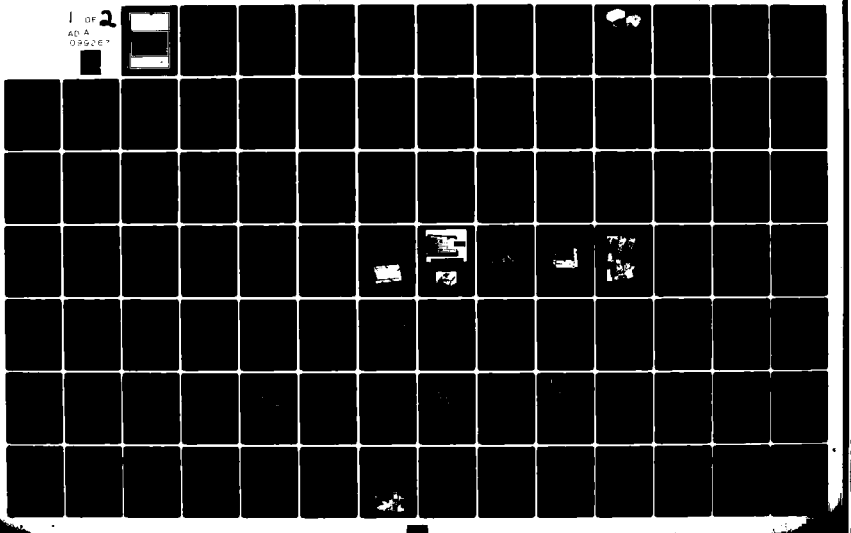
AD-A099 267 ADVISORY GROUP FOR AEROSPACE RESEARCH AND DEVELOPMENT--ETC F/G 9/2
MICROCOMPUTER APPLICATIONS IN POWER AND PROPULSION SYSTEMS.(U)
MAR 81
UNCLASSIFIED AGARD-LS-113 NL



AGARD-LS-113

AGARD

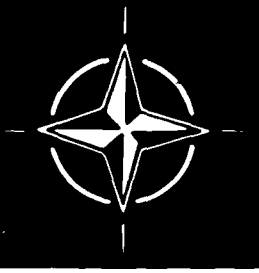
ADVISORY GROUP FOR AEROSPACE RESEARCH & DEVELOPMENT

7 RUE ANCELLE 92200 NEUILLY SUR SEINE FRANCE

AGARD LECTURE SERIES No. 113

Microcomputer Applications in Power and Propulsion Systems

NORTH ATLANTIC TREATY ORGANIZATION



DISTRIBUTION AND AVAILABILITY
ON BACK COVER

AGARD-LS-113

NORTH ATLANTIC TREATY ORGANIZATION

ADVISORY GROUP FOR AEROSPACE RESEARCH AND DEVELOPMENT

(ORGANISATION DU TRAITE DE L'ATLANTIQUE NORD)

AGARD Lecture Series No.113

MICROCOMPUTER APPLICATIONS IN POWER AND PROPULSION SYSTEMS

The material in this publication was assembled to support a Lecture Series under the sponsorship of the Propulsion and Energetics Panel and the Consultant and Exchange Programme of AGARD presented on 2–3 April 1981 in London, UK, 6–7 April 1981 at Oberpfaffenhofen, Germany and 9–10 April 1981 in Genoa, Italy.

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 March 1981

ISBN 92-835-1381-9

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

INTRODUCTION

The use of digital computers in aircraft has grown tremendously over the last ten years. Originally, the use was for noncritical functions such as navigation and more recently the use has spread to many more areas including the more critical functions such as flight control (e.g. DC-9-80, B-767). Over the last five years, microprocessors have become widespread throughout many industries and are being studied extensively for aerospace applications. Due to the great demand from nonaerospace industries, however, the microprocessor manufacturers have been reluctant to build flight-qualified versions of their microprocessors for the relatively limited volumes of the aerospace market. In the coming years the technology of microprocessors will no doubt enter flight qualified hardware in some fashion and will have the same impact as on other industries: i.e. prices will drop for a given level of computation.

Aircraft propulsion systems have become more complex simultaneously with the revolution in digital computers, particularly in terms of the number of functions to be controlled. Therefore, the application of digital computers and ultimately microprocessor technology to aircraft propulsion systems will eventually come about.

This lecture series is designed to acquaint the propulsion control system designer with the important aspects of digital control systems with specific attention to microprocessors where available. The first topic deals with digital hardware; beginning with a discussion of the various microprocessor characteristics as applied to propulsion controllers (Burrage) and ending with a discussion of digital system packaging and its best aircraft location (Vizzini).

The second topic deals with software. A discussion of the design methods for the algorithm in any digital control system contains a section on small word size and slow sampling effects, both of which take on added importance when using microprocessors (Powell). Software programming languages and checkout procedures are then discussed with an emphasis on good overall management techniques of the entire software generation activity (Miller).

The next topic concerns the area of digital system test and monitoring. The discussion covers the system hardware, real time operating system, system software, and control software (Evans).

Reliability is then discussed with specific attention to safety analysis and prediction through the use of fault trees (Evans). Furthermore, basic techniques in arriving at reliable software along with methodologies for its verification are covered (Amoia).

Applications of digital propulsion control systems are then discussed (Collins) and (Miller).

J.DAVID POWELL
Professor
Dept. Aeronautics and Astronautics
Stanford University
Stanford, Calif. 94305

LIST OF SPEAKERS

Lecture Series Director: Professor J.D.Powell
Department of Aeronautics and Astronautics
Stanford University
Stanford, California 94305
USA

SPEAKERS

Professor V.Amoia
Istituto di Macchine
Politecnico di Milano
Piazza Leonardo da Vinci 32
20133 Milan
Italy

Mr C.Beounes
Centre National de la Recherche Scientifique
Laboratoire d'Automatique et d'Analyse des Systèmes
7 Avenue du Colonel Roche
31400 Toulouse
France

Mr R.G.Burrage
Lucas Aerospace Ltd.
Shaftsmoor Lane
Birmingham B28 8SW
UK

Mr J.F.O.Evans
Smiths Industries Ltd.
Winchester Road
Basingstoke
Hampshire RG22 6HP
UK

Mr R.Miller
Pratt-Whitney Aircraft, R-18
P.O.Box 2691
West Palm Beach
Florida 33402
USA

Mr R.Vizzini
NAPC – Code PE 43
Box 7176
Trenton
New Jersey 08628
USA

CONTENTS

MICROPROCESSOR CHARACTERISTICS AND COMPARATIVE FEATURES

by

R G Burrage
Lucas Aerospace Limited
York Road, Birmingham B28 8LN
United Kingdom

SUMMARY

A modern design of control for a gas turbine engine is used to introduce the concept of computer control. This shows the function of the microprocessor, which associated circuits are needed to complete the control, and the features of the microprocessor that suit it to control tasks.

Many tasks other than control can be undertaken by microprocessors. These are discussed next to establish which general features are of importance to propulsion systems.

These features, plus the normal criteria applied to the procurement of any component, can be used as a guide to the selection of a microprocessor. Using this approach comparisons are made of different manufacturer's products.

1. INTRODUCTION

In 1972 the advanced projects team at Lucas surveyed the digital computer scene; the first of the commercial microprocessors had arrived, but where were the military specification devices that were needed in engine controls? None were available, and it was not easy to predict who would be the first manufacturer to provide anything approaching the computer capability of the mini computers that we were using in experimental engine controls at that time.

Now the variety of microprocessor designs from different manufacturers and even from individual manufacturers is truly impressive. Table 1 shows 46 microprocessor designs and the manufacturers who are their original design source. However, this is only part of the story. Some designs are also manufactured by other suppliers so the actual number of manufacturers is much greater than named here. Some of the microprocessors are just one member from a whole family of devices, for instance the 8048 is the first member of a family of 12 closely related microprocessors made by Intel. There are many other families and many original designs and manufacturers omitted, so table 1 is perhaps only the tip of the iceberg!

AMD		**Motorola**		**Rockwell**	
2900s,	4, B, M, 3	14500,	1, C, M, 0	PPS-4/1,	4, P, 0, 0
29116,	16, B, 0, 0	6802,	8, N, M, 5	6500s,	8, N, 0, 2
AMI		6805s,	8, N/C, 0, 2	**Texas Instruments**	
S2000s,	4, N, 0, 0	6809,	8, H, M, 3	TMS1000s,	4, P/C, 0, 0
Commodore		6801,	8, H, 0, 2	TMS9980s,	8, N, 0, 2
650Xs,	8, N, M, 2	68000,	16, H, 0, 1	TMS9900,	16, N, 0, 2
Data General		**Mostek**		SBP9900,	16, B, M, 0
mN601s,	16, N, 0, 0	3870,	8, N, 0, 2	**Western Digital**	
Fairchild		**National Semiconductor**		1872/2272,	4, P, 0, 0
F8,	8, N, 0, 3	COP400s,	4, N, 0, 1	**Zilog**	
9400,	4, B, M, 1	NSC800,	8, C, 0, 0	Z8,	8, N, 0, 1
F10023X,	8, B, 0, 1	8060,	8, N, 0, 0	Z80,	8, N, M, 5
9445,	16, B, 0, 0	8070,	8, N, 0, 0	Z8000,	16, N, 0, 2
Ferranti		16000,	16, N, 0, 0		
F100-L,	16, B, M, 0	**NEC**			
Intel		uCOM42,	4, P, 0, 0		
8048,	8, N, 0, 6	uPD75XX,	4, P/C, 0, 0		
8051,	8, H, 0, 0	uCOM87,	8, N, 0, 0		
8085,	8, N, M, 2	**OKI**			
8086,	16, N, 0, 2	MSM5840,	4, C, 0, 0		
iAPX432,	32, H, 0, 0	**Panasonic**			
2920,	25, N, 0, 0	MN1400,	4, N, 0, 0		
Intersil		**RCA**			
6100,	12, C, M, 1	1802s,	8, C, M, 1		
80C48,	8, C, 0, 3				

Table 1 - The Wide Variety of Microprocessor Designs

As a first assessment of any microprocessor it is useful to know its data word length, the manufacturing process and how many manufacturers make it. For convenience this is represented on table 1 by the use of four terms enclosed in brackets, such as (4, B, M, 3) placed alongside each microprocessor. The first item gives the length of the data word in bits and typically may have the values 1, 2, 4, 8, 12 or 16. The second term gives the semiconductor process in which it is manufactured:

P	PMOS
N	NMOS
C	CMOS
H	HMOS
B	Bipolar

and if available in two processes would be given as P/C meaning available in either PMOS or CMOS. The third term indicates operating temperature capability: M means it is believed that military temperature range devices are available and may be qualified to a full military specification, O means may not be available to a military temperature range. The fourth term is the number of manufacturers believed to second-source the design either by a licence or by copy.

From this present day wide range of microprocessors, the question is which devices are suitable for propulsion controls. This paper provides some answers - hopefully in a way that will be constructive to microprocessor users and suppliers - acting as a reminder of those features that can be important in power plant propulsion systems. Before attempting a selection let us consider some of the background on powerplant controls.

Traditionally gas turbine engines have been controlled hydromechanically or by using analogue electronics. However many development projects and some production engines are being controlled digitally at the present time. The complexity of tasks undertaken ranges from single controls for throw away propulsion units, through controls for accessory power units, helicopter engines and large civil turbo fans to the complex controls needed for reheated engines used on supersonic transports and military aircraft. Table 2 shows Lucas's activity in this area and illustrates how microprocessors can be used even for the most exacting and complex tasks.

ENGINE APPLICATION	CONTROL BUILD STANDARD	PROCESSOR
Military Engine	Breadboard	SBP9900
Industrial Olympus	Production	SBP9900
RB211/535	Production	SBP9900
LTS101	Production	IM6100
Lucas APU	Production	IM6100
ATDE	Prototype	IM6100
Olympus 593	Prototype	Minicomputer
Advanced Full Authority	Breadboard	SBP9900
Missile Project	Prototype	MC6802

Table 2 - Lucas Controls run on engines in 1980

We have elected to use three different processors in these various applications and this prompts the question, are there fundamental differences in their use, or has the choice simply been a matter of detail? To my knowledge there are no fundamental differences - the selection is in the last analysis dependant upon the detailed differences in price, performance, availability and familiarity with suitable devices. By discussing some of these units in a general way I hope to show how such selections can be made.

Some of these control units are shown in Figure 1. Their external appearance reflects that they are designed for flight applications, to be mounted either within the airframe or directly on the engine. This brings us back immediately to the fact that any components, including microprocessors, used in these units must be to the high levels of quality needed for aircraft equipment and engine equipment in particular.

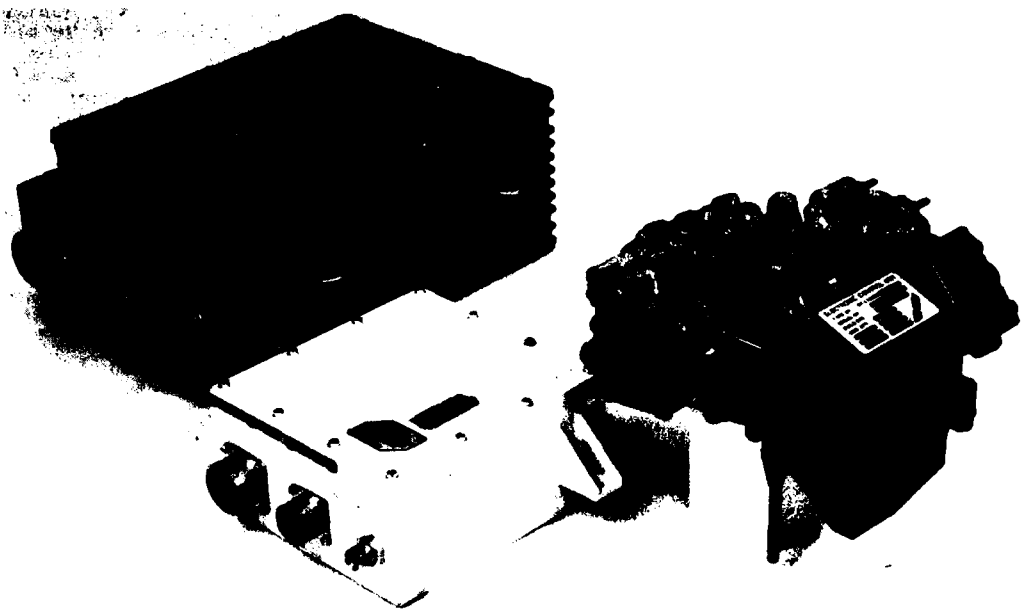


Figure 1 - Three Microprocessor Based Engine Controls

Usually the electronic components will be specified to MIL 883B, BS9000, or an equivalent high standard. This immediately eliminates many microprocessors which cannot meet the military temperature range, -55°C to +125°C, that is part of these standards. Often microprocessor manufacturers are not prepared to adjust manufacturing and selection processes to yield military temperature range devices simply because it would disrupt the existing product line. This is an important point because it emphasises that some designs of microprocessor and some manufacturing processes may be committed to entirely different markets. One way of separating out the different areas of interest is to split the microprocessor market according to temperature range and determine the value of each segment, this is shown on Figure 2.

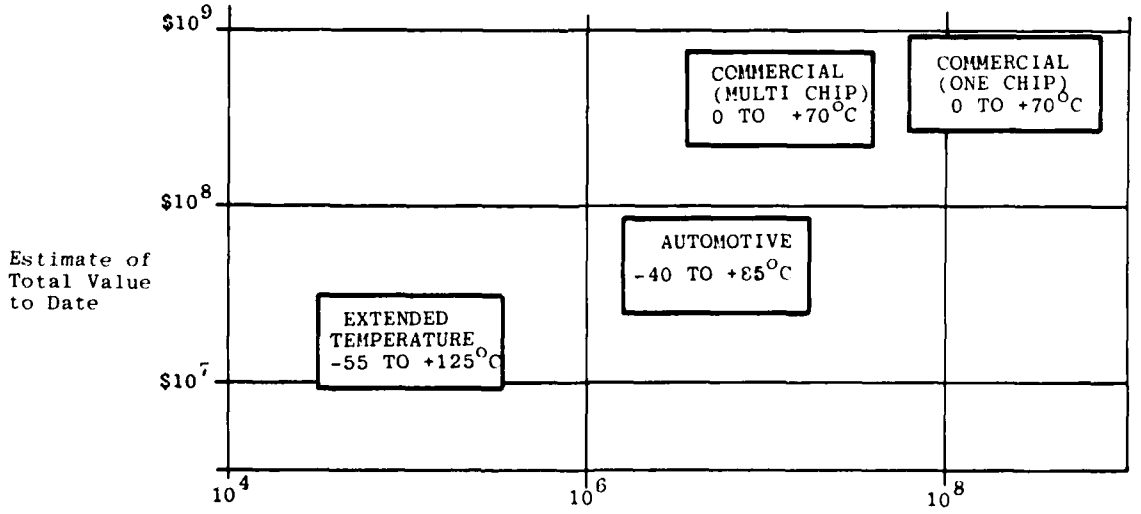


Figure 2 - Estimate of Number of Parts Shipped to Date

Temperature range does matter - first because the power control environment can be extreme, secondly because running a microprocessor near to the limits of its temperature specifications can force transient failures which do not show up when the unit returns to the normal temperature. So microprocessors that are available to the full military specification are of great interest. Some of the manufacturing processes, Bipolar and CMOS have excellent temperature capability and this alone may make particular processors of interest.

Production quantities for military components are small, for example the usage during 1980 of microprocessors in European defence applications has been estimated to be only 4% of the European total. By comparison with the vast commercial and fast growing automative microprocessors the "military only" designs are likely to be very expensive, with no back-up of vast production quantities or second sourcing. On the other hand "militarised" versions of popular commercial products will tend to be non-standard products and so relatively expensive, but because of the popular high yield processes used are likely to be second sourced.

These observations provide a basis for a first selection: is the microprocessor available to a full temperature range and military specification and available from a number of different suppliers? Unfortunately this criterion would exclude many important designs and so devices that are CMOS or Bipolar, or are full temperature range or that the manufacturer promises will eventually be available to a full military specification may need to be considered. Applying this arbitrary criteria to find 15 devices of possible interest from Table 1 has produced the selection listed in Table 3.

8048	Single chip, 8 bits, CMOS version by Intersil
1802	8 bit, CMOS by RCA, Military temperature range
8085	8 bit, NMOS by Intel, enhanced 8080
Z80	High performance 8 bit by Zilog
6802	8 bit, NMOS by Motorola, RAM and Clock on chip
146805	6800 subset, RAM, Clock, Timer on chip
6809	High performance 8 bit, by Motorola
6100	12 bit Mil Spec. CMOS, from Intersil
F100-L	Full military grade, 16 bit, Bipolar, by Ferranti
SBP9900	Full military grade, 16 bit, Bipolar, by T.I.
8086	By Intel, first of the new high performance, 16 bit, NMOS
Z8000	Zilog's reply to the 8086, also NMOS
68000	Even more advanced 16 bit HMOS design by Motorola
iAPX432	A future 32 bit NMOS, expected soon from Intel
F10022X	Very high speed, 8 bit slice, Bipolar, by Fairchild
2920	Interesting 'analogue' processor by Intel

Table 3 - A Selection of 15 Microprocessors

To refine this list further consider an application example of powerplant control and use it to provide more specific selection criteria.

2. APPLICATION EXAMPLE

Returning to the engine controls shown in Figure 1, the actual tasks performed are similar in each case so let us choose one as the application example. Controlling the thrust or power delivered by the engine is the primary duty, but in achieving this, other important tasks are performed: sequencing the engine through starting and shut down procedures, providing engine safety, limiting against inadvertent overspeeding or over temperature, and finally providing self test features for the purposes of safety (detecting faults and taking safe action) or for maintenance information. The microprocessor can be programmed to assist in each of these tasks, as represented by the electronic control system diagram in Figure 3.

The microprocessor and its memory are at the centre of all signal flow - it receives all the input signals, determines the appropriate control action, and sets the output signals accordingly. Usually there are safety circuits monitoring the power supply, output drive amplifiers and the microprocessor itself, which act independently of the microprocessor. Apart from these safety circuits, the microprocessor has complete command over all control action.

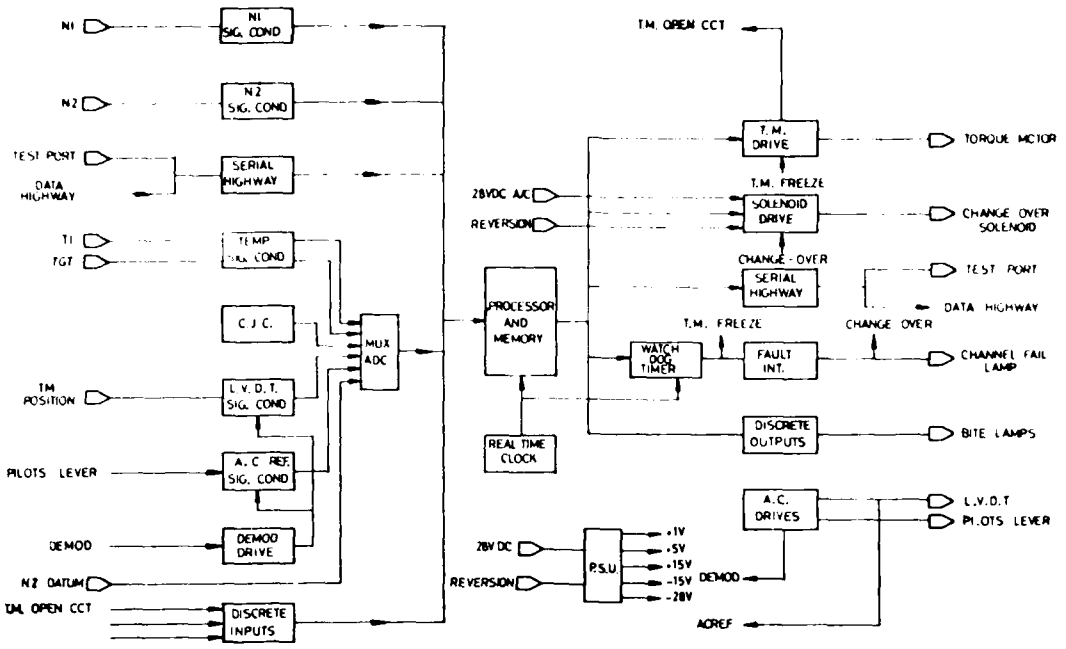


Figure 3 - Block diagram for example electronic control

The control program is an item by item sequence comprising 3000 or more instructions to be obeyed by the microprocessor and depending on its functional capability will take more or less time to complete its calculations. The dynamics of the engine and control specified determine the desired sample period of the control program, 20 milliseconds in the present example, and so the microprocessor is required to work within this limitation. Typically the 20 millisecond sample period is spent on responding to the real time clock to start the new period, reading inputs on control calculations for power or sequencing operations, on limit calculations and on self test, and finally setting the outputs that determine fuel flow to the powerplant. Table 4 below lists some of the tasks undertaken.

Power-up	-	check whether engine is out, windmilling or already running, initialise control variables as appropriate.
Built in Test	-	check that BITE enable signal is present and that engine is not running, if okay go through tests responding to cockpit prompts, and display fault codes etc.
Starting	-	initialise for starting, set light-up flow, detect light-up, accelerate engine to idle, control engine stably at idle.
Range control	-	control engine on range control parameter engine speed, pressure ratio as appropriate, in response to pilot's lever position.
Transients	-	control engine during transients so that it does not accelerate into stall or decelerate into flame-out regions.
Limiting	-	protect engine from running beyond transient or steady state limits of temperature, speeds, pressures etc.
Shut-down	-	shut-down outputs in good order.
Safety	-	check all inputs and outputs for credibility, freeze outputs during transient faults, revert control for persistant fault.
Ground Test	-	service ground test port via serial highway.

Table 4 - Tasks performed by the microprocessor

If a microprocessor with poor functional capability were to be used the control designers would be faced with a trade-off situation to preserve the desired sample period. This could involve reducing the accuracy of calculations, eliminating or simplifying software safety checks by moving some functions into hardware, generally reducing the microprocessor task at the expense of system performance or increased hardware complexity.

Usually the hardware complexity and cost of the engine control is dominated by the number and signal conditioning needs of its input transducers and output drive functions and by the power supplies, see Table 5.

Circuit Function	Percentage share by Volume	Cost
Microprocessor, memory etc.	10	20
Digital/analogue/time interfacing	10	10
Input transducer conditioning	20	15
Discretes and test highways	10	10
Control output conditioning	20	15
RFI and power supplies	30	30

Table 5 - Cost and volume contributed by circuit functions within a powerplant control.

The microprocessor, its memory and its interface components take up a relatively small proportion of the space available, considering the great complexity of the tasks the microprocessor undertakes. This of course is the result of the high level of integration achieved in digital circuit technology exemplified by microprocessors. Because this technological change continues apace, it presents the control designer with further dilemmas. New microprocessors may offer very high functional capability that would allow a better quality of control, BITE etc. or enable him to simplify the hardware in the input signal conditioning, output drive and power supply circuits. Therefore the new device may promise a better control at a lower cost, but what risks are involved? Principally there are two risks to be considered - the technical risk during development and the procurement risk during manufacture and customer support of the control. Bearing in mind that the production quantities for engine controls are relatively small and the timescales for product support are relatively long, technical and procurement risks must be taken very seriously.

Policy decisions and exceptional design requirements will alter the selection criteria considerably, so the control designer must establish these mandatory criteria before starting the selection process. The paragraphs that follow discuss technical function, and the ease of development of the product, effects on procuring the final product, where the criteria are applied according to the author's personal opinion and no company policy on the part of Lucas Aerospace Limited is implied.

3. METHOD OF COMPARISON

The method used below involves calculating three indices for each microprocessor and then combining them, by multiplication, to obtain a figure of merit that represents the relative suitability of the microprocessor for a particular powerplant control application. The advantage of the method is its simplicity. It uses a check list of questions for each index that can be answered by simple yes/no procedures. A disadvantage is that the importance placed on each question is numerically weighted and each weighting factor is a judgment that needs to be reassessed from one control project to another.

The indices reflect the three major questions to be asked when choosing any component for a production unit: is it functionally suitable, is its procurement status suitable for our production contract, is it going to be easy for us to develop the unit to the full production standard using this component? If the answers to all these questions are favourable then the component is suitable. If any of the answers is unfavourable then the component is unsuitable. For example supposing a component is easy to use (within its specified limits) during development and presents no procurement problems for production but does not have the capacity for the required function - it is definitely unsuitable. Similarly suppose that a component has an excellent functional specification but is going to make the development task or production difficult - it is definitely unsuitable. Transposing these comments into the three numerical indices implies that good scores are needed on all three indices, a zero score on any one index produces zero figure of merit for suitability. This multiplicative relationship is shown in Table 6.

Index	Symbol
Function	X_f
Procurement Status	X_p
Ease of Development	X_d
Figure of merit for suitability	$M = X_f \cdot X_p \cdot X_d$

Table 6 - Definition of Suitability

The method used to define each index is first to write down a check list of relevant questions, expressed so that yes/no answers can be used. Then each question is given a weighted score mark for a 'yes' answer, the weighting chosen to reflect the importance of the question. The index value for a particular microprocessor is calculated by answering the check list and summing the scores for each question achieving a 'yes'.

Note that the accuracy of the indices and the resulting figure of merit are adequate to produce a short list - but not accurate enough for design decisions. The real choice of microprocessor always should be based on assessing detailed design proposals. Now consider how the 15 microprocessors selected compare, firstly assessed for Function.

4. FUNCTION

In the example control the microprocessor obeys a well defined program to act upon all incoming and outgoing signals. This program, that is the sequence of instructions and the fixed data, is stored in permanent memory, in this instance Programmable Read Only Memory, PROM, see Figure 4. In working through the program the microprocessor is instructed to fetch or send data to an input or output device and to manipulate data representing the control variables using its internal registers and external temporary storage in the random access memory, RAM. Some of the program tasks involve self-test and monitoring of the unit and produce variable data that is to be stored or accumulated from flight to flight, and must therefore not be lost when electrical supplies are interrupted or shut down at the end of a flight. Such data is stored in non-volatile memory called Electrically Alterable Read Only Memory, EAROM.

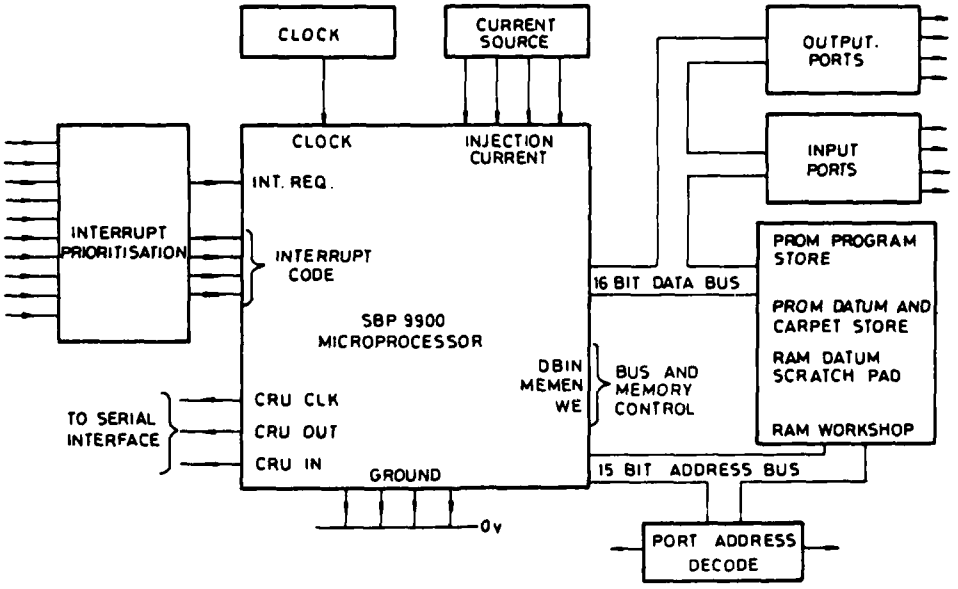


Figure 4 - The Microprocessor and Associated Components

A single run through the control part of a program varies, taking about 16 milliseconds on average, depending on the "path". Order is imposed on this somewhat irregular arrangement by the Real Time Clock, RTC, which promps the microprocessor to repeat the control tasks at precisely 20 millisecond intervals. This has a number of advantages. First it means that the time dependant calculations using the sample rate for integration, dynamic filtering, sequencing of simply tuning are now based on an accurate interval rather than a randomly variable interval. Secondly there is during each 20 millisecond interval some free time left over after the control calculations have been completed, this free time is used for lower priority tasks in base level, which continue at a more leisurely rate and can be interrupted safely. Finally the microprocessor's response to the RTC and to a fixed time test task is monitored by a Watch Dog Timer, WDT, which expects to be addressed within a narrow time window every control interval. This simple safety circuit confirms that the microprocessor is responding sensibly to the RTC, and that RTC itself is functioning within tolerance.

The real time nature of the control tasks has introduced another function of microprocessor; it must be capable of being interrupted in order to respond to external events. Figure 5 shows how the interrupt facility is used in this engine control.

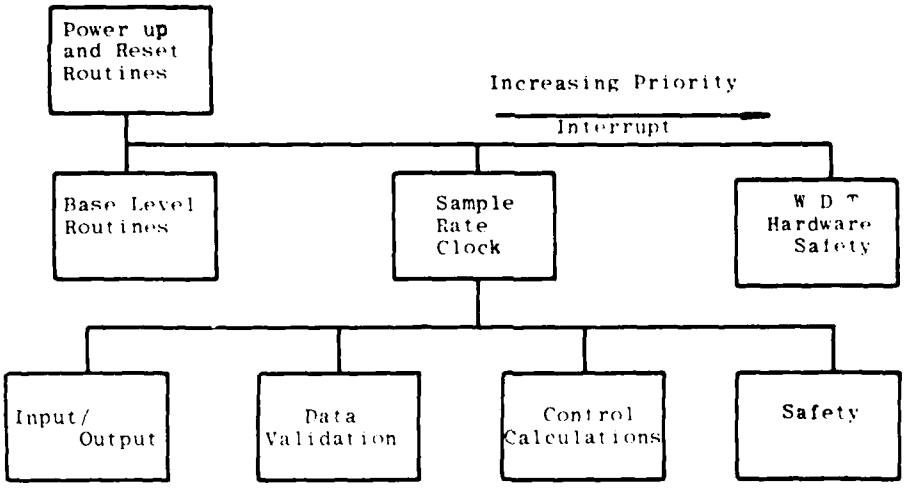


Figure 5 - Interrupt Structure

The highest priority interrupt is associated with servicing the Watch Dog Timer hardware, the sample rate clock has the next highestpriority causing the main program tasks for input/output, data validation, control calculations and safety to be performed. Ideally the base level should be used for ground test routines, built-in-test routines, fault recording and similar 'slow' tasks.

The input/output routines involve exchanging data with analogue to digital converters, digital to analogue converters, discrete inputs from switches, discrete outputs to relays and lamps, and any other hardware that links the microprocessor to its tasks. The accuracy and resolution required when converting the analogue signals is likely to be 8, 10 or 12 bits and this in turn imposes an accuracy requirement on the software calculations within the microprocessor. An 8-bit microprocessor can be programmed to work at higher accuracy, for example 16-bit accuracy is possible because the microprocessor can use two eight bit data words to represent a single 16-bit answer. Working at higher accuracy involves more program steps and therefore slows the program down. This is acceptable if the program is completed well within the sample data target, in the present example 20 milliseconds. If this is difficult then choosing an 8-bit machine with 16-bit intervals, or choosing a 12-bit or 16-bit machine may be one way to solve the accuracy/time problem.

The accuracy/time target is one example of how a software design aspect can influence the design and choice of hardware. Another is 'ease' of programming. Programming involves choosing one out of 100 types for each of the 3000 or so instructions in the control program, and Lucas use a high level language to make this complex task a manageable and well structured activity. Our choice of microprocessor will be influenced by whether its instruction set helps or hinders the high level language and this is best explained by describing the language and its purpose and identifying the types of instruction that are useful.

No matter how well designed the hardware of a microprocessor based control system may be, the success of the project hinges on the quality of the software used. Quality, in this context, encompassing - simplicity of application, visibility, controllability and modification capability. When considering the needs of a suitable high level language for control system applications, six objectives of such a language were identified, these being:

> The language to be, as far as the control engineer is concerned, independant of the microprocessor being used.
>
> The control engineer to be relieved of the need to have an extensive knowledge of the microprocessor being used.
>
> The user interface to be tailored to produce an application-oriented programming system so that the control engineer can transcribe his control system design into runnable software accurately and rapidly.

The software stages to have full visibility so that traceability of the software from the original control system design to the runnable code in the system store is ensured.

The software system to be so structured that good design practices are enforced and that rigorous testing and cross checking can be applied.

The software system to be structured so that the raising of high quality documentation becomes a natural and integral part of the software writing process.

With these six objectives in mind, together with a knowledge of the technical needs of the control engineer, Lucas Aerospace developed a high level language known as LUCOL (LUcas COntrol Language) as part of a total software system package. The language enables control engineers to program directly from their control diagrams thus enabling them and other engineers to have full software visibility in a form traditional to engineering disciplines.

The basis of the language is a series of modules representing commonly used analogue type control system blocks as well as sequential logic operations. The control engineer solves his problem by specifying an appropriately ordered network of modules. These modules are drawn from a library of rigorously tested assembly language programs which also includes input/output and safety routines.

Each module has assigned to it a mnemonic and a standard functional diagram. The control engineer draws his system block diagram using the LUCOL elements - this block diagram then forms a pictorial representation of the software, see Figure 6a.

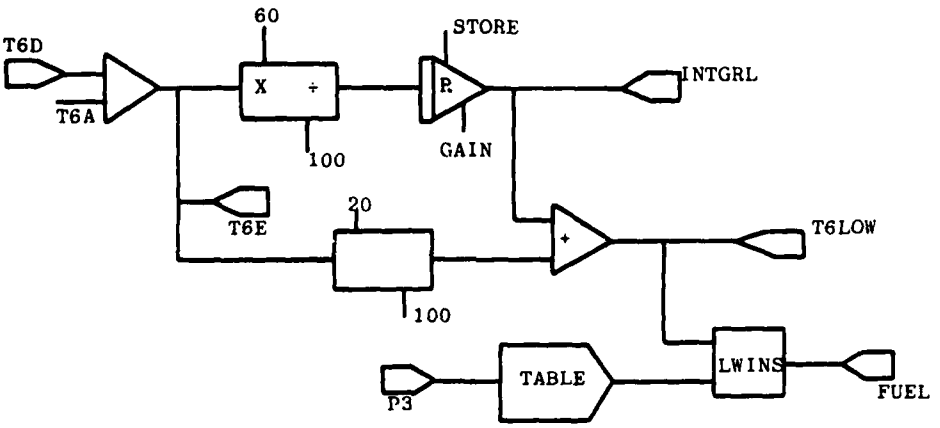


Figure 6a - LUCOL example: block diagram

The example shows a proportional plus integral control loop for a variable T6 which, when passed through a lowest wins with a function of P3, generates a fuel demand signal. The next step is to list this in standard LUCOL form, as shown below:

```
GET         (T6D)            ;    Read T6D into implicit register
SUB         (T6A)            ;    Subtract T6A to form error signal
PUT         (T6E)            ;    Save error signal
MUL DIV     (*,60,100)       ;    Scale error by 0.6
INTR        (GAIN, STORE)    ;    Integrate result
PUT         (INTGRL)         ;    Save result
MUL DIV     (T6E,20,100)     ;    Scale T6E by 0.2
ADD         (INTGRL)         ;    Add integral term
PUT         (T6LOW)          ;    Save result
GET         (P3)             ;    Read P3 into implicit register
FGEN1       (TABLE)          ;    Look up P3 schedule
LWINS       (T6LOW)          ;    Lowest wins with T6 loop error
PUT         (FUEL            ;    Save fuel demand
```

Figure 6b - Example LUCOL list

The listing can be seen to have a good correlation with the block diagram: the basic control program is generated simply by producing a calling sequence listing the modules, in mnemonic form, and their associated parameters. These parameters specify the data flow between modules (analogous to the signal flow on a conventional block diagram) and the direct parameters such as gains, time constants etc. Comments are added by using a semicolon to separate each LUCOL statement from its comment. Figures 6(a) and 6(b) show the visibility achieved in this high level language source program. The LUCOL translator comtinues this visibility translating the source program into runnable code by preserving the list virtually unchanged except that now each item is the binary address of the LUCOL module rather than merely a mnemonic, or is the binary address of the variable in the random access memory.

These features of LUCOL depend on the use of subroutines, each module is a well specified subroutine, and on threaded code which allows a list of mnemonics or addresses to completely define the control programs in which the modules are being used. A microprocessor with good instructions for handling subroutines, indirect addressing and auto-incrementing will assist the efficiency and visibility of this type of high quality software.

From all of the above discussion the following general questions can be asked of a microprocessor: what is its data word length, what is the average time for an instruction, can it deal effectively with input and output duties, is it good at control arithmetic and subroutine handling, do all these functions come in one neat physical package or set of chips? Table 7 below expands on these questions to make a check list for the functional index.

Data word length	-	8 bits or less 12 bits 16 bits or more
Instruction average	-	longer than 5 microseconds 2 to 5 microseconds less than 2 microseconds
Input/Output features	-	single interrupt multiple interrupt bit handling serial I/O timer I/O analogue I/O
Arithmetic features	-	multiply divide double precision
Addressing structure	-	subroutine call indirect addressing auto-increment
Hardware integration of the above functions achieved with	-	on-chip RAM one chip 2 or 3 chips 4 or more chips

Table 7 - Check List of Function

Applying the checklist to the 15 microprocessors selected in Table 3, and with suitable weighting factors, has produced the histogram for the functional index shown in Figure 7.

By this index all but one of the microprocessors are functionally stable. The exception is the Fairchild F10022X 8 bit ECL slice which, being aimed at the highest performance 64 bit computers, would require considerable effort and complexity to bring it down to the level of powerplant control.

At the 8 bit end of the scale the Zilog Z80 and Motorola 6809 score well because both are fast with good instruction sets.

The Z80 is a very powerful processor but lacks certain features which would make threaded code very easy, principally it requires three instructions for a parameter acquisition as opposed to two instructions, for example in the 6809 or the 9900. However the implementation of our control language in the Z80 would still be straightforward.

The 6809 has many internal 16 bit features, is ideal for threaded code having comprehensive auto-incrementing and indirect addressing, also it has a useful 8 x 8 bit multiply instruction. Its powerful instruction set allows programs to be written with very efficient use of memory space and would make this microprocessor our preferred 8 bit choice from a software point of view.

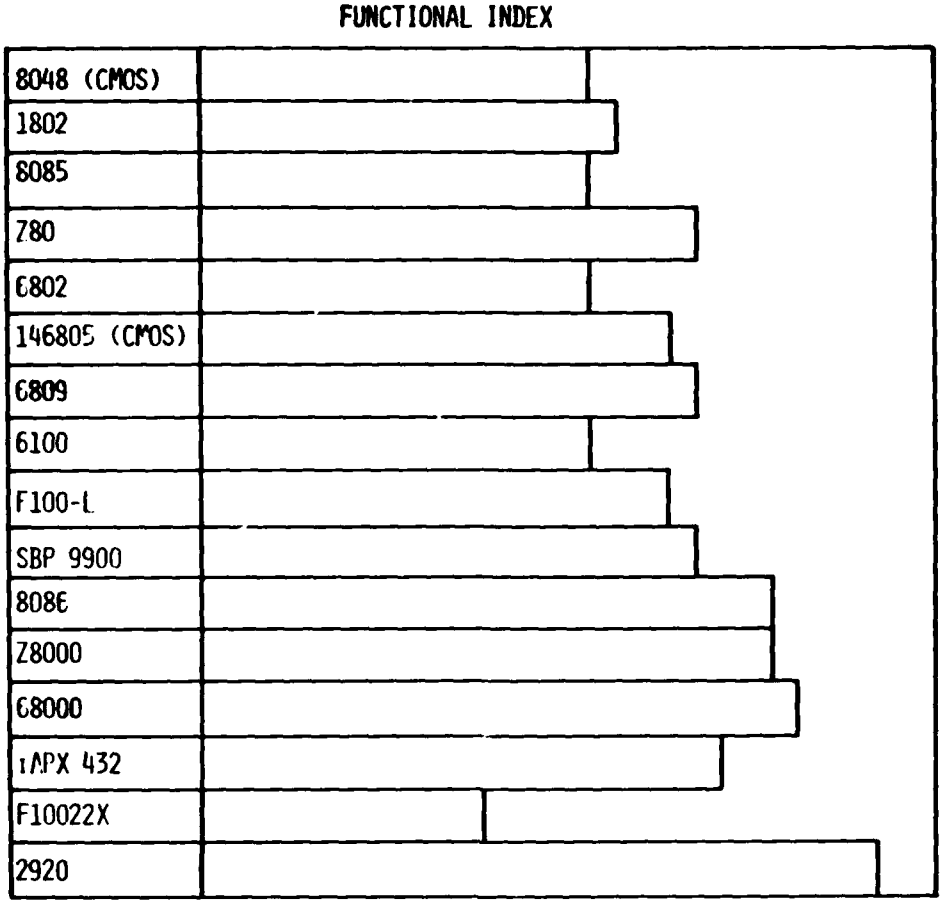


Figure 7 - Histogram of Functional Index

The Motorola 6802 and 6805, although much slower, are potentially attractive because of the high level of physical integration - both include RAM and the 6805 has an event timer on chip. The structure of these machines (the 6805 has a variation of the 6802 instruction that allows more efficient bit manipulation) is not suited to the implementation of threaded code. This is because they lack indirect addressing and auto-increment codes - a deficiency that can be overcome by the use of subroutine calls and in-line code but with speed and space penalties.

All the 16 bit microprocessors score well because of their speed and 16 bit arithmetic. In general they need a number of support chips and interface circuits to link with memories and input and output devices. This gives a poor score on physical integration - but is outweighed by their computing speed and powerful instruction sets. Certainly the Motorola 68000 looks most attractive for high quality software. Its powerful arithmetic instructions (includes signed and unsigned multiply and divide) and program manipulation instructions are well suited to a threaded code approach to engine control programs.

The 68000 incorporates features designed to make program development more reliable. Its structure is highly regular so that each type of operation is uniquely defined and independent of the type of data on which it operates and of the data addressing modes. This makes the instruction set consistent in use and easier for the programmer to remember and understand. Special hardware traps are provided to indicate various illegal instructions, addressing modes or overflows. These are complemented by 16 software traps that the programmer can use to program error detection routines suited to his specific application.

The Zilog Z8000 and the Intel 8086 approach the computing power of the 68000. All three represent a considerable software advance upon the earlier 8 bit and 16 bit microprocessors.

The microprocessor with the highest score is complete unorthodox. Designed for processing analogue signals, the Intel 2920 has "on-chip" facilities that are quite different to conventional microprocessors which borrow the features of data-processing computers with few concessions to scientific computing needs and even fewer to processing analogue signals. So the 2920 scores well because it integrates all memory and a complete input/output interface to the analogue world within its processor chip, a specialised instruction set giving high speed and 25 bit working. At present it is limited by memory space and digital

interfacing - so is unlikely to be used for an engine control - however it should point the way for future devices either for a total control or acting as a peripheral to a more conventional microprocessor.

The brief functional assessment of the 15 microprocessors must now be supplemented by considering the effects on the final product and the ease of developing it.

5. PROCUREMENT STATUS

The manufacturers of the power or propulsion control or system expect the contract for their final product to define quantities, delivery schedules, guarantees of quality, weight, reliability, future availability and price. Technical merit in meeting the functional specification is worth nothing if the production contract cannot be met. It is therefore essential that selection of a microprocessor should be compatible with meeting the contract for the final product.

A suitable index can be constructed by considering quite simple procurement issues. For example, how does the choice of microprocessor affect the production cost of the final unit? Is the microprocessor manufactured by one, two or more independent sources? Is it available to a full military temperature range and quality specification, to automobile industry or only to commercial quality standards?

Proportion of bought out cost to be attributed to the microprocessor and associated digital components	-	10% or less between 10% and 20% 20% or more
What non-recurring expenditure will be required to handle these components in production	-	Use existing ATE need new specialising cards and schedules subcontract testing need new ATE
How many manufacturers make this microprocessor	-	1 source 2 sources 3 or more sources
Quality standard available	-	Qualified to a full military standard Military temperature range Motor industry temperature range Commercial temperature range
Manufacturing process	-	Bipolar CMOS NMOS

Table 8 - 'Procurement Status' Check List

Using the check list in Table 8 and suitable weighting factors to define a 'Procurement Status' index, and applying the index to our microprocessors, results in Figure 8.

This histogram indicates that there is a fair selection of 8 bit microprocessors that are equally suitable components for our specific type of final product. They score well because of their wide commercial success, having one or more second source of manufacturer, relatively low prices and well integrated designs. The resulting powerplant control will see this as a direct benefit in lower bought out costs, reduced printed circuit board area, and lower component availability. There is a further important benefit for the established 8 bit designs - the existence of automatic test equipment that can be used for bought out inspection and for card testing.

An area of doubt for the 8 bit machines is establishing how many of the sources provide the military specification part, - and the commitment to long term availability, after all most of their market is non-military. Many of the devices have been designed in the first place for business and office machines, computer peripherals, instrumentation, industrial controls, telecommunications and automotive applications. The bipolar and the CMOS devices have been well suited to the specialised needs of aerospace and defence applications and in some instances specifically funded via defence budgets. Since our customers are in the aerospace and defence markets this makes any microprocessor that is specifically committed to that market of great interest.

Some comments are worth making about the 16 bit microprocessors. the two bipolar microprocessors, the F100L from Ferranti and the SBP9900 from Texas Instruments, would have scored more highly if they were multiply sourced or even second sourced. Their bipolar semiconductor technologies enable high performance, high temperature capability and radiation hardness - yet have not persuaded other manufacturers to second source these useful microprocessors.

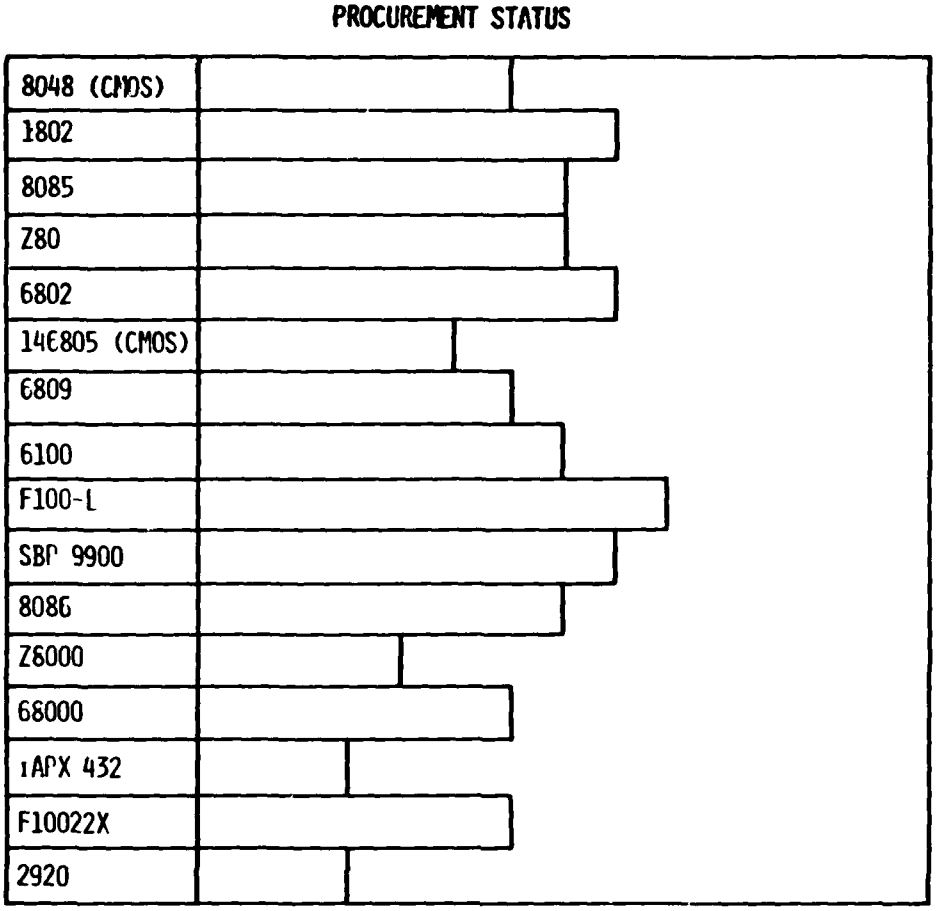


Figure 8 - Histogram of Procurement Status

The three new 16 bit microprocessors, the Intel 8086, the Zilog 8000 and the Motorola 68000 will achieve higher scores as they become more established. They are manufactured in N-channel MOS and so will not achieve the radiation hardness of the bipolar devices, but already have the advantage of being second sourced. The MOS technologies provide numerous examples of design portability.

Finally it is disappointing to note that two functionally interesting devices, the Motorola 146805 and the Intel 2920 score relatively badly: I believe neither are second sourced and available to full military temperature range and specifications, at the time of writing.

Armed with some feel for the function and product credibility, the next question is how the choice of microprocessor will affect development of the final product.

6. EASE OF DEVELOPMENT

The development of a digital system comprises a series of interconnected activities, which can be loosely described as hardware development, software development and integration of the software and hardware. The total activity becomes more efficient if the development team already have experience of the given microprocessor and have the appropriate development aids. An index for ease of development should consider what software is available and the complexity of the microprocessor relative to its task. The checklist shown in Table 9 raises many of the relevant questions.

The development phase of a powerplant is characterised by a process of learning that extends over many months. The controlling laws specified for the 'paper' engine will evolve and be changed to meet its actual needs as the design is developed from bench engines through to qualification of the final production standard. The control unit must be able to take all these modifications in its stride - preferably by software changes only. This means that the original choice of microprocessor should be well matched to its 'paper' task and have room for expansion. Flexibility is equally important, and this depends on the experience of the control project team and how well it is equipped with hardware and software development aids.

Is the computing capability and complexity of the device	-	adequate only for part of task well matched to task, with room for expansion excessive
Experience of the device within the control project design team	-	none indirect experience, via emulator indirect, via 'family' compatability direct experience to breadboard stage direct experience to prototype production direct experience from production units in service
Availability of development aids	-	'stand-alone' development system development system with in-circuit emulation available on one 'universal' development system available on a wide choice of 'universal' development systems
Cross-product software available	-	micro-code assembler assembler emulator etc. on time-sharing network assembler, emulator etc. in portable high level language
Resident software available	-	de-bug firmware assembler FORTRAN BASIC PASCAL CORAL

Table 9 - Check list for 'Ease of Development'

To achieve an easy development phase the microprocessor hardware, software and system integration must be trouble free - there will be more than enough technical challenges in following the 'moving' target of the powerplant as it is developed.

Choosing too much microprocessor computing capacity will aggravate development as much as choosing a microprocessor with too little computing capacity. This is because the simple microprocessors have been around a long time now, and are well down their learning curve by comparison with the newer more powerful designs. Every year more powerful and complex microprocessors appear, attempting to leap frog the technical merits of their competitors and predecessors. It is important to recognise that each new more powerful design is a coincidence of two major technological jumps for that manufacturer - a new complexity of computer and a new or improved semiconductor process capable of yielding chips of that complexity. The resulting microprocessors are at the start of their learning curves - they promise great potential, and until experience proves otherwise, risk. The questions posed in Table 9 therefore measure how well the microprocessor and its available development aids match the control task and experience of the control project team.

Answering these questions in the Table and assigning suitable weighting factors against each aspect, the index can be calculated for each microprocessor, resulting in the histogram shown on Figure 9.

Note that this index must reflect the experience of the company making the assessment and the complexity of control task for which the company is selecting the microprocessor. Naturally for this exercise I have assumed Lucas' actual experience, see Table 2. I also assumed that the application is of moderate complexity.

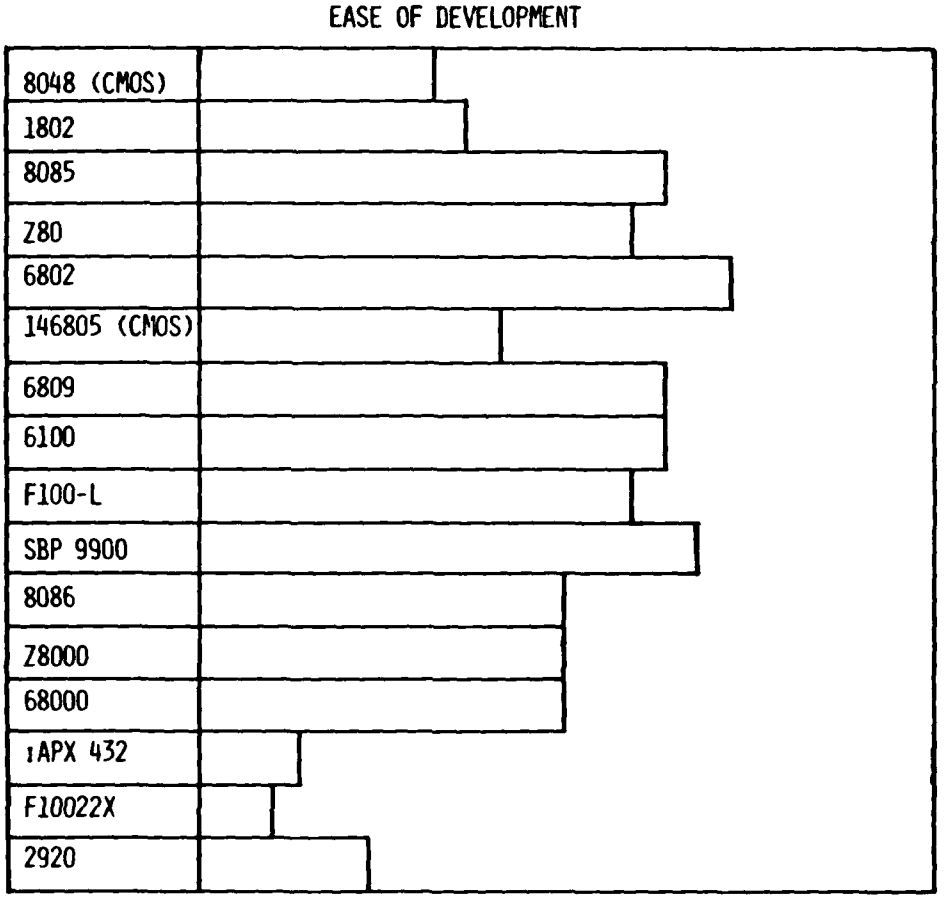


Figure 9 - Histogram of Ease of Development

One trend emerged from the assessment - the more powerful 8 bit devices are extremely well supported having a wide choice of high level languages, hardware and software development systems. No doubt the new 16 bit machines from Intel, Zilog and Motorola will achieve similar support in due course. The two bipolar 16 bit machine manufacturers supply software support, high level language and hardware development aids, but support from other sources is isolated.

This completes the three indices, which can now be combined to give the final figure of merit.

7. FIGURE OF MERIT FOR SUITABILITY

To complete this comparison of the 15 microprocessors selected for Table 3. the indices for Function, Procurement Status and Ease of Development are multiplied together as described in Table 6 to produce the Figure of Merit for Suitability. Two of the indices included some questions that assess the ability of the powerplant control manufacturer to handle each microprocessor. For a given microprocessor, will the control manufacturer be able to use existing automatic test equipment, does his design team have any experience of the microprocessor, do they have to correct development aids it will need? Another question is how does each microprocessor's capability match the needs of the particular control application. Therefore bear in mind that the final figure of merit must assess suitability for a particular complexity of control and assume the experience and resources of the control manufacturer for whom the assessment is being made. So combining the indices from Figures 7, 8 and 9 produces the histogram for Suitability shown on Figure 10.

Half of the original 15 microprocessors appear 'suitable', three or four are marginal and three appear unsuitable. This is quite a change from the functional index on Figure 7 where virtually all 15 scored well. This reflects the relatively poor scores that some of the microprocessors achieved for ease of development and/or their procurement status.

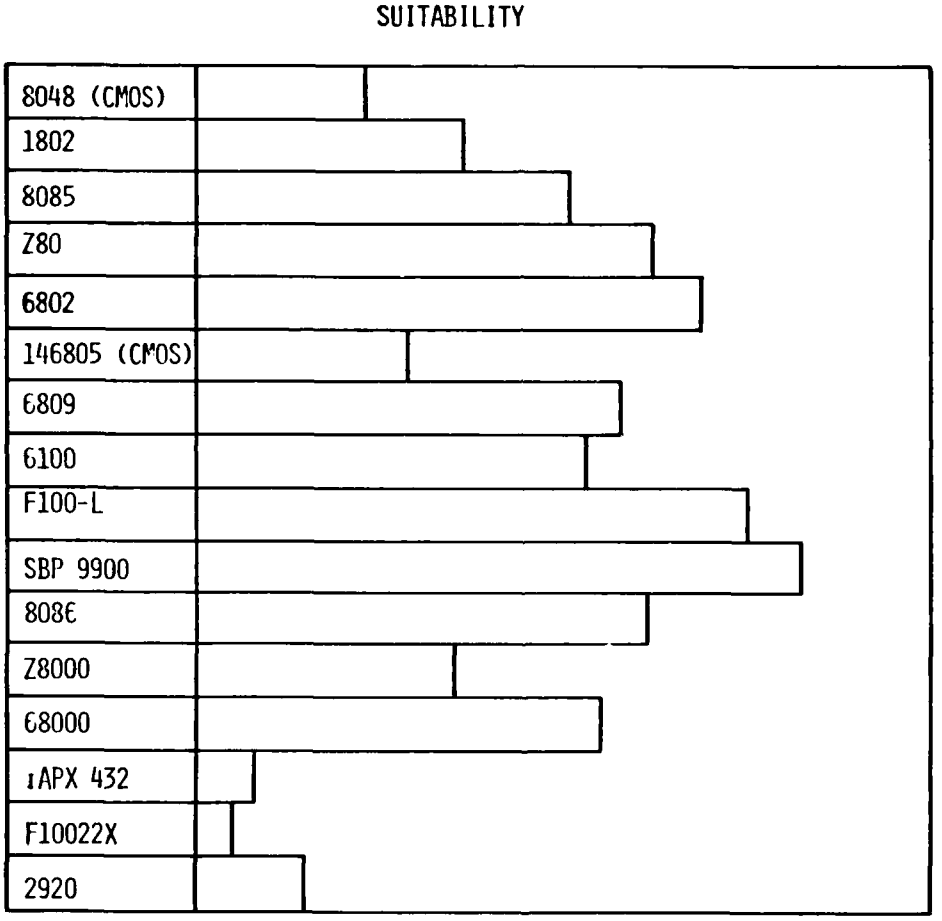


Figure 10 - Figure of Merit for Suitability

Please recognise that the above selection method is still very coarse: it is at the next stage in the real selection when the hard work begins and our depth of experience as a control manufacturer becomes an overriding factor. The customer's specification must be interpreted into a detailed system design that will define the microprocessor related tasks and their requirements. At this point will emerge the system architecture, such as the need for single or multiple microprocessors for reliability or safety considerations, or very low electrical power consumption for some thermal design or power supply constraint. The control tasks must be analysed in detail to determine the algorithms needed to meet the accuracy, bandwidth and overshoot specification. The safety and BITE tasks must be analysed so that the full implications of safety interlocks, fault detection and resulting actions are understood. These analyses define the detailed requirement of the microprocessor - against which the 'suitable' microprocessors can be bench-marked, assessed for all impact on function, size, cost, reliability, ease of development, certification, maintainability, etc. of production control unit.

The selection method for 'suitability' is also quickly overtaken by events - the assessment has used 1980 data and taken into account the control manufacturer's experience up to 1980. Change the date to 1982, or the application, or the control manufacturer, and the whole histogram will be transformed. So I will stop at this point and offer you the following challenge: Devise your own indices, weighting factors, and formulae for a figure of merit - how well do the wide variety of microprocessors meet your needs? After all, the choice is yours, and you alone will be accountable for it.

ACKNOWLEDGMENTS

The author is indebted to the Directors of Lucas Aerospace Limited for permission to publish this paper. Equally it is a pleasure to acknowledge the help and support of my colleagues at Lucas Aerospace Limited, Engine Electronics, York Road, Birmingham, at Lucas Industries Group Research Centre, Solihull, England, and at Lucas North America, Aerospace Division, Englewood, New Jersey, U.S.A.

THE PACKAGING OF ELECTRONIC ENGINE CONTROL UNITS AND RELATED SUBCOMPONENTS

by
R. W. Vizzini
Program Manager, Advanced Controls
Naval Air Propulsion Center
Advanced Development Division
P. O. Box 7176
Trenton, New Jersey 08628
USA

SUMMARY

SUBCOMPONENT PACKAGING

Pratt and Whitney Aircraft (P&WA)/Hamilton Standard Division (HSD), Divisions of United Technologies and General Electric Company (GE) represent two significantly different on-engine full authority digital electronic control (FADEC) subcomponent packaging technologies. P&WA/HSD are generally in concert with the majority of the engine control industry that utilize single and/or multi-layer printed circuit boards (PCB) in conjunction with dual-in-line (DIP) chip packaging techniques. The structural integrity of this type of technology has been verified by current usage in aircraft (A/C) systems and is the present state-of-the-art for on-engine subcomponent packaging. Whereas the GE on-engine FADEC technology is multi-layer ceramic modules (MCM) which is an advanced concept and a unique approach to propulsion control design. The inherent benefits of the MCM design are increased reliability, decreased component weight and volume. The MCM approach on-engine control unit design is the advanced state-of-the-art and may be the way to the future for control unit subcomponents.

Both P&WA and GE FADEC subcomponent technologies presented are currently under Navy contracts.

ENGINE CONTROL LOCATION EVALUATIONS

In the mid-seventies two significant studies were performed to identify the most suitable location for the engine control unit, on or off-engine. One study was conducted in-house by McDonnell Douglas Corporation (MCAIR) for a current U. S. twin-engine fighter aircraft (A/C). The second study was performed under Navy contract by P&WA and addressed the control locations of advanced lift-cruise engines for application on a vertical/short take-off or landing A/C (V/STOL). Each study evaluated the benefits of current control units and the full authority digital electronic control approach. Both studies recommended full authority digital electronic controls on next generation propulsion systems.

Trade studies were initiated as the evaluation criteria to identify the advantages and disadvantages of the on/off control locations. These studies were based upon maintainability, cooling, reliability, vulnerability, safety, weight and life cycle costs (LCC). LCC was used as the final standard of selection, combining the results of all the preceeding criteria with initial acquisition cost.

Both studies have indicated that LCC, engine replacement time and weight penalties are associated with off-engine control units. Also, the vulnerability of the propulsion system is increased. For example, if multiple engine controls were located in the A/C bay area, a single battle damage occurrence at that location would result in the total loss of aircraft propulsion. Furthermore, a failure in the A/C supplied electrical power would adversely effect the reliability of control unit resulting in mission abort or aircraft loss. However, with the control unit mounted on the engine, electrical power is generated by individual, dedicated engine driven alternators,which are completely separate from the aircraft electrical system, thereby, increasing propulsion system reliability for single as well as multi-engine aircraft configurations.

If the environmental control system (ECS) were to be used for control cooling, the liquid cooling system has the weight advantage over the air cooling system as indicated by the V/STOL study. However, a comparison of four coolant configurations studies by MCAIR; fuel from the aircraft boost pump, ECS cooling air and a liquid coolant system indicated that ECS air cooling is generally less complex and needs no special controls or valving and is therefore easier to maintain. The maintainability of the cooling system selected is dictated by the control installation location. It should be noted that if the A/C cooling system fails, so does the control. Presently, on-engine controls use fuel cooling from the A/C fuel tank for supersonic flight and engine supplied cooling air for subsonic applications. Both approaches avoid dependence upon the A/C environmental control system.

The off-engine control locations, i.e. aircraft bay, with or without pressure transducers is considered undesirable. If the pressure transducers are mounted on the engine, packaging and cooling of each transducer and a multiplexer circuit would be required to communicate with the bay-mounted control. The multiplexer system would require the same environmental design criteria as the on-engine control. When the bay-mounted control is integral with the pressure transducers, excessive pressure lag time is evident. Other penalties associated with the off-engine control locations are excessive cost, weight, engine replacement time and increased control system vulnerability.

Of the four control locations studies, A/C bay, boom, nacelle and the engine, the nacelle location indicated a significant improvement over the A/C bay and the boom, with only a slight advantage over the engine mounted unit in line replacement time. This counteracted by an increase in engine change time and the required modifications to aerospace ground equipment (AGE). Engine change time is increased because the control must be removed from the nacelle before engine removal proceedures are initiated. The boom location has the highest height and since the control is remote from the engine, increased maintenance man hours per flight hour are required for an engine change. The shortest access and replacement times is in the forward bay and is comparable to the engine mounted unit. However, this slight advantage is negated

because of the significant increase in engine change time and the anticipated increase in trouble shooting time due to the remote location of the control.

A weight analysis was performed for each of the four control locations. This analysis clearly showed the on-engine control to be significantly lighter. The nacelle, forward bay and the boom locations have a weight increase between 32 and 68.5 kilograms.

The LCC system for evaluation is composed of three main categories, non-recurring cost and recurring cost. Data analysis for all control locations sutdies indicated an insignificant difference in non-recurring cost. However, the on-engine control location showed a significant reduction in recurring cost and acquisition cost. Logistic support cost analysis indicated that the on-engine control configuration has the lowest cost per flight hour when compared to the nacelle boom and bay locations.

The on-engine control configuration was selected based upon LCC and by combining the results of the evaluation criteria (trade studies) with acquisition cost. The engine mounted control provides the complete propulsion system with a fully assembled and tested control system prior to A/C installation providing fault isolation and accountability with the engine manufacturer, thus avoiding potential areas of conflict between the airframer and the engine manufacturer.

NOMENCLATURE

SYMBOL	=	DEFINITION
A/C	=	Aircraft
AIC	=	Air inlet control
AIT	=	Auto-ignition temperature
AGE	=	Aerospace ground equipment
ALT	=	Altitude
B/M	=	Bill of material
^{o}C	=	Degrees centigrade
CLB	=	Closed loop bench
CM	=	Centimeters
D,Δ,CHG	=	Change
DIP	=	Dual-in-line package
ECS	=	Environmental control system
EEC	=	Electronic engine control
EMI	=	Electromagnetic interference
FADEC	=	Full authority digital control
FH	=	Flight hours
Fus	=	Fuselage
FWD	=	Forward
GA	=	Gauge
GE	=	General Electric Company
GG	=	Gas generator
Hr	=	Hours
HSD	=	Hamilton Standard Division
Hz	=	Hertz, cycles per second
IC	=	Integrated circuit
I/O	=	Input/output
J-Box	=	Junction box
JTDE	=	Joint technology demonstrator engine
K	=	10
Kg	=	Kilograms
L/C	=	Lift cruise
LRU	=	Line replaceable unit
LSC	=	Logistic support cost
LSI	=	Large scale integration
M	=	Meters
MAX	=	Maximum
MCM	=	Multi-layer ceramic module
MCAIR	=	McDonnell-Doublas Corporation
MIN	=	Minimum
MISC	=	Miscellaneous
MMH	=	Maintenance manhours
MSI	=	Medium scale integration
MTBF	=	Mean time before failure
MTD	=	Mounted
PCB	=	Printed circuit board
Ph	=	Probability of a hit
Pk	=	Probability of a kill
P&WA	=	Pratt and Whitney Aircraft
QD	=	Quick disconnect
RAM	=	Random access memory
RDTE	=	Research Development Technology and Evaluation
RWR	=	Radar warning receiver
SDRS	=	Signal data recording set
SLS	=	Sea level static
SSI	=	Small scale integration
S/V	=	Survivability/vulnerability
Va	=	Vulnerable area

SYMBOL	=	DEFINITION
VLSI	=	Very large scale integration
Xducer	=	Transducer
λC	=	A portion of the total failure rate which is safety critical λ = failure rate, C = critical
%	=	Percent

INTRODUCTION

This paper is divided into two parts, Electronic Packaging of the Subcomponents, and the Location of the Engine Control Unit (on or off-engine).

THE TRANSITION OF SEMI-CONDUCTOR TECHNOLOGY AND PACKAGING

Semi-conductor technology and packaging has transitioned from discrete parts to integrated circuits (IC) of small scale integration (SSI), medium scale integration (MSI), large scale integration (LSI) and of very large scale integration (VLSI). A few decades ago, a single transistor or logic function occupied one package or chip device; presently, hundreds (MSI), thousands (LSI) and tens of thousands (VLSI) transistors can be fabricated in a single semi-conductor chip. The selection of the circuit logic family of the semi-conductor varies with the application. For example, the desired characteristics in microprocessors are fast response, low power dissipation, high circuit density, cost and availability of the chip are factored into trade-offs for a given application. This paper will confine itself to the packaging of the electronics rather than the drivers associated with IC selection.

ELECTRONIC CONTROL SUB-COMPONENT PACKAGING

The U. S. Navy has sponsored two advanced on-engine full authority digital electronic control (FADEC) with technical direction provided by the Naval Air Propulsion Center, Trenton, New Jersey, USA. One with General Electric (GE), (N00019-76-C-0423) and the second with Pratt and Whitney Aircraft (P&WA), (N00019-76-C-0422).

The packaging of integrated circuits and discrete parts have a significant impact on maintainability, reliability, volume, weight and cost. The Navy FADEC programs, by design, consist of two entirely different electronic packaging concepts. GE and P&WA have successfully demonstrated through test and evaluation the viability of each advanced design.

ELECTRONIC CONTROL UNIT PACKAGING

The progression of electronic technology has led engine control systems from purely hydromechanical to electronic supervisory with hydromechanical to full authority digital electronic control, dual or single channel, depending upon the desired application. Traditionally, engine controls have been considered as part of the power plant despite the harsh environment. For the past decade, the location of the engine control unit(s) has been a somewhat emotional issue, and as a result, various government agencies, engine manufacturers and airframe manufacturers performed in-house trade studies with few of the results being released for publication. Therefore, the major thrust of this paper is to present quantitative assessment of the packaging of a conceptual Full Authority Digital Electronic Control unit for high performance aircraft systems. The data and information compiled in this report is the result of a survey of major engine manufacturers, airframe manufacturers and government agencies and will address maintainability, reliability, vulnerability, safety, weight, cooling, logistics and life cycle cost for a control location on the engine and several locations within the airframe.

SUBCOMPONENT CONTROL DESCRIPTIONS

The FADEC electronics described herein were designed, developed, bench and engine tested as on-engine controls consistent with military specifications (MIL-E-5007D) and standards (MIL-STD-810C and 461A). There are basically two types of current electronic pagkaging techniques:

- Multi-layer ceramic modules (MCM) - General Electric Co. (GE)
- Multi-layer polyimide boards using dual-in-line packaging (DIP) - Pratt and Whitney Aircraft (P&WA)/Hamilton Standard Division (HSD)

The objectives and approach to the Navy GE and P&WA FADEC programs were similar; only the mechanical packaging design and fabrication of the modules differed significantly. Both technologies are representative of the advanced state-of-the-art in the packaging and/or control system design. This section briefly describes the highlights of the electronic units associated with each FADEC program with emphasis on the multi-layer ceramic module technology developed by GE; since this is the most unique electronic control technology developed for on-engine application.

General Electric - The multi-layer ceramic module (MCM) is composed of six co-fired layers of laminated alumina (Al_2O_3). Each layer consists of tungsten metallized vias (holes) and electrical conducting patterns which overlap the vias, providing electrical continuity through the module. The vias and the electrical circuit patterns on the topology of the module are gold plated over tungsten. The chips are conventionally attached to gold plated bonding pads using a gold/germanium braze. Tape-automated bonding (TAB) is also used as a means for attaching the chips/dies to conductors on the surface of the module.

Figure 1 is a cross-section of the module and shows the vias, the six layers of alumina, the die and the die attachment to the surface of the module. Approximately 200 IC's and related capacitor and resistor circuit elements can be mounted in the active area. The active area of the module is surrounded

by a Kovar ring and is hermetically sealed with a Kovar cover plate which is bolted to the ring. Electrical connections are accomplished via three rows of gold-flashed Kovar pins located on the periphery of each side of the module. Figure 2 is a photograph of the microprocessor module and identifies the microprocessor chips, programmable read-only-memory (PROM) chips, random access memory (RAM) chips, capacitors and resistors.

These Proto-thpe modules contain analog and digital engine control circuits which are partitioned int nine modules and are tabulated below (Note the small size of the modules).

Type	Number Required	Function	Overall Size - cm
MCM	3	Program Memory	7.5 x 5.5 x 1.0
MCM	1	Processor	7.5 x 11.5 x 1.0
MCM	1	Analog Circuits	7.5 x 11.5 x 1.0
MCM	3	Input/Output	7.5 x 11.5 x 1.0
MCM	1	Interface to Aircraft	7.5 x 11.5 x 1.0

Inter-connections of the modules are achieved by means of a wiring harness with cryogenic sockets or conventional soldering techniques which mate to the terminal pins. Harness sockets can be grouped into the connectors to provide quick disconnect capability at the module level. Figure 3 compares to the module with a conventional wire-wrap printed circuit board (PCB) of the expoxy/melamine variety with the standard DIP technology. Note the relative size. The significant reduction in size and weight is accomplished by the higher-density chip packaging which results in the reduction of module size. Approximately 80% reduction in volume is realized with the modules mounted against heat sink plates. When compared to the production state-of-the-art, there is a significant decrease in unit weight as indicated below:

	Volume	Weight
Production state-of-the-art	37.6×10^3 cm^3	54.4 Kg
FADEC (MCM)	7.3×10^3 cm^3	11.2 Kg
Net reduction	30.3×10^3 cm^3	43.2 Kg

The MCM approach provides improved reliability in comparison with conventional electronic packaging techniques.

The selected materials have low, well matched, coefficients of thermal expansion to avoid thermally induced differential expansion stress:

Alumina MCM:	4.9×10^{-6}cm/cm ^oC
Silicon chip:	3.1×10^{-6}cm/cm ^oC
Kovar cover:	5.0×10^{-6}cm/cm ^oC
Tungsten circuits:	4.7×10^{-6}cm/cm ^oC

Commonly used high expansion electrical materials such as copper are avoided. The MCM approach enhances reliability through reduction of series connections in comparison with approaches which mount the silicon integrated circuits on individual alumina modules and in turn interconnect these in multilayer boards. Figure 4 shows a photograph of the assembled proto-type unit indicating vibration isolator, adjustment potentiometers, clock, pressure transducer, power supply, electromagnetic interference (EMI) filter and transformer locations.

There are three FADEC on-engine units in this program. Each unit is dedicated to a specific engine for test and evaluation. These are as follows:

- Advanced variable cycle demonstrator engine, sea level static (SLS) test - completed October 1980, S/N 001
- Current U. S. Navy engine, SLS test scheduled for April 1980 and an altitude test scheduled for July 1981, FADEC S/N 003
- Advanced Joint Technology Demonstrator Engine (JTDE), SLS test scheduled for June 1981 and altitude testing in 1981-1982, FADEC S/N 003

Previous to each engine test, a series of Design Assurance Tests are conducted to validate the software and the structural integrity of the system. These tests are as follows: open loop, extreme temperature (-55 C to 125 C), vibration (2-g scans, 10-1350 Hz) and system integration. FADEC, S/N 001 has successfully completed the pre-engine test requirements and is currently undergoing a series of severe environmental tests (military specification and standard requirements) designed to expose the control unit to a "real world" engine environment. Briefly, these tests include: open loop tests, accelerated aging, temperature cycling, EMI vibration, impact and explosion proof. A post-test calibration is conducted to identify component malfunction.

The three FADEC units will undergo over 1600 hours of testing which include design assurance, environmental and engine tests as shown in Table 1. The control technology developed from these Navy programs (GE and P&WA) is currently being transitioned into future commercial and military FADEC systems.

Pratt and Whitney Aircraft/Hamilton Standard Division - The intent of this FADEC program was to ultimately engine mount a dual-redundant FADEC system connected via fiber optics on a current Navy fighter. This configuration would demonstrate the electronic control back-up approach providing full engine operational control capability and integrate aircraft control systems. However, for development purposes, an on-engine unit communicated with a control room unit (breadboard) in real time via a serial optic data link during all engine tests. The technical effort of this program was completed in August 1979. For detailed information, see references 1 and 2. This discussion will confine itself to the packaging of the control unit.

The basis subcomponent packaging technology developed by P&WA/HSD is representative of the majority of the control industry which utilize single/multi-layer printed circuit board (PCB) and dual-in-line (DIP) packaging techniques. The geometry and the packaging technique may differ from other manufacturers but the technology is similar.

UNIT DESCRIPTION

The FADEC engine-mounted package is comprised of three components: a main housing, a cover, and a pressure sensor manifold, as shown in Figure 5. The parts were machined from aluminum alloy. Provisions for dowel pins are incorporated in the mating surfaces of the main housing and cover to control relative positioning during assembly. The cover has mounting surfaces for the four vibration isolator mounts. A handle is provided for transporting the assembled unit. Also shown in Figure 5 are ports for a tube for transfer of cooling fluid between the main housing and the cover halves of the control. Similar features of both the main housing and cover are the outboard and inboard printed-circuit board (PCB) mounting surfaces. The outboard PCB's are cooled by coolant flowing through the housing heat exchanger mounting surfaces. The inboard PCB's are cooled by conduction to the mounting lugs and then to the frame heat exchanger. Figure 6 shows the main housing assembled with the cover, but without PCB's or other internal components. This view of the housing shows the cooling-flow supply and return port bosses, the optic receiver and transmitter connector ports, and the four electrical connectors. The port in the center of the main housing is for installation of the pressure control valve which regulates internal control assembly pressure with varying ambient pressures. Also shown is the main housing pin-fin heat exchanger, which provides air cooling via convection and radiation to the surrounding environment. This is in addition to internal fuel cooling. The cover also contains a similar pin-fin heat exchanger which is not shown.

The unit contains four major multi-layer polyimide PCB's (each 35 x 21 cm), a small memory expansion board (14 x 10 cm), pressure transducers (3) and ribbon connectors, as shown in Figure 7.

The Navy requirements of subcomponent, component and engine testing of GE and the P&WA FADEC programs were similar since the contractural requirements were identical. Table II summarizes all tests and hours accumulated in the P&WA program. Over 7000 hours of testing were recorded which included development, design assurance, environmental and engine testing.

Planned follow-on programs consists of the following:

- Combined environmental reliability test - verification of Phase I 8000 hours MTBF, 1981.
- Aircraft control integration study - 1981
- Navy fighter flight demonstration - demonstration of the on-engine dual-redundant FADEC configuration and aircraft control integration.

QUANTITATIVE ASSESSMENTS OF ENGINE CONTROL LOCATIONS

In the mid-seventies two significant studies were performed to identify the most suitable location for the engine controller. One study was performed in-house by McDonnell-Douglas Corporation (MCAIR) for engine control location on a current high performance U. S. twin-engine fighter. The second study was performed under Navy contract (N00019-72-C-0612)(reference (3)) and addressed the control locations of advanced lift cruise engines for application on a vertical/short take-off or landing aircraft (V/STOL). The results of both these studies are presented herein.

V/STOL ENGINE CONTROL LOCATION STUDY

BACKGROUND

As part of a Navy/P&WA Program (July '73 to June '75) to identify the attitude control/engine control system requirements of a V/STOL aircraft (reference (3)); one of the major objectives was to establish a control system configuration considering fluidic, hydromechanical and an electronic approach and identify the most suitable location of control components for V/STOL aircraft implementation. This discussion deals only with the assessment of electronic control packaging. However, it was concluded that a full authority digital electronic control possessed a significant advantage over alternative technologies for controlling complex advanced turbine engines and that projected advancements in electronic technology will continue to improve its ability to handle increasingly complex computational tasks in the future. Therefore, the full authority digital electronic control (FADEC) system was considered a clear choice for the V/STOL control system.

CONTROL CONFIGURATIONS AND TRADE STUDY GROUND RULES

The lift-cruise engine control system configuration definition considered two locations for the digital electronic control. In the first configuration the control is mounted on the engine, and in the second configuration it is mounted in the electronic equipment bay. See Figure 8. A mounting location in the nacelle was not considered since, except for less severe vibration, this location was not considered significantly different from the engine-mounted location.

Calculations were made to determine the impact of cost, weight, reliability, and maintainability on the aircraft for the engine-mounted and bay-mounted control systems. Only the components, tubing, and cabling which differed for the two control locations needed to be assessed. Data sources for this evaluation included previous in-house P&WA studies conducted with control vendors and airframe manufacturers and previous P&WA experience with control system components in flight applications.

The following ground rules were established for various control system components:

- All sensors and effectors are engine mounted.
- For the bay-mounted control, pressure transducers are integral with the control unless the pressure

sensing line length exceeds 15 feet, in which case the transducers must be engine mounted in a separate module.
- Both liquid and air cooling from the aircraft ECS is considered for both engine and bay-mounted control systems.
- Aircraft power is used for the bay-mounted control and an engine-driven alternator is used for the engine-mounted control.
- Back-up control is accomplished with completely redundant digital electronic control computers, redundant transducers, sensors, torque motors, feedbacks, alternators, and fuel-metering valves.

WEIGHT

Control system weight, assuming current technology components, cables and tubing for the lift-cruise engine control system, was calculated as a function of the straight-line distance between the aircraft bay electrical connectors and the engine nacelle bulkhead location for both the environmental control system (ECS) air-cooling and liquid-cooling of the control unit. See Figure 9. This figure shows an increase in control system weight proportional to the cable length for the bay location and the various on-engine locations evaluated. Note that bay mounted control required longer cable runs when compared to the on-engine locations resulting in increased weight and control system vulnerability. Table III shows the changes required for the bay-mounted control which resulted in step increases in weight and cost. A summary for the weight differences for each engine location (1, 2 and 3) further substantiates the increased weight and cost associated with off-engine control location, see Table III. Figure 9 also indicates that the ECS liquid-cooled system has a greater relative weight advantage than the air-cooled system.

COST

Estimated costs, assuming current-technology cables, tubing, and control system components were calculated. A comparison of the engine-mounted and bay-mounted control costs is presented in Figure 10. With the pressure transducers integral with the bay-mounted locations, there is very little cost difference between the engine and bay-mounted locations. The cost of the bay-mounted control configuration increases significantly when it becomes necessary to engine-mount the pressure transducers. This cost increase results primarily from the requirement to package and cool each transducer separately and to provide multiplexer circuitry to communicate with the bay-mounted electronic control. The slopes of the cost curves (Figure 10) show the engine-mounted control would be more cost-effective than the bay-mounted control.

From an acquisition cost standpoint, the engine-mounted configuration should be selected as the best electronic control location for the V/STOL lift-cruise engine locations considered. Acquisition cost differentials for the two control locations are summarized in Table IV for lift-cruise engine locations 1, 2, and 3.

RELIABILITY

Reliability factors affecting the comparative control locations are as follows:

(1) Reliability of the cooling source for the electronic control and transducers.
(2) Reliability of the power source for the electronic control.
(3) Reliability of the harness between the engine and the electronic control.
(4) Reliability of the electronic control and pressure transducers.

The first three factors for both locations were determined insignificant compared to the electronic control and pressure transducer configuration. This conclusion was based upon previous control vendor studies and on reference 4. The bay-mounted control unit with bay-mounted pressure transducers would experience increased reliability only because of a more conducive vibration and temperature environment. However, engine pressure signal lag time, cost, weight, vulnerability and reliability penalties are associated with bay-mounted control unit with engine mounted pressure transducers is 1.3 times that of the on-engine control unit. The numerous parts required to separately package the pressure transducers and provide the required additional circuitry with an on-engine multiplexer for communication with the bay-mounted control discount the bay-mounted control environment advantage. Therefore, it was concluded that the on-engine control locations indicated an increase in reliability when compared to the bay-mounted location. Detailed information is presented in reference 3.

MAINTAINABILITY

Organizational level (flight line) maintenance costs which vary with the control system configuration include replacement time of components and engine change time. Component replacement time has a small effect, relative to engine change time on maintenance manhours per flight hour (Table V). The data in this table indicates that the engine-mounted control system has an a[illegible]ge over the bay-mounted systems. This results primarily from the lower number of tubes and electrical [illegible]ectors which have to be disconnected when removing the engine with an engine-mounted control.

LIFE-CYCLE COST

The results of life cycle cost (LCC) studies (reference 5) have shown that the effects of overall engine maintenance costs on LCC are small compared to the effects of acquisition costs and aircraft weight. The magnitude of the possible LCC savings for the engine-mounted system, based on cost and weight only, over the life of a fleet of these V/STOL aircraft, was evaluated using LCC trade factors for similar fighter aircraft. The results are summarized in Table VI for lift-cruise engine locations 1, 2, and 3. The data in the table, incremental life cycle cost, is equal to the incremental weight times the LCC weight trade factor plus the incremental cost times the LCC acquisition cost trade factor. Significant

life cycle cost savings can be achieved with the engine-mounted control system, relative to the bay-mounted system, regardless of engine location and type of cooling.

It is concluded from the above V/STOL trade studies performed by P&WA that the electronic engine control unit(s) should be mounted on the engine regardless of the bay-mounted control unit configuration.

A HIGH PERFORMANCE AIRCRAFT ENGINE CONTROL LOCATION STUDY

BACKGROUND

Early in 1974, a review of a current engine control system was conducted by the Air Force to determine if an engine electronic control designed to execute all the hydromechanical/electronic supervisory functions would increase control system reliability. Preliminary results indicated that improvements could be realized if the hydromechanical fuel control (HFC) were reduced to a servo/actuation (metering valve) system and full authority electronics were used as the engine controller. Thus, simplifying the control system and increasing overall reliability.

As a result of this review, MCAIR in cooperation with P&WA established the redesign of the base line system, unified HFC and the supervisory electronic engine control (EEC), to a Digital Full Authority Electronic Control (FAEC). Trade studies were conducted using the conceptual design of the FAEC which were structured around a high performance twin-engine fighter aircraft. These studies not only compared the baseline control configuration to the FAEC system but also evaluated four mounting locations FAEC unit: three within the airframe and the other on the engine. The FAEC installation trade studies were based on the evaluation criteria discussed herein.

CONTROL LOCATIONS EVALUATED

Figure 11 shows the four electronic control locations as packaged on the study aircraft. Traditionally, control systems are mounted on the engine and are designed to tolerate the environmental extremes as specified by military standards and specifications. The nacelle location encounters less vibration and temperature extremes and when located in the boom or the forward bay areas the control is effectively isolated from all engine environmental extremes. However, weight and cost penalties are significant. In order to evaluate the locations (Figure 11) trade studies were initiated as the evaluation criteria to identify the benefits of each location based upon the following: maintainability, cooling, reliability, vulnerability, safety, weight and life-cycle cost.

Maintainability includes changes to the aerospace ground equipment (AGE) along with increases or decreases in maintenance man-hours due to the different access and Line Replaceable Unit (LRU) rep acement times. The effect on reliability, vulnerability and safety are evaluated by comparing each control configuration with the baseline system. The changes in aircraft weight include that of the electrical cables, cooling plumbing, aircraft structure and the necessary ballast to balance the aircraft. The different cooling systems were also investigated for their effect on maintainability, reliability, survivability, vulnerability, safety and weight for each of the candidate mounting locations. Life cycle cost is used as the final standard of selection, combining the results from the preceding criteria along with initial acquisition cost to attain a final value of each system.

COOLING SYSTEM

Three different cooling systems were initially analyzed to determine which is the most suitable for maintaining the Full Authority Electronic Control (FAEC) within acceptable temperature limits as shown in Figure 12. The baseline system uses fuel downstream of the main pump for EEC cooling, but this is unacceptable for the FAEC electronic control since stricter temperature limits are specified for this microprocessor. The systems investigated were fuel from the aircraft boost pump, cooled air from the EEC bay and a special coolant system.

A qualitative comparison of the different coolant systems (Table VII) shows that air cooling has a safety advantage over both the fuel and liquid systems. In reliability, each of the three systems has its advantages and disadvantages with no one system decidedly better than the others. Air cooling is generally less complex, needs no special controls or valving and is therefore somewhat easier to maintain. The actual maintainability of the coolant system though, is dependent on control installation location.

Table VII summarizes fuel, air and liquid coolants with respect to control locations quantitatively.

MAINTAINABILITY

The study engine with the unified HFC and the EEC was used as the baseline system throughout the study. Access time to the control is twenty minutes while exactly twice that amount is needed to remove and then replace the unit. The baseline EEC has no impact on engine removal time when the unit is engine mounted as it contributes no additional off engine connection points. The time required to remove the engine is used as the standard to which the other control configurations will be compared.

Changes to the engine aerospace ground equipment (AGE) can have a considerable impact on aircraft maintenance costs. In this trade study, it is assumed similar AGE will be used in trimming the engine and the required time for trimming the engine will be independent of the control location. This study covered more than just control installation location and changes in the existing AGE are included in the life cycle costs.

The engine mounted FAEC and the baseline EEC have the same access time since both are situated in essentially the same location. Replacement time for the FAEC control is considerably less due to the use of MIL-STD-38999 quick disconnect connectors. The fuel cooled version of the FAEC has its own source of cold tank fuel and requires an additional engine/airframe interface. The additional cooling interface increases engine change time, increasing life cycle cost. Plumbing connections for the ECS cooled version

of the FAEC are generally simpler and faster to connect/disconnect than on the fuel cooled versions. Faster unit replacement and engine change times reflect this difference in hardware.

The nacelle mounted FAEC has the same access time as the baseline control and shows a slight improvement over the engine mounted units in LRU replacement time. This is counteracted by an increase in engine change time and by the requirement that some of the aerospace ground equipment (AGE) will have to be modified. Engine change time is increased because the FAEC must either be disconnected from the engine or removed from the nacelle wall before the engine is pulled. The installation rail of the present AGE interferes with allowable unit nacelle mounting locations and will have to be changed to fit the unit in the study aircraft.

When the electronic control is mounted in the boom area of the study aircraft there is considerable impact on system maintainability. The boom area is high off the ground making servicing of the control much more difficult. Access time is over five times that of the baseline and LPU replacement time is the longest of four locations studied. While replacement time for the boom mounted FAEC is half that of the baseline EEC, this is due to the use of quick disconnect (QD) connectors rather than control unit location. Additional work stands are required to service the control and, since the control is remote from the engine, the most maintenance manhours per engine flight hour are required for an engine change.

The shortest access and replacement times result when the FAEC control is installed in the forward bay. In this configuration the pressure transducers must be located remote from the control and mounted on the engine to keep the pressure signal time lag from being excessive. The access and replacement time for these pressure transducers is comparable to an engine mounted FAEC. However, this is negated due to the considerable increase in the engine change time and the increased trouble shooting time required for this remote location of the control.

If aircraft fuel is supplied to the FAEC for cooling, four extra valves and some necessary plumbing will have to be added to the system. This will result in an increase of .0035 maintenance manhours per engine flight hour over the current cooling system which uses the relatively hot pump discharge fuel. Using the existing environmental control system for the cooling of the FAEC unit will have little impact on the maintenance costs of the aircraft.

The comparison of the impact on system maintainability by the different control configurations presented in Table IX shows that the engine mounted, air cooled FAEC requires the smallest increase in engine change time, while the forward bay mounted unit has the shortest access and replacement times. Even though some of the systems have maintainability advantages over the others, they are all on about equal footing with the exception of the boom mounted FAEC. This configuration has a distinct maintenance problem in its poor accessability and large increase in engine change time.

RELIABILITY

The reliability analysis considers changes in both engine and airframe configurations. The baseline engine configuration, wiring and actuation systems are considered along with the airframe's Air Inlet Control, additional instrumentation, cooling and wiring. Some of the configuration changes are due to the control installation location (such as the installing of a remote trim if the control is placed in the test boom) but most are brought about by use of the FAEC system instead of the Bill-of-Material (B/M) control.

The EEC failure rates are based on operating in continuous maximum Mach Number environment, but because of the cooling and isolation of the units, reliability is independent of mounting location. EEC reliability is dependent on the method of cooling and on configuration changes that alter the unit's part count.

A summary of the failure rates per million operating hours for the control system components in the various mount locations is presented in Table X. The bottom value is the MTBF for each configuration. Note that MTBF is calculated as the reciprocal of the sum of the failures per million hours multiplied by 10^6.

Note that all of the FAEC systems are more reliable than the current system. The engine control system reliability improvement may be traced to a reduction in failures of both the electronic and hydromechanical portions of the system. The electronic control reliability has increased by a factor of approximately three due mainly to the increased level of integration in the FAEC system. The elimination of the unified fuel control (UFC) and it's replacement with a simpler gas generator (GG) and augmentor control is the basis of the hydromechanical improvement. It should be remembered that, for the FAEC mounted in the forward bay, engine pressure transducers must remain on the engine to keep pneumatic line lengths to a minimum. This accounts for the double entry for the EEC in this configuration.

The baseline system as shown in Table X has a lower overall aircraft MTBF as compared to the four FAEC locations being evaluated. However, this is not significant since the study engine system cannot be effectively integrated with the aircraft control systems; whereas the FAEC system has that capability designed within the configuration with an overall MTBF improvement between 26 and 32% for the mount locations evaluated.

VULNERABILITY, SURVIVABILITY AND SAFETY

The topics of Vulnerability/Survivability and Safety have been defined as follows for this study:

- Vulnerability is the likelihood that the FAEC control system will sustain a hit from an incoming projectile.
- Survivability is that given a hit from a projectile, the aircraft either continues to operate, continues the mission or suffers the loss of a subsystem or component.
- Safety is the effect of the FAEC system on the aircraft accident rate.

The survivability/vulnerability (S/V) and safety analyses for the FAEC presentation are divided into

two sections in this material. The S/V aspects are addressed first with the following listing of the major factors considered in the analysis. This analysis considers the impact of increasing or decreasing component areas and the effect of a hit in the vulnerable area. The vulnerability of a given area (VA) is the probability of a hit in that area (PA) times the probability that the hit will result in loss of the aircraft (PK).

The S/V analysis ground rules for the use of the three coolant mediums, fuel, liquid and air are presented below:

- Fuel Coolant - break in coolant line in engine bay results in loss of engine.
  Only aircraft vulnerability increments are due to FAEC fuel coolant lines external to the engine bay.
  Fuel loss due to rupture of coolant line is not critical to aircraft survival.
- Air Coolant - typical combat damage to air lines will result in loss of FAEC within 30 minutes for all locations.
  Only additional line routings considered in the vulnerability assessment.
  (Note: Existing lines from ECS bay is included in VA assessment since RWR is not considered mission critical for reasons other than FAEC operation.)
- Liquid Coolant - typical combat damage will result in FAEC loss within 30 minutes.
  Only additional coolant line routings considered in vulnerability assessment.
- All Coolant Configurations - FAEC input/output vulnerability considered the same for all coolant/location configurations.
  S/V impacts quantified in terms of -

Aircraft vulnerable area	-	Areas which result in loss of aircraft.
Aircraft loss rates	-	Percent change in aircraft losses per 1000 sorties.
Engine loss rates	-	Areas which result in loss of engine. This V/A contributes both to aircraft loss rates (possible to impact both engines) and to abort rates (loss of one engine only).
Controller vulnerable area	-	Areas which result in loss of engine controller only, loss of controller can be considered as resulting in a mission abort.

In all cases the aircraft vulnerable area is considered to be 4 square feet.

Air cooled systems are vulnerable only to control damage. Being air cooled there is no danger that loss of a control due to a hit would cause loss of the aircraft or engine due to a fire. It should be remembered from the groundrules that the loss of the control only results in mission abort. This is reflected in the negative value of loss rate when compared to the baseline fuel-cooled system, Table XI.

This table also presents a comparison of liquid cooling for the FAEC when bay or boom area mounted against the baseline. The engine mounted FAEC control and the FAEC control in the forward fuselage are not evaluated. The engine mounted control would be cooled with fuel if liquid cooling is advantageous. The forward fuselage location would be air cooled to be consistent with the existing equipment in this area. As with the air cooled system a hit in the two locations considered leads only to control damage and thus the abort situation established in the groundrules. Again this leads to a negative impact on the aircraft loss rate.

For fuel cooled systems only the forward fuselage locations is not evaluated. It was considered undesirable to introduce a new coolant into this area where the adjacent avionics uses air cooling. All of the locations present a positive, or increased, impact on the aircraft loss rate. The engine bay and boom location fuel lines are new and their loss can lead to loss of the engine or aircraft. The FAEC system on engine mount location is positive with respect to the baseline EEC system due to an added line which was introduced to bring fuel across the aircraft/engine interface for control cooling. In the baseline system this fuel is taken from pump interstage and returned to pump inlet.

The fuel cooled boom location places the FAEC in an area in which damage or fire is more prone to loss of the aircraft than any other location. This attributed to the location of the aircraft control surface cable routing in the study aircraft.

The airframer conclusion on vulnerability and survivability is that either liquid or air cooling is the preferred medium for engine electronic control cooling. Fuel is considered slightly less desirable on an S/V basis due to the potential for loss of aircraft or engine that does not exist with the other cooling methods.

SAFETY

The use of the FAEC provides a slight improvement in the safety of the study aircraft. Safety in this context may be defined as the lack of any hazard which may cause loss of the aircraft. Table XII presents an evaluation of the safety of air and fuel cooled FAEC systems. The safety of the air cooled version of the FAEC system benefits mainly from the reconfiguration of the hydromechanical portions of the system when compared to the baseline unified fuel control. The fuel interface remains the same in that only one fuel line connect aircraft and engine. The fuel cooled FAEC systems again receive the benefits of the hydromechanical improvements to the system. The use of a separate fuel line for fuel cooling causes a small decrease in the safety of the configuration. An overall evaluation of both the air and fuel cooled FAEC systems shows a slight improvement for this system over the current B/M engine control. However, no significant difference is found between the safety of the air and fuel cooled systems.

WEIGHT

The weight of the electrical wiring and the shielding necessary to protect against specified levels of EMI was the major driver for keeping the electronic control on or very near to the engine. The strength of this factor and whether or not other factors are of more importance in the final analysis depends greatly on the design criteria employed in installing the cables. In this study two different cabling philosophies were examined in detail and their effects on aircraft weight documented. The engine manufacturer's design

criteria specified the use of a minimum of 20 gage wire in a stainless steel braid for those cables mating directly with the engine and 22 gage wire in any other high temperature areas. Four engine/control cables are defined as the only wiring that could introduce a significant weight penalty and each is treated as an individual cable. The interface between engine and aircraft cabling is established at the firewall with pigtails leading from the engine to the firewall.

The airframe manufacturer uses a similar but less conservative cabling criteria and assumed that some 80% of the wiring can be reduced in gages where environment permits. Weight could also be saved by moving the interface between engine and aircraft wiring closer to the engine and to accomplish this, J-boxes and jumper cables are instituted. The use of tinned copper braid shielding is also specified as sufficient for the aircraft cables.

A summary of the two sets of design criteria is presented in Table XIII.

The results of the electrical cabling trade study, Table XIV, shows the unsurprising results that the further away from the engine the control is placed the heavier the cables. It also shows that the magnitude of weight gain is heavily dependent on the criteria used in selecting the cables. The airframer's wiring concept lists cable weight additions approximately half that of those specified using the engine manufacturer's criteria. The table also shows that the weight penalties are significant for both the boom and forward bay locations no matter whose criteria is used. The forward bay is not located on the centerline explaining why airframe cable lengths are dependent on whether one measures from the left or right engine.

The results of the electrical cabling/EMI study show that the preferred control mounting location is on the engine since cable length and weight are at a minimum. MIL-C-38999 connectors have been specified along with the prohibition of filters or metal overbraid in the EEC/airframe wiring. The study also showed that existing electrical cable connections are sufficient to handle any of the control configurations and no additional penetration points need be made.

If the cabling is to meet the EMI compatibility requirements as laid down in the baseline engine specification then there will be a further weight penalty to some of the control mounting locations. For the engine mounted control there is no additional penalty as the steel braid cables already have sufficient shielding, while over 14 Kg can be added to the cable weights for a control mounted in the forward bay, Table XV. The penalty is higher for those cables designed to the less conservative airframer standards than those designed to the engine manufacturer's specifications.

Table XVI summarizes the effect on engine weight on the different control mounting configurations. Fuel cooled control are generally a few pounds lighter than their air cooled counterparts because the improved cooling performance of the fuel allows for greater packing density because the shape of the available volume to mount the control in these two regimes does not permit efficient packaging. Two values are given for the forward bay control weight because the pressure transducers and associated electronic interfaces are engine mounted, while the rest of the control is located off engine. Electrical cable weights shown on this figure represent the engine manufacturer's wiring concept weights. The boom location has a separate remote trim control (and the associated 2.2 Kg penalty) because the EEC cannot be approached when the engine is running due to its proximity to the jet fuel starter. The last line shows the engine weight change with respect to the engine baseline of each of the control mounting configurations.

Table XVII summarizes the change in aircraft weight for the different control mounting configurations. The first row is a carry over from the preceding table of the change in engine weight for each configuration, but is here doubled since the baseline is a twin engine aircraft. The electrical cable weights reflect the airframe wire weight and structure reflects some minor changes in the weight of the aircraft structure. The individual items are summed under total change in weight, but since the added weight does not coincide with the location of the center of gravity, the aircraft must be rebalanced. Ballast is either added or removed depending on the weight distribution of each configuration and the final aircraft change in weight is given in the last row.

This analysis clearly shows only the engine mounted control configurations to be preferable. The nacelle, forward bay and boom locations all add between 32 to 68.5 kilograms extra weight, while the air cooled engine mounted control configuration adds less than 3-4 Kg and the fuel cooled verions but 0.5 Kg.

LIFE CYCLE COST

Previous analysis has shown boom mounting as too heavy and fuel cooling for the forward fuselage location has been discounted as impractical. The consideration of the use of liquid coolant was not included in the LCC due to other increased aircraft requirements. This leaves five cooling/mounting configurations to be compared with the baseline system under life cycle cost analysis. These systems are air and fuel cooled engine mountings, air and fuel cooled nacelle mounting and air cooled forward fuselage mounting. The life cycle cost groundrules assume that the FAEC control will be installed on the study aircraft between S/N 300 and will be procured for the following 400-450 aircraft. It is also assumed there will be no retrofit of the control to earlier aircraft, that each plane will be utilized 45 hours a month for the next ten years, and that 25% of the engines will be spares. None of the fuel saving effects of the improved performance were incorporated into this analysis and costs are based on 1974 dollars. The system of life cycle cost is composed of three main cost categories which are as follows:

- Nonrecurring (RDTE) Cost -
  - Design and Development, Ground Test, Flight Test, Prototype Hardware, Qualification Testing, Support of Flight Test Program Rate Tooling
- Recurring (Production, Including Support) Cost -
  - Cost of Production Incorporation, Hardware, Mods, Retrofit if any, AGE, Data, Training De-

vices, Initial Spares.

. Logistic Support Cost - 10 Year Cost to Support Changes of Hardware in the Field.

. Maintenance, Labor, Material, Replenishment Spares, Supply Administration, Transportation.

The first category is the nonrecurring costs. This includes the research, design development and test costs that are incurred but once no matter how many units are produced. The second category is the recurring cost of producing or retrofitting units. Combined, these first two categories are referred to as the acquisition cost. The third cost category is the logistic support cost, which is the labor, maintenance, material, and miscellaneous costs incurred to support the hardware in the field over the ten year study life.

Table XVIII lists the acquisition costs with respect to the baseline for each of the control configurations broken down into non-recurring and recurring costs. A look at the total nonrecurring or RDTE costs shows little difference among the five configurations. A study of the recurring costs reveals the engine mounted control, either air or fuel cooled, as the best choice, and when comparing total acquisition cost this is reconfirmed. The forward bay mounted control shows up almost as well as the engine mounted units, but both nacelle mounted control configurations are significantly more expensive.

A comparison of logistic support cost trends, Table XIX, but here excluding removal and replacement of labor and airframe components, shows that any of the FAEC configurations will have a dramatic improvement over the current system costs. Among the different configurations the fuel cooled units have the lowest cost per flight hour averaging about seven percent less than the air cooled units. The best configuration presented here is the engine mounted fuel cooled control, while the worst is the forward fuselage mounted control.

The maintenance impact on the aircraft is summarized in Table XX. Only one configuration, the engine mounted air cooled control, demonstrates a reduced maintenance cost with respect to the baseline. The most expensive system is the nacelle mounted fuel cooled control which, when compared with the baseline, shows over a seven percent increase in maintenance costs while the other systems have somewhat smaller increases.

Logistic support costs over the ten year period set down in the groundrules are indicated in Table XXI. The engine mounted fuel cooled system is the most cost effective of the five different systems evaluated, 18.64 million, while the forward fuselage mounted control shows the least savings.

Table XXIII summarizes the results of the installation trade study. The system with the fastest LRU replacement time is the forward bay control. The system with the fastest engine change time is the air cooled engine mounted control. The engine and nacelle mounting locations are equal for the most reliable systems while all systems were equal for the loest vulnerability while the system with the lightest weight and lowest cost is the fuel cooled engine mounted control configuration.

Based upon the two studies presented, the full authority digital electronic engine control unit(s) should be mounted on the engine to achieve the significant improvements associated with engine replacement time, life cycle cost and control system weight. However, both studies have not projected the potential problems associated with off-mounted engine controls specifically in the areas of control accountability, data transmission and vulnerability. Further substantiation of the need for engine-mounted control is discussed below.

The on-engine control provides an element of complete power plant assembly by the engine manufacturer and assures accountability to the airframer and the operating service. The engine control and its interface connections are essential parts of the propulsion system. By engine mounting the control system, the complete propulsion system is fully assembled and tested prior to aircraft installation providing fault isolation and accountability. Thus, avoiding airframer and engine manufacturer logistic problems.

The engine controller represents the lowest level in the computer hierarchy of spatially separated or distributed computer systems. These include the auto throttle system, the aircraft propulsion management computer, the flight controls and other aircraft computers, recorders, sensors, and displays as shown in Figure 13.

Efficient distributed computer computation and control are made possible by advances in minicomputers and microprocessors. The advantage of distributed computer computation and control is that single failures do not propagate and the failed element may be bypassed for continued operation. For example, in a propulsion control system, the pilot would normally command the engines through the automatic propulsion management computer and autothrottle system which have interactions with other aircraft systems. However, in the event of any loss of performance of these systems, the pilot can command the engine controls directly and continue uninterrupted flight. This applies to single engine aircraft and multiple engine aircraft systems.

Distributed computers for computation and control offer an additional advantage in that data may be consolidated and utilized locally with only that data rate needed for transmission. The weight and complexity of the transmission paths are less and their vulnerabilities and failure consequences reduced. For example: typically, the engine control measures a combination of 20 pressures, temperatures, and speeds throughout the engine every 0.01 second. On the basis of these sensed variables, together with the pilot, aircraft, and inlet inputs, the control positions up to eight engine variable geometries and controls two fuel flows. Interruption of this operation for even a fraction of a second can cause gross engine damage. Mounting the control on the engine assures the shortest, least vulnerable, paths for this critical high performance control logic.

The on-engine control provides a degree of protection for momentary interruptions of command data from the pilot and aircraft by continuing uninterrupted engine operation using the most recent good thrust command until faults are cleared on redundant paths energized.

The design necessity that the aircraft engines be separate systems to avoid multiple or single engine failures applies to other engine subsystems as well. For example, each engine includes a separate alternator dedicated to control so that multiple engine and engine controls are independent of the aircraft electrical power systems. Each engine includes its own fuel and hydraulic pumps to avoid multiple engine dependence upon these aircraft elements.

An important element in propulsion control system design is to assure that a single failure will not propagate to the engine controls of more than one engine. The aircraft designer seeks propulsion reliability through multiple engines, a goal which would be completely compromised were the control systems of those engines in any way combined. For example, if multiple engine controls were located in the same aircraft electronic bay, a single battle damage occurrence at that location could result in total loss of aircraft propulsion which would not have occurred had the engine controls been spatially separated and individually on-engine mounted. Similarly, if the engine controls were moved to the electronic bay and there supplied from common aircraft electrical and cooling systems, a failure in those aircraft systems could affect multiple engine controls. On-engine controls, on the other hand, obtain electrical power from individual, dedicated, engine-driven alternators which are completely separate from the aircraft electrical system. To increase reliability, these control alternators have no brushes, bearings, or constant-speed drives; the on-engine control having been designed to accept variable frequency, variable voltage directly from the alternator. Present on-engine controls utilize fuel cooling for supersonic applications and engine-supplied cooling air for subsonic applications. Both approaches avoid dependence upon the aircraft environmental control system.

THE OFF ENGINE-MOUNTED CONTROL

Installing the control unit(s) in a more benign environment such as the aircraft electronic bay may enhance the control component reliability. However, a net decrease in overall propulsion system reliability would be evident because the gains in electronic reliability provided by on-engine electronics is lost in increased vulnerability to the overall propulsion system. The increased vulnerability comes about in two respects:

- Risk of loss of continuity between control and engine.
- Risk that a single event could affect multiple engine controls. The risk of loss continuity between a remotely mounted control and the engine and the severe consequences that would result are due to the following:
  - There are a large number of connections associated with up to 20 on-engine pressure, temperature, speed, and variable geometry position sensors from the engine to the control.
  - There also are many additional connections from the control to the engine required for control of two fuel valves and up to eight variable geometries.
  - The engine sensor, electronic, actuator combinations are high gain, fast, multivariable servo loops. Even momentary interruption in long vulnerable aircraft conductors or their connectors could result in propulsion loss or malfunction.
  - The removal of electronics from the engine would preclude present preamplification, conversion, and multiplexing and/or use of fiber optics. Certain sensors will not produce present accuracies with remote electronics. In addition, the present pressure sensors require electronic level temperature control for present accuracies. If on-engine electronic environmental control were provided for the sensors, the sensor, signal processing, the multiplexer, and the power amplifiers, all of the present on-engine electronic technology would be required and one might as well include the microprocessor and program memory to complete the on-engine control as at present.
  - The removal of electronics from the engine would preclude the present practice of continuously testing the redundant propulsion command data link from the aircraft to the engine by transmitting the received propulsion command from the engine control to the aircraft and holding present value of thrust until a validated thrust command is received.

MOUNTING LOCATIONS FOR FUTURE SYSTEMS

The systems analyzed in this paper are based on engine control systems employing a single electronic control unit. Future control systems such as those developed in the U. S. Navy FADEC program (Ref. 2) will employ dual electronic control systems. This trend toward multiple controls is being driven by the need for improved system mission reliability and the ability to integrate with the aircraft control system. Increased integration with the engine inlet and aircraft systems allows technology advances such as thrust control, advanced approach control concepts, power by wire, optic communication, advanced fuel management and inlet distortion accommodation. These advanced concepts provide a system approach to the design of future aircraft and engine systems. For these integration benefits to be realized the on-board controls must operate at a reliability level equivalent to the flight control systems with which they will be trading information. Advanced multiple control system concepts are in the planning stages which will increase the MTBF of the engine control system to these desired levels of reliability.

CONCLUSIONS

SUBCOMPONENT/CONTROL PACKAGING

P&WA/HSD electronic control subcomponent packaging reflects the state-of-the-art in engine application. Whereas the GE multi-layer ceramic module (MCM) packaging technology is an advanced concept and a unique approach to propulsion control design.

ELECTRONIC CONTROL LOCATION EVALUATION

The data evaluated in this study leads to the engine mounted electronic control system with a choice between two versions; air cooled and fuel cooled. When maintainability, reliability, survivability, and

nerability, safety and life cycle cost are considered there is no clear choice between an air cooled engine mounted control and a fuel cooled engine mounted control. The air cooled version could be selected based on the slight S/V and safety benefits and the fuel cooled version on the basis of weight and cost benefits. Clearly the two systems are both acceptable and the final decision on configuration must be considered a designer's choice, driven by the overriding selection criteria for the particular aircraft.

REFERENCES

1. Lennox, T. G., "Full Authority Digital Electronic Control" Contract N00019-76-C-0422, June 1978, Technical Report FR 8652
2. Lennox, T. G., Vizzini, R. W., Miller, R. J., "Full Authority Digital Electronic Control, Turbofan Demonstration", ASME Conference, New Orleans, Louisiana, March 1980
3. Beattie, E. C., Carlisle, L. B. Katzer, T. J., Spock, W. R., "Attitude Control/Engine Control Systems", Contract N00019-72-C-0612, June 1975, Final Report PWA-5275
4. Lenox, T. G., "Electronic Control Cooling System Design Study", Contract F33657-73-C-0619, July 1974 Technical Report AFAPL-TR-74-23
5. Anon, "Reduced Cost Technology Studies (U)", Pratt and Whitney Aircraft, September 1974, Report PWA-5074

ACKNOWLEDGEMENTS

The author sincerely appreciates the support and cooperation of the following corporations:

. McDonnell Douglas, St. Louis, MO
. Pratt and Whitney Aircraft, Government Products Division, West Palm Beach, FL
. Pratt and Whitney Aircraft, Commercial Products Division, East Hartford, CT
. General Electric Company, Aircraft Engine Group, Evendale, OH

Table I - Summary of GE FADEC Test Hours

Test	FADEC Units S/N 001 (Hrs)	S/N 002 (Hrs)	S/N 003 (Hrs)	Totals (Hrs)
Subassembly	20	20	20	60
Component	120	120	120	360
Vibration	20	-	-	20
System	100	120	80	300
Engine	28	80	170	278
Environmental (MIL-STD & Spec)	600	-	-	600
Subtotals:	888	340	390	1618

Table II - Summary of the PWA/HSD FADEC Operating Hours

Test	Breadboard #1	FADEC #1	Breadboard #2	FADEC #2
Advanced Engine Simulation (CLB)	230	0	251	251
F401 System Test (Design Assurance)	778	380	0	0
SLS Test	162	54 (31)	0	13 (12)
Alt. Test	90	60 (25)	0	0
Environmental	0	0	126	126
Development	1784	98	2580	144
Subtotals:	3044	592	2957	534
Total Hours	7127			
Engine Test Hours		(56)		(12)
Total Engine Test Hours	68			

Table III - Control System Component Redesign Required For Mounting Of Pressure Transducers On Engine With Control Mounted In Equipment Bay

	Weight Increment (Kg)	Cost Increment (% of Total System Cost)
Pressure transducers packaged separately, multiplexer added, transducers eliminated from control	+ 5.4	+ .06
Serial digital transducer cable added	+ 1.8	+ 0.14
Cooling fuel lines added	+ 1.1	+ 0.01
Pressure lines eliminated	- 1.1	-0.03
Total:	+ 7.2	+ .18

Table IV - Estimated Weight And Acquisition Cost Increment For Bay-Mounted Electronic Control Relative To Engine-Mounted Control

Type of ECS Cooling	Engine Configuration*	Weight Increment (Kg)	Acquisition Cost Increment (% of Total System Cost)
Liquid	1	+30	+5.4
Liquid	2	+40	+6.1
Liquid	3	+20	+4.8
Air	1	+15	+5.2
Air	2	+20	+5.8
Air	3	+9	+4.7

* Lift-cruise engine configurations identified in Figure 8

Table V - Relative Organizational Level Maintenance Labor For Engine-Mounted And Bay-Mounted Electronic Control Systems

	Component Replacement Time	Engine Change Time	Total Time
Engine-mounted control and pressure transducers	0.016	0.984	1.000
Bay-mounted control and pressure transducers	0.015	1.025	1.040
Bay-mounted control with engine-mounted pressure transducers	0.011	1.027	1.038

Table VI - Life Cycle Cost Increment For Bay-Mounted Electronic Control Relative To Engine-Mounted Control

Type of ECS Cooling	Engine Configuration*	Calculated Life Cycle Cost (Dollars)**
Liquid	1	+43,100,000
Liquid	2	+54,100,000
Liquid	3	+34,500,000
Air	1	+33,900,000
Air	2	+41,600,000
Air	3	+28,500,000

* Lift-cruise engine configurations identified in Figure 8

** Cost calculations based on 1973 economy

Table VII - Coolant Systems Comparison

	FUEL	AIR	LIQUID
Safety	- Leakage-Spillage	+ No spillage or Leakage	- Leakage-Spillage
	- Highly Flammable	+ Non-Flammable	- Semi-Flammable
	- Toxic Decomposition	+ Non-Corrosive	- Toxic Decomposition
	- A. I. T. Exposure		- Corrosive Products
Reliability	+ Good Component Temp.	- Higher Component Temp.	+ Good Component Temp.
	- Wide Coolant Temp. Range	+ Good Coolant Temp. Range	+ Good Coolant Temp. Range
	+ Redundant Coolant Supply	- Single Coolant Supply	- Single Failure Loss
		+ Emerg. Coolant Supply	
Maintainability	- Coolant at Engine Idle	- Coolant at Engine Idle, ECS On	- Coolant at Engine Idle, ECS & Radar On
	- Valve & Control Req'd	+ No Controls	- Disconnects & Control Valve Required
	- Disconnects & Elec. Interface		

Table VIII - Coolant System Summary At Various FAEC Locations

	ENGINE MOUNTED			NACELLE			BOOM			FORWARD BAY		
TRADE FACTORS	FUEL	AIR	LIQUID	FUEL	AIR	LIQUID	FUEL	AIR	LIQUID	FUEL	AIR	LIQUID
Weight (Kg)	5.4	6.3	6.4	7.2	5.8	6.4	9.1	5.8	8.2	6.6	4.2	5.7
Ballast (Kg)	0.3	0.3	0	0.9	0.7	0	2.7	1.5	0.2	0.3	0.2	0.3
Total (Kg)	5.7	7.1	6.4	8.1	6.5	6.4	11.8	7.3	8.4	6.3	4.0	5.4
Line Size (cm)	1.2	3.7	1.2	1.2	3.7	1.2	1.2	3.0	1.2	1.2	3.0	1.2
Coolant Flow (Kg/Hr)	15.5	13.6	11.8	15.5	13.6	11.8	14.6	11.4	10.0	11.4	8.7	7.7
Growth Impact Load-Watts	158	158	158	158	158	158	133	133	133	101	101	101
*Component Temp °C	99	116	116	99	116	99	99	105	116	99	105	116

* Based on 74°C Maximum EEC Surface Temperature

Table IX - FAEC Maintainability Impact

LOCATION	ENGINE	NACELLE	BOOM	FORWARD	FUEL COOLED ENGINE
	AIR COOLED				
Access Time - Min	20	20	101	6-EEC 20-XDUCERS	29
LRU Replacement Time - Min	7	6	10	2-EEC 8-XDUCERS	9
Increase in Engine Change Time - DMMH/FH	.00007	.00252	.04628	.00335	.00128
Maintenance of A/C Cooling System - DMMH/FH	0	0	0	0	.00351

Table X - FAEC System Reliability

(Failures Per 10^6 Hours)

ITEM	BASELINE (HOT FUEL)	ENGINE MOUNTED AIR	ENGINE MOUNTED FUEL	NACELLE MOUNTED AIR	NACELLE MOUNTED FUEL	TAIL BOOM AIR	TAIL BOOM FUEL	FORWARD BAY AIR
ENGINE								
EEC	2000	693	542	693	542	695	542	414 - EEC 414 - Engine Mtd Transducer
UFC	1000	--	--	--	--	--	--	--
GG Control	--	765	765	765	765	765	765	765
Aug. Control	--	--	--	--	--	--	--	--
Wiring	36	53	53	53	53	53	53	53
Actuation	2897	2574	2574	2574	2574	2574	2574	2574
Total	5933	4085	3934	4085	3934	4085	3934	4281
AIRCRAFT								
B Probe	0	400	400	400	400	400	400	400
Wiring	0	8	8	8	8	84	84	54
Fuel Cooling Valves	0	--	150	--	150	--	150	--
Remote Trim	0	--	--	--	--	20	20	--
AIC Addition	0	6	6	6	6	6	6	6
Total	0	414	564	414	564	510	660	460
TOTAL	5933	4499	4498	4499	4498	4595	4591	4741
MTBF	168 Hr.	222 Hr.	222 Hr.	222 Hr.	222 Hr.	218 Hr.	218 Hr.	211 Hr.

Table XI - Vulnerability Comparisons Of FAEC Configurations

LOSS RATE IMPACT

	AIR COOLED VERSIONS	LIQUID COOLANT VERSIONS	FUEL COOLED VERSIONS
Baseline (Fuel Cooled)	N/A	N/A	0
On Engine Mounted	-0.2%	N/A	+0.1
Nacelle	-0.2%	-0.2%	+0.2%
Boom Area Location	-0.2%	-0.2%	+ .4%
A/C Bay	-0.2%	N/A	N/A

Table XII - Safety Evaluation

EVALUATION CRITERIA	BASELINE	AIR COOLED	FUEL COOLED
λ_{C^*} Engine	144.4×10^{-6}	144.4×10^{-6}	144.4×10^{-6}
λ_{C^*} Controller (Causing in-flight shutdown)	189.0×10^{-6}	37.0×10^{-6}	37.0×10^{-6}
λ_{C^*} Fuel Interface (Causing fire)	7.6×10^{-6}	7.6×10^{-6}	9.6×10^{-6}
AIRCRAFT LOSS PER 100K HR.	5.2626	5.2375	5.2376

λ_C: That Portion Of the Total Failure Rate Which Is Safety Critical (Contributes to a Critical Hazard).

Table XIII - Design Criteria Rationale

ENGINE MANUFACTURER CRITERIA	AIRFRAME MANUFACTURER CRITERIA	BOTH
o	o	o The four engine/EEC cables are the only wiring introducing a significant weight penalty
o No reduction in originally wire gages	o Assume 80% of wiring can be reduced in gage where environment permits	o Wiring mating directly to the engine may not be smaller than 20 gage and wire in high temperature areas may not be smaller than 22 gage
o Pigtails on engine	o J-Box on engine o Jumper Cable Length: Nacelle Mtd.- 0.9 meters Boom Mtd. - 2.4 meters Fwd. Mtd. - 1.2 meters	
o	o	o Each engine/EEC cable must be treated as an individual cable and not combined with others
o Stainless steel braid on engine wiring	o Tinned copper braid on aircraft wiring	o Each engine/EEC cable requires overall metal bundle braid

Table XIV - Electrical Cable Trade Study

	LOCATION				
INSTALLATION CRITERIA	BASELINE EEC	ENGINE FAEC	NACELLE FAEC	BOOM FAEC	FWD FUS FAEC
Engine Manufacturer Concept					
Engine Cable Length per Engine (M)	--	0	3.1	6 2	3.1
Airframe Cable Length per Electronic Control (M)	--	0	0	0.6	10.4 Right 7.3 Left
System Disconnects (per electronic control)	0	0	1	2	3
Engine Wire Weight (Kg/Aircraft)	23.0	22.5	38.0	53.0	38.0
Airframe Wire Weight (Kg/Aircraft)	0.7	1.6	1.9	6.0	34.0
Total System Wire Weight (Kg/Aircraft)	23.5	24.5	39.0	59.0	72.0
Weight Penalty (Kg over baseline)	--	0.4	16.0	36.0	48.0
Airframe Manufacturer Concept					
Engine Cable Length (Ft) per engine	--	0	0	0	0
Airframe Cable Length per control (M)	--	0	3	10	11.5 Right 8.7 Left
System Disconnects (per control)	0	0	2	3	4
Engine Wire Weight (Kg/Aircraft)	23.0	22.5	23.0	23.0	23.0
Airframe Wire Weight (Kg/Aircraft)	0.7	1.6	7.0	16.4	29.9
Total System Wire Weight (Kg)	23.5	24.5	31.0	40.0	53.0
Weight Penalty (Kg over baseline)	--	0.4	7.2	16.4	30.0

Table XV - EMI Protection Weight Penalty (Aircraft Wiring Only)

DESIGN CRITERIA	ENGINE	NACELLE	BOOM	FORWARD
Engine Manufacturer Concept: Pigtails & Max GA Wire (Kg/Aircraft)	--	--	1.2	14.0
Airframe Manufacturer Concept: J-Box, Jumper Bundles & Min Gage Wire (Kg/Aircraft)	--	1.8	5.6	14.5

NOTE: Above weights do not include 0.2 Kg/Aircraft weight penalty (estimated) for EEC internal EMI provisions

Table XVI - Effect on Engine Weight for the Various Control Mounting Locations

	EEC	AIR COOLED-FAEC				FUEL COOLED - FAEC		
ITEM	BASE-LINE	ENGINE	NACELLE	BOOM	FORWARD BAY	ENGINE	NACELLE	BOOM
Electronic Control	11.0	14.6	17.9	16.0	13 + 2.7	14.0	7.0	15.0
Mounts	1.1	0.5	0.5	0.5	0.5 + 0.5	0.5	0.5	0.5
UFC	40.0	36.0	36.0	36.0	36.0	36.0	36.0	36.0
Sensors/Plumbing	1.2	3.2	4.2	5.0	4.5	3.2	4.2	5.0
Misc. Control Hardware	24.5	20.0	20.0	20.0	20.0	20.0	20.0	20.0
Electrical Cables	11.5	11.0	18.9	26.0	18.9	11.0	18.9	26.0
Remote Trim	0	0	0	2.2	0	0	0	2.2
Δ WT. (Kg)	0	-2.3	+9.0	+18.0	+7.8	-3.2	+8.5	+15.6

Table XVII - Summary of the Change in Aircraft Weight For The Various Control Mounting Configurations

FAEC AIRCRAFT WEIGHT STATUS

		AIR COOLED				FUEL COOLED		
ITEM	BASE-LINE	ENGINE	NACELLE	BOOM	FORWARD BAY	ENGINE	NACELLE	BOOM
Δ WT. - 2 Engines	0	-4.8	+18.0	+35.0	+15.2	-6.4	+16.0	+33.5
Sideslip Probe		2.8	2.8	2.8	2.8	2.8	2.8	2.8
Electrical Cables		0.3	0.6	3.0	34.0	0.5	0.9	3.2
Cooling Plumbing		6.0	5.0	6.0	4.2	5.4	7.3	9.1
Structure		0.1	0.5	1.1	0.1	0.1	0.5	1.1
Total Δ Wt.	0	+4.2	+27.0	+48.0	+55.0	+2.5	+27.0	50.0
Ballast	-	=0.7	+6.2	+20.0	-11.5	-2.1	+6.1	+20.0
Aircraft ΔWT.	-	+2.8	+32.5	+68.0	+44.0	+0.5	+33.5	+69.0

Table XVIII - Acquisition Cost Comparison, Cost

Increment From The Current Baseline

(Dollars in Millions)

	ENGINE MOUNTED		NACELLE MOUNTED		FORWARD FUSELAGE MTD
	AIR COOLED	FUEL COOLED	AIR COOLED	FUEL COOLED	AIR COOLED
RDTE	$34.0	$34.0	$34.1	$34.2	$33.5
PRODUCTION	-$71.1	-$71.1	-$58.2	-$58.1	-$69.3
ACQUISITION	-$37.1	-$37.1	-$24.1	-$23.9	-$35.8

Table XIX - Summary Of

Logistic Support Cost Trends

(Recurring Only)

Normalized With Respect To The Baseline System

BASELINE (Current System)	1.000
FAEC	
o Engine Mounted - Air Cooled	.390
o Engine Mounted - Fuel Cooled	.365
o Nacelle Mounted - Air Cooled	.396
o Nacelle Mounted - Fuel Cooled	.371
o Forward Fuselage - Mounted - Air Cooled	.402

Table XX - Summary Of Maintenance Impact

(Organizational Level Only)

Normalized With Respect To The Baseline

Alternate	
Current Baseline	1.000
Engine Mounted - Fuel Cooled	1.020
Engine Mounted - Air Cooled	.965
Nacelle Mounted - Fuel Cooled	1.075
Nacelle Mounted - Air Cooled	1.006
Forward Fuselage - Air Cooled	1.005

Table XXI - Logistic Support Cost Increments

(Dollars in Millions)

	ENGINE MOUNTED		NACELLE MOUNTED		FORWARD FUSELAGE MTD.
	AIR COOLED	FUEL COOLED	AIR COOLED	FUEL COOLED	AIR COOLED
LOGISTIC SUPPORT COST (LSC)	-$17.96	- $18.69	-$17.78	-$18.53	-$17.60
ON-A/C MAINT-ENANCE	-$ .08	+ $ .05	+$ .01	+$ .18	+$ .01
TOTAL LSC (10 YEARS)	-$18.04	-$18.64	-$17.77	-$18.35	-$17.59

Table XXII - Life Cycle Cost Comparison

Cost Increment From Current Baseline

(Dollars in Millions)

	ENGINE MOUNTED		NACELLE MOUNTED		FORWARD FUSELAGE MTD.
	AIR COOLED	FUEL COOLED	AIR COOLED	FUEL COOLED	AIR COOLED
TOTAL ACQUISITION	-$37.1	-$37.1	-$24.1	-$23.9	-$35.8
10 YEAR LOGISTIC SUPPORT COST (LSC)	-$18.04	-$18.64	-$17.8	-$18.35	-$17.6
TOTAL - LIFE CYCLE COST	-$55.1	-$55.7	-$41.9	-$42.3	-$52.4

Table XXIII - Installation Trade Summary

EVALUATION CRITERIA	BASE-LINE	AIR COOLED				FUEL COOLED		
		ENGINE	NACELLE	BOOM	FORWARD BAY	ENGINE	NACELLE	BOOM
Maintainability - LRU Replacement Time - Minutes	60	27	26	127	8 (EEC) 28 (Trans-ducers)	29	28	129
- Engine Change MMH	3.0	3.007	3.45	5.12	3.13	3.04	3.47	5.15
Reliability System MTBF - Hr.	168	222	222	218	211	222	222	218
Safety - % Change in Aircraft Losses/ $.10^5$ Hours	-	-.5%	-.5%	-.5%	-.5%	-.5%	-.5%	-.5%
Vulnerability Loss Rate Chg -%	-	-0.2	-0.2	-0.2	-0.2	+0.1	+0.1	+0.1
EMI Aircraft Protection,(Kg)	-	0	2.1	6.2	14.5	0	2.1	6.2
Weight Kg/Aircraft	-	+2.8	+32.5	+67.0	+45.0	+.5	+33.5	+69.0
Life Cycle Cost Saving $ Million		-55.1	-41.9	-	-52.4	-55.7	-42.3	-

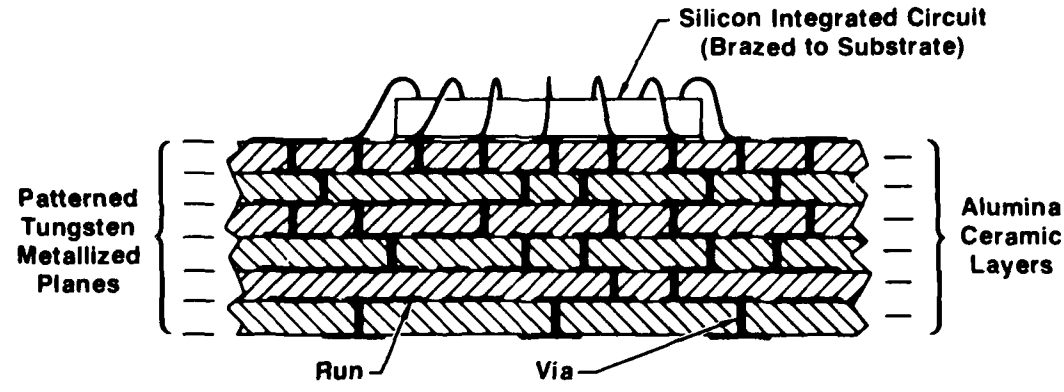


FIGURE 1: MCM CROSS-SECTION SHOWING REFRACTORY METAL CONDUCTOR RUNS AND VIAS

U.S. NAVY/GE FADEC PROGRAM

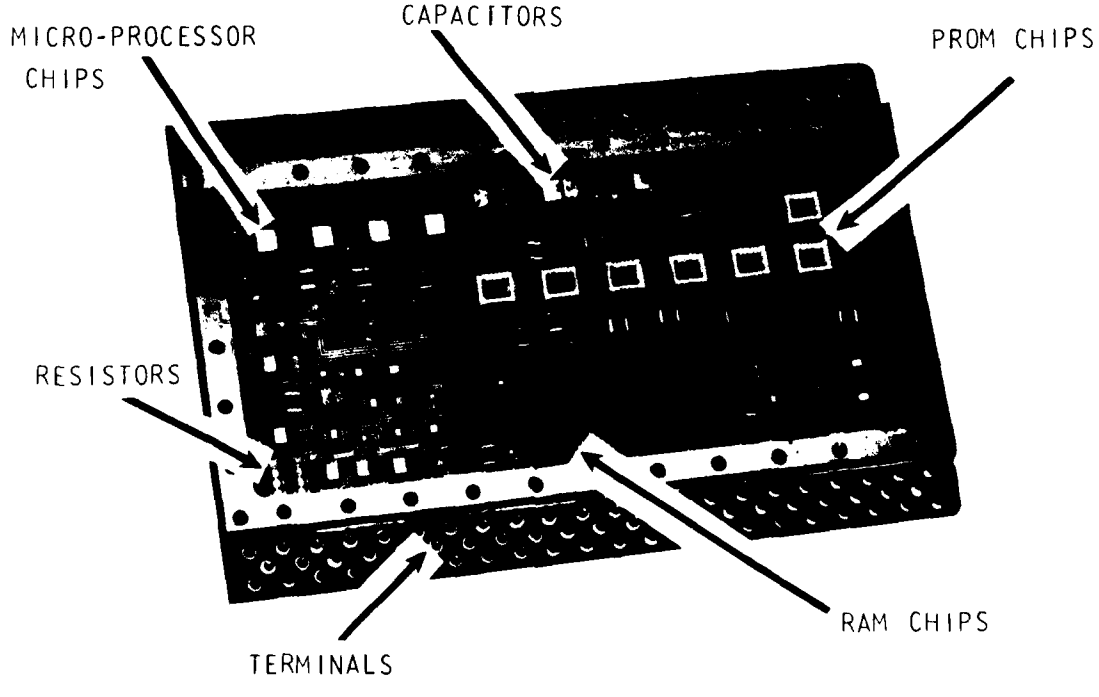


FIGURE 2: MICRO-PROCESSOR, TYPICAL MULTI-LAYER CERAMIC MODULE

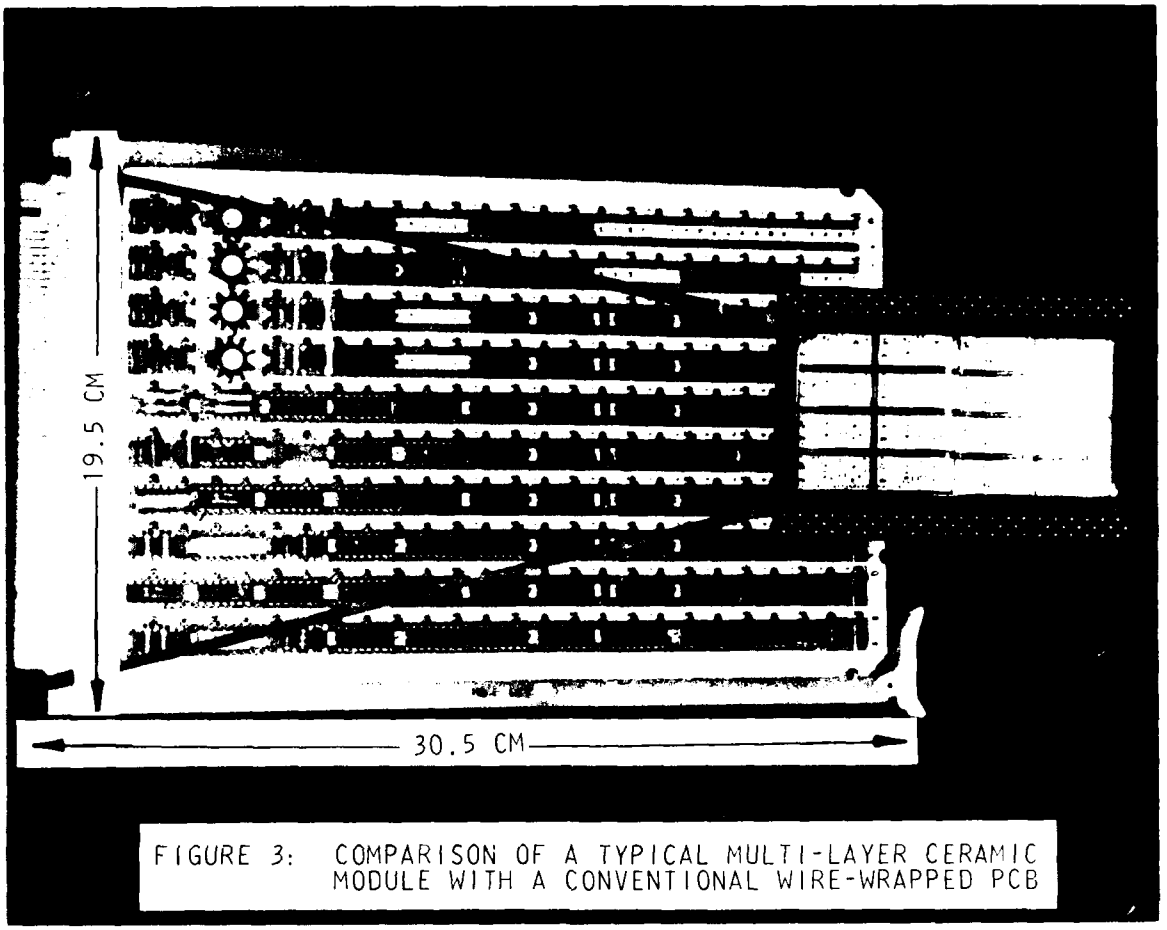


FIGURE 3: COMPARISON OF A TYPICAL MULTI-LAYER CERAMIC MODULE WITH A CONVENTIONAL WIRE-WRAPPED PCB

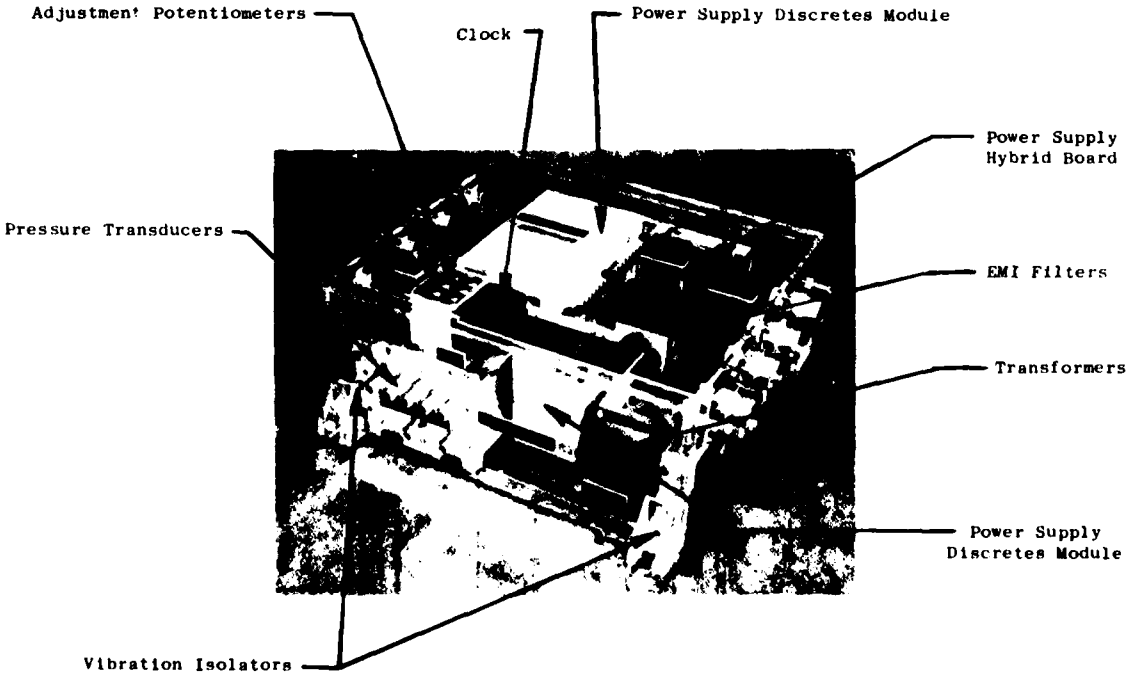


FIGURE 4: GE FADEC MODEL, FRONT VIEW

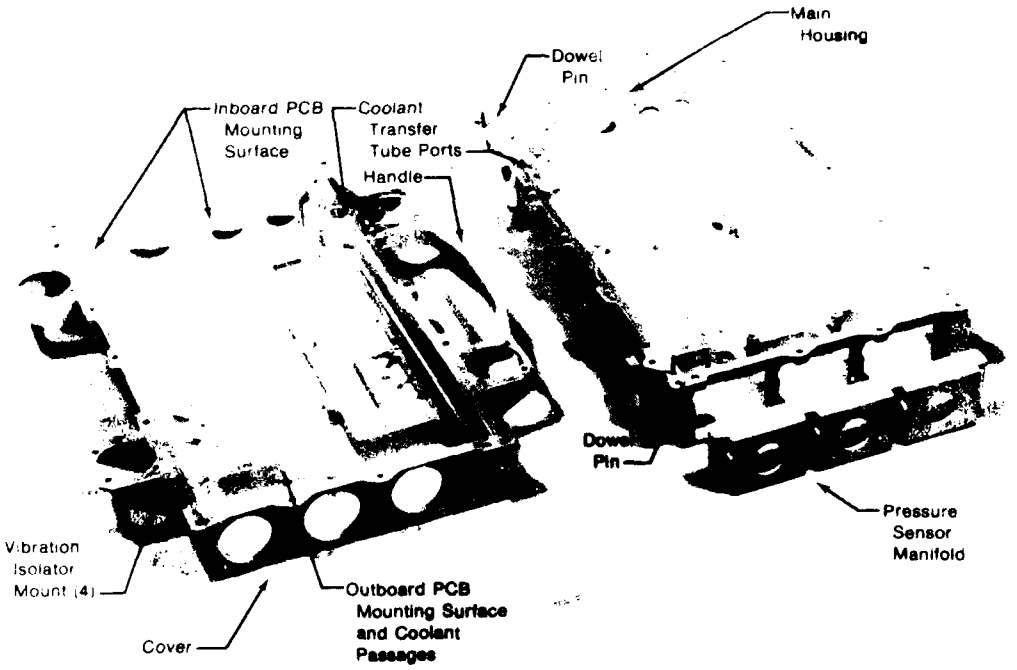


FIGURE 2. ENGINE-MOUNTED CONTROL MODULE COMPONENTS

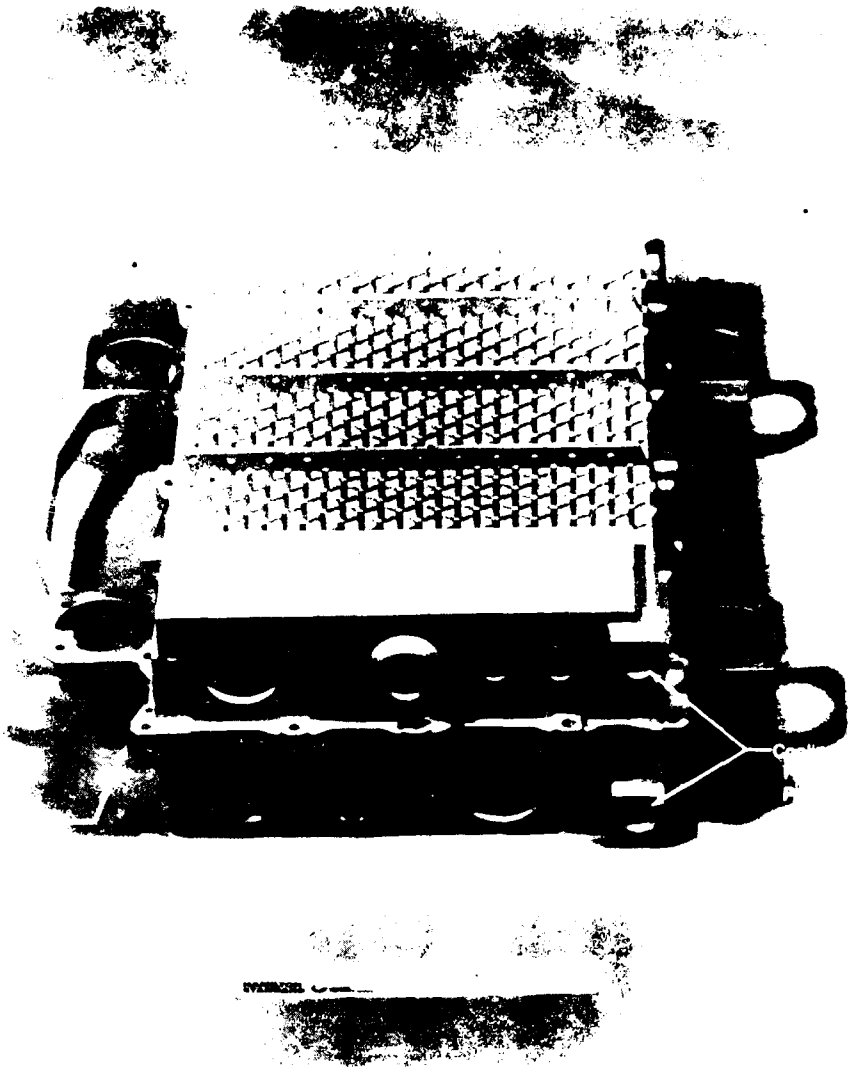
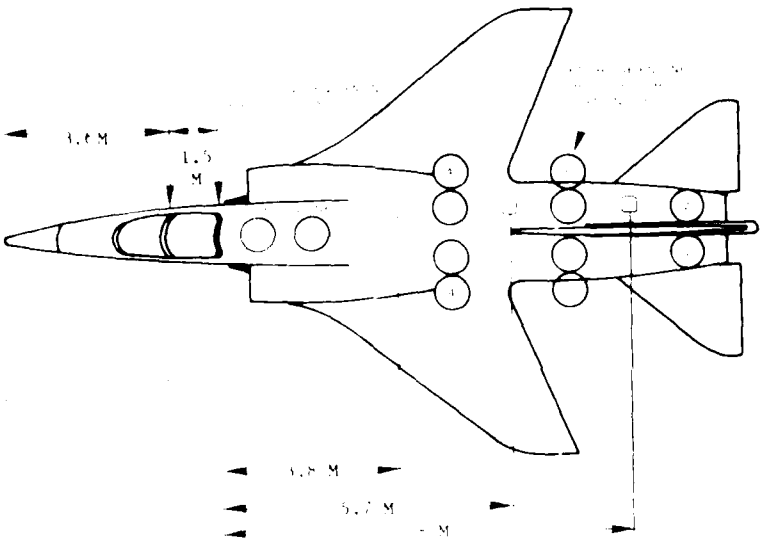


FIGURE 6: ENGINE-MOUNTED CONTROL MODULE ASSEMBLY



STOL A/C CONFIGURATION WITH INTERNAL CABLE ROUTING DISTANCES AND ENGINE-MOUNTED CONTROL LOCATIONS

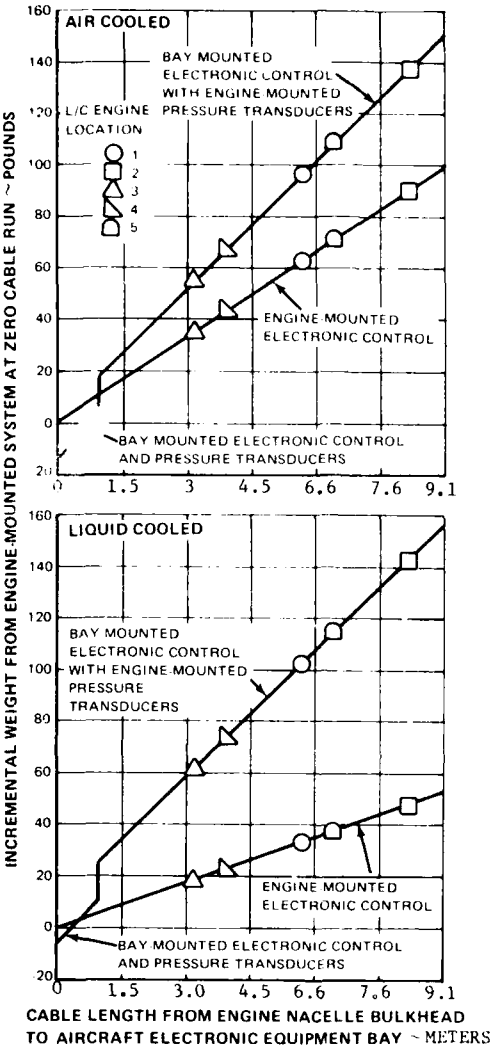


FIGURE 9: ENGINE CONTROL SYSTEMS WEIGHT IMPACT WITH ECS AIR-COOLED (TOP) AND LIQUID-COOLED (BOTTOM) ELECTRONIC CONTROL VS DISTANCE FROM ELECTRONIC EQUIPMENT BAY TO ENGINE NACELLE BULKHEAD

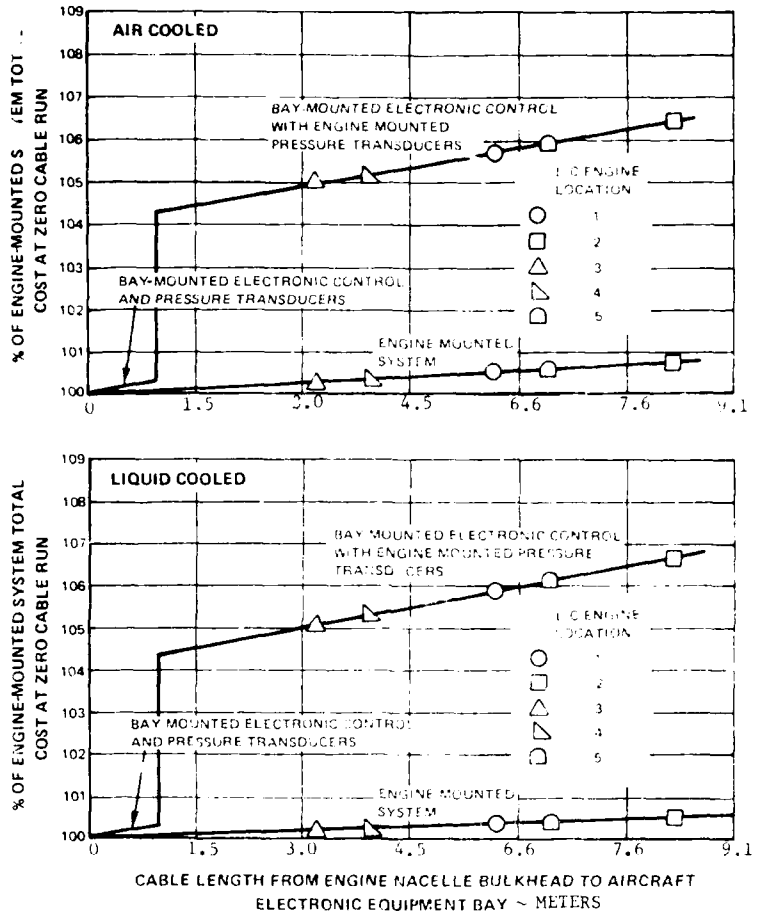


FIGURE 10: ENGINE CONTROL SYSTEM COST IMPACT WITH ECS AIR-COOLED (TOP) AND LIQUID-COOLED (BOTTOM) ELECTRONIC CONTROL VS DISTANCE FROM ELECTRONIC EQUIPMENT BAY TO NACELLE BULKHEAD

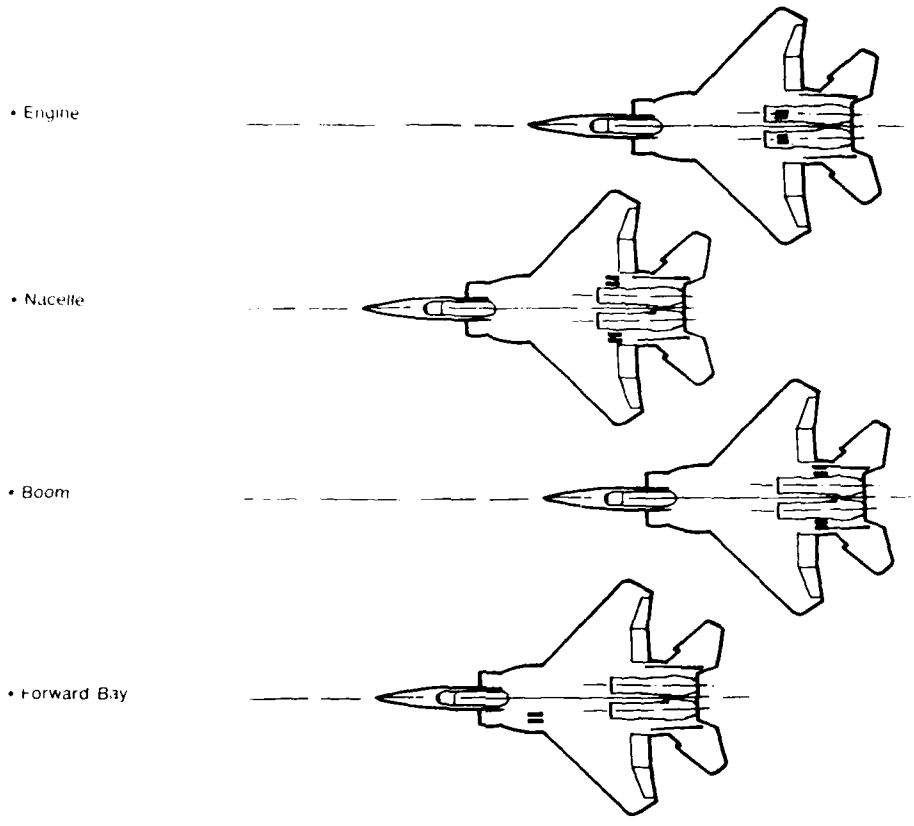


FIGURE 11: FAEC INSTALLATION LOCATIONS STUDIED ON A CURRENT HIGH PERFORMANCE TWIN-ENGINE AIRCRAFT

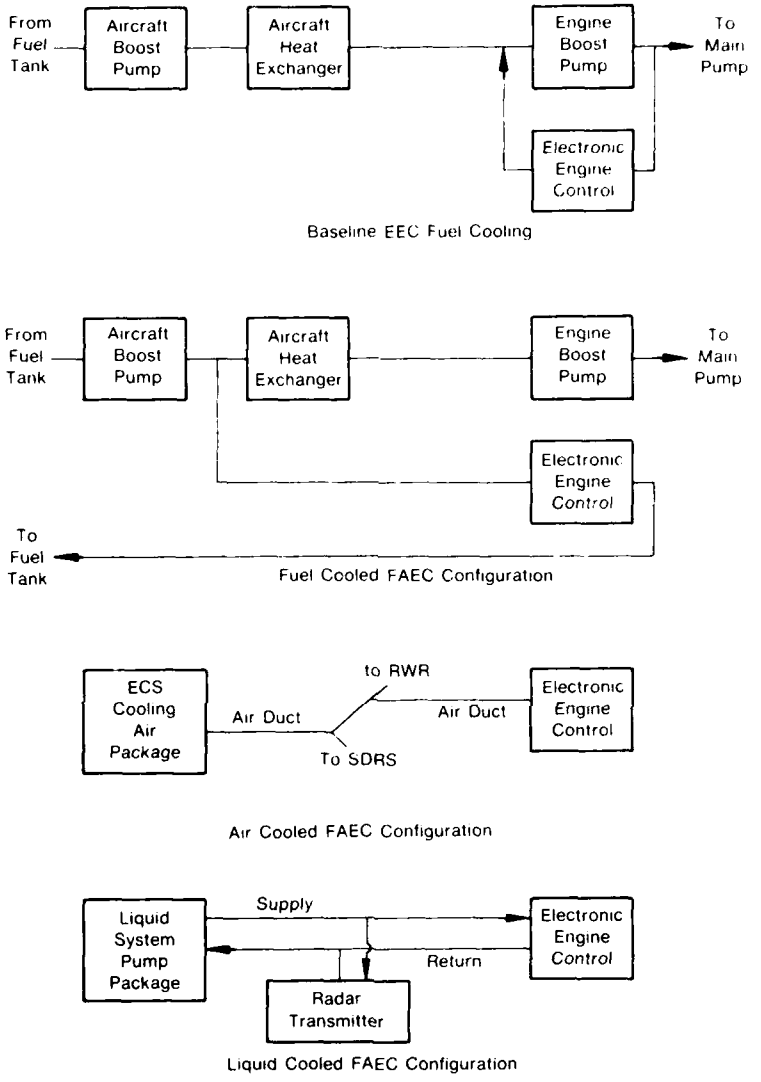


FIGURE 12: FOUR COOLING CONFIGURATIONS, THE BASELINE AND THE STUDIED FUEL, AIR AND LIQUID FAEC COOLING SYSTEMS

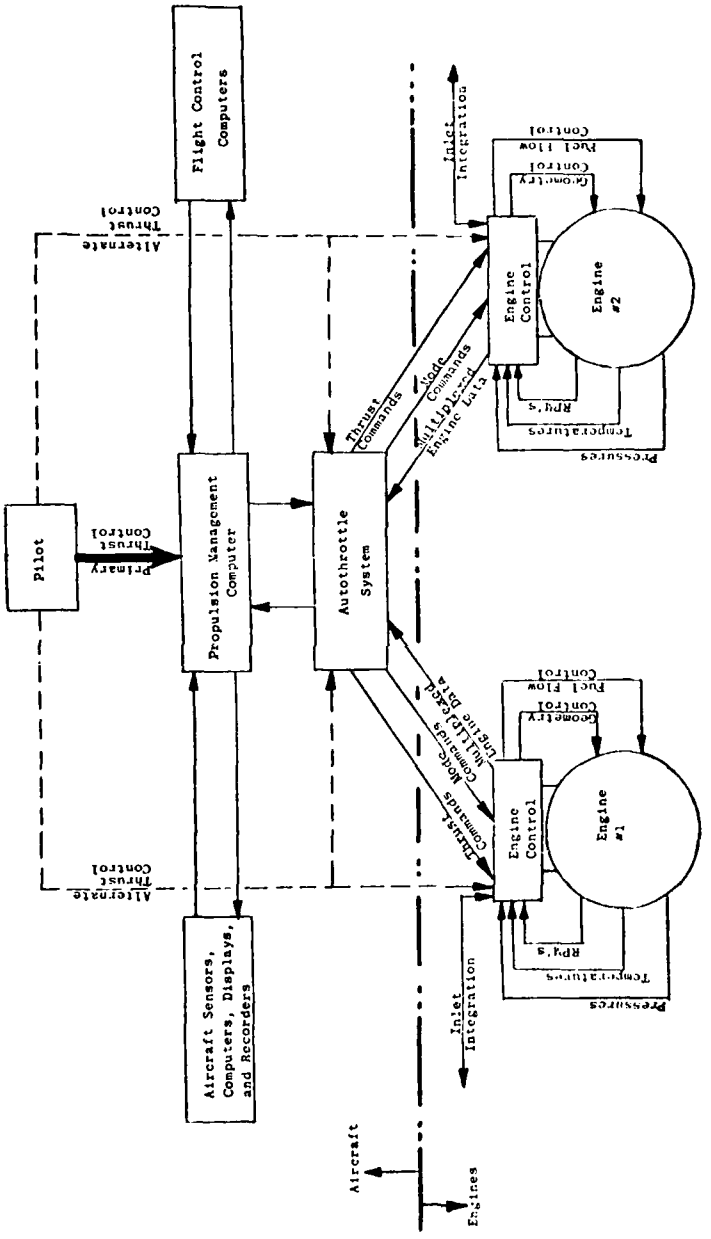


FIGURE 13: HIERARCHY OF DISTRIBUTED PROPULSION CONTROL COMPUTERS.

ALGORITHM DESIGN FOR DIGITAL FEEDBACK CONTROL SYSTEMS

by
J. David Powell
Associate Professor
Department of Aeronautics/Astronautics
Stanford University
Stanford, CA 94305 U.S.A.

SUMMARY

The various design methods available to obtain control equations suitable for programming in the microprocessor will be described. In order to analyze and compare the methods, the z-transform will be briefly discussed and the correspondence between the z and s planes established.

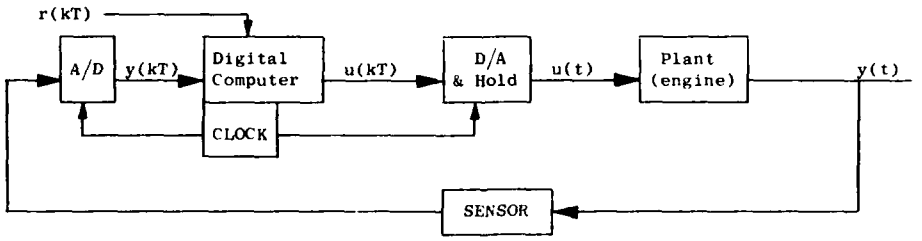
There are two broad categories of design methods: (1) continuous design followed by a digitization and (2) direct digital design. The various digitization methods for the first design category will be described and compared in terms of design ease and accuracy. The second category, direct digital design, will then be described and compared to the first category in terms of design ease and accuracy. Finally, small word size effects and the basic issues in selecting the sample rate will be addressed.

PREFACE

An important aspect of designing any digital control system is the determination of the equations to be programmed in the control computer. The intent of this paper is to provide the theoretical background and practical tools to enable a designer to arrive at these equations. The methods to be studied are for closed-loop (feedback) systems in which the dynamic response of the process being controlled is a major consideration in the design, as is the case for aircraft engine control. The design methods are applicable to any type of computer (from μ-processors to large scale computers); however, the effects of small word size and slow sample rates take on a more important role when using μ-processors.

It will be assumed in the paper that the reader has some knowledge of control system design methods for continuous (or analog) systems such as those covered in the textbooks by Dorf [1] or Ogata [2]. Furthermore, a more complete reference for the subject material of this paper can be found in a digital control textbook by Franklin and Powell [3].

A typical topology of the type of system to be considered is shown in Fig. 1.



r = reference or command input
y = output quantities
u = actuator input signals
A/D = analog-to-digital converter
D/A = digital-to-analog converter

Fig. 1 Basic Control System Block Diagram.

There are two fundamentally different methods for the design of digital algorithms:

(1) Continuous Design: perform a continuous design, then digitize the resulting compensation.

(2) Discrete Design: digitize the plant model, then perform a direct digital design.

Both methods will be covered and their advantages and disadvantages discussed.

The paper will first present the theoretical background for the analysis of any discrete system, followed by a discussion of the two design methods. The effects of small word size will then be discussed followed by a concluding section on sample rate selection.

1. THEORETICAL BACKGROUND

(a) z-Transform - In the analysis of continuous systems, we use the Laplace Transform which is defined by

$$\mathcal{L}\{f(t)\} = F(s) = \int_0^\infty f(t)e^{-st}dt \qquad (1)$$

which leads directly to the important property that

$$\mathcal{L}\{\dot{f}(t)\} = s\,F(s) \quad . \tag{2}$$

This relation enables us to easily find the transfer function of a linear continuous system given the differential equations of that system.

For discrete systems, a very similar procedure is available. The "z-transform" is defined by

$$Z\{f(n)\} = F(z) = \sum_{n=0}^{\infty} f(n)z^{-n} \tag{3}$$

which also leads directly to a property analogous to (2), specifically, that

$$Z\{f(n-1)\} = z^{-1}F(z) \quad . \tag{4}$$

This relation allows us to easily find the transfer function of a discrete system given the difference equations of that system. For example, the general 2nd order difference equation,

$$y(n) = -a_1y(n-1)-a_2y(n-2)+b_0u(n)+b_1u(n-1)+b_2u(n-2) \tag{5}$$

can be converted from y(n), u(n), etc., to the z-transform of those variables by invoking equation (4) once or twice to arrive at,

$$Y(z) = (-a_1z^{-1} - a_2z^{-2})Y(z) + (b_0 + b_1z^{-1} + b_2z^{-2})U(z) \tag{6}$$

which results in the transfer function

$$\frac{Y(z)}{U(z)} = \frac{b_0+b_1z^{-1}+b_2z^{-2}}{1 + a_1z^{-1} + a_2z^{-2}} \quad . \tag{7}$$

(b) <u>z-Transform Inversion</u> - Tables relating simple discrete time functions to their z transform are available in most digital control textbooks, as is also the case for continuous time functions and their Laplace transform.

Given a general z-transform, one can break it up into a sum of elementary terms using partial fraction expansion and find the resulting time series from the tables described above. Again, these procedures are exactly the same as those used for continuous systems.

A z-transform inversion technique which has no continuous counterpart is called long division. Given a z-transform,

$$Y(z) = \frac{N(z)}{D(z)} \tag{8}$$

one simply divides the denominator into the numerator using long division. The result is a polynomial (perhaps infinite) in z, from which the time series can be found by using Eq. (3).

For example, a first order system described by the difference equations,

$$y(n) = ky(n-1) + u(n) \tag{9}$$

yields

$$\frac{Y(z)}{U(z)} = \frac{1}{1-kz^{-1}} \tag{10}$$

for an impulsive input,

$$u(0) = 1$$

$$u(n) = 0 \qquad n \neq 0$$

$$\Rightarrow \quad U(z) = 1$$

and

$$Y(z) = \frac{1}{1-kz^{-1}} \quad . \tag{11}$$

Therefore, to find the time series, divide:

$$1-kz^{-1} \;\Big/\; \frac{1+kz^{-1}+k^2z^{-2}+k^3z^{-3}+\ldots}{1}$$

$$\frac{1-kz^{-1}}{kz^{-1}+0}$$

$$\frac{kz^{-1}-k^2z^{-2}}{k^2z^{-2}+0}$$

$$\frac{k^2z^{-2}-k^3z^{-3}}{k^3z^{-3}}$$

$$\vdots \qquad \ddots$$

The quotient, $1 + kz^{-1} + k^2z^{-2} + k^3z^{-3} + \ldots$ is $Y(z)$ which means that,

$$\begin{aligned} y(0) &= 1 \\ y(1) &= k \\ y(2) &= k^2 \\ &\vdots \\ y(n) &= k^n \end{aligned}$$

(c) Relationship Between s and z - For continuous systems, one often associates certain behavior with different values of the complex variable s. Figure 2 (from Cannon [4]) shows this correspondence. The same kind of association is also useful to designers for discrete systems. The equivalent characteristics in the z-plane are related to those in the s-plane by the expression

$$z = e^{sT} \qquad (12)$$

where T = sample period. This is obtained by comparing the z-transform of the sampled version of a signal with the Laplace transform of the signal itself. A table of z-transforms which also includes the Laplace transforms, such as in [3], demonstrates the $z = e^{sT}$ relationship in the denominators of all the table entries.

Figure 3 shows the mapping of lines of constant damping, ζ, and natural frequency, ω_n, from the s-plane to the upper half of the z-plane using Eq. (12). The mapping has several important features:

1) The stability boundary is the unit circle, $|z| = 1$.
2) The small vicinity around $z = +1$ is essentially identical to the vicinity around $s = 0$.
3) z-plane locations give response information normalized to the sample rate, rather than respect to time as in the s-plane.
4) The negative real z axis always represents a frequency of $\omega_s/2$, where ω_s = sample rate.
5) Vertical lines in the left hand s-plane (constant real part or time constant) map into circles within the unit circle.
6) Horizontal lines in the s-plane (constant imaginary part or frequency) map into radial lines in the z-plane.
7) There is no location in the z-plane that represents frequencies greater than $\omega_s/2$. Physically, this is because you must sample at least twice as fast as a signal's frequency to represent it digitally and mathematically because of the nature of the trig functions imbedded in Eq. (12).

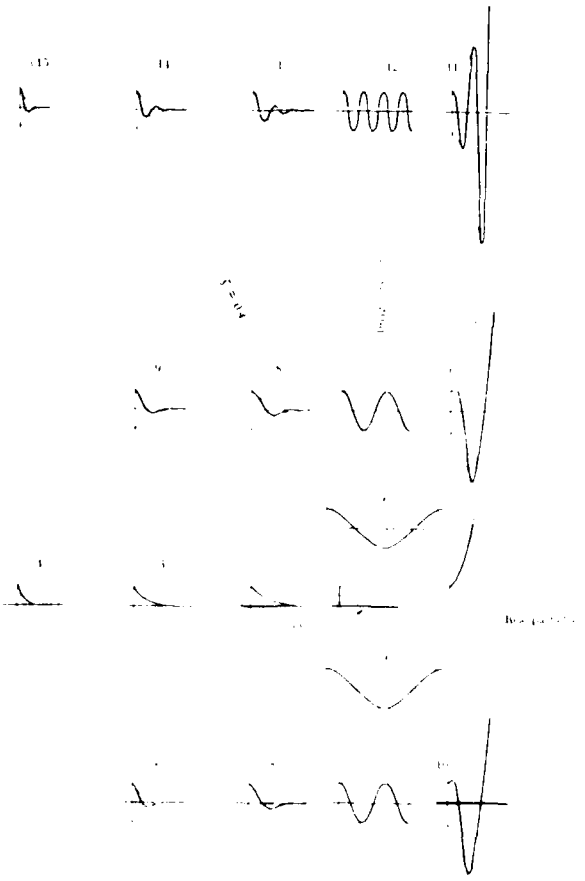


Fig. 2 Continuous System Dynamic Behavior vs. s-Plane Pole Locations (from [4]).

(d) Final Value Theorem - The final value theorem for continuous systems

$$\lim_{t \to \infty} x(t) = \lim_{s \to 0} sX(s) \qquad (13)$$

is often used to find steady state system errors and or steady state gains of portions of a control system. The analog for discrete systems is obtained by noting that a continuous steady response is denoted by $X(s) = A/s$ and leads to the multiplication by s in Eq. (13). Therefore, since the steady response for discrete systems is $X(z) = \frac{A}{1-z^{-1}}$, the discrete final value theorem is:

$$\lim_{n \to \infty} x(n) = \lim_{z \to 1} (1-z^{-1})X(z) \qquad (14)$$

For example, to find the DC gain of the transfer function, $G(z) = \frac{X(z)}{U(z)} = \frac{.8z(1+z)}{z - .16}$, let $u(n) = 1$ for

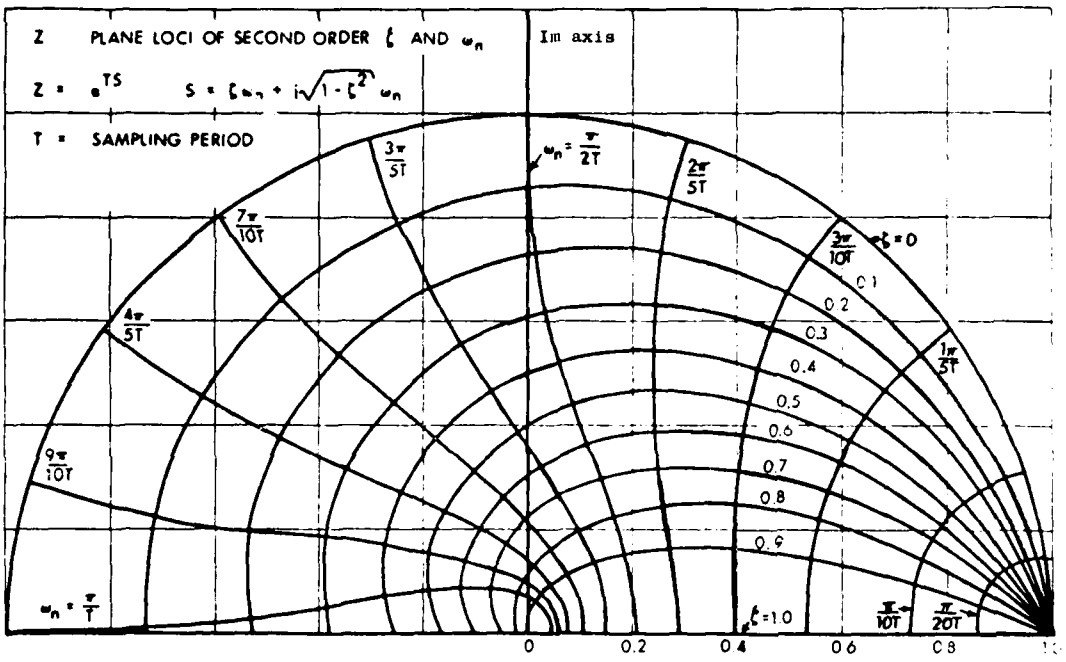


Fig. 3 Natural Frequency and Damping Loci in z-Plane (Lower half is the mirror image of that shown).

$n \geq 0$, so that

$$U(z) = \frac{1}{1-z^{-1}}$$

and

$$X(z) = \frac{.58(1+z)}{(1-z^{-1})(z+.16)} .$$

Applying the final value theorem yields

$$x(\infty) = \lim_{z \to 1} \left[\frac{.58(1+z)}{z + .16} \right] = 1$$

and therefore, the DC gain of $G(z)$ is unity. In general, we see that to find the DC gain of any transfer function, simply substitute $z = 1$ and compute the resulting gain.

Since the gain of systems does not change whether represented continuously or discretely, this calculation is an excellent check on the calculations associated with determining the discrete model of a system.

2. CONTINUOUS DESIGN

The first part of this design procedure should be already familiar to the reader; that is, the design of feedback control compensation for a continuous system. This design is carried out as if the system were continuous and no changes are required to represent the fact that the control will eventually be implemented digitally.

(a) Digitization Procedures - The second part of the procedure is to digitize the resulting compensation. Therefore, the problem to be addressed is: Given a $D(s)$, find the best equivalent $D(z)$. Or more exactly, given a $D(s)$ from the control system shown in Fig. 4, find the best digital implementation of that compensation. A digital implementation requires that y is sampled at some sample rate and that the computer

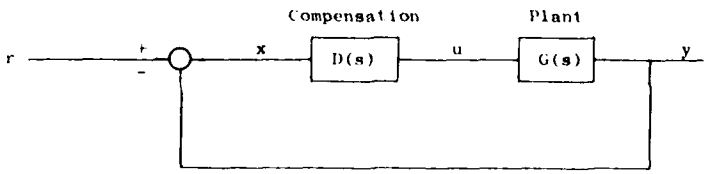


Fig. 4 Continuous Control System

output samples are smoothed in some manner so as to provide a continuous u. For ease of hardware design, the smoothing operation is almost always a simple hold (or zero order hold, "ZOH") which is shown in Fig. 5.

Therefore, we can restate the problem as: Find the best $D(z)$ in the digital implementation shown in

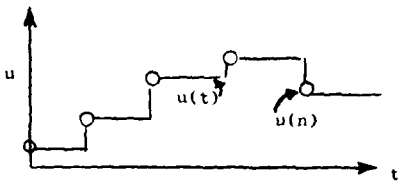


Fig. 5 Zero Order Hold.

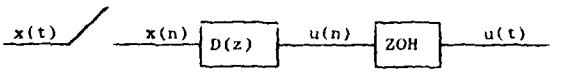


Fig. 6 Digital Compensation Implementation.

Fig. 6* to match a desired D(s). It is important to note at the outset that there is no exact solution to this problem because D(s) responds to the complete time history of x(t) whereas D(z) only has access to the samples, x(n). In a sense, the various digitization approximations (and approximations they are) simply make different assumptions about what happens to x(t) between the sample points.

Tustin's Method: One digitization method is tc approach the problem as one of numerical integration. Suppose

$$\frac{u(s)}{x(s)} = D(s) = \frac{1}{s}, \quad \text{i.e., pure integration;}$$

Therefore,

$$u(nT) = \int_0^{nT-T} x(t)dt + \int_{nT-T}^{T} x(t)dt \qquad (15)$$

$$= u(nT-T) + (\text{area under } x(t) \text{ over last } T)$$

where T = sample period, u(nT) is usually written u(n) for short and the task at each step is to use trapezoidal integration, i.e., approximate x(t) by a straight line between the two samples. Therefore Eq. (15) becomes:

$$u(nT) = u(nT-T) + \frac{T}{2}\left[x(nT-T)+x(nT)\right] \qquad (16)$$

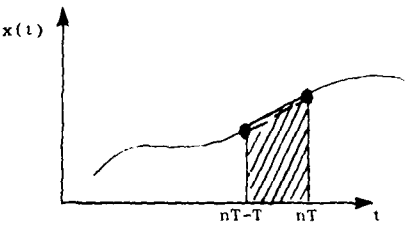


Fig. 7 Trapezoidal Integration.

or taking the z-transform,

$$\frac{u(z)}{x(z)} = \frac{T}{2}\frac{1+z^{-1}}{1-z^{-1}} = \frac{1}{\frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}} \qquad (17)$$

For

$$D(s) = \frac{a}{s+a}$$

application of the same integration approximation yields

$$D(z) = \frac{a}{\frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}} + a}$$

and, in fact, the substitution

$$s \rightarrow \frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}} \qquad (18)$$

in any D(s) yields a D(z) based on the trapezoidal integration formula. This is called "Tustin's" or the "Bilinear" approximation.

Matched Pole Zero Method: Another digitization method, called the "matched pole-zero" is found by extrapolation of the relation between the s and z planes stated in Eq. (12). If we take the z-transform of a sampled x(t), then the poles of x(z) are related to the poles of x(s) according to $z = e^{sT}$. However, we must go through the z-transform process to locate the zeros of x(z). The idea of the matched pole-zero technique is to apply $z = e^{sT}$ to the poles and zeros of a transfer function. Since physical systems often have more poles than zeros, it is also useful to arbitrarily add zeros of D(z) at $z = -1$ (i.e., a $(1+z^{-1})$ term) which causes an averaging of the current and past input values as in the trapezoidal integration (Tustin's) method. The gain is selected so that the low frequency gain of D(s) and D(z) match one another. The method summarized is:

1) Map poles and zeros according to $z = e^{sT}$.
2) Add $(1+z^{-1})$ or $(1+z^{-1})^2$, etc., if numerator is lower order than the denominator.
3) Match DC or low frequency gain.

*This figure assumes for simplicity that the command input, r, is entered outside the computer when in reality it is normally entered inside as shown in Fig. 1. This will have no impact on the design methods.

For example, the matched pole-zero approximation of

$$D(s) = \frac{s+a}{s+b} \quad is \quad D(z) = k\,\frac{z - e^{-aT}}{z - e^{bT}} \tag{19}$$

where $k = \frac{a}{b}\,\frac{1-e^{-bT}}{1-e^{-aT}}$ and for

$$D(s) = \frac{s+a}{s(s+b)} \Rightarrow D(z) = k\,\frac{(z+1)(z-e^{-aT})}{(z-1)(z-e^{-bT})} \tag{20}$$

where $k = \frac{a}{2b}\,\frac{1-e^{-bT}}{1-e^{-aT}}$.

In both digitization methods, the fact that an equal power of z appears in numerator and denominator of $D(z)$ implies that the difference equation output at time n will require a sample of the input at time n. For example, the $D(z)$ in Eq. (19) can be written

$$\frac{u(z)}{x(z)} = D(z) = k\,\frac{1 - \alpha z^{-1}}{1 - \beta z^{-1}}$$

which results in the difference equation,

$$u(n) = \beta\, u(n-1) + k[x(n) - \alpha\, x(n-1)] \quad . \tag{21}$$

The $D(z)$ in Eq. (20) would also result in $u(n)$ being dependent on $x(n)$, the input at the same time point. If the structure of the computer hardware prohibits this relation or if the computations are particularly lengthy thus rendering Eq. (21) impossible to implement, it may be desirable to arrive at a $D(z)$ which has one less power of z in the numerator than denominator and hence the computer output, $u(n)$, only requires input from the previous time, i.e., $x(n-1)$. To do this, we simply omit step (2) in the Matched Pole-Zero procedure. The second example

$$D(s) = \frac{s+a}{s(s+b)}$$

would then become

$$D(z) = k\,\frac{z-e^{-aT}}{(z-1)(z-e^{-bT})} \quad \text{where} \quad k = \frac{a}{b}\,\frac{1-e^{-bT}}{1-e^{-aT}}$$

which results in

$$u(n) = (1+e^{-bT})u(n-1) - e^{-bT}u(n-2) + k[x(n-1) - e^{-aT}x(n-2)] \quad .$$

Method Comparison: *A numerical comparison of the magnitude of the frequency response is made in* Fig. 8 for the three approximation techniques at two sample rates. *The results of the* $D(z)$ *computations* used in arriving at Fig. 8 are shown in Table I.

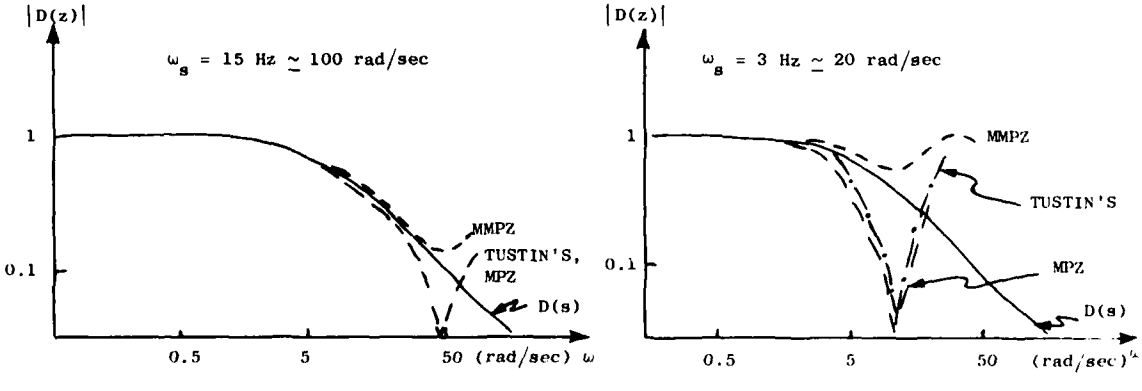


Fig. 8 Comparison of Discrete Approximations.

The figure shows that all the approximations are quite good at frequencies below about 1/4 the sample rate, $\omega_s/4$. If $\omega_s/4$ is sufficiently larger than the filter break frequency, i.e., if the sampling is fast enough, the break characteristics are accurately reproduced. Tustin's and the MPZ show a notch at $\omega_s/2$ due to their zero term, $(z+1)$. Other than the large difference at $\omega_s/2$ which is typically outside the range of interest, the three methods have similar accuracies. Since the MPZ techniques require much simpler algebra than Tustin's, they are typically preferred.

TABLE I DIGITAL APPROXIMATIONS

	D(z) for $D(s) = \frac{5}{s+5}$	
	ω_s = 15 Hz	ω_s = 3 Hz
Matched Pole-Zero (MPZ)	$.143 \frac{z+1}{z-.715}$	$.405 \frac{z+1}{z-.189}$
Modified MPZ (MMPZ)	$.285 \frac{1}{z-.715}$	$.811 \frac{1}{z-.189}$
Tustin's	$.143 \frac{z+1}{z-.713}$	$.454 \frac{z+1}{z-.0914}$

(b) Design Example - For a $1/s^2$ plant, we wish to design a digital controller to have a closed-loop natural frequency, ω_n, of ~ 0.3 rad/sec and $\zeta = 0.7$. The first step is to find the proper D(s) defined in Fig. 9. The specifications can be met with

$$D(s) = k \frac{s+a}{s+b} \qquad (22)$$

where

$a = 0.2$

$b = 2.0$

$k = 0.81$

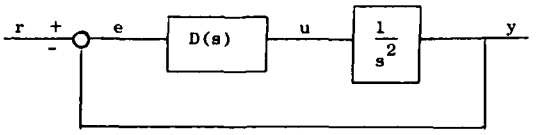


Fig. 9 Continuous Design Statement.

as can be verified by the root locus in Fig. 10. To digitize this D(s), we first need to select a sample rate. For a system with $\omega_n = 0.3$ rad/sec, a very "safe" sample rate would be a factor of 20 faster than ω_n, yielding

$$\omega_s = 0.3 \times 20 = 6 \text{ rad/sec} .$$

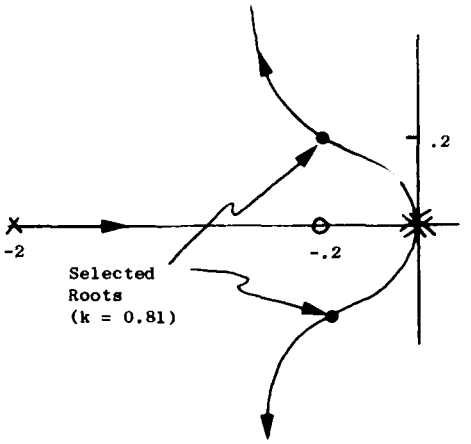


Fig. 10 s-Plane Locus vs. k.

Thus, let's pick T = 1 sec. The matched Pole-Zero digitization of Eq. (22) is given by Eq. (19) and yields

$$D(z) = (0.389) \frac{z - 0.82}{z - 0.135} \qquad (23)$$

or

$$D(z) = \frac{0.389(1 - 0.82\, z^{-1})}{1 - 0.135\, z^{-1}}$$

which leads to

$$u(n) = 0.135u(n-1)+0.389e(n)-0.319e(n-1) \qquad (24)$$

where

$$e(n) = r(n) - y(n)$$

and completes the digital design. The complete digital system is shown in Fig. 11.

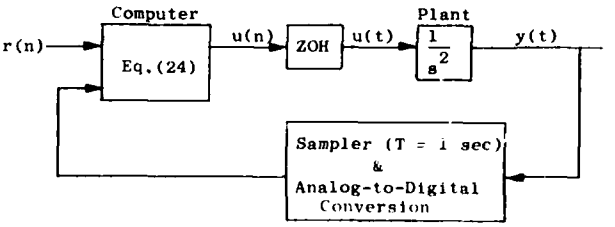


Fig. 11 Digital Control System.

(c) Discussion - If an exact discrete analysis of the design was performed and the digitization was determined for a wide range of sample rates, the system would be unstable for sample rates slower than approximately $5 \times \omega_n$ and the damping would be substantially degraded for sample rates slower than $10 \times \omega_n$. At sample rates on the order of $20 \times \omega_n$ (or 20 x bandwidth for more complex systems), this design method can be used with confidence.

Basically, the errors come about because the technique ignores the lagging effect of the ZOH. An approximate method to account for this is to assume that the transfer function of the ZOH is:

$$G_{ZOH}(s) = \frac{2/T}{s + 2/T} \qquad (25)$$

This is based on the idea that, on the average, the hold delays by $T/2$ and the above is a first order lag with a time constant of $T/2$, DC gain = 1. We could therefore patch-up the original D(s) design by inserting this $G_{ZOH}(s)$ in the original plant model and finding the D(s) that yields satisfactory response.

One of the advantages of using this design method, however, is that the sample rate need not be selected until after the basic feedback design is completed. Therefore the modification eliminates this advantage, although it does partially alleviate the approximate nature of the method, which is the primary disadvantage.

3. DISCRETE DESIGN

(a) Analysis Tools - The first step in performing a control design or analysis of a system with some discrete elements in it is to find the discrete transfer function of the continuous portion. For a system similar to that shown in Fig. 1, we wish to find the transfer function between u(n) and y(n). Unlike the previous section, there is an exact discrete equivalent for this system because the ZOH precisely describes what happens between samples and the output, y(n), is only dependent on the input at the sample times, u(n).

For a plant described by a G(s) and preceded by a ZOH, the discrete transfer function is:

$$G(z) = (1-z^{-1})Z\left\{\frac{G(s)}{s}\right\} \tag{26}$$

where $Z\{F(s)\}$ means the z-transform of the time series whose Laplace transform is F(s), i.e., the same line in the tables. The formula has the term G(s)/s because the control comes in as a step input during each sample period. The term $(1-z^{-1})$ is there because a one-sample duration step can be thought of as an infinite duration step one cycle delayed. A complete derivation can be found in [3]. This formula (Eq. 26) allows us to replace the mixed (continuous and discrete) system shown in Fig. 12a with the pure discrete equivalent system shown in Fig. 12b.

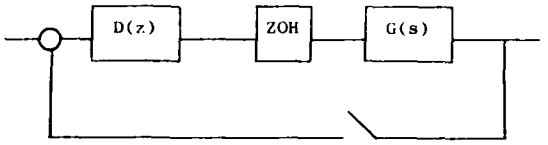


Fig. 12a Mixed Control System.

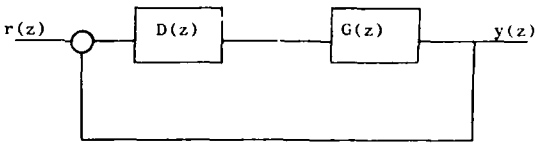


Fig. 12b Pure Discrete Equivalent.

The analysis and design of discrete systems is very similar to continuous ones; in fact, all the same rules apply. The closed-loop transfer function of Fig. 12b is obtained using the same rules of block diagram reduction, i.e.,

$$\frac{y(z)}{r(z)} = \frac{DG}{1+DG} \quad . \tag{27}$$

Since we'd like to find the characteristic behavior of the closed-loop system, we wish to find the factors of the denominator of Eq. 27, i.e., find the roots of the characteristic equation:

$$1 + D(z)G(z) = 0 \quad . \tag{28}$$

The root locus techniques used in continuous systems to find roots of a polynomial in s apply equally well here for the polynomial in z. The rules apply directly without modification; however, the interpretation of the results is quite different as we saw in Fig. 3. A major difference is that the stability boundary is now the unit circle instead of the imaginary axis.

A simple example of the discrete design tools discussed so far should help fix ideas. Suppose G(s) in Fig. 12a is:

$$G(s) = \frac{a}{s+a} \quad .$$

It follows from Eq. (26) that

$$G(z) = (1-z^{-1})Z\ \frac{a}{s(s+a)}$$

$$= (1-z^{-1})\left[\frac{(1-e^{-aT})z^{-1}}{(1-z^{-1})(1-e^{-aT}z^{-1})}\right]$$

$$G(z) = \frac{(1-\alpha)}{z-\alpha} \tag{29}$$

where

$$\alpha = e^{-aT} \quad .$$

To analyze the performance of a closed-loop proportional control law, i.e., D(z) = k, we use standard root locus rules. The result is shown in Fig. 13a and for comparison, the root locus for a continuous controller is shown in Fig. 13b. In contrast to the continuous case which remains stable for all values of k, the discrete case becomes oscillatory with a decreasing damping ratio as z goes from 0 to -1 and eventually becomes unstable. This instability is due to the lagging effect of the ZOH which is properly accounted for

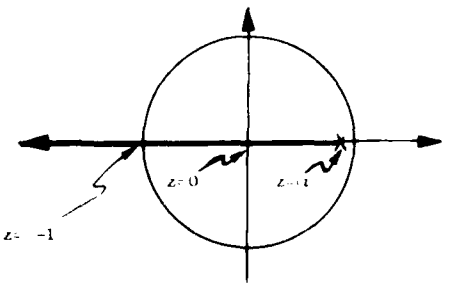


Fig. 13a z-Plane Root Locus.

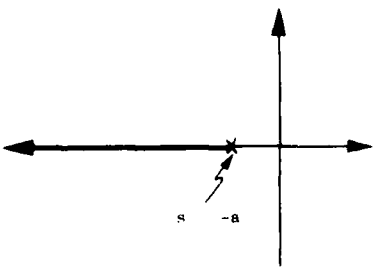


Fig. 13b s-Plane Root Locus.

in the discrete analysis.

(b) Design - In continuous systems, we typically start the design process by using proportional, derivative, or integral control laws or combinations of these, sometimes with a lag included. The same ideas are used in discrete designs directly or perhaps the $D(z)$ that results from the digitization of a continuously designed $D(s)$ is used as a starting point.

The discrete control laws are:

-- Proportional:

$$u(n) = k_p e(n)$$

$$\Rightarrow D(z) = k_p \quad . \tag{30}$$

-- Derivative:

$$u(n) = k_D[e(n)-e(n-1)]$$

$$\Rightarrow D(z) = k_D(1-z^{-1}) = k_D \frac{z-1}{z} \quad . \tag{31}$$

-- Integral:

$$u(n) = u(n-1) + k_I e(n)$$

$$\Rightarrow D(z) = \frac{k_I}{1-z^{-1}} = \frac{k_I z}{z-1} \quad . \tag{32}$$

For an example, let's use the same problem as we used for the continuous design; the $1/s^2$ plant. Using Eq. (26), we have

$$G(z) = \frac{T^2}{2} \frac{z+1}{(z-1)^2} \tag{33}$$

which becomes with $T = 1$ sec,

$$G(z) = \frac{1}{2} \frac{z+1}{(z-1)^2} \quad . \tag{34}$$

Proportional feedback in the continuous case yields pure oscillatory motion and in the discrete case, we should expect even worse results. The root locus in Fig. 14 verifies this. For very low values of k (very low frequencies compared to the sample rate) the locus is tangent to the unit circle ($\zeta \simeq 0$ and pure oscillatory motion) thus matching the proportional continuous design. For higher values of k, the locus diverges into the unstable region due to the effect of the ZOH and sampling.

To compensate for this, let's add a velocity term to the control law, or

$$u(z) = k[1 + \gamma(1-z^{-1})]e(z) \tag{35}$$

which yields

$$D(z) = k(1+\gamma) \frac{z - \frac{\gamma}{1+\gamma}}{z} \quad . \tag{36}$$

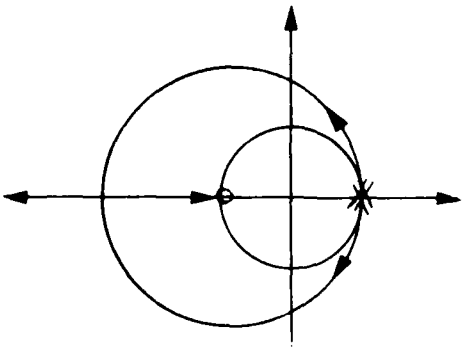


Fig. 14 z-Plane Locus for $1/s^2$ Plant.

Now the task is to find values of γ and k that yield good performance. When we did this design previously, we wanted $\omega_n \simeq 0.3$ rad/sec and $\zeta = 0.7$. Figure 3 indicates that this s-plane root location maps into a z-plane location of:

$$z \simeq 0.8 \pm 1.7j \quad .$$

Figure 15 shows that for $\gamma = 1$ and $k = 0.08$ or

$$D(z) = 0.4 \frac{z-0.8}{z} \tag{37}$$

the roots are at the desired location. Normally, it is not particularly advantageous to match specific z-plane root locations, rather it is only necessary to pick k and γ to obtain acceptable z-plane roots, a much easier task. In this example, we wanted to match a specific location only so we could compare the result with the previous design.

The control law that results is

$$u(z) = 0.08[1 + 4(1-z^{-1})]$$

or

$$u(n) = 0.4e(n)-0.32e(n-1) \qquad (38)$$

which basically only differs from the continuously designed controller, Eq. (24), by the absence of the u(n-1) term. The u(n-1) term in Eq. (24) resulted from the lag term, (s+b), in the compensation, Eq. (22), which is typically included in analog controllers because of the difficulty in building pure analog differentiators and for noise attenuation. The equivalent lag in discrete design naturally appears as a pole at z = 0 (see Fig. 15) and represents the one sample delay in computing the derivative by a first difference. For more noise attenuation, the pole could be moved to the right of z = 0, thus resulting in less derivative action and more smoothing, the same tradeoff that exists in continuous control design.

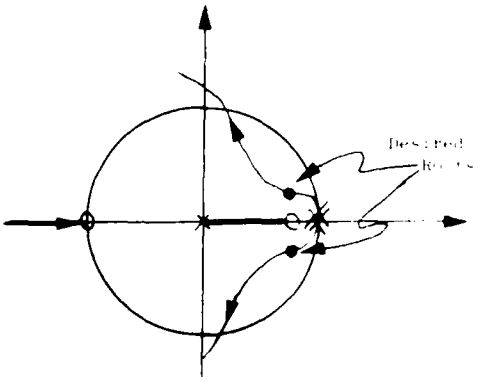


Fig. 15 Compensated z-Plane Locus for 1 s^2 Example.

Other than the u(n-1) term, the two controllers are very similar (Eq. (24) and Eq. (38)). This similarity resulted because the sample rate is fairly fast compared to ω_n, i.e., $\omega_s = 20 \times \omega_n$. For designs at slower sample rates, the numerical values in the compensations would become increasingly different as the sample rate decreased. For the discrete design, the actual system response would follow that indicated by the z-plane root locations, while the continuously designed system response would diverge from that indicated by the s-plane root locations.

As a general rule, discrete design should be used if sampling slower than $10 \times \omega_n$. At the very least, a continuous design with slow sampling ($\omega_s < 10\omega_n$) should be verified by a discrete analysis or simulation and the compensation adjusted if needed.

4. WORD SIZE EFFECTS

A numerical value can only be represented with a limited precision, in a digital computer. For fixed point arithmetic, the resolution is 0.5% of full range for 8 bits and 0.1% for 10 bits and drops by a factor of two for each additional bit. The effect of this limited precision shows up in the analog-to-digital (A/D) conversion which often has a smaller word size than the computer, multiplication truncation, and parameter storage errors. If the computer uses floating point arithmetic, the resolution of the multiplication and parameter storage changes with the magnitude of the number being stored, the resolution only affecting the mantissa while the exponent essentially continually adjusts the full scale.

(a) Random Effects - As long as a system has varying inputs or disturbances, A/D errors and multiplication errors act in a random manner on the system and essentially produce noise at the output of the system. The output noise due to a particular noise source (A/D or multiplication) has a mean value of

$$\bar{n}_0 = H_{DC}\bar{n}_I \qquad (39)$$

where

H_{DC} = DC gain of transfer function between noise source and output;

$\bar{n}_I$ = mean value of noise source.

The mean value of the noise source will be zero for a round-off process but has a value

$$\bar{n}_I = q/2$$

where q resolution level for a truncation process. Although most A/D's round-off producing no mean error, some truncate producing an error. The total noise effect is the sum of all noise sources.

The variance of the output noise is always nonzero irrespective of whether the process truncates or rounds. It is [3]:

$$\sigma_0^2 = \sigma_I^2 \frac{1}{2\pi j} \oint H(z)H(z^{-1}) \frac{dz}{z} \qquad (40)$$

where

σ_0 output noise variance;

σ_I input noise variance;

H(z) transfer function between noise input and system output.

The input noise variance has magnitude

$$\sigma_I^2 = q^2/12 \qquad (41)$$

for either round-off or truncation.

In general, evaluation of noise response using Eq. (40) would show that the discrete portion of the controller becomes more sensitive to noise as the sampling rate increases. However, this trend with sampling rate is partly counter-balanced by the decreasing total system response due to the increasing noise frequency.

The sensitivity of a system to A/D errors can be partly alleviated by adding lag in the digital controller or by adding more bits to the A/D converter. Different structures of the digital controller have no effect.

On the other hand, multiplication errors can be reduced substantially for high order ($\geq$ 2nd order) controllers by proper structuring of a given control transfer function. For example, a 2nd order transfer function with real roots:

$$D(z) = \frac{u(z)}{e(z)} = \frac{z - 0.8}{(z-0.2)(z-0.3)} \qquad (42)$$

can be implemented in a "direct" manner yielding

$$u(n) = 0.5u(n-1)-0.06u(n-2)+e(n)-0.8e(n-1) \qquad (43)$$

or could be implemented in a "parallel" manner that results from a partial fraction expansion of Eq. (42). The result is shown in Fig. 16 and yields:

$$\begin{aligned} x_1(n) &= 0.2x_1(n-1) + 6e(n) \\ x_2(n) &= 0.3x_2(n-1) + 5e(n) \\ u(n) &= x_1(n) - x_2(n) \end{aligned} \qquad (44)$$

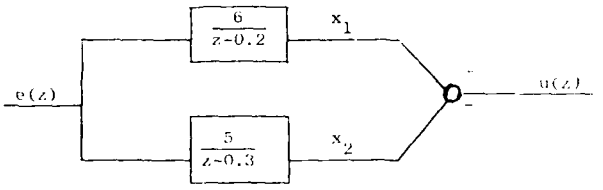


Fig. 16 Parallel Implementation.

Note that the transfer functions to the output from the multiplications in Eq. (43) are substantially different than from those in Eq. (44). It is also possible to implement the D(z) with a "cascade" factorization which would be two 1st order blocks arranged serially for this example.

Either cascade or parallel implementations are preferred to the direct implementation and for transfer functions higher than 3rd order they are almost mandatory.

(b) Systematic Effects - Parameters such as the numerical values in Eqs. (43) and (44), if in error, will change dynamic behavior of a system. In a high order controller with a direct implementation a very small percentage error in a stored parameter can result in substantial root location changes and sometimes cause instability. In the Apollo Command Module, a 14-bit word size for parameter storage would have resulted in instability if a direct implementation had been used in the 6th order compensator! These effects are amplified when there are two compensator poles close together (or repeated) and at fast sample rates where all poles tend to clump around $z = +1$ and are all close together.

These effects can typically be managed by using parallel or cascade implementations. In some situations it may also be necessary to use a double precision parameter storage.

Under conditions of constant disturbances and input commands, multiplication errors can also cause systematic errors. Typically, the result is a steady state error or possibly a stable limit cycle. The steady state error results from a deadband that exists in any digital controller whose magnitude is proportional to the DC gain from the multiplication error to the output and to the resolution level. Stable limit cycles only occur for controllers with lightly damped poles.

5. SAMPLE RATE SELECTION

The selection of the best sample rate for a digital control system is a compromise among many factors. The basic motivation to lower the sample rate (ω_s) is cost. A decrease in sample rate means more time is available for the control calculations, hence slower computers are possible for a given control function or more control capability is available for a given computer. Either result lowers the cost per function. For systems with A/D converters, less demand on conversion speed will also lower cost. These economic arguments indicate that the best engineering choice is the slowest possible sample rate which meets all performance specifications.

Factors which could provide a lower limit to the acceptable sample rate are: (1) tracking effectiveness as measured by closed-loop bandwidth or by time response requirements, such as rise time and settling time; (2) regulation effectiveness as measured by the error response to random plant disturbances; (3) sensitivity to plant parameter variations; and (4) error due to measurement noise and the associated prefilter design methods. A fictitious limit occurs when using continuous design techniques. The inherent approximation in

the method may give rise to system instabilities as sample rate is lowered and this can lead the designer to conclude that a lower limit on ω_s has been reached when in fact the proper conclusion is that the approximations are invalid; the solution is not to sample faster but to switch to the discrete design method.

(a) Tracking Effectiveness - An absolute lower bound to the sample rate is set by a specification to track a command input with a certain frequency (the system bandwidth). The "sampling theorem" [3] states that in order to reconstruct an unknown band-limited continuous signal from samples of that signal, one must sample at least twice as fast as the highest frequency contained in the signal. Therefore, in order for a closed-loop system to track an input at a certain frequency it must have a sample rate twice as fast, i.e., ω_s must be at least twice the system bandwidth ($\omega_s = 2 \times \omega_{BW}$). We also saw from the z-plane mapping, $z = e^{sT}$, that the highest frequency that can be represented by a discrete system is $\omega_s/2$, supporting the conclusion above.

It is important to note the distinction between the closed-loop bandwidth, ω_{BW}, and the highest frequencies in the open-loop plant dynamics since these two frequencies can be quite different. For example, closed-loop bandwidths could be an order of magnitude less than open-loop modes of resonances for some vehicle control problems. Information concerning the state of the plant resonances for purposes of control can be extracted from sampling the output without satisfying the sampling theorem because some a priori knowledge is available (albeit imprecise) concerning these dynamics and the system is not required to track these frequencies. Thus a priori knowledge of the dynamic model of the plant can be included in the compensation in the form of a notch filter.

The "closed-loop bandwidth" limitation provides the fundamental lower bound on the sample rate. In practice, however, the theoretical lower bound of sampling at twice the bandwidth of the reference input signal would not be judged sufficient in terms of the quality of the desired time responses. For a system with a rise time on the order of 1 second and a required closed-loop bandwidth on the order of 0.5 Hz it is not unreasonable to insist on a sample rate of 2 to 10 Hz, which is a factor of 4 to 20 times ω_{BW}. This is so in order to reduce the delay between a command and the system response to the command and to smooth the system output response to the control steps coming out of the ZOH.

(b) Disturbance Rejection - Disturbance rejection is an important aspect of any control system if not the most important one. Disturbances enter a system with various characteristics ranging from steps to white noise. For purposes of sample rate determination, the higher frequency random disturbances are the most influential.

The ability of the control system to reject disturbances with a good continuous controller represents a lower bound on the error response that can be hoped for when implementing the controller digitally. In fact, some degradation over the continuous design must occur due to the sampled values being slightly out of date at all times except precisely at the sampling instants. However, if the sample rate is very fast compared to the frequencies contained in the noisy disturbance, no appreciable loss should be expected from the digital system compared with the continuous controller. At the other extreme, if the sample time is very long compared with the characteristic frequencies of the noise, the response of the system due to noise is essentially the same as could result if there were no control at all. The selection of a sample rate will place the response somewhere in between these two extremes and thus the impact of sample rate on the disturbance rejection of the system may be very influential to the designer in selecting the sample rate.

Although the best choice of sample rate in terms of the ω_{BW} multiple is dependent on the frequency characteristics of the noise and the degree to which random disturbance rejection is important to the quality of the controller, sample rate requirements of 10 or 20 times ω_{BW} are not uncommon.

(c) Parameter Sensitivity - Any control design relies to some extent on knowledge of the parameters representing plant dynamics. Discrete systems exhibit an increasing sensitivity to parameter errors for a decreasing ω_s when the sample interval becomes comparable to the period of any of the open-loop vehicle dynamics. For systems with all plant dynamics in the vicinity of the closed-loop bandwidth or slower, root location changes due to parameter errors would not likely be a constraining factor unless the parameter error was quite large. However, for systems with a structural (or other) resonance which is stabilized by a notch filter, imperfect knowledge of the plant resonances characteristics will lead to changes in the system roots and possibly instabilities. This sensitivity to plant parameters increases as the sample rate decreases and could limit how slow the sample rate is in some cases. Typically, however, some other factor limits the sample rate and, at worst, some effort needs to be made in the controller design to minimize its sensitivity.

(d) Effect of Prefilter Design - Digital control systems with analog sensors typically include an analog prefilter between the sensor and the sampler or A/D converter as an anti-aliasing device. The prefilters are low pass, and the simplest transfer function is

$$G_p(s) \qquad \frac{a}{s+a} \tag{45}$$

so that the noise above the prefilter breakpoint, a, is attenuated. The design goal is to provide enough attenuation at half the sample rate ($\omega_s/2$) so that the noise above $\omega_s/2$, when aliased into lower frequencies by the sampler, will not be detrimental to the control system performance.

A conservative design procedure is to select the breakpoint and ω_s sufficiently higher than the system bandwidth so that the phase lag from the prefilter does not significantly alter the system stability and thus the prefilter can be ignored in the basic control system design. Furthermore, for a good reduction in the high frequency noise at $\omega_s/2$, the sample rate should be selected about 5 or 10 times higher than the prefilter breakpoint. The implication of this prefilter design procedure is that sample rates need to be on the order of 20 to 100 times faster than the system bandwidth. If done this way, the prefilter design procedure is likely to provide the lower bound on the selection of the sample rate.

An alternate design procedure is to allow significant phase lag from the prefilter at the system bandwidth and thus to require that the control design be carried out with the analog prefilter characteristics

included. This procedure allows us to use very low sample rates, but at the expense of increased complexity in the original design since the prefilter must be included in the plant transfer function. Using this procedure and allowing low prefilter breakpoints, the effect of sample rate on sensor noise is small and essentially places no limits on the sample rate.

6. REFERENCES

[1] Dorf, R C., Modern Control Systems, Addison-Wesley, 1980.

[2] Ogata, K., Modern Control Engineering, Prentice-Hall, 1970.

[3] Franklin, G.F. and Powell, J.D., Digital Control of Dynamic Systems, Addison-Wesley, 1980.

[4] Cannon, R.H., Dynamics of Physical Systems, McGraw-Hill, 1967.

PROPULSION CONTROL SYSTEM COMPUTER SOFTWARE DEVELOPMENT AND MANAGEMENT

by

R. J. MILLER
Government Products Division
Pratt & Whitney Aircraft Group
West Palm Beach, Florida

and

W. J. BARRETT
Government Products Division
Pratt & Whitney Aircraft Group
West Palm Beach, Florida

ABSTRACT

With the growing importance of the digital computer, software must be recognized as a vital and essential element to successful engine control system development and operation. This paper discusses such basic issues as pros and cons of high and low level languages and procedures, and techniques to assist in acquisition, verification and management of software for propulsion controls.

INTRODUCTION

Digital propulsion control systems are here! During 1980 each of the major U.S. engine manufacturers demonstrated new or derivative engines equipped with full authority digital electronic control systems. P&WA demonstrated this type of system on advanced versions of the F100, the F401 and the JT9D. Within the next two or three years, a digital engine control system will be certified for operation with the JT10D. From a historical perspective, electronics were first accepted as temperature-limiting devices, then as supervisory trim controls of selected variables, and now finally as full authority controls for the entire engine. This profound step in the field of propulsion controls has been propelled by the growing availability of very large scale integrated electronic devices. Microprocessors will undoubtedly form the basis of most, if not all, propulsion control systems in the future. Government and Industry research programs (Reference 1) have repeatedly projected substantial benefits for microprocessor-based control systems in terms of improved engine efficiency, performance and operations. Tangible payoffs identified in the NAPC-sponsored FADEC program are lower procurement (−41%) and operational (−43%) costs, lighter weights (−25%) and increased reliability (+130%). Major improvements are also projected for these future systems in the areas of reliability and maintainability.

The computational simplicity provided by electronic controls can be noted by comparing a section of a hydromechanical control computer to its electronic counterpart. The initial major attraction offered by digital computation was the tantalizing feature of easy changes of system characteristics via software. Furthermore, with increased computational power it is possible to implement more sophisticated control laws and self-test logic. However, these benefits are offset by the added complications encountered in designing and developing the associated software. Thus the control designer must decide to use the computational power of the microprocessor either to simplify the hardware that satisfies a given requirement or to enhance the performance of the controller.

Many new techniques have been brought forth throughout the control industry aimed at various aspects of the software development process: top-down design, structured programming, modular decomposition, validation, baselining, design confirmation — and on and on it goes. The sheer magnitude of the techniques available seems at conflict with the goal to establish some degree of software control. However, the basis of all good software management techniques can be summarized in two words: Discipline and Visibility. These two words form the basis for the systematic procedures presented here to develop and implement cost-effective and reliable propulsion control software.

The four major phases of the software development cycle are illustrated in Figure 1. The System Design and Software Requirements phase must establish and document precisely *what* the software is supposed to do. This is contrasted with the Software Design, Coding and Test phase which will establish *how* the requirements will be implemented. While there is often a tendency to press on to design and coding before a baseline requirements document has been completed and approved, the potential impact of building something on a poorly laid foundation must be recognized. Table I summarizes the relative costs of making changes in the various software development phases. An error in the requirements specifications can cost up to 100 times as much to correct once the system is in operation than if the requirement was correctly identified in the beginning (Reference 2). One reason for this severe error correction cost growth is that each error in requirements tends to proliferate into a whole family of errors in design and code. Another is that, as the program develops and information detail expands, the discovery of a given error becomes proportionally more difficult. Thus the application of intense discipline in the preparation and maintenance of the software requirements specification, directed specifically at minimizing the number of requirements errors, will pay off in reduced overall program cost. From a management perspective, this software requirements specification is important because it is the point of departure for the entire software development effort

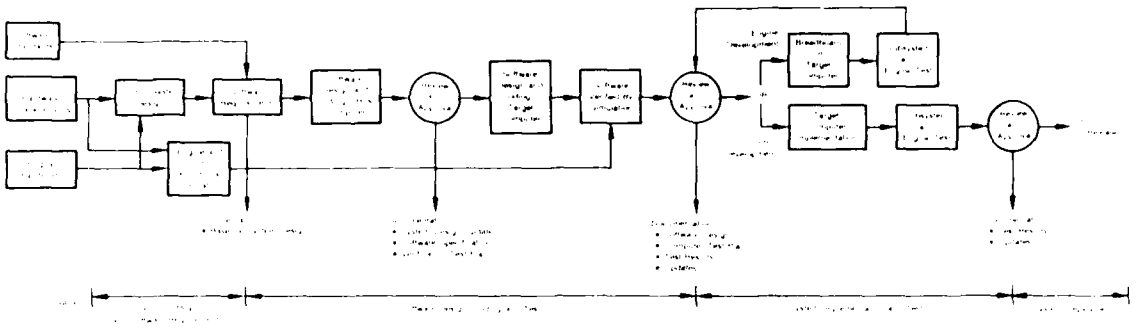


Figure 1. Software Development and Configuration Management Process

TABLE I. SOFTWARE CHANGES BECOME PROGRESSIVELY MORE EXPENSIVE

Software Status	*Relative Cost To Correct Error*
Requirements Definition	1
Design	2
Coding and Debugging	4
Subsystem Test	8
System Test	20
Operational	64-100

The Software Design, Coding and Test phase definitizes, in a logical and organized manner, the necessary functions and operations to satisfy all the software requirements. This includes all arithmetic and logic operations which must be performed. Here high level languages provide both an effective development tool and a means to document the baseline functional logic. As the system design progresses, functional simulations are a key to refining details of control system inputs, outputs, control laws and logic. In this critical phase for software development, a complete, unambiguous, testable set of requirements must be developed to avoid difficulties in subsequent activities. The software program requirements are then translated into the language of the target computer. Checkout begins by executing the program as individual or combined elements and is completed with formal tests to evaluate overall operational performance. These final tests must assure that the software performs as intended and that all system requirements are satisfied. Finally, all system components, hardware as well as software, are brought together as an integrated system where subsystem and engine tests are conducted to verify operation in the intended environment.

Successful software development is best accomplished with procedures and techniques that provide in-depth visibility of the technical development. Strategically timed check points for software document audit and approval, combined with normal review and concurrences, force an orderly development process and keep the major design issues in perspective throughout the system development process. Even after system deployment the software must be maintained, engine design modifications accommodated, and new system requirements satisfied. Techniques and procedures similar to those used during development are also needed for effective software maintenance.

SYSTEM DESIGN AND SOFTWARE REQUIREMENTS

The System Design and Software Requirements phase brings together the basic engine characteristics, the desired functional and operational features, the selected control hardware and software standards to establish the overall control system and software design requirements. The major tasks to be performed are presented in Figure 2. The system defined and documents published at the end of this phase will serve as the baseline for all future work. Baselining serves the dual purpose of allowing and controlling subsequent changes in requirements. Each change to the baseline is evaluated in terms of its overall impact and implemented via updates to the baseline published at appropriate intervals. The baseline system design includes block diagrams of the control system modes showing logic, schedules, filters, signal conditioning, output devices and multiplexing. The major interrelationships between redundancy management, mode control, and failure protection and isolation tasks are identified in this phase. The baseline software design includes software design trees, control modules, data flow charts showing module interaction, data organization, throughput timing, and test features.

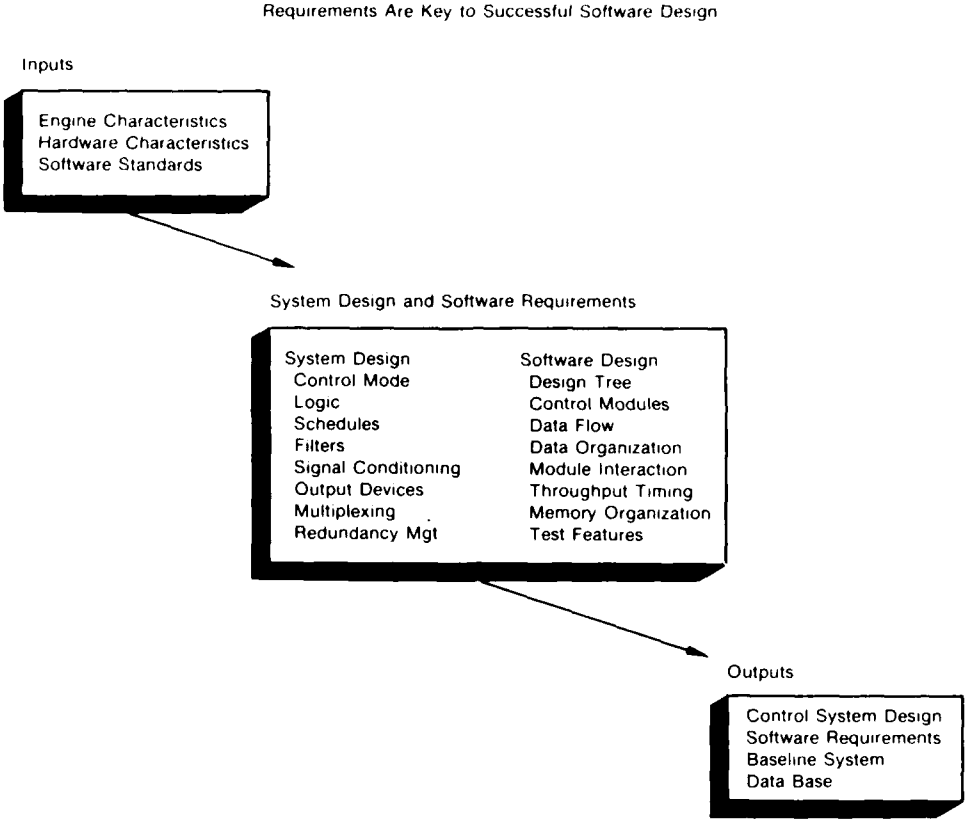


Figure 2. Clearly Stated System Design

To illustrate the engine characteristics that the control system must accommodate, consider the augmented turbofan shown in Figure 3. There are six engine variables to be set by the control: main burner fuel flow (Wfp) augmentor fuel flow (Wfa), compressor inlet variable vanes (CIVV), rear compressor variable vanes (RCVV) compressor bleeds (bld) and nozzle jet area (Aj). This advanced propulsion system operates at or near design limits requiring tight control of speed, pressure, temperature, and airflow to achieve maximum performance while maintaining engine durability. An accurate control system is required to ensure high engine performance and operational stability throughout the flight envelope. The control system must sense pilot commands, airframe requirements, and critical engine parameters; compute the necessary schedules; and actuate system variables for total engine control over the full range of operation. The desired engine functional and operational features are combined with these requirements to generate the control system modes and logic. An approximate priority list for the control mode design criteria is presented in Table II. The main burner fuel control logic to satisfy this criteria is illustrated in Figure 4. This logic diagram portrays a fan speed control with turbine temperature, burner pressure, acceleration and deceleration limits. Similar control modes and logic must be defined for each of the six engine variables.

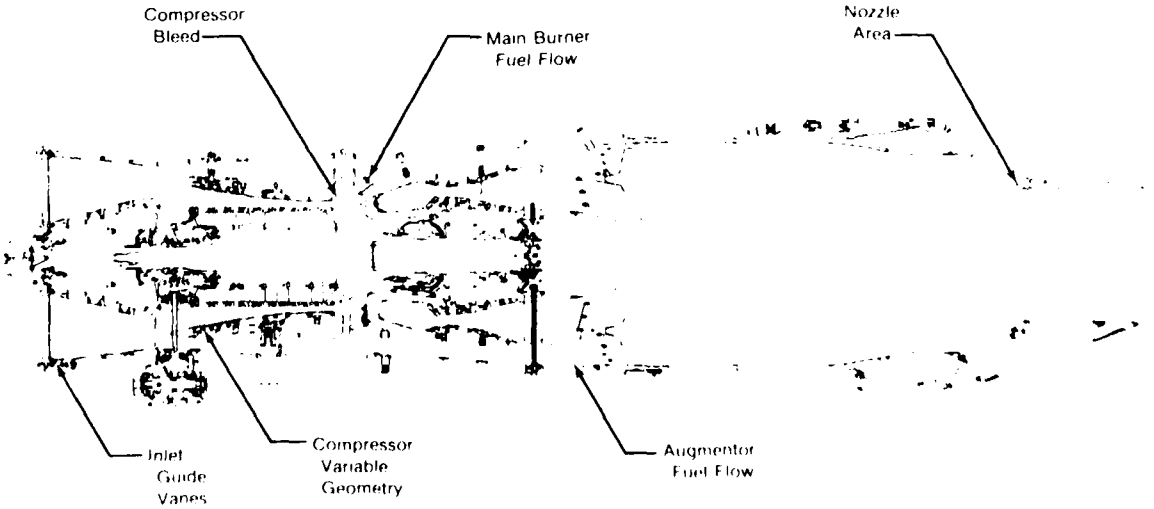


Figure 3. Typical Augmented Turbofan Engine Control Variables

TABLE II. INFORMATION NEEDED FOR DEFINITION OF CONTROL SYSTEM DESIGN

Engine Control Mode Design Criteria	*Control Hardware Characteristics*
Protection Limits	Target Computer
Stability	Actuator Dynamics and Interfaces
A/C Compatibility	Sensor Dynamics and Interfaces
Performance	Data Links
Accuracy	Power Supply
Transients	Redundancy and Reliability Requirements
Dynamics	Diagnostics
Trim	Maintainability
Starting	

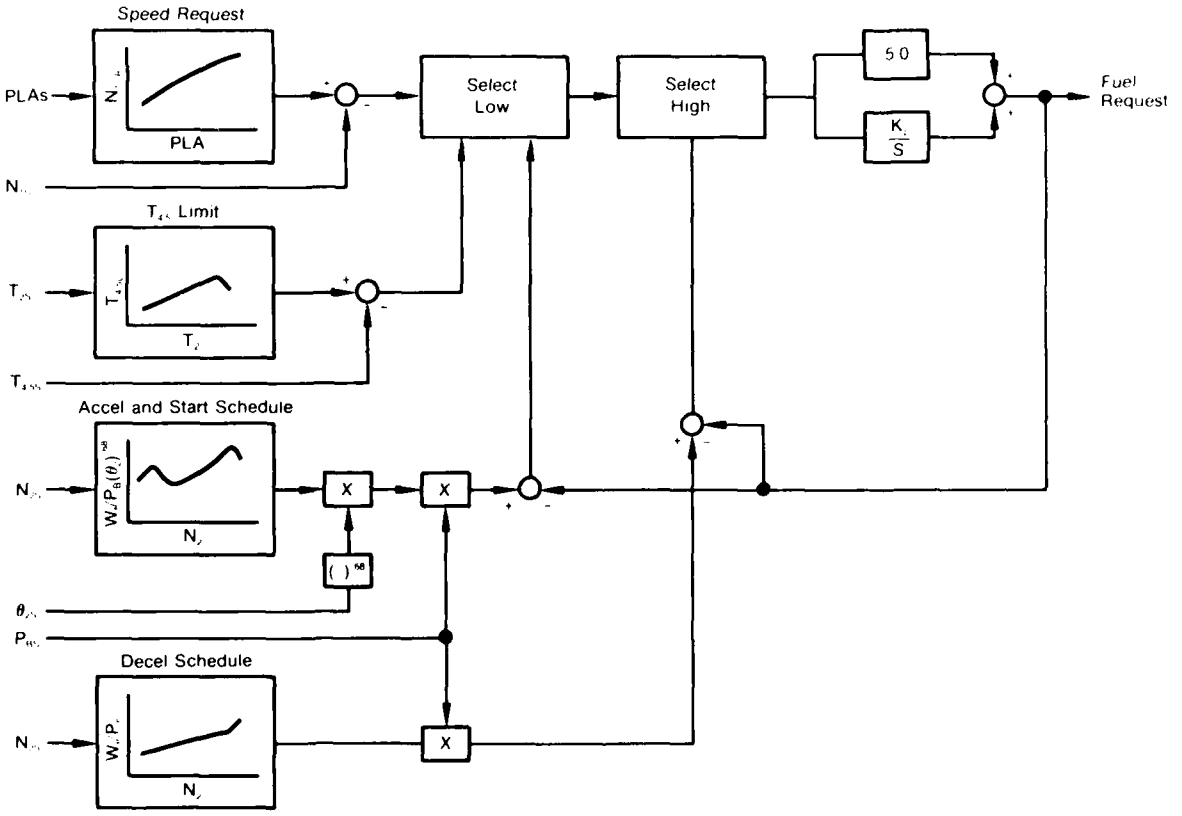


Figure 4. Main Burner Flow Control

Next, the control hardware that has been selected to implement the system must be defined. The key hardware characteristics that must be determined are also presented in Table II. The dynamic characteristics of the actuators and sensors must be defined for proper analysis of the overall system interactions and transient response requirements. The specific actuator and sensor device will also define the interface signal conditioning requirements. Definition of the data link system and computational capacity is needed to assure that the demands for operational and functional integration of the engine and aircraft controls can be satisfied. System redundancy features and redundancy management are key elements in the overall definition of the control system. Redundancy management and built-in test features commonly consume from 40 to 60% of the overall computer software. The diagnostics and maintainability requirements stated early in the control development cycle will avoid patches and add-ons that generally accomplish only part of their potential because of hardware limitations. The overall control system interfaces with the electronic control are illustrated in Figure 5. The resulting electronic control computer architecture showing approximate input signal conditioning and output drivers is presented in Figure 6. Multiplexing of input signals is used to minimize the number of resolver, analog, and frequency-to-digital converters needed to process the various sensor signals.

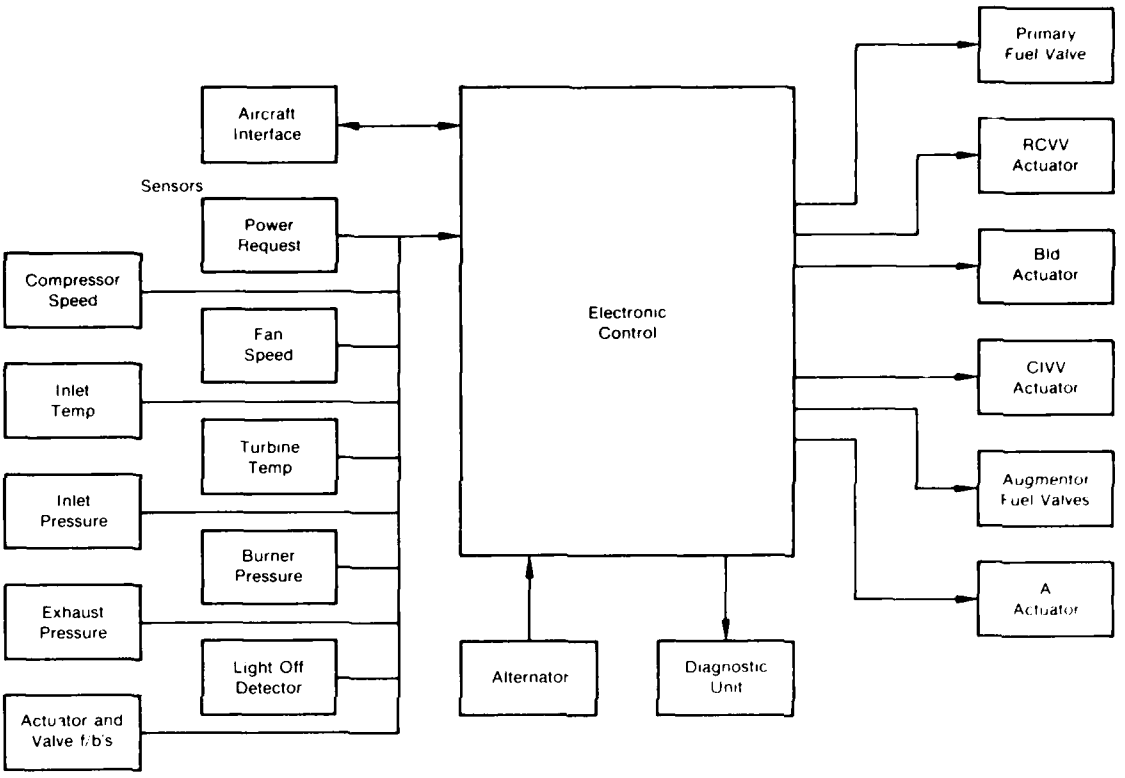


Figure 5. Control System Interfaces

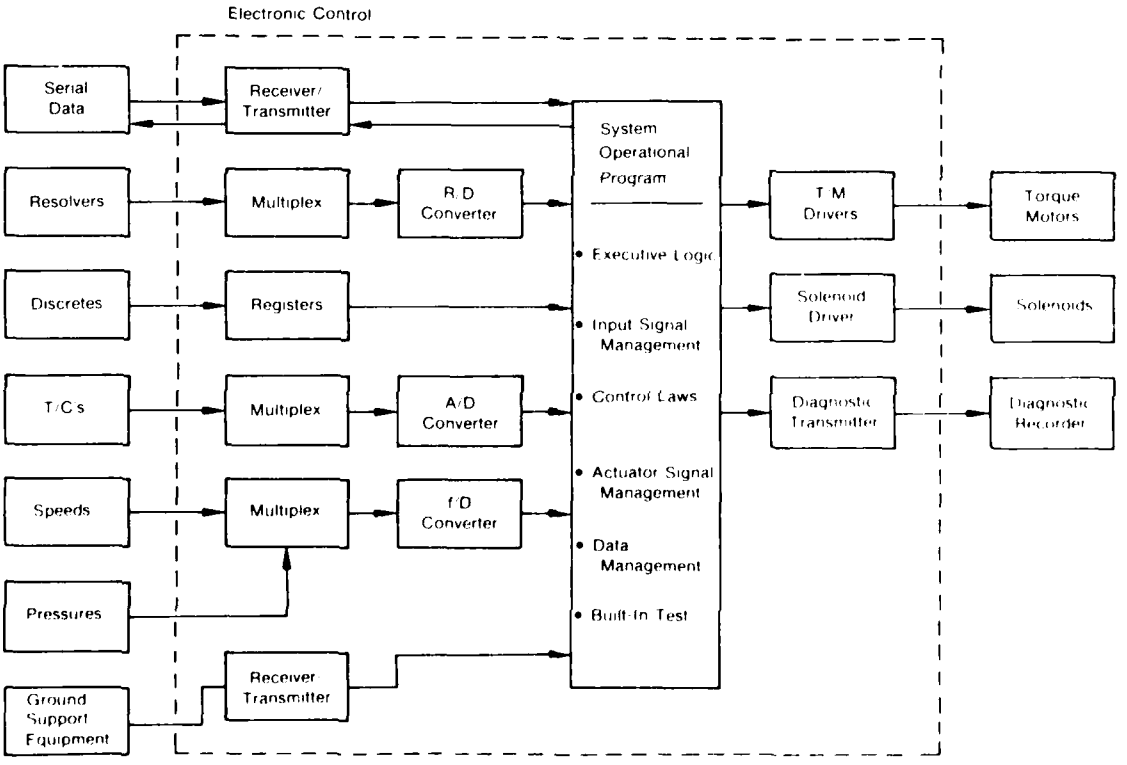


Figure 6. Electronic Control Computer Architecture

Software standards such as the "767 Airborne Computer Software Standards" (Reference 4) provide the basis from which the overall software requirements are defined and documented. In the propulsion control field the standards used are somewhat simplified because we do not have to deal with a large variety of Avionics systems. These standards guide the design and development of software through a "top-down" design process as illustrated in Figure 7. In this process all computer programs are developed using a hierarchical methodology that utilizes design trees to illustrate the software architecture. These trees clearly show the relationship between the modules and the activation hierarchy. In general, the design tree should show no more that three levels of software modules. Nodes in the design trees correspond to software design modules. In principle, all but the bottom level modules perform data "control" functions. The bottom level modules are subroutines that perform the fundamental computational tasks. Communications between horizontal modules (that is, the same level) are to be avoided. The goal of top-down design is to minimize module coupling (especially at level II in Figure 8) and to maximize internal binding within each module as defined in Tables III and IV, respectively. This methodology provides an easily comprehended and logically complete definition of the software functions and organization.

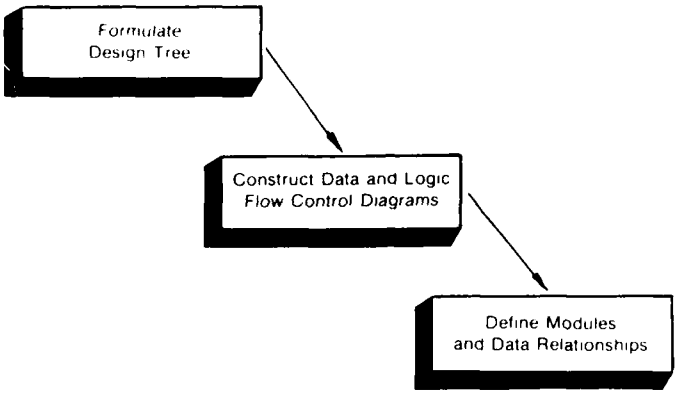


Figure 7. Software Design Requirements Process

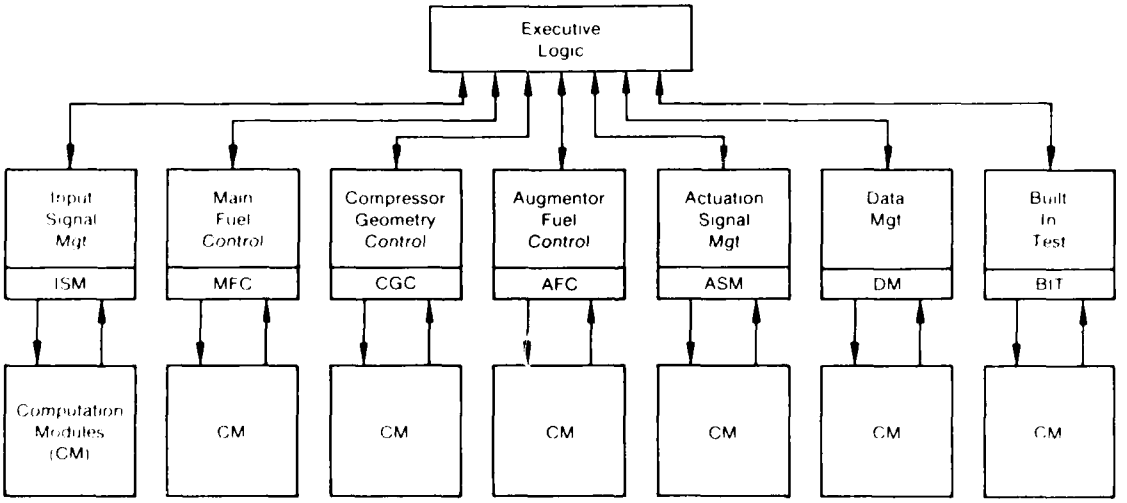


Figure 8. Design Tree for System Operating Program

TABLE III. LEVELS OF MODULAR COUPLING

1. Data	Module communication via parameter passing or reference to a common data source
2. Control	Parameter passing such that parameters generated in one module affect the data flow sequence of another module
3. Symbol Reference	Reference to an external symbol in one module that is declared in another module
4. Data Reference	One module makes direct reference to the data content of another module

TABLE IV. LEVELS OF MODULE BINDING

1. Functional	All elements in module are related to the same function
2. Communication	All elements are related to the same data file
3. Timing	Elements are related in terms of execution sequence/timing
4. Logic	Elements are related to Similar Task

A design tree for the turbofan example is shown in Figure 8. The top level module is the executive logic that controls the overall flow of information and the sequencing of computations. The executive logic provides task scheduling mode control, channel status, cross channel data transfer, inter-channel frame synchronization (if required), interrupt handling and power-on/recovery logic.

The second level modules define the partitioning of the operating program into its major functional segments. In the example presented here the program is broken down into seven functional elements. The modules noted as Main Fuel Control, Compressor Geometry Control, and Augmentor Fuel Control represent the control law computations and form the core of the operating program. For example, the control laws and logic for the Main Fuel Control module were presented in Figure 4. All data flow for computations, schedules and logic with respect to the main burner fuel control loop is controlled in this module. The Compressor Geometry Control module handles the control laws and logic for the CIVV's, RCVV's and compressor bleed. The Augmentor Fuel Control module does the same for all augmentation fuel flow metering and distribution requirements and for the exhaust nozzle area control. The Input Signal Management module supports these three control law modules by providing sensor signal fault detection and redundant sensor management. Calculations such as corrected speed, temperature and pressures and dynamic compensation are performed. This module also services the cross-channel data link. The Actuator Signal Management module supports the control law modules by providing actuator interface logic and redundancy management.

The Data Management module provides sensor and actuator data to, and processor data from, the aircraft computer. This interface is handled by means of a multiplexed data bus. Flight test instrumentation data would also be processed in this module. The six modules discussed to this point comprise the foreground activity that has top computational priority in the propulsion control computer. The Built-in-Test (BIT) module is of lower priority and treated as background. BIT is divided into two main functional segments: Periodic BIT and Initiated BIT. Periodic BIT execution is allocated at least one-tenth of each major cycle. Initiated BIT is executed only when the aircraft is on the ground in conjunction with system diagnostic activities. The Sensor and Actuator Signal Management Modules support the BIT by accomplishing failure detection. The tasks of failure isolation and enunciation for maintenance action are allocated to the BIT module. The fundamental computational tasks such as table look-up, data interpolation, and frequency compensation are performed in the third level modules. These subroutines are already essentially in modular form.

This technique of top-down design not only results in a functionally meaningful and architecturally balanced partitioning of the operating program, but it also becomes the basis for organizing the software development team. Each segment can be assigned milestones and attacked separately at different rates of progress depending on the complexity of that segment. Collectively these software design trees, control modules, data flow charts, data organization, and diagnostic features represent the baseline software design requirements.

SOFTWARE DESIGN, CODING AND TEST

The Software Design activity describes precisely how the requirements identified in the prior phase will be implemented. The activities of this phase are presented in Figure 9. Detail module software is generated, debugged, and documented on both a host and the target computer.

The use of a high level language in the host computer software design step is one of the most important factors in developing cost-effective and reliable software. The programming language permeates every aspect of the software development and utilization. A high level language serves as a communication medium that all parties can easily understand and review. Programming in a high level language such as FORTRAN, BASIC, PASCAL, ADA, etc., rather than a low level assembly language, reduces the time required to generate and/or modify code while simultaneously reducing the likelihood of inducing or failing to detect errors. It is far easier to utilize a high level language that has well developed compilers to transform the program into the target computer machine language. In contrast, a programmer working in a low level machine assembly language is required to know each detail of machine "housekeeping" tasks and to include these tasks in his program. A single line of a high level language statement will be equivalent to approximately ten lines of assembly language coding, and the compiler (program) which converts a high level language program into the required ones (1) and zeros (0) object program will automatically take care of the detail housekeeping tasks. In general, the high level language allows the programmer to concentrate on the application program rather than programming per se, as in assembly language.

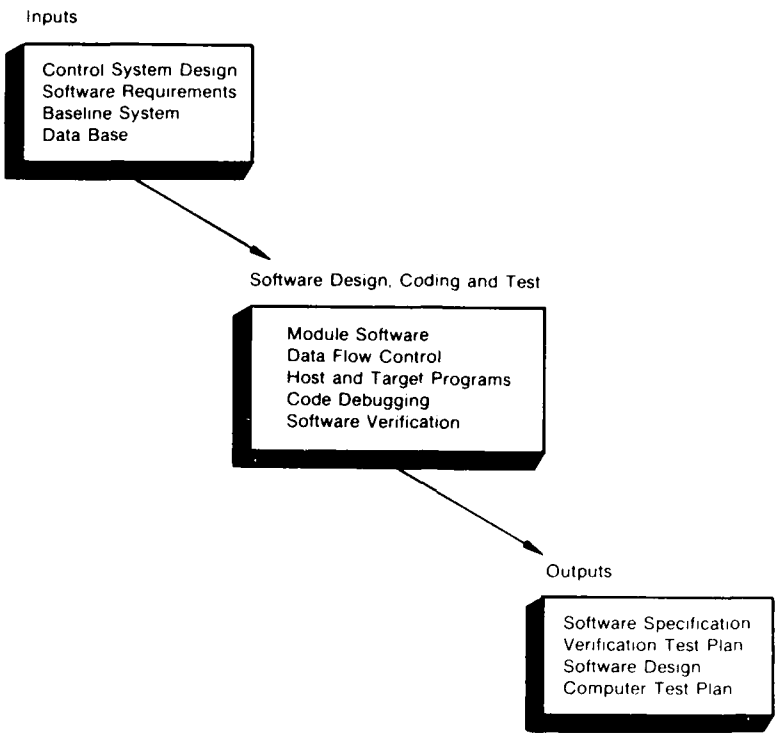


Figure 9. Software Design Provides Detail Target Computer Specification and Test Plan

A high level language, in addition to providing a good level of visibility, also greatly facilitates the construction of top-down, structured code. This structuring feature of the languages introduces a natural discipline in program and data organization that complements the modularity objectives discussed in the previous section. Writing and debugging a high level language program is, in general, an order of magnitude faster than in assembly language and therefore will cost about one-tenth as much. Errors associated with any given program can be classified into four basic types:

1. Compile time errors which can be recognized during program translation.
2. Run time errors which are recognized during the execution of the object program.
3. Program logic errors which are not detected by the computer during compilation or execution.
4. Input data errors which are not confined to the program.

Type 1 and 2 errors are identified and fixed in a relatively short time period. Type 3 and 4 errors usually absorb great amounts of time to correctly identify the logic or data problem(s) and to ensure the fix does not have an error-side-effect to the balance of the program. High level language with diagnostic capability excels in identifying all types of errors, especially the type 3 and 4 errors which will be handled ten to fifty times faster than can be accomplished using machine low level assembly language.

The reliability of support software has matured over the last 10 years to the point t[illegible]pport software is not an issue for critical applications. Otherwise the high level language implementatio[illegible]ld [illegible]s, ne a major part of the problem rather than a major element in the timely development of software. Th[illegible]ination [illegible]of (1) structured, high level language, (2) fully matured support software, and (3) program visibility will [illegible]mize program costs and schedule risks for design of the module software.

An example of a high level language instruction set is presented in Figure 10. This instruction set, based on BASIC, was chosen as an example because of its simplicity and widespread utilization. Several special functions have been added to tailor it for propulsion control applications, namely: integration with respect to time, highest wins, lowest wins, and look-up tables (interpolation is performed automatically). For this language, all instruction lines are preceded by line numbers in ascending order. To illustrate the use of this programming language, the main burner fuel flow control logic presented in Figure 4 has been programmed. Figure 11 shows the resulting high level language source text. Ten lines are simply comments to clarify the sensed parameter nomenclature. These variables are computed in the Input Signal Management Module. The next 9 lines define the table look up data and co[illegible]tants. The control logic module took only 7 lines of code.

Type	Example
Assignment	X = A + B* (C- INT2 (D))
Integration	X = INT3 (A)
Highest Wins	X = Max (A,B,C,D)
Lowest Wins	X = Min (A,B,C,D)
Table Lookup	X = Lookup 4(A)
Loop Initialize	For I = 1 to 10
Loop Terminate	Next I
Program Jump	GOTO 15
Conditional Jump	1F (X = 7) GOTO 59
Subroutine Call	GOSUB 100
Conditional Subroutine Call	1F (Cond) GOSUB 45
Return Subroutine	RETURN
Rename Variable	DEFINE AOUT1 = X3
Table Data	Set LX1 = 0 0, 0 50, 2 0 LY1 = 4 0, 4 0, 6 5
Constants	Set K1 = 50
End	End of Program

Conditional Operators

= Equal To
≠ Not Equal To
> Greater Than
< Less Than

User Defined Variables

A,B,C,D,X,Y,Z

Figure 10. High Level Language Instruction Set

```
10    Example Program for Fuel Control
20    .
30    .          AIN1        PLAS       Sensed  Power Request
31    .          AIN2        NICS       Sensed  Fan Speed, Corrected
32    .          AIN3        T2S        Sensed  Engine Face Temp
33    .          AIN4        T45S       Sensed  Turbine Temp
34    .          AIN5        N2S        Sensed  Compressor Speed
35    .          AIN6        Theta 2S   Sensed  Corrected T.
36    .          AIN7        PBS        Sensed  Burner Pressure
37    .          AOUT1       FR         Fuel Request
40    .
50    Set LX1    0, 20, 90, 200,                 Speed Request Schedule
55    Set LY1    0, 4000, 9000, 9000
60    Set LX2    -100, 0, 100, 500,              T4 5 Limit Schedule
65    Set LY2    1500, 1500, 2000, 2000
70    Set LX3    0, 2000, 4000, 6000, 12,000, 14,000   Accel Schedule
75    Set LY3    25, 30, 22, 30, 35, 25,
80    Set LX4    0, 2000, 14,000                 Decel Schedule
85    Set LY4    10, 10, 15
90    Set KP     1
95    W1         Lookup 1 (AIN1) - AIN2          Speed Error
100   W2         Lookup 2 (AIN3) - AIN4          T45 Error
105   W3         (Lookup 3 (AIN5))*(AIN6** 68)*(AIN7)   AOUT1
106                                              Accel Error
110   W4         MIN (W1, W2, W3)                Select Low Error
115   W5         (Lookup 4 (AIN5))*AIN6 - AOUT1  Decel Error
120   W6         Map (W4, W5)                    Selet High Error
125   AOUT1      5 * W6 + INT1 (W6)
130   GOTO       135                             Branch Back to
                                                 Exec Module
135   End                                        Needed Only in
                                                 Checkout
```

Figure 11. High Level Language for Fuel Control Loop

This initial coding and debugging is done on a general purpose host computer facility. The debug, editing and simulation facilities inherent to the general purpose computer provide highly efficient resources for initial program development and testing. The sequence in which the modules are coded and tested is illustrated in Figure 12. The various modules are coded in parallel and tested on a stand-alone basis. The computational modules are coded and tested with dummy inputs. The upper level modules are coded and tested using stubs for the lower level modules. After the computational modules have completed the stand-alone tests, they are integrated with the upper level modules. The transition from purely module testing to the total system is progressive in reverse order of the hierarchy specification level. After each module is coded and debugged the listing of that module becomes a detail part of the software specification. When all modules have completed their integration tests, the program is subjected to a final series of system tests which are made against an appropriate simulation of the engine.

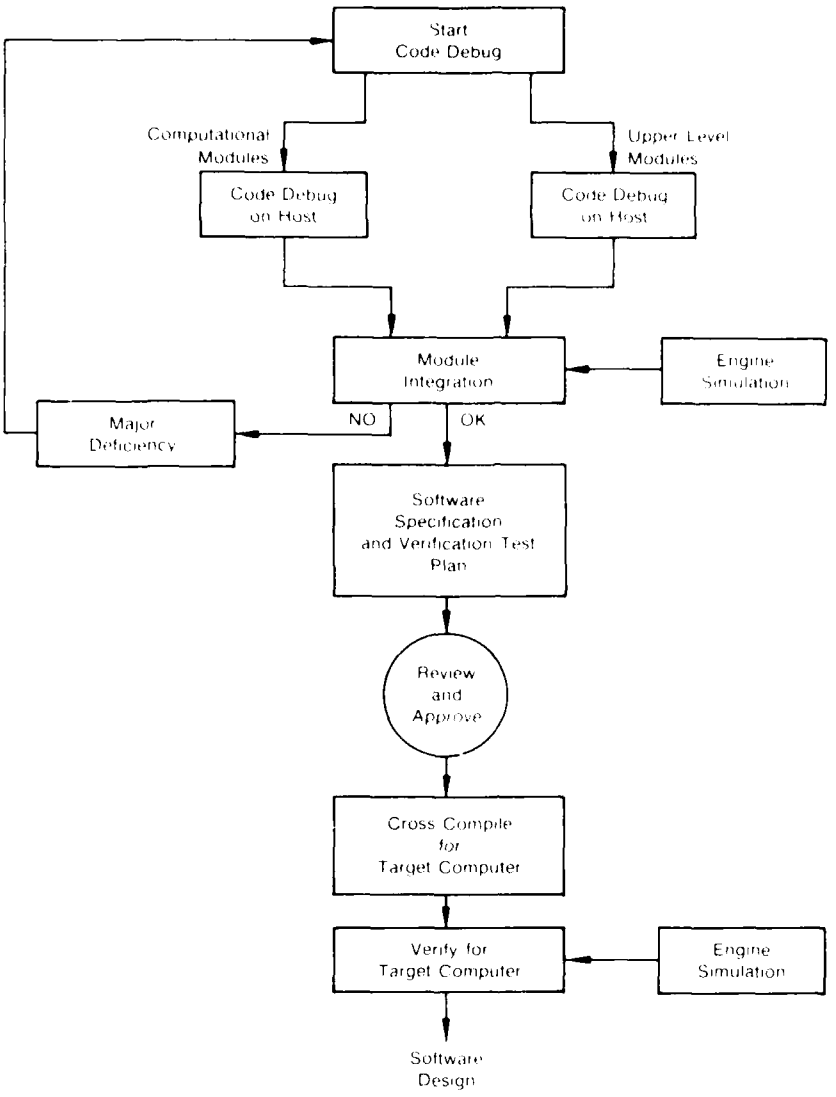


Figure 12. Incremental Software Buildup

The Software Specification generated during this activity is the key document that becomes the cornerstone of all future software work.

This specification is in the form of detailed block diagrams, high level language listings, and dynamic performance specifications. The Software Specification, unlike hardware specifications, does not reflect minimum requirements, but rather is a statement of the actual implementation requirement. The software specification is used to generate the specific target computer software design.

Software Verification test plans are also generated during this step identifying the various levels of testing to be performed on the target computer software. For every functional and operational requirement in the software specification, there is a correlated verification requirement. The main objective of this software testing is to verify that the modules are error free and that the subsystem satisfies the software specification requirements. It is the goal of testing to exercise as many of the permutations and combinations of computer program paths as practical. Examples of these tests are presented in Table V. Module and Integration Test results are fully documented, and required module changes identified during the integration test ar implemented under change control procedures.

At the completion of this host computer software development, the software specification, results of the hardware/software trade studies, and the Verification Test Plan requirements are presented to the project management for review and approval. Upon approval, these elements provide a key input to the software documentation process, and the target computer software programming phase is initiated. After the system software is cross compiled on the target computer it is again tested against the verification test plan requirements to assure proper operation in the target computer environment

TABLE V. SOFTWARE TESTS

A.	Steady State	Tables, Curvefits Discrete Functions Calibration
B.	Data Sweeps	Full Range of Input Signals Combinations of Input Signals
C.	Transient	Small Step Inputs Large Accels, Decels, Bodies Mode Transfers Reset Functions

A Software Design Document, generated by the target computer supplier, includes items resulting from the software design coding and test process such as data flow diagrams, timing charts and memory maps. This document provides the functional description, detailed design, interfaces and correlation between the functional requirements and the module implementation. It includes descriptions of the breakdown of individual modules, identification of modules and subroutines, processing methods, program listings, and definition of any limitations on the usage and design application. The Software Design Document is finalized at a critical design review and goes under vendor configuration control prior to the software being implemented for Subsystem and Engine Tests.

This incremental software design approach, combining high level languages with both host and target computer evaluations, puts early emphasis on control law performance, confirms the proper interrelationships between modules, and provides high visibility into the target machine performance.

SUBSYSTEM TESTS

As indicated in the previous section, much of the software verification is done on an incremented basis such that module verification, integration tests and system verification represent mini-milestones during the programming and debugging phase. Testing depends primarily on a rigorous "design-for-test" philosophy being applied throughout the software development cycle. The software is verified on the actual target computer according to the same Computer Test Plan and data documentation procedures generated in the previously mentioned system integration tests. Computer subsystem testing, performed with the actual computer, Input/Output hardware, sensors and effectors, is also conducted to determine software and hardware compatibility. The results are analyzed to verify proper open and closed loop system control. Closed loop testing is conducted with the engine simulated at sea level and altitude conditions under steady-state and transient situations and subjected to various simulated system faults. The tests conducted are summarized in Table VI. Testing is pursued to a level of confidence that assures the software is ready for evaluation with the engine.

The engine test program verifies hardware/software operation in an installed configuration under actual engine operating conditions. The engine test program is geared to assure operation under all normal conditions, including starting, idle, acceleration, setting rated power, deceleration, and rapid power lever changes from various power levels. The acceptance criteria are based on comparisons of the test results with data from the nonlinear simulation during control design.

TABLE VI. SUBSYSTEM TESTING

	Open Loop		*Closed Loop*	
A.	Repeat Software Tests (Table V)	A.	Steady State	Calibration Stored Data
B.	Input/Output Response	B.	Minor Loop Test	Response Accuracy
C.	Input Noise	C.	Major Loop Test	Engine Response System Faults
D.	Failure and Range Checks	D.	Hardware Tests	Noise Faults
E.	Startup and Initialization			
F.	Power Interrupt			

SUMMARY

This paper presented an overview of the major software development requirements for an advanced propulsion control system. The methodology described provides systematic design discipline and program visibility. As part of the software requirements preparation, a top-down process based on modular techniques has been defined that results in functionally balanced partitioning of the operating program. The resulting software organization is easily comprehended. The combination of high level languages and fully matured support software provides an effective development tool as well as a means to document the system logic implementation. Strategically timed program reviews keep the development activities on schedule and major design issues in perspective. Configuration control and documentation procedures span the entire software life cycle. Finally, subsystem and engine tests assure that the software and hardware are compatible and that the system is ready for production release.

REFERENCES

1. Lenox, T. G., et al, "Full Authority Digital Electronic Control," Phase I, June 1978, FR-8652.

2. Boehm, B. W., "Software Engineering," IEEE Transactions on Computers, Vol C-25, No. 12, December 1976.

3. Miller, R. J., Hackney, R. D., "F100 Multivariable Control System Engine Models/Design Criteria," AFAPL-TR-76-74, November 1976.

4. "767 Airborne Computer Software Standards," Boeing Commercial Airplane Company, June 1978.

BIBLIOGRAPHY

1. Bosch, J. A., Briggs, P., "Software Development for Fly-By-Wire Flight Control Systems," AIAA 78-1276.

2. Eccles, E. S., "Software for Flight Critical Digital Engine Controls," ASME Paper, March 10, 1980.

3. Rutherford, D. A., "Simplifying the Development of Programs for Digital Engine Controllers," ASME Paper, March 10, 1980.

4. Schindler, M., "Pick a Computer Language That Fits the Job," Electronic Design, July 19, 1980.

5. Larsen, W. E., "Methods for the Verification and Validation of Digital Flight Control Systems," SAE 801137, October 13, 1980.

MICROPROCESSOR SYSTEM TEST AND MONITORING

AUTHOR: Mr. J. F. O. Evans, B.Sc., Computer Manager
Smiths Industries Aerospace & Defence Systems Company,
Winchester Road, Basingstoke, Hants. RG22 5HP.

SUMMARY

Digital control systems for aircraft jet engines require extensive and careful development testing if they are to meet the rigorous performance and safety requirements of the engine and airframe manufacturers and air certification authorities. In order to achieve the level of testing required whilst minimising expensive engine running hours, relatively sophisticated test procedures and equipment are required. These tests need to cover the system hardware, real time operating system software and control software.

The paper outlines the development of the required testing techniques within the author's company and describes the current equipment and procedures.

INTRODUCTION

The paper describes work done within Smiths Industries in the development and test of engine control systems. This work has largely been carried out within the Newbury and Smiths Industries Controls System organisation, although the association of Newbury and Smiths Industries goes back many years before their formal association in this field.

An aircraft engine digital control system consists typically of one or more microprocessor-based control units or "lanes" built into a redundant system which controls the fuel flows and certain mechanical adjustments required for correct engine operation.

The redundant system organisation will be considered in more detail in another paper, but it typically consists of an arrangement such as that of Figure 1.

Each of the "lanes" acquires the desired engine thrust signal from the pilot's lever and a variety of pressures, temperatures and shaft speeds from transducers mounted on the engine. These signals are conditioned, digitised and the required output actuator movements computed. These computations may require as many as 15 or more separate control requirements to be evaluated of which the most significant are used for control at any one time.

The primary controls are ususally:

(a) Main engine thrust.

(b) Afterburner thrust (Military and supersonic applications).

(c) Engine mechanical limiters (Maximum pressures, speeds and temperatures).

(d) Engine thermodynamic, combustion and aerodynamic limits (Mass - flow, mixture, compressor and turbine stall, etc).

(e) Engine life control (Reduction of stress/time integrals).

These controls must be achieved using techniques which ensure that the control is stable and safe. For these reasons they are subjected to extensive test and design scrutiny.

It is the purpose of this paper to outline some of the test procedures and equipment which are currently being used to achieve these aims.

DEVELOPMENT PHASES AND REQUIRED TEST FACILITIES

The design of the new engine controller usually starts with a requirement from an engine manufacturer stating the type of engine to be controlled and its characteristics together with a statement as to the control techniques to be employed and the tolerance, cost, weight and safety budget.

The specification will usually state the control features and stability margins required but will seldom give data on the best method of achieving stable control. In the same way, the manufacturer will state the safety requirements but may only indicate the preferred system organisation he wishes the control supplier to use.

The first major tasks are thus to examine the system organisations and the control algorithms. At this stage, system organisation is essentially a paper exercise but to examine control stability it is necessary to establish a reasonably good

dynamic model of the engine. In many cases, such a model will be available from the engine manufacturer as a simplified and developed form of his computer design model.

Aircraft engine performance characteristics are very non-linear and it is usually easier to simulate their behaviour on either hybrid or digital computers and to use the model to derive any point design data, than to rely on other forms of design computation. In any case a good engine model is essential in the later stages of test in order to reduce the very expensive engine running times.

The author's company has, therefore, developed a range of very flexible simulation techniques which have proven more than adequate for these tasks.

When the simultion model is complete and control system design data obtained, provisional control system designs are evaluated in simulated form against the simulated engine.

As soon as sufficient design data is available from the system and control studies for a microprocessor timing and core budget to be established, hardware and software design is begun.

All software is modular and as each module is completed it undergoes extensive design scrutiny and test. Design scrutiny covers such points as the completeness of the module specification with respect to the next higher specification, verification of interfacing standards and completeness and accuracy of any algorithmic requirements. The module test specification is then written by a "third party" and a test harness generated.

Tests cover static, dynamic and random inputs and the results are verified first against the specification and also by comparison with the results of an independently coded module running on a different type of computer.

The software modules are then built up slowly into the overall system and each functionally meaningful sub-system is checked in a similar manner.

The system hardware is also modular and built up in a similar manner. Again, a wide range of tests are required to minimise the possbility of data-dependent errors causing errors at a later stage.

The hardware and software for each system lane are then brought together for initial system tests. At this stage these tests include correct operation of the input/output sub-systems, the executive program which schedules the various operations at the correct time within the control iteration and the control functions.

The various system lanes are then brought together and their operation checked in the overall system configuration against the engine simulation operating in real time. Tests include control performance, recovery from simulated fault conditions and environmental and other stress tests.

The equipment is then usually taken for confirmatory tests with the hydromechanical system before tests on an engine test bed. Finally, the engine, hydromechanical and digital control are tested in a prototype or experimental aircraft.

It will be seen that very exhaustive testing is required at all stages of development, firstly to secure that the stringent safety requirements can be met and, secondly, to minimise the risk of needing to repeat the later, more expensive, test phases. In order to carry out such an extensive test program economically, relatively sophisticated test equipment is required.

SUMMARY OF TEST FACILITIES

The test facilities which have been built up for these tasks are based on a well known range of 16-bit minicomputers. These machines have a wide range of operating systems covering both real-time, interactive and background/batch operating and in both memory and disk based configurations. They also have a comprehensive inter-machine communications facility which can be easily adapted for direct communication to a microprocessor by the use of a pair of LSI communications chips. Particularly helpful is their program code standardisation at machine and high level and within the various operating systems and performance range.

Software development and test is carried out on configurations providing multi-terminal access to source editers cross-support compliers, assemblers, loaders and simulators for the various microprocessors used. Test harnesses are written in a mixture of high and low level lenguage and tests run in both interactive and batch modes. The cross support loaders and simulators are able to accept complete systems programs as they are built up thus minimising the test work which must be done in the final microprocessor hardware.

These configurations are also used for general engineering computations associated with the hardware design, for example, thermal and mechanical stress calculations as well as the engine and control system simulation facilities.

The simulation program suite consists of a basic interactive simulation language using mnemonic references to engine and control system signals and building blocks. A simulation, such as shown much simplified in Figure 2, be built up from these blocks quickly and modified as required.

Both linear and non-linear blocks are available including a full range of one and two dimension function generators, hysteresis elements, time delays, counters, relays, data dependent switches such as high/low wins and median selection, integrators and special purpose transformations. Display facilities include tabulation and continuous graphics with numerical back-up on a raster CRT. Data recording and analysis facilities are provided together with operator interaction allowing data and other changes to be made on a running simulation.

The maximum size of simulation currently available is approximately 600 equivalent analogue computing elements with no limit on the number of integrators or other dynamic elements. Real time simulation can be run at about 4000 analogue blocks per second with firmware floating point and up to about 8000 blocks per second with hardware floating point facilities. The differential equation solvers available include single pass predictor-correctors which are fast and reasonably accurate for systems of low stiffness and slightly slower but accurate solvers capable of handling very stiff systems. Facilities are also provided for obtaining the small-signal performance of the simulation at any specified working point, together with a range of control design tools such as Nyquist Bode and pole locus plotters.

Engine simulation generated and evaluated on the larger computer configurations interactively can be transferred without modification to the smaller special purpose configurations used for system test. This commonality has been achieved by combining the ease of data manipulation of full floating point operation with a very high speed interpreter which can be used for real time simualtion as well as in a batch/interactive environment. This avoids any need for time consuming, hand coded simultions for real-time working.

Some work has also been done on the automatic conversion of floating point to fixed point working for the automatic generation of the control calculations used in the microprocessor controllers. This will lead to automatic coding of the control program from the control which has been simulated in floating point.

Some work has also been one on the "blocking" of simulations into, for example, engine, control lane "A", control lane "B", etc. This work, when complete, will enable more complete simulation of the system response to failure by making it possible to study the interaction of multiple controls operating either in synchronous or asynchronous modes.

The test facilities available consist, therefore, of large multi-user minicomputer configurations used for software, hardware and control system development and test essentially prior to the construction of the control hardware and smaller dedicated configurations used in real-time testing of the hardware.

These dedicated systems are usually referred to as Rig Systems and are used in the laboratory during system build and test, for inspection testing and for engine test bed running of prototype systems.

THE RIG SYSTEM

The final system test rig again uses a minicomputer as shown in Figure 3. This minicomputer is always provided with an operator's terminal, a raster video display for continuous data, an engine simulation and analogue interfaces between the simulation and the microprocessor controllers under test. To this is usually added a magnetic tape unit for data logging and some form of disc back-up storage either directly off via a Hewlett Packard DS1000 communications link to one of the larger configurations.

This basic facility can be augmented by the addition of DS1000 hardware compatible serial digital commuinication to the microprocessor controllers. This link can be used both for monitoring data within the controllers in real-time and/or to provide diagnostic/debugging facilities from the operator's terminal. Further occasional enhancements include fast printers, multi-terminal access and miscellaneous peripherals such as PROM blowers, graphics, tablets, etc.

RIG OPERATION

The rigs operate in two modes, in the first, shown diagramatically in Figure 4, the hardware under test is run against an engine simulation in the rig computer. In the second mode, shown in Figure 5, the hardware under test is running a real engine rig acts as a data logger and monitor of the engine, controller and test bed environment.

Figure 6 shows a typical display which can be obtained during simulation running or engine running or by the "play back" of data recorded during a previous run. The recorded data is also available for analysis at any time and limited real-time analysis is also possible.

Software available includes the rig operating program suite which runs in the standard memory-based or disc-based operating system, a full set of development cross support software for the microprocessor being used and the general purpose simulation and control development software.

The overall rig system typically occupies a single standard rack and is robust and portable. They are often taken to relatively difficult environments with none of the protection usually associated with computer equipment and used continuously. This is essential for the type of work which we normally undertake and makes it possible to continue development at the customer's site with minimal back-up from the main base.

RIG PROGRAM SUITE

The rig program suite consists of the following programs:

(a) TIMER

Timer is scheduled by the operating system at the iteration rate required by the simulation (normally 50 mS). It then schedules the real time forground program CYCLE once per iteration.

(b) CYCLE

CYCLE is a forground program which is scheduled by Timer and terminates when it, in turn, has performed all the cyclic functions of recording, data input, simulation, display and data output.

(c) MTX

MTX is a forground program which is scheduled by CYCLE whenever one of the recording buffers is full. It epties the buffer to the Magnetic Tape Unit asynchronously using DMA transfer.

(d) RECOP

RECOP is a background program running in available time and performs all the operator interactive dialogue which controls the other programs.

The rig software is again modular and is largely data driven. This makes it possible to store a number of simulations, display and data logging formats as data files which can be called upon with minimum delay. Indeed, the prime objective of the rig system is to minimise the time needed to set up system test of all kinds.

OPERATOR INTERFACE

The operator controls the rig system by means of interactive commands to the program RECOP, either directly or by calls to Command Files. Using these commands, he can:

(a) Control the loading and running of the simulation package.

(b) Control the collection and analogue and digital data in the main recording buffer using mnemonic calls to the data available from the analogue I/O package, data available from any of the microprocessor controllers and data from the simulation. During rig running standard sets of data can be introduced by the use of command transfer files stored on the local or remote disc system.

(c) Control the display of data on the continuous grahical display screen. Up to 8 analogue quantities and 8 digital quantities can be viewed on a monochrome display, rather more can be usefully displayed on a multicolour display. The data is shown as a stationary graph which is updated in real time by "wiping" a vertical cursor across the screen. Hence, the data is read from the cursor to the right hand edge of the screen and then up to the cursor. This allows the data to be stationary for a significant time.

The current value of the various parameters is also shown in numeric form down the right hand edge of the screen as shown in Figures 5 and 6.

The display may be "Frozen" at any time for more detailed inspection and the cursor moved over the stationary display. When this is done the numerical data always corresponds to the values "under the cursor", thus enabling detailed analysis to be obtained instantly.

Freezing the display does not stop the magnetic tape recording, however, and at the end of the test the system can be switched into a playback mode in which the data is read back from the magnetic tape and displayed as if the engine were running. Important events can thus be analysed in detail either visually or by computer program. Simple analyses can be coded as simulation programs and hence carried out within the rig program - including during engine running when the simulation would not otherwise be required. Dual tape decks can be used to allow continuous recording of long engine runs.

(d) Examine the operation of any of the microprocessor controllers by running a "monitor" or "debug" program within them entering commands from the operator console and print console and printing out data, register contents or program on the same console or printer.

DATA PATHS

The first part of the CYCLE program reads the analogue data inputs from the pilot's controls and from the actuator pick-offs as shown in Figure 5. This data may take many forms, e.g. d.c. voltage, d.c. ratiometer, a.c. voltage or ratiometer, digital pick-off signals, frequencies, etc. For convenience all ranged outputs are converted by acquisition circuits to standard d.c. signals which can be fed into a standard A/D converter. The resulting digital signals are read into the minicomputer memory by a software driver. At the same time any "state" signals are converted to a standard d.c. level and read into the computer memory packed into 16-bit words. This part of the input process is synchronous with the minicomputer iteration rate set by the program TIMER.

Digital communications with the microprocessors are carried out synchronously with the microprocessor iteration rates and are therefore asynchronous to the TIMER controlled cycle. These signals are handled by privileged interrupt within the minicomputer and when an interrupt is received a "handshake" is followed by a block of up to 64 16-bit words of data. The maximum data from each microprocessor is thus approximately 64/iteration. The microprocessor iteration rate is of the order of 50 mS, hence, for a three microprocessor system, the minicomputer is handling up to 3840 words per second.

One of these words is a flag word enabling the same communication path to be used as a medium for interacive control of the microprocessor from the recording program RECOP.

When monitoring or debugging operatings are being carried out via this link a small link-controlled monitor program acquires and stores data within the microcomputer under the command of a larger remote monitor in the minicomputer. This small monitor is typically 100 words long and thus has minimal effect on the microprocessor memory budget.

The serial link between the minicomputer and the microcomputers requires two 16-pin DIL LSI circuits to be mounted on each microcomputer and one serial interface card per microcomputer on the minicomputer host.

The same type of LSI interface circuits are used for communications between the "lanes" of the microprocessor control system.

All the acquired data is held in temporary buffers within the CYCLE program and at the end of the cyclic input the data specified in the RECORD list is transferred to the current magnetic tape buffer. When a buffer is full, buffer switching occurs and the full buffer is dumped to tape by the program MTX.

The CYCLE program next schedules the simulation (if it is required). The simulation program consists of a series of subroutines which can be called or linked in an interpretive mode. The calling sequence defines the simulation and is generated or compiled separately. The calling sequence consists of link addresses to the required subroutine and to the next part of the calling sequence; a variable amount of fixed data relevant to this particular call to the subroutine and any work space required which is local to this call. A short entry routine starts the run through the simulation, another transfers control from one part of the calling sequence to the next, i.e. where the next subroutine call is to be found, and a final routine exists back to CYCLE.

Apart from the usual simulation facilities the simulation can input data from the CYCLE data buffer and send simulated data back to other parts of the same buffer.

CYCLE now searches the DISPLAY list for the data which is to be put onto the video graphics display and sends this data to the display in graphical and numeric form and at the same time updates the position of the cursor.

Finally, CYCLE terminates saving all its data for the next iteration, effectively passing control of the rig program to RECOP until the start of the next iteration is signaled by TIMER.

APPLICATIONS

The above equipment has proved its worth over a number of years in the development of control and instrumentation systems as diverse as a minimal reversionary digital control for a VTOL engine, helicopter engine control and full authority multiplex controllers. The concept of real-time graphical data display during engine running

together with comprehensive recording and replay facilities has been rapidly and enthusiastically accepted by engine test bed personnel as well as by the electronic development teams and will certainly be continued in all our future programmes. The only drawback is the growing mountain of magnetic tape from previous tests.

On the simulation and control design front, we find that the equipment and techniques described to be a very flexible design tool because they can deal easily with the non-linear aspects of our control problems. At the same time the design tools are being continuously upgraded to allow the use of the more modern approaches to control design including, for example, the design of true multi-variable systems.

CONCLUSIONS

We are living in a design age in which the performance of yesterday's large corporate computer is matched by today's mini, yesterday's mini by today's micro, yesterday's micro by today's hand calculator and yesterday's hand calculator by today's watch. We are already running faster and faster to make use of this technology and if we are to have even a hope of success we must develop and use systems and hardware which can be built upon and not rebuilt with each turn of technology.

The cry must be, therefore, for modular design, common languages and hardware families which are ALWAYS upwards compatible but which do not prevent the user taking advantage of improvements in computing power and miniaturisations.

Nowhere is this so important than in the tools of the trade such as described in this paper for they need to be used and developed by today's development engineers and be available for tomorrows production and maintenance personnel whilst the new upwards compatible tools are in the laboratory.

In the work described here we hope that we are progressing in this manner and we are thankful that we see in the present software and hardware many of the items which were built into the earliest of these test systems and many more which are straightforward developments of earlier ideas. In this we are considerably aided by the very comprehensive range of computing power in modern mini-computers of hardware and software over a wide range of applications.

ACKNOWLEDGEMENTS

The author wishes to acknowledge the assistance of his colleagues in the preparation of this paper, especially Mr. K. Harris, Mr. P. Matthews and Mr. R. Pitfield.

He also wishes to thank the Directors of Dowty and Smiths Industries Controls Limited for permission to publish this paper, although the views expressed are his own.

Part of the work described in this paper has been carried out with the support of Procurement Executive, Ministry of Defence for which the author also offers his thanks.

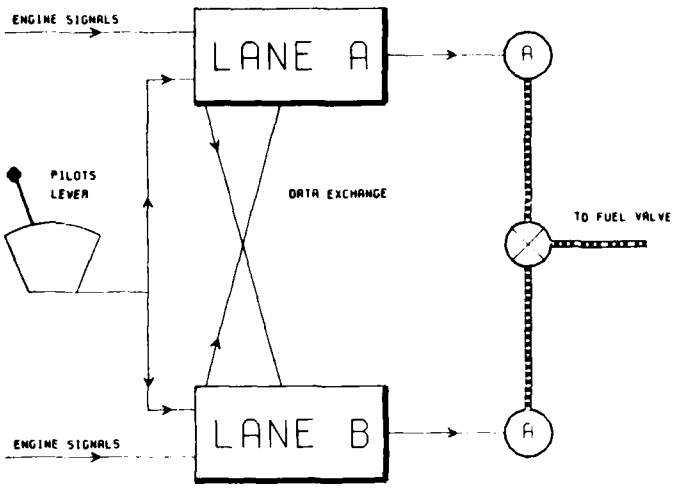


Figure 1 Typical Dual Lane Engine Control System

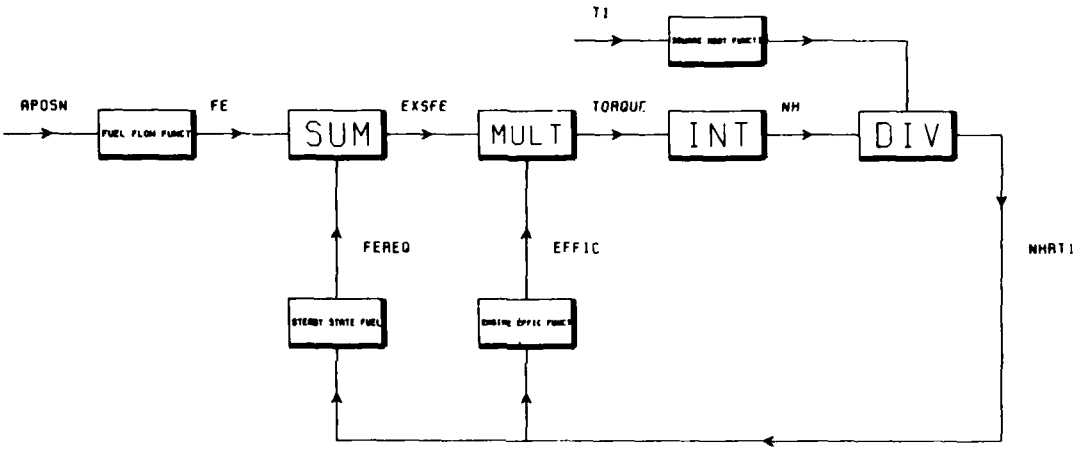


Figure 2 Elementary Simulation Block Diagram

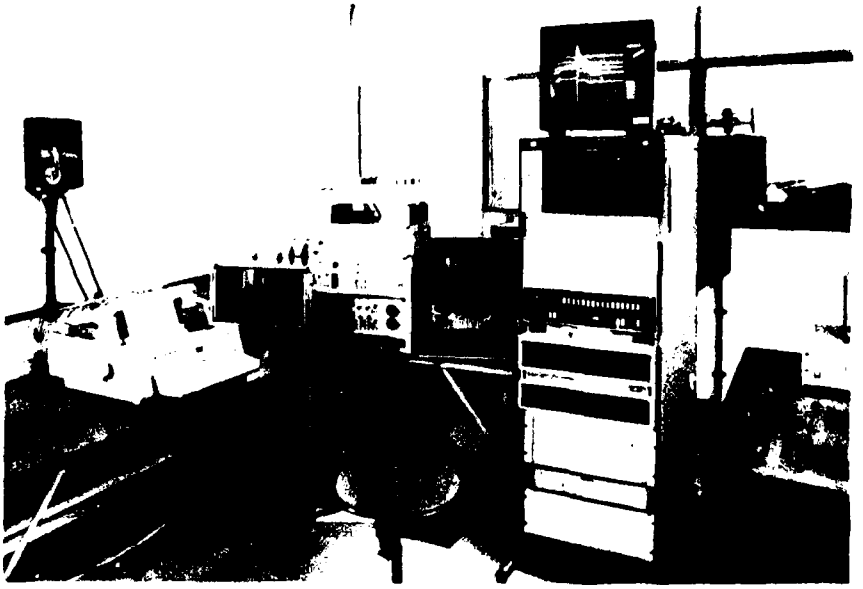


Figure 3 HP1000 6ft Rack, VDU Display and Raster Display

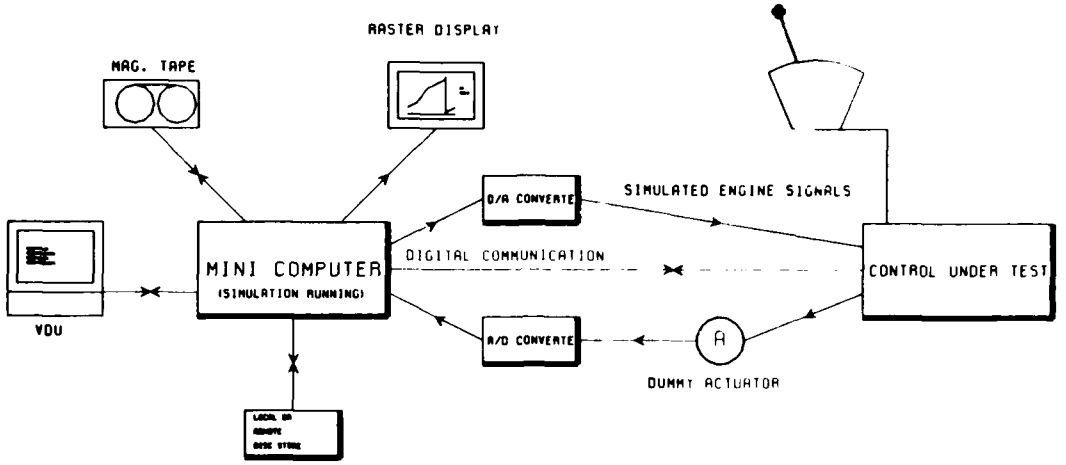


Figure 4 Test System with Simulated Engine

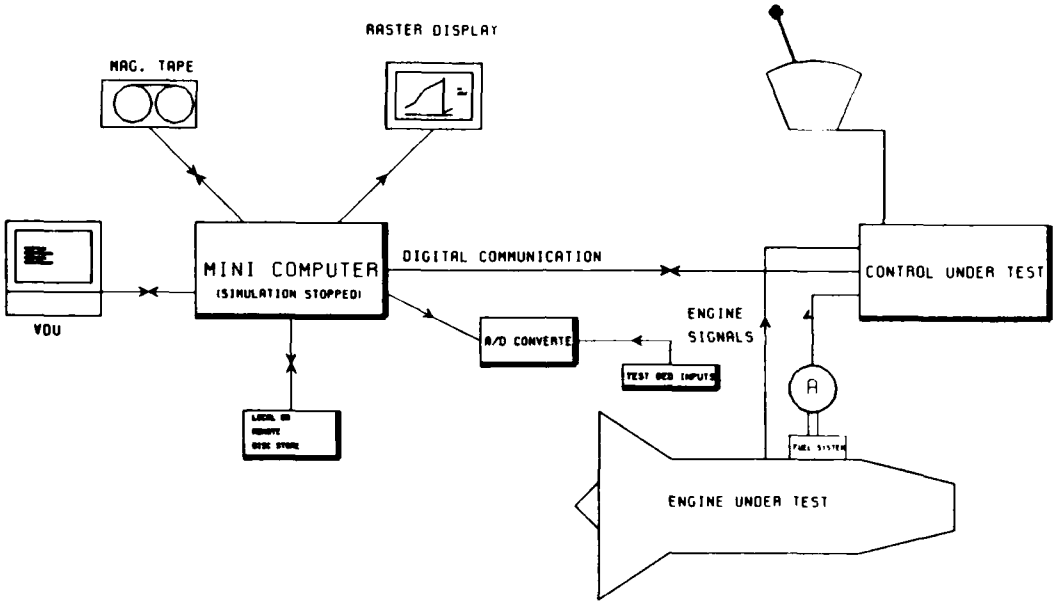


Figure 5 Test System with Actual Engine

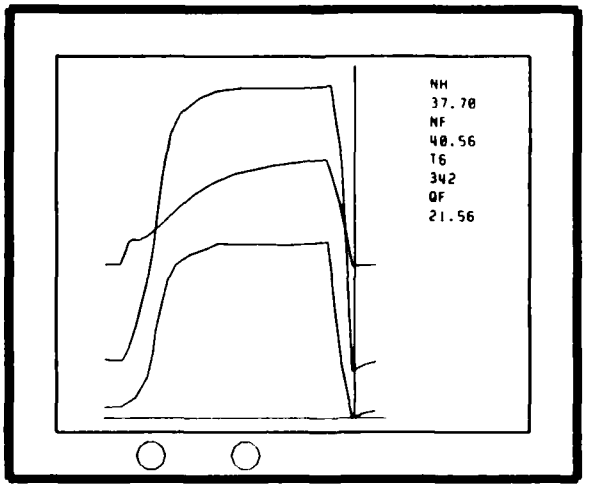


Figure 6 Raster Display During Engine Run

FAULT TREES AND SYSTEM RELIABILITY ANALYSIS WITH REFERENCE TO THE CONTROL OF AIRCRAFT ENGINES

AUTHOR: Mr. J. F. O. Evans, B.Sc.
Smiths Industries Aerospace & Defence Systems Company,
Winchester Road, Basingstoke, Hants. RG22 5HP.

SUMMARY

Control systems for aircraft engines are very precisely and stringently specified with respect to performance and safety. At the same time there is a real need to minimise cost and weight and to improve reliability.

These requirements may conflict unless the overall system organisation is very carefully designed and proven. It is not possible to prove that the safety requirements have been met within the acceptable confidence level by testing alone. Hence, testing needs to be backed up by safety analysis.

The paper outlines some current engine control systems organisations, the related analysis techniques, such as fault trees, and describes some of the special difficulties associated with analysing systems which include multi-task processors.

INTRODUCTION

The author has been involved in engine control system design for many years, firstly with Smiths Industries and now within Dowty & Smiths Industries Controls Ltd., whose joint activities in this field go back many years before their formal association.

In a previous paper (Reference 2) the history of this association was given with reference to the joint development of aircraft engine controls, particularly digital control systems. The intent of the present paper is to cover similar ground but to discuss in more detail than was then possible the design thougnt processes and the design verification processes which have been considered and used. Whilst referring to that previous paper the author takes the opportunity to thank his previous co-authors and to state that the other content of the present paper is his own and does not necessarily reflect their views or the views of Dowty & Smiths Industries Controls Ltd.

TECHNICAL REQUIREMENTS

The engine control system designer often finds himself in an unusual position compared with the designers of many other control systems. On the one hand only a relatively small budget in terms of size, weight, power consumption and system cost is available compared with the budget associated with, for example, automatic flight control, and yet, on the other hand, the safety implications of loss of engine power and even more so of secondary airframe damage due to engine power runaway, approach the safety implications of loss of flight control.

There are also differences in the evolution of the two types of control. The move from simple "rod and cable" operation of flight control surfaces was accompanied by a significant level of redundancy in the electro-mechanical systems. Engine control, on the other hand, moved rapidly to the sophisticated simplex (i.e. non-redundant) but highly hydromechanical systems to be found in the majority of present-day aircraft.

All such systems rely, for continuity of control, on the high integrity of well designed simplex parts. Many of the systems also rely on these simplex parts in order to avoid overspeed. This extensive reliance on single components is made possible by the grading of "authority", i.e. the degree to which the fuel flow can be affected by various parts of the system under fault or any other conditions.

Such a limitation of authority can be achieved in a hydromechanical system by finding points in the basic fuel control, such as a balanced beam in which an auxilliary control can modify the working point of the basic by an amount which is limited by an easily defined amount. This can be by the compression of a spring acting on a beam by movement of a carrier between fixed stops, by similarly limited movement of a fulcrum point, etc.

For some critical controls, e.g. protection from overspeed, many hydromechanical systems used conventional redundancy in the form of separate overspeed limiters, as shown in Figure 1.

With the development of engine technology demanding more and more complex controls the designers of the 50s and 60s started to use electronic trimming controls for Temperature Limiters, secondary shaft speed control and protection, and non-dimensional limiters. The controls used an extension of the limited authority trim of the earlier hydromechanical system in which an electric actuator compressed a spring acting on a governor beam or moved a beam fulcrum point.

These and [illegible] trimming controls are used on many current aero-engines, e.g. RR..1., [illegible], [illegible], Pegasus, etc.

Such trimmers [illegible] redundant in the sense that adequate control can be retained after [illegible] trimming function. This makes it possible to accept higher failure rates [illegible] could be tolerated for primary basic control functions. The [illegible] later service exploited this feature by applying sophisticated trimming facilities to a fairly conventional hydromechanical control system [illegible] called "Supervisory" (Figure 2).

This arrangement [illegible] the conflict between economics (or performance) and safety which, as mentioned earlier, is a dominant feature of the design of electronic control [illegible]. The benefit of the supervisory system is that it improves performance when it is operating but its failures cannot hazard the aircraft. The other side of the coin is the need to carry a complete hydromechanical system as well as the electronics and the fact that the very authority limitation which makes it safe can also prevent it achieving the potential control advantages of a digital control system. The full authority systems described later in this paper seek to resolve the safety issue without imposing limits on performance. At the same time, they allow the hydromechanical content of the sytem to be reduced and simplified.

While engine controls had been developing the electronic trimmer theme, the flight control designers developed electronic and electro-mechanical full authority redundant systems for such applications as auto-landing and fly-by-wire. From this flight control experience came a new world of semi-statistical design analysis techniques which were required to build up confidence in the system safety prior to actual flight experience and to be used for models by which to monitor the safety performance in service. Indeed, it soon became obvious that flight experience could never directly measure the demanded safety levels. Hence, when the time came to introduce full-authority electronic engine controls, the safety and reliability analysis was usually done from the background of this flight control experience rather than from the background of the traditional hydromechanical engine control. This led initially almost to the abandonment of the graded authority techniques and towards the multi-lane redundant controls in which a lane was considered to be wholly good or bad.

As illustrated by the supervisory system above, the electronic engine control system designer, whether working in the analogue or the digital field, was always under two pressures. Firstly, to continue to design with the economy of traditional hydromechanical control and, secondly, to meet ever increasing demands of statistically provable safety.

A major part of our research and design effort has, therefore, been put into a continuous search for the best system organisations available within the current technology. This has almost always led to the use of a mixture of full multi-lane redundancy and partial redundancy involving the principles of graded authority.

As a result our design team has been accused at intervals of being pedants, starry eyed idealists and downright specification dodgers! We continue to hope that these mild disagreements have always been taken in good part and recognised as inevitable if true advances are to be made by the interaction of independent design teams.

The teams concerned in this search for better and safer control of engines know that one of the essential keys to success is to maintain a good match between system design and safety analysis techniques. Hence, the available analysis requirements and techniques are a vital part of any discussion on design philosphy.

ANALYSIS REQUIREMENTS

Most people who design safety critical flight systems like to sleep easy in their beds, and anyway they fly about the world more than most. They need to have easy consciences and to avoid being rung up at all hours by people wanting yet more and more explanation about how the beast works. This means not only good and conscientious design but also a system whose operational safety is easily understood by the design team, the customer's acceptance team, the certification authorities and the maintenance and operational personnel. Indeed, if the system goes wrong either because it is wrong or because it is misunderstood or mishandled, there is only one guy they can and should blame.

Simplicity and the obvious completeness of all safety arguments must, therefore, be the order of the day, and this is no easy task with the complexity of modern control requirements.

SAFETY MODELLING AND ANALYSIS TECHNIQUES

By way of an example of safety modelling, let us look at the overspeed limiter of Figure 1. This could be modelled for safety analysis by either a "bottom up" approach, i.e. by considering each component in turn and building up a picture of the complete system; by a "fault analysis", i.e. by identifying the possible fault consequences and building a model of the ways in which these can arise; or by dividing the system into functional blocks, identifying their function and possible malfunction within the system and building a model by the analysis of these blocks and their effects on the system.

This latter approach is considered here because it has all the advantages of a systematic "top down" procedure, and especially because it enables the designer to achieve an "overview" of his design.

In order to attempt such a "top down" analysis of the limiter, we can reduce the system into two blocks at the top level, a range speed governor and an overspeed governor or limiter. We then need to consider how many states need to be assigned to each of these in order to model the system. We may decide that it is sufficient to define a "good" state and a "bad" stage, or we may wish to divide up these, for example by defining "failed high", "failed low" and perhaps "failed inoperative" systems. The criterion of sufficient definition will be to view the possible system in the light of the proposed model and to assess whether it is sufficiently complete for the purpose in hand. If we are interested only in the ability of the system to prevent overspeed we must be satisfied that there are no ways of the real system allowing or causing an overspeed which cannot be represented or which do not form part of another failure representation. If there are two ways in which the limiter can allow we do not need to represent them separately unless their system affects are different or their interactions with other parts of the system are different.

It is then possible to generate all the overall system states defined by this model by drawing a system state diagram as shown in the upper part of Figure 3. In order to use this diagram we must determine the consequences of all the fault permutations and it is clear that for a system with even a modest number of blocks some form of computer assistance will be required. Having done this the various system effects can be combined as shown in the lower part of the diagram.

An alternative approach is to redraw this diagram for a single system final state, e.g. High Speed or Runaway as shown in Figure 4. This presentation of the information is called a fault tree and is more usually obtained directly by inspection of the system. When a fault tree is used it must be remembered that it does not give a complete definition of the system with respect to anything other than the described fault.

When we believe that our model and our fault tree or logic diagram are a sufficient representation of the system we can proceed to a quantitative analysis. Taking each system block in turn we determine the probability of each of its states by conventional means. In the context of overall system analysis we will always prefer that our system blocks can be analysed by simple summation of the reliability of the component parts, but if this is not the case each block can be broken down recursively until the sub-blocks can be analysed in this way. It is particularly helpful if any common mode failures (to be discussed later) affect only the highest system block level.

The probability of arriving at any system state defined in the tree can be evaluated by starting at the bottom of the fault tree and combining the reliability probabilities at each junction. If the junction is a logical "AND" the combination will be by multiplication and if it is a logical "OR" by addition. (N.B. The exact form is $P_a+P_b-P_a \times P_b$ but the product term can usually be ignored.)

In the case of the system state diagram the probability of arriving at each state is computed similarly, starting at the top of the diagram. In this case, each time the states divide the probability of each new state is given by multiplication as for an "AND" gate. When states combine the total probability of the new state is found by addition as for an "OR" gate. A useful check on the system logic state diagram and the related calculations is to verify that the sum of all the final state probabilities is unity. Since a fault tree is by definition an incomplete statement of the possible system states it cannot be used for this cross-check.

In the case of our limiter, we might find that the probability of failure of the range speed governor is 0.001 per hour split into 0.0001 per hour upwards runaway, 0.0007 per hour downwards and 0.0002 per hour inactive. For the limiter the equivalent figures might be 0.0004 per hour total, 0.0001 per hour upwards, 0.0002 per hour downwards and 0.0001 per hour inactive giving the overall performances shown on the diagram.

This shows a probability of 2×10^{-8} per hour upwards runaway. This is a good time in the process to review our model, its analysis and the results obtained in the most critical manner possible. Do they ring true in the light of experience? Is there a factor which normally appears which is missing? In this case experience should tell us that we have omitted to consider the possible effects of common-mode failure, i.e. there is a finite probability of a single event affecting both the range speed governor and the limiter. Such an event might result from any service item, e.g. hydraulic or electric power supply, environmental problems, defective design or maintenance.

If such a common-mode effect is to be insignificant its probability must be less than $10E^{-8}$ showing that the penalty of trying to use any common blocks such as simplex pumps, fuel supplies or electric power is the need to keep the probability of a common-mode failure affecting critical system functions down to values many orders less than the individual block reliabilities. This is one of the most severe tests of the system designer's skills, in that he often has only experience and common sense to help him.

It is clearly desirable to be able to rationalise and quantify the common mode failure possibility. A simple means is to draw parallel paths to the final failure state, in other words define the possible common-mode failures and estimate their probabilities. This is a way of quantifying the effect but offers little other assistance to the designer.

The following alternative has been considered as a possible way of drawing on past experience and the records of previous equipments.

The unreliability of any system block will usually be made up of two parts, the unreliability and defects of an internal nature and unreliability and defects introduced by external events. Clearly, if two system blocks are vulnerable to the same external event then such an event will be a potential cause of common-mode failure. We now define a distribution vector, each element of which will represent the probability that once such an external event has occurred it will affect a particular combination of system blocks.

Each system block can then be examined in turn to assess the proportion of its unreliability which might be apportioned to external rather than internal effects. Summing these external effects gives an estimate of the total potential failure rate due external events which can be equated to the sum of the distribution vector. The form of the distribution vector may be determined by experience of similar equipments but a useful starting point is one in which each event is equally likely to affect any permutation of blocks.

Having done this the critical failure possibilities can be calculated in terms of the elements of the distribution vector and a hill-climbing or ball-park assessment made of the worst-case and mean critical probabilities when the form of the distribution vector is changed.

In the case of our limiter system, we may thus look up the past records of the probability of external events affecting the range speed governor and the limiter separately and from this form an estimate of system safety under common-mode failure conditions.

From the fault tree we can see that the probability of the system failing high due to non-common-mode failure is given by:

$$Psh = (Prh \times Plh) + (Prh \times Pli) = 2 \times 10^{-8}$$

Where Prh is the probability of the range speed governor failing high (0.0001 per hour)
Where Plh is the probability of the limiter failing high (0.0001)
Where Pli is the probability of the limiter failing inactive (0.0001)

Similarly, the probability of the system failing high due to common-mode effects is given by:

$$Pcsh = Pe \times (Drhlh + Drhli)$$

Where Pe is the total probability of external effects
Where Drhlh is the element of the distribution vector for the range speed governor failing high and the limiter failing high
Where Drhli is the element of the distribution vector for the range speed governor failing high and the limiter failing inactive

Looking at the system logic diagram we see that there are two blocks each with 3 failure possibilities. If an external effect is equally likely to affect any one of the 16 possible combinations of block failures, them:

$$Drhlh = Drhli = Pe/16$$

Thus the common mode failure rate to high can be rewritten:

$$Pcsh = Pe \times (Drhlh + Drhli) = Pe/8$$

Therefore, if Pcsh is to be of the same order as Psh it follows that we must keep Pe down to about 1×10^{-7} per hour.

In order to make this analysis rigorous we must show that if the unreliability of any block is divided between the internal and external effects we must be able to sum the probabilities of the states due to the internal component of the unreliabilities together with the probabilities of the states due to the external or common-mode unreliabilities and find unity.

This can be done by inspection, we know that the sum of the state probabilities for Pe = 0 is unity, hence, for finite Pe the same summation must give 1 - Pe. Since the sum of the distribution vector is unity by definition then the sum of the products of each element x Pe must be Pe. It follows that the sum of the state probabilities if 1 - Pe + Pe or unity as required.

These results indicate that either the external component of the unreliability must be only a very small fraction of the internal component or the distribution vector must be very different from the above if common-mode effects are not to predominate. In practice, every possible attempt is made to change the distribution vector by the use of:

(a) Isolation between blocks, e.g. fuel filters, separate electrical power supplies.

(b) The use of dissimilar hardware blocks in which a given common cause may have different effects or the same effect at different times.

(c) The reduction of the probability of external effects causing failure by good filtration, good power supply design and generous stress margins.

(d) Design in such a way as to make the time separation of failures induced by a single external event as large as possible. At the same time to attempt to achieve correct system response to a single failure as fast as possible.

Within an established technology, e.g. hydromechanics or analogue electronics, there has been built up a working knowledge of the effectiveness of these techniques. In other areas, e.g. digital electronics, our knowledge is often less certain. We can reasonably, however, use the same techniques provided we design with wider margins.

ENGINE SYSTEM DESIGNS

In the early days of redundant systems it was felt that there was no real possibility of the industry accepting engine control system designs which relied on more than two complete paths or "LANES" of control. The first system proposed, which was for the PS50 engine, consisted of two sets of input transducers, two digital computers and a simplex actuator of the proportional solenoid type for the main engine control and permanent magnet stepping motor actuators for the various afterburner fuel flows and nozzle control (Figure 5).

The two computers exchanged transducer data via optical unidirectional data links and used average data for control provided that their own data and the data from the other "LANE" were within a specified tolerance. If this comparison was not within the specified tolerance the computer used its own data instead of the average data. The two computers also exchanged their output data and again made a comparison. If both agreed that the comparison was good one lane drove the actuator controlling the engine. No provision was made for controlling the engine from the other lane, or for determining which of the two lanes was faulty. It followed that if either lane detected a failure, the actuator was disconnected and the overall system degraded into

(a) hydromechanical back-up control of the main engine, or

(b) frozen afterburner flows and nozzle positions which could be manually shut down.

This system was successfully operated on a test bed and shown to have all the necessary safety features except the ability to continue operation after the first failure. A simple fault tree of the aircraft system can be drawn showing that if the automatic control fails then the ability of the aircrew to absorb the additional workload and control the engine manually becomes important to the aircraft safety.

This overall solution was therefore rejected in favour of alternatives which would maintain automatic control after a first failure. Considerable effort was thus put into systems which still only required two "LANES' of control and yet could isolate a fault and reconfigure to single lane operation safely.

Conventional fail-operational redundant controls, such as autopilots or fly-by-wire systems, tend to use 3 or even 4 lanes and some form of majority voting procedure or, in the case of four lane systems, comparisons by pairs. If a fail-operation system is to have only two physical lanes it must have some form of 'phantom' lane to achieve fault identification and various such 'phantom' lanes have been considered by the industry.

One approach is to rely on internal monitoring of the two lanes using either computer self-test procedures or a minimal monitor lane in parallel with each operational lane. The first approach is always suspect, how do you know that the self-test will work? and the second is only a cut-back version of the 4-lane approach.

A better approach is to extract additional information from the system run-time history and to use this to back up self-test information. This was used in our second system run on an ADOUR engine under test bed conditions.

Examination of the PS50 system showed that additional attention needed to be given to three main areas of design:

(a) The number of simplex system blocks must be minimised.

(b) The reconfiguration mechanism must be as reliable as the remaining simplex blocks.

(c) Any failed blocks must be reliably excluded from the system.

It was found that (a) could be achieved if the simplex main engine and nozzle control actuators were replaced by dual drive units as shown in Figure 6. In this system, as in all DSIC systems, isochronous governing was achieved by using the stepping motor as an output integrator. This allows the digital computers to recover rapidly from any power failure causing temporary loss of stored data by minimising the dependance of the system on such data. The relationship between actuator position and fuel flow was stabilised by the fuel valve power servo, so that in the event of digital control shut-down, the fuel flow remains constant until recovery action is taken. In the case of afterburner control, a known and accurate relationship between actuator position and fuel flow is maintained as a faster alternative to fuel flowmeters in the delivery lines.

The system operated with both lanes actively sharing control until the first failure. Inputs were exchanged and averaged as in the PS50 system and output data was also exchanged as before. If both lanes agreed, both motors were driven and the mechanical outputs consolidated by summation in a differential gearbox. If the two lanes disagreed the idler cage of the differential rotated operating a mechanical switch which "froze" the system by disconnecting both motors.

Both lanes then entered a self- and cross-check procedure and based on the results of these chelf-checks control was re-established by one lane.

A simple system state diagram is shown in Figure 7. This diagram was analysed and it was decided that a lane would be reconnected if its own self-test indicated it to be good and the self-test in the other lane indicated that it was faulty. Wrong connection can only occur if a good lane self-tests as bad AND a bad lane self-tests as good, the first of these being considered to be very improbable.

In practice, further tests were introduced to separate computer faults from input signal transducer and actuator faults and the system analysis considered these as well as computer faults. It was noted that some second order effects were sufficiently common to outweigh some secondary effects and as a result all the possible permutations of failure were included in the computer analysis.

Empirically, this seems to be a reasonable solution but to check it out the system was analysed in a manner analogous to the method used earlier in the paper for common-mode faults. In this case a fault within a single lane due to computer error was treated as a common mode fault coupling the control function and the self-test procedures. Each element of the distribution vector defines the probability that a computer fault will cause a particular combination of failures in the various computations within the computer. The system was analysed for a wide range of distribution vectors and it was found that the worst system performance occurred when it was equally likely that a computer fault affected control functions and self-test results as shown in Figure 8. Systems with higher or lower ratios of these probabilities always performed better and this was taken as the worst case design.

The above analysis can be stated relatively simply in words. If on one hand every computer fault which affects the computer's ability to control the engine also affects the self-test, then clearly all computer faults and other system faults will be detected and the system will always revert to the correct lane. If, on the other hand, computer faults which affect the control capability never affect the computer self-test the self-check will always report "computer good" and the system will fail to revert. A mid-way situaton will sometimes cause incorrect reversion.

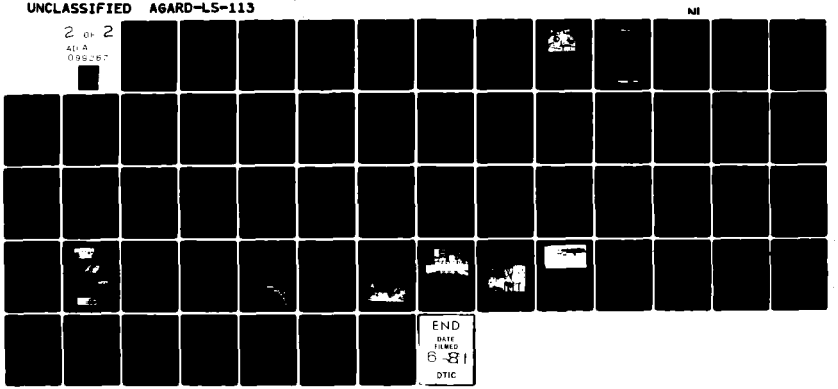
Such an analysis may give the designer confidence in his design approach but it may take many thousands of test and flight hours before such a statistical analysis can be given a high enough confidence (low variance) for it to be used in the certification of a flight safety critical system. This is to be compared with the relative ease with which the common-mode problems can be accepted as being reasonable risks in existing technology systems. If such self-test techniques are necessary for the advancement of flight-critical engine control, then we must accept very high test and evaluation costs or the trial of the system idea in a less critical application.

This system was built and tested on an engine test bed as shown in Figure 9. Full fault recovery tests, including tests in which faults were introduced in the middle of an acceleration or afterburner acquisition and throttle movements, showed the ability of the system to react correctly to induced faults. Later in the test some problems were experienced in the mechanical consolidation following suspected fuel contamination showing the need for late consolidation and rigorous analysis of all possible common-mode effects.

In many other respects, this second system made dramatic progress towards the ideal goal of a reliable, safe, full-authority control. Two new principles aided the avoidance of common-mode failures:

(a) The synchronisation of the two computers was reduced from word level in the PS50 system to iteration level, thus making possible the use of two computers running at approximately the same speed and with similar programs but PERFORMING IDENTICAL TASKS. Thus dissimilar software would have been possible (for cost reasons it was not used, but little progress was made in the avoidance of common algorithmic errors since algorithms which have to result in essentially identical numerical output are unlikely to be independent.

AD-A099 267 ADVISORY GROUP FOR AEROSPACE RESEARCH AND DEVELOPMENT--ETC F/G 9/2
MICROCOMPUTER APPLICATIONS IN POWER AND PROPULSION SYSTEMS.(U)
MAR 81
UNCLASSIFIED AGARD-LS-113 NL



(b) The control design was, however, significantly redundant, for example, major engine mismatch due to errors in the nozzle loop was protected by a pressure ratio control using different input signals and actuation.

OTHER SYSTEM CONFIGURATIONS

The supervisory control and the fully duplicated system design represent the two poles of current system design. Between them lies a range of full authority systems using limited application and technology which may be dissimilar.

PEGASUS ENGINE CONTROL

The existing system on the Pegasus engine uses an electronic limiter and also a redundant and much simplified hydromechanical flow control for use after a failure in the main hydromechanical control, see Figure 10. Full authority digital control of the Pegasus engine has been demonstrated by Dowty and Smiths Industries Controls Ltd., together with both hydromechanical and digital electronic reversionary control. Both provided automatic changeover after a failure of the full authority digital control with little or no transient power disturbance.

The reversionary systems are simpler than either the existing hydromechanical control or the full authority digital control. They are primarily intended to enable the aircraft to be landed safely after a failure in the main control although, in the case of the digital reversionary control, mission completion is possible and in all cases safe reversion is possible in the hover mode.

Figures 11 and 12 show the levels of performance achieved during the early test-bed development. Figure 11 shows transient free reversion to the emergency hydromechanical system and Figure 12 transient free reversion to the digital reversionary system during an acceleration.

HELICOPTER ENGINE CONTROL

The electronic arrangement demonstrated for Pegasus applies one or other of the electronic control outputs to the same flow control valve using a single drive motor unlike the arrangement for the Adour system, in which the outputs from two motors were combined in a mechanical differential to drive a single valve. In the Helicopter system now undergoing test bed trials, two motors are again used but the mechanical differential has been replaced by a differential fuel valve. This gives the maximum isolation between the two digital electronic "LANES" and avoids the possibility of common-mode problems due to switchover failures or mechanical failures in the differential. In other respects the system structure is somewhat similar to that described for the Pegasus engine. Different microprocessors are used for the main and digital reversionary controls. In this case the engines are used in pairs to drive a single rotor system through a gearbox and significant additional system checking can be achieved by comparing the two engines.

The system allows power modulation to be provided by one or other engine after a first failure of the main control over large parts of the flight envelope by holding the other engine at a fixed power setting. The infrequent need for power changes on the second engine make it possible to use a very simple reversionary system and yet, in terms of the overall vehicle, to retain unimpaired performance.

The full system arrangement is shown in Figure 13. The hydromechanical control section of both the Pegasus and the helicopter control system is much simpler than a conventional hydromechanical control. Figure 14 shows the valve arrangement for the helicpter control illustrating the method of consolidation of the electrical inputs at the actual valve.

ELECTROMECHANICAL INTERFACING

The simplex possible fuel control design would comprise an actuating motor and a tap. In practice, this cannot be achieved because of the very large power gain involved. The maximum practical power output from an electronic system is perhaps 50 watts and many designers feel that the optimum is nearer 5-10 watts. The high pressure fuel delivered to the engine represents some hundreds of horsepower, and to date no safe direct operating system has been proven, indeed, as many as three stages of hydromechanical amplification have been used in some systems.

As a result of the need to provide some hydromechanical amplification it is often possible to achieve some form of flow metering in the hydromechanical system with minimum additional complexity. For some applications pressure sensing bellows can be used to provide a Fe/P1 or Fe/P3 relationship between actuator movement and fuel flow.

In the PS50 and Adour engine runs, electrical pressure transducers were used instead of the above approach and we believe this is still acceptable for afterburner fuel scheduling controls where the pressure signals are used in a complex manner more suitable to electronic than hydromechanical implementation.

For a helicopter application with restricted altitude and Mach No. requirements satisfactory control can be achieved using the Fe/P1 fuel metering interface, whereas for the Pegasus, with its wider operational envelope, an Fe/P3 interface is more appropriate.

The advantages of this approach is that the hydromechanical part of the fuel system can provide significant fast inherent surge protection which could only be provided by the electronic system by increasing the actuator slew rate and power. At the same time the need for electrical pressure transducers can often be eliminated.

ACTUATION SYSTEMS

The actuators used in the systems described in this paper are fuel-immersed and use stepper motors in preference to other types because:

(a) The drive signal is pulsed and hence easy to interface with digital equipment.

(b) The actuator does not then need a "velocity feedback transducer", thus eliminating the classic runaway failure due to loss of such feedback.

(c) Drive circuit failures seldom, if ever, result in runaway, the typical result of drive failure being to stop the motor.

Generally, a "fail frozen" or "fail fixed" response is preferable to runaway but there are a few controls, such as Inlet Guide Vane control, where failure to stop or other fixed position is preferred. In such a case, a torque motor actuator is preferred.

Experience with actuators on engines indicates that large excess power capability is highly desirable to overcome occasional high friction loads in valves due to various causes. This leads us to avoid the mechanical differential approach of the Adour engine actuator in favour of simpler units, such as those used for the helicoper application. These have the additional advantage of later consolidation and hence lower common-mode failure probabilities.

FUTURE ENGINE CONTROLS

The author does not believe that with these third generation controllers entering service there will be a finalisation of design but rather that the system designs will continue to progress as in the past.

It will be clear from some of the earlier discussion that there can be no instant solution to the problem of quantifying common-mode failure probabilities or computer self-test capabilities and, therefore, there must be a continuing move towards system designs which place the minimum reliance on determining these. This, in turn, demands multi-lane systems with the greatest possible dissimilarity in hardware, software, power sources, environmental susceptability and so on. The Pegasus and especially the Helicopter systems described contain significant moves in these directions but are limited by our current ability to configure systems with widely dissimilar lanes.

This inability is based on the need in current systems to compare the outputs between "lanes" or between parallel elements in a single lane. If such a comparison is to be made it must be:

(a) Made in a form which is mutually recognisable or recognisable by an external arbiter.

(b) Made in a time scale which enables faults to be eliminated before they have an adverse effect on the engine.

(c) To minimise implementation dependency it must be made using parameters which can be determined in terms of permissible deviations from the normal control of the engine.

Requirement (a) tends to prevent dissimilarity because it is an inconvenience, requirement (b) tends to force frequent comparisons as a result of the fast-moving actuators and wide dynamic ranges in engine control and requirement (c) tends to be the Cinderella and observed only when convenient.

A study of the history of the designs described in this paper indicate that an attack on (b) might, as it were, unlock the problem and make possible new system designs.

Progress in these new designs has already been made. Whislt investigating afterburner control techniques it was realised that it would be of considerable value to the controller for it to compute a predicted engine working point a few tenths of a second in advance. This would enable the control to identify trajectory errors in time to avoid overshoots. At the same time such a technique can simultaneously predict overfuelling, overspeed, overtemperature or any other relevant future limit exceedance and thus achieve significantly smoother take-over from one control to another. By a simple modification it can also establish the permissible maximum actuator movement for a given time ahead. Conventional controllers usually have to wait until the system is very close to or at a limit before initiating any corrective action. Similarly, any fault condition may be at the point of producing an exceedance when detected or it may be some way away. If the system cannot distinguish between these two cases it will need to eliminate the fault immediately for safety, typically well within one iteration. In conventional systems, this usually requires that multiple lanes check each other or be checked very frequently leading to close synchronisation and close data identity in conflict with the need for dissimilarity.

With a predictive limit, however, it is much easier to operate two control lanes asynchronously only exchanging or passing to an external arbiter at intervals the future safe actuator movement.

Asynchronous operation can have far reaching consequences, one can, for example, use two totally different control lanes with different computers, different software and even different control algorithms. It is possible to mix analogue and digital and even hydromechanical controls in a number of redundant "lanes" thus giving back to the engine system designer the freedom of economical and safe design he nearly lost when the statisticians first joined in the design process.

CONCLUSION

The author hopes that this paper will have prompted thought in at least three main areas of system design. Firstly, a recognition of the fact that we have only been able to live with the problems of common-mode failure in existing systems because of our practical experience with the manifestation of this effect in current technology hardware. Secondly, a recognition that some of the problems of multiple information paths through computers resemble the earlier problems of common-mode failure and, thirdly, that the cure is the old and tried cure of maximum dissimilarity.

The configuration of systems with such wide dissimilarity will, as ever, exercise the designer's skills to the limit and we must continue to learn to use the power of the new computing technology to improve engine systems.

ACKNOWLEDGEMENTS

The author wishes to acknowledge the work done by his colleagues within Dowty and Smiths Industries, especially his previous co-authors Mr.E.S.Eccles and Mr.E.D.Simons, and also Mr.K.Harris, Mr.P.Matthews and Mr.R.Pitfield, who have been deeply involved in the work reported here.

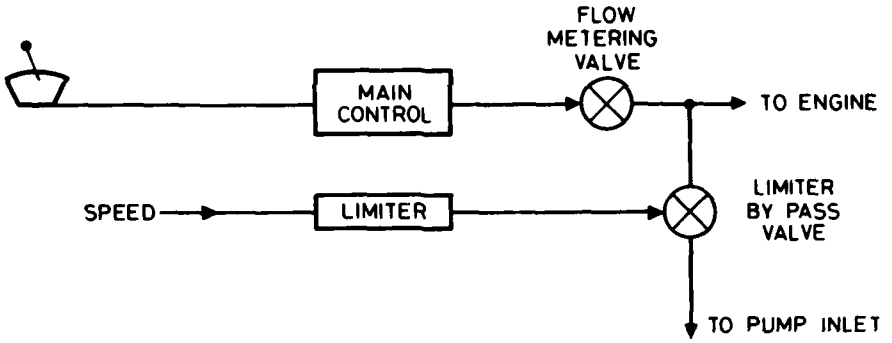
He also wishes to thank the directors of Dowty and Smiths Industries Controls Limited for permission to publish this paper although the views expressed are his own.

Part of the work described in this paper has been carried out with the support of Procurement Executive, Ministry of Defence for which the author also offers his thanks.

REFERENCES

Reference 1 Fault Tree Analysis to Anticipate Potential Failure. R.L.Eisner, ASME 72 DE 22.

Reference 2 Advanced Control Systems for Aircraft Powerplants. E.S.Eccles, E.D.Simonds and J.F.Evans, AGARD Conference Proceedings No.274.

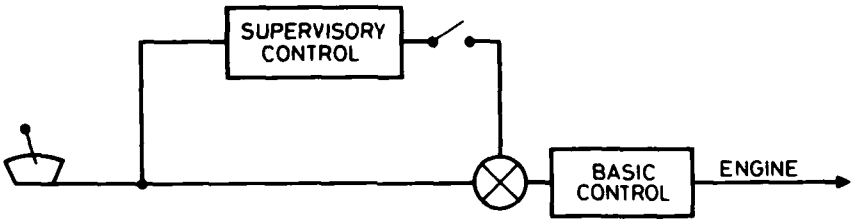


— WIDELY USED ON MANY ENGINES.

— MAY USE TWO VALVES AS SHOWN OR A SINGLE VALVE.

— REQUIRES PERIODIC CHECKS FOR LATENT FAILURE.

Figure 1 Overspeed Limiter



ADVANTAGES

— EASY TO CERTIFICATE.

— ELECTRONIC CONTROL NOT A DESPATCH ITEM.

— SYSTEM AVAILABILITY REASONABLY HIGH.

— BENEFITS OF SOPHISTICATED CONTROL WHEN AVAILABLE.

DISADVANTAGES

— RELIABILITY/MAINTENANCE/WEIGHT BURDEN.

— FULL BASIC CONTROL CAPABILITY NEEDED.

Figure 2 Supervisory System

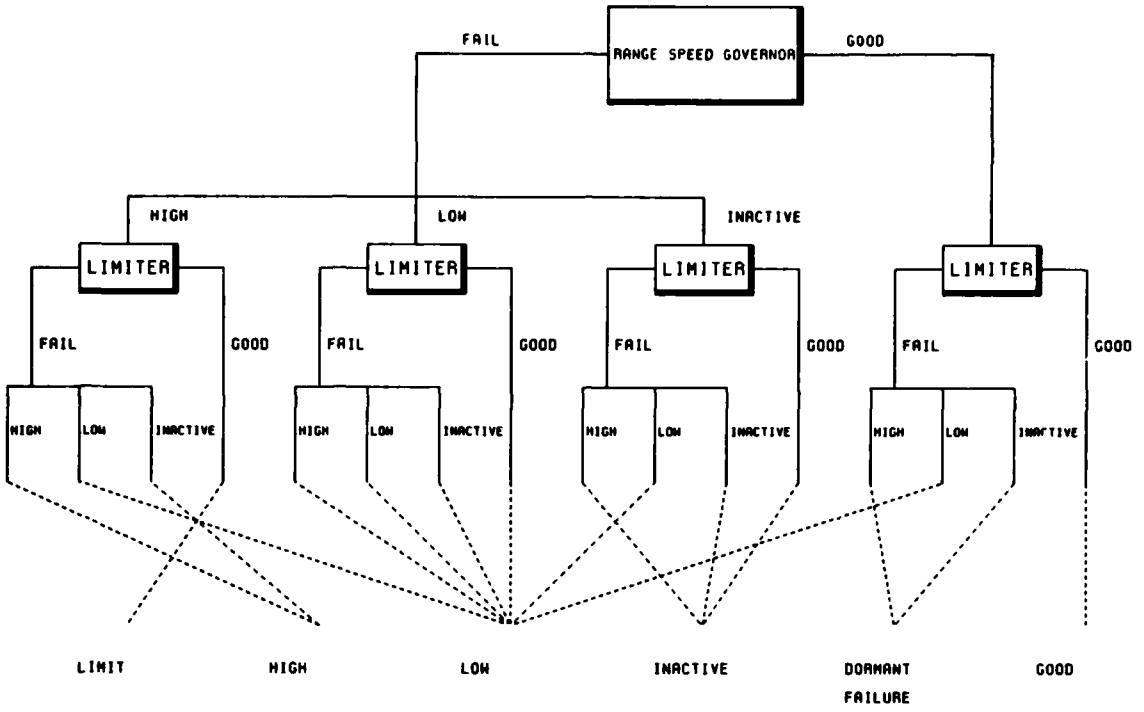


Figure 3 System State Diagram

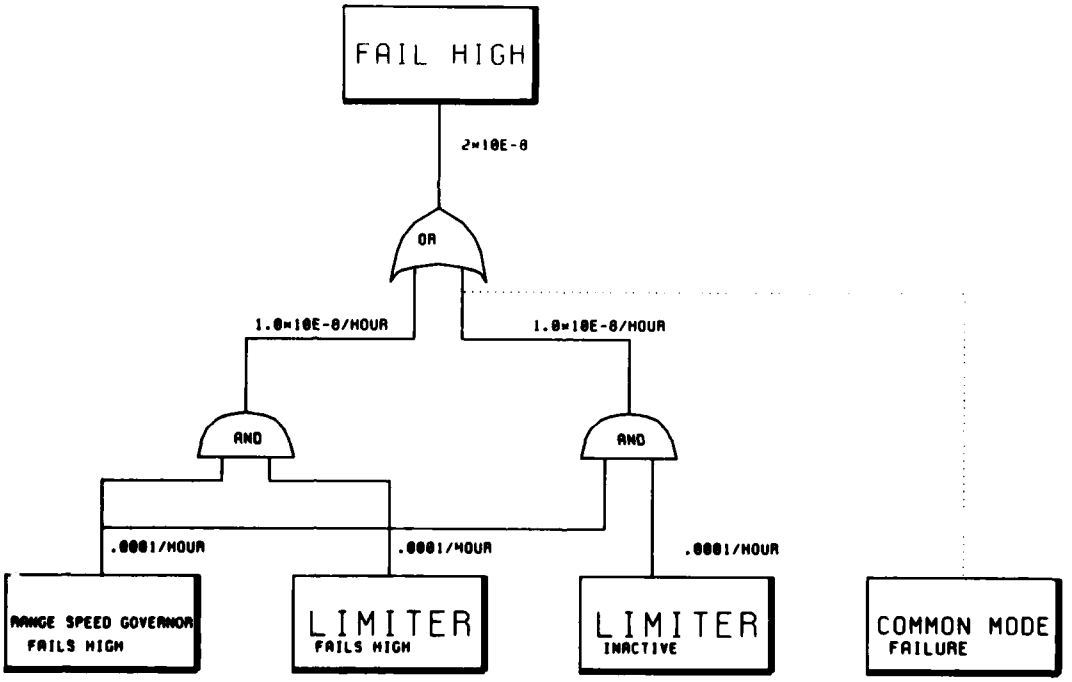


Figure 4 Fault Tree

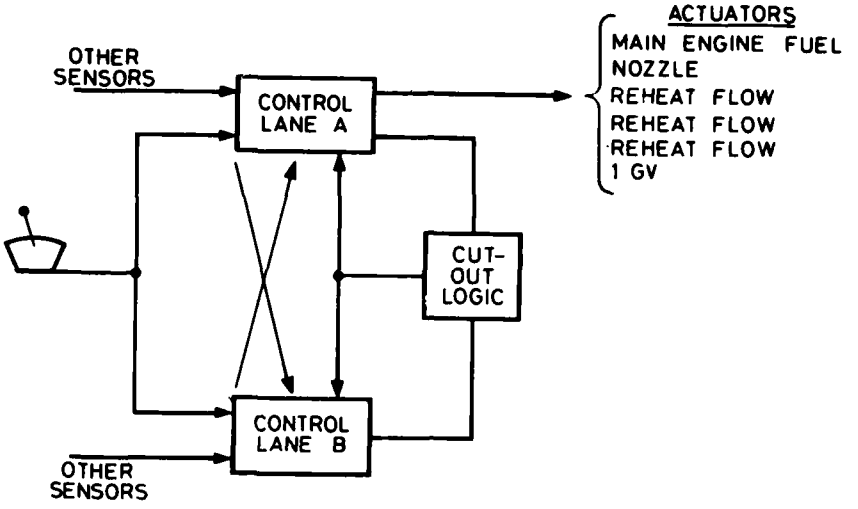


Figure 5 PS5O Engine System

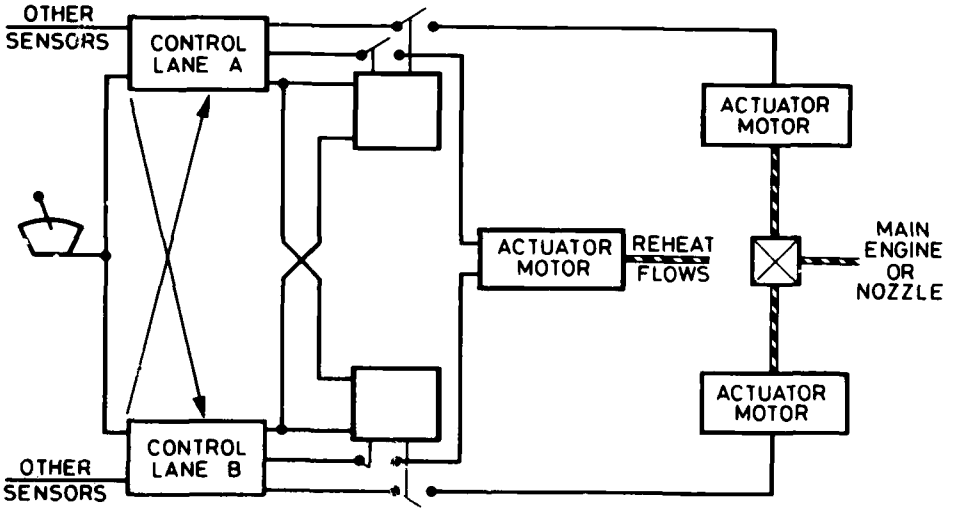


Figure 6 Adour System

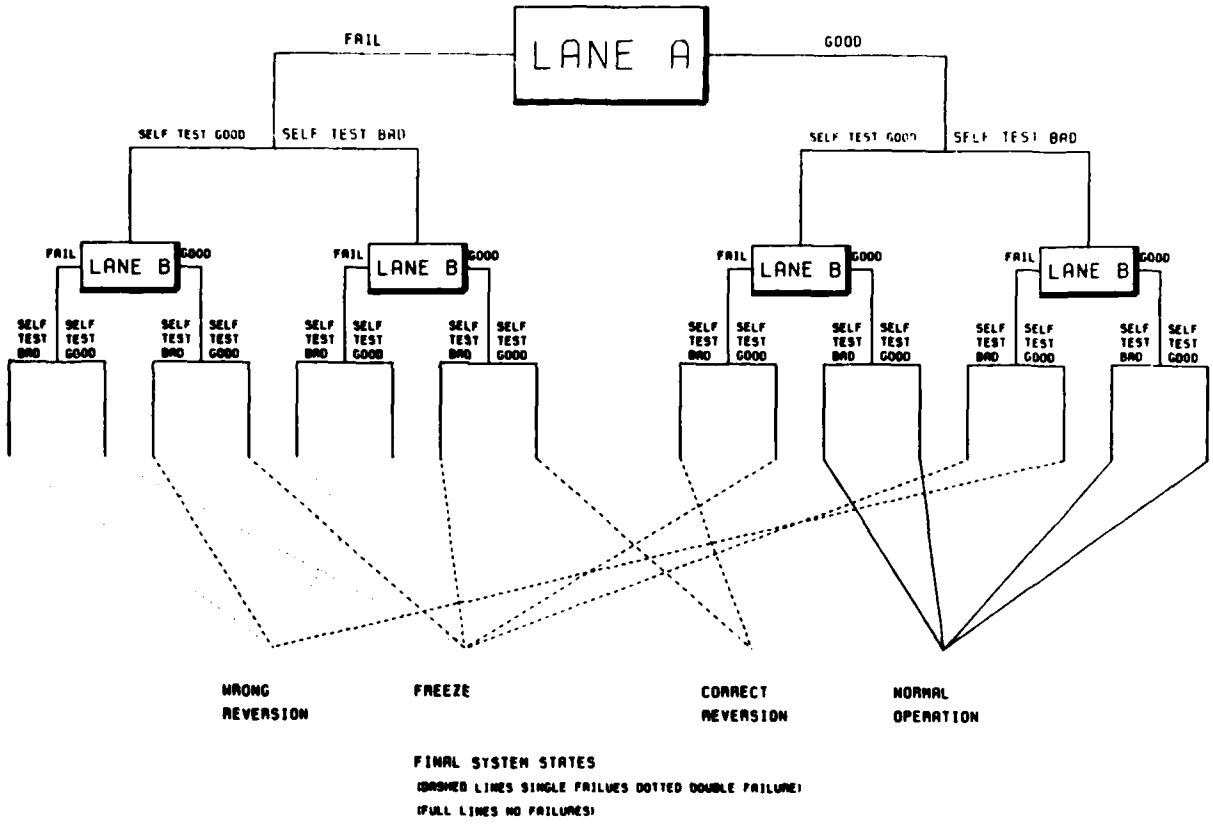


Figure 7 Adour Duplex System State Diagram

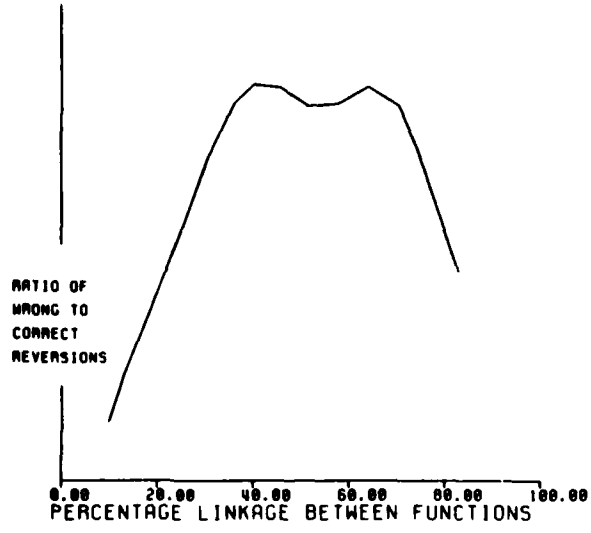


Figure 8 Effect of Fault Coupling

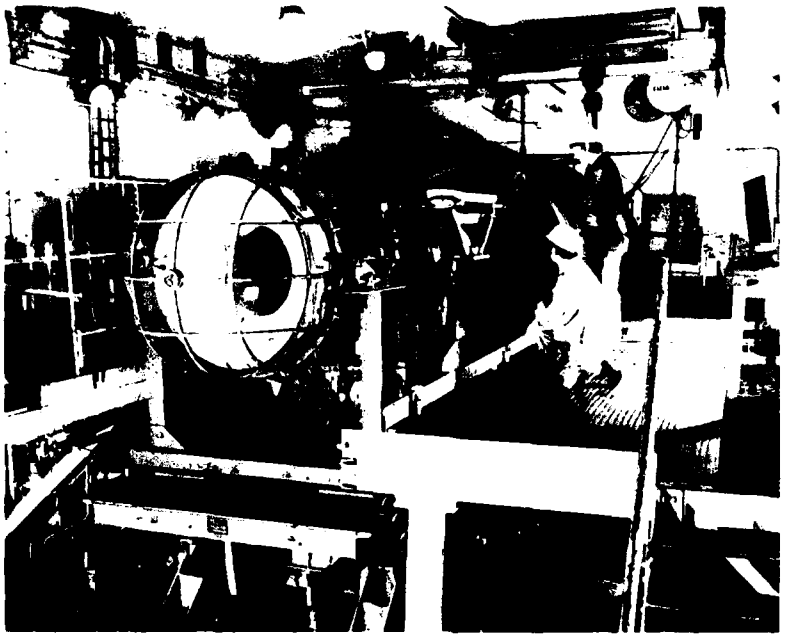
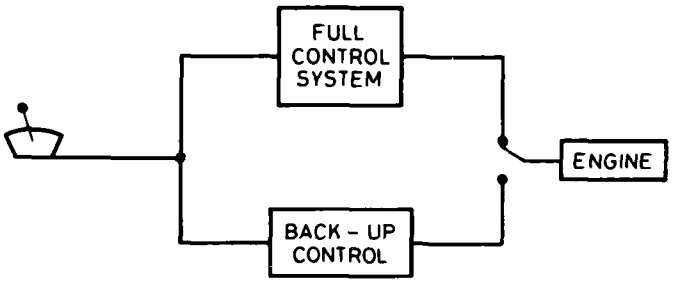


Figure 9 Adour Engine on Test Bed



— MAJOR CHANGE IN PERFORMANCE ON FIRST FAILURE.

— CHANGEOVER MAY NEED TO BE AUTOMATIC.

— LEVELS OF SAFETY DEPEND ON SERVICE AND TIMES AT RISK.

— FULL CONTROL AVAILABILITY NEEDED FOR DESPATCH.

— BACK UP AND HYDROMECHANICAL CONTROL SECTIONS CAN BE SIMPLIFIED.

Figure 10 Pegasus Engine Control

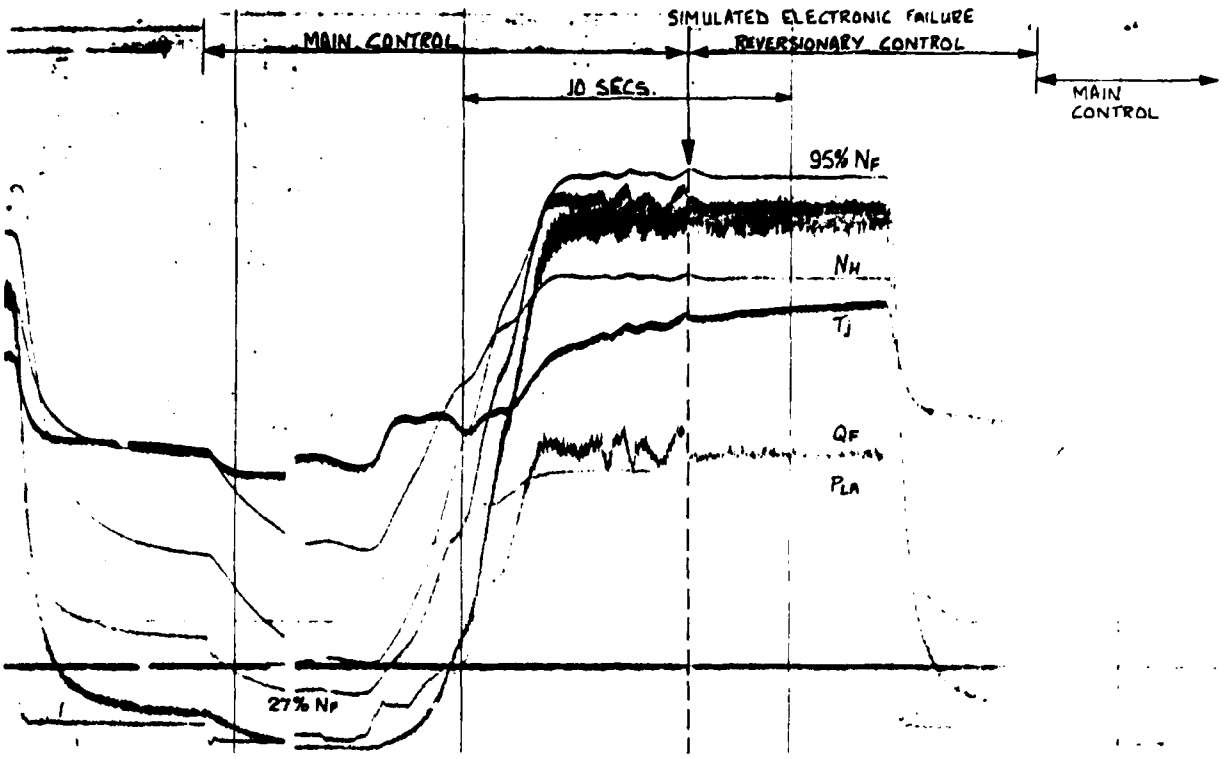


Figure 11 Simulated Failure of Main Electronic Channel with Automatic Change to Manual Reversionary Control at 95%N_F

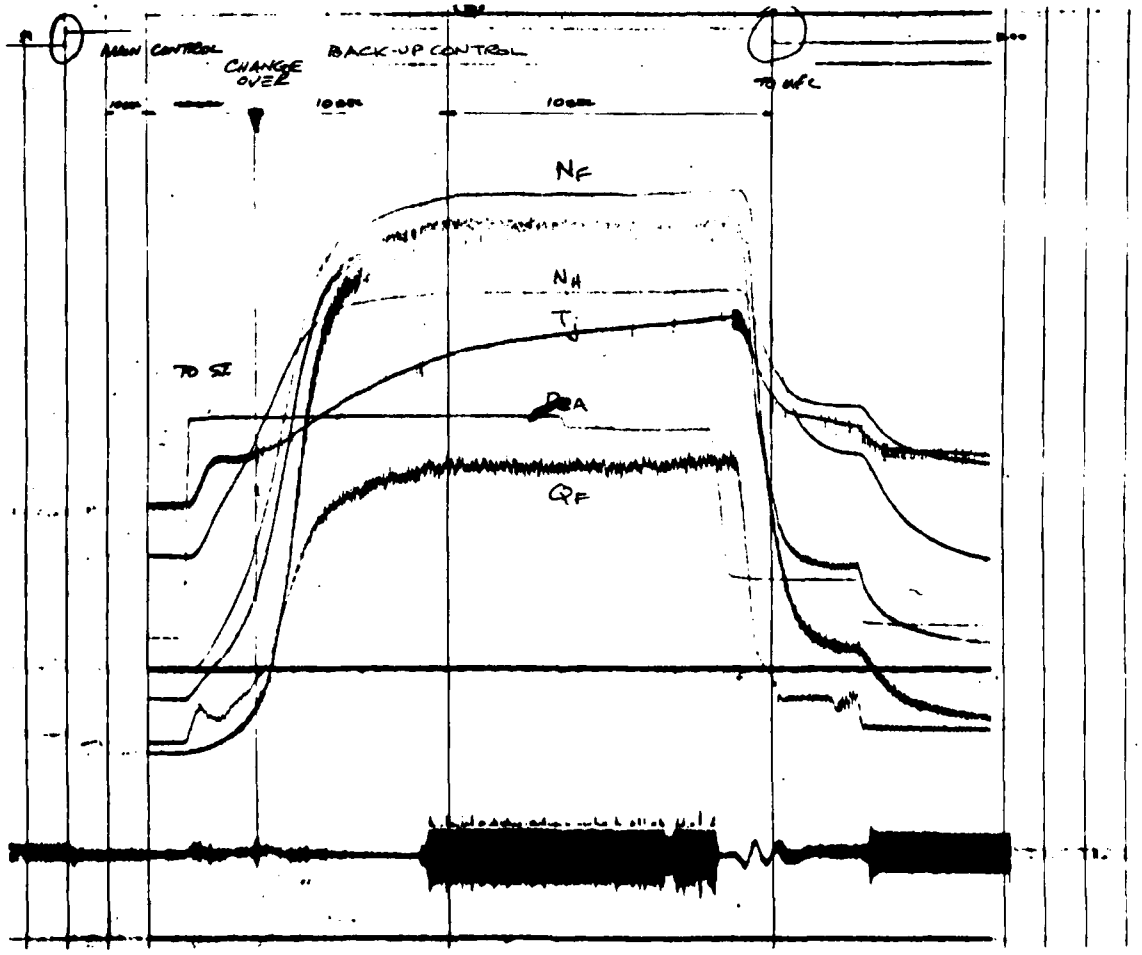


Figure 12 Automatic Changeover During Acceleration

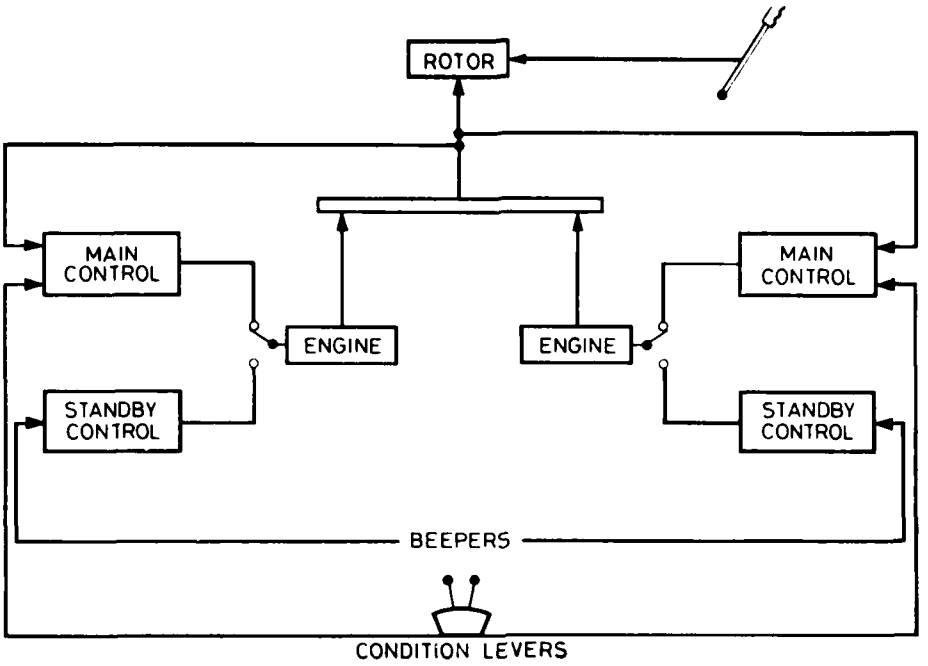


Figure 13 Twin Engined Helicopter System

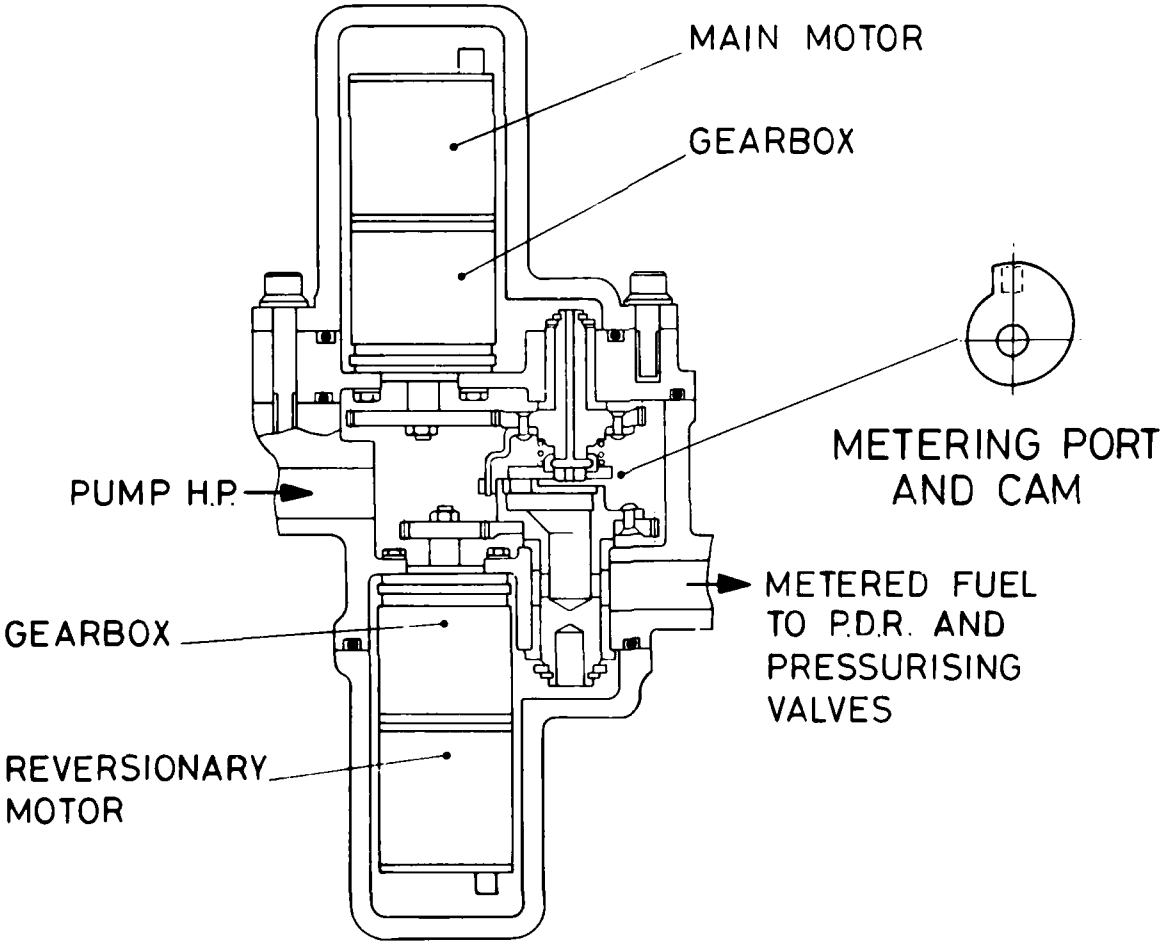


Figure 14 Section through Metering Valve for Helicopter Fuel Control

FAULT TOLERANT SOFTWARE PROGRAMMING

Vito Amoia - Mariagiovanna Sami
Istituto di Elettrotecnica ed Elettronica
Politecnico di Milano
Piazza L. da Vinci 32
20133 MILANO
Italy

Abstract. The reliability of a control system equipped with a µ-processor is dependent both on safe operation of its hardware and on ability of its software to correctly implement the operations required. We briefly discuss the two traditional approaches: fault-tolerant and fault-intolerant; some significant examples of system fault tolerance implemented by software mechanism are included (SIFT, C. vmp, PLURIBUS). Basic techniques for reliable software are then presented. Validation and testing methodologies are outlined. Emphasis is given to the most recent fault tolerant approach (Randell, Avizienis, etc.).

1. INTRODUCTION

High reliability requirements for data processing systems - first introduced in aerospace applications - have spread from areas where the continued operation of the computer is critical for accomplishing the mission to areas in which continued operation assumes a "social" meaningfulness (e.g. telecommunications) and presently also to any area where not so much the mission, but rather maintenance and down-time costs are critical.

Obviously, such a wide range of possible applications leads to varied requirements; we try to list some of the main ones:

- reliability, defined as the probability that no faulty operation occurs in the time interval 0,T
- availability, i.e. the probability of correct operation at time T
- safety, i.e. the probability that in the time interval 0,T the system is either down or correctly operating
- credibility, i.e. the probability that at time T the system is either down or correctly operating.

The increasing complexity of the systems has made it more and more difficult to evaluate the above parameters and even to represent the systems by means of adequate models. Moreover, the incidence of software upon reliability of the whole system has been growing; as representative instances of such trend, it might be recalled that in the first trial year of ESS1 about 90% of the causes of system failures were tracked back to software. As a matter of fact, the difficulty (if not impossibility) of complete debugging for a great software project like OS 360 led to studying software engineering techniques and criteria for design of reliable software.

Additional problems are presented by the reliability of multiprocessor structures; the interaction of hardware and software in an intrinsically reconfigurable environment easily creates new sources of faulty behaviour and increases the difficulty of state definition for subsequent recovery procedures. On the other hand, the same characteristics of dynamic reconfigurability and of task redistribution make such structures very attractive when high availability is required.

Two approaches have been taken for the implementation of reliable computers: the "fault-intolerant" one and the "fault-tolerant" one.

With the "fault-intolerant" or "zero-defect" policy, the aim is to obtain faultless components for the system: this has been notably the first (and still the most widely used) approach to software reliability.

On the other hand, the "fault-tolerant" approach allows the existence of faulty components and reaches the pre-assigned reliability through some form of redundancy and suitable reconfiguration policies. Adopted from the first for achieving reliable computer hardware, this approach has recently been suggested also for software. Even if comparatively few instances of reliable software implemented through redundancy have been presented until now, the intermediate steps in the fault-tolerant approach - i.e. testing, diagnosis, error confinement - have been followed.
We briefly consider first the use of software solutions for achieving system-level fault tolerance in some advanced architectures; while no particular actions concerning software reliability are present there, the overall approaches seem very interesting and some guidelines could be reasonably deduced for software.

Then, software validation will be examined both in the aspects presented by the fault-intolerant approach and through definition of testing strategies. Then, design criteria will be introduced. Some particular problems proper to real-time multiprocessor systems will be pointed out. Last, the modeling problem will be briefly afforded.

2. SOFTWARE MECHANISMS FOR SYSTEM FAULT TOLERANCE

The evolution in design of fault-tolerant architectures has led to increase the functions implemented by software. Early systems achieved the main functions - from testing to error confinement and to reconfiguration - through hardware mechanisms only: the SATURN IV guidance system and the JPL-STAR were noteworthy examples in this line. Software - controlled recovery and diagnosis were soon advocated: anyhow, it is with the more recent structures (such as PLURIBUS, C.vmp and most notably SIFT) that software mechanisms have been extensively used to achieve all functions from testing to dynamic reconfiguration.

Although such "software-implement fault-tolerance" (as it was defined for SIFT) does not directly pertain to the subject of this paper, it is evident that delegating such critical functions as the ones for reliability control to software subsumes techniques for implementation of high reliable software, with particular reference to operating systems. Therefore, we consider it reasonable to give some space to analysis of the most interesting examples. It is almost inevitable to start with SIFT /1/, whose name (SIFT stands for Software Implemented Fault-Tolerance) is already indicative of the main design philosophy. In fact, in SIFT all tasks of error identification, confinement and masking are software-implemented, while the hardware structure is based upon standard, "off-the-shelf" modules (*) - processors, memory modules and I/O processors connected by means of a bus structure (fig. 1). High reliability is obtained here - as in many other cases - by module redundancy and voting; anyway, comparison and vote are performed by software instead than by special hardware. The redundant processors execute in parallel the same programs, suitably subdivided into "steps"; voting is performed both upon input information to each step (in order to mask previous errors) and upon output information.

Further fault-tolerant actions are performed by applicative programs (e.g. periodical test and diagnosis both of the single processors and of the whole system); even transient faults are analyzed by software only.

The above philosophy leads to a very complex and structured operating system; it also implies the availability of a very reliable operating system, and in fact it makes no provision for intrinsic fault-tolerance of software itself. Another - and quite different - interesting instance of operating system actions oriented to Fault-tolerance implementation is presented by C.vmp /2/. The basic structure is the conventional triple modular redundancy, with three processors and three memories connected through a voting device (see fig. 2). This device is software-controlled and capable of acting in one of three different modes:

a) vote: a vote is performed upon information exchanged between processors and memories (the three processors performing in parallel the same tasks) and the result is forwarded

b) broadcast: one of the processors (in the particular case, processor 2) effects the same information transfer with all three memories

c) independent operation: the system configures as three independent processor-memory pairs.

Any processor can, at run time, switch the voter from one mode to another one: thus, the system software performs a dynamic reconfiguration of the system structure so as to achieve high reliability (with the inherent cost) only with reference to critical functions. Here again - as in SIFT - very high reliability is required to the operating system, while no reference is made to intrinsic fault-tolerance of the software.

A third interesting instance is presented by PLURIBUS /3/; one of the motives for its interest is in the fact that PLURIBUS is not simply one system but the base of a complete line of modular structure. The design philosophy centers on a set of three "busses", each actually consisting of a dedicated bus and of a set of functional modules directly connected with it (fig. 3). The "processor bus" involves from one to two CPU's, a variable number of bus couplers, a local memory area; to the "memory bus" there are connected memory modules and, again, bus couplers; the "I/O bus" involves (besides bus couplers) a number of interfaces and special-purpose units. Moreover, each bus has a "bus arbiter" which decides upon access requests and allocates control of the bus. An architecture can be implemented by suitable interconnections through the bus couplers.

As in the previously described cases, the operating system performs a number of critical operations, both where configuration of the system and resource allocation is concerned and for typical fault-tolerance actions.

System diagnosis is performed by software, at various levels (single processor, system communication, etc.); protection mechanisms guarantee that no faulty process will be able to destroy the system operation. We would like to stress the reference to faulty processes rather than simply processors: in fact, the PLURIBUS approach allows to track faults both in hardware and software modules and aims at creating a complete system image, repeated upon each operating processor and accessible also outside the system.

The above presentation of SIFT, C.vmp and PLURIBUS is intentionally restricted to the basic elements of their architecture. For details, the interested reader is referred to the pertinent literature (see Section 7).

(*) In more conventional systems, at least part of error identification and masking are performed by dedicated hardware.

3. PROGRAM VALIDATION AND TESTING

Defining reliability of software might seem to be rather improper, since faults do not appear as a consequence of a physical failure but are due to design or coding errors not detected in the validation phase. Anyway, definition of software reliability can be given in very general terms by considering its "fitness for use". Thus:

- Reliability of a given software module at a given time T in the probability that at the same instant of time the module performs the expected operations.

A similar definition can - of course - be given also for a complete hardware-software system.

A measure of a software package reliability can be obtained as the ratio of correct to total runs in a sufficiently large interval of time. Since no aging phenomena are present, this probability is expected to be constant with time (or even growing with time in the case of proper maintenance). With hardware components it is possible to identify a first part of the life time in which the fault-rate decreases rapidly, while afterwards (and for a usually long time) it keeps constant. The same can be said to occur with software; if the initial debug phase is performed efficiently, after a time the number of "faults" becomes constant (and, if the system is complex, it is common experience that it does not become zero).

It can also be noticed that after a well-designed debug phase, software errors tend to present themselves as "transient rather than "solid" faults; in other words, they are such as to create an error state only for very particular input data and operation conditions. This peculiarity will justify the approaches to design of fault-tolerant software that will be introduced later on.

Improvement of software reliability has involved such different areas a programming methodology, software design and software validation (some approaches are summarized in /4/). While programming methodologies and software design techniques aim at decreasing the probability of errors in a program "a priori", software validation aims at identifying errors present in an existing program.

We do not here discuss general techniques in software design and programming methodologies: such subjects belong to the wide area of software engineering. Some particular techniques aiming at production of fault-tolerant software will be seen in a later section.

Among the various techniques for program validation, we recall three: "dynamic analysis", "interpretative analysis" and "static analysis".

Dynamic analysis actually constitutes a guide for testing strategies development: it consists in monitoring program behavior at run time and gathering information over a number of runs. The main drawback consists in data-dependence; as with other techniques for reliable software design, asking the designer to identify sets of test data or possible error areas easily leads to ignore all "non-foreseen" fault causes and decreases the meaningfulness of the techniques. Data dependence does not affect interpretative analysis methods such as "symbolic execution". With this technique, a particular path in the program is chosen and validated by assigning symbolic values to the input data and then "executing" the program path with reference to such symbolic values. When execution has been completed, analysis of the final symbolic expression gives clues concerning path correctness. It can be easily understood that for a program of even not great complexity, such final expressions easily become unmanageable.

Last, static analysis comprises such different techniques as "information gathering", "data flow analysis" and "program verification". While the first two lead to accumulate such information as variable usage tables, routine control graphs, illegal data usage instances etc., program verification (also called "proof of correctness") consists in "demonstrating the consistency between (1) a computer program and (2) specifications or assertions describing what the program is supposed to do"/5/.

Program verification should actually lead to implement zero-defect software; anyway, it has been stated that "because of the complex human interaction (required), it is usually only applied to small segments of a program" /6/. Moreover, we note that necessity of such complex human interactions introduces in fact probability of a number of new faults.

As opposed to the above criteria, testing aims not at proving correctness (i.e. absence of errors) but at identifying a number of errors (hopefully as near as possible to completeness). Testing originates (whenever an error is found) a phase of debugging, i.e. error localisation and removal.

It can be underlined that testing does not guarantee complete absence of errors; in Dijkstra's words, testing shows "presence, not absence of errors". While a completely thorough testing is not possible for a complex program, some analysis techniques have been proposed for identifying program segments most likely to create problems and which deserve more accurate testing. Anyway, it has to be accepted that a program will not be completely free of errors even after extensive testing, both because not all possible input data sets can be applied and because debugging actions can in turn introduce new errors (the so-called "ripple effect") (for an extensive review, see /7/).

A qualitative evaluation of error identification vs. time is given in fig. 4 (full line); the effect of particular testing strategies can modify the shape as shown by the dotted line.

Non-trivial software products are usually decomposed in a hierarchical multi-level structure, to which a corresponding organisation of testing can be superimposed. Thus, with any level of the structure there is associated a level of diagnosis characterized both by its object and by the type of errors it is capable of detecting. A typical organization is the following:

- "unit test": "unit" corresponds to a module, which is tested so as to verify its internal correctness, independently of interactions with external world;
- integration test: it operates upon a set of interconnected modules, thus allowing to check the corresponding interfaces;
- subsystem or system test: it operates upon a set of interconnected modules implementing a complete function - possibly the whole system itself. Rather than correctness, consistency with design specifications is checked;
- acceptance test, leading to product certifying.

4. SOFTWARE FAULT-TOLERANCE

Contrasting with the previous section - involving, in fact, techniques for reducing the number of faults - we consider now techniques aiming at design of fault-tolerant software, i.e. software capable of:

a. identifying presence of faults
b. confining errors due to faults
c. recovering from errors

An a-priori assumption to be made is that the software package will be tested and debugged in a manner capable of guaranteeing correct operation in a (very) high percentage of runs: no major "bugs" will be present, invalidating the results of most operations. On the other hand, it is accepted that for some particular (but not identified) data sets and operation conditions the system will not perform correctly, and it is asked that such failures will be detected and that their effects will not spread in catastrophic way. Quite often, a designer aiming at fault-tolerance makes some assumptions, such as:

- correctness of the basic algorithm(s)
- knowledge of failure modes (depending on explicit or implicit fault hypoteses)
- knowledge of all possible interactions between system and external world.

These assumptions are very limitative and do not guarantee fully acceptable results: in particular where software is concerned, the designer is quite liable to be "biased" in relation to a class of foreseen faults only. It is rather necessary to make a fault tolerant design even for non-foreseen faults.

To this purpose, we need to check the validity of control and data flow, restricting in the design phase the opportunities for error and providing confinement and recovery mechanisms *without leaning upon specific fault hypotheses*. We note that with software, error confinement and recovery will lead to temporary (i.e. restricted to the present run only) abandonment of a particular software module in favor of another one; as already said, software faults are considered as transient - not solid - ones, and "faulty" modules are not removed for maintenance. In other words, fault-tolerance approaches to software do not foresee a "maintenance" or "debug" phase.

Design techniques suggested aim at protection of data and guidance of control flow (in particular, of process interactions) so that possibilities of data destruction or of unwanted interactions as a consequence of faults will be restricted. (A very extensive analysis of problems and solutions is presented in /8/). Typical of such techniques may be considered the two concepts of "atomic actions" and of "capabilities". An atomic action is defined as a seemingly instantaneous action performed by a process or a set of processes and excluding all interaction between the processes performing it and the rest of the system and/or the external world; interactions are seen as information exchanges. Thus, if a set of operations has to be performed without interruption (and possible disruption of the control flow) and without destruction of own information, by defining it as an atomic action we guarantee that any outside attempt at interaction will be considered as erroneous and rejected. A simple graphic representation of three interacting processes, of atomic actions and of allowed interactions is given in fig. 5.

Use of atomic actions for error confinement and recovery will be discussed later on: anyway, we already notice that it leads to:

a) restricting process interaction to well-defined areas
b) protecting information and identifying the "state" of a single process with reference to the last atomic action successfully concluded.

A further step in restriction of occurrences of allowed actions is presented by capabilities. The basic idea can be summarized by stating that "any action not explicitly authorized is forbidden" /9/.

A capability is a protected piece of data, referred to an "object" (i.e. a resource or another entity of the system). Capabilities can be created only by the system and cannot be modified by any valid operation : they enable and control the parallel activation of different atomic actions by means of "locks". Each capability contains protection information (i.e. access rights or attributes) specifying how the associated object can be used; any attempt of access to an object by a process not explicitly endowed with access rights immediately fails. Thus, a capability-based system facilitates resource control and process isolation; moreover, process interaction can take place only following pre-determined paths, and all interactions must be explicitly described.

Approaches as the ones described allow to perform checks on the results of various actions at pre-arranged points. Besides the "built-in" checks we have recalled, tests on process results can be performed either by comparison with suitably inserted "masks" or by voting upon the results of alternative, equivalent processes. These two different philosophies

lead to recovery and reconfiguration strategies that recall, respectively, the "dynamic" and "static" redundancy in hardware fault-tolerance.

We consider first th approach that most closely recalls classical fault-tolerance techniques. This is the so-called "N-version programming method" /10/. The basic idea is to achieve high reliability through static redundancy: obviously, exact replication of a program (as if it were a hardware module) would be meaningless, since in software a "fault" actually is an error - which would also be replicated. Rather, the suggestion is to implement several idependently developed alternative routines: $N \geq 2$ functionally equivalent programs are independently generated (the ideal would be to use even different languages and/or translators) and all versions are executed. All versions must provide comparison and status information, and a voting algorithm must then be designed in order to compare results provided by the different versions. The results are voted, and the ones accepted by the majority are forwarded; anyway, routines producing discarded results are kept active for subsequent runs.

Suitable synchronization mechanisms are used to synchronise the versions with outside activities and to obtain a meaningful vote even when the voting apparatus has to compare non-identical results (non-identical due to their creation, not to possible faults).

N-version programming (for $N \geq 3$) has an error-masking effect: therefore, no problems concerning error confinement or recovery arise (provided, of course, hypotheses on maximum number of errors present are valid). This simplicity of approach is counterbalanced by the very high redundancy required in terms of program development, of storage occupation and of execution time: costs rise steeply if the criterion is applied to a large program, and execution time may render the approach unacceptable for real-time implementations. Moreover, the approach is valid only if the errors are at coding or at logical level, (in these cases independent versions will not exhibit the same faults) not if they are at specification or system-design level. A reasonable approach may be to adopt N-version programming for safety-critical (but not time-critical) routines.

In contrast, the "recovery block" approach recalls the "spare" technique with dynamic redundancy used in hardware /8/. A program segment is executed and its results are validated by some suitably defined mechanism; if they are acceptable, the program is continued through its logical sequencing, otherwise a (possible) backup routine is executed upon the same data (and further spares can be provided, if so wished). This requires first of all that the program be organized as a set of modules with well-defined interfaces, allowing the interface information to be checked (much as it happens with hardware modules where, e.g. , error detecting codes may be used).

If no "spare" gives acceptable results, we must guarantee first of all that no result produced by a faulty module will spread through the system. It is easy to imagine how a suitable structuring of the system into atomic activities and use of a capability-based philosophy will help in obtaining these error-confinement features. For instance, a faulty process attempting to write into a storage area to which it is not entitled will be denied access, if capabilities are used.

Besides avoiding error propagation, it is necessary also to find out the most recent correct state of the whole system, so that the system will be able to "back up" and to continue with operation starting from that known condition. While a single-process organization will create no problems - under an atomic activity, capability-based philosophy - for this "state identification", a multi-processing (or, even worse, multi-processor) system will require careful design for this purpose.

Consider for instance the three processes in fig. 6; for each process, a number of "recovery points" are defined, corresponding to a known state of the process. If a single process fails, it is "backed up" to the nearest previous recovery point and all information processed since is considered invalid and potentially dangerous. Assume first that a failure is detected in process P_1 at some point between R_{12} and R_{13}; information processed (and possibly exchanged) between the two points is considered invalid. P_1 is backed up to R_{12}, P_2 is backed up to R_{22}, and the system restarts from that state (note that P_3 has not been affected).

Assume now that P_2 fails between R_{22} and R_{23}. For the same reasons discussed above, the three processes are backed up to - respectively - R_{11},R_{22},R_{32}. Anyway, at this point, also information processed between R_{11} and R_{12} is considered to be invalid, and P_2 is backed up to R_{21}. This, in turn, requires that P_3be backed up to R_{31}, and - finally - that the whole system be brought to the initial point. This effect ("domino effect") is clearly undesirable but it can be avoided only by very careful design; actually, designing a multiprocessing or - which is even more critical - a distributed system so as to guarantee implementation of a recovery block-like scheme without domino effect is an open research subject.

Backward error recovery schemes such as the recovery block approach or other "rollback" criteria always resort to discarding a number of results and to repeating a number of actions (possibly by applying different algorithms). On the other hand, it may be noticed that such techniques refer to error identification , and can be made reasonably independent from too critical assumptions on fault nature or cause. Considering error recovery mechanisms at a given abstraction level, the above characteristics may allow a relative freedom from knowledge of implementation details at lower abstraction levels.

The same degree of freedom cannot be guaranteed by forward error recovery techniques; here , the aim is to preserve a (reduced) meaningfulness of the state just found to be erroneous, and to use available information gathered from this state in order to proceed

in system operation. It can be immediately inferred that recovery mechanisms have to be tightly integrated into the system design, and that they must be based upon a number of fault hypotheses. (It might be contended that even error-correcting codes are a form of forward error recovery). Forward error recovery schemes seem to be an interesting alternative for tolerance of failures originating from run-time problems (e.g. environment-machine interactions, marginal process synchronization, etc.) rather than from basic software design errors.

A technique belonging to this recovery philosophy is implemented by "exception handling" facilities provided in programming languages. In this case, the programmer will insert into the algorithm sections capable of recognizing "exceptions", (i.e. well defined unwanted conditions) and of coping with them. For instance, in a multi-processor structure using message-passing as a communications means, appearance of erroneous message configuration (or even of particular signals) will be handled as an exception and cause a suitable sequence of actions (e.g. self-test of the interacting processors involved, use of alternative communications links, etc.).

While it is clear that exception handling does not provide a complete solution, it can be well used in conjunction with other fault-tolerance techniques. It is interesting to note that both ADA and CHILL definitions provide space for exception handling mechanisms.

5. SOFTWARE RELIABILITY

The traditional aim of software reliability studies is prediction of a software package's behavior "on the field". This is usually accomplished by building a proper mathematical model of the given package; of course, the parameters present in the model should be determinable on the base of the general properties of the package, such as size, complexity etc. In this way, it is possible to evaluate reliability even during the design phase. During the testing phase, a better knowledge of the package's quality is achieved and then a better estimate of the set of parameters entering the model is possible.

Many contributions can be found in the literature on this problem; in this paper we restrict ourselves to one of them, and refer the interested reader to /11/ for details.

The basic characteristic of this model is the fact that distinction is made between calendar time and cumulative execution time ; all evaluations are performed with reference to cumulative execution time, and only at the end a conversion is made to calendar time for practical purposes.

The basic assumptions are:

a) The fault detection rate is proportional to the number of faults remaining .

b) The fault correction rate is proportional to the fault detection rate.

Among the main results, we pick up the following two:

- The number of failures experienced is an exponential function of the cumulative execution time. This relationship is illustrated in fig. 7; M_o is the total number of failures possible during the maintained life of the software. The time constant is $\frac{M_o T_o}{C}$ where T_o is the MTTF at the start of test and C is the "testing compression factor". "Maintained life" is the period extending from the start of test to discontinuance of program failure correction.
- The present mean time to failure is also an experimental function of the cumulative execution time. It is illustrated in fig. 8, and the time constant is the same as in fig. 7.

As we mentioned, there is also the possibility for execution time prediction to be converted into dates. This is based on a model of the debugging process: the reader is referred to /12/ for the formula that relates execution time and calendar time.

6. CONCLUDING REMARKS

An ever increasing number of control problems are today solved by using programmable digital systems; starting with the most simple microprocessor-based structures which operated upon a single process, implementations are evolving towards more sophisticated multiprocessing systems and to distributed, multiprocessor structures. Noteworthy examples can be found in the aerospace area.

The above trend from one hand underlines the importance of designing reliable hardware-software systems; on the other hand, it stresses the inadequacy of conventional reliability ideas restricted to non-dynamically reconfigurable systems.

In the present paper, we tried to outline the subject area and we attempted to show how fault-tolerance techniques are entering the picture and can complement the more classical fault-avoidance approach.

In any case, both fault-tolerant design and reliability evaluation have to cope with the many problems arising from an enironment where a number of coöperating processes/ processors are present. This is a research area today; it is possible to foresee that in the next years suitable design techniques will be available.

7. REFERENCES

/1/ Wensley, Lamport et al. "SIFT: design and analysis of a fault-tolerant computer for aircraft control" Proc. of the IEEE, vol. 66, n. 10, pp.1240-1254, oct.1978

/2. D.Siewiorek et al. "A case study of C.mmp, C.m*, and C.vmp: Part I, experiences with fault tolerance in multiprocessor systems' ibidem, pp. 1178-1184

/3/ Katsuki, Elsan et al. "PLURIBUS: an operational fault-tolerant multiprocessor" ibidem, pp.1146-1159

/4/ R.Negrini, M.G.Sami, R.Stefanelli "Verification of complex programs and microprograms" in Designing and Programming Modern Computer Systems, vol.I, S.P.Kartashev and S. Kartashev eds, Prentice-Hall, 1981

/5/ R.L.London "A view of program verification " Int'l Conf. on Reliable Software, Los Angeles, 1975

/6/ L.A. Clarke "Testing: achievements and frustrations" Computer Software and Applications Conference (COMPSAC) 1978

/7/ A. Celentano, C. Ghezzi, F. Signori "Metodologie e tecniche di convalida del software" Int.Rep. IEE-LC 79-19, Politecnico di Milano

/8/ R. Randell,P.A. Lee, P.C. Treleaven "Reliability issues in computing system design" ACM Computing Surveys, vol.70, N.2, 1978, pp. 122-165

/9/ W.C.Carter "Fault detection and recovery algorithms for fault tolerant systems" EURO IFIP Conference , London, 1979

/10/ L. Chen, A. Avizienis "N-version programming: a fault-tolerance approach to reliability of software operation" FTCS-8, Toulouse, 1978

/11/ J.D. Musa "The measurement and management of software reliability" Proc. of the IEEE vol. 68, N.9, sept.1980, pp. 1131-1143

/12/ J.D.Musa "A theory of software reliability and its applications" IEEE trans. Software Engineering, vol. SE-1, pp. 312-327, Sept.1975

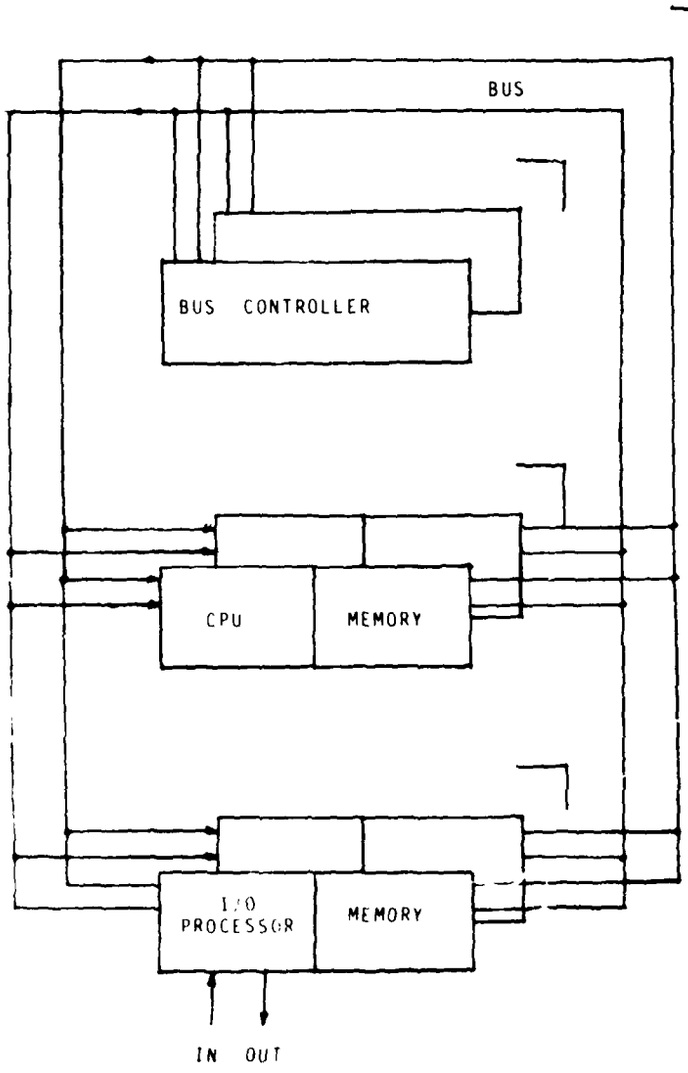


Figure 1. Basic SIFT structure

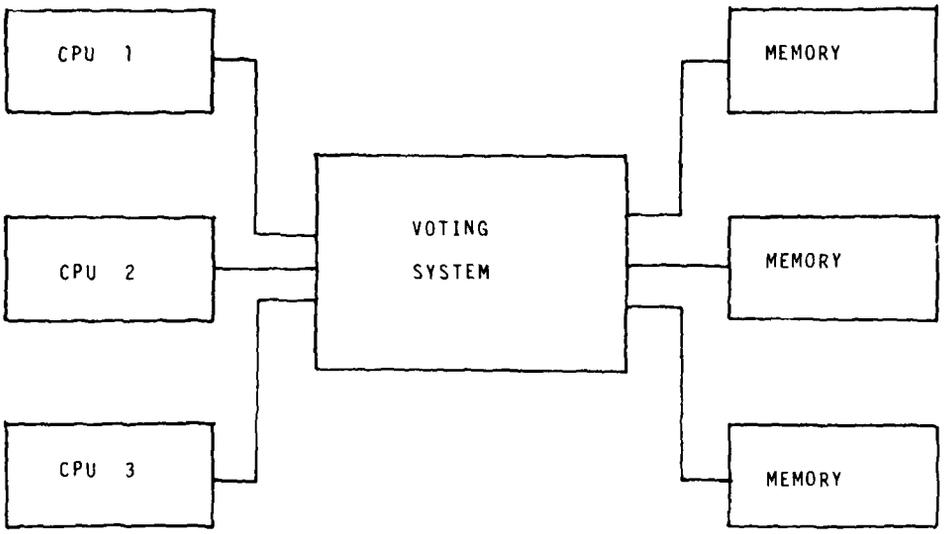


Figure 2. c.vmp structure

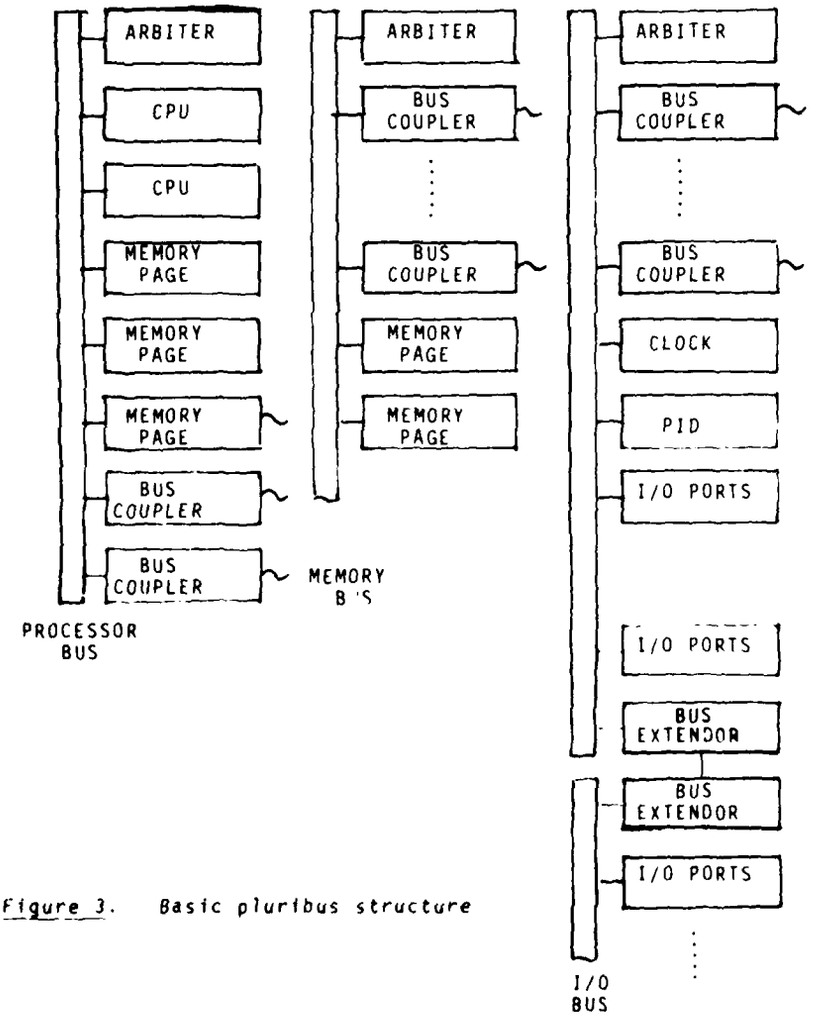


Figure 3. Basic pluribus structure

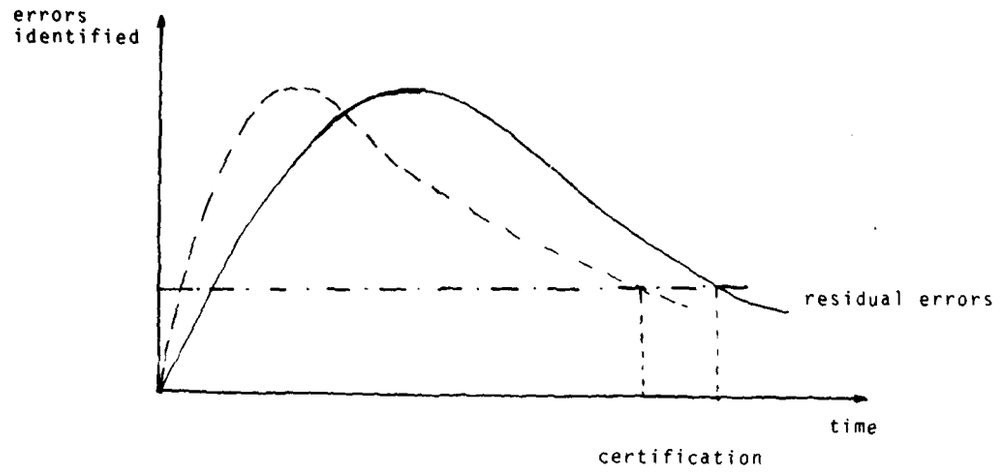


Figure 4. Qualitative evaluation of error identification versus time

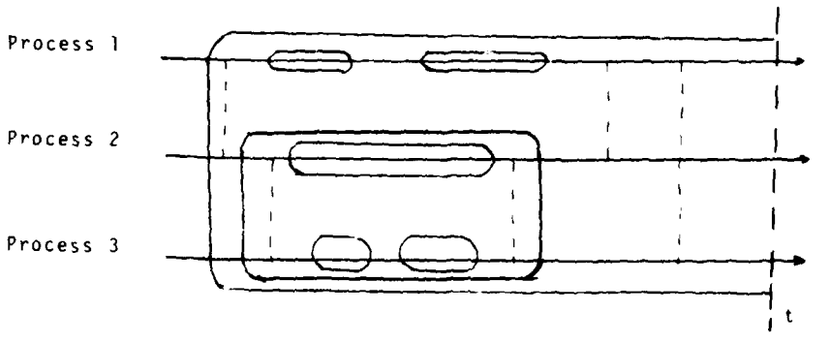


Figure 5. Full lines circle atomic actions; dotted lines represent process interactions

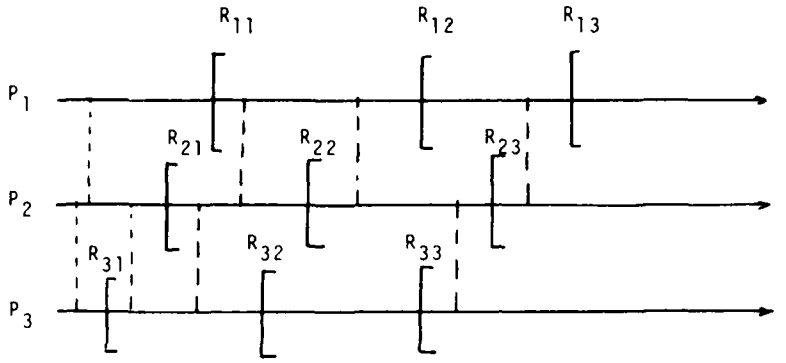


Figure 6. Recovery points leading to possible domino effect

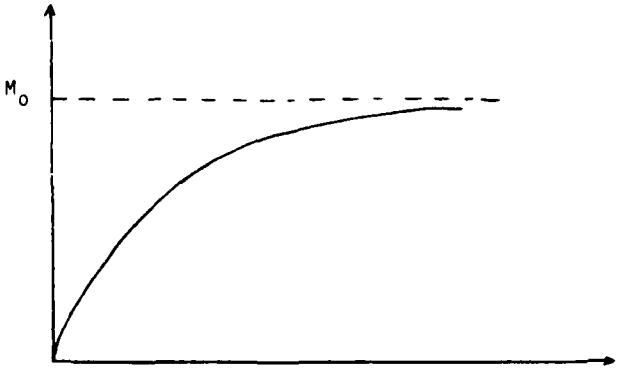


Figure 7. Failures experienced versus execution time

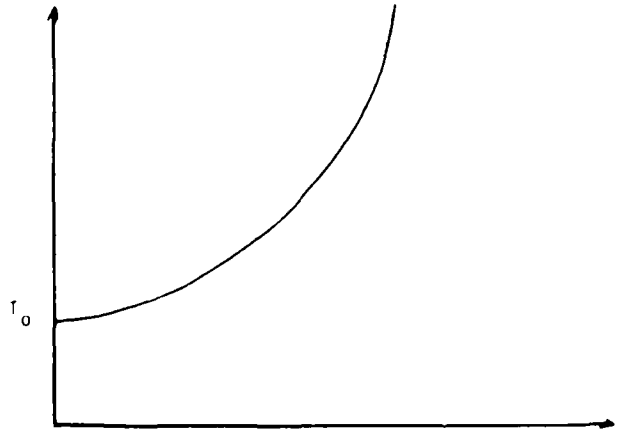


Figure 8. Present MTTF versus execution time

DESIGN OF SECURE AND MODULAR MICROCOMPUTERS FOR ENGINE CONTROL

BY

C. BEOUNES and J.C. LAPRIE
CNRS - LABORATOIRE D'AUTOMATIQUE ET D'ANALYSE DES SYSTEMES (TOULOUSE)

J.M. COLLIN
ELECMA - DIVISION ELECTRONIQUE DE LA SNECMA (SURESNES)

SUMMARY

The large scale integration circuits which are now available provide designers with new possibilities for the rational definition of digital automated devices integrated in the power plant control system.

In designing these automated devices, the following factors must be taken into consideration :

- functional specifications,
- operational specifications, especially those related to reliability and mission safety,
- technological constraints imposed by adverse environments.

This paper summarizes a systematic design methodology and describes some special purpose microcomputers for turbo-jet engine control.

1. - INTRODUCTION

The digitalization of an ever-increasing number of avionic systems can be explained by the improvements achieved at the functional, operational safety, and economic levels (1). These improvements are largely due to recent developments in large scale integrated circuit technology particularly in the field of microprocessors, microcomputers, memories and arithmetical coprocessors.

ON THE FUNCTIONAL LEVEL, digitalization brings about a high computational accuracy and repeatability and a large flexibility of use. Moreover, it is often the only way to mechanize sophisticated control laws (non-linear functions, multivariable feedback...).

ON THE OPERATIONAL LEVEL, digital systems are globally better than their analog or hydromechanical counterparts (2) due to their capabilities to perform self (or inter) monitoring and to their potential of being fault tolerant.
Although maintenance is more frequent, it is greatly facilitated by use of automatic fault localization.

ON THE ECONOMIC LEVEL, analog circuits are competitive only for simple applications because their cost/complexity curve rises much faster than for digital circuits (figure 1). This evolution will probably be emphasized since microprocessors and microcomputers are becoming more and more powerful (figure 2 shows rate of change as per Moore's well-known graph). The operating cost of digital systems tends to be lower because they are easier to maintain (they provide fault detection and localization facilities) and because they are made of interchangeable standardized subsystems.

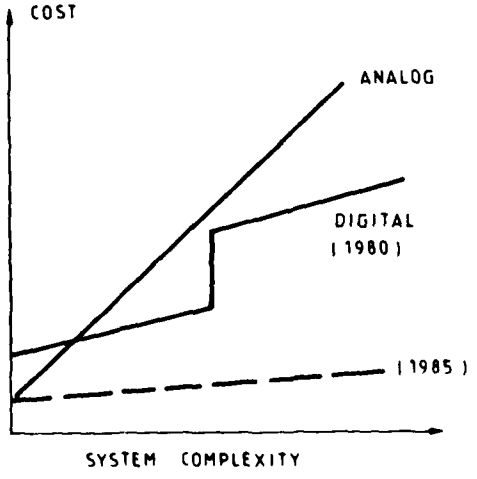


FIG 1

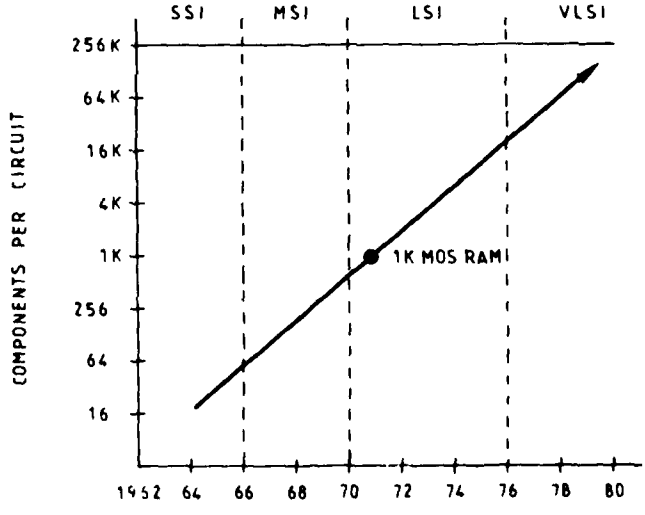


FIG 2

It is necessary however to underline that this evolution towards digital systems is not possible without solving numerous problems related to the extreme severity of the avionic environment (3).

Also, designers faced with the impossibility of dealing with the ever-increasing number of often contradictory constraints and criteria have searched for a rigorous design methodology using design tools (mainly for description and performance evaluation).

2. ENGINE CONTROL SYSTEMS

Modern aircraft operational requirements involve the availability of propulsion systems having increased performances over wider operating envelopes. The sophistication of turbo-jet engines (e. g. multi-shaft engines equipped with an afterburner and variable geometry components) is such that conventional hydromechanical control units can not handle alone the functions required by the control system. Therefore the system designer is forced to associate electronic and hydromechanical equipments. The authority conferred on the electronics may be very different according to the type of engine, aircraft and mission.

We may identify two main types according to the authority of the electronics (4) :

- A LIMITED AUTHORITY ELECTRONIC CONTROLLER ("TRIM"). Improves the control by modulating the commands around an operating point determined by the hydromechanical regulator (figure 3),

- A FULL AUTHORITY ELECTRONIC CONTROLLER with a changeover in case of failure to a further electronic unit (figure 4) or to a hydromechanical back-up (figure 5).

However on the functional level the role of the electronic controller is very similar for both organizations. A trim or a full authority controller must have the following resources :

- AN ACQUISITION UNIT, which consists of signal conditioners, multiplexers, analog to digital converters...

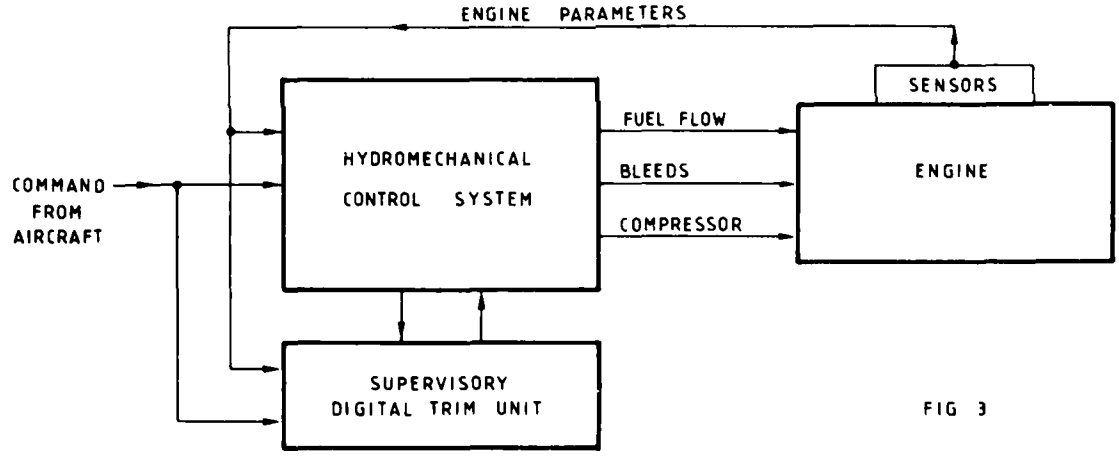


FIG 3

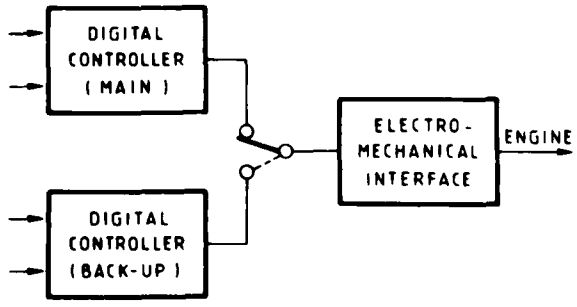


FIG 4

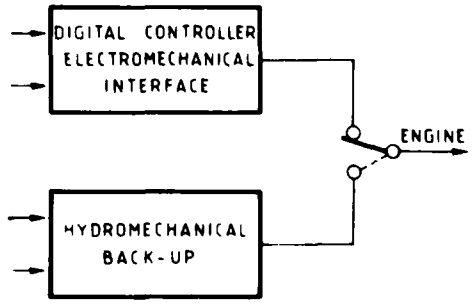


FIG 5

- A CONTROL BLOCK AND A COMPUTATION BLOCK. The control block (which may be hardware or software) is responsible for organizing the system, the complexity of the computation block is defined by the level of sophistication of the control system requirements.

- A MEMORY consists of two areas: one read-only memory for program storage, one random access memory for storage of temporary results.

- AN ACTUATION BLOCK able to drive various actuators (motors, valves).

- A POWER SUPPLY MODULE connected to aircraft power bus or/and to an engine mounted alternator.

The choice between a trim and a full authority controller depends upon the specification of each application. In fact the two main types we have identified previously are the extreme range of options. In this paper after some comments on design methodology we will describe different units :

- simple and sophisticated trim units,
- high performance multiprocessor with possible graceful degradation,
- full authority, fault tolerant microcomputer.

3. - DESIGN METHODOLOGY

3.1. - DESIGN PROCEDURE

Except for very simple systems, it does not seem possible to use a design approach that simultaneously takes into account all of the constraints and performance specifications. Consequently, and in harmony with the normal human thought process, the most efficient and realistic solution is a top-down step-by-step iterative procedure. At each step in the design, an evaluation of the functional and operational performances is carried out. The principle of this design method is given by the flow chart of figure 6.

In the first phase, an analysis of the specifications leads to the distinction of two categories of constraints according to whether or not they leave any degree of freedom in the choice of solutions.

The resulting choices are of different types according to the category of constraints which induce them :

- The necessary choices resulting from constraints having no degree of freedom, which could lead to redefinition of the specifications if the latter are not validated by the performance evaluation,

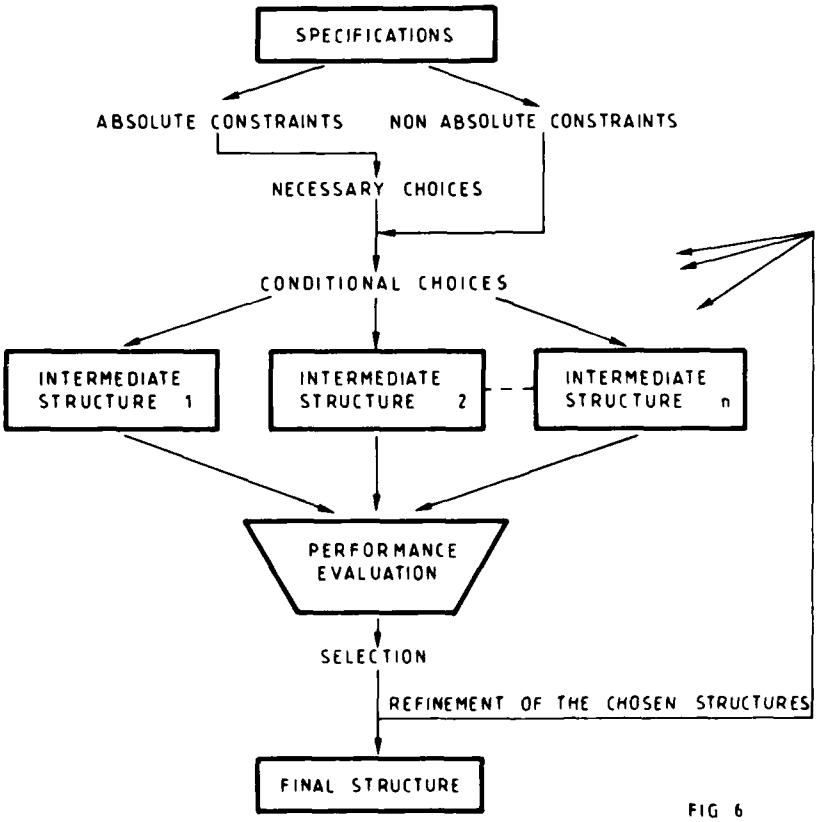


FIG 6

- The conditional choices resulting from constraints having some degree of freedom, which are directly related to the results of the performance evaluation.

The number of iterations in the stepwise refinement of the structure is obviously limited and specific to the application.

3.2. - STEPS OF THE DESIGN

Four of the six abstraction levels proposed by M. SU (5) are at least to be taken into account :

- The algorithmic level which concerns the determination of the tasks that must be carried out. This level is dealt with during the definition of the specifications,
- The architectural level which concerns the definition of the different entities that must be used (processors, memories, etc...),
- The hardware level which concerns the specification of the interconnections between the different entities and the test and redundancy policies,
- The realization level which concerns the mounting of the different integrated circuits chosen for the design.

3.3. - PERFORMANCE EVALUATION

The necessary choices (§ 3.1.) lead to an initial definition of the structure that corresponds to the architectural level which is then refined by modelling the structure's behaviour. Therefore two types of model must be realized :

- A functional model which enables an evaluation of the structure's performance (speed, computation accuracy, hardware and software capabilities...),
- An operational model in order to evaluate the operational specification (availability, reliability, safety...).

Setting up the models can be done by two different approaches :

- Descriptive where the different parts and their interconnections are listed,
- Interpretative where the system behaviour is analysed.

The descriptive model is static. By injecting predeterminated events it becomes dynamic. The readings supplied directly or after specific treatments describe the system behaviour characteristics (e.g by MONTE CARLO METHODS).

For the interpretative model, mathematical tools enable one to determine analytically system characteristics (e.g by MARKOV PROCESSES).

Model processing gives two types of results :

- Qualitative (general properties of the system in terms of well-defined behaviour, consistency with requirements...),
- Quantitative (computation of timings, operational security, economic...).

These results are used to select comparatively the final structure from among the intermediate ones.

Useful design-aid tools for functional and operational evaluation are listed figure 7.

NOTE : Even though numerous tools can give efficient performance forecasting, their use may be detrimental to design consistency as there is an increase in criticity of the method (it is difficult to fully master all of the different tools) and an increase in the number of errors involved in the modelling steps (model inaccuracy).

4. - DESCRIPTION OF SOME DIGITAL CONTROLLERS

4.1. - SIMPLE AND SOPHISTICATED TRIM UNITS

4.1.1. - A CMOS PROCESSING BOARD

This processing board it built around an 8 bit CMOS microprocessor (COSMAC).

It consists of (figure 8):

- A microprocessor clocked at 4 MHz,
- A random access memory (1K bytes),
- A reprom memory (4K bytes),

MODEL / RESULTS	DESCRIPTIVE		INTERPRETATIVE
QUALITATIVE	PETRI NETWORKS (6) (7) APL	SIMULATION	PETRI NETWORKS
QUANTITATIVE	QUEUES		STATE GRAPHS TASK POTENTIAL GRAPHS (8) (9)

Table 1 : FUNCTIONAL

QUALITATIVE	PETRI NETS	SIMULATION	FAULT TREES
QUANTITATIVE	RELIABILITY GRAPHS		FAULT TREES (10) MARKOV PROCESSES (11) STATE GRAPHS (12)

Table 2 : OPERATIONAL

FIG.7

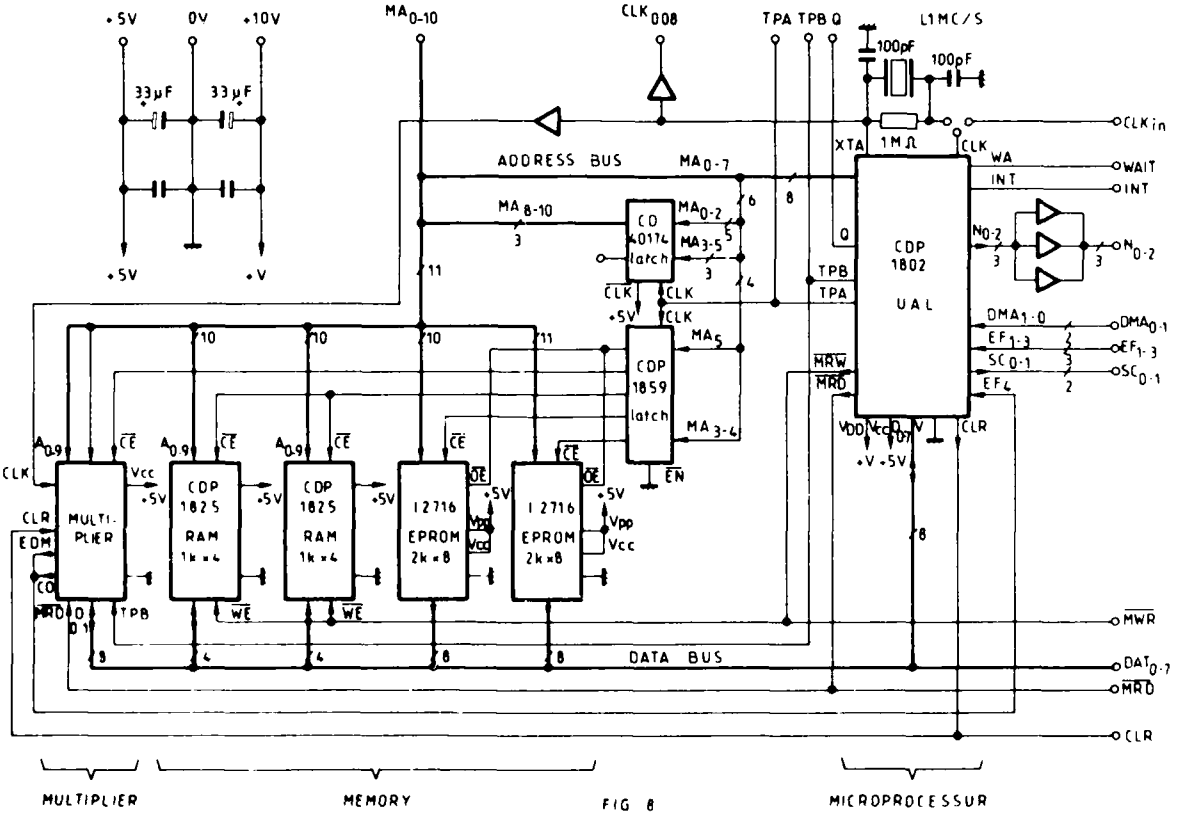


FIG 8

- A 16 by 16 bit multiplier (a microelectronic circuit built from serial-parallel multiplier chips).

The board consumption is less than 500 mW without the multiplier and less than 1.5 W with it.

The arithmetic calculation capabilities are average even with the enhancement due to the multiplier (a double-precision, memory to memory, operation including tests takes about 200 microseconds).

The processing board (photograph 1) is able to work up to +125°C.

Its applications are :

- Random logic replacement,
- Small controllers,
- Engine parameters monitoring,
- Multi-microcomputers systems.

NOTE : More complex controllers with large input/output and processing facilities may be implemented in few chips (for instance 8-bit MOS processors associated with high performance arithmetic processors) but their higher speed has to be paid by a higher consumption (5 to 10 W).

4.1.2. - A HIGH PERFORMANCE TRIM UNIT SYREN

SYREN (SYsteme de REgulation Numérique) is an engine mounted controller (photograph 2). Its main features are :

- A high throughput and a high calculation power,
- A very easy-to-use instruction set,
- A good level of safety.

4.1.2.1. - HARDWARE STRUCTURE

SYREN has the typical structure of a monoprocessor minicomputer designed around a bit-slice microprogrammable microprocessor.

The system is made of the following functional parts :

- Digital processor,
- Input/output interface,
- Monitoring circuits,
- Power supply unit.

THE DIGITAL PROCESSOR (figure 9) CONSISTS OF :

- A 16-bit arithmetic and logic unit with a clock cycle of 250 ns. Three of the sixteen internal registers are used as program counter, stack pointer, status register,

- A micro-sequencer. The micro-instruction size is 56 bits (20 for ALU control, 4 for peripheral interfaces control). The microprogram memory contains 1024 words, 256 of which are available for instruction set expansion,

- A ram array of 1 K words,

- A program memory with 10 K words of programmable read only memory (possible extension up to 18 K words),

- An asynchronous coupler (RS 232 C type),

- A clock oscillator and some additional circuitry for real-time generation,

- A four level interrupt controller.

The digital processor consumption is about 25 W, therefore the cards on which it is wired have thermal drains (photograph 3).

THE INPUT/OUTPUT INTERFACE PERFORMS SEVERAL FUNCTIONS :

- Isolation of inputs and outputs subject to high common mode voltage (isolation is obtained by use of transformers, optoelectronic devices, floating preamplifiers),

- Processing of signals delivered by sensors (SYREN manages 48 analog inputs, 2 frequency inputs, 32 discrete inputs),

- Power amplification for actuator driving (5 DC motors, 4 electro-valves),

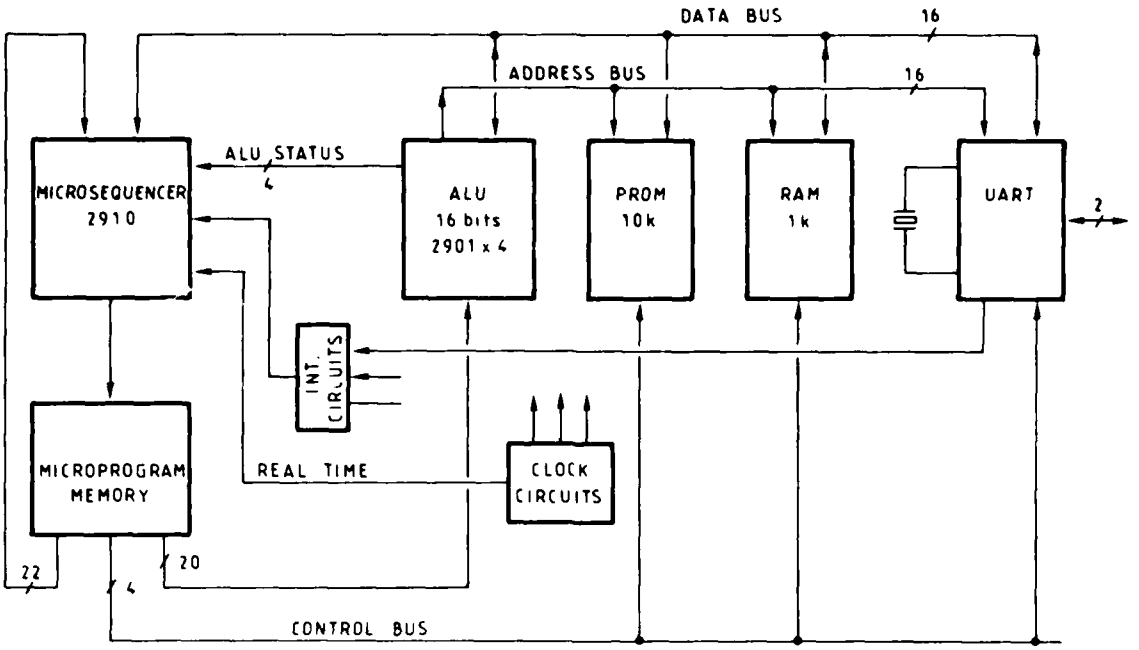


FIG 9

- Various outputs (8 analog - 8 discrete for monitoring or maintenance purposes).

THE MONITORING CIRCUITS

Some circuits have been added for testing *I/O interfaces and monitoring* the correct execution of the program (see safety section).

THE POWER SUPPLY UNIT

SYREN is DC powered via a non-interruptable line. Its power supply consists of a redundant set of switching regulators. For an input range from 15 V to 80 V its efficiency is better than 80 %. All the circuits are protected against overvoltage, short circuits, and radio-interferences.

4.1.2.2. - SOFTWARE STRUCTURE

THE SOFTWARE ORGANIZATION of SYREN is based upon the notion of confined modules (figure 10):

. Electronic hardware resources management module named GA (Groupe Automate),
. Engine control module GR (Groupe Régulation),
. Development module GD (Groupe Développement).

GA and GR are run on a real-time basis. For safety reasons, GD cannot be processed by the real-time executive (an operator can access through GD to all the modules by means of a console).

Data are transfered between GA and GR through a common data segment.

The internal organization of each module is similar to those of the machine (figure 11). For each module a table of tasks is initiated by hardware (at power-on) or by software. This table is also updated in relation with the engine control mode.

The engine control mode is memorized from the previous state in order to facilitate the determination of the present state and the corresponding processing ; Petri Nets are very useful to determine the organization of the program.
A study of the marking and of the transition to be fired (figure 12) facilitates the writing of the related flow chart. (Figure 13 gives an example of reheat selection. Each task may be defined by using the same procedure).

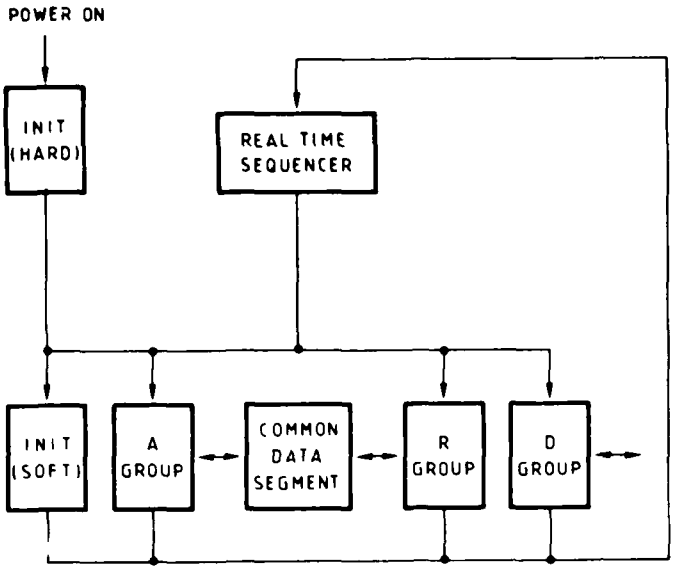


FIG 10

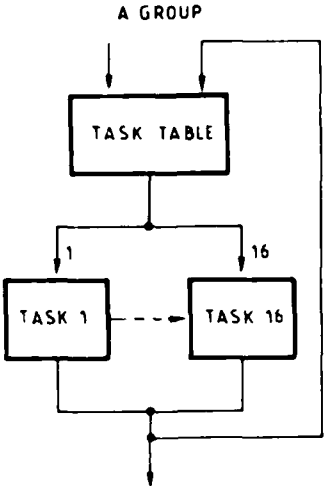
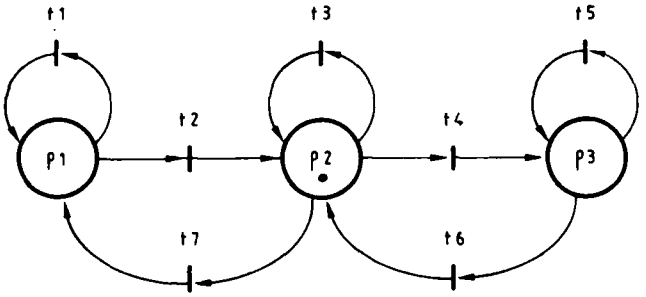
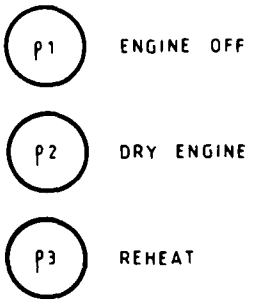


FIG 11





t1 = SENSOR MONITORING

t2 = ENGINE START-UP

t3 = DRY ENGINE CONTROL

t4 = REHEAT START-UP

t5 = DRY + REHEATED ENGINE CONTROL

t6 = REHEAT TURN-OFF

t7 = ENGINE TURN-OFF

FIG 12

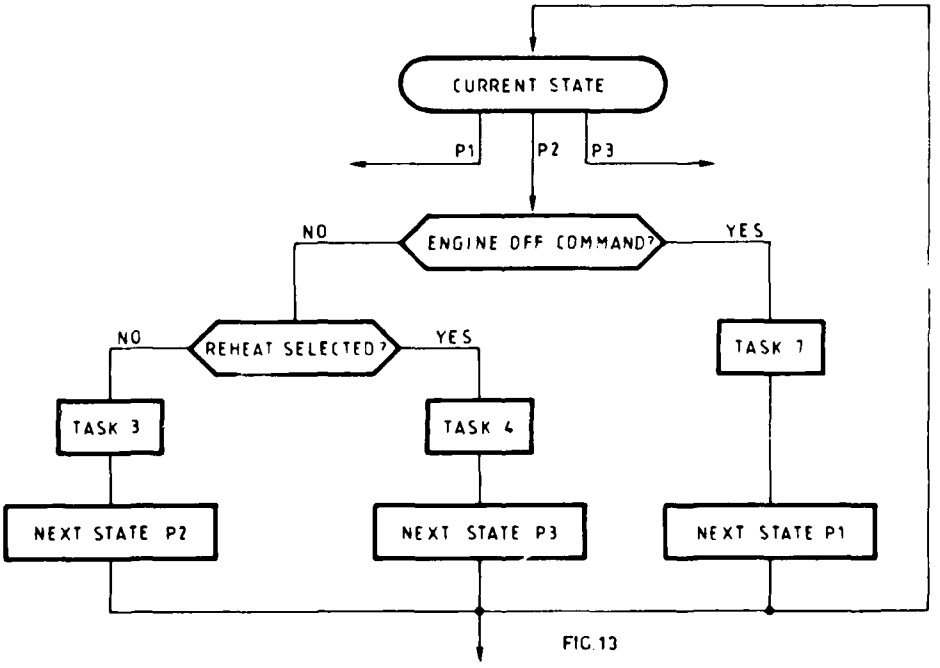
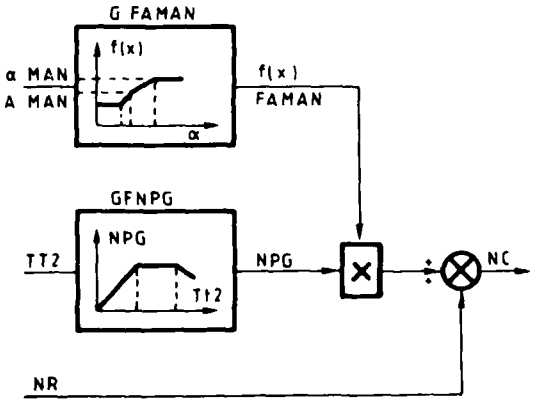


FIG 13

THE INSTRUCTION SET

Using a microprogrammable processor permits the definition of an instruction set best suited to the application.

Programs are easier and shorter to write and execution time is fast due to long microprogrammed sequences which avoid a lot of instruction fetches. In addition to the usual logical and arithmetic instruction, the instruction set of SYREN contains dedicated instructions (function generator, second order filter, hysteresis...).
Also, simple addressing mode and multi-adress instructions induce easy learning and use (figure 14).



INSTR	INPUT	DATA	OUTPUT
GEF	AMAN ,	GFAMAN ,	FAMAN
GEF	TT2 ,	GFNPG ,	NPG
MUL	NPG ,	FAMAN ,	NC
ADD	NC ;	NR ;	NC

G FAMAN
G FNPG
} DATA TABLE

FIG 14

4.1.2.3. - SAFETY

Although SYREN is a trim unit it has several facilities for improving its safety :

- Transducers and conditioner likelihood tests (value and gradient),
- Acquisition of supplementary signals in order to detect failure of actuators,
- Arithmetic and logic unit,
- Test by using precalculated functions,
- Analog to digital converter and digital to analog output converters are tested by acquiring known values and by looping outputs with inputs,
- Program memory checksum,
- RAM memory tests by writing/reading and comparison,
- Program execution : A watch-dog (a hardware monostable) is triggered by a pulse generated by instructions incorporated within the program. A processor failure or a program looping due to hardware or software faults are detected,
- Local redundancy,
- Safe self-position of outputs...

4.1.2.4. - REALIZATION (see photograph 2)

SYREN is engine mounted. It consists of 12 printed card with thermal drains and 2 power blocks, all of them plugged into a double box equiped with shock absorbers.

The main physical characteristics are:

- Size : 250 x 350 x 110 mm
- Weight : 12 kg
- Consumption : 100 watts max.
- Thermal insulation
- Cooling by forced air

4.1.2.5. - PERFORMANCE

The current engine control tasks processed by SYREN consists of :

- Processing and monitoring of various sensors (2 speeds, 3 temperatures, 1 pressure, 6 positions, 8 discrete...) and actuators (5 DC motors, 4 valves, 2 warning devices...),

- Assuming start-up sequences,

- Controlling within 5 ms four non-linear loops including the sophisticated generation of their set points.

Today SYREN is 70000 times more powerful than a very good pocket calculator and 5 times more than the 16-bit new generation monolithic microprocessor.

4.1.3. - MULTIPLE PROCESSOR CONTROLLERS

Microprocessors and large scale integrated circuits have now reached such a level of integration (at a reasonable cost) that the concept of applying multiple processors to meet the performance requirements of engine mounted controllers is now very attractive.

The benefits from such an approach are :

- Enhanced performance achieved through partitioning system functions into tasks handled by individual processors. For instance one of the processors may be used to handle input/output, interrupts, which very often overhead a monoprocessor.

- Modular system expansion capabilities when the appropriate hardware/software have been taken into account at the design level.

- Improved system reliability : generally a failure within a non-redundant monoprocessor is catastrophic for the system. Adding hardware and/or software failure detection capabilities at the level of individual processor provides graceful degradation of a multiprocessor system (14).

- Better cost and better maintainability by using standard subassemblies.

The range of possible multiple processor structures is very open both for hardware and software. Two distinct and opposite approaches may be used :

(a) The first one consists of distributing the function of the system on specialized (well-adapted) processors (figure 15). Such a structure is very attractive if the work to be done is well identified and if there is no need for graceful degradation, in fact it looks like a monoprocessor with powerful peripherals. Therefore it is easy to design and use.

(b) The second approach does not imply specialization of the processors and of their interconnections.

Figure 16 shows an example of a structure implemented with four identical processors.

Two similar systems have been built at ELECMA : one with a 16-bit microprocessor (9900) the other with an 8-bit one (6800). They are used as laboratory supports to investigate the following factors :

- Communication procedures,
- Task allocation,
- Overall bandwith,
- Degradation modes,
- Trouble shooting methods.

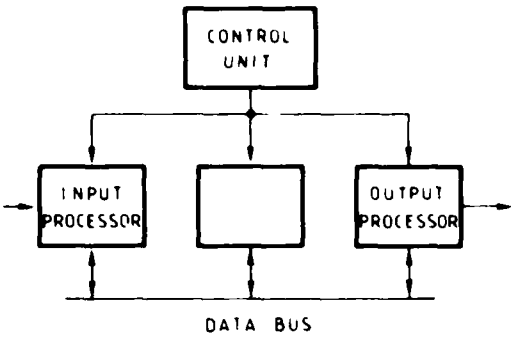


FIG 15

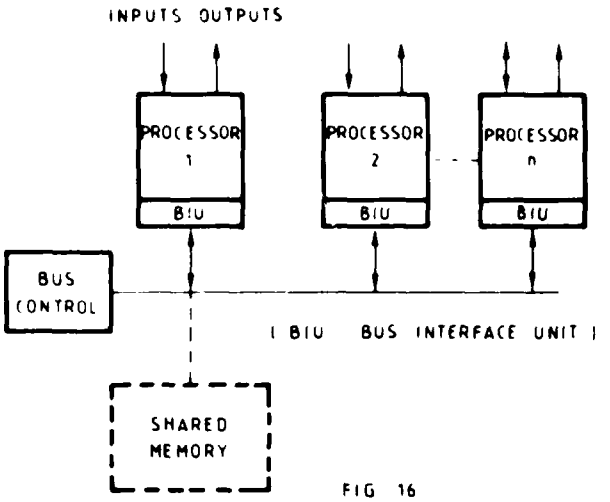


FIG 16

It should be mentioned that it is almost impossible to design and efficiently program multiple processor controllers without the help of evaluation tools.
Especially Petri Nets, task-potential graphs, stochastic processes modelling.

Although designing multiprocessors with microprocessors and LSI circuits is a difficult task their benefits - in terms of performance, cost, reliability - are such that they constitute an attractive and viable way to build trim and full authority controllers.

4.2. - A FULL-AUTHORITY CONTROLLER : ASMARA

ASMARA ("Automate Sûr et Modulaire Adapté aux Régulations Avioniques" : a modular and secure microcomputer adapted to avionic regulation) is not an operational system but just a prospective one. The design of ASMARA has been supported by the "Direction des Recherches et des Essais Techniques". A detailed description of ASMARA can be found in (15).

4.2.1. - SPECIFICATIONS

The specification for a full authority digital to the fully authorative controller for a turbo-jet engine were established by distinguishing three broad categories of specifications :

- Technological specifications directly related to the environment,
- Functional specifications defining the processing to be carried out and the behaviour of the regulation,
- Operational specifications defining the operational safety constraints with which the processing must be carried out.

4.2.1.1. - TECHNOLOGICAL SPECIFICATIONS

The microcomputer is engine mounted, its volume is limited. It is subjected to severe thermal constraints and its components must obey military specifications. The available power supply is liable to have drop-outs as long as 50 ms. The use of batteries or high value electrolytic capacitors is impossible since these elements are not usable at high temperatures.

4.2.1.2. - FUNCTIONAL SPECIFICATIONS

The engine possesses three main functional modes :

- Engine without after-burning (dry engine)
- Engine with primary after-burning (primary AB)
- Engine with total after-burning (total AB).

The control law is made up of six main loops which are distributed according to the engine's functional modes. The table of figure 17 gives the number of operations that must be carried out during parameter acquisition and during execution of the different control loops.

	DRY ENGINE	ENGINE WITH PRIMARY AB		ENGINE WITH TOTAL AB				
	ACQUISITION DRY ENGINE	TACHYMETRIC LOOP	TEMPERATURE LOOP	ACCELERATION LIMIT	ACQUISITION AB	DRY/AB INTERACTION	PRIMARY AB	AB FAN
Addition	3	-	3	-	-	2	2	2
Subtraction	-	1	-	-	-	3	1	2
Multiplication	8	2	1	2	-	5	2	3
Division	5	-	1	1	-	-	2	-
Square root	-	-	-	1	-	-	-	-
One variable function	3	2	2	-	1	4	2	4
Two variable function	-	-	-	1	-	-	-	-
Derivative	-	-	1	-	-	3	-	3
Bonding of derivative	-	-	-	-	-	-	1	-
Threshold	-	-	-	-	-	1	-	1
First order filter	-	-	-	-	-	2	-	2
Corrective network	-	1	-	-	-	-	-	-
Digital filter	8	-	-	-	1	-	-	-
Likelihood control	4	-	-	-	1	-	-	-
Equality control	7	-	-	-	-	-	-	-

FIG.17

4.2.1.3. - OPERATIONAL SPECIFICATIONS

The process being regulated is critical and consequently we must be sure that every failure is detected and that inhibition of the microcomputer is signaled. The regulation has a back-up mode (a simplified version that nevertheless gives better performances than a simple hydromechanical back-up).

Due to the physical location of the microcomputer, maintenance must be possible "in situ".

4.2.2. - IMPERATIVE CONSTRAINTS - BASIC CHOICES

In order to deal with the architectural level we must take into account all the imperative constraints resulting from the specifications which in turn lead us to several necessary choices. The imperative constraints and the resulting necessary choices are :

FULL AUTHORITY CONTROLLER	FAULT TOLERANCE
VOLUME OF TASKS	FAST BIPOLAR PROCESSOR
LOW POWER CONSUMPTION	WITH SWITCHED POWER SUPPLY
TOLERANCE TO POWER DROP-OUT (50 ms)	C/MOS SAFEGUARD MEMORY (CAPACITOR)

4.2.3. - ARCHITECTURAL LEVEL

4.2.3.1. - THE FUNCTIONAL ASPECT

An initial analysis of the functional specifications using Petri nets enables a rough approximation of the architecture of the regulation. This leads to an initial definition (structure SØ) of the microcomputer using a graph of its resources (figure 18).

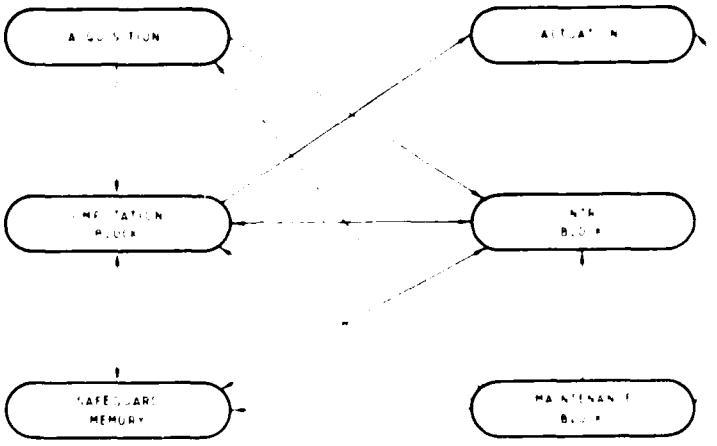


FIG 18

The control block is responsible for organizing the system and for on/off powering of the computation block. Since the simplified regulation only requires a relatively small volume of computation, it may be implemented by the control block. The safeguard memory saves the vital data during transient power failures. The maintenance block is non-resident and has been represented symbolically as being able to access every element of the structure, if not physically then at least by test procedures.

Since the computation block is the one which decides the functional performance of the structure, we have considered the following structures :

- Monoprocessor (MONO)
- Biprocessor (BI)
- Triprocessor (TRI)
- Quadriprocessor (QUADRI)

The restriction to four processors comes from the specified volume constraints.

The large number of multiplications to be carried out also led us to consider the performance of the preceding structures when associated with a hard-wired high-speed multiplier.

The functional performances were thus forecast using a scheduling program "MILORD" (9). The execution times of the different operations were calculated assuming the use of a processor based on INTEL 3000 family.

The results are given in figure 19 where the bold lines represent the execution times given by sub-optimal scheduling program and the dotted lines represent the theoretically attainable minimum time. These results show that the degree of parallelism is such that the greatest gain in time is obtained with two processors. Due to the time allowed for the processing and the level of accuracy of our evaluations, we have chosen the monoprocessor configuration associated with a fast multiplier.

4.2.3.2. - THE OPERATIONAL ASPECT

Starting with the basic structure S0, we have envisaged the implementation of increasing levels of redundancy. However keeping in mind the volume constraint which was imposed, this led us to the selection of four possible operational structures :

- S1 : Obtained from structure S0 by adding failure detection devices,

- S2 : Obtained using functional redundancy by implementing a simplified regulation within the control block,

- S3 : Obtained using hardware redundancy by letting both blocks tolerate one fault,

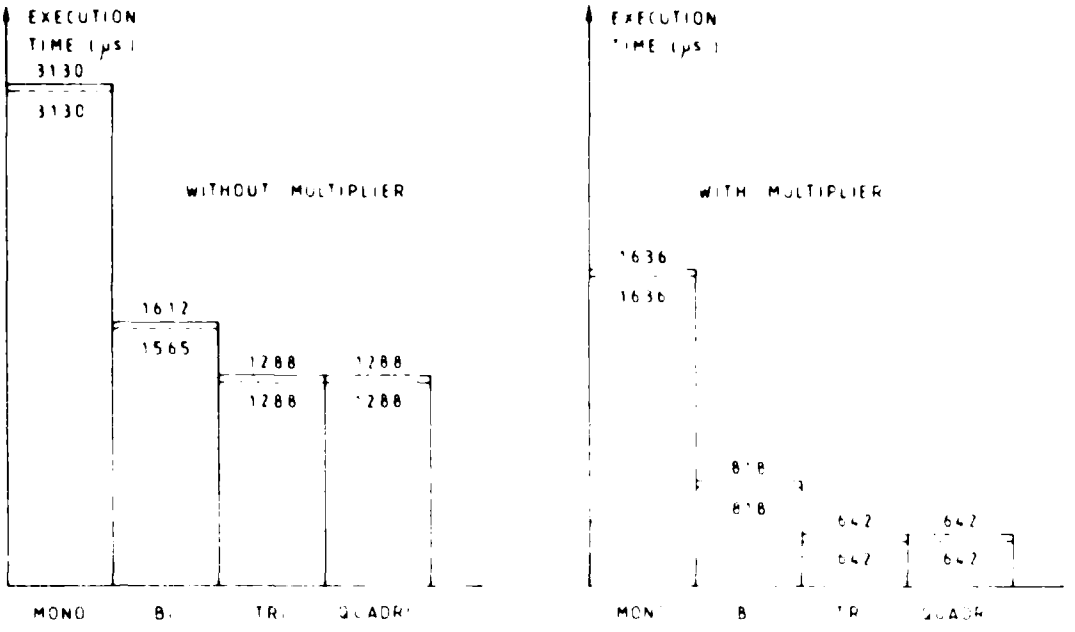


FIG 19

- S4 : Obtained using hardware redundancy by letting both the control and computation blocks tolerate one fault.

The calculation of the reliability and safety of each structure was carried out using Markov processes (12). For instance the state diagram of figure 20 gives for S3 the three operating modes related to an increasing number of failures.

The operational security components that were chosen for the evaluation are reliability and safety.

Safety can be defined as the probability of no catastrophic failure. This notion of catastrophic failure is extremely dependent on the application considered. In process control we have defined a catastrophic failure as one that leads to the delivery of erroneous orders.

Due to the chosen definition of safety (probability that the structure functions correctly or that it has been subject to a detected fault) it was necessary to take into account the efficiency of the detection procedures. This was done using the parameter P_d defined as follows :

P_d =Pr.(a fault is self-detected in a unit/a fault has occured in that unit).

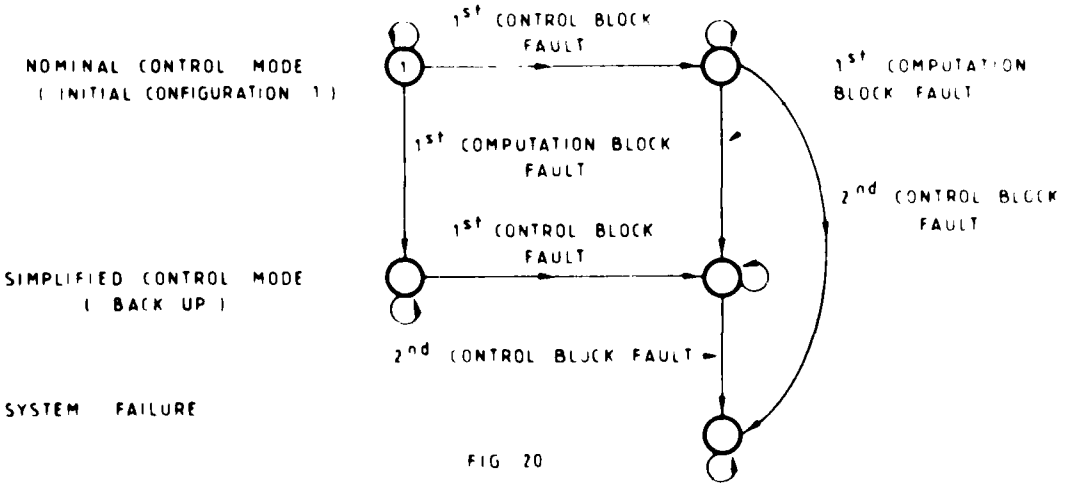


FIG 20

The presence of two control algorithms, the nominal one and the back-up one, must be taken into account during the evaluation of structures S2, S3 and S4. This led us to define two types of reliability and safety according to whether we consider execution of either only the nominal or the back-up regulation.

The curves of reliability and safety of the different structures are given in function of ΛT in figures 21 and 22 where :

- Λ is the total failure rate of the circuits required for the back-up regulation,
- $k\Lambda$ is the total failure rate of the other circuits specifically required for the nominal regulation.

An examination of these two sets of curves leads to the following remarks :

- Whatever the failure rate ratio k, the best reliability of the nominal regulation is that of structure S4, structure S3 following close behind,
- The nominal safety performance of structure S1 is better than or equal to that of the other structures ; the latter all have the same safety performance,

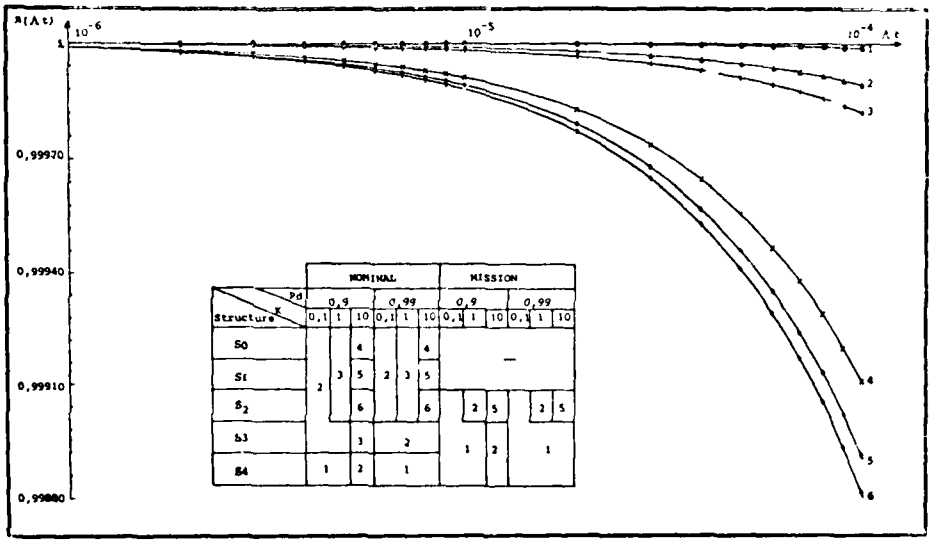


FIG 21

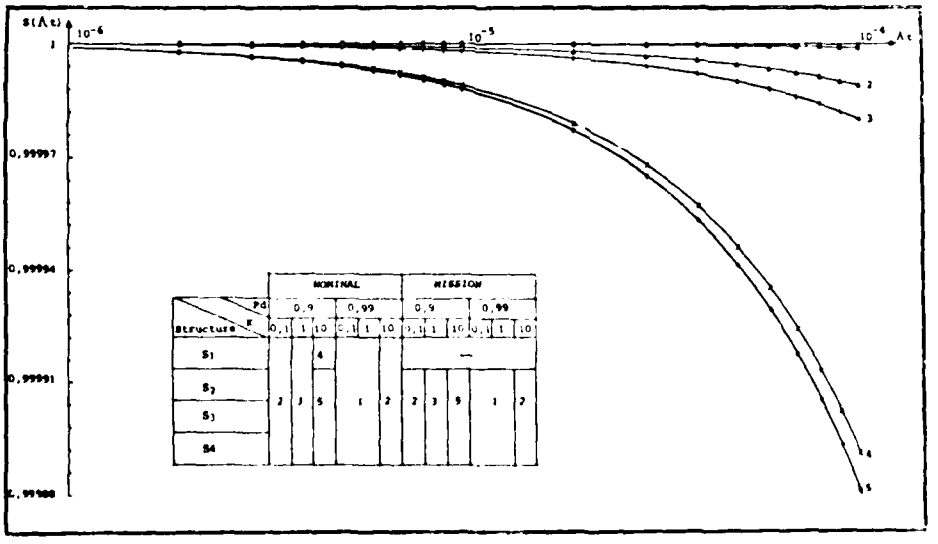


FIG 22

- A better detection efficiency reduces the differences between the safety performances of structures S1, S2, S3 and S4,

- the mission reliability and safety of structures S2, S3 and S4 are not different enough to be able to classify them.

4.2.3.3. - CONCLUSIONS (at the architectural level)

The following conclusions can be drawn at this level of description :

- On the functional aspect :

  . the microcomputer's hierarchical structure is essentially made up of a monoprocessor control block and a computation block with a switched power supply.

  . the computation block's structure is made up of a processor associated with a hardwired multiplier,

- On the operational aspect : structure S4 can be eliminated due to its high volume/performance ratio ; a better compromise is obtained by choosing structure S3.

The functional block diagram of the microcomputer is given in figure 23.

4.2.4. - HARDWARE LEVEL

The hardware level will enable us :

- To define the basic functional structure S0 which we will use as a reference,

- To choose and to implement the fault detection and fault recovery techniques in relation to the different structures,

- To evaluate more accurately the operational security components that arise as choice criteria.

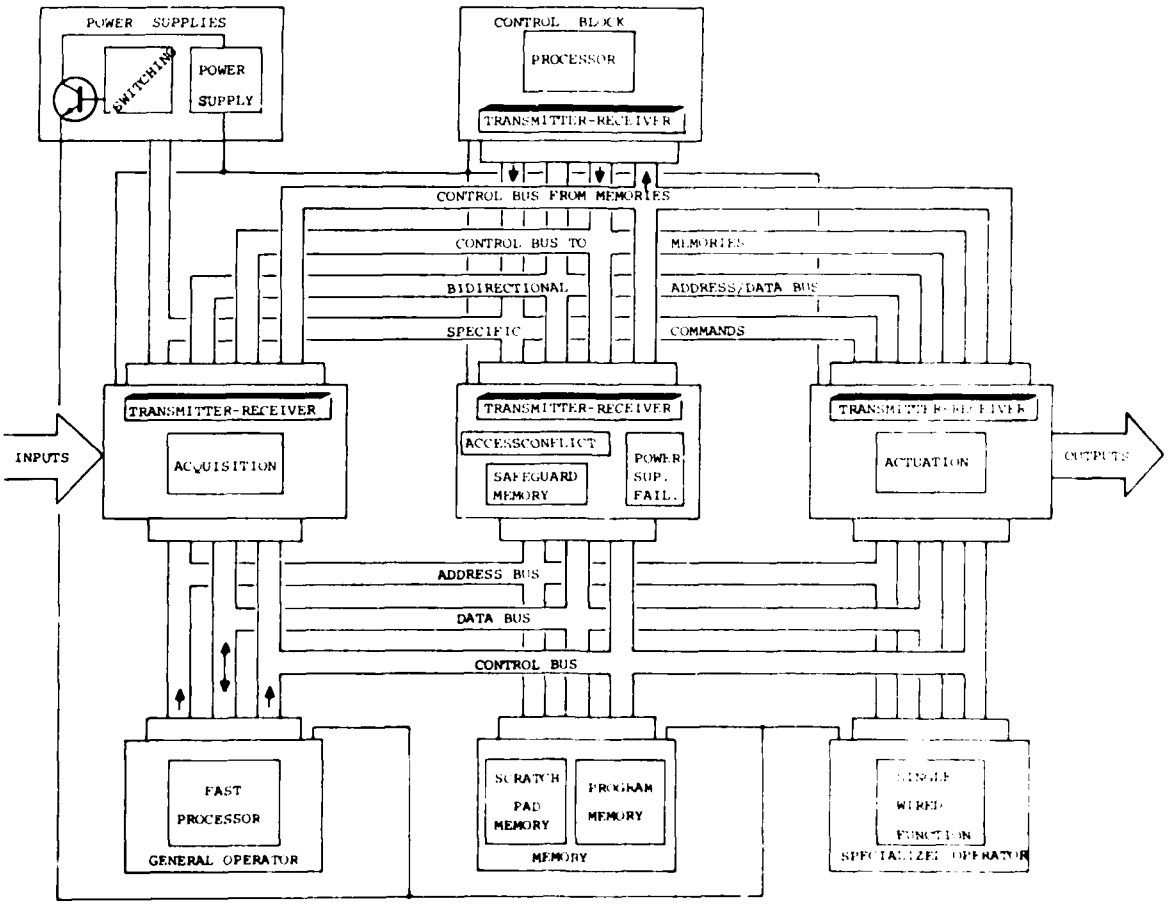


FIG 23

4.2.4.1. - DETAILS OF STRUCTURE S0

Structure S0 is made up of four blocks : the control block, the acquisition block, the actuation block, the computation block.

The principal components used for its realization are given in the table of figure 24.

CONTROL BLOCK	CENTRAL UNIT	Microprocessor INTERSIL IM 6100
	PROGRAM MEMORY	ROM 1K x 12 bits
	SCRATCH PAD MEMORY	RAM 512 x 12 bits
	SAFEGUARD MEMORY	2 Banks of RAM 128 x 16 bits
ACQUISITION BLOCK		Analog Multiplexor 9 channels A/D Converter 12 bits RAM 16 x 12 bits
ACTUATION BLOCK		5 x D/A Converters 12 bits 5 x Registers 12 bits 1 x Register 6 bits
COMPUTATION BLOCK	CENTRAL UNIT	MCU 3001 Bit slice Microprocessor 8 x CPE 3002 ROM 1K x 32 bits INTEL 3000 series 2 x Multiplier parallel/serial 8 x 1
	PROGRAM MEMORY	ROM 1K x 16 bits
	SCRATCH PAD MEMORY	RAM 256 x 16 bits

FIG. 24

4.2.4.2. - CHOICE AND IMPLEMENTATION OF THE FAULT DETECTION TECHNIQUES

The choice and implementation of the fault detection techniques (16) are carried out under the following assumptions :

- A module failure corresponds to an integrated circuit failure defined as follows : the failure of an integrated circuit is a single or multiple stuck at Ø or stuck at 1 fault affecting any number of inputs and/or outputs,

- A module is said to be totally fault tolerant if and only if it detects and corrects the first failure,

- The safety constraints lead us to supply a switch-over to the hydro-mechanical regulation in the event of a failure of the back-up regulation ; it is thus necessary to detect the second failure.

The table of figure 25 gives fault detection and tolerance techniques implemented on each of the sub-systems making up the three structures envisaged. These techniques result from an evaluation of the required hardware thus enabling us to compare the different fault detection and tolerance techniques with the aim of achieving local optimization. This leads to a total hardware volume less than that obtained with a duplex system (fault detection by comparison).

The reliability and safety curves of the different structures obtained by processing their respective state-graphs are given in relation to time in figures 26 and 27.

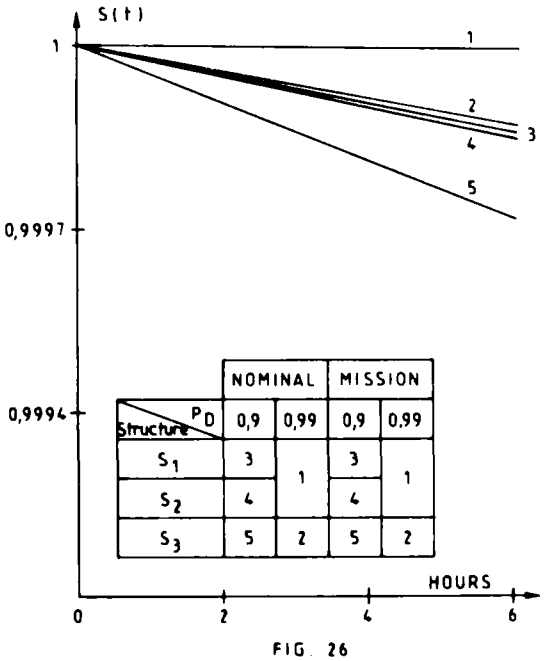
Analysis of the results enables us to draw the following conclusions :

- Structures S1 and S3 have complementary characteristics,
- Structure S2 is on all respects slightly worse than structure S1.

Structure S3 was thus chosen since it seems to be the best compromise between volume and operational performance ; its block diagram is given in figure 28.

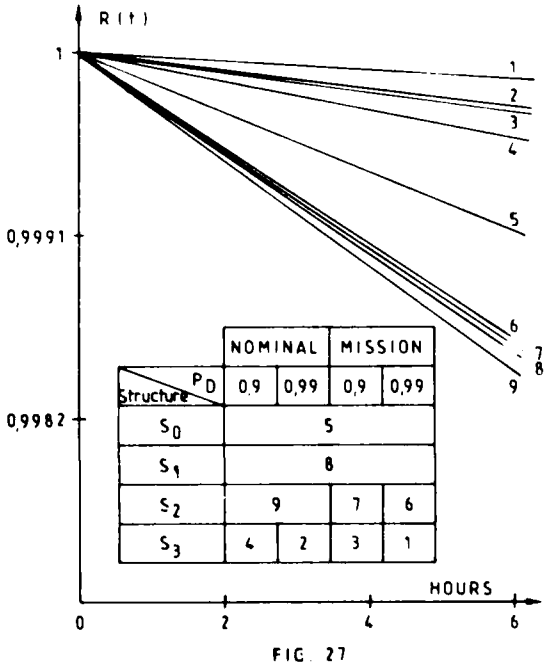
		FAULT DETECTION			FAULT TOLERANCE	
		S1	S2	S3	S2	S3
COMPUTATION BLOCK		Likelihood control by the control block			Back-up regulation on the control block	
CONTROL BLOCK	Processor	Doubling and comparison				Standby redundancy
	Bus	Parity		Double error detecting single error correcting code		Double error detecting single error correcting code
	Memory	Single error correcting code				
SAFEGUARD MEMORY						
ACQUISITION	Dedicated to the back-up regulation	Likelihood Control by the control block		Acquisition of hardwired values		Standby redundancy
	Dedicated to the nominal regulation					
ACTUATION	Dedicated to the back-up regulation	Likelihood control by the control block		Feedback of the outputs on the acquisition		Standby redundancy
	Dedicated to the nominal regulation					

FIG.25



Structure \ P_D	NOMINAL 0,9	NOMINAL 0,99	MISSION 0,9	MISSION 0,99
S_1	3	1	3	1
S_2	4		4	
S_3	5	2	5	2

FIG. 26



Structure \ P_D	NOMINAL 0,9	NOMINAL 0,99	MISSION 0,9	MISSION 0,99
S_0	5			
S_1	8			
S_2	9		7	6
S_3	4	2	3	1

FIG. 27

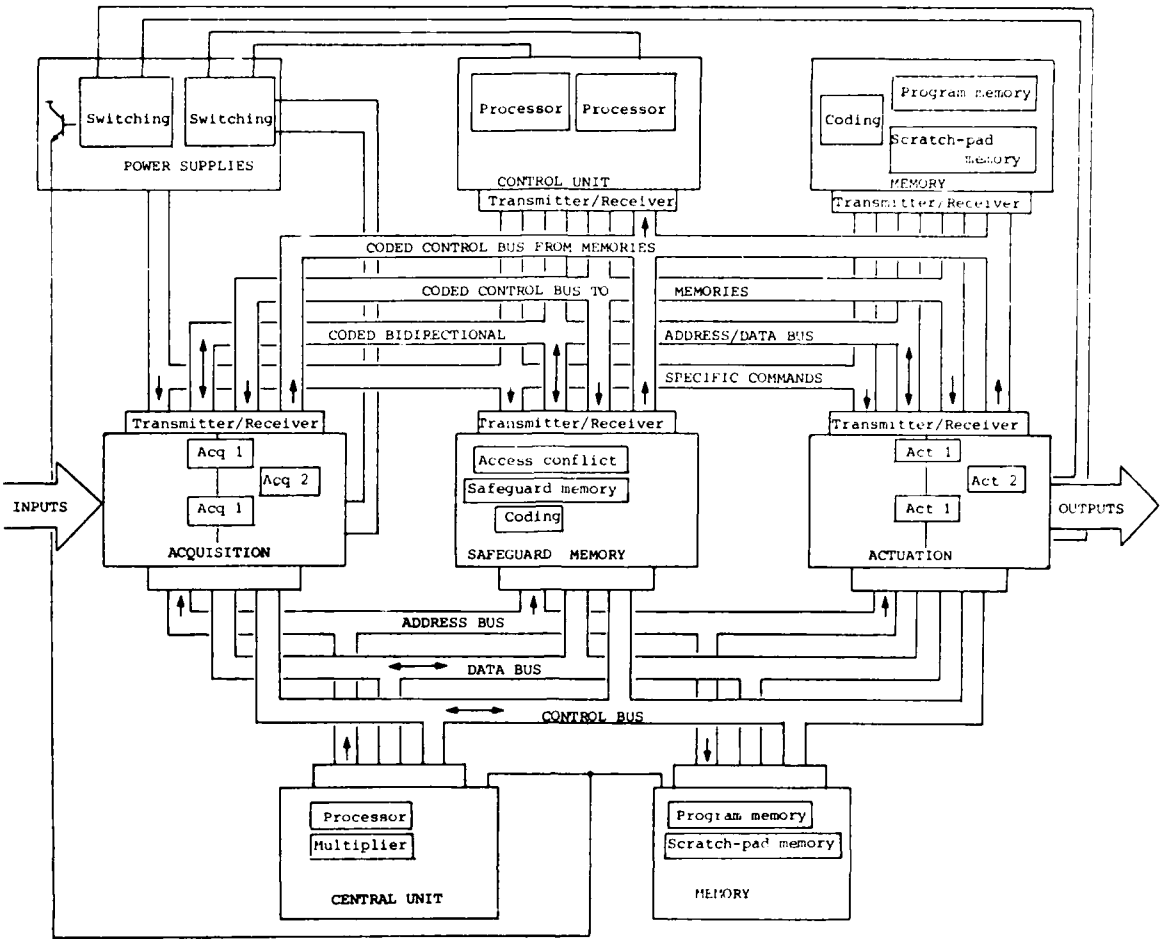


FIG.28

4.2.4.3. - REALIZATION LEVEL

A prototype has been constructed in order to verify the numerous choices carried out during the architectural and hardware levels and to demonstrate the effectiveness of the models chosen during the different evaluations. This prototype has a structure that is a materialization of structure S3.

The table of figure 29 gives the hardware balance of ASMARA (not including low level signals conditionners and the maintenance interface).

I.C. SIZE	40 Pins	24/28 Pins	16/14 Pins	Discrets element sockets	TOTAL
CONTROL BLOCK	4	1	114	24	143
MEMORY OF THE CONTROL BLOCK	0	9	61	3	73
SAFEGUARD MEMORY	0	4	128	11	143
CALCULATION BLOCK	1	9	60	12	82
CALCULATION BLOCK MEMORY	0	6	36	5	47
	5	29	399	55	488

FIG.29

You may notice the high percentage of low level integrated circuits that are required. This is explained by two facts :

- The lack of circuits associated with the main integrated circuits (processor, I/O management, etc...),

- The absence of large scale integrated circuits specific to the functions of fault detection (self-checking checkers, voters, etc...).

5. - CONCLUSION

The availability of large-scale integrated circuits enables ONE to obtain small volume, low consumption digital systems with very high computational power.

This enables a considerable increase in the possibilities of the engine control system. However, an increasing dependency of the process on its control system means that the latter must be more perfect and invulnerable.

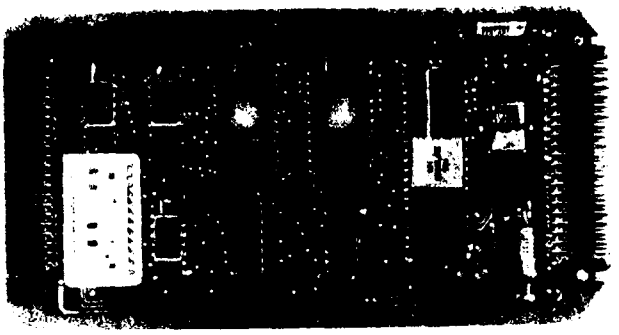
This goal can be achieved if one designs and realizes systems which although functionally complex are such that a fault is a naturally foreseen and tolerated event.

As this paper shows, the realization of such systems must obey the following steps :

- At the design level, the use of a rigorous and well-structured methodology that enables one to take account of numerous functional and operational constraints that are often in conflict.

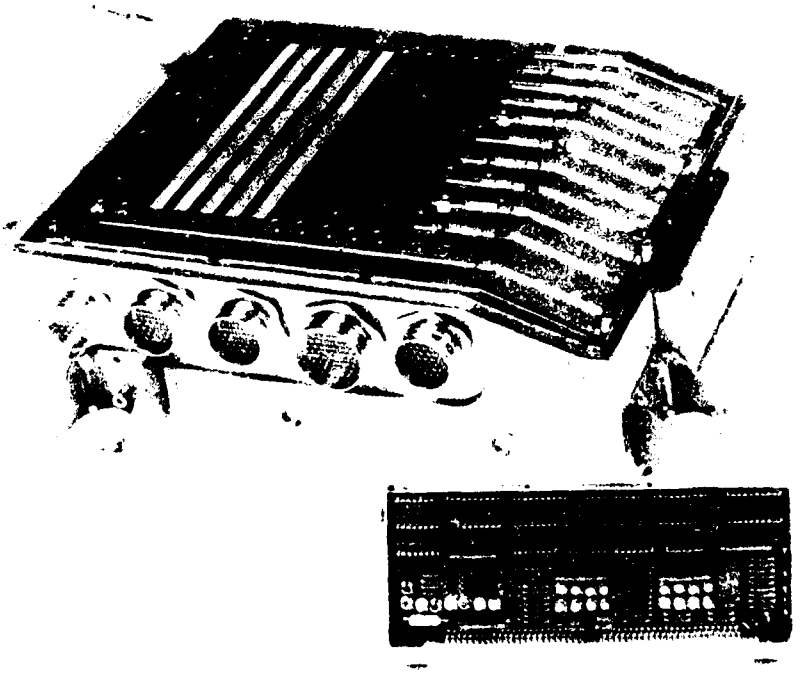
- At the hardware implementation level, the use of sufficiently powerful and integrated components that enable the implementation of the design with reasonable dimensions.

This is now possible thanks to recent progress both in the integrated circuit technology and in the forecast of functional and operational performance of digital systems.



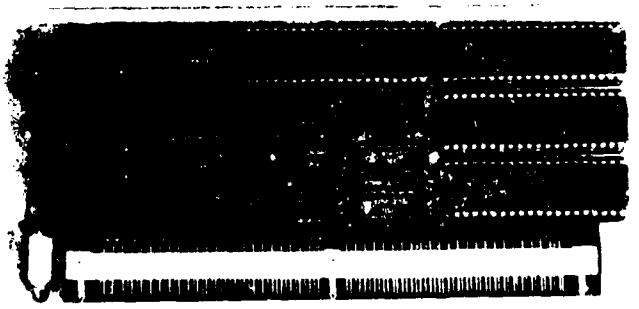
1 inch

PHOTOGRAPH 1
A CMOS CPU



PHOTOGRAPH 2
DIGITAL CONTROLLER

1 inch



PHOTOGRAPH 3
A 16 bit ALU

BIBLIOGRAPHY

(1) COLLET-BILLON A., "Les équipements pour engins balistiques"
Air et Cosmos n° 699, 1977.

(2) NEWIRTH D.M., "Engine reliability with electronic controllers"
Proceedings of Annual Reliability and Maintainability, 1975.

(3) PRICE D.C., "Trends in failure survival techniques for avionic systems"
The Radio and Electronic Engineer, July 1976.

(4) BARBOT A., "Evolution des systèmes de régulation des turboréacteurs"
L'Aéronautique et l'Astronautique, n° 54, Mai 1975.

(5) SU H.Y.S. "A survey of computer hardware description languages in the USA"
Computer, Dec. 1974.

(6) RAMCHANDANI C. "Analysis of asynchronous concurrent systems by Petri Nets"
Pr. D. Thesis, MIT, 1973.

(7) HACK M.H.T., "Analysis of production schemata by Petri Nets"
Master of Science Thesis, MIT 1972.

(8) MAUREL E., ROUX D., DUPONT D., "Techniques opérationnelles d'ordonnancement"
Eyrolles, 1977.

(9) DIBON L.M., "Ordonnancement et potentiels, Méthode MPM"
Hermann, 1970.

(10) ASTOLFIM, VAN DEN MUYZENBERG C.L., CONTINIS "Une technique pour l'analyse sur mini-calculateurs des grands arbres de défaillance".
Colloque international sur la fiabilité et la maintenabilité, Paris 1978.

(11) LANDRAULT C., "Prévision de la sûreté de fonctionnement des systèmes numériques réparables"
Thèse de Doctorat ès-Sciences, Institut National Polytechnique, Toulouse, Mars 1977.

(12) LAPRIE J.C., "Prévision de la sûreté de fonctionnement et architecture de structures numériques temps réel réparables"
Thèse de Doctorat ès-Sciences, Université Paul Sabatier, Toulouse, Juin 1975

(13) COLLIN J.M., GAJ B., "Application des microprocesseurs à la régulation des moteurs d'avions militaires"
AGARD 274.

(14) BELLON C. "Graceful degradation in fault tolerant multiprocessor system"
Euromicro Venice 1976 - NORTH HOLLAND

(15) BEOUNES C., "Automate sûr et modulaire adapté aux régulations avioniques : ASMARA"
Thèse de Docteur-Ingénieur, Université Paul Sabatier, Toulouse, Novembre 1977.

(16) WAKERLY John "Error detecting codes self-checking circuits and applications".
The computer science library - NORTH HOLLAND NY

FULL AUTHORITY DIGITAL ELECTRONIC CONTROL TURBOFAN ENGINE DEMONSTRATION

by
R. W. Vizzini
Advanced Development Div.
Naval Air Propulsion Center
Trenton, NJ

T. G. Lenox
Commercial Products Div.
Pratt & Whitney Aircraft Group
East Hartford, CT

R. J. Miller
Government Products Div.
Pratt & Whitney Aircraft Group
West Palm Beach, FL

ABSTRACT

This paper describes the design, demonstration and evaluation of a Full Authority Digital Electronic Control (FADEC) capable of controlling an advanced variable cycle gas turbine engine in an advanced supersonic Navy fighter aircraft application.*

The FADEC design incorporates many advanced technology features including the latest microelectronics, extensive fault tolerance capability, and high-speed digital communication using a fiber optic data link. The advanced technology FADEC system was successfully demonstrated in a comprehensive test program, which included open loop environmental bench testing, closed loop bench testing, and testing on an F401 afterburning turbofan engine at sea level and at nine altitude conditions from 7000 to 50,000 ft, and at Mach numbers from 0.3 to 1.6. This testing was a major milestone in a program to design, develop, and demonstrate an engine-mounted full authority digital electronic control for advanced military aircraft engine application in the 1980's and beyond.

Over 7000 hr of electronic control operation were achieved during this program. Over 1100 hr of testing were achieved with the engine-mounted control unit, which included over 68 hr of engine testing without a hardware malfunction. In addition to the advanced electronic circuitry employed in the FADEC, the first demonstration of optic communication with engine-mounted equipment was achieved.

The success of the FADEC program to date has led to plans to flight test the FADEC in either an F-15 or F-14 aircraft in the early 1980's.

INTRODUCTION

The challenge of high-performance, multi-mission aircraft has resulted in a profound increase in gas turbine engine thrust-to-weight ratio and a corresponding increase in control system requirements. The Full Authority Digital Electronic Control (FADEC) program, sponsored by the Naval Air Systems Command, directed by the Naval Air Propulsion Center, and performed by Pratt & Whitney Aircraft was initiated in 1976 with the realization that these current and projected high-performance engines were stretching hydromechanical control technology to its limits. Full authority electronic control may be the only practical solution to providing satisfactory engine control for advanced engines.

The FADEC program objective was to design, fabricate and test an engine-mounted, flight-type, digital electronic control for advanced military aircraft gas turbine engines. This control was designed to satisfy increased engine control functional and environmental requirements, reduce life cycle costs and improve reliability and maintainability. The FADEC system functional requirements were selected to satisfy a most demanding set of engine control variables as illustrated in figure 1. The FADEC system concept features gas generator fail-operational fault tolerance capability through the use of redundant sensing, computation, and command paths, parameter synthesis and self-test techniques. Incorporation of advanced electronic circuit technology and a reduction in the use of complex hydromechanical hardware are projected to result in more than a 43% reduction in acquisition cost, a 26% reduction in weight, and a 130% improvement in piece part reliability on an overall control system basis relative to a system configured with the latest current production control technology. Other FADEC payoffs include optimized engine performance, stability and capability for aircraft control system integration.

*This paper was printed as SAE No. 801199 for Aerospace Congress & Exposition, Los Angeles Convention Center, Oct. 13-16, 1980.

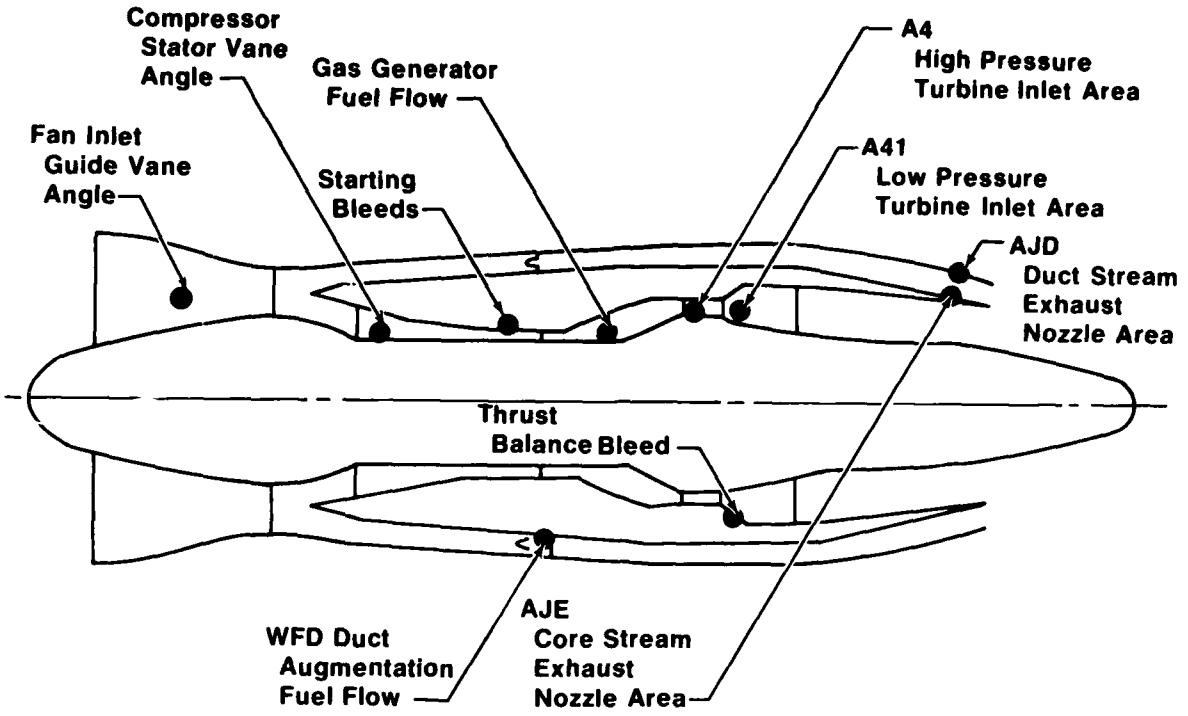


Figure 1. Advanced Engine Requirements

The FADEC program was a 41-month, two-phase program with the activities as illustrated in figure 2. During Phase I comprehensive design trade studies were conducted to establish the overall system requirements. The major design trade studies are listed in figure 3. The detail system requirements resulting from these design studies were reported in References 1 and 2. During Phase II the demonstration FADEC control was designed and fabricated. The intent of this paper is to report on this Phase II demonstration control design and performance and the bench and engine tests that were used to evaluate the design.

Preliminary plans for flight test of the FADEC in the NASA Integrated Research Aircraft Control Technology (INTERACT) program are discussed at the end of the paper.

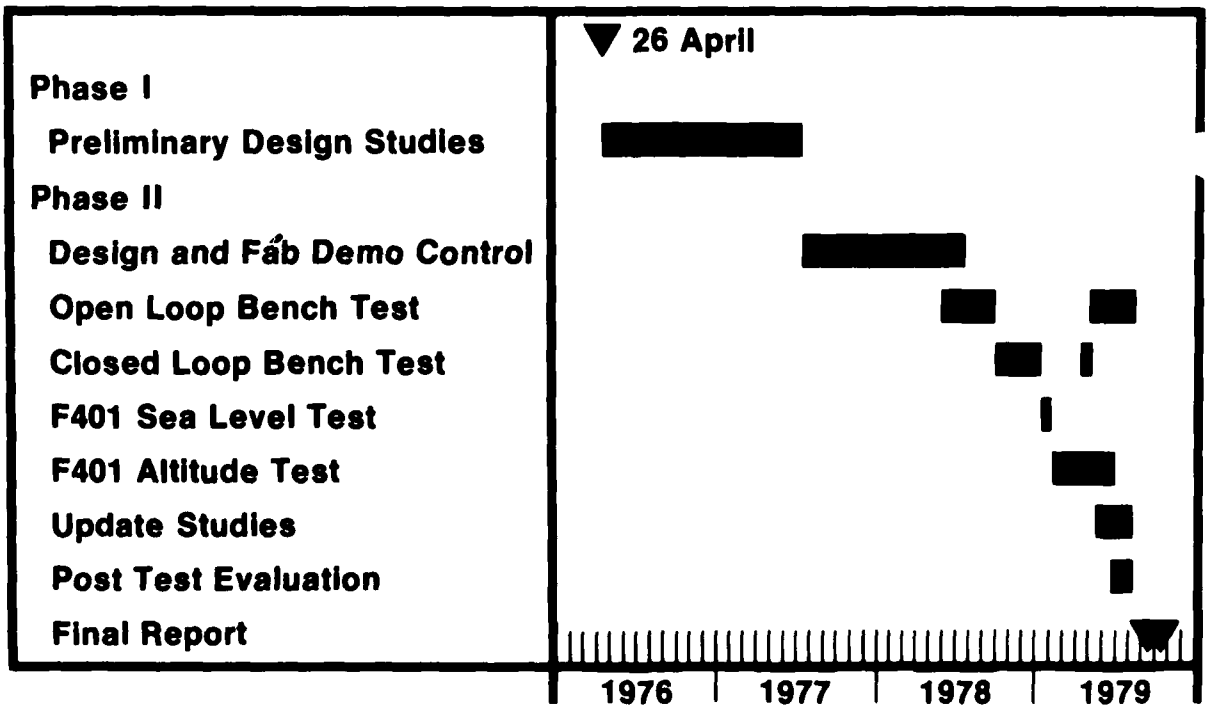


Figure 2. P&WA FADEC Program Schedule

- Circuitry selection
- Backup control
- Cooling
- Power supply
- Sensors
- Fuel pump system
- Actuators
- Signal transmission
- Condition monitoring/diagnostics
- Inlet/aircraft integration
- Maintainability

Figure 3. Design Trade Studies

FADEC Design

The design studies performed in the first phase of the FADEC program, described in Reference 2, defined a concept featuring fail-operational fault tolerance capability through use of redundant electronic sensing, computation, power supply, and command paths. In the P&WA FADEC concept, the electronic control is organized into primary and secondary sections, as shown in figure 4, each with its own digital processor, memory, and complement of input and output signal conditioning circuitry. Both the primary and secondary sections are capable of operating the engine from startup to maximum nonaugmented power. The primary also incorporates circuitry necessary for augmentation control functions. Digital communication between the primary and secondary processors allows exchange of sensed and calculated information to provide extensive fault tolerance capability. Parameter synthesis provides for continued safe operation of the engine in the event that measurement capability is disabled. Separate dual windings are employed on electrohydraulic interfaces, with each winding dedicated to one section of the controller. Output switching logic is employed such that only one processor commands a particular output at a given time, but individual outputs can automatically be transferred from primary to secondary command. Should a processor or power supply malfunction occur in the primary, total command is automatically transferred to the secondary section. Comprehensive software and hardware features are incorporated to identify malfunctions within the controller and other system components for facilitating maintenance and for implementing redundancy management.

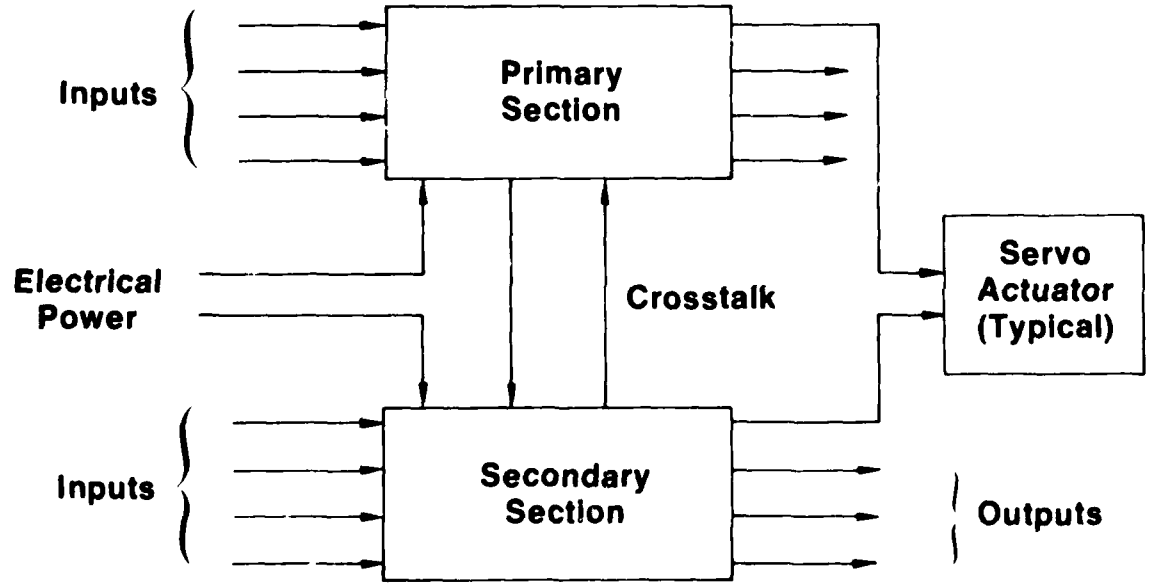


Figure 4. FADEC System Concept

Many advanced technology features are incorporated in the Pratt & Whitney Aircraft FADEC design. The 16-bit parallel digital processor uses very large scale integration (VLSI) complementary metal oxide semiconductor/silicon on sapphire (CMOS/SOS) circuit technology for high computational speed with low power requirements. CMOS/SOS is also used for the random access memory (RAM). Digital communication between processors is implemented with dual port RAM to provide rapid information access, yet prevent a malfunction in one processor from disabling the other. High-density devices are used for nonvolatile programmable read only memory (PROM) to contain the control program and data. Electrically alterable read only memory (EAROM®) is also incorporated to permit modification of particular stored control parameters without requiring physical parts replacement and to enter fault detection data to direct maintenance actions. Fiber optic data links provide high-speed serial-digital communication capability using a light-emitting diode as a radiation source. Precision pressure measurements are provided by vibrating cylinder transducers located within the control unit. Closed loop control modes are employed to provide accurate and responsive control of thrust and critical parameters for engine protection and eliminate the need to trim the FADEC for engine tolerance or deterioration.

The P&WA FADEC demonstrator control design implemented the primary section of the controller in a flight-weight, fuel-cooled, engine-mounted unit, shown in figure 5, which incorporated all of the advanced technology features mentioned above. The secondary section, located in an off-engine breadboard circuit console as shown in figure 6, incorporated reprogrammable core memory to maximize development flexibility. In a production design, both the primary and secondary sections would be packaged in a single unit to minimize cost and weight. In the demonstrator system, a 150-ft fiber optic cable was used to provide high-speed digital communication between the engine-mounted unit and the breadboard unit and simulates a data link between the engine and other aircraft systems for propulsion system integration in future aircraft applications.

Figure 5. Full Authority Digital Electronic Control (FADEC)

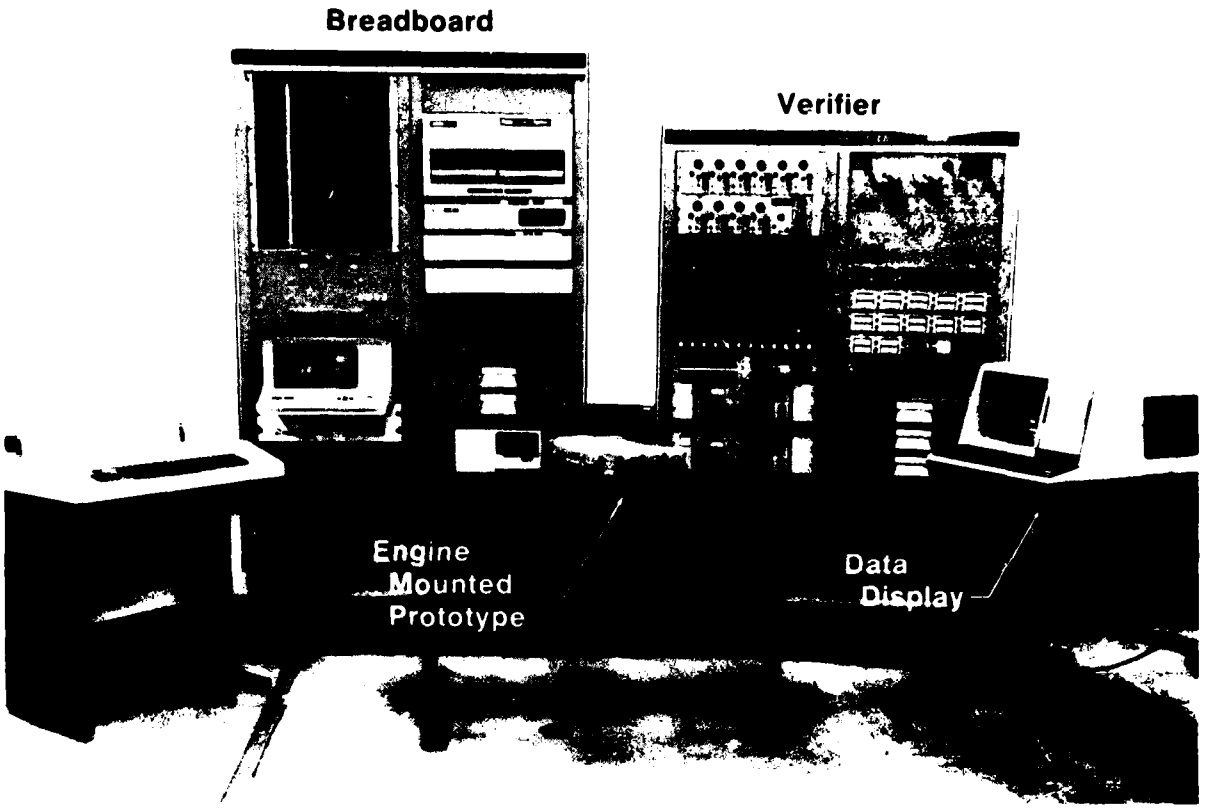


Figure 6. FADEC Dual Channel System

The FADEC system incorporated input and output capacity for control of an advanced variable cycle engine configuration typifying the sophisticated, high-performance engines projected for future supersonic military aircraft applications. The P&WA F401 turbofan engine, selected as a vehicle to demonstrate FADEC technology in a realistic engine environment, required approximately 70% of the functional capability provided in the FADEC design. The F401 engine is representative of the most advanced current military power plants, incorporating variable fan and compressor geometry as well as fully modulated afterburner and exhaust nozzle.

The FADEC engine-mounted unit utilized a packaging configuration shown in figure 5, which is the same approach defined during FADEC design studies. This approach incorporated environmental tolerance/reliability enhancement through vibration isolation mounts, fuel cooling for supersonic operation, and pin fin heat exchanger for supplemental air cooling during subsonic operation. Maintainability features include modular design, quick disconnect electrical cabling, and a handle to facilitate transport and installation. The demonstrator unit measures 22 × 43 × 12 cm (8.75 × 17 × 5 in.), weighs 13.6 kg (30 lb), and has a maximum power dissipation of 110 watts.

Test Program

A comprehensive development program was conducted on the FADEC demonstrator control system. The program included functional and environmental open loop bench testing, closed loop bench testing in both variable cycle engine and F401 configurations, and sea-level and simulated altitude testing with an F401 engine. The objective of this testing was to demonstrate the feasibility of the FADEC system concept and to verify the functional integrity of the FADEC design for engine-mounted operation throughout the aircraft flight environment. Two complete sets of engine-mounted and breadboard controllers were fabricated and used for this testing.

Bench testing was initiated at Hamilton Standard early in 1977 with the breadboard control unit, confirming that all FADEC circuitry performed within design performance predictions over the entire range of operating temperature. The breadboard controls were also used extensively for development of software for both the variable cycle engine and F401 system configurations, and for checkout of the engine-mounted unit circuit board assemblies as they became available. Operation of primary and secondary controls with the fiber optic communication link was developed initially with the two breadboard units, displacing one breadboard with the engine-mounted unit when it became operational as an assembly.

Environmental testing was conducted at Hamilton Standard using simulated input signals and output loads. This testing included portions of MIL-E-5007D component assurance tests and other specific tests designed to verify the capability of the engine-mounted unit to operate satisfactorily within the complete control system. Alternator and output compatibility were verified over the entire range of operating temperature. Thermal mapping was conducted over a range of coolant and ambient temperatures, verifying the effectiveness of the thermal design. MIL-E-5007D high- and low-temperature endurance and vibration testing demonstrated the functional and physical integrity of the FADEC design. Electromagnetic interference (EMI) testing was conducted to MIL-STD-461A (paragraph 6.19.1 Notice 3) levels and beyond. This testing demonstrated satisfactory operation with fuel temperatures from 222°K (−60°F) to 361°K (+190°F), ambient temperatures from 218°K (−65°F) to 516°K (+470°F), vibration up to 20 G's 2 kHz; and EMI to over 130 volts/meter, frequencies to 10 GHz. Three hundred thirty eight hours of operation were accrued at component temperatures greater than 344°K (+160°F).

Closed loop bench testing was conducted at P&WA using computer simulations of the variable cycle engine and the F401. This testing demonstrated the total functional capability of the FADEC system in the variable cycle engine configuration. In the F401 configuration, closed loop bench testing verified compatibility of the FADEC system with fuel handling and actuation system components, and demonstrated steady-state and transient performance over the range of operation between idle and maximum augmentation for sea level and the nine altitude test conditions. The extensive fault tolerance capability of the FADEC was demonstrated by simulating various malfunctions in sensing and command paths within the system. Figure 7 shows the FADEC engine-mounted unit operating with fuel handling and actuation hardware during F401 closed loop bench testing. Twelve hundred fifty hours of control operation including 630 hr on the engine-mounted units, were accrued in closed loop bench testing.

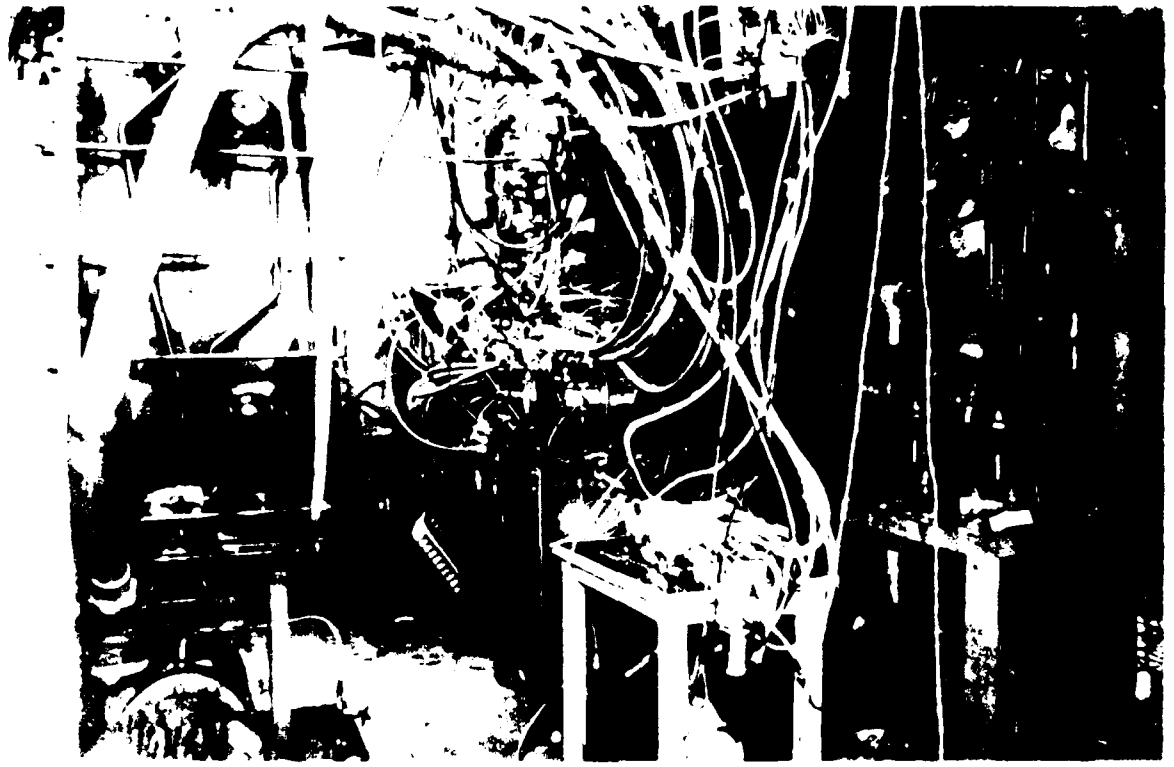


Figure 7 FADEC Closed Loop Bench Setup

Following F401 closed loop bench testing, the FADEC system was installed on a Navy F401 afterburning engine for sea-level testing at the P&WA West Palm Beach facility. Sea level testing encompassed the entire range of engine operation. Forty-three hours of engine operation were accomplished over a 3 wk time period during March 1979. Included were the most demanding transients, including Bodie accelerations. A variety of malfunctions were simulated with the FADEC operating the engine to demonstrate its extensive fault tolerance capability. A photograph of the F401 operating at maximum afterburning with the FADEC system is shown in figure 8. Figure 9 illustrates transient data acquired in the course of sea-level testing, showing the rapid smooth and stable operation provided by the full authority control with closed loop control modes.

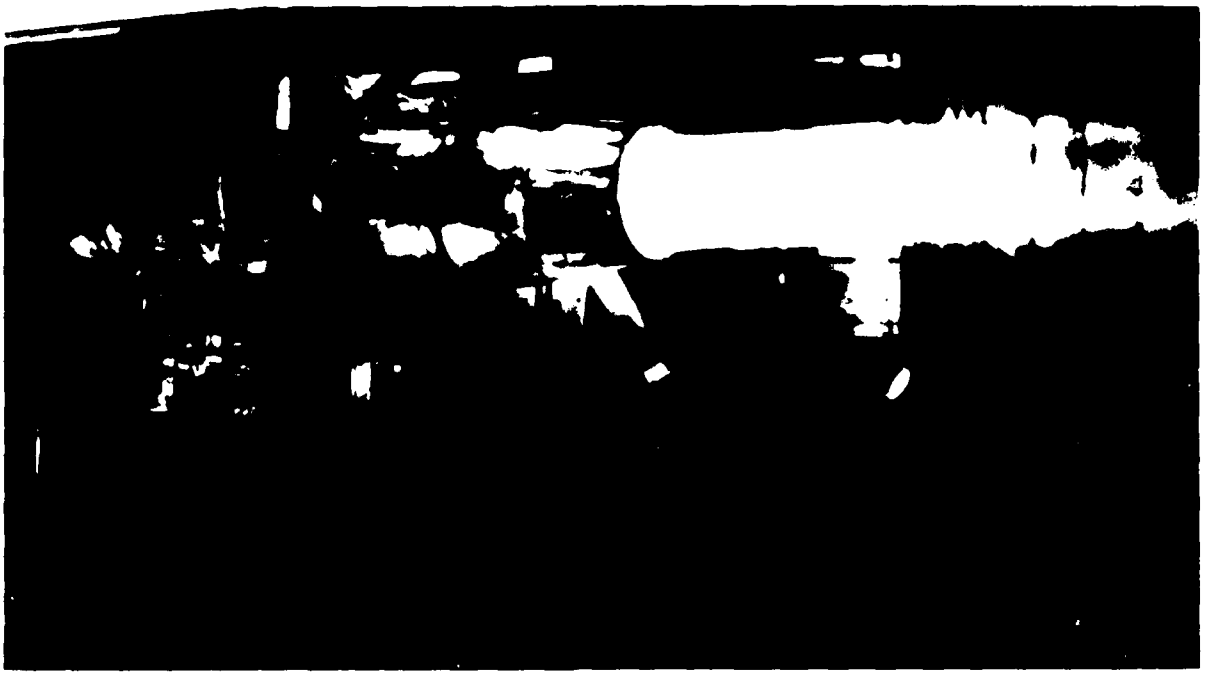


Figure 8. F401 in Maximum Afterburning

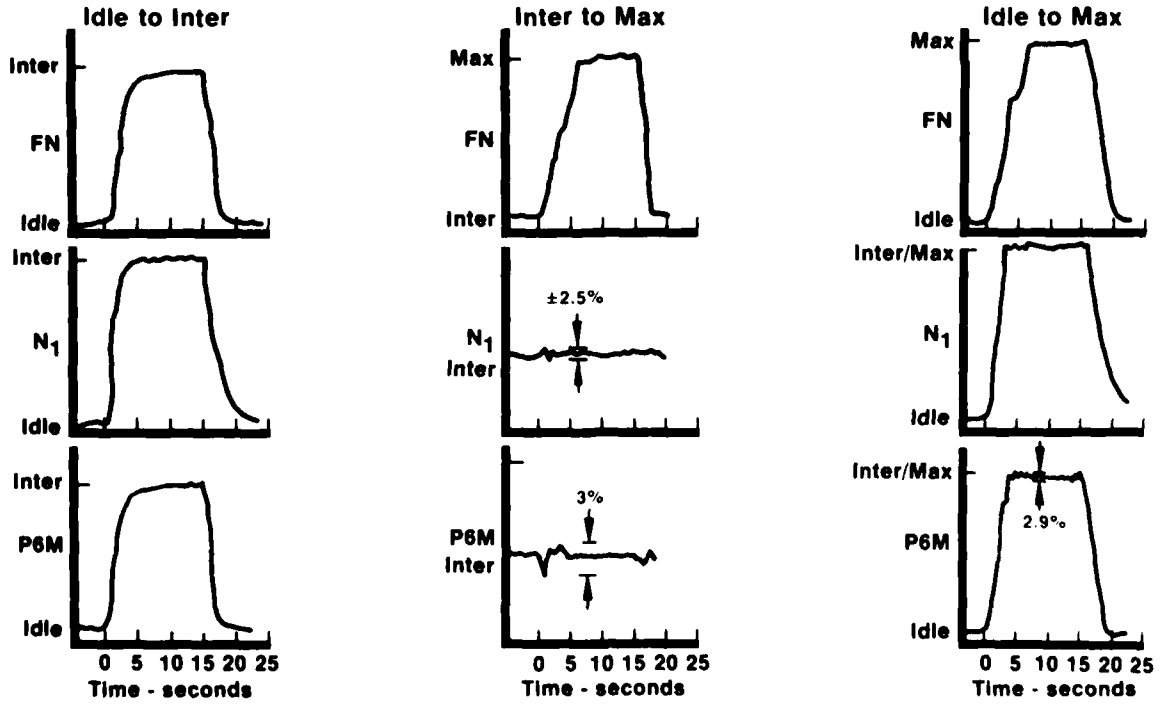


Figure 9. FADEC Provides Fast, Smooth and Stable Operation at Sea Level

Simulated altitude engine operation was demonstrated with the FADEC F401 engine at the NASA-Lewis Research facility in Cleveland, Ohio. Rigorous engine and afterburner operation was conducted with the FADEC system at nine altitude test conditions at Mach numbers between 0.3 and 1.6 and altitudes ranging from 2,100m (7,000 ft) to 15,240m (50,000 ft). The test conditions are shown on figure 10. The entire altitude test sequence was completed in six test periods, with 25 hr engine operation, due to the excellent operation of the FADEC system. The full range of steady-state and transient operation, including Bodie acceleration, was demonstrated at each altitude condition. Fault accommodation was also demonstrated during this test. Figure 11 illustrates transient data from this testing, which shows the same rapid, smooth and stable operation exhibited during sea-level operation.

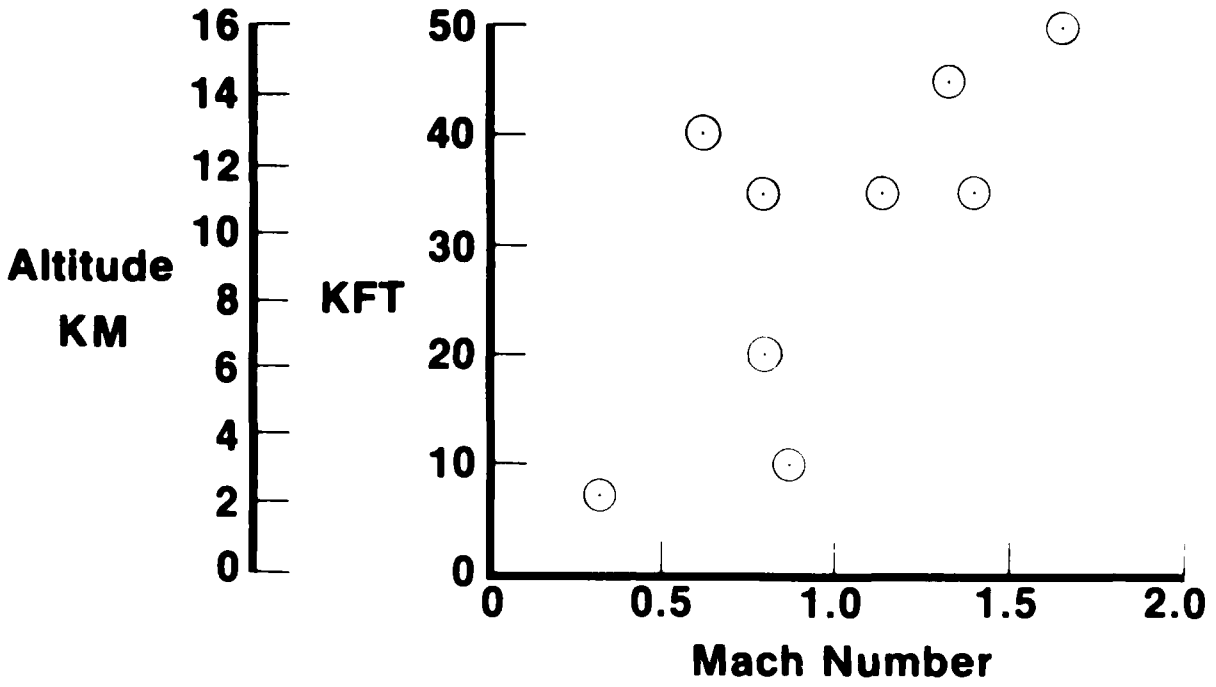


Figure 10 FADEC/F401 Engine Test Conditions

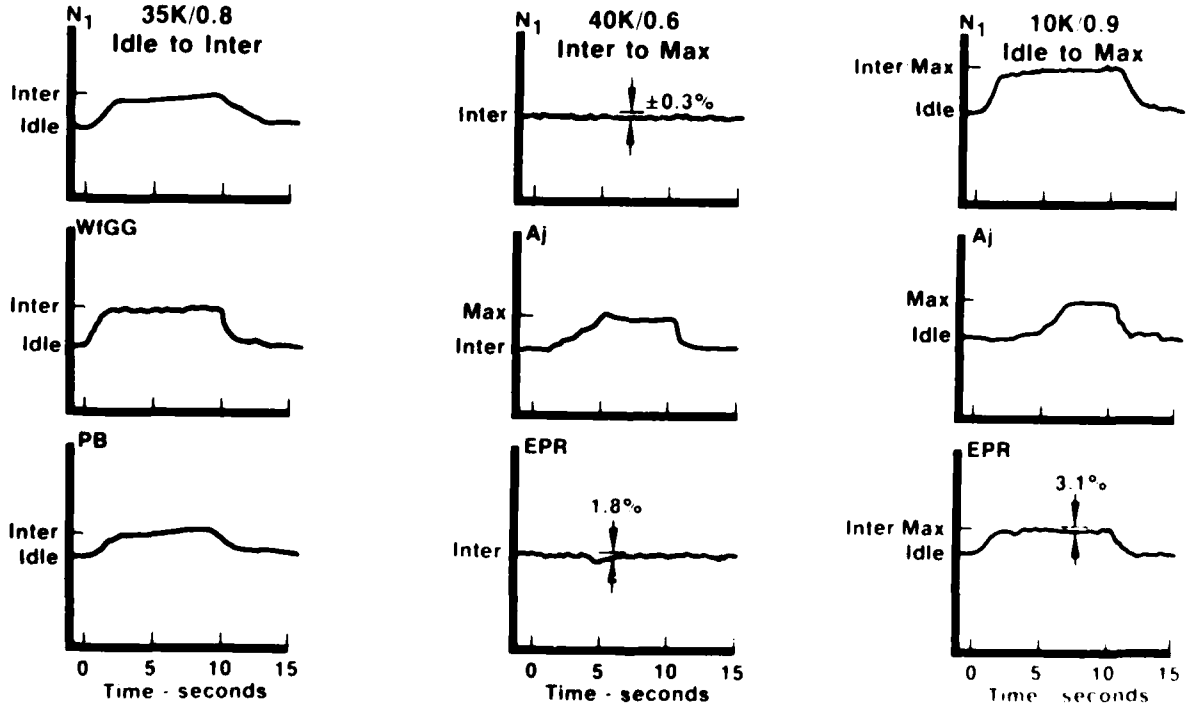


Figure 11 FADEC Demonstrated Fast, Smooth and Stable Operation at All Altitude Conditions

Post-Test Summary

At the completion of altitude testing in May 1979, a total of 68 hr of engine operation with the FADEC F401 system had been obtained without a hardware malfunction. In addition to the advanced electronic circuitry employed in the FADEC, this is the first demonstration of optic digital data communication with engine mounted equipment. Post-calibration and inspection of the FADEC units have shown no problems after having successfully completed a rigorous test program, which was very similar to the Preflight Rating Test requirements of MIL-E-5007D.

A total of over 7000 hr of control operation was completed including 1100 hr operation with engine-mounted units. The successful completion of this program has shown that the Full Authority Digital Electronic Control concept has the potential to satisfy the demands of future military engine control applications while promising substantial benefits in control system life cycle cost, reliability and weight.

Many of the technology features of the P&WA FADEC design are already being incorporated in new engine control designs for both military and commercial applications.

FADEC Flight Demonstration Planned

The success of the FADEC program to date has led to Navy planning for a demonstration in either an F-15 or F-14 research aircraft. In either case the flight program objectives would be:

1. Flight demonstration of dual redundant control and related failure accommodation logic.
2. Development and test of high-speed optic data communication from the engine control to the avionics system in accordance with the MIL-STD-1553B format.
3. Flight demonstration of an integrated inlet/engine control system.
4. Flight demonstration and environmental verification of the engine-mounted, advanced electronic control hardware.
5. Collection of diagnostic and environmental data for development of a sound data base for future electronic control design and for input to current simulated mission environmental test programs.

The FADEC/F-14 demonstration system is illustrated in figure 12. Dual FADEC computers control the TF30 engine through electro-mechanical interface units. The inlet actuator on this system, three ramps, and a variable bleed door are also controlled by FADEC. Aircraft information on Mach number and angle of attack are sent to the FADEC units from the Central Air Data Computer (CADC). Communication between these systems is through an advanced optical data bus utilizing the MIL-STD-1553B data format.

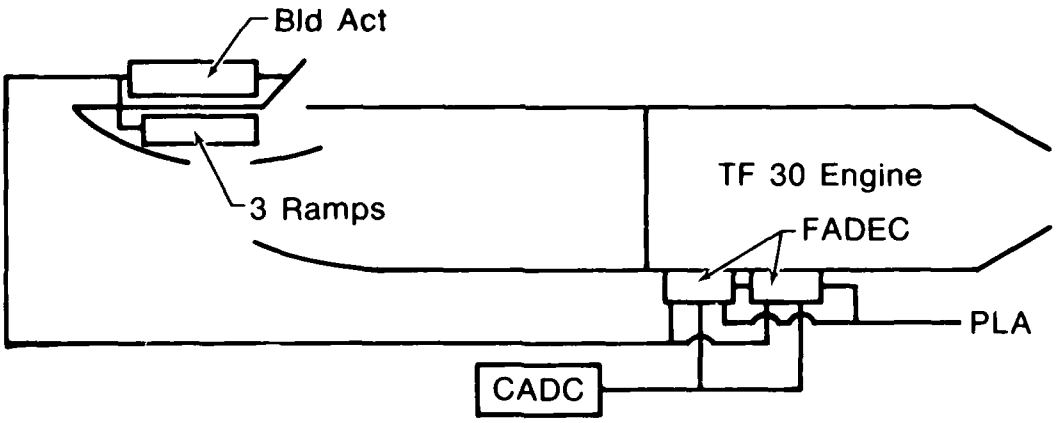


Figure 12. FADEC/F-14 System Arrangement for Flight Test Demonstration

To accomplish the stated objectives, a program is being formulated for system development, system integration and flight testing. The system development phase consists of airframe systems development, propulsion systems development, interface procurement, and modifications of the engine-mounted FADEC. Systems integration activities include bench, engine and combined tests of the system with the modified FADEC controllers, high-speed optic data links, and advanced data bus systems. This work culminates in a flight demonstration program anticipated to occur in early 1983. Once in place the FADEC system would be a key element in demonstrating advanced control systems technologies for application to the weapons systems of the 1980's.

REFERENCES

1. Barclay, B. A., "*FADEC — Digital Propulsion Control of the Future*," AIAA Paper 76-652, July 1976.
2. Barclay, B. A., T. G. Lenox, and C. J. Bosco, "*Full Authority Digital Electronic Control — Highlights of Next Generation Propulsion Control Technology*," ASME Paper 78-GT-165, April 1978.

SYMBOLS

FADEC	— Full Authority Digital Electronic Control
RAM	— Random Access Memory
PROM	— Programmable Read Only Memory
EAROM	— Electrically Alterable Read Only Memory
EMI	— Electromagnetic Interference
A4	— High-Pressure Turbine Inlet Area
A41	— Low-Pressure Turbine Inlet Area
AJD	— Duct Stream Exhaust Nozzle Area
AJE	— Core Stream Exhaust Nozzle Area
WFD	— Duct Augmentation Fuel Flow
FN	— Net Thrust
N1	— Fan Speed
PGM	— Augmentor Static Pressure
INTER	— Intermediate Power Setting
AJ	— F401 Nozzle Area
EPR	— Engine Pressure Ratio
P_b	— Burner Pressure
W_{fg}	— Fuel Flow, Gas Generator
MAP	— Maximum Power Setting
A/C	— Aircraft

SELECTIVE BIBLIOGRAPHY

This Bibliography with Abstracts has been prepared to support AGARD Lecture Series No. 113 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, Professor J.D.Powell.

80A42208 ISSUE 17 PAGE 3193 CATEGORY 37 RPT#: ASME PAPER 80-GT-78 80/03/00 6 PAGES UNCLASSIFIED DOCUMENT

UTTL: Ten years of digital computer control of combustion turbines AUTH: A/YANNONE, R. A.; B/REUTHER, J. F. SAP: MEMBERS, $1.50; NONMEMBERS, $3.00

ABA: (Author)

ABS: Design criteria are established for the digital computer control systems first produced in 1969, and currently in the tenth year of history for startup, control, and monitoring of combustion turbine units in the 25-125 MW range. The advantages of digital control and the fall-out obtained from the application of an advanced state-of-the-art control in the area of diagnostics and historic data handling are reviewed. The integration of the digital computer back-up system, composed of a solid-state fuel scheduler and logic sequencer provided for total system performance and reliability is reviewed, and the feasibility of a computer retrofit using existing software with the latest state-of-the-art microprocessor emulator is discussed.

80A40455 ISSUE 16 PAGE 3049 CATEGORY 60 80/06/00 6 PAGES UNCLASSIFIED DOCUMENT

UTTL: Microprocessor requirements for implementing modern control logic AUTH: A/FARRAR, F. A.; B/EIDENS, R. S.

ABA: (Author)

ABS: Analytical procedures for establishing microprocessor accuracy, computational capability, and memory requirements for implementing linear-quadratic-Gaussian (LQG) control logic were developed and evaluated. The developed procedures were evaluated and illustrated by application to a linearized fifth-order F100 turbofan engine model. Results were verified using a digital simulation of a continuous system/microprocessor controller.

80A38938 ISSUE 16 PAGE 2910 CATEGORY 7 RPT#: AIAA PAPER 80-1149 80/06/00 7 PAGES UNCLASSIFIED DOCUMENT

UTTL: Full authority microprocessor digital control AUTH: A/MCLYNN, H. J., JR.; B/COLLINS, J. W. F.

ABA: (Author)

ABS: This paper describes a prototype full authority microprocessor digital control that was developed for an advanced technology engine. The control architecture and self-test features are discussed. Control functions include speed governing, variable geometry scheduling, engine start sequencing, and stall and flame-out detection and recovery. The control interfaces with a digital test set which was used to monitor control and engine performance and to adjust selected control system parameters during test cell operation. The digital control and associated test set provided a significant degree of flexibility during engine test.

80A32472 ISSUE 12 PAGE 2137 CATEGORY 6 79/00/00 8 PAGES UNCLASSIFIED DOCUMENT

UTTL: The Fault Tolerant Multiprocessor engineering model /A report/ AUTH: A/SCHOLZ, C. O.

ABA: M.E.P.

ABS: It is noted that soon avionics systems will be expected to exhibit reliability presently required only of structural and mechanical control systems on aircraft. The Fault Tolerant Microprocessor (FTMP) which is designed to be the central computer of a homogeneous, wholly integrated avionics system is described, noting that it is being developed under funding of NASA's Energy Efficient Transport program. An overview is presented of the FTMP architecture concepts in order to describe the underlying principles. The engineering model design is described in detail and emphasis is given to areas where new concepts were developed during the EM design.

80A32423 ISSUE 12 PAGE 2114 CATEGORY 6 79/00/00 5 PAGES UNCLASSIFIED DOCUMENT

UTTL: Single chip custom LSI microcomputers for avionics applications AUTH: A/KANTA, M. J. A.

ABA: A.T.

ABS: The paper discusses a single chip custom LSI microcomputer with flexible architecture and a variable instruction set. The device was developed as an alternative to a full custom LSI and its associated long lead time, high development cost and difficulties in incorporating changes. The microcomputer contains all the computer elements similar to MOSTEK 3870, it is much more efficient, the hardwired logic can be included on the chip, and its costs are cheaper than standard system implementations. Software development can take place on existing systems using reconstruction, and can be fully debugged in its application system using a simulator board.

80A29499 ISSUE 11 PAGE 1944 CATEGORY 9
79/00/00 7 PAGES UNCLASSIFIED DOCUMENT

UTTL: Instrumentation for a tactical aircraft air-to-ground full-mission simulation AUTH: A/KING, B. C., JR.; B/RUSE, J. P.

ABA: (Author)

ABS: This paper describes instrumentation methods used in the Martin Marietta Aerospace Simulation and Test Laboratory (ST in conjunction with a nighttime, tactical air- ground mission simulation. This experimental ort, supporting the USAF-funded Precision A k Enhancement (PAE) feasibility demonstrati flight test program, required wide-ranging data gathering capabilities to effectively measure combined pilot-equipment performance with various candidate avionics configurations. As background information a brief review of the basic STL six degree-of-freedom visual simulation set-up is provided together with a summary of full-mission tasks performed by the pilot. Principal instrumentation hardware and software elements utilized in gathering and processing the digital, analog, video and audio test signals are described, and representative data samples are presented and discussed. Key test results derived from the data are then briefly reviewed.

79A54436 ISSUE 24 PAGE 4471 CATEGORY 6 RPT#: AIAA 79-1964 79/00/00 6 PAGES UNCLASSIFIED DOCUMENT

UTTL: Fixed byte and bit slice microcomputer survivability techniques AUTH: A/ROGERS, J.

ABA: V.T.

ABS: The paper describes techniques for achieving survivability to the transient upset phenomena, due to electromagnetic pulse (EMP) or nuclear radiation. Existing 8-bit fixed-byte micro-processors, in addition to 16-bit bit-slice architecture , are being used without considering nuclear radiation or EMP effects. A procedure is investigated whereby if a logic upset level is encountered, the main memory is first tested and then rolled back to the last check point, which contains a previous state of the computer. A central processor unit (CPU) is then reloaded with this information, and reprocessing begins. The method also considers either double or triple redundant store in a time-dependent manner, so that least one or two good state captures has been achieved. The approach is adaptable for either fixed-byte or bit-slice microcomputer systems.

79A41113 ISSUE 17 PAGE 3183 CATEGORY 7
79/00/00 10 PAGES UNCLASSIFIED DOCUMENT

UTTL: Multivariable control design principles applied to a variable cycle turbofan engine AUTH: A/DE HOFF, R. L.; B/ROCK, S. M.

ABA: (Author)

ABS: Described is a preliminary design of a controller for an advanced turbine engine concept, the variable cycle engine. This controller provides transient and steady-state regulation, fault detection and accommodation, and supervisory engine protection logic. It is designed for engine-mounted microprocessor implementation. Emphasis is on regulator design and trajectory generation. New techniques are described which have been developed to satisfy the requirements imposed by the variable cycle engine technology. The regulator is based on output feedback. It incorporates dc gain considerations and fixed structure gain matrices in its design. The trajectory generator mimics optimal trajectories for both large and small transients without the need for solving a two-point boundary value problem.

79A32440 ISSUE 13 PAGE 2332 CATEGORY 7 RPT#: ASME PAPER 79-GT-181 79/03/00 17 PAGES UNCLASSIFIED DOCUMENT

UTTL: Practical 'on-engine' microprocessor control and monitoring systems for gas turbines AUTH: A/JUSTICE, N. A. SAP: MEMBERS, $1.50, NONMEMBERS, $3.00

ABA: S.D.

ABS: With the advent of the microprocessor, the conventional hydromechanical approach is being superseded by systems mounted directly on the gas turbine engine. The paper discusses the single and twin gas turbine primary fuel control systems, along with such ancilliary controls as IGV, overspeed trip and reversionary or standby systems. Attention is focused on choice of microprocessors, input/output structures, on-engine package and its scope, reliability and integrity, failure of transducer, and a practical approach. What has presented an on-engine package before, viz. vibration and temperature, is now well within the scope of the packaging engineer, because the size, weight and heat dissipation of microprocessor components are within usable limits.

79A18682 ISSUE 6 PAGE 946 CATEGORY 7 78/00/00 16 PAGES UNCLASSIFIED DOCUMENT

UTTL: Engine fuel control systems as a determining factor on modern helicopters AUTH: A/EYASSON, H.

ABA: B.J.

ABS: Fuel control system requirements for achieving

high-reliability helicopter performance are examined with reference to twin-engine free-turbine helicopters. These requirements are considered on the basis of an analysis of all the engine operating phases. The design of hydromechanical and electronic fuel control systems is discussed, and consideration is given to the implementation of such controls through analog, digital, and programmable microprocessor techniques.

79A10759 ISSUE 1 PAGE 17 CATEGORY 7 RPT#: ASME PAPER 78-GT-99 78/04/00 9 PAGES UNCLASSIFIED DOCUMENT

UTTL: A digital fuel control system for gas turbines AUTH: A/HARRISON, P. G. SAP: MEMBERS, $1.50; NONMEMBERS, $3.00

ABA: G.R.

ABS: Control of any gas turbine requires the measurement of a number of engine parameters. All of the parameters must be measured by suitable transducers, often in an analog form, and must be linked to the fuel control system where the parameter analog values will be converted to digital values for use by the control computer. An approach involving the employment of an integrated engine fuel controller is considered. In one physical interpretation of this concept, both electronic and hydromechanical controls are integrated with the fuel control valve as a bolt-on package. The characteristics of incoming transducer signals are examined and a description of the digital processing system is presented. A microcomputer with the 8080 microprocessor is used. Attention is given to aspects of interfacing to the processor, the analog to digital converter, fuel valve position measurement, temperature measurement, spool speed measurement, the stepper motor drive system, and the power supply.

78A50626 ISSUE 23 PAGE 4139 CATEGORY 7 77/00/00 9 PAGES UNCLASSIFIED DOCUMENT

UTTL: Research and development of digital jet-engine controls AUTH: A/ENDO, M.; B/NISHIO, K.; C/SUGIYAMA, N.; D/KOSHINOWA, T.; E/WATANABE, Y.

ABA: S.C.S.

ABS: The research and development program of digital jet-engine controls at the National Aerospace Laboratory, Tokyo, Japan is discussed. Attention is given to: (1) the real-time simulation of jet engines, (2) the step responses of engine speeds, turbine inlet and outlet gas temperatures, and compressor discharge pressure, and (3) the general-purpose digital mini-computer. The prototype hardware electronic engine controls are described noting demand and sensing devices, the digital electronic computer section, and variable stator vanes control.

78A49959 ISSUE 22 PAGE 4069 CATEGORY 61 78/00/00 7 PAGES UNCLASSIFIED DOCUMENT

UTTL: Higher order languages for avionics software - A survey, summary and critique AUTH: A/RUBEY, R. J.

ABA: (Author)

ABS: This paper surveys the activities of the last ten years with regard to avionics Higher Order Languages (HOLs). It presents reasons why HOLs were late arriving in the avionics arena and why they have not been more widely used today. In particular, the published experiences with existing HOLs in avionics applications are summarized. Descriptions of important HOL evaluation criteria, such as 'efficiency' and 'programmer productivity' are presented and the reported measurements with respect to these criteria are discussed. The problems and deficiencies of past reporting with respect to these criteria are highlighted. In addition to this summary of the quantitative information regarding avionics HOL use, the need for improvements are discussed. This includes the relationship of the HOL to the total software development process, the improved software tools that can be employed, and the level of HOL documentation available.

78A23809 ISSUE 8 PAGE 1393 CATEGORY 37 RPT#: SAE PAPER 770961 77/11/00 18 PAGES UNCLASSIFIED DOCUMENT

UTTL: A microprocessor based sequencer for gas turbine engines AUTH: A/RISCH, D.

ABA: J.M.B.

ABS: A microprocessor-based governor sequencer for industrial gas turbine engines is described; the microprocessor employs a high-level programming language specifically developed for the gas turbine application. A flow chart analysis illustrates the ability of the sequencer to control lubrication oil temperature, and to deal quickly with serious engine conditions by interrupting its normal series of monitoring operations (the flow chart oriented sequencer language in effect allows the sequential processor to emulate several parallel processors). Use of the governor sequencer to provide full plant and engine sequencing authority for a gas pipeline is discussed.

78A15581 ISSUE 4 PAGE 670 CATEGORY 61 77/00/00
3 PAGES UNCLASSIFIED DOCUMENT

UTTL: Avionics applications of a software compatible processor - The 9900/990 family AUTH: A/HUGHES, J. M.

ABA: (Author)

ABS: Texas Instruments 9900/900 family of microprocessors and minicomputers is a software compatible computer family. Members of this family have been used in applications ranging from consumer products to military systems. Texas Instruments has applied the 9900/990 family as a distributed system element in a variety of avionics applications. This paper analyzes the common characteristics of these applications and the development scenario which has been followed. Particular emphasis is given to the impact of software compatibility on the development process. The requirements for and degree of software compatibility in evolutionary members of the 9900/990 family are considered.

77A39594 ISSUE 17 PAGE 2826 CATEGORY 7 RPT#: AIAA PAPER 77-899 77/07/00 11 PAGES UNCLASSIFIED DOCUMENT

UTTL: Diagnostic system requirements for helicopter propulsion systems AUTH: A/MURPHY, J. A.

ABA: C.K.C.

ABS: Requirements for a simple, low-cost diagnostic system for helicopter propulsion systems are considered. To keep costs at a relatively low level, such a system should concentrate on automatic detection and diagnosis rather than on automatic prognosis. It can be biased toward direct operator interfaces, but must reliably detect defects with a low level of 'false alarms' (less than 2%). Every effort should be made to avoid customized baselines. A hybrid configuration consisting of a basic airborne system with optional ground-based accessory equipment for extended capability seems to offer best overall cost effectiveness. System functions should include engine health, stall, coastdown, and vibration monitors, drive train vibration analysis, hangar bearing temperature check, operational limit monitor, and maintenance data recording.

77A17532 ISSUE 5 PAGE 639 CATEGORY 7 76/00/00
12 PAGES UNCLASSIFIED DOCUMENT

UTTL: The application of microprocessors to the control of small /helicopter/ gas turbines AUTH: A/SHARPE, A.

ABA: R.D.V.

ABS: Development and testing of a microprocessor-based fuel control system for a specific small gas turbine designed as power and propulsion unit for helicopters are described. Modifications of the control system for the application, hardware added to the system, dedicated software, and propulsion plant functions responding to analog and digital control inputs are discussed. Real-time program timing, and techniques for improving the dynamic range and resolution of signals for internal variables, are discussed (speed of power turbine and engine, error accumulation and lag). Test bed trials are sketched. Since gas turbine control tasks take up little more than half the microprocessor capacity, additional tasks are proposed for the available spare capacity.

80N26316 ISSUE 17 PAGE 2226 CATEGORY 7
80/02/00 21 PAGES UNCLASSIFIED DOCUMENT

UTTL: Design, evaluation and test of an electronic, multivariable control for the F100 turbofan engine AUTH: A/SKIRA, C. A.; B/DEHOFF, R. L.; C/HALL, W. E., JR. CORP: Air Force Aero Propulsion Lab., Wright-Patterson AFB, Ohio. SAP: HC A11/MF A01

ABA: R.C.T.

ABS: A digital, multivariable control design procedure for the F100 turbofan engine is described. The controller is based on locally linear synthesis techniques using linear, quadratic regulator design methods. The control structure uses an explicit model reference form with proportional and integral feedback near a nominal trajectory. Modeling issues, design procedures for the control law and the estimation of poorly measured variables are presented.

80N26315 ISSUE 17 PAGE 2226 CATEGORY 7
80/02/00 9 PAGES UNCLASSIFIED DOCUMENT

UTTL: The design concept and experimental results using the INTEL 8080/8085 microprocessor AUTH: A/JUSTICE, N. A. CORP: Hawker Siddeley Dynamics Ltd., Hatfield (England). SAP: HC A11/MF A01

ABA: R.C.T.

ABS: Prototype flight equipment was built using the 8080A and is flying with full authority in a twin engine helicopter. Isochronous load sharing on torque with simultaneous data logging output of transducer inputs and control functions was provided for monitoring purposes. This detailed background provided valuable insight to the true flexibility of a microprocessor controller and also illustrated any shortcomings that the later generation devices will need to overcome.

REPORT DOCUMENTATION PAGE

1. Recipient's Reference	2. Originator's Reference	3. Further Reference	4. Security Classification of Document
	AGARD-LS-113	ISBN 92-835-1381-9	UNCLASSIFIED

5. Originator Advisory Group for Aerospace Research and Development
North Atlantic Treaty Organization
7 rue Ancelle, 92200 Neuilly sur Seine, France

6. Title MICROCOMPUTER APPLICATIONS IN POWER AND PROPULSION SYSTEMS

7. Presented at a Lecture Series under the sponsorship of the Propulsion and Energetics Panel and the Consultant and Exchange Programme of AGARD presented on 2–3 April 1981 in London, UK, 6–7 April 1981 at Oberpfaffenhofen, Germany and 9–10 April 1981 in Genoa, Italy.

8. Author(s)/Editor(s)	9. Date
Various	March 1981
10. Author's/Editor's Address	**11. Pages**
Various	160

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

Microprocessors
Aircraft engines
Control equipment
Computer programming
Design criteria

14. Abstract

The objective of this Lecture Series is to familiarise the participants with microprocessor technology, design methods, and current applications in the aeronautical power and propulsion field. Topics include microprocessor characteristics by manufacturer, memory characteristics, software HI and LO level language tradeoffs, sensor and actuator interfacing, control logic design methods, redundancy management, and a description of several current applications to engine control.

AGARD Lecture Series No.113 Advisory Group for Aerospace Research and Development, NATO MICROCOMPUTER APPLICATIONS IN POWER AND PROPULSION SYSTEMS Published March 1981 160 pages The objective of this Lecture Series is to familiarise the participants with microprocessor technology, design methods, and current applications in the aeronautical power and propulsion field. Topics include microprocessor characteristics by manufacturer, memory characteristics, software HI and LO level language tradeoffs, sensor and actuator interfacing, control logic design P.T.O.	AGARD-LS-113 Microprocessors Aircraft engines Control equipment Computer programming Design criteria	AGARD Lecture Series No.113 Advisory Group for Aerospace Research and Development, NATO MICROCOMPUTER APPLICATIONS IN POWER AND PROPULSION SYSTEMS Published March 1981 160 pages The objective of this Lecture Series is to familiarise the participants with microprocessor technology, design methods, and current applications in the aeronautical power and propulsion field. Topics include microprocessor characteristics by manufacturer, memory characteristics, software HI and LO level language tradeoffs, sensor and actuator interfacing, control logic design P.T.O.	AGARD-LS-113 Microprocessors Aircraft engines Control equipment Computer programming Design criteria
AGARD Lecture Series No.113 Advisory Group for Aerospace Research and Development, NATO MICROCOMPUTER APPLICATIONS IN POWER AND PROPULSION SYSTEMS Published March 1981 160 pages The objective of this Lecture Series is to familiarise the participants with microprocessor technology, design methods, and current applications in the aeronautical power and propulsion field. Topics include microprocessor characteristics by manufacturer, memory characteristics, software HI and LO level language tradeoffs, sensor and actuator interfacing, control logic design P.T.O.	AGARD-LS-113 Microprocessors Aircraft engines Control equipment Computer programming Design criteria	AGARD Lecture Series No.113 Advisory Group for Aerospace Research and Development, NATO MICROCOMPUTER APPLICATIONS IN POWER AND PROPULSION SYSTEMS Published March 1981 160 pages The objective of this Lecture Series is to familiarise the participants with microprocessor technology, design methods, and current applications in the aeronautical power and propulsion field. Topics include microprocessor characteristics by manufacturer, memory characteristics, software HI and LO level language tradeoffs, sensor and actuator interfacing, control logic design P.T.O.	AGARD-LS-113 Microprocessors Aircraft engines Control equipment Computer programming Design criteria

methods, redundancy management, and a description of several current applications to engine control.

The material in this publication was assembled to support a Lecture Series under the sponsorship of the Propulsion and Energetics Panel and the Consultant and Exchange Programme of AGARD presented on 2–3 April 1981 in London, UK, 6–7 April 1981 at Oberpfaffenhofen, Germany and 9–10 April 1981 in Genoa, Italy.

ISBN 92-835-1381-9

methods, redundancy management, and a description of several current applications to engine control.

The material in this publication was assembled to support a Lecture Series under the sponsorship of the Propulsion and Energetics Panel and the Consultant and Exchange Programme of AGARD presented on 2–3 April 1981 in London, UK, 6–7 April 1981 at Oberpfaffenhofen, Germany and 9–10 April 1981 in Genoa, Italy.

ISBN 92-835-1381-9

methods, redundancy management, and a description of several current applications to engine control.

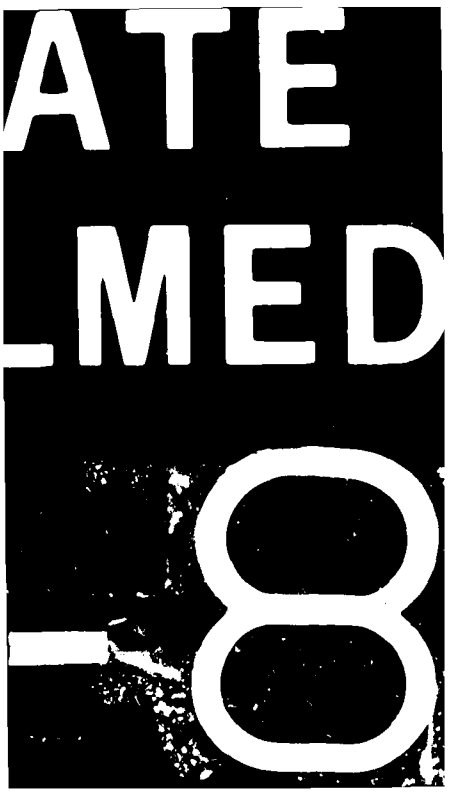
The material in this publication was assembled to support a Lecture Series under the sponsorship of the Propulsion and Energetics Panel and the Consultant and Exchange Programme of AGARD presented on 2–3 April 1981 in London, UK, 6–7 April 1981 at Oberpfaffenhofen, Germany and 9–10 April 1981 in Genoa, Italy.

ISBN 92-835-1381-9

methods, redundancy management, and a description of several current applications to engine control.

The material in this publication was assembled to support a Lecture Series under the sponsorship of the Propulsion and Energetics Panel and the Consultant and Exchange Programme of AGARD presented on 2–3 April 1981 in London, UK, 6–7 April 1981 at Oberpfaffenhofen, Germany and 9–10 April 1981 in Genoa, Italy.

ISBN 92-835-1381-9


ATE
LMED
8