not tested	0.90	Not isolated	Not isolated
Lateral accelerometer	0.24	0.84	Not tested	0.06	Not isolated	Not isolated
Normal accelerometer	4.38	4.56	4.32	0.42	Not isolated	1.26
Roll rate	0.66	1.56	Not isolated	0.12	0.18	Not isolated
Pitch rate	0.82	1.74	Not isolated	0.18	0.48	Not isolated
Yaw rate	0.80	1.68	Not tested	0.18	0.48	Not isolated
Roll attitude	0.16	Not tested	Not tested	0.02	0.02	Not isolated
Pitch attitude	0.18	Not tested	Not tested	0.02	0.02	0.06
Heading angle	0.12	Not tested	Not tested	0.02	0.08	0.02

TABLE 16.—FDIR TEST MATRIX

Location of induced faults	Approximate number of faults	Types of faults induced
Interface unit	200	Input/output data lines Crosslink communication lines Synchronization lines Loss of entire card Loss of entire connector Loss of power
Encoder/decoder	400	Data lines Strobe lines Clock lines
Computer hardware	20	Input/output connector Spurious interrupts Loss of power
Computer software	150	Faults producing program interrupts Faults producing machine interrupts Input/output communication errors Vital sign errors

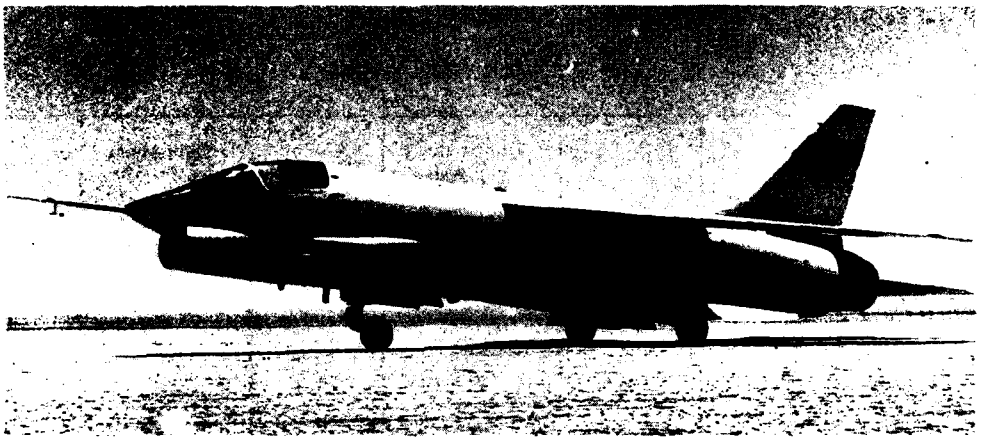


Figure 1. F-8 DFBW airplane.

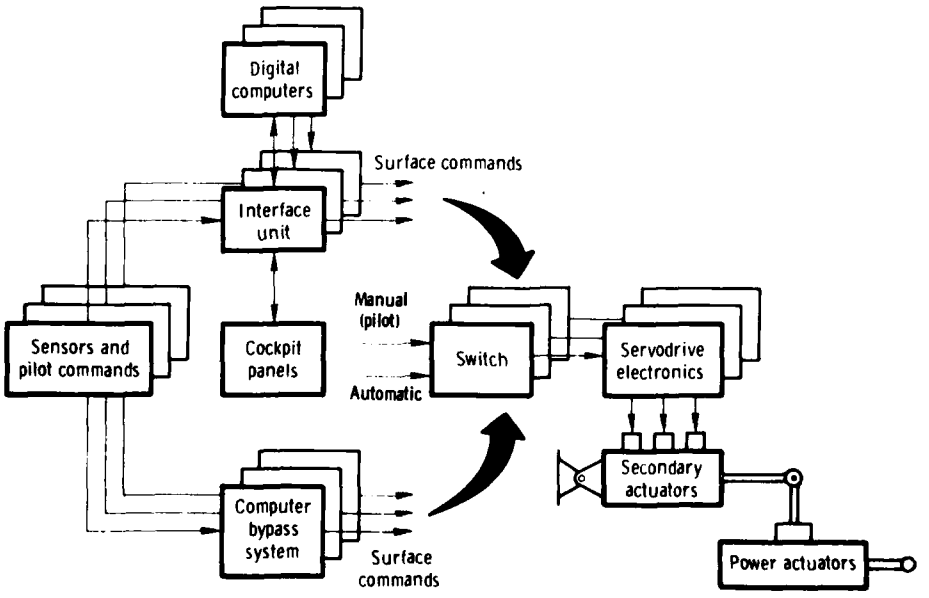


Figure 2. F-8 DFBW control system mechanization.

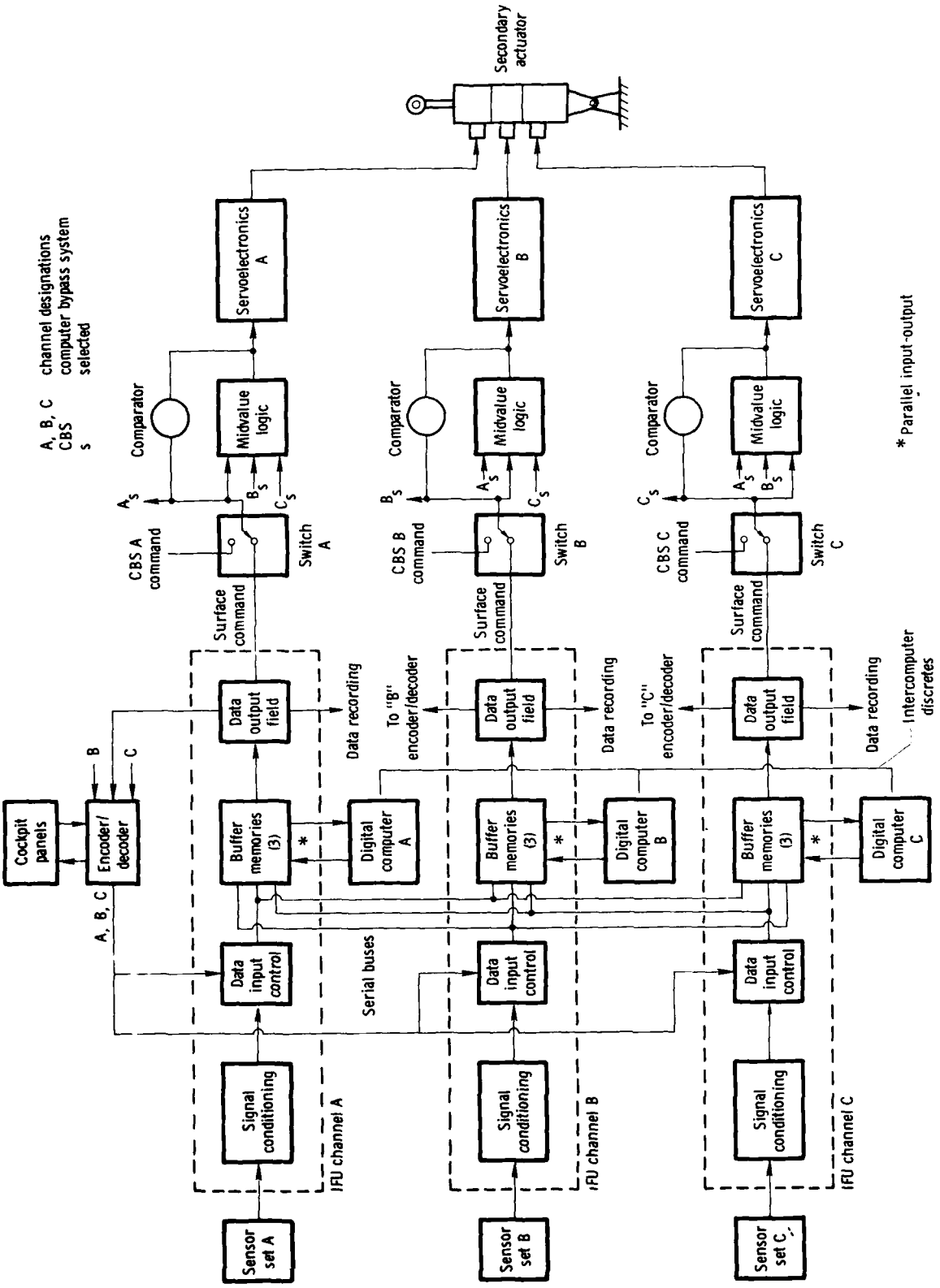
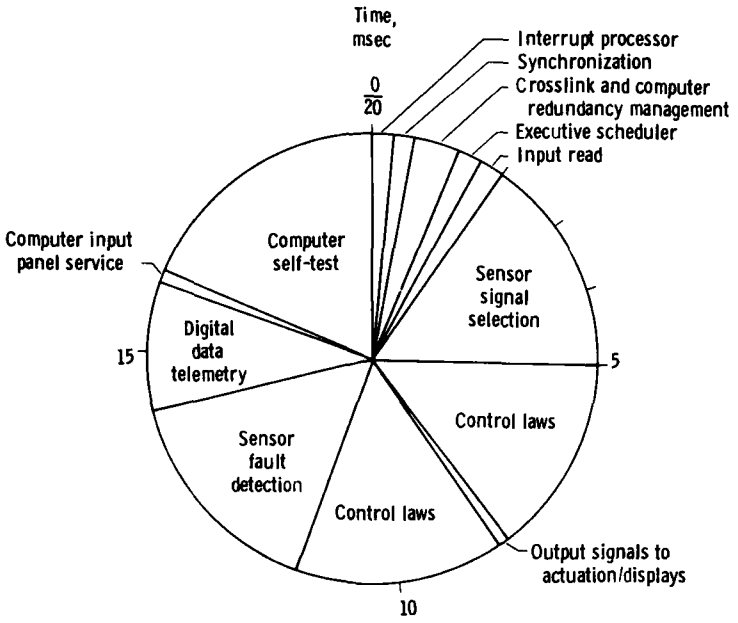
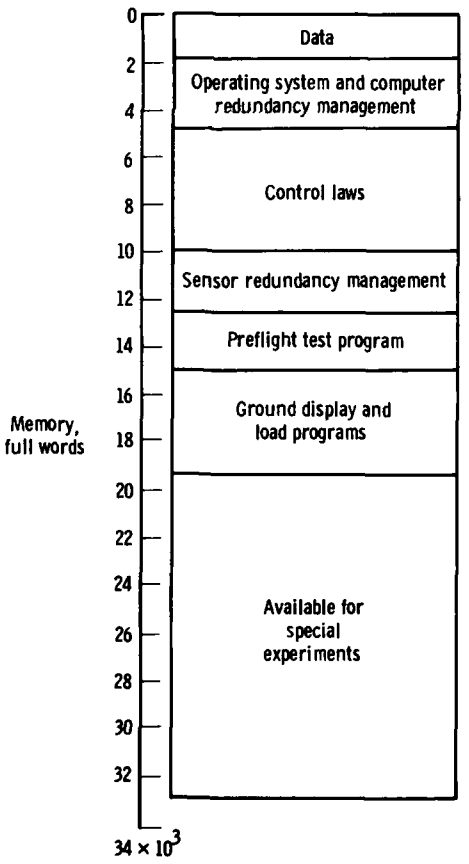


Figure 3. F-8 DFBW primary digital system mechanization.



(a) Timing sequence with all control modes active. 20-msec minor cycle.



(b) Memory allocation.

Figure 4. F-8 DFBW software system.

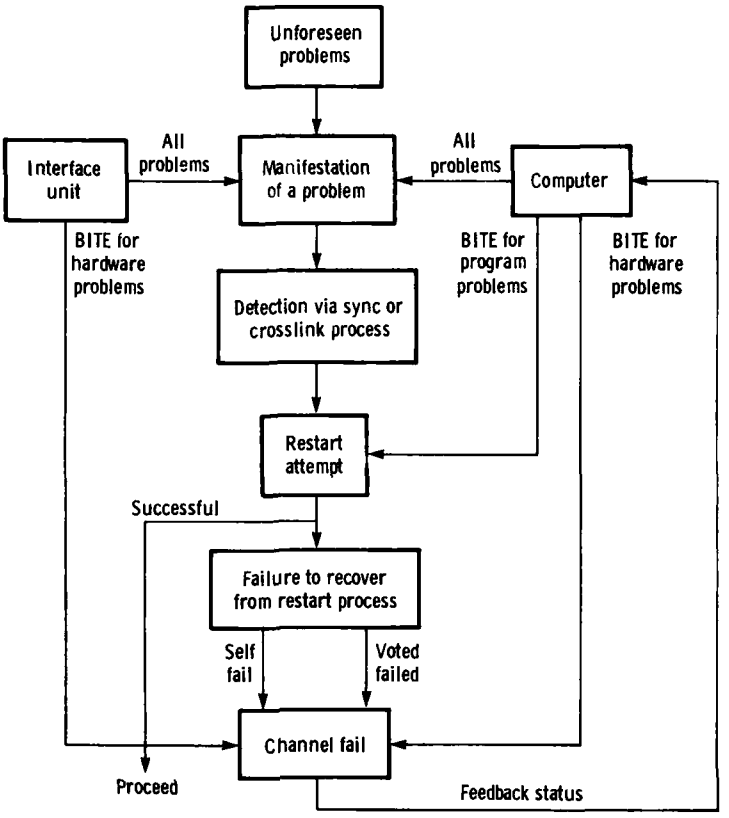


Figure 5. Computer FDIR hierarchy.

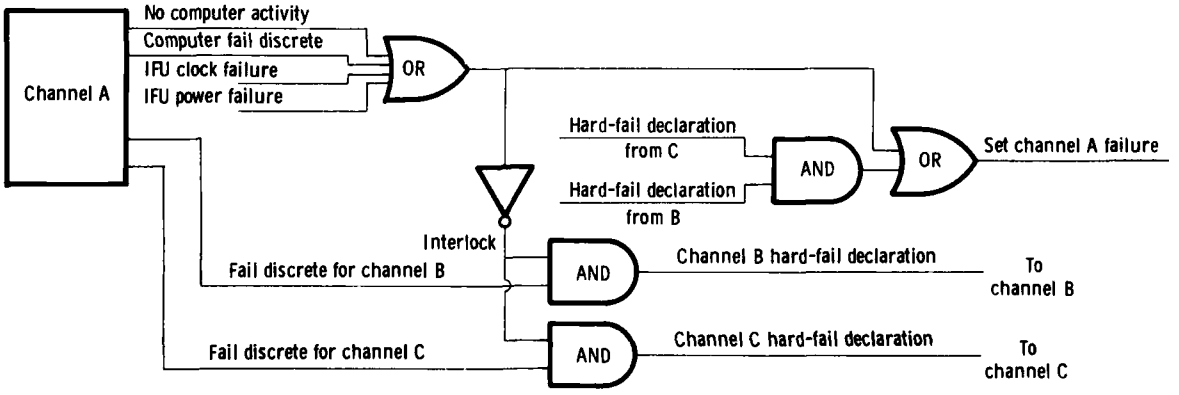


Figure 6. Functional diagram of fault-detection hardware in IFU.

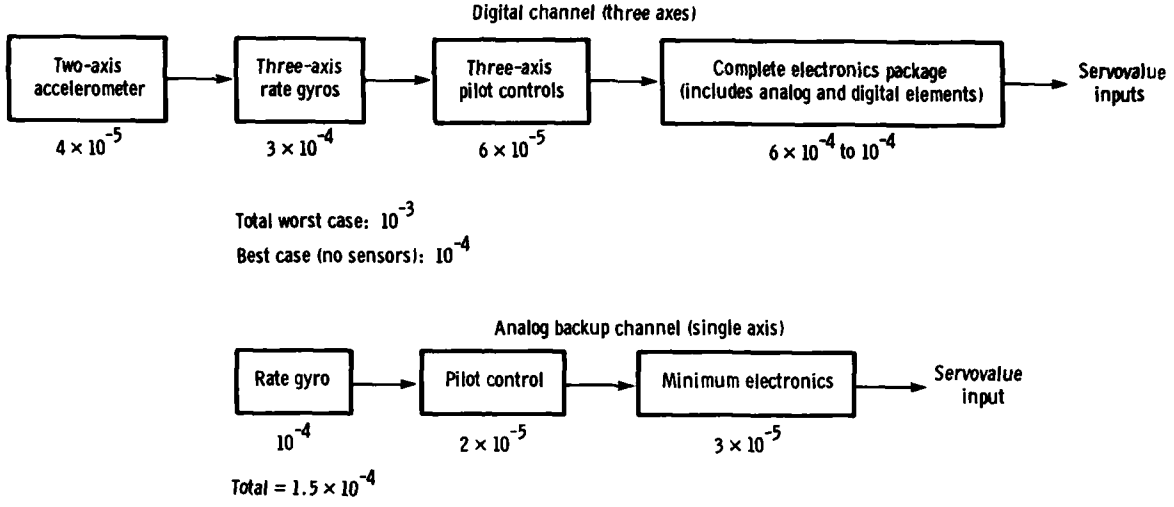


Figure 7. Probability analysis for flight control system reliability assessment. Failure rates per hour are indicated for each subsystem.

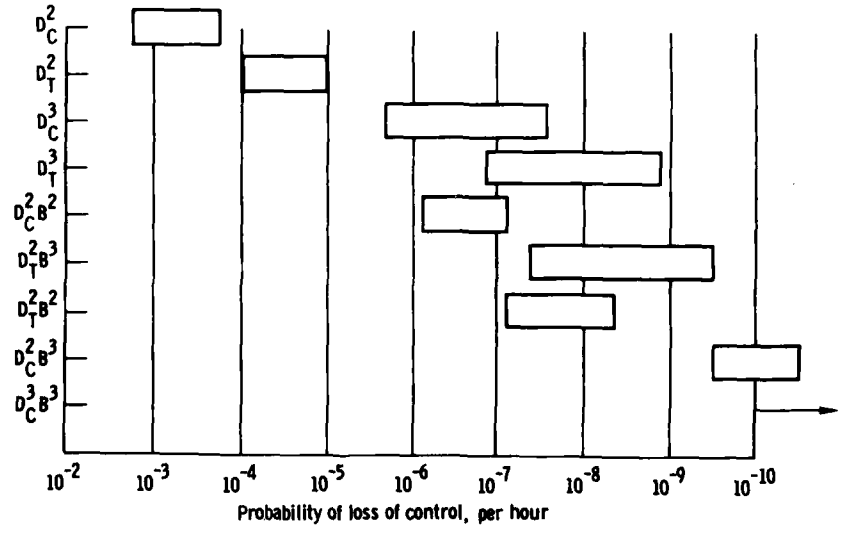


Figure 8. Probability of total system failure for different configurations. D_C denotes digital channel with comparison monitoring; D_T denotes digital channel with 95 percent self-test; B denotes backup analog channel. Superscript denotes number of channels; thus, $D_C^2B^2$ denotes dual digital with comparison monitoring and dual analog backup.

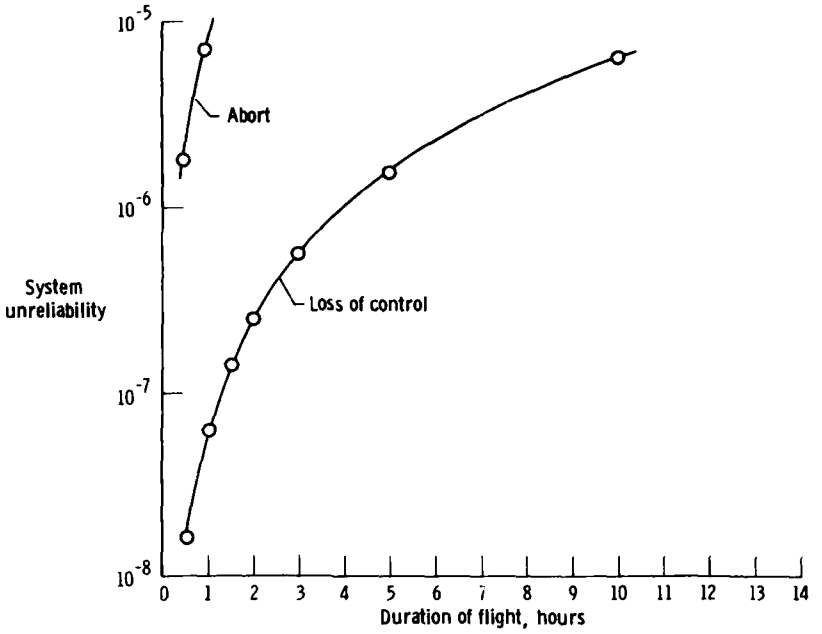


Figure 9. F-8 DFBW system reliability analysis. Loss of control and abort probabilities.

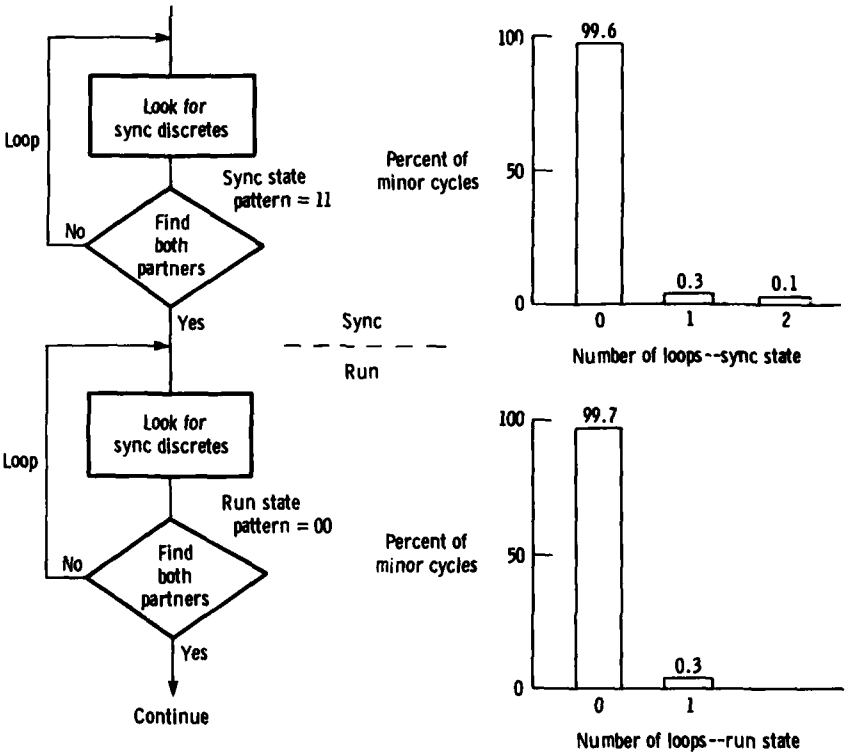


Figure 10. Synchronization performance.

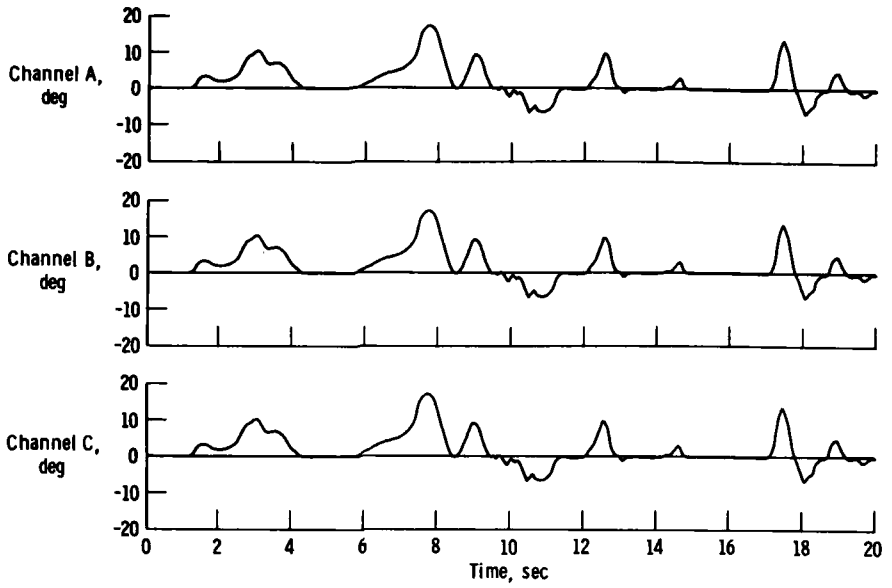


Figure 11. Triplex aileron command tracking.

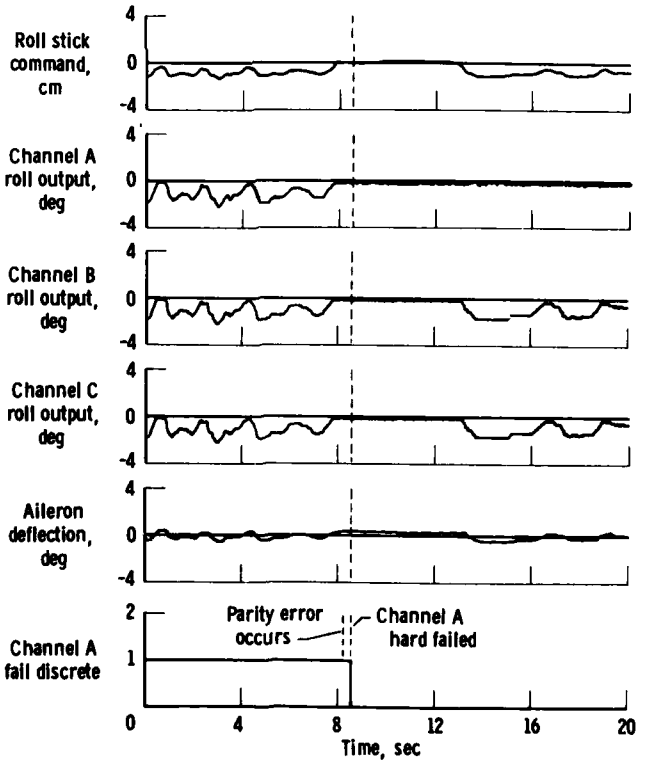


Figure 12. In-flight computer failure.

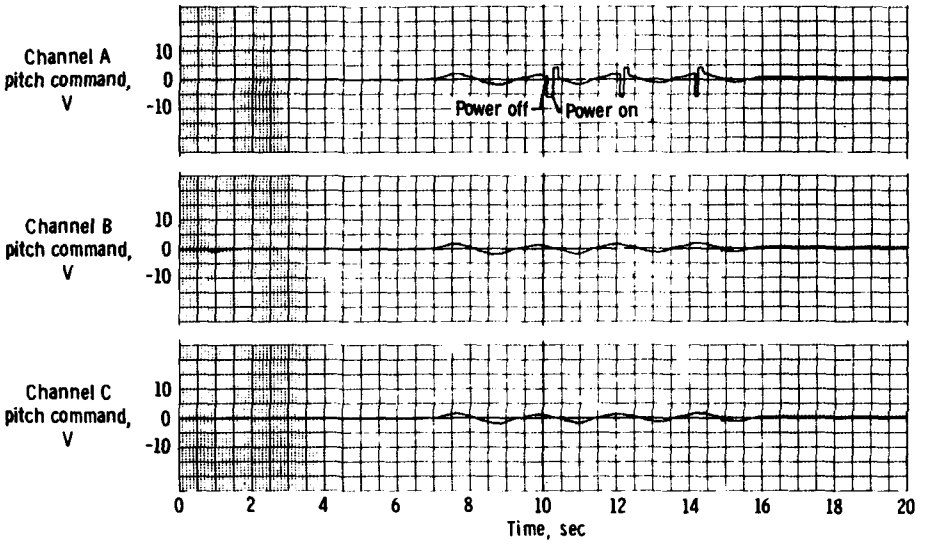


Figure 13. Power-failure-induced restarts during iron bird tests.

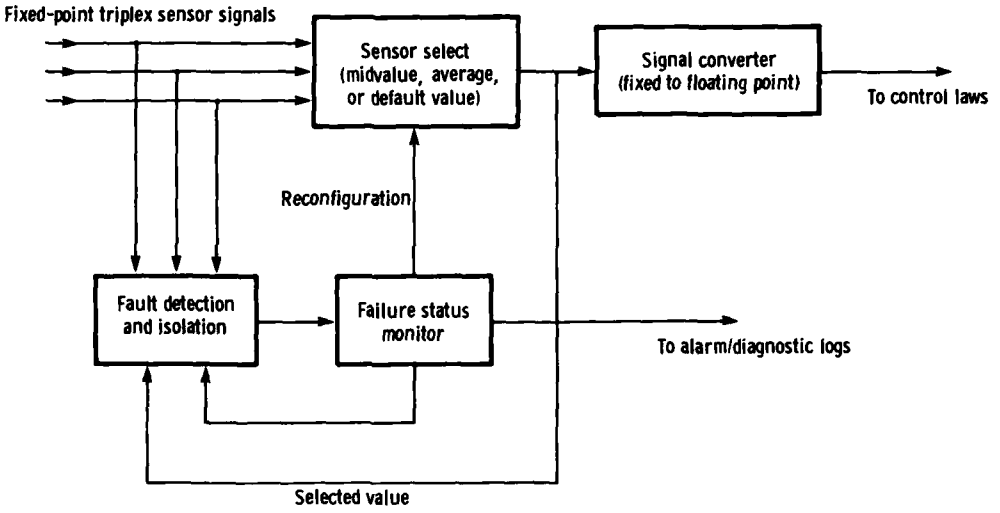
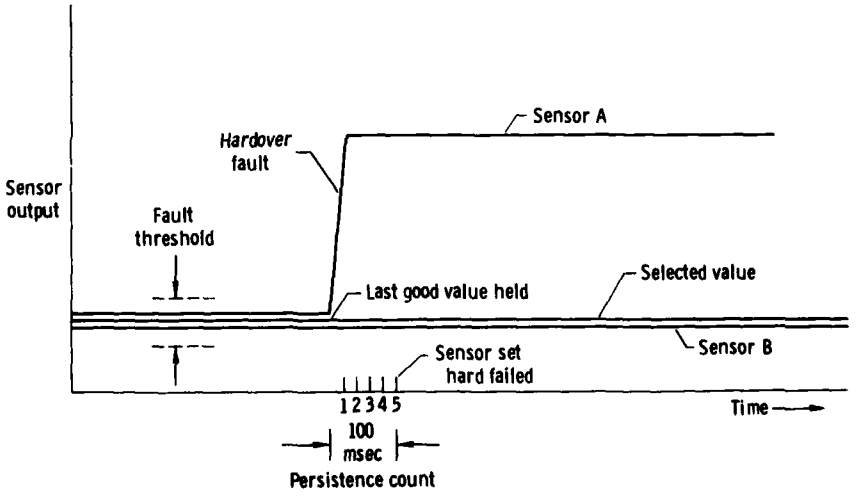
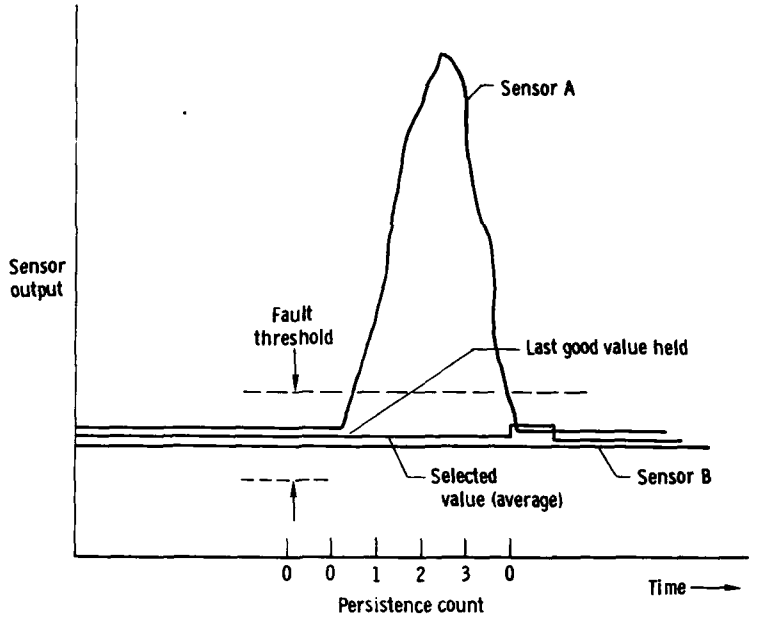


Figure 14. Analog sensor fault detection isolation and reconfiguration algorithm.



(a) Hardover failure of sensor A.

Figure 15. Dual sensor FDIR performance.



(b) Transient fault of sensor A.

Figure 15. Concluded.

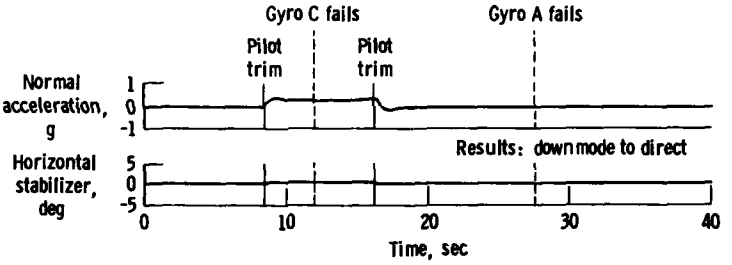


Figure 16. Successive hardover pitch-rate-gyro-induced failures. Pitch stability augmentation system mode.

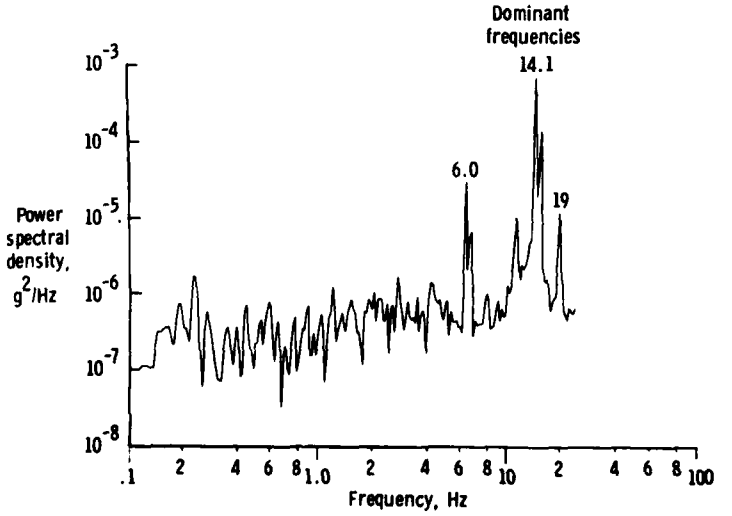
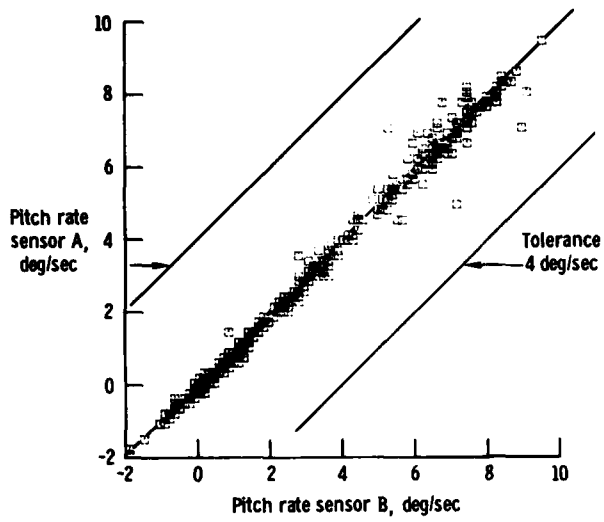
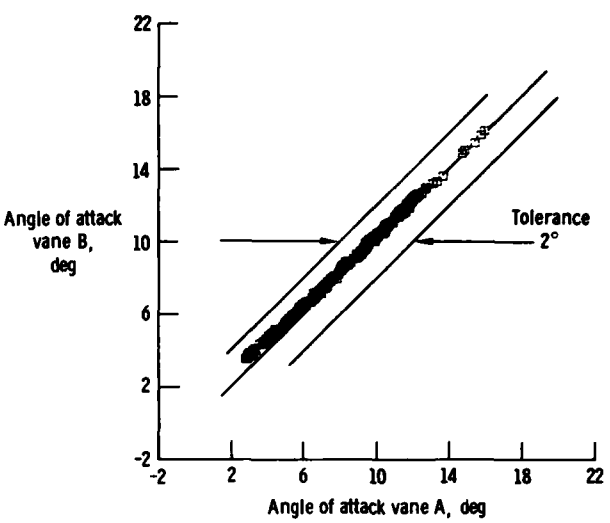


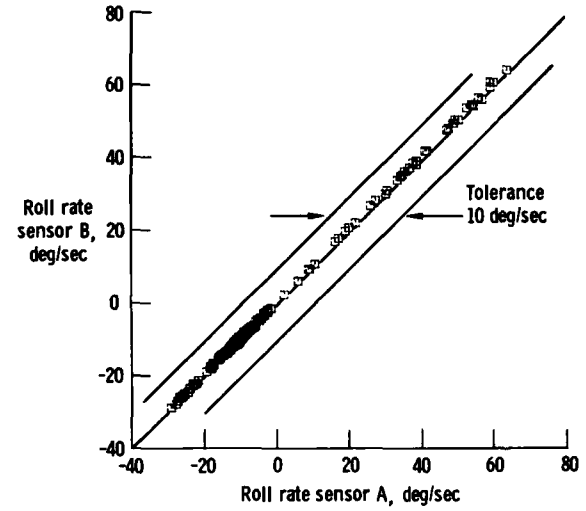
Figure 17. Pitch rate gyro power spectral density. Engine is at 80 percent rpm.



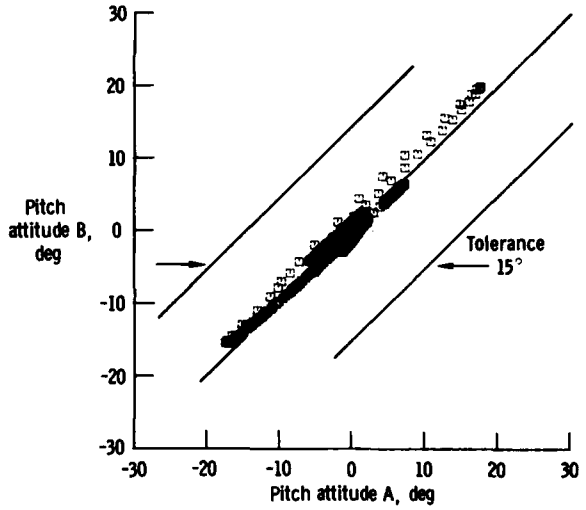
(a) Pitch rate.



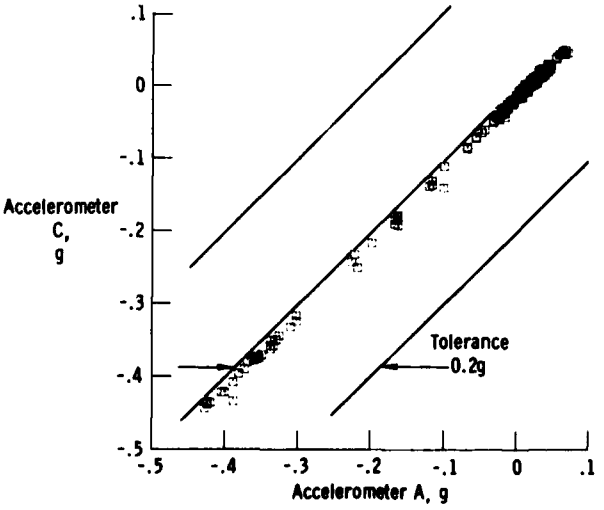
(d) Angle of attack.



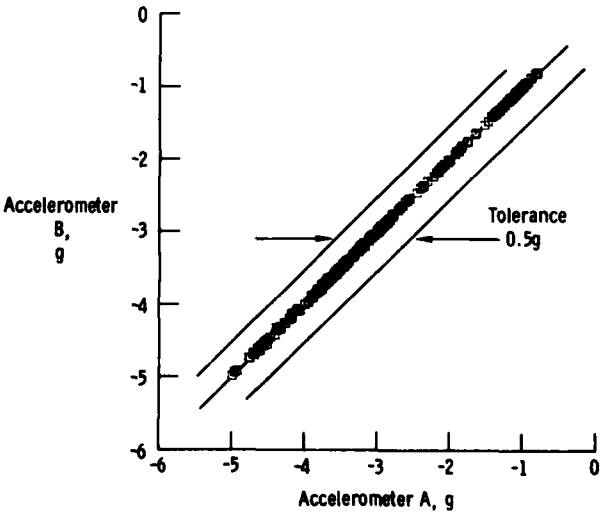
(b) Roll rate.



(e) Pitch attitude.



(c) Lateral acceleration.



(f) Normal acceleration.

Figure 18. Sensor tracking during maneuvering flight.

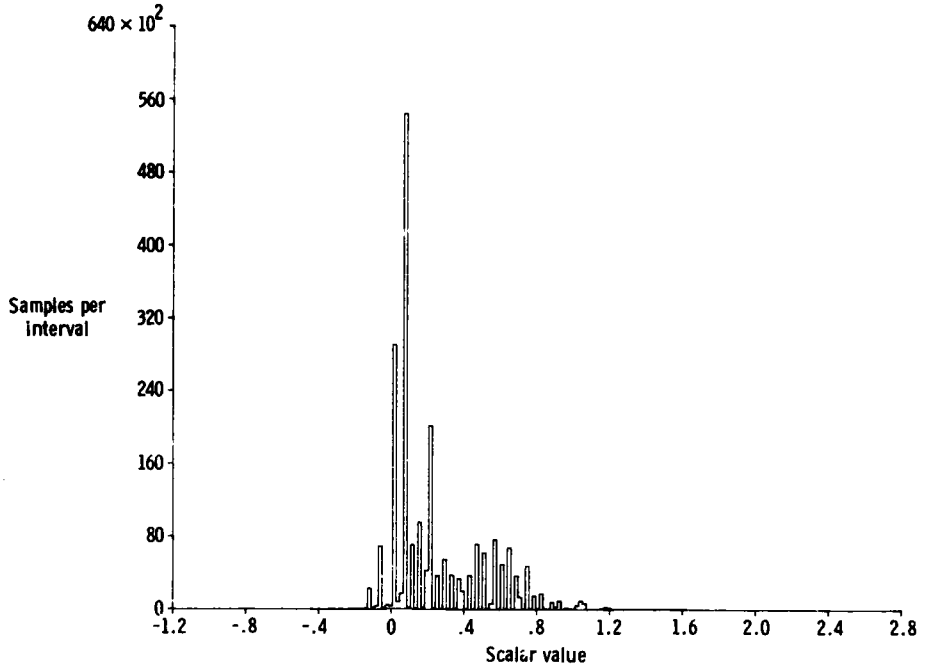
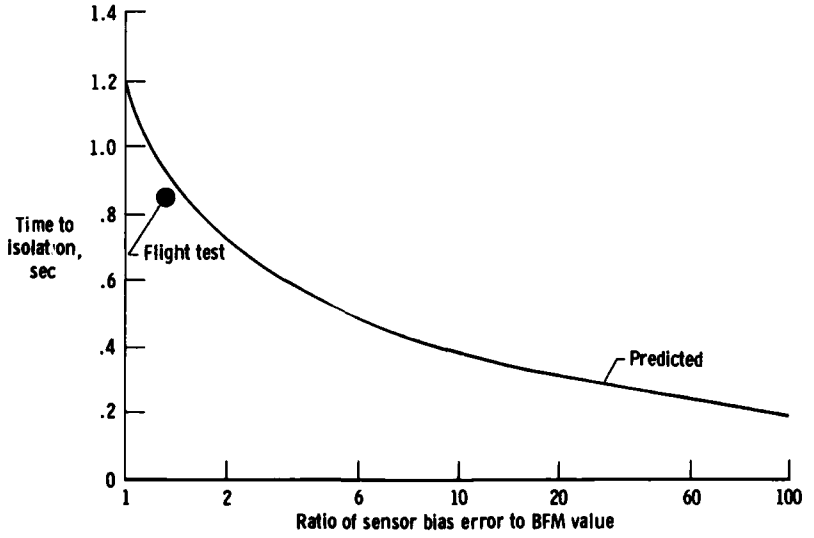
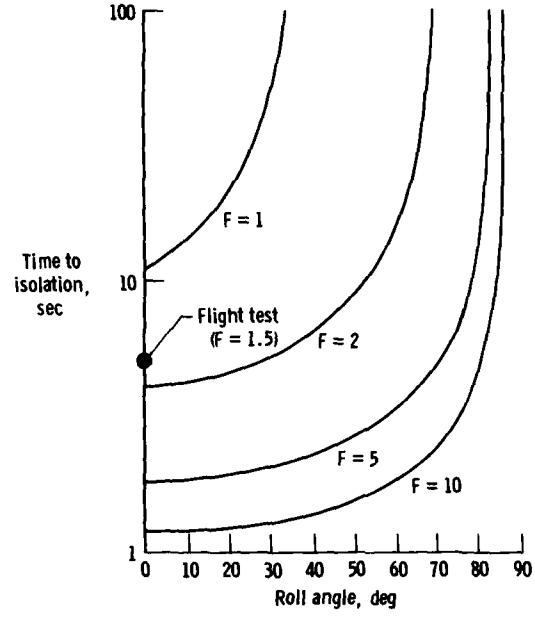


Figure 19. Histogram of yaw rate gyro pair (A, C).



(a) Pitch rate gyro with bias failure magnitude of 1.5 times BFM.

Figure 20. Fault isolation time.



(b) Normal accelerometer with bias failure magnitude of 0.23 g. $F = \frac{\textit{Fault magnitude}}{BFM}$.

Figure 20. Concluded.

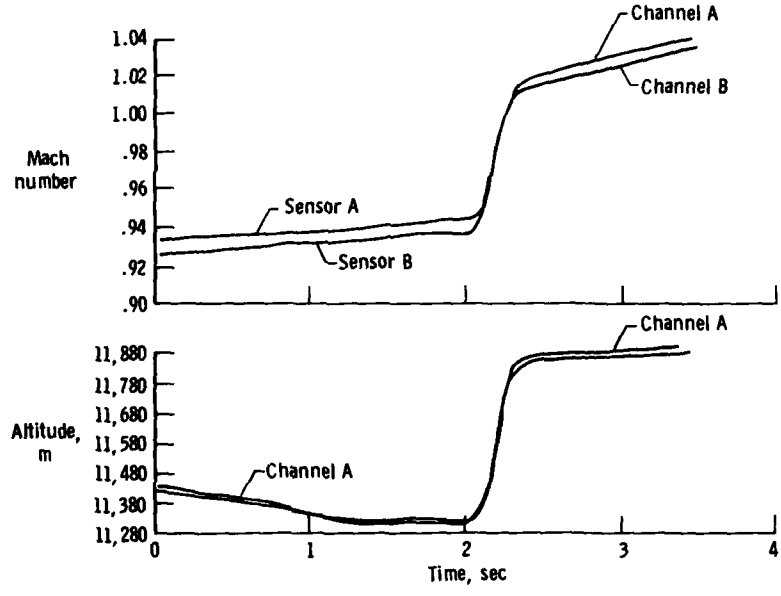


Figure 21. Mach and altimeter transonic jump.

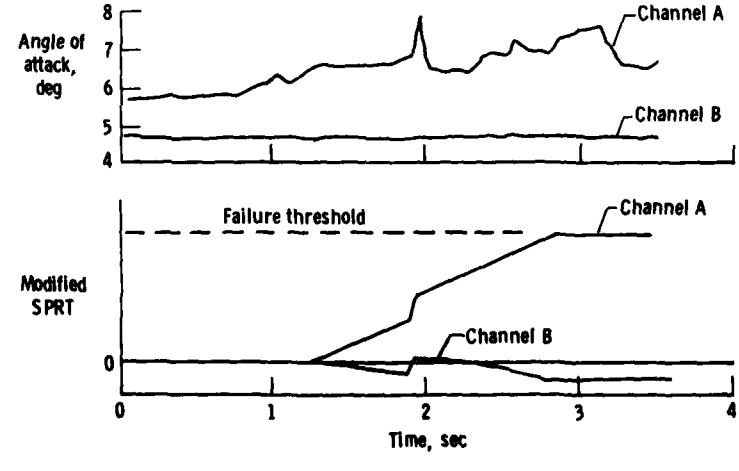


Figure 22. In-flight angle-of-attack vane fault isolated by ARM algorithm.

ROBUST CONTROL SYSTEM DESIGN

by

J. Ackermann
Director of DFVLR Institute for Flight System Dynamics
DFVLR Oberpfaffenhofen
8031 Wessling
W.-Germany

SUMMARY

The short period longitudinal mode of an F 4-E with horizontal canards is unstable in subsonic flight and unsufficiently damped at supersonic speed. The control system has to provide acceptable pole locations according to military specifications for flying qualities. A fixed gain controller using three paralleled gyros is designed, such that the pole region requirements in four typical flight conditions are robust with respect to gain reduction to one third. Thus nothing bad happens immediately after one or two gyro failures. Failure detection and redundancy management may be performed at a higher hierarchical level, which does not have to be extremely fast. The use of accelerometers or air data sensors for angle of attack or dynamic pressure is totally avoided in this concept and no gain scheduling is necessary. The design for robustness with respect to different flight conditions and sensor failures is performed by a novel parameter space design tool.

I. INTRODUCTION

Redundancy management in control systems is usually viewed separately from the control algorithm. The control system is designed under the assumption, that sensors do not fail. Then redundancy management has to provide the required measurements with only very short interruptions by failures of individual sensors. If the plant is for example an unstable aircraft, this means that failure detection is vital for stabilization, it has to operate fast and this requirement is in conflict with the requirement of low probability of false alarms.

In this paper a hierarchical system is proposed. Its basic level is a fixed gain control system, which is designed such, that pole region requirements are robust with respect to component failures. All more sophisticated tasks like failure detection and redundancy management, plant parameter identification and controller parameter adaptation or gain scheduling are assigned to higher levels, if they are required for best performance. The higher levels processes more information and are operating in a slower time scale than the basic level. Since the higher levels are not vital for stabilization they can make their decisions without panic haste.

This paper deals with the design of the robust basic level control system. The particular example is an F 4-E, which is destabilized by horizontal canards, see Fig. 1. Only the short period longitudinal mode is considered, i.e. second order dynamics. The actuator is modelled as a first order low pass with transfer function $14/(s + 14)$, its state variable is δ_e, the deviation of the elevator deflection from its trim position. δ_e is not fed back, because this would require an estimate of the trim position.

In a previous study [1], [2] measurement of normal acceleration N_z and pitch rate q is assumed and the linearized state equations are written in sensor coordinates with the state vector $\underline{x}^T = [N_z \quad q \quad \delta_e]$. Thus

$$\dot{\underline{x}} = \underline{A}\,\underline{x} + \underline{b}\,u \qquad (1)$$

$$\underline{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 14 \end{bmatrix} \qquad \underline{b} = \begin{bmatrix} b_1 \\ 0 \\ 14 \end{bmatrix}$$

Data for the four typical flight conditions of Fig. 2 were taken from [3] and are given in the appendix. The eigenvalue locations of the short period mode are given in table 1.

Table 1

FC	MACH	ALTITUDE	OPEN LOOP SHORT PERIOD EIGENVALUES
1	0.5	5000'	-3.07 1.23
2	0.85	5000'	-4.90 1.78
3	0.9	35000'	-1.87 0.56
4	1.5	35000'	-.87 ± j4.3

The aircraft is unstable in subsonic flight and unsufficiently damped in supersonic flight, such that adequate handling properties must be provided by the control system.

Note that in stationary flight the elevator and canard are not used independently. The commanded deflections are coupled as

$$\delta_{ecom} = u$$

$$\delta_{ccom} = -0.7u$$

where the factor -0.7 was chosen for minimum drag. Thus the short period mode stabilization is a single-input problem.

The required closed loop eigenvalue locations are given by military specifications for flying qualities of piloted airplanes [4]. For the short period mode described by

$$s^2 + 2\zeta_{sp}\omega_{sp}s + \omega_{sp}^2 = 0 \tag{2}$$

the restricted range of damping ζ_{sp} and natural frequency ω_{sp} is

$$0.35 \leq \zeta_{sp} \leq 1.3 \tag{3}$$

$$\omega_a \leq \omega_{sp} \leq \omega_b$$

where ω_a and ω_b depend on the flight condition and are given in the appendix for the four conditions considered here.

Fig. 3 shows the nominal region Γ_j, eq.(3) together with the open loop eigenvalues for a subsonic flight condition j. Damping greater than one in eq.(3) corresponds to two real eigenvalues. Eq.(3) would admit some real pairs of poles with one of them outside the region Γ_j. In the following no use is made of this possibility. For all real pairs inside Γ_j condition (3) is satisfied. We require, that the closed loop short period poles of each flight condition j = 1, 2, 3, 4 are located in the respective region Γ_j.

The military specifications do not contain requirements for the location of additional closed loop poles originating from actuator or feedback dynamics. Quick response is essential for a fighter,therefore the non short period eigenvalues should not unnecessarily slow the dynamic response. In order to keep them fast enough and separate from short period eigenvalues an additional region to the left of Γ_j is prescribed. The damping requirement $\zeta \geq 0.35$ is kept from eq.(3) and a natural frequency range $\omega_b \leq \omega \leq \omega_d$, ω_d = 70 rad/sec is chosen in order to maintain a bandwidth limitation below the first structural mode frequency. The extended region is shown in Fig. 3.

The assumed type of sensor failure is that the nominal gain v = 1 is reduced to some value $0 \leq v < 1$ and an additional bias or noise term is added at the sensor output. As far as eigenvalue location is concerned, only the gain reduction to zero is important.

The objective of this paper is to design the basic level control system such that the pole region requirements of Fig. 3 are robust with respect to changing flight conditions and sensor failures. A novel "$\mathcal{K}$-space" design technique [5] is applied in this design. It will be reviewed briefly in the following paragraph. In application to the example it is then shown, how robustness with respect to changing flight conditions can be achieved by appropriate choice of k_{Nz} and k_q in an output feedback control law

$$u = -[k_{Nz} \quad k_q \quad 0]\,\underline{x} \tag{4}$$

For robustness with respect to sensor failures in [1], [2] a configuration with two gyros and one accelerometer and dynamic feedback was studied. It showed the disadvantage of using the accelerometer. Therefore here a different solution with three gyros and dynamic feedback is given. For this solution the responses in C* for a pilot step input are given, where

$$C^* = (N_z + 12.43q)/C_\infty \tag{5}$$

The stationary value C_∞ is used for normalization.

II POLE REGION ASSIGNMENT

The most essential aspect of $\mathcal{K}$-space design is pole region assignment. Other features will be discussed later using the design example. If a tradeoff with other design requirements has to be made it is not satisfactory to find one solution, for which all eigenvalues are in their respective regions in s-plane, e.g. by pole placement or root locus techniques. It is desirable to find all such solutions. This is achieved by mapping the region Γ in s-plane into a region P_Γ in the parameter space $\mathcal{P}$ of coefficients of the desired characteristic polynomial first. Then P_Γ is mapped into a corresponding region K_Γ in the parameter space of feedback gains. The first step only deals with properties of polynomials

$$\begin{aligned} P(s) &= p_o + p_1 s + \ldots + p_{n-1} s^{n-1} + s^n \\ &= [\underline{p}^T \quad 1]\,[1 \quad s \ldots s^n]^T \\ &= \prod_{i=1}^{n} (s - s_i) \end{aligned} \tag{6}$$

The problem is: Find the region P_Γ in $\mathcal{P}$ space such that $\underline{p}^T \in P_\Gamma$ if and only if $s_i \in \Gamma$ for $i = 1, 2 \ldots n$. The boundaries of P_Γ for a connected region Γ with two real axis intersections at σ_L and σ_R consists of three parts corresponding to the cases that a real eigenvalue crosses the boundary in s-plane at σ_L or at σ_R or a complex conjugate pair crosses the complex boundary. For the real values these boundaries in $\mathcal{P}$ space are the n-1 dimensional hyperplanes $P(\sigma_L) = 0$ and $P(\sigma_R) = 0$.
For the complex case

$$\begin{aligned} P(s) &= (s - \sigma - j\omega)\,(s - \sigma + j\omega) \cdot R(s) \\ &= [s^2 - 2\sigma s + \sigma^2 + \omega^2(\sigma)] \cdot R(s) \end{aligned} \tag{7}$$

If for example the boundary $\omega^2(\sigma)$ is a circle with real center σ_o and radius r, i.e. $(\sigma - \sigma_o)^2 + \omega^2 = r^2$, then $P(s) = [s^2 - 2\sigma s + r^2 + 2\sigma_o\sigma - \sigma_o^2]\,R(s)$. For a fixed $R(s)$, $\underline{p}^T$ depends linearly on σ, i.e. in $\mathcal{P}$ space a straight line segment from a point on $P(\sigma_L) = 0$ (with a double root at σ_L) to a point on $P(\sigma_R) = 0$ (with a double root at σ_R) results. As the n-2 coefficients of R(s) vary, this straight line is moved and forms the complex boundary in $\mathcal{P}$-space. For $n = 3$ this is illustrated by Fig. 4. If $\underline{p}$ crosses the plane containing the triangle ABC, then a real root crosses the unit circle at $s = -1$ and analogously for the triangle BCD and $s = 1$. If $\underline{p}$ crosses the hyperbolic paraboloid, which is formed by a family of straight lines, then a complex conjugate pair of eigenvalue crosses the unit circle. If $\omega^2(\sigma)$ is a conic section

$$\omega^2(\sigma) = c_o + c_1\sigma + c_2\sigma^2 \tag{8}$$

then the image for a fixed R(s) is a conic section instead of the straight line above. The complex boundary may be defined piecewise as in Fig. 3.

The complex and the two real boundaries partition the $\mathcal{P}$ space into regions distinguished by the location of the eigenvalues relative to Γ.

In the second step a controller structure is assumed, e.g. state feedback

$$u = -\underline{k}^T \underline{x}\,, \quad \underline{k}^T = [k_1 \quad k_2 \ldots k_n] \tag{9}$$

and the region P_Γ is mapped into a region K_Γ in the controller parameter space $\mathcal{K}$ with coordinates $k_1, k_2 \ldots k_n$ such that $\underline{k}^T \in K_\Gamma$ if and only if $\underline{p}^T \in P_\Gamma$. It was shown in [1], that for state feedback, eq.(9), this is accomplished by an affine mapping

$$\underline{k}^T = [\underline{p}^T \quad 1]\,\underline{E} \tag{10}$$

where the pole assignment matrix $\underline{E}$ describing the plant is determined by a controllable pair $\underline{A}$, $\underline{b}$ as follows:

Let $\underline{R} = [\underline{b} \ \underline{A}\underline{b} \ \dots \ \underline{A}^{n-1}\underline{b}]$

$\underline{e}^T = [0 \ \dots \ 0 \ \ 1] \ \underline{R}^{-1}$

Then

$$\underline{E} = \begin{bmatrix} \underline{e}^T \\ \underline{e}^T\underline{A} \\ \vdots \\ \underline{e}^T\underline{A}^n \end{bmatrix} \qquad (11)$$

By this affine mapping hyperplanes remain hyperplanes and conic sections remain conic sections. Thus all principal properties of the regions can be studied in the canonical parameter space $\mathcal{P}$. A system $(\underline{A},\underline{b})$ is interpreted as an affine mapping from $\mathcal{P}$-space to $\mathcal{K}$-space.

For each pair $\underline{A}_j$, $\underline{b}_j$, i.e. for each flight condition, a different mapping $\underline{E}_j$ results and the solution set is the intersection of the regions $K_{\Gamma j}$ in $\mathcal{K}$-space. Graphical representation of such regions is easy for n = 2 and possible for n = 3 by computer graphics. For higher system orders the design may proceed stepwise by fixing n-2 gains in each step, also for output feedback some gains are fixed. In the aircraft example n = 3 and $k_z = 0$, i.e. we are looking at a two dimensional cross-section of the three-dimensional region K_Γ.

By eq.(10) each fixed gain k_i implies a linear relationship $k_i = [\underline{p}^T \ \ 1] \ \underline{n}_i$, where $\underline{n}_i$ is the ith column of $\underline{E}$. These n-2 linear equations can be used to express the two free gains by coefficients of a second order factor of the characteristic polynomial, which is varied along the boundary in s-plane to produce the boundary in the plane of the two free gains[2]. This tool will be applied to the aircraft example in the next section.

III ROBUSTNESS WITH RESPECT TO FLIGHT CONDITION

The first design objective will be to design an output feedback controller, eq.(4), which meets the nominal pole region requirements at all four flight conditions.

The boundary for flight condition 2 is shown in Fig. 5. On a-b eigenvalues are on the lower natural frequency boundary $\omega_{sp} = 3.5$, on b-c they are on the damping 0.35 lines. At c-a real root boundary takes over: on c-d the actuator eigenvalue is at $\sigma = -70$. On d-e a real short period eigenvalue is at the upper natural frequency limit $\sigma = -12.6$ and for e-a the actuator eigenvalue is at $\sigma = -12.6$. The condition for having no real real root $\sigma = -3.5$ is satisfied in the total region. This region R_{nom2} is bounded by two straight lines c-d and d-a resulting from real root conditions and by the two complex boundary curves a-b and b-c. Note that the boundaries in s-plane are conic sections and thus a-b and b-c are segments of conic section also.

The regions R_{nom1} - R_{nom4} for the other flight conditions were found by mapping the eigenvalue constraints for each flight condition into the k_{Nz} - k_q-plane. These four regions have the intersection R_{nom} shown in Fig. 6. Thus robustness with respect to changing flight conditions can be achieved by static output feedback of the accelerometer and gyro signals. More precisely: All eigenvalues at all four flight conditions are in their prescribed regions in s-plane if and only if the pair k_{Nz}, k_q is chosen in the region R_{nom}.

As an example choose the design point Q_1, i.e. $k_{Nz} = -0.115$, $k_q = -0.8$. The closed loop eigenvalues are given in table 2.

Table 2

FC	Short period eigenvalues		Actuator eigenvalue
	damping	natural frequency	
1	0.94	4.68	- 18.31
2	0.61	9.18	- 37.29
3	0.79	4.63	- 17.78
4	0.55	8.11	- 27.04

The selection of a design point in R_{nom} is a tradeoff, in which the designer learns, which requirements are conflicting. E.g. structural vibrations are most critical in flight condition 2 (high speed, low altitude). They can be reduced by avoiding the vicinity of the $\sigma_2 = -70$ boundary. Low damping is most critical at the supersonic flight condition 4. Damping can be increased by avoiding the vicinity of the $\zeta_4 = 0.35$ boundary. Sluggish responses in landing approach can be avoided by avoiding the vicinity of the $\sigma_1 = -2.02$ boundary. The $\sigma_1 = -7.23$ boundary is only necessary in order to separate actuator and short period poles, the design point may be chosen close to this boundary.

IV ROBUSTNESS WITH RESPECT TO SENSOR FAILURES

As far as stability is concerned, a failure of the accelerometer(gyro) is equivalent to a reduction of k_{Nz} (k_q) from the nominal value to zero or some value in between. Fig. 6 shows that the nominal region does not intersect the axes, thus in the assumed output feedback structure it is not possible to maintain nominal specifications after their failure.

Fig. 6 shows however that a considerable gain reduction inside R_{nom} is admissible, if k_{Nz} and k_q are reduced simultaneously. This can be achieved by replacing the accelerometer measurement N_z by an estimate $\hat{N}_z$, which is produced by a filter from q. It is not necessary that this is a true estimate, e.g. generated by an adaptive observer. It is sufficient, that this is a constant filter connected to q such that the frequency response from u to the filter output $\hat{N}_z$ is an approximation to the frequency response from u to N_z for an average over the four flight conditions. Of course separation does not hold, i.e. we can not take the same pair of feedback gains k_{Nz}, k_q as in the case of accelerometer measurement. However this consideration leads to a structure of the feedback system with a two dimensional signal basis q and $\hat{N}_z$, and a new exact determination of admissible regions in the plane of the two feedback gains k_{Nz}, k_q, see Fig. 7 , can be made.

Both transfer functions from u to N_z and to q have the same denominator, thus the filter has to cancel approximately the zeros in the q-channel and to replace them by the averaged zeros of the N_z channel. Table 3 shows the zeros and gain ratios of the transfer functions at the four flight conditions.

Table 3

Open Loop Zeros and Gain Ratio

FC	MACH	ALTITUDE	q-ZERO	Nz-ZEROS	K_N/K_Q
1	.5	5000'	- .884	-.542±j5.33	.527
2	.85	5000'	-1.57	-.929±j9.12	.536
3	.9	35000'	- .637	-.392±j5.67	.537
4	1.5	35000'	- .826	-.481±j8.05	.577
AVERAGED VALUES			- .98	-.586±j7.04	.543

Fortunately the gain ratio is almost constant. The filter is then

$$\frac{\hat{N}_z}{q} = 0.543 \frac{s^2 + 1.172s + 49.9}{(s + 0.98)} \quad \frac{10}{s + 10} \tag{12}$$

The term 10/(s+10) was included to make the filter realizable. The pole at s = -0.98 approximately cancels the q-zero and is therefore weakly controllable from u, i.e. the corresponding closed loop pole will remain in the vicinity of -0.98. This however has little effect on the C* step responses and is exempted from the pole region requirements.

Note that the corresponding idea to use the accelerometer only and to omit the gyro leads to the inverse filter of eq.(12). Here the approximate cancellation occurs for a complex pair in the vicinity of s = -0.586 ± j7.04, i.e. close to the imaginary axis and no robustness with respect to changing flight condition can be achieved [1].

Fig. 8 shows the intersection of the admissible regions for the four flight conditions. It is seen, that flight conditions 2 and 3 are the critical ones, in the accelerometer feedback case flight conditions 1 and 4 were the most critical ones. However the two feedback gains have the same order of magnitude and the shape and extension of the admissible region still admits a choice of $\underline{k}$ as shown in Fig. 8 , namely

$$\underline{k} = \begin{bmatrix} k_{Nz} \\ k_q \end{bmatrix} = \begin{bmatrix} -0.09 \\ -0.8 \end{bmatrix} \qquad (13)$$

such that the pole region requirements are satisfied for $2\underline{k}/3$, $\underline{k}/2$ and $\underline{k}/3$.

The failure detection logic in Fig. 7 decides as follows

a) Three gyros unfailed: $g_1 = g_2 = g_3 = 1/3$

b) Gyro i failed: $g_i = 0$, $g_j = 1/2 \quad j \neq i$.

Before the decision b) has been made, we have a case between $\underline{k}$ and $2\underline{k}/3$. If a second gyro fails after decision b) has been made, we have a case between $\underline{k}$ and $\underline{k}/2$. Only in the unlikely case that a second gyro fails before the first failure has been detected, the gain may be reduced to $\underline{k}/3$. For the two typical cases $\underline{k}$, $2\underline{k}/3$, $\underline{k}/2$ and $\underline{k}/3$ the eigenvalues are given in the appendix. They meet all pole region requirements. Also the C^* step responses have been simulated for the open loop $k_{Nz} = k_q = 0$, Fig. 9, and for the closed loop with $\underline{k}$ and $\underline{k}/2$, Fig. 10. Fig. 11 shows the corresponding elevator deflections δ_e.

V CONCLUSION

It has been demonstrated, that an integrated view of redundancy and handling quality requirements for the control of the short period mode of an unstable fighter plane can lead to a simple fault tolerant control system. It is interesting that the measurement by gyros alone not only saves the cost of additional accelerometers but even has advantages for the control system. It is also interesting that a constant controller can control the aircraft at very different flight conditions. Here it must be noted of course that nice stabilization for several typical stationary flight conditions, i.e. different linearizations of a nonlinear system is only a necessary, not a sufficient condition for the stability of the nonlinear system. The nonlinear system is usually tested in simulations. Also for these simulations it is an advantage if the control system is simple, i.e. does not require gain scheduling and different sensor types and if failure detection is not vital for stabilization.

VI REFERENCES

[1] S.N. Franklin — Design of a Robust Flight Control System, MS Thesis, University of Illinois, Urbana-Champaign, EE Dept., August 1979.

[2] S.N. Franklin, J. Ackermann — Robust Flight Control - A Design Example. To be published.

[3] R.L. Berger, J.R. Hess and D.C. Anderson — Compatibility of Maneuver Load Control and Relaxed Static Stability Applied to Military Aircraft. AFFDL-TR-73-33, April 1973.

[4] Flying Qualities of Piloted Airplanes — MIL-F-8785B (ASG), 7 Aug. 1969.

[5] J. Ackermann — Parameter Space Design of Robust Control Systems, IEEE Transactions on Automatic Control, Oct. 1980.

VII ACKNOWLEDGEMENT

The author gratefully acknowledges the contributions of D. Kaesbauer und N. Franklin.

APPENDIX

1.) Aerodynamic data for eq.(1)

	FC 1	FC 2	FC 3	FC 4
Mach	0.5	0.85	0.9	1.5
Altitude	5000'	5000'	35000'	35000'
a_{11}	- 0.9896	- 1.702	- 0.667	- 0.5162
a_{12}	17.41	50.72	18.11	26.96
a_{13}	96.15	263.5	84.34	178.9
a_{21}	0.2648	0.2201	0.08201	- 0.6896
a_{22}	- 0.8512	- 1.418	- 0.6587	- 1.225
a_{23}	-11.39	- 31.99	-10.81	-30.38
b_1	-97.78	-272.2	-85.09	-175.6

2.) Military specifications for flying qualities, see eq.(3)

Natural frequency (rad/sec)	FC 1	FC 2	FC 3	FC 4
ω_a	2.02	3.50	2.19	3.29
ω_b	7.23	12.6	7.86	11.8

3.) Closed-loop eigenvalues

Complex eigenvalues $s^2 + 2\zeta\omega s + \omega^2$ are written (ζ,ω). The short period eigenvalues are listed first.

Gain	FC 1	FC 2	FC 3	FC 4
$\underline{k}$	(0.60, 4.30) (0.60, 17.2) -0.87	(0.68, 4.63) (0.38, 26.4) -1.63	(0.57, 4.38) (0.64, 16.2) -0.62	(0.65, 5.34) (0.45, 20.9) -0.86
$2\underline{k}/3$	(0.57, 3.86) (0.71, 15.3) -0.86	(0.66, 4.41) (0.47, 22.0) -1.67	(0.52, 3.95) (0.74, 14.5) -0.61	(0.60, 5.47) (0.55, 17.6) -0.87
$\underline{k}/2$	(0.55, 3.47) (0.77, 14.4) -0.85	(0.64, 4.17) (0.54, 19.6) -1.72	(0.50, 3.59) (0.80, 13.8) -0.60	(0.56, 5.54) (0.62, 1.57) -0.88
$\underline{k}/3$	(0.56, 2.86) (0.84, 13.5) -0.83	(0.61, 3.69) (0.64, 17.0) -1.84	(0.48, 3.05) (0.87, 13.1) -0.59	(0.48, 5.53) (0.74, 13.9) -0.90
0 open loop	1.23 - 3.07 -14 -10 - 0.98	1.78 - 4.90 -14 -10 - 0.98	0.56 - 1.87 -14 -10 - 0.98	(0.20, 4.4) -14 -10 - 0.98

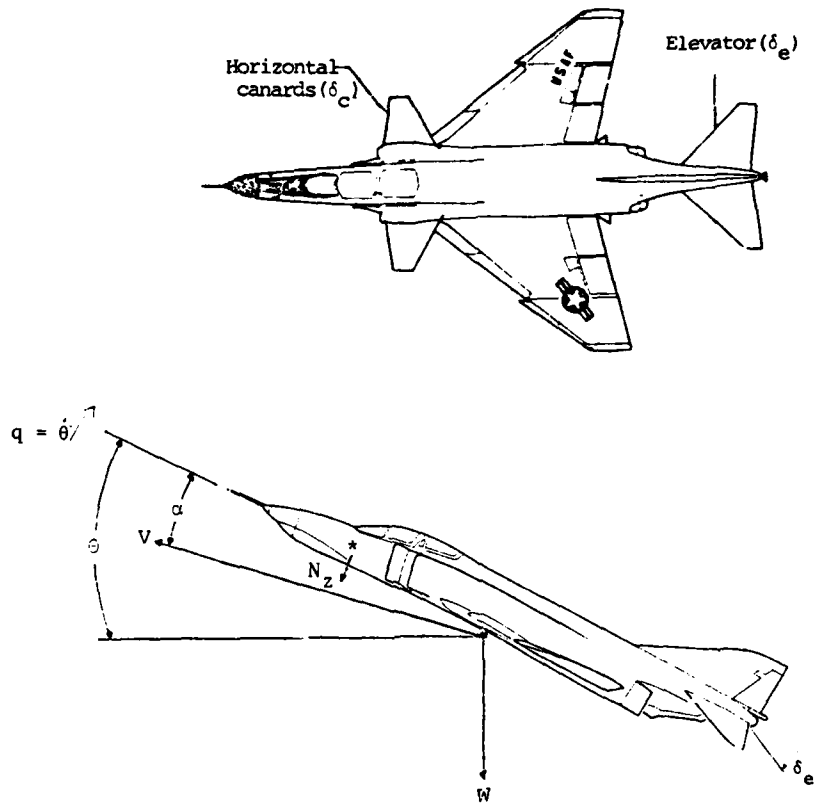


Fig. 1 F4-E with canards

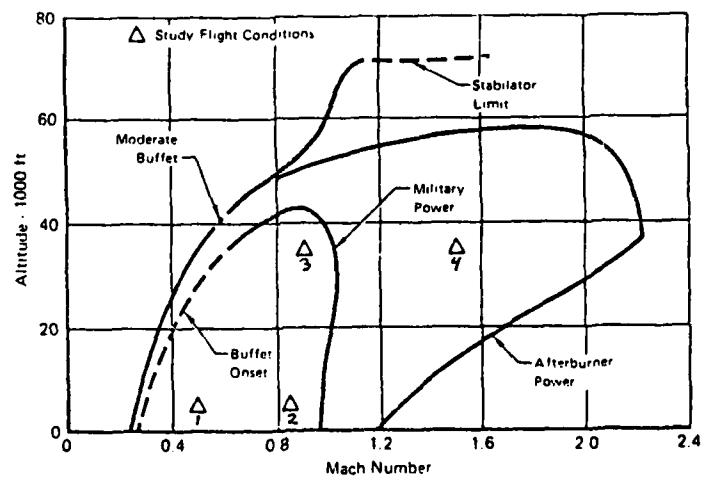


Fig. 2 Flight envelope and operating points [3]

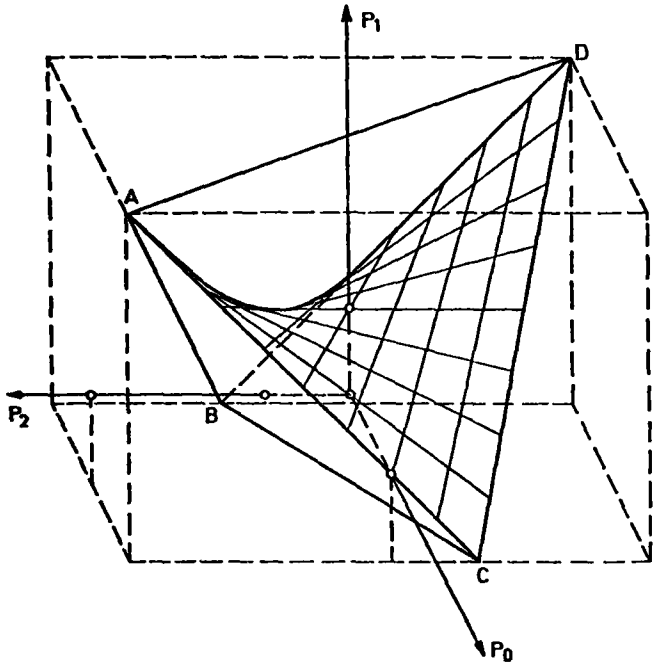


Fig. 4 Zeros of the polynomial $p_o + p_1 s + p_2 s^2 + s^3 = 0$ are in the unit circle if and only if the coefficients are inside the region bounded by the triangles ABC, BCD (real root boundaries) and by the hyperbolic paraboloid (complex root boundary)

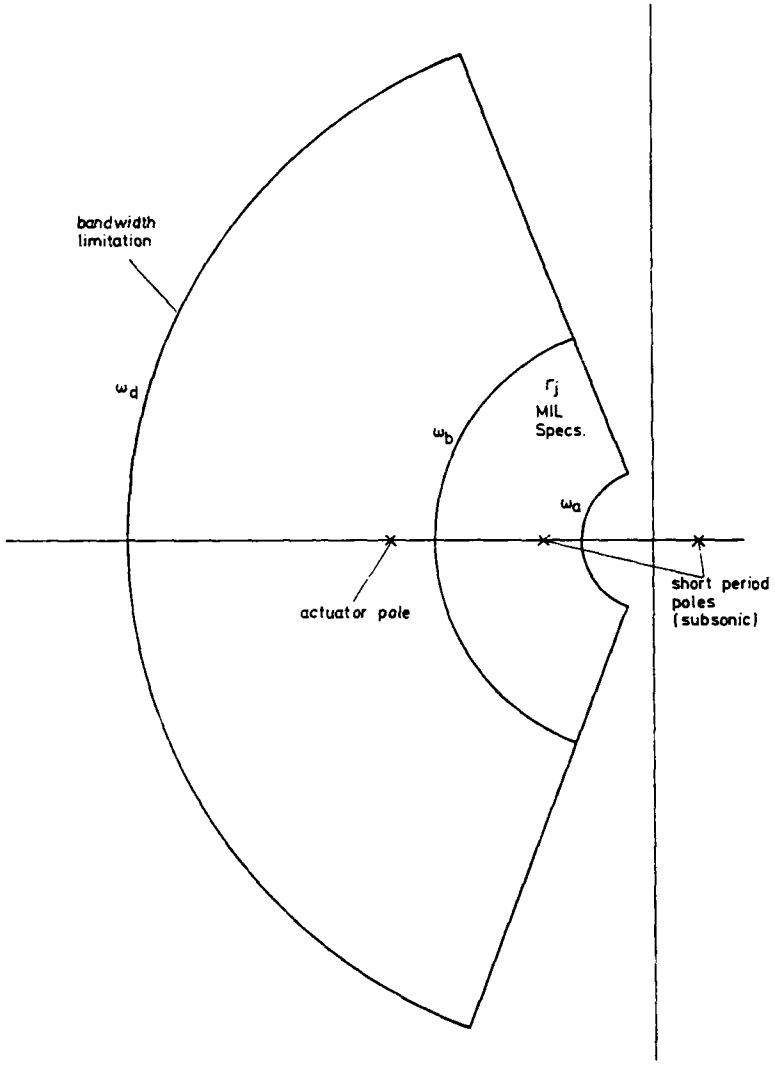


Fig. 3 Required closed loop pole regions

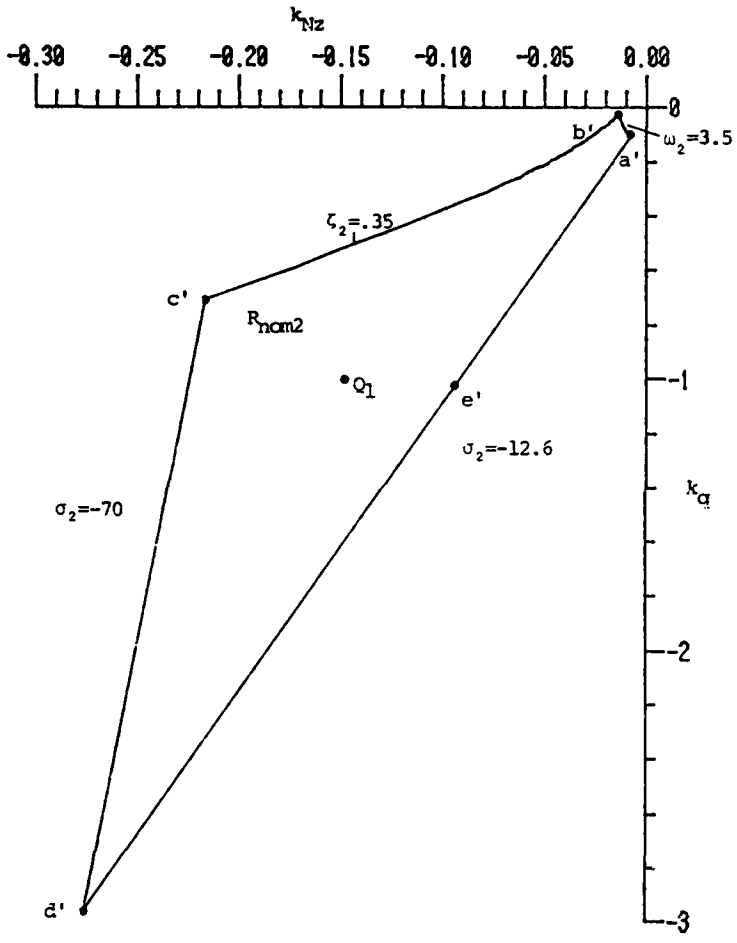


Fig. 5 Admissible region for flight condition 2

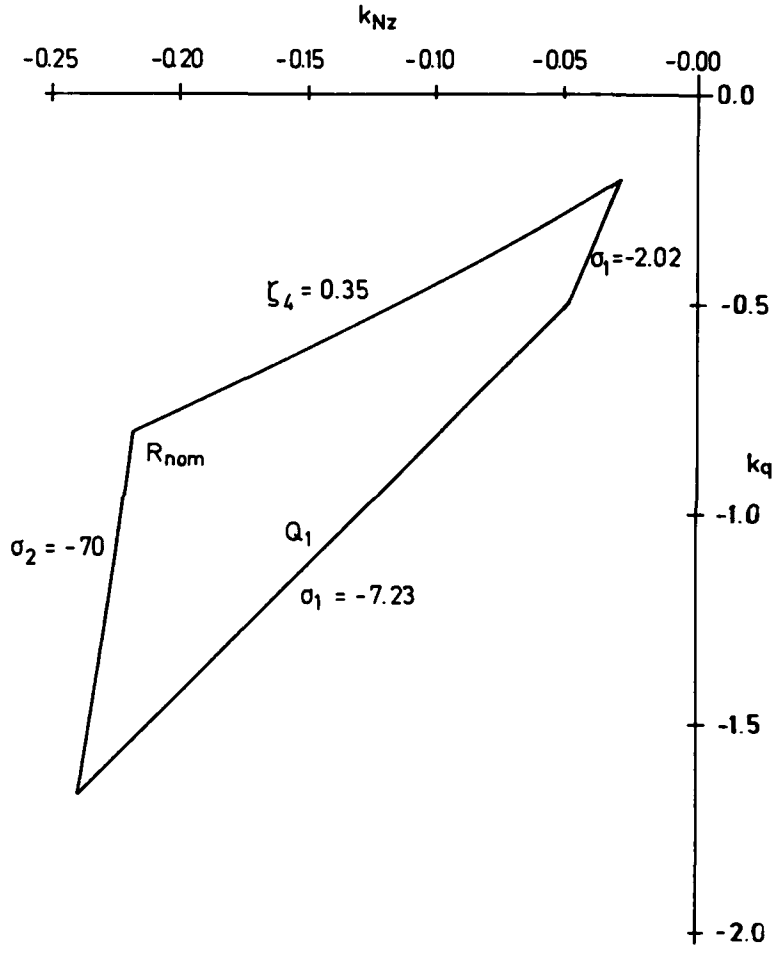


Fig. 6 Intersection of admissible regions for flight conditions 1 through 4

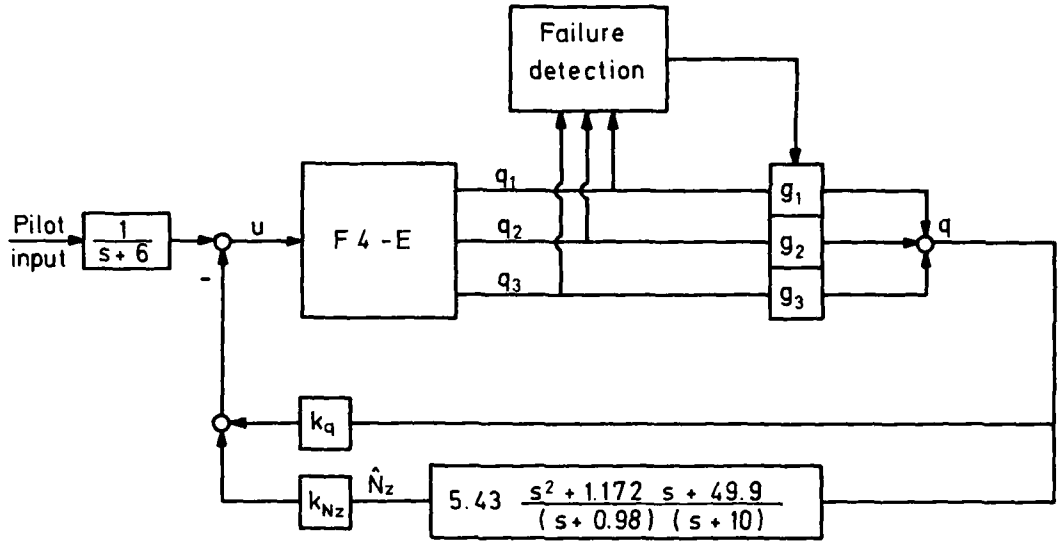


Fig. 7 Robust flight control system with 3 gyros

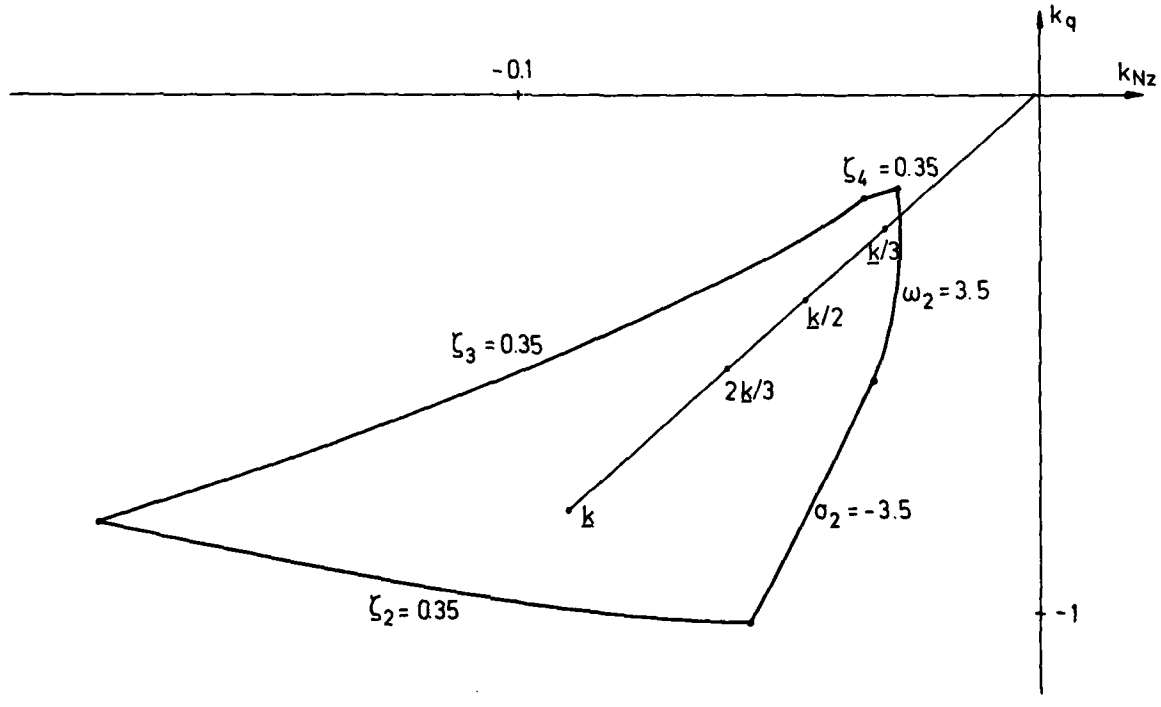


Fig. 8 Admissible region for control system configuration of Fig. 7

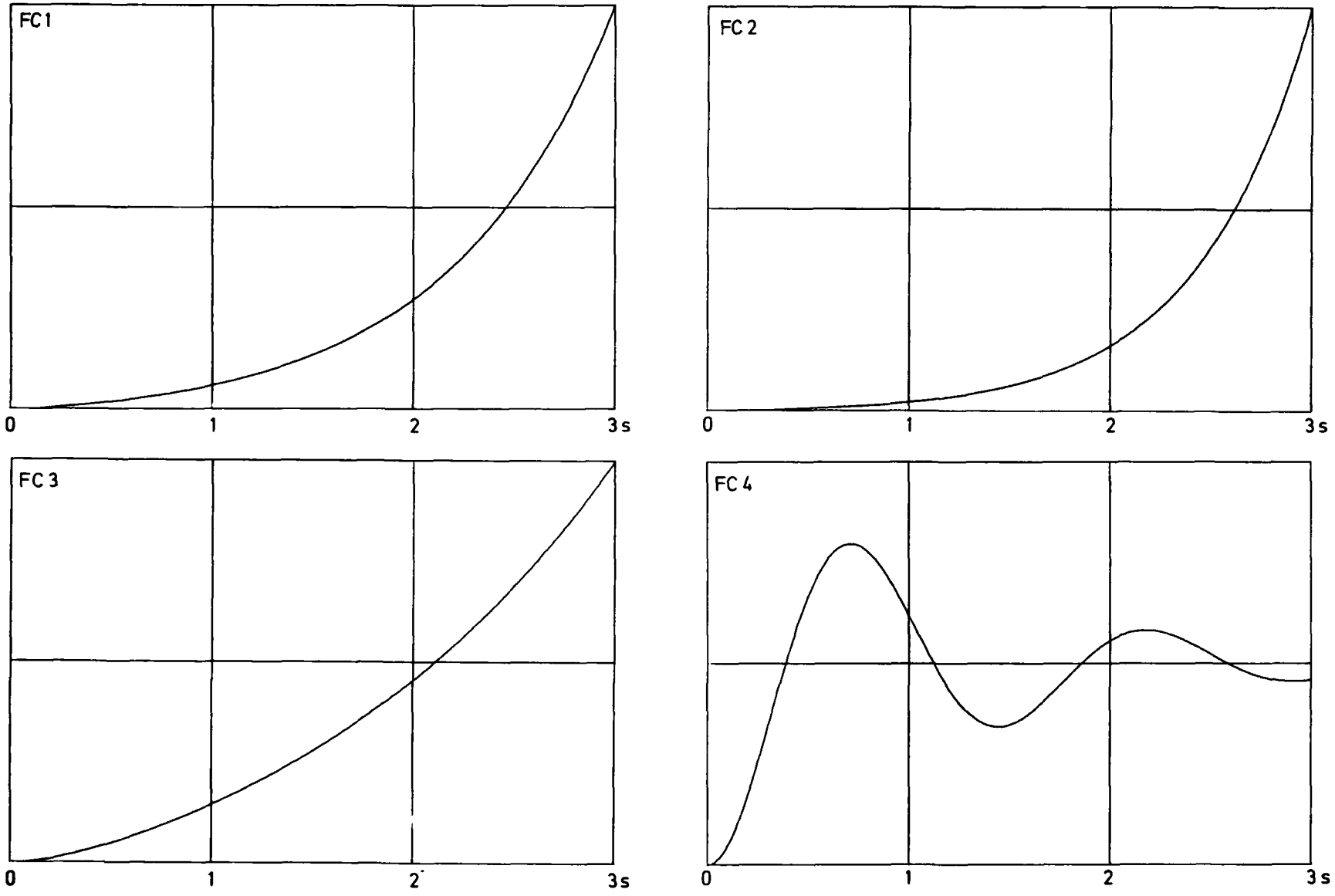


Fig. 9 C* step responses of the open loop

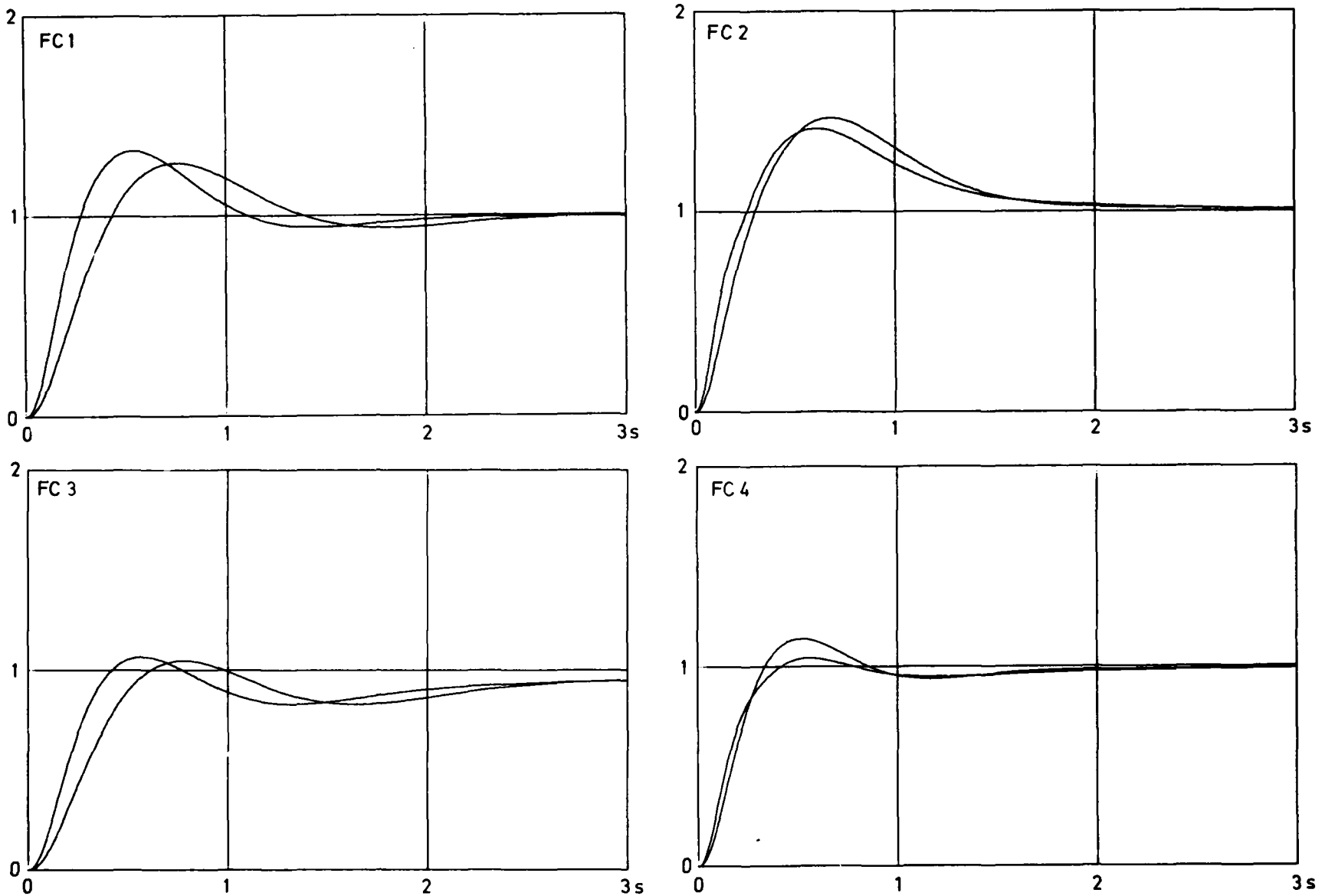


Fig. 10 C^{*} step responses of the closed loop for $\underline{k}$ and $\underline{k}/2$
($\underline{k}$ corresponds to the curve with the steeper initial rise)

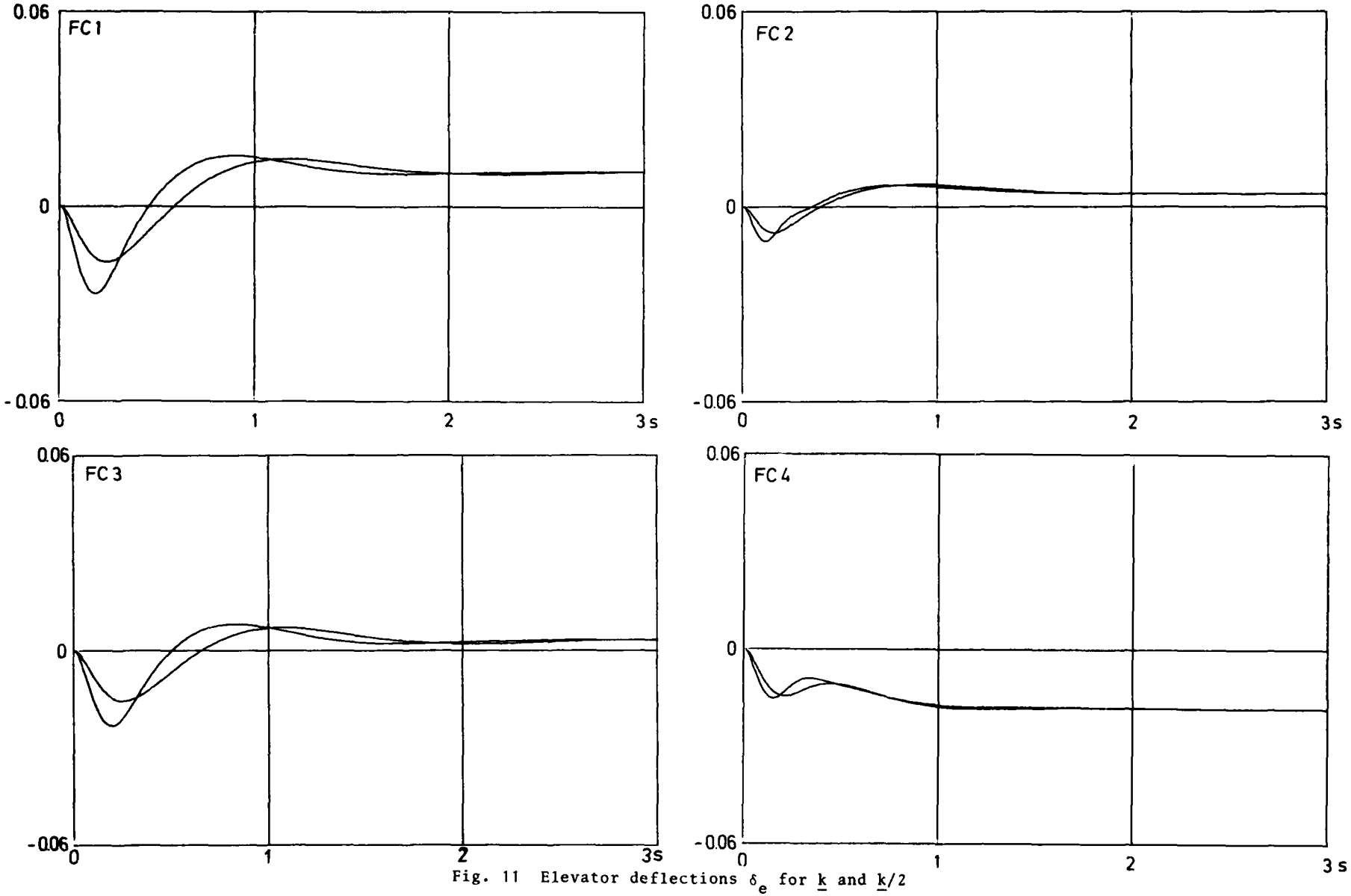


Fig. 11 Elevator deflections δ_e for $\underline{k}$ and $\underline{k}/2$

SELECTIVE BIBLIOGRAPHY

This Bibliography with Abstracts has been prepared to support AGARD Lecture Series No. 109 by the Scientific and Technical Information Branch of the US National Aeronautics and Space Administration, Washington, D.C., in consultation with the Lecture Series director, Thomas B.Cunningham, of the Honeywell Systems Research Center, Minneapolis, Minnesota.

UTTL: CH-53E DIGITAL AUTOMATIC FLIGHT CONTROL SYSTEM
A/Murphy, R.D.
PAA: A/(United Technologies Corp., Sikorsky Aircraft Div., Stratford, Conn.) In: Specialists Meeting on Helicopter Flight Controls, Arlington. Tex., October 11--13, 1978. Technical Papers. (A79-53626 24-08) Washington D.C., American Helicopter Society, 1979. 8p.

The CH-53E system represents the first digital automatic flight control system in a production helicopter. This paper examines the development and testing of the new digital system, with attention given to design in terms of both hardware and software. Consideration is given to hardware and software structure, the redundancy management scheme, and self-test features. The validation of the design in flight and in formal reliability tests is discussed along with the results achieved.
79/00/00 79A53638

UTTL: FAULT-TOLERANT SOFTWARE
A/Hecht, H.
PAA: A/(SoHar, Inc., Los Angeles, Calif.) IEEE Transactions on Reliability, vol. R-28, Aug. 1979, p. 227-232.

Limitations in the current capabilities for verifying programs by formal proof or by exhaustive testing have led to the investigation of fault-tolerance techniques for applications where the consequence of failure is particularly severe. Two current approaches, N-version programming and the recovery block, are described. A critical feature in the latter is the acceptance test, and a number of useful techniques for constructing these are presented. A system reliability model for the recovery block is introduced, and conclusions derived from this model that affect the design of fault-tolerant software are discussed.
79/08/00 79A51217

UTTL: APPLICATION OF ANALYTICAL REDUNDANCY MANAGEMENT TO SHUTTLE CRAFTS
A/Montgomery, R.C.: B/Tabak, D.
PAA: A/(NASA, Langley Research Center, Flight Dynamics and Control Div., Hampton, Va.); B/Texas, University, Austin, Tex.) CORP: National Aeronautics and Space Administration. Langley Research Center, Langley Station, Va.; Texas Univ., Austin. In: 1978 Conference on Decision and Control, 17th, San Diego, Calif., January 10--12, 1979, Proceedings. (A79-47930 21-63) New York, Institute of Electrical and Electronics Engineers, Inc., 1979, p.442-448.

The study involves the bank of filters approach to analytical redundancy management since this is amenable to microelectronic implementation. Attention is given to a study of the UD factorized filter to determine if it gives more accurate estimates than the standard Kalman filter when data processing word size is reduced. It is reported that, as the word size is reduced, the effect of modelling error dominates the filter performance of the two filters. However, the UD filter is shown to maintain a slight advantage in tracking performance. It is concluded that because of the UD filter's stability in the serial processing mode, it remains the leading candidate for microelectronic implementation.
79/00/00 79A47970

UTTL: TOWARDS FAULT-TOLERANT OPTIMAL CONTROL
A/Chizeck, H.J.; B/Willsky, A.S.
PAA: B/(MIT, Cambridge, Mass.) CORP: Massachusetts Inst. of Tech., Cambridge. In: 1978 Conference on Decision and Control, 17th, San Diego, Calif., January 10 12, 1979, Proceedings. (A79-47930 21-63) New York. Institute of Electrical and Electronics Engineers, Inc., 1979, p.19, 20.

The paper considers the design of fault-tolerant controllers that may endow systems with dynamic reliability. Results for jump linear quadratic Gaussian control problems are extended to include random jump costs, trajectory discontinuities, and a simple case of non-Markovian mode transitions.
79/00/00 79A47934

UTTL: DIGITAL FLIGHT CONTROL RELIABILITY EFFECTS OF REDUNDANCY LEVEL, ARCHITECTURE AND REDUNDANCY MANAGEMENT TECHNIQUE
A/Rice, J.W.; B/McCorkle, R.D.
PAA: B/(Boeing Aerospace Co., Seattle, Wash.) In: Guidance and Control Conference, Boulder, Colo., August 6 8, 1979. Collection of Technical Papers. (A79-45351 19-12) New York, American Institute of Aeronautics and Astronautics, Inc., 1979, p.645-657.

The reliabilities of several digital flight control systems (DFCSs) are compared, considering effects of redundancy level, control system architecture, redundancy management philosophy and, where applicable, fault detection and isolation coverage. Realistic reliability data are used for the system components. Each system is described and its success criteria established. It is shown that for longer missions, systems employing interunit selection at the LRU level can be more reliable than systems employing one higher level of redundancy and using midvalue signal voting as the only means of fault detection.
AIAA 79-1893 79/00/00 79A45418

UTTL: F-16 FLIGHT CONTROL SYSTEM REDUNDANCY CONCEPTS
A/Ammons, E.E.
PAA: A/(General Dynamics Corp., Fort Worth, Tex.) In: Guidance and Control Conference, Boulder, Colo., August 6-8, 1979, Collection of Technical Papers. (A79-45351 19-12) New York, American Institute of Aeronautics and Astronautics, Inc., 1979, p.484-490.

The analog fly-by-wire flight control system (FCS) of the F-16 is discussed. In order to provide undergraded performance following any two like-failures in the stability augmentation electronics, quad-redundant implementation of the pitch stability augmentation was selected, i.e., four branches that are physically and electrically isolated. The branches provide the system back up necessary for safe operation by rejecting the output from a branch that disagrees with two others, and then selecting the middle value of the remaining three. A flight path control, the FCS redundancy implementation, redundancy management, and the FCS gain scheduling are outlined with consideration given to the active selector and hydraulic actuator redundancy.
AIAA 79-1771 79/00/00 79A45400

UTTL: DUAL DIGITAL FLIGHT CONTROL REDUNDANCY MANAGEMENT SYSTEM DEVELOPMENT PROGRAM
A/Blair, J.D.; B/McCorkle, R.D.
PAA: B/(Boeing Aerospace Co., Seattle, Wash.) In: Guidance and Control Conference, Boulder, Colo., August 6-8, 1979, Collection of Technical Papers. (A79-45351 19-12) New York, American Institute of Aeronautics and Astronautics, Inc., 1979, p.40-46.

A dual digital flight control system incorporating interunit selection and redundancy management of device pairs was developed for laboratory demonstration. Four minicomputers connected via dual MIL-STD-1553A data buses perform flight control and input/output functions. The system was interfaced with a piloted flight simulator to provide closed-loop operation. Software was developed for redundancy management of system components and for flight control modes typical of modern transport aircraft. The system was demonstrated by flying simulated mission sequences during which multiple faults were inserted, showing the capability to maintain system integrity in the presence of multiple failures.
AIAA 79-1701 79/00/00 79A45356

UTTL: FAULT-TOLERANT, HIGH RELIABILITY ELECTRONIC ENGINE CONTROL SYSTEM
A/Mosca, V.G.; B/Rabinowitz, C.; C/Kreamer, H.
PAA: C/(United Technologies Corp., Electronic Systems Dept., Windsor Locks, Conn.) AIAA, SAE, and ASME, Joint Propulsion Conference, 15th, Las Vegas, Nev., June 18–20, 1979, AIAA 9p.

The paper introduces and applies the principles of redundancy-management techniques to the design of highly reliable fault-tolerant electronic engine controls. The evaluation starts with a baseline electronic final control system design. The baseline system is then altered to evaluate the benefit of successive applications of redundancy management techniques such as selective triple redundancy, majority voting, fault coverage, built-in test, and reliability mathematical modeling methods. These trends and methods used in fly-by-wire system reliability are evaluated for applicability to fuel controls. An optimum mix of MTBP, safety, and hardware complexity can be achieved through application of selected dual and triple redundancy at the functional modular level with a high-degree of software cross-strapping.
AIAA PAPER 79-1202 79/06/00 79A38983

UTTL: SIFT – DESIGN AND ANALYSIS OF A FAULT-TOLERANT COMPUTER FOR AIRCRAFT CONTROL
A/Wensley, J.H.; B/Lamport, L.; C/Goldberg, J.; D/Green, M.W.; E/Levitt, K.N.; F/Melliar-Smith, P.M.; G/Shostak, R.E.; H/Weinstock, C.B.
PAA: H/(SRI International, Menlo Park, Calif.) CORP: SRI International Corp., Menlo Park, Calif. IEEE Proceedings, vol. 66, Oct, 1978, p.1240-1255.

SIFT (Software Implemented Fault Tolerance) is an ultrareliable computer for critical aircraft control applications that achieves fault tolerance by the replication of tasks among processing units. The main processing units are off-the-shelf minicomputers, with standard microcomputers serving as the interface to the I/O system. Fault isolation is achieved by using a specially designed redundant bus system to interconnect the processing units. Error detection and analysis and system reconfiguration are performed by software. Iterative tasks are redundantly executed, and the results of each iteration are voted upon before being used. Thus, any single failure in a processing unit or bus can be tolerated with triplication of tasks, and subsequent failures can be tolerated after reconfiguration. Independent execution by separate processors means that the processors need only be loosely synchronized, and a novel fault-tolerant synchronization method is described.
78/10/00 79A25718

UTTL: FTMP – A HIGHLY RELIABLE FAULT-TOLERANT MULTIPROCESSOR FOR AIRCRAFT
A/Hopkins, A.L., Jr.; B/Smith, T.B., III; C/Lala, J.H.
PAA: C/(Charles Stark Draper Laboratory, Inc., Cambridge, Mass.) CORP: Draper (Charles Stark) Lab., Inc., Cambridge, Mass. IEEE, Proceedings, vol. 66, Oct, 1978, p.1221-1239.

The FTMP (Fault-Tolerant Multiprocessor) is a complex multiprocessor computer that employs a form of redundancy related to systems considered by Mathur (1971), in which each major module can substitute for any other module of the same type. Despite the conceptual simplicity of the redundancy form, the implementation has many intricacies owing partly to the low target failure rate, and partly to the difficulty of eliminating single-fault vulnerability. An extensive analysis of the computer through the use of such modeling techniques as Markov processes and combinatorial mathematics shows that for random hard faults the computer can meet its requirements. It is also shown that the maintenance scheduled at intervals of 200 hrs or more can be adequate most of the time.
78/10/00 79A25717

UTTL: FAULT-TOLERANCE - THE SURVIVAL ATTRIBUTE OF DIGITAL SYSTEMS
A/Avizienis, A.
PAA: A/(California Institute of Technology, Jet Propulsion Laboratory, Pasadena; California, University, Los Angeles, Calif.) CORP: Jet Propulsion Lab., California Inst. of Tech., Pasadena.; California Univ., Los Angeles. IEEE, Proceedings, vol. 66, Oct. 1978, p.1109-1125.

Fault-tolerance is the architectural attribute of a digital system that keeps the logic machine doing its specified tasks when its host, the physical system, suffers various kinds of failures of its components. A more general concept of fault-tolerance also includes human mistakes committed during software and hardware implementation and during man/machine interaction among the causes of faults that are to be tolerated by the logic machine. This paper discusses the concept of fault-tolerance, the reasons for its inclusion in digital system architecture, and the methods of its implementation. A chronological view of the evolution of fault-tolerant systems and an outline of some goals for its further development conclude the presentation.
78/10/00 79A25716

UTTL: FAULT TOLERANCE USING SELF-CHECKING BUILDING-BLOCK COMPUTERS
A/Rennels, D.A.
PAA: A/(California Institute of Technology, Jet Propulsion Laboratory, Pasadena, Calif.) CORP: Jet Propulsion Lab., California Inst. of Tech., Pasadena. In: Industry/Joint Services Automatic Test Conference and Workshop on Advanced Test Technology, Management, Acquisition Support, San Diego, Calif., April 3–7, 1978, Proceedings. (A79-16426 04-38) Washington, D.C., National Security Industrial Association, 1978, p.140-142.
Navy-sponsored research;

The paper attempts to define and characterize a set of VLSI (very large scale integration) building-block circuits which can be used to combine existing microprocessors and memories into a wide variety of fault-tolerant computing systems. Such VLSI circuits would transform fault-tolerant computing into an off-the-shelf technology and enable its routine use for new applications. The self-checking computer module (SCCM) is the basic component of which fault-tolerant computer systems are constructed. Several fault-tolerant configurations of SCCM are discussed, including the standby redundant uniprocessor, the voted/hybrid uniprocessor, and the distributed computer network.
78/00/00 79A16437

UTTL: DEVELOPMENT AND TESTING OF ADVANCED REDUNDANCY MANAGEMENT METHODS FOR THE F-8 DFBW AIRCRAFT
A/Deyst, J.; B/Deckert, J.; C/Desai, M.; D/Willsky, A.
PAA: C/(Charles Stark Draper Laboratory, Inc., Cambridge, Mass.); D/(MIT, Cambridge, Mass.) CORP: Draper (Charles Stark) Lab., Inc., Cambridge, Mass.; Massachusetts Inst. of Tech., Cambridge. In: Conference on Decision and Control, and Symposium on Adaptive Processes, 16th, and Special Symposium on Fuzzy Set Theory and Applications, New Orleans, La., December 7–9, 1977, Proceedings. Volume 1. (A79-14957 04-63) Piscataway, N.J., Institute of Electrical and Electronics Engineers, Inc., 1977, p.309-315.

A reliable aircraft sensor failure detection and identification (FDI) technique is presented. The technique exploits the kinematic and dynamic relationships that exist between variables measured by dissimilar sensors to identify failures in the sensors. The method is applied to management of dual redundant sensors on the NASA F-8 digital fly-by-wire (DFBW) research aircraft.
77/00/00 79A14976

UTTL: AN OPTIMAL APPROACH TO FAULT TOLERANT SOFTWARE SYSTEMS DESIGN
A/Gannon, T.F.; B/Shapiro, S.D.
PAA: A/(Sperry Univac Technical Research Center, Blue Bell, Pa.); B/Stevens Institute of Technology, Hoboken, N.J.) IEEE Transactions on Software Engineering, vol. SE-4 Sept. 1978, p.390-409

A systematic method of providing software system fault recovery with maximal fault coverage subject to resource constraints of overall recovery cost and additional fault rate is presented. This method is based on a model for software systems which provides a measure of the fault coverage properties of the system in the presence of computer hardware faults. Techniques for system parameter measurements are given. An optimization problem results which is a doubly-constrained 0.1 Knapsack problem. Quantitative results are presented demonstrating the effectiveness of the approach.
78/09/00 78A53076

UTTL: MULTI-LEVEL SELF-DIAGNOSIS AND FAULT TOLERANCE IN A MULTIMICROPROCESSOR SYSTEM
A/Negrini, R.; B/Sami, M.G.
PAA: B/(Milano, Politechnico, Milan, Italy) Alta Frequenza (English Edition), vol. 47, July 1978, p.325 E-333 E.

A multimicroprocessor system, dedicated to particular classes of application defined through process-processor interaction, is considered. The interconnection structure is analyzed, with the purpose of defining some standards, both for physical communication management and for message protocol, as far as possible independent of a particular application. Diagnosis and graceful degradation of the system are then discussed, exploiting the possibilities offered by the particular structure and by general control distribution. A multi-level organization is suggested for self-diagnosis, together with basic diagnostic standards; a correspondent multi-level graceful degradation technique is outlined.
78/07/00 78A51507

UTTL: OPTIMAL REDUNDANCY MANAGEMENT OF INERTIAL SENSORS BY AN EXTENSION OF THE MIDVALUE SELECTION TECHNIQUE TO THREE DIMENSIONS
A/Potter, J.E.; B/Suman, M.C.
PAA: A/(Charles Stark Draper Laboratory, Inc., Cambridge, Mass); B/(Northrop Corp., Precision Products Div., Norwood, Mass.) In: Guidance and Control Conference, Palo Alto, Calif., August 7 9, 1978. Technical Papers, (A78-50159 22-01) New York, American Institute of Aeronautics and Astronautics, Inc., 1978, p.535-540.

In order to achieve absolute tolerance to some specified number of failures in many redundant inertial sensor applications, sufficient performance margin must be provided to cover the worst-case degradation which can occur. The worst-case degradation depends on the particular algorithm used. An optimal algorithm can be defined implicitly by taking as the estimate that value which minimizes an appropriately chosen worst-case performance index. In the present paper a simple realization has been discovered for the important case of first-failure tolerance with five skewed sensors. The algorithm is the direct generalization of midvalue selection to three dimensions and: (1) uses simple logic, (2) requires no statistical assumptions, (3) provides absolute tolerance to a single failure, (4) is self-healing, (5) ignores noise spikes, and (6) achieves the optimal performance.
AIAA 78-1320 78/00/00 78A50218

UTTL: IMPROVED COMBAT SURVIVABILITY FOR FLY-BY-WIRE SENSOR SYSTEMS
A/Berman, H; B/Boudreau, J.
PAA: B/(Grumman Aerospace Corp., Bethpage, N.Y.) In: Guidance and Control Conference, Palo Alto, Calif., August 7 9, 1978, Technical Papers. (A78-50159 22-01) New York, American Institute of Aeronautics and Astronautics, Inc., 1978, p.251-263.

Recent developments in Digital fly-by-wire flight control technology can offer improved survivability for combat aircraft. Redundancy, which is used to achieve the desired levels of reliability and failure tolerance, can also lead to decreased vulnerability. Results are presented that show that sensor dispersion, in combination with analytic redundancy techniques, enhances flight control system survivability. However, dispersion of flight control sensors, e.g., gyros and accelerometers, can cause problems in sensor redundancy management and in control law dynamic performance. It is shown that these problems, which are due to like sensors measuring different elastic motions and rigid body kinematic effects, can be eliminated by using state estimators to remove these effects from the sensor data.
AIAA 78-1277 78/00/00 78A50186

UTTL: FAULT TOLERANT FLIGHT CONTROLS
A/Poupard, R.
PAA: A/(IBM Corp., Federal Systems Div., Owego, N.Y.) Aviation Engineering and Maintenance, vol. 2. Jan. 1978, p.19, 21, 22, 29.

A method which uses multiple digital computer systems in fly-by-wire flight control is described in terms of its tolerance to faults. Attention is given to the concept of 'coverage', which means that a fault can be detected, traced to a specific unit, and correct operation can be continued even after failure has occurred. The number of faults which can be tolerated within a given system is the subject of a hard- and software analysis. The NASA F-8 digital fly-by-wire research program is presented as an illustration of a two-fault tolerant redundant system, with particular applications in Space Shuttle avionics.
78/01/00 78A28900

UTTL: REDUNDANCY MANAGEMENT OF DIGITAL FLY-BY-WIRE SYSTEMS
A/Westermeier, T.F.
PAA: A/(McDonnell Aircraft Co., St. Louis, Mo.) In: Joint Automatic Control Conference, San Francisco, Calif., June 22 24, 1977, Proceedings. Volume 1. (A78-23851 08-63) New York, Institute of Electrical and Electronics Engineers, Inc., 1977, p.272-277.

Aircraft fly-by-wire systems employ redundancy to achieve a reliability that at least approximates that of mechanical control systems. Redundancy management, that is, management of the redundant resources (sensors, computers, actuators, power sources), is the means to achieving the required reliability. Coverage impacts redundancy management in many important ways. Reliability models must contain coverage as an independent variable and show the sensitivity of system reliability to coverage. All redundancy management failure detection techniques and reconfiguration strategies

must have high coverage as their main design goal. Although the principles discussed are specifically directed at a digital fly-by-wire system, they have wider application to any system in which redundancy management is used to achieve higher reliability.
77/00/00 78A23861

UTTL: FAULT TOLERANT DIGITAL FLIGHT CONTROL USING ANALYTICAL REDUNDANCY
A/Poyneer, R.D.; B/Cunningham, T.B.
PAA: A/(USAF, Flight Dynamics Laboratory, Wright-Patterson AFB, Ohio); B/(Honeywell, Inc., Minneapolis, Minn.) In: NAECON '77; Proceedings of the National Aerospace and Electronics Conference, Dayton, Ohio, May 17-19, 1977. (A78-15551 04-33) New York, Institute of Electrical and Electronics Engineers, Inc., 1977, p.111-120.

Of the many sensor reduction techniques, analytical redundancy contains a number of candidate filters which are developed here from a fundamental relationship of variables through hybrid simulation. Three basic concepts were developed for the A-7D aircraft to span the range of complexity and capabilities of anaytical redundancy. Two monitor systems were used for comparison: multiple trip level exceedance criteria currently in use with voting systems and a Sequential Likelihood Ratio Hypothesis Test (SLRT) of error signal mean value. Results indicate that the fault detection whether designed by classical means or Kalman filtering performs similarly, and that gust estimation improves performance. Also, failure isolation for a single set of unlike sensors is difficult to achieve with a reasonable computation load.
77/00/00 78A15565

UTTL: DUAL DEVICE REDUNDANCY MANAGEMENT
A/Smith, L.A.; B/Williams, P.G.
PAA: B/(Boeing Aerospace Co., Seattle, Wash.) In: Computers in Aerospace Conference, Los Angeles, Calif., October 31 – November 2, 1977, Collection of Technical Papers. (A78-12651 02-59) New York, American Institute of Aeronautics and Astronautics, Inc., 1977, p.338-343.

Dual control systems which can detect and isolate a faulty component are an attractive solution to automatic control systems that require a high degree of reliability with minimum cost, weight, and volume. The analysis described in this paper is based on a RPV flight control system for which a failure would mean an air vehicle loss. Each of the flight control system's dual components is monitored as a pair and can be individually isolated from the system by the onboard automatic redundancy management function or by the ground operator. A performance exceedance monitor provides a complete swap of all online/offline components should the air vehicle exceed its performance boundaries for an unknown cause, and a minimum operational software subset is defined as the operational recovery configuration.
AIAA 77-1441 77/00/00 78A12695

UTTL: DESIGN OF SELF-CHECKING AND FAULT-TOLERANT MICROPROGRAMMED CONTROLLERS
A/Williamson, I.
PAA: A/(Cambridge Consultants, Ltd., Cambridge, England) Radio and Electronic Engineer, vol. 47, Oct. 1977, p.449-454. European Space Research and Technology Centre

A realization of a self-checking and fail-safe programmable controller which uses a new control memory organization to give a simple and elegant implementation is described. Using this self-checking controller as an example, a new approach to fault-tolerant design referred to as Dual-Fail-Safe (DFS) is presented which utilizes two self-checking modules. The resulting fault-tolerant system is shown to be less costly and significantly more reliable than a conventional Triple-Modular-Redundancy (TMR) system.
77/10/00 78A12409

UTTL: THE FLIGHT CONTROL COMPUTERS OF THE F-18 ELECTRONICS SET-FLIGHT CONTROL
A/Katt, D.R.; B/Raymont, P.A.
PAA: A/(McDonnell Douglas Corp., St. Louis, Mo.); B/(General Electric Co., Binghamton, N.Y.) In: Digital Avionics Systems Conference, 2nd, Los Angeles, Calif., November 2-4, 1977, Supplement. (A78-12226 02-04) New York, American Institute of Aeronautics and Astronautics, Inc., 1977, 8p.

The US Navy's Hornet is designed to be survivable and fault tolerant. It is the first production fighter to utilize a digital processor within its flight control computers. An overview of the failure performance of the flight control system is presented. This is followed by an explanation of the functional partitioning of the Hornet flight control computers and a description of the mechanization of the hardware and software elements related to the redundancy management of these flight control computers. In addition, the divisioning of the software tasks is explained.
AIAA 77-1479 77/00/00 78A12275

UTTL: NAVY ADVANCED SENSOR PROGRAMS FOR FLY-BY-WIRE AIRCRAFT
A/Abrams, C.; B/Weinstein, W.; C/Solomon, R.
PAA: A/(US Naval Material Command, Naval Air Development Center, Warminster, Pa.); C/(Grumman Aerospace Corp., Bethpage, N.Y.) In: Digital Avionics Systems Conference, 2nd, Los Angeles, Calif., November 2-4, 1977, Collection of Technical Papers. (A78-12226 02-04) New York, American Institute of Aeronautics and Astronautics, Inc., 1977, p.162-173.

Digital fly-by-wire (DFBW) requirements for future Navy air missions present imposing goals for system safety, cost, performance, survivability and operational readiness. The Integrated Sensory Subsystem (ISS) and the Integrated Inertial Sensor Assembly (IISA) are examples of advanced concepts which employ redundant sensors as a means of meeting these requirements. The ISS program achieves these goals by maximizing modularization, interchangeability, mature sensor technology, and fault indication while avoiding undue hardware proliferation by minimizing the number of sensors. The Advanced Skewed Sensory Electronic Triad (ASSET) system, a unique array of rate sensors and digital redundancy management program, is incorporated within the ISS concept. Results of laboratory testing with a simulated A-6A aircraft, utilizing a two-axis fly-by-wire (FBW) flight control system, are presented.
AIAA 77-1504 77/00/00 78A12251

UTTL: FAILURE DETECTION WITHOUT EXCESSIVE HARDWARE REDUNDANCY
A/Maybeck, P.S.
PAA: A/(USAF, Institute of Technology, Wright-Patterson AFB, Ohio) In: NAECON '76; Proceedings of the National Aerospace and Electronics Conference, Dayton, Ohio. May 18–20, 1976. (A77-37352 17-33) New York, Institute of Electrical and Electronics Engineers, Inc., 1976, p.315-322.

It is shown that inherent functional redundancy among signals of different inertial-system sensors aboard an aircraft can be utilized without resorting exclusively to hardware duplication to achieve fault tolerance and high reliability in data systems. The concept feasibility and the extent of performance capabilities of a functional redundancy failure detection algorithm were demonstrated. In a performance improvement program, both missed and false alarms were minimized.
76/00/00 77A37394

UTTL: REDUNDANCY MANAGEMENT OF SHUTTLE FLIGHT CONTROL SENSORS
A/Gelderloos, H.C.; B/Wilson, D.V.
PAA: B/(Honeywell, Inc., Aerospace Div., St. Petersburg, Fla.) In: Conference on Decision and Control and Symposium on Adaptive Processes, 15th. Clearwater, Fla., December 1–3, 1976, Proceedings. (A77-28801 12-63) New York, Institute of Electrical and Electronics Engineers, Inc., 1976, p.462-475.

The paper reviews some of the considerations pertaining to the development of sensor redundancy management (RM) on the Shuttle, with particular reference to achievement of fail-operational/fail-safe performance in the Shuttle flight control system. The sensor RM algorithms being implemented in software are described. The discussion covers RM requirements, design approaches and analysis, with special emphasis on the analysis required to verify the design during the approach/land test phase.
76/00/00 77A28822

UTTL: FAULT-TOLERANT SYSTEMS
A/Avizienis, A.
PAA: A/(California, University, Los Angeles, Calif.) IEEE Transactions on Computers, vol. C-25, Dec. 1976, p.1304-1312.

The paper examines the basic concepts, motivation, and techniques of fault tolerance in computer systems, dealing mainly with the problem of hardware (operational) faults, but giving some attention to aspects of design fault tolerance as well. Operational faults are classified in terms of duration (transient versus permanent), extent (local versus distributed), and value (determinate versus indeterminate). Three types of redundancy for protecting computer systems against operational faults are characterized: hardware redundancy, which can be either static (or 'masking') or dynamic redundancy; software redundancy; and time (execution) redundancy. Some of the principles of theoretical and experimental prediction of reliability and effectiveness of the redundancy design are discussed. The basic characteristics of hardware-controlled recovery and software-controlled recovery are set forth.
76/12/00 77A16965

UTTL: INTEGRATED FLIGHT CONTROL SYSTEM DESIGN FOR CCV
A/Boudreau, J.A.
PAA: A/(Grumman Aerospace Corp., Bethpage, N.Y.) American Institute of Aeronautics and Astronautics, Aircraft Systems and Technology Meeting, Dallas, Tex., Sept. 27–29, 1976, 15p.

The advent of Controlled Configured Vehicle (CCV) design approaches has imposed severe reliability and fault tolerance requirements on aircraft flight control and supporting systems. This paper establishes the requirements for, and develops the configuration of an integrated Fly-By-Wire (FBW) flight control system suitable for an unstable CCV fighter/attack aircraft design. The hydraulic and electric power systems are an integral part of the design problem, since their functions are essential to safety of flight. A three-channel FBW system configuration was chosen as optimum. The system features in-line monitored active/on-line secondary actuators, skewed rate gyros and triplex digital computers, accelerometers and pilot input transducers.
AIAA PAPER 76-941 76/09/00 76A45415

UTTL: DESIGN AND TEST EXPERIENCE WITH A TRIPLY REDUNDANT DIGITAL FLY-BY-WIRE CONTROL SYSTEM
A/Szalai, K.J.; B/Felleman, P.G.; C/Gera, J.; D/Glover, R.D.

PAA: A/(NASA, Flight Research Center, Edwards, Calif.); B/(Charles Stark Draper Laboratory, Inc., Cambridge, Mass.); C/(NASA, Langley Research Center, Hampton, Va.); D/(NASA, Johnson Space Center, Houston, Tex.) CORP: Draper (Charles Stark) Lab., Inc., Cambridge, Mass.; National Aeronautics and Space Administration. Flight Research Center, Edwards, Calif.; National Aeronautics and Space Administration. Langley Research Center, Langley Station, Va. In: Guidance and Control Conference, San Diego, Calif., August 16–18, 1976, Proceedings. Conference sponsored by the American Institute of Aeronautics and Astronautics. New York, American Institute of Aeronautics and Astronautics, Inc., 1976. 30p.

A triplex digital fly-by-wire flight control system was developed and then installed in a NASA F-8C aircraft to provide fail-operative, full authority control. Hardware and software redundancy management techniques were designed to detect and identify failures in the system. Control functions typical of those projected for future actively controlled vehicles were implemented. This paper describes the principal design features of the system, the implementation of computer, sensor, and actuator redundancy management, and the ground test results. An automated test program to verify sensor redundancy management software is also described.
AIAA 76-1911 76/00/00 76A41491

UTTL: RECONFIGURABLE REDUNDANCY MANAGEMENT FOR AIRCRAFT FLIGHT CONTROL
A/Bosch, J.A.; B/Kuehl, W.J.
PAA: B/(General Electric Co., Binghamton, N.Y.) CORP: General Electric Co., Binghamton, N.Y. In: Guidance and Control Conference, San Diego, Calif., August 16–18, 1976, Proceedings. (A76-41426 20-12) New York. American Institute of Aeronautics and Astronautics, Inc., 1976, p.138-145. Research supported by the Boeing Co. and NASA.

A highly fault tolerant digital computer system has been configured based on extensive experience with flight proven, redundant digital flight control systems. The feasibility of minimizing hardware complexity is shown while maintaining high levels of fault tolerance. The emerging hardware design combines reconfiguration concepts with conventional hardware redundancy techniques and special operational software to provide dual fail operate performance with a basic triplex system. The design provides high reliability and flight safety, enhances maintainability, and reduces life cycle cost while offering improved performance for future aircraft.
AIAA 76-1932 76/00/00 76A41443

UTTL: FAULT TOLERANT SYSTEM RELIABILITY MODELING/ANALYSIS
A/Masreliez, C.J.; B/Bjurman, B.E.
PAA: B/(Boeing Commercial Airplane Co., Seattle, Wash.) In: Guidance and Control Conference, San Diego, Calif., August 16–18, 1976, Proceedings. (A76-41426 20-12) New York, American Institute of Aeronautics andAstronautics, Inc., 1976, p.130-137.

A formulation of a reliability analysis approach for a redundant channel system was a primary task of the NASA sponsored Airborne Advanced Reconfigurable Computer System (ARCS) study. Major design objectives for the ARCS were to achieve two-fail-operational capability for the majority of failures in a triplex system by degradation from triplex to duplex to simplex operation, and to provide transient fault survivability. This paper presents the Markov model technique used for the reliability assessment of the application model flight control system, including sensors, computer system, servos, and hydraulic power. The application model included a fly-by-wire control wheel steering mode and an all-weather autoland mode. Primary reliability parameters evaluated were functional survivability and functional readiness as a function of time since last verification of a fault-free system.
AIAA 76-1931 76/00/00 76A41442

UTTL: SOFTWARE CONTROL PROCEDURES FOR THE JA-37 DIGITAL AUTOMATIC FLIGHT CONTROL SYSTEM
A/Bailey, D.G.; B/Folkesson, K.
PAA: A/(Honeywell, Inc., Minneapolis, Minn.); B/(Saab-Scania AB, Goteborg, Sweden) In: Guidance and Control Conference, San Diego, Calif., August 16–18, 1976, Proceedings. (A76-41426 20-12) New York, American Institute of Aeronautics and Astronautics, Inc., 1976, p.122-129.

The software control procedures described were developed for the high authority fail-safe single-processor digital control system of the JA-37 Viggen interceptor. The control system authority is close to 10 g at low altitude high-speed flight conditions. The procedures involve software organization, documentation, and change procedures. A functional computation flow chart of the control system is discussed.
AIAA 76-1930 76/00/00 76A41441

UTTL: SPACE SHUTTLE FLIGHT CONTROL SYSTEM
A/Klinar, W.J.; B/Kubiak, E.T.; C/Peters, W.H.; D/Saldana, R.L.; E/Smith, E.E., Jr; F/Stegall, H.W.
PAA: E/(NASA, Johnson Space Center, Avionics Systems Engineering Div., Houston, Tex.); F/(McDonnell Douglas Aeronautics Co., Houston, Tex.) CORP: National Aeronautics and Space Administration. Lyndon B.Johnson Space Center, Houston, Tex. In: International Federation of Automatic Control, Triennial World Congress, 6th, Boston and Cambridge, Mass., August 24–30, 1975, Proceedings. Part 4. (A76-28778 13-63) Pittsburgh, Pa., Instrument Society of America, 1975, p.6.2 1-6.2 9;

The Space Shuttle is a control stabilized vehicle with control provided by an all digital, fly-by-wire flight control system. This paper gives a description of the several modes of flight control which correspond to the Shuttle mission phases. These modes are ascent flight control (including open loop first stage steering, the use of four computers operating in parallel and inertial guidance sensors). On-orbit flight control (with a discussion of reaction control, phase plane switching logic, jet selection logic, state estimator logic and OMS thrust vector control), entry flight control and TAEM (terminal area energy management to landing). Also discussed are redundancy management and backup flight control.
75/00/00 76A28872

UTTL: ACT SYSTEM DESIGN FOR RELIABILITY, MAINTAINABILITY AND REDUNDANCY MANAGEMENT
A/Emfinger, J.E.
PAA: A/(Sperry Rand Corp., Sperry Flight Systems Div., Phoenix, Ariz.) Society of Automotive Engineers, National Aerospace Engineering and Manufacturing Meeting, Culver City, Calif., Nov. 17-20, 1975, 11p.

The specifications of design requirements for Active Control Technology (ACT) flight control systems are addressed based on current state-of-the-art trends with emphasis placed on the impact of specific requirements on system mechanization. Of particular interest is the sensitivity of the ACT system design to redundancy management, reliability and maintainability requirements, and to the related subsystem interface concepts. Experience on both military and commercial aircraft programs is cited to provide insight to establishment of practical design requirements for ACT systems.
SAE PAPER 751052 75/11/00 76A22290

UTTL: FLY-BY-WIRE FLIGHT CONTROL SYSTEM DESIGN CONSIDERATIONS FOR FIGHTER AIRCRAFT
A/Livingston, E.C.
PAA: A/(General Dynamics Corp., Fort Worth, Tex.) Society of Automotive Engineers, National Aerospace Engineering and Manufacturing Meeting, Culver City, Calif., Nov. 17-20, 1975, 9p.

The application of fly-by-wire flight control systems in fighter aircraft influences the basic design of the aircraft and requires special attention to certain design characteristics of the control system. The use of control-configured vehicle concepts for performance benefits makes fly-by-wire a logical choice. Redundancy management, protection against power loss, lightning protection and controller selection are prime design factors to be considered. Flight testing of the YF-16 aircraft has demonstrated excellent performance and operating characteristics of its fly-by-wire control system.
SAE PAPER 751046 75/11/00 76A22284

UTTL: PROBABILISTIC RELIABILITY OF A CANONICAL FAULT-TOLERANT STANDY REDUNDANCY
A/Dennis, N.G.
PAA: A/(General Electric Co., Space Div., Bay St. Louis, Miss.) Institution of Electrical Engineers, Proceedings, vol. 123, Feb. 1976, p.135-139.

The paper presents a family of fundamental fault-tolerant systems using standby spares. Recursive algebraic expressions are given both for the smallest combinations of disabling failures and for the probabilistic reliabilities of each system in the family. It is shown that multifault tolerance by itself is not exactly synonomous with a best reliability or a best cost/benefit parameter. The 'crossover' surface of any redundancy is defined as the infinite set of all combinations of component reliabilities (as they decrease) for which the reliability of the redundant system has deteriorated to the same reliability as the non redundant system. For component reliabilities outside the 'crossover' surface, the presence of the redundancy further decreases system reliability. This fact is established numerically for $n = 6$.
76/02/00 76A20603

UTTL: REDUNDANCY MANAGEMENT TECHNIQUE FOR SPACE SHUTTLE COMPUTERS
A/Sklaroff, J.R.
PAA: A/(IBM Corp., Federal Systems Div., Owego, N.Y.) IBM Journal of Research and Development, vol. 20, Jan. 1976, p.20-28.
Research supported by the Rockwell International Corp.

This paper describes how a set of off-the-shelf general purpose digital computers is being managed in a redundant avionic configuration while performing flight-critical functions for the Space Shuttle. The description covers the architecture of the redundant computer set, associated redundancy design requirements, and the technique used to detect a failed computer and to identify this failure on-board to the crew. Significant redundancy management requirements consist of imposing a total failure coverage on all flight-critical functions, when more than two redundant computers are operating in flight, and a maximum failure coverage for limited storage and processing time, when only two are operating. The basic design technique consists of using dedicated redundancy management hardware and software to allow each computer to judge the 'health' of the others by comparing computer outputs and to 'vote' on the judgments. In formulating the design, hardware simplicity, operational flexibility, and minimum computer resource utilization were used as criteria.
76/01/00 76A19177

UTTL: A FAULT-TOLERANT ESTIMATOR FOR REDUNDANT SYSTEMS
A/Broen, R.B.
PAA: A/(McDonnell Aircraft Co., St. Louis, Mo.) IEEE Transactions on Aerospace and Electronic Systems, vol. AES-11, Nov. 1975, p.1281-1285.

This paper proposes a digital estimator for redundant systems which is superior to Kalman Filtering if a failure is present and reduces to Kalman Filtering if no failure is present. Fault tolerant estimation is achieved by defining the non-stationary weighting matrix associated with the nominal least squares estimator (Kalman filter) as a continuous nonlinear function of the measurements. Despite the nonlinear character of the failure detection and isolation feature, the estimator equations have closed form and hence require no iterative computations or approximations for implementation.
75/11/00 76A17935

UTTL: PERFORMANCE OF FAULT TOLERANT ESTIMATORS IN A NOISY ENVIRONMENT
A/Broen, R.B.
PAA: A/(McDonnell Aircraft Co., St. Louis, Mo.) American Institute of Aeronautics and Astronautics, Guidance and Control Conference, Boston, Mass., Aug. 20-22, 1975, 8p.

This paper describes the performance of several voters and voter-estimators for triply redundant systems in various noisy environments. Voter performance is compared in terms of output variance and bias levels. A group of continuous value voters are shown to be advantageous compared with either the mean value or the median voter in the presence of a single channel failure. It is suggested that voter selection be based on simulation including the actual system characteristics. A voter configuration is recommended as 'best' for the case where actual simulation is not feasible. Voter-estimators are shown to be advantageous compared with nominal Kalman filtering. The use of a voter (single stage estimate) versus a voter-estimator (multiple stage estimate) when applied to the same plant model is contrasted and the effect of imperfect knowledge of the plant dynamics on voter-estimator performance is shown using a first-order example.
AIAA PAPER 75-1062 75/08/00 75A41632

UTTL: OPTIMUM REDUNDANCY FOR ENGINE MANAGEMENT AND CONTROL
A/Evans, J.F.O.
PAA: A/(Smiths Industries, Ltd., Wembley, Middx., England) In: International Aerospace Instrumentation Symposium, 8th, Cranfield, Beds., England, Mar. 24-27, 1975, Proceedings. (A75-28765 12-06) London, Royal Aeronautical Society, 1975, 7p.

The redundancy which needs to be built into an engine management and control system in order for the pilot to assume a managerial executive role rather than the current mixture of executive, automatic monitor and servomechanism functions is considered. The tasks that such a system must carry out are first briefly reviewed, and a new duplex approach is then explained.
75/00/00 75A28789

UTTL: RECENT DEVELOPMENTS IN SOFTWARE FAULT TOLERANCE THROUGH PROGRAM REDUNDANCY
A/Kim, K.H.; B/Ramamoorthy, C.V.
CORP: University of Southern California, Los Angeles. Presented at 10th Hawaii Intern. Conf. on System Sciences, 6-7 Jan. 1977.
AD-AO37467 AFOSR-77-0148TR 76/00/00 77N82117

UTTL: FAULT TOLERANT AVIONIC SYSTEMS ARCHITECTURES STUDY
A/Murphy, L.R.; B/Avizienis, A.A.; C/Rennels, D.A.; D/McNeely, L.D.; E/Fulton, R.L.
CORP: Ultrasystems, Inc., Newport Beach, Calif.
AD-784879 REPT-74/6.20-24 AFAL-TR-74-102 74/06/00 75N77922

UTTL: EMULATION APPLIED TO RELIABILITY ANALYSIS OF RECONFIGURABLE, HIGHLY RELIABLE, FAULT-TOLERANT COMPUTING SYSTEMS
A/Migneault, G.E.
CORP: National Aeronautics and Space Administration. Langley Research Center, Langley Station, Va. In AGARD Avionics Reliability, Its Tech. and Related Disciplines. 11p. (SEE N80-19519 10-38).

Emulation techniques applied to the analysis of the reliability of highly reliable computer systems for future commercial aircraft are described. The lack of credible precision in reliability estimates obtained by analytical modeling techniques is first established. The difficulty is shown to be an unavoidable consequence of: (1) a high reliability requirement so demanding as to make system evaluation by use testing infeasible: (2) a complex system design technique, fault tolerance; (3) system reliability dominated by errors due to flaws in the system definition; and (4) elaborate analytical modeling techniques whose precision outputs are quite sensitive to errors of approximation in their input data. Next, the technique of emulation is described, indicating how its input is a simple description of the logical structure of a system and its output is the consequent behaviour. Use of emulation techniques is discussed for pseudo-testing systems to evaluate bounds on the parameter values needed for the analytical techniques. Finally an illustrative example is presented to demonstrate from actual use the promise of the proposed application of emulation.
79/10/00 80N19541

UTTL: REDUNDANCY MANAGEMENT CONSIDERATIONS FOR A CONTROL-CONFIGURED FIGHTER AIRCRAFT TRIPLEX DIGITAL FLY-BY-WIRE FLIGHT CONTROL SYSTEM
A/Watson, J.H.; B/Yousey, W.J.; C/Railey, J.M.
CORP: General dynamics/Fort Worth, Tex. In AGARD Advan. In Guidance and Control Systems Using Digital Tech. 23p (SEE N80-14017 05-01)

To preclude the shut down of the flight control computers for control configured fighter aircraft, redundant (parallel) processing is used in conjunction with redundancy management concepts. Using reliability requirements and goals as expressed in loss-of-control per flight hour, a digital flight control system architecture is evolved with specific emphasis placed on the input, processor and output subsystems. The incorporation of an analog cross strapping of lower reliability sensors is shown to be an effective means of increasing system reliability by retaining sensor redundancy after a computer failure. A technique called control law reconfiguration is developed which insures system survival after a second like sensor failure. Computer contribution to loss-of-control is reduced by the addition of system monitors which increase the computer self-test confidence level. The resultant architecture is shown to have an inherent reliability which is relatively insensitive to the configuration of the actuator interface, thus allowing this interface to be designed based on hardware/software complexity tradeoffs.
79/08/00 80N14026

UTTL: TRENDS IN RELIABILITY MODELING TECHNOLOGY FOR FAULT TOLERANT SYSTEMS
A/Bavuso, S.J.
CORP: National Aeronautics and Space Administration. Langley Research Center, Langley Station, Va.

Reliability modeling for fault tolerant avionic computing systems was developed. The modeling of large systems involving issues of state size and complexity, fault coverage, and practical computation was discussed. A novel technique which provides the tool for studying the reliability of systems with nonconstant failure rates is presented. The fault latency which may provide a method of obtaining vital latent fault data is measured.
NASA-TM-80089 79/04/00 79N26810

UTTL: AN INTEGRATED FAULT-TOLERANT AVIONICS SYSTEM CONCEPT FOR ADVANCED AIRCRAFT
CORP: Draper (Charles Stark) Lab., Inc., Cambridge, Mass.

A conceptual baseline design for a highly integrated fault- and damage-tolerant avionics architecture is presented. The architecture is generic in nature, and applicable to a broad range of aircraft types; including CTOL, VTOL, and V/STOL: and all classes from supersonic fighters to transports. The architecture embodies pools of modular resources, configured to flexibly serve required functions on a priority basis. By including system elements which can serve multiple functions and taking maximum advantage of systematic fault-tolerance methods and procedures, the design tends to minimize replication of elements and overall complexity. In concert with logistics and maintenance procedures designed around the pooled modular element approach, the architecture can provide required performance, reliability, damage tolerance, and availability at minimum life-cycle costs. Its inherent flexibility allows it to readily incorporate a wide variety of mission-specific elements and to easily adapt to growth and change as new elements and requirements arise.
AD-AO65136 R-1226 79/02/01 79N23082

UTTL: A STUDY OF REDUNDANCY MANAGEMENT STRATEGY FOR TETRAD STRAP-DOWN INERTIAL SYSTEMS
A/Hruby, R.J.; B/Bjorkman, W.S.; C/Schmidt, S.F.; D/Carestia, R.A.
PAA: B/(Analytical Mechanics Associates); C/(Analytical Mechanics Associates); D/(Southern Colorada Univ) CORP: National Aeronautics and Space Administration. Ames Research Center, Moffett Field, Calif.

Algorithms were developed that attempt to identify which sensor in a tetrad configuration has experienced a step failure. An algorithm is also described that provides a measure of the confidence with which the correct identification was made. Experimental results are presented from real-time tests conducted on a three-axis motion facility utilizing an ortho-skew tetrad strapdown inertial sensor package. The effects of prediction errors and of quantization on correct failure identification are discussed as well as an algorithm for detecting second failures through prediction.
NASA-TM-78576 A-6725 79/02/00 79N17842

UTTL: DIGITAL FLY-BY-WIRE FLIGHT CONTROL VALIDATION EXPERIENCE
A/Szalai, K.J.; B/Jarvis, C.R.; C/Krier, G.E.; D/Megna, V.A.; E/Brock, L.D.; F/O'Donnell, R.N.
PAA: D/(Charles Stark Draper Lab., Inc.,); E/(Charles Stark Draper Lab., Inc.,); F/(Charles Stark Draper Lab., Inc.,)
CORP: National Aeronautics and Space Administration. Hugh L.Dryden Flight Research Center, Edwards, Calif.

The experience gained in digital fly-by-wire technology through a flight test program being conducted by the NASA Dryden Flight Research Center in an F-8C aircraft is described. The system requirements are outlined, along with the requirements for flight qualification. The system is described, including the hardware components, the aircraft installation, and the system operation. The flight qualification experience is emphasized. The qualification process included the theoretical validation of the basic design, laboratory testing of the hardware and software elements, systems level testing, and flight testing. The most productive testing was performed on an iron bird aircraft, which used the actual electronic and hydraulic hardware and a simulation of the F-8 characteristics to provide the flight environment. The iron bird was used for sensor and system redundancy management testing, failure modes and effects testing, and stress testing

in many cases with the pilot in the loop. The flight test program confirmed the quality of the validation process by achieving 50 flights without a known undetected failure and with no false alarms.
NASA-TM-72860 R-1164 H-1080 78/12/00 79N14109

UTTL: STRAPDOWN SYSTEM REDUNDANCY MANAGEMENT FLIGHT DEMONSTRATION
CORP: Litton Systems. Inc., Woodland Hills, Calif.

The suitability of strapdown inertial systems in providing highly reliable short-term navigation for vertical take-off and landing (VTOL) aircraft operating in an intra-urban setting under all-weather conditions was assessed. A preliminary design configuration of a skewed sensor inertial reference system employing a redundancy management concept to achieve fail-operational, fail-operational performance, was developed.
NASA-CR-145356 78/09/00 78N32079

UTTL: INHERENT ERRORS IN ASYNCHRONOUS DIGITAL FLIGHT CONTROLS
A/Slivinsky, C.
CORP: Missouri Univ. – Columbia. CSS: (Dept. of Electrical Engineering.)

This report describes research on redundancy management in digital flight control systems. The emphasis is on the properties, techniques, and requirements associated with the operations of monitoring and voting and their effects on the closed loop system operation when asynchronous sampling is used. Part 1 is concerned primarily with the monitoring operation for quadredundant input signals. Part 2 presents three extensions to a previously reported model for closed loop flight control systems that have dual-redundant, asynchronous digital controllers.
AD-AO55649 AFOSR-78-1054TR 78/03/31 78N31124

UTTL: FLIGHT TEST RESULTS OF THE STRAPDOWN HEXAD INERTIAL REFERENCE UNIT (SIRU)
Volume 3: Appendices A–G
A/Hruby, R.J.; B/Bjorkman, W.S.
PAA: B/(Analytical Mechanics Associates, Inc., Mountain View, Calif.) CORP: National Aeronautics and Space Administration. Ames Research Center, Moffett Field, Calif.

Results of flight tests of the Strapdown Inertial Reference Unit (SIRU) navigation system are presented. The fault tolerant SIRU navigation system features a redundant inertial sensor unit and dual computers. System software provides for detection and isolation of inertial sensor failures and continued operation in the event of failures. Flight test results include assessments of the system's navigational performance and fault tolerance. Selected facets of the flight tests are also described in detail and include some of the following: (1) flight test plans and ground track plots; (2) navigation residual plots; (3) effects of approximations in navigation algorithms; (4) vibration spectrum of the CV-340 aircraft; and (5) modification of the statistical FDICR algorithm parameters for the flight environment.
NASA-TM-73224 A-6974 77/07/00 78N20098

UTTL: FAULT-TOLERANT SOFTWARE FOR AIRCRAFT CONTROL SYSTEMS
CORP: Aerospace Corp., El Segundo, Calif. CSS: (Advanced Programs Div.)

Concepts for software to implement real time aircraft control systems on a centralized digital computer were discussed. A fault tolerant software structure employing functionally redundant routines with concurrent error detection was proposed for critical control functions involving safety of flight and landing. A degraded recovery block concept was devised to allow collocation of critical and noncritical software modules within the same control structure. The additional computer resources required to implement the proposed software structure for a representative set of aircraft control functions were discussed. It was estimated that approximately 30 percent more memory space is required to implement the total set of control functions. A reliability model for the fault tolerant software was described and parametric estimates of failure rate were made.
NASA-CR-145298 ATR-78(7640)-1 78/02/01 78N18797

UTTL: FAULT TOLERANT DIGITAL FLIGHT CONTROL WITH ANALYTICAL REDUNDANCY
A/Cunningham, T.; B/Carlson, D.; C/Hendrick, R.C.; D/Shaner, D.; E/Hartmann, G.
CORP: Honeywell, Inc., Minneapolis, Minn. CSS: (Systems and Research Center.)

Analytical redundancy offers high potential for solving the sensor redundancy problem. The current state-of-the-art in analytical redundancy contains a number of candidate filters which are developed here from a fundamental relationship of variables through hybrid simulation. Three analytical redundancy concepts of varying complexity were developed for the A-7 D aircraft. Two monitor techniques were used: a multiple trip level exceedance criteria currently in use with voting systems, and a sequential likelihood ratio hypothesis test SLRT on the mean value of the error signal. Results indicate that fault detection, whether designed by Classical methods or Kalman filtering, perform similarly and that gust estimation improves fault detection performance. Also, failure isolation for a single set of unlike sensors is difficult to achieve with a reasonable computation load. The output recommendation is to proceed to flight test with a set of nonlinear diagnostic filters to be used with comparison monitors for dual sensor failure isolation, creating a fail-operative dual sensor system. Fail-safe operation will also be achieved by continuing to monitor the filters and to detect, but not isolate, a second failure.
AD-AO45671 AFFDL-TR-77-25 77/05/00 78N13074

UTTL: FUTURE TRENDS IN HIGHLY RELIABLE SYSTEMS
A/Arnold, J.I.
CORP: Boeing Co., Wichita, Kans. In AGARD Integrity in Electron. Flight Control Systems, 14p (SEE N77-25055 16-01)

The need for highly reliable flight control systems in both control configured vehicles and conventionally designed aircraft is discussed. Technology trends in the area of control system computation, electronics, sensors and actuation are addressed. Increased use of digital computation and signal multiplexing in future control systems is considered inevitable. Recent technology developments in high density electronic packaging, large scale integration, and fiber optics will be applied to achieve highly reliable electronic systems. Component designs will be required to withstand potentially severe environments in the presence of lightning or nuclear phenomena. Redundancy management will continue to be a prime driving force in reliable system designs. The use of in-line monitoring to limit the proliferation of redundant channels should find application in future systems. Maintenance and preflight self-test systems will play an increasingly vital role in assuring the integrity of redundant flight-critical systems.
77/04/00 77N25059

UTTL: CHRONOLOGICAL OVERVIEW OF PAST AVIONIC FLIGHT CONTROL SYSTEM RELIABILITY IN MILITARY AND COMMERCIAL OPERATIONS
A/Osder, S.S.
CORP: Sperry Flight Systems, Phoenix. Ariz. In AGARD Integrity in Electron. Flight Control Systems, 17p (SEE N77-25055 16-01)

Flight control system mechanization advances are traced from the perspective of reliability. Despite dramatic advances in device technology and miniaturization, the demand for more functions tended to exceed the progress in electronics. By the latter 1960's, complexity growth related to system monitoring and redundancy management reached limitations of analog technology and set the stage for introduction of digital flight control systems.
77/04/00 77N25057

UTTL: FAILURE DETECTION AND CONTROL-SYSTEM RECONFIGURATION: PAST, PRESENT, AND FUTURE
A/Montgomery, R.C.
CORP: National Aeronautics and Space Administration. Langley Research Center, Langley Station, Va. In its Systems Reliability Issues for Future Aircraft, p.69-78 (SEE N77-22808 13-59)

The history of failure detection and redundancy management in aircraft applications is reviewed. To date, techniques related to that subject have been based mainly on hardware duplication of like components with failure monitoring and switchover or averaging for redundancy management. Specific examples of these techniques are discussed as they have been applied to the NASA F-8 Digital Fly-by-Wire aircraft and are to be applied to the space shuttle vehicle in the near future.
75/00/00 77N22813

UTTL: ELECTROMECHANICAL ACTUATION FEASIBILITY STUDY
A/Wood, N.E.; B/Echols, E.F.; C/Ashmore, J.H.
CORP: AiResearch Mfg. Co., Torrance, Calif.

An alternate to hydraulic actuation of primary flight control surfaces is desirable. Electromechanical actuation of primary flight controls is feasible and practical using the most recent advances in the state-of-the-art in magnetic materials for high-performance servomotors, high-current-capacity transistor switches for motor power control, and digital microprocessors for servo and redundancy management logic. Electromechanical actuation is consistent with current, proven, fly-by-wire technology and with the developing power-by-wire techniques. The control surface performance requirements and methods of analysis are presented for direct drive electromechanical power servos. Trade studies were conducted to evaluate the aircraft power source and distribution options; actuator motor and controller options; and geared rotary hinge actuator configuration options. A weight comparison is given between the hydraulic actuation system for the F-100 and the electromechanical system studied. Reliability assessments and redundancy management considerations also are included. A preliminary design of the electromechanical actuation system and test plan is presented. These data were developed using a selected baseline problem statement which was derived from existing hydraulic actuator specifications.
AD-AO31146 AFFDL-TR-76-42 76/05/00 77N19063

UTTL: AIRBORNE ADVANCED RECONFIGURABLE COMPUTER SYSTEM (ARCS)
A/Bjurman, B.E.; B/Jenkins, G.M.; C/Masreliez, C.J.; D/McClellan, K.L.; E/Templeman, J.E.
CORP: Boeing Commercial Airplane Co., Seattle, Wash.

A digital computer subsystem fault-tolerant concept was defined, and the potential benefits and costs of such a subsystem were assessed when used as the central element of a new transport's flight control system. The derived advanced reconfigurable computer system (ARCS) is a triple-redundant computer subsystem that automatically reconfigures, under multiple fault conditions, from triplex to duplex to simplex operation, with redundancy recovery if the fault condition is transient. The study included criteria development covering factors at the aircraft's operation level that would influence the design of a fault-tolerant system for commercial airline use. A new reliability analysis

tool was developed for evaluating redundant, fault-tolerant system availability and survivability; and a stringent digital system software design methodology was used to achieve design/implementation visibility.
NASA-CR-145024 D6-42476 76/08/00 76N30865

UTTL: ANALYSIS OF THE SURVIVABILITY OF THE SHUTTLE
(ALT fault-tolerant avionics system, appendices)
CORP: Ultrasystems, Inc., Irvine, Calif.

The analytic model program which calculated the baseline parameter data is described along with the input deck setup which includes four groups of input parameters.
NASA-CR-147661 REPT-76/6.43-9 76/04/00 76N22289

UTTL: PRELIMINARY INPUT TO THE SPACE SHUTTLE REACTION CONTROL SUBSYSTEM FAILURE DETECTION AND IDENTIFICATION SOFTWARE REQUIREMENTS (UNCONTROLLED)
A/Bergmann, E.
CORP: Draper (Charles Stark) Lab., Inc., Cambridge, Mass.

The current baseline method and software implementation of the space shuttle reaction control subsystem failure detection and identification (RCS FDI) system is presented. This algorithm is recommended for conclusion in the redundancy management (RM) module of the space shuttle guidance, navigation and control system. Supporting software is presented and recommended for inclusion in the system management (SM) and display and control (D&C) systems. RCS FDI uses data from sensors in the jets, in the manifold isolation valves, and in the RCS fuel and oxidizer storage tanks. A list of jet failures and fuel imbalance warnings is generated for use by the jet selection algorithm of the on-orbit and entry flight control systems, and to inform the crew and ground controllers of RCS failure status. Manifold isolation valve close commands are generated in the event of failed on or leaking jets to prevent loss of large quantities of RCS fuel.
NASA-CR-147461 C-4576 76/01/00 76N18217

UTTL: IMPACT OF COVERAGE ON THE RELIABILITY OF A FAULT TOLERANT COMPUTER
A/Bavuso, S.J.
CORP: National Aeronautics and Space Administration. Langley Research Center, Langley Station, Va.

A mathematical reliability model is established for a reconfigurable fault tolerant avionic computer system utilizing state-of-the-art computers. System reliability is studied in light of the coverage probabilities associated with the first and second independent hardware failures. Coverage models are presented as a function of detection, isolation, and recovery probabilities. Upper and lower bonds are established for the coverage probabilities and the method for computing values for the coverage probabilities is investigated. Further, an architectural variation is proposed which is shown to enhance coverage.
NASA-TN-D-7938 L-10050 75/09/00 75N30862

UTTL: DIGITAL FLIGHT CONTROL SYSTEM REDUNDANCY STUDY
A/McGough, J.; B/Moses, K.; C/Platt, W.; D/Reynolds, G.; E/Strole, J.
CORP: Bendix Corp., Teterboro, N.J. CSS: (Flight Systems Div.)

Redundancy requirements and trade-off criteria are established for flight critical digital flight control systems with particular emphasis on the fly-by-wire application. The use of general purpose digital computers is considered, with self-test and cross-channel comparison monitoring techniques to obtain the necessary flight safety reliability. A reliability model is presented which includes the effects of detected and undetected failures and provides a basis for establishing in-flight and preflight test coverage requirements consistent with a given reliability goal.
AD-A006411 AFFDL-TR-74-83 74/07/00 75N29129

UTTL: THEORY OF RELIABLE SYSTEMS
A/Meyer, J.F.
CORP: Michigan Univ., Ann Arbor.

An attempt was made to refine the current notion of system reliability by identifying and investigating attributes of a system which are important to reliability considerations. Techniques which facilitate analysis of system reliability are included. Special attention was given to fault tolerance, diagnosability, and reconfigurability characteristics of systems.
NASA-CR-142608 75/04/00 75N21650

UTTL: FAULT TOLERANT PROGRAMMABLE DIGITAL ATTITUDE CONTROL ELECTRONICS STUDY
A/Sorensen, A.A.
CORP: TRW Systems Group, Redondo Beach, Calif.; Jet Propulsion Lab., California Inst. of Tech., Pasadena.

The attitude control electronics mechanization study to develop a fault tolerant autonomous concept for a three axis system is reported. Programmable digital electronics are compared to general purpose digital computers. The requirements, constraints, and tradeoffs are discussed. It is concluded that: (1) general fault tolerance can be achieved relatively economically, (2) recovery times of less than one second can be obtained, (3) the number of faulty behavior patterns must be limited, and (4) adjoined processes are the best indicators of faulty operation.
NASA-CR-142289 TRW-26084-6001-RU-00 74/07/25 75N17400

UTTL: FAULT-TOLERANCE FEATURES OF AN AEROSPACE MULTIPROCESSOR
A/Miller, J.S.
CORP: Intermetrics, Inc., Cambridge, Mass. In AGARD Real Time Computer Based Systems, p.9 (SEE N75-16257 07-62)

Processor errors are detected by comparing results from duplexed units executing concurrently. Local processor storage is also duplexed, and segregated from processing units. Parity checking is used to identify the invalid copy when a comparison failure is signalled. Instruction execution is split into phases such that no phase overwrites its input. A hard-core redundant unit is used to command instruction-phase retry following a fault. If retry fails, another processor is interrupted to unload the faulty processor's local storage and prepare the disrupted process for immediate resumption at the point of failure. Recovery from faults in main memory capitalizes on the descriptor-based memory multiplexing scheme used for normal operation. A novel use of interleaving allows hardware-supported duplicated safe storage of data segments in main memory, since these change too frequently to be duplicated on secondary storage.
74/12/00 75N16278

REPORT DOCUMENTATION PAGE

1. Recipient's Reference	2. Originator's Reference	3. Further Reference	4. Security Classification of Document
	AGARD-LS-109	ISBN 92-835-0274-4	UNCLASSIFIED

5. Originator Advisory Group for Aerospace Research and Development
North Atlantic Treaty Organization
7 rue Ancelle, 92200 Neuilly sur Seine, France

6. Title FAULT TOLERANCE DESIGN AND REDUNDANCY MANAGEMENT TECHNIQUES

7. Presented on 13–14 October 1980 in Athens, Greece; 16–17 October 1980 in Rome, Italy and 20–21 October 1980 in London, UK.

8. Author(s)/Editor(s)	9. Date
Various	September 1980
10. Author's/Editor's Address	**11. Pages**
Various	182

12. Distribution Statement This document is distributed in accordance with AGARD policies and regulations, which are outlined on the Outside Back Covers of all AGARD publications.

13. Keywords/Descriptors

Redundancy
Management methods
Computer programs
Failsafe systems
Reliability (electronics)
Failure
Guidance and control

14. Abstract

These lectures are intended to provide the basic theory on concepts involved in the application of advanced software, state estimation, and implementation techniques involved in redundancy management, and to give a review covering the necessary background and state-of-the-art involved in the application of advancing technologies.

The material in this publication was assembled to support a Lecture Series under the sponsorship of the Guidance and Control Panel and the Consultant and Exchange Programme of AGARD.

AGARD Lecture Series No.109 Advisory Group for Aerospace Research and Development, NATO FAULT TOLERANCE DESIGN AND REDUNDANCY MANAGEMENT TECHNIQUES Published September 1980 182 pages These lectures are intended to provide the basic theory on concepts involved in the application of advanced software, state estimation, and implementation techniques involved in redundancy management, and to give a review covering the necessary background and state-of-the-art involved in the application of advancing technologies. P.T.O.	AGARD-LS-109 Redundancy Management methods Computer programs Failsafe systems Reliability (electronics) Failure Guidance and control	AGARD Lecture Series No.109 Advisory Group for Aerospace Research and Development, NATO FAULT TOLERANCE DESIGN AND REDUNDANCY MANAGEMENT TECHNIQUES Published September 1980 182 pages These lectures are intended to provide the basic theory on concepts involved in the application of advanced software, state estimation, and implementation techniques involved in redundancy management, and to give a review covering the necessary background and state-of-the-art involved in the application of advancing technologies. P.T.O.	AGARD-LS-109 Redundancy Management methods Computer programs Failsafe systems Reliability (electronics) Failure Guidance and control
AGARD Lecture Series No.109 Advisory Group for Aerospace Research and Development, NATO FAULT TOLERANCE DESIGN AND REDUNDANCY MANAGEMENT TECHNIQUES Published September 1980 182 pages These lectures are intended to provide the basic theory on concepts involved in the application of advanced software, state estimation, and implementation techniques involved in redundancy management, and to give a review covering the necessary background and state-of-the-art involved in the application of advancing technologies. P.T.O.	AGARD-LS-109 Redundancy Management methods Computer programs Failsafe systems Reliability (electronics) Failure Guidance and control	AGARD Lecture Series No.109 Advisory Group for Aerospace Research and Development, NATO FAULT TOLERANCE DESIGN AND REDUNDANCY MANAGEMENT TECHNIQUES Published September 1980 182 pages These lectures are intended to provide the basic theory on concepts involved in the application of advanced software, state estimation, and implementation techniques involved in redundancy management, and to give a review covering the necessary background and state-of-the-art involved in the application of advancing technologies. P.T.O.	AGARD-LS-109 Redundancy Management methods Computer programs Failsafe systems Reliability (electronics) Failure Guidance and control

The material in this publication was assembled to support a Lecture Series under the sponsorship of the Guidance and Control Panel and the Consultant and Exchange Programme of AGARD, presented on: 13–14 October 1980 in Athens, Greece; 16–17 October 1980 in Rome, Italy and 20–21 October 1980 in London, UK.

ISBN 92-835-0274-4

The material in this publication was assembled to support a Lecture Series under the sponsorship of the Guidance and Control Panel and the Consultant and Exchange Programme of AGARD, presented on: 13–14 October 1980 in Athens, Greece; 16–17 October 1980 in Rome, Italy and 20–21 October 1980 in London, UK.

ISBN 92-835-0274-4

The material in this publication was assembled to support a Lecture Series under the sponsorship of the Guidance and Control Panel and the Consultant and Exchange Programme of AGARD, presented on: 13–14 October 1980 in Athens, Greece; 16–17 October 1980 in Rome, Italy and 20–21 October 1980 in London, UK.

ISBN 92-835-0274-4

The material in this publication was assembled to support a Lecture Series under the sponsorship of the Guidance and Control Panel and the Consultant and Exchange Programme of AGARD, presented on: 13–14 October 1980 in Athens, Greece; 16–17 October 1980 in Rome, Italy and 20–21 October 1980 in London, UK.

ISBN 92-835-0274-4

B285
4

AGARD
NATO OTAN
7 RUE ANCELLE · 92200 NEUILLY-SUR-SEINE
FRANCE
Telephone 745.08.10 · Telex 610176

DISTRIBUTION OF UNCLASSIFIED AGARD PUBLICATIONS

AGARD does NOT hold stocks of AGARD publications at the above address for general distribution. Initial distribution of AGARD publications is made to AGARD Member Nations through the following National Distribution Centres. Further copies are sometimes available from these Centres, but if not may be purchased in Microfiche or Photocopy form from the Purchase Agencies listed below.

NATIONAL DISTRIBUTION CENTRES

BELGIUM
Coordonnateur AGARD – VSL
Etat-Major de la Force Aérienne
Quartier Reine Elisabeth
Rue d'Evere, 1140 Bruxelles

CANADA
Defence Science Information Services
Department of National Defence
Ottawa, Ontario K1A OK2

DENMARK
Danish Defence Research Board
Østerbrogades Kaserne
Copenhagen Ø

FRANCE
O.N.E.R.A. (Direction)
29 Avenue de la Division Leclerc
92320 Châtillon sous Bagneux

GERMANY
Fachinformationszentrum Energie,
Physik, Mathematik GmbH
Kernforschungszentrum
D-7514 Eggenstein-Leopoldshafen 2

GREECE
Hellenic Air Force General Staff
Research and Development Directorate
Holargos, Athens

ICELAND
Director of Aviation
c/o Flugrad
Reykjavik

ITALY
Aeronautica Militare
Ufficio del Delegato Nazionale all'AGARD
3, Piazzale Adenauer
Roma/EUR

LUXEMBOURG
See Belgium

NETHERLANDS
Netherlands Delegation to AGARD
National Aerospace Laboratory, NLR
P.O. Box 126
2600 A.C. Delft

NORWAY
Norwegian Defence Research Establishment
Main Library
P.O. Box 25
N-2007 Kjeller

PORTUGAL
Direcção do Serviço de Material
da Força Aerea
Rua da Escola Politécnica 42
Lisboa
Attn: AGARD National Delegate

TURKEY
Department of Research and Development (ARGE)
Ministry of National Defence, Ankara

UNITED KINGDOM
Defence Research Information Centre
Station Square House
St. Mary Cray
Orpington, Kent BR5 3RE

UNITED STATES
National Aeronautics and Space Administration (NASA)
Langley Field, Virginia 23365
Attn: Report Distribution and Storage Unit

THE UNITED STATES NATIONAL DISTRIBUTION CENTRE (NASA) DOES NOT HOLD STOCKS OF AGARD PUBLICATIONS, AND APPLICATIONS FOR COPIES SHOULD BE MADE DIRECT TO THE NATIONAL TECHNICAL INFORMATION SERVICE (NTIS) AT THE ADDRESS BELOW.

PURCHASE AGENCIES

Microfiche or Photocopy
National Technical
Information Service (NTIS)
5285 Port Royal Road
Springfield
Virginia 22161, USA

Microfiche
Space Documentation Service
European Space Agency
10, rue Mario Nikis
75015 Paris, France

Microfiche
Technology Reports
Centre (DTI)
Station Square House
St. Mary Cray
Orpington, Kent BR5 3RF
England

Requests for microfiche or photocopies of AGARD documents should include the AGARD serial number, title, author or editor, and publication date. Requests to NTIS should include the NASA accession report number. Full bibliographical references and abstracts of AGARD publications are given in the following journals:

Scientific and Technical Aerospace Reports (STAR)
published by NASA Scientific and Technical
Information Facility
Post Office Box 8757
Baltimore/Washington International Airport
Maryland 21240, USA

Government Reports Announcements (GRA)
published by the National Technical
Information Services, Springfield
Virginia 22161, USA

Printed by Technical Editing and Reproduction Ltd
Harford House, 7–9 Charlotte St, London W1P 1HD

FAULT TOLERANCE DESIGN AND REDUNDANCY