A FORWARD VIEW ON RELIABLE COMPUTERS FOR FLIGHT CONTROL*

Jack Goldberg and John H. Wensley
Stanford Research Institute

SUMMARY

We examine the requirements for fault-tolerant computers for flight control of commercial aircraft and conclude that the reliability requirements far exceed those typically quoted for space missions. Examination of circuit technology and alternative computer architectures indicates that the desired reliability can be achieved with several different computer structures, though there are obvious advantages to those that are more economic, more reliable, and, very importantly, more certifiable as to fault tolerance. Progress in this field is expected to bring about better computer systems that are more rigorously designed and analyzed even though computational requirements are expected to increase significantly.

INTRODUCTION

Current NASA developments in aircraft and aviation systems design require a great increase in on-board computing. Most of the advanced aircraft designs--e.g., configuration-controlled vehicles, and certain STOL modes--require extremely reliable computations. NASA must therefore be assured that it will be possible to build computing systems having the high capacity and extreme reliability that its current advanced aircraft designs will require.

The reliability requirements far exceed those typically quoted for space missions (95% success after five years). This implies that the probability of error of spaceborne computers is designed to be on the order of 10^{-6}/hr for long missions while the acceptable figure for advanced avionic systems for the commercial environment is on the order of 10^{-9}/hr for short missions. The commercial environment also has different certification requirements, not only because of the high public demand for safety, but because the users are more diversified. Thus the hardware and software components of a computer for commercial avionics must not only satisfy the reliability criteria of computer designers, but the reliability must be convincingly demonstrated to aircraft system designers and users. It is well understood that computers of the needed power will require a large number of components, and that this number is so large ($>10^4$) and the assured reliability is so low ($>10^{-6}$

*This work was supported in part by the National Aeronautics and Space Administration, Langley, Virginia, Contract NAS1-10920.

failures/hr) that some form of built-in fault tolerance is essential. Unfortunately, the simpler forms of fault tolerance (e.g., error correcting codes and triple modular redundancy) are inadequate for computers of the required size.

Realization of this inadequacy has given rise to several research and development efforts in the design of automatically reconfigurable computers. Some examples of computers carried to a fairly detailed design level are STAR (JPL) [ref. 1], EXAM (NASA-ERC) [ref. 2], ARMMS (NASA-Marshall) [ref. 3], and MSC (SAMSO) [ref. 4]. Other recent designs, at a less-detailed level, include SIFT (NASA-Langley) [refs. 5 and 6], an unnamed computer, hereafter called HS (MIT C. S. Draper Laboratory) [refs. 7 through 9]. There has also been considerable research in techniques for designing redundant logic networks and memories, for testing arbitrary logic networks, and for modelling redundant systems. For a discussion of these topics, see reference 10.

These design and technique studies comprise a well-rounded, but relatively unproven art. They do not yet comprise a base of technological practice sufficient for the design of computer systems whose reliability can be specified with a high degree of assurance. This is a consequence of the basic fact that (1) faults and errors can occur in extremely varied ways, and (2) the fault-tolerant behavior of an automatically reconfigurable computer can be extremely complex.

Subsequent sections of this paper examine the computational and reliability requirements, the technology constraints, and estimates of the likelihood of achieving the goals.

COMPUTATIONAL REQUIREMENTS

In this section we consider the computational and reliability requirements of a representative aircraft computer system. The example we choose is that of a commercial transonic four-engine aircraft. We assume that advanced control systems will be required for such functions as flutter control and attitude control. We further assume that an advanced blind landing system would be used.

The requirements are reported in detail in reference 6, and are summarized in table 1. The most critical phase of the flight from a computational standpoint is during an instrument landing. Those applications involved in that phase are indicated with an "#". Small tasks that are not required during that phase do not influence the design of the computer system and therefore have not been estimated to the same accuracy as the more important tasks.

The column headings of table 1 are defined as follows:

Task-----------------	The name given to the application program.
Criticality Class----	1. Immediate safety-of-flight impact.
	2. Eventual safety-of-flight impact.
	3. Significant change-of-mission impact.
	4. Operational impact.
	5. Economic impact.

Table 1

COMPUTING REQUIREMENTS FOR EACH COMPUTATIONAL FUNCTION

Task	Criticality Class	Iteration Rate(s)/ Sec	Equivalent MIPS	Memory Required		Missed Iterations
				Inst.	Data	
Attitude control	1	5,20	0.023	1845	230	2-3
Flutter control	1	250	0.069	70	22	2-3
Load control	3,5#	240	0.014	45	15	2-3
Autoland, horiz.	#	20				
Autoland, vert.	1#	160	0.055	750	275	2-3
Autoland, throttle	#	33				
Autopilot	4	5	?	150	100	4-5
Elec. att. control	1#	30	0.077	790	520	?
Supervisor	4	?	?	75	15	?
Inertial	2#	1,25	0.034	2100	150	0-4
VOR/DME	4	5	0.004	250	50	4-5
DME, OMEGA	4	5	?	400	105	4-5
Air data	4	?	?	110	25	4-5
Kalman filter	4	1/5	0.001	250	65	2-3
Flight data	4	5	0.028	450	100	2-3
Airspeed, altitude	4#	8,16	0.009	360	70	2-3
Graphic display	4#	1,8	0.032	890	5360	2-3
Text display	4	10	0.019	640	8700	4-5
Collision avoidance	4#	1/3,670	0.021	550	650	1-2
Data comm, A/C	#	Non-iterative	0.006	210	400	?
Data comm, ground	4#	≤ 4	0.001	450	112	?
AIDS	5#	1/4 to 4	0.002	650	650	4-5
Inst. monit.	4#	5	0.014	800	100	2-3
Syst. monit.	1-4#	1/2	0.001	900	50	2-3
Life support	1-4#	<1/2	0.001	900	50	3-4
Engine control	1-2#	33	0.119	1300	200	1-2

Tasks to be run during blind landing, the most critical flight mode, are marked "#".
Tasks marked "?" exert a negligible load for the parameter in question.
The column headings are defined in the text.

Iteration Rates/Sec--The number of times per second that the calculation must be carried out. When two figures are quoted, they represent two calculations within the same functional task.

Equivalent MIPS------The Millions of Instructions Per Second to carry out the calculations.

Memory Required------	The number of words of memory required for instructions and data.
Missed Iterations----	The maximum number of consecutive iterations that can be missed before the application is jeopardized.

In interpreting the table and discussing its implications on computer architecture, we consider reliability, roll-back delay, main memory requirements, processor speed, processing variations within a mission, and data rates.

Reliability

We assume that the probability of not successfully carrying out the most critical computation should be less than 10^{-8} per mission. These computations, corresponding to criticality classes 1 and 2, could cause an aircraft crash if not carried out or if carried out with gross errors. With this assumed computation reliability, for a fleet of 1000 aircraft flying four daily missions, each of five hours without repair between flights within a day, about one crash due to a computer failure would occur in 100 years. For the other criticality classes, the assumed reliability is not as stringent--a typical failure probability is 10^{-4}--since the failure to carry out these less critical computations results in only a mission change or an economic loss. In a system design, it would be beneficial to so allocate redundancy that each task is carried out with the indicated reliability.

Roll-back

An important parameter of a fault-tolerant computer is the maximum time interval that the computer can be in a roll-back/reconfiguration mode in responding to a failure. During this interval some processing of certain computations may cease, and newly appearing data might be lost. The missed iterations column of table 1 indicates the number of iterations that can be ignored in a given computation without adversely affecting the aircraft. In the worst case (collision avoidance) the system must be "down" for no more than 1.5 msec. Several other critical computations--flutter control, load control, autoland--require reconfiguration times nearly as short. For these computations, it might be necessary to reload programs, which indicates that the computer might be required to be totally engaged in reconfiguration following a failure. Fortunately, the computations with large amounts of data, e.g., display, can tolerate a downtime of approximately 0.5 sec., thus allowing ample time for the possible reloading of data, interleaved with the more critical computations.

Memory Requirements

The application programs for the critical phase require approximately 20K words. This figure is a low estimate for two reasons:

- The difficulty of estimating accurately

- The need for memory space for the executive routines.

Hence we assume a total memory requirement of 24K words. Note that this is a nonredundant requirement; the demand for fault tolerance will increase this figure. For architecture relying totally on triplication, this storage requirement must be tripled to 72K. For architectures utilizing only single-byte correction (ref. 10) in memory (plus possibly a few extra bytes for double-byte detection and sparing), the figure is about one-third in excess of 24K or about 32K.

Processor Speed

For the critical phase, the application tasks require 0.386 MIPS (millions of instructions per second). Once again we must regard this figure as being low in part due to inaccuracies, but mostly due to the "wasted" CPU power in multiprogramming and the processing of executive routines. For these reasons we assume a processor load of 0.5 MIPS. An important attribute of the computations is their relative independence. That is, the sharing of functions and data among the computations does not substantially reduce the overall memory or processor requirements. Each computation requires access to the state of the aircraft, but most other data can be considered to be local. Hence it is quite simple to impose a multiprocessor discipline on the computations, with almost an arbitrary number of processors.

Under certain allocation of tasks to processors it is not necessary to do any task interruption within a processor. That is, a task can be allowed to run through completion before initiating another task. Five processors each of 0.1 MIPS would enable such an allocation. However, near the end of the useful life of the computer, say if just one or two unfailed processors remain, it is possible that a high-rate task (flutter control) might be allocated to the same processor as a low-rate but long task (graphic display). If such a joint allocation is unavoidable, then interruption of the longer task will be essential.

Processing Variations Within a Mission

All applications marked with "#" are required during an instrument landing. This represents about 60 percent of the total CPU requirement and about 50 percent of the memory requirement. Hence some graceful degradation is possible as, during the mission, tasks will be naturally deallocated as they are no longer needed as part of the flight. Hence, when a task is no longer needed, its memory area can be allocated to another task, or, a failure in a memory module is automatically handled by a memory module with a reduced requirement. However, we note that the degradation with respect to memory is not uniform, assuming that all programs and constants are retained in main memory.* For example, in mid-flight, although not all tasks are being processed, all programs must be stored reliably in the main memory. Hence the

*The issue of back-up memory in an aircraft environment is yet to be completely resolved. Rugged discs can be obtained but their cost per bit is not significantly less than that for LSI main memories.

graceful degradation with regard to main memory is not exploitable until the last minutes of the flight, and hence is of questionable utility to the architecture.

Data Rates

An important measure of computer power required is the load on the bus structure for transfer of instructions and data. Given a computing load of 0.5 MIPS, we assume that an instruction will, on average, require 24 bits.* Different instructions require varying amounts of data including the following cases:

- 0 bits for register-to-register operations
- 8 bits for byte operations, e.g., text display
- 16 bits for integer operations
- 32 bits for floating point operations.

Based on an estimate that the average data required is 16 bits, the total flow between memory and CPU is 20 Mbits/sec. In some architectures (e.g., the JPL STAR), the bus would have to be capable of maintaining this rate. In the case of the Hopkins scheme, a significant reduction would be achieved by the use of the local CACHE on the processors. An additional reduction is achieved by providing a multi-bus structure or allowing multiple ports into main memory. In the SIFT system, most of the bus load would be in individual modules, with only an estimated one percent between modules.

TECHNOLOGICAL ADVANCES

The most important future development in technology is expected to be the continued improvements in LSI. The cost of LSI circuits will continue to drop throughout the 1970s, and will result in processor and memory costs that are low enough so that extensive redundancy of units is practical from a cost viewpoint. This redundancy can be either by replication or by coding, the latter being more applicable to memories. It is expected that the cost of a computer system to carry out all computation within an aircraft will be comparable with the present cost of existing single-function avionic units (e.g., inertial navigation).

A second advantage in the use of LSI is the small size of such units, making it possible to achieve far more efficient shielding from both electric and magnetic fields, thereby reducing the probability of noise and crosstalk. It is expected that fault modes of this type (which are manifested as data-dependent transient faults) will be insignificant within the central units. However, such faults may still exist in connections to external sensors and actuators.

---

*In a 16-bit computer this implies equal number of single- and double-length instructions.

With the use of LSI most of the connections at the device and gate level take place within the semiconductor device, or chip, rather than on a board or through a connector as in the use of discrete circuits. The number of soldered and wrapped joints is estimated to be at least an order of magnitude less than that associated with, say, integrated circuits, thus there would be consequent reduction of faults in the connection system.

LSI circuits, though relatively cheap in high-volume production, have a high development cost. This implies that an efficient design would contain as small a number of different chip types as possible. This affects architectural decisions at two levels. At the unit level (memory, bus, arithmetic unit, control, etc.), there will be strong advantage in using replication of identical units rather than units designed specifically for particular functions. At the logic level, the high development cost of custom built units makes it more attractive to transfer arbitrary logic to a form of memory as in the use of microprogramming.

Replacement and maintenance strategies in a reconfigurable computer are also influenced by LSI. The large number of gates per chip, together with the tendency for a chip fault to affect many gates, implies that groups of registers on the same chip should be replaced, rather than replace small units such as registers.

The choice of LSI technologies is between the lower-speed, lower-cost MOS and the higher-speed and higher-cost bipolar technologies. The total computing power required among the elements of the several candidate architectures is such that MOS will be sufficiently fast for memories, buses and arithmetic units. In addition, the use of a multiprocessor organization permits the attainment of high computation capacity with slower processors. The higher speed of bipolar circuits may still be necessary in the control sections where the microprogram cycle time will typically be an order of magnitude faster than the instruction cycle time. Recent advances in technology have tended to bring the two types closer in both speed and cost.

We note that the choice between different LSI technologies, discussed above, was on the basis of speed and cost. The lower-cost alternative of MOS is possible because of the higher density within the chip, thereby enabling the use of fewer chips. This will have the desirable effect of increasing the inherent reliability due to the reduction in number of chips. LSI memory systems appear to be potentially more reliable than core or plated wire, because of the reduced numbers of discrete semiconductor devices and interconnections. The use of batteries is deemed to be a fully adequate assurance of non-volatility.

The MTBF for LSI circuits is estimated to be between 10^6 and 10^7 hours. The requirement to achieve a MTBF of 10^9 hours for the whole system can be shown to be achievable by several architectures.

The use of optical coupling between units can provide great protection against damage propagation through several units. The architecture must therefore be more concerned with fault propagation through erroneous data than by adverse electrical phenomena. The added cost for such protection is substantial, though not prohibitive, so careful design to achieve fault-isolation is required.

DESIGN CONSIDERATIONS FOR FAULT-TOLERANT COMPUTER ARCHITECTURES

In the preceding sections we have discussed the requirements for fault-tolerant aircraft computers, and the impact of new technology on their design. We now consider some representative computer architectures from the viewpoint of cost and reliability.

Many possible computer structures exist to satisfy the requirements and it is not our intent here to survey all existing or possible designs, but rather to look at a small number of designs in order to compare the use of different fault-tolerance techniques. We choose three designs--multichannel, SIFT, and SIFT with coding in memory.

In the multichannel design, a number of identical computers are used with all computers operating identically on the tasks to be performed. The computers are operated in a lock-step mode with all data movement being checked by voters that are connected to the buses. A typical number of channels would be three, four or five, higher numbers being unnecessary and tending to complicate the design of the voters.

In the SIFT design, a number of computers are also used but they do not operate in lock-step mode, and they do not all operate on the same tasks. Error-detection is achieved by comparison of results of calculations carried out in several computers, this comparison being by program, not by a hardware voter. An important characteristic of the design is that the buses connecting computers are constrained so that each computer cannot write into the memory of the other computers. This greatly improves fault isolation between computers. Reconfiguration is also carried out by software in a system executive that is itself replicated to assure adequate reliability.

In the third design to be considered, the processors operate as in the SIFT design, but coding is applied to protect against faults in memory.

We now consider each of the above designs. In all cases we assume a chip failure probability of 10^{-6}/hr. We use the notation that P[event] = probability of the event occurring per hour.

We distinguish between the most critical (MC) tasks where error* probabilities should be below 10^{-9}/hr and the least critical (LC) tasks where errors should be below 10^{-4}/hr. We also distinguish those tasks required for automatic

*In this analysis, we do not distinguish between erroneous outputs to actuators and null outputs. A more comprehensive analysis would need to make this distinction.

'blind' landing and other tasks. The landing phase is the most demanding in terms of computing load. We summarize in table 2 a representative set of requirements, where M is memory requirements in thousands of words and P is processor requirements in MIPS.

Table 2

COMPUTATION AND MEMORY REQUIREMENTS

	Landing	Other
Most Critical	$P = 0.29$ $M = 8.8$	$P = 0.09$ $M = 2.2$
Least Critical	$P = 0.9$ $M = 6.8$	$P = 0.05$ $M = 5.5$

We assume that words contain, on the average, 24 information bits. We further assume that a memory chip contains 4K bits, and that it requires 30 chips/MIPS to realize the CPU.

Case 1: Multichannel

We assume 10% extra memory and processor requirement to handle the multiprogramming and other executive requirements (interrupt handling, etc.). The multichannel concept requires enough memory in each channel to hold all tasks (23.2K + 10% $\approx$ 26K), and the CPU must handle the heaviest task load (0.38 + 10% $\approx$ 0.42 MIPS). Therefore for each channel we have

$$\left.\begin{array}{l}\text{26K words} = \text{156 chips} \\ \text{0.42 MIPS} \approx \text{ 13 chips}\end{array}\right\} \approx \text{170 chips}$$

Assume that the chips in the voter (sufficiently replicated for reliability) are negligible and consider the probability of error for three-, four- and five-channel configurations. The results are displayed in table 3.

Case 2: SIFT With Fault Tolerance Achieved by Uniform Replication

For this case, the strategy is to triplicate all tasks, and when faults occur to reduce the LC tasks to duplicate, then single processors, finally removing them entirely in the event that resources are drastically reduced. We assume 20% overhead for executive plus voting routines.*

The memory and processor requirements are as in table 4. The reliability results are displayed in tables 5 and 6, for a SIFT system decomposed into four and ten modules, respectively.

*This estimate (of 20%) is not critical in determining the component count, the cost or the reliability of the design.

Table 3

RELIABILITY ESTIMATES FOR MULTICHANNEL SYSTEM

3 Channel		
Total chips	= 540	
P[1 fault]	= 0.51×10^{-3}	,....voting masks error, discard faulty channel
P[2 faults]	= 0.17×10^{-6}	,....system failure
4 Channel		
Total chips	= 680	
P[1 fault]	= 0.68×10^{-3}	,....voter removes faulty channel
P[2 faults]	= 0.34×10^{-6}	,....voter masks second fault, discard faulty channel
P[3 faults]	= 1.2×10^{-10}	,....system failure
5 Channel		
Total chips	= 850	
P[1 fault]	= 0.85×10^{-3}	,....voter removes faulty channel
P[2 faults]	= 0.58×10^{-6}	,....voter removes faulty channel
P[3 faults]	= 0.3×10^{-9}	,....voter masks fault, discard faulty channel
P[4 faults]	= 1×10^{-13}	,...system failure

Table 4

SIFT PROCESSOR AND MEMORY REQUIREMENTS

	Landing	Other

Most Critical	P = 0.35
	M = 10.4
		P = 0.11
		M = 2.6

Least Critical	P = 0.11
	M = 8.2
		P = 0.06
		M = 6.6

Total memory requirement = $27.8 \approx 28K$

Maximum CPU requirement = 0.46 MIPS

Table 5
RELIABILITY ESTIMATES FOR A 4-MODULE SIFT

Each memory = (28 X 3)/4 = 21K = 126 chips Each CPU = (0.46 X 3)/4 = 0.35 ≈ 10 chips } 136 chips Total chips = 544 We denote MC during landing as MC/L etc.	
P[1 fault] = 0.54 X 10^{-3}, M = 63K, P = 1.05 During Landing: Remove LC, MC survive During Other: MC survive, all LC to SIMPLEX, future removal LC/L	
P[2 faults] = 0.22 X 10^{-6}, M = 42K, P = 0.70 During Landing: MC/L only survive in DUPLEX During Other: All MC to DUPLEX in memory, all LC to SIMPLEX, future removal of LC/L	
P[3 faults] = 0.6 X 10^{-10}, System failure	

Table 6
RELIABILITY ESTIMATES FOR A 10-MODULE SIFT

Each memory = (28 X 3)/10 = 8.4K = 51 chips Each CPU = (0.46 X 3)/10 = 0.14 ≈ 4 chips } 55 chips Total chips = 550 chips	
P[1 fault] = 0.55 X 10^{-3}, M = 75.6K, P = 1.26 During Landing: Fault masked, LC to DUPLEX During Other: Fault masked, LC/O to DUPLEX, Future LC/L to DUPLEX	
P[2 faults] = 0.27 X 10^{-6}, M = 67.2K, P = 1.12 During Landing: MC fault masked, LC failed During Other: Fault masked, Future LC/L fail	
P[3 faults] = 0.19 X 10^{-9}, M = 48.8K, P = 0.98 During Landing: MC fault masked, MC/L to DUPLEX During Other: Fault masked, Future LC/L fail	
P[4 faults] = 0.73 X 10^{-13}, M = 40.4K, P = .84 During Landing: Possibility of system failure During Other: Possibility of LC failure, future MC/L in DUPLEX	

Case 3: SIFT with Coding in Memory

The majority of chips for SIFT in Case 2 are used in the memory. We can add protection by using an error detecting/correcting code. The analysis displayed in table 7 is for a single-error-correcting, double-error-detecting code with an assumption of 25% increase in memory cost. A module failure requires failure of one chip in the CPU or two chips in the memory. Low criticality tasks are run in SIMPLEX mode.

Table 7

RELIABILITY ESTIMATES FOR FOUR- AND SIX-MODULE SIFT
WITH CODING IN MEMORY

<u>4 Module</u>

$$\left.\begin{array}{ll}\text{Memory per module} & = (13 \times 2 + 15)/4 + 25\% \approx 13\text{K} = 78 \text{ chips} \\ \text{CPU per module} & = (0.35 \times 2 + 0.11)/4 \approx 0.2 = 6 \text{ chips}\end{array}\right\} 84 \text{ chips}$$

Total chips = 332

P[CPU fault] = 0.6×10^{-5}/per module

P[single memory fault] = 0.8×10^{-4}

P[double memory fault] = 0.6×10^{-8}

P[LC task failure] = 0.6×10^{-5}

P[reconfiguration] = 0.3×10^{-3}

P[second module fail] = 0.8×10^{-7}

P[MC task fail] = 1.3×10^{-11}

<u>6 Module</u>

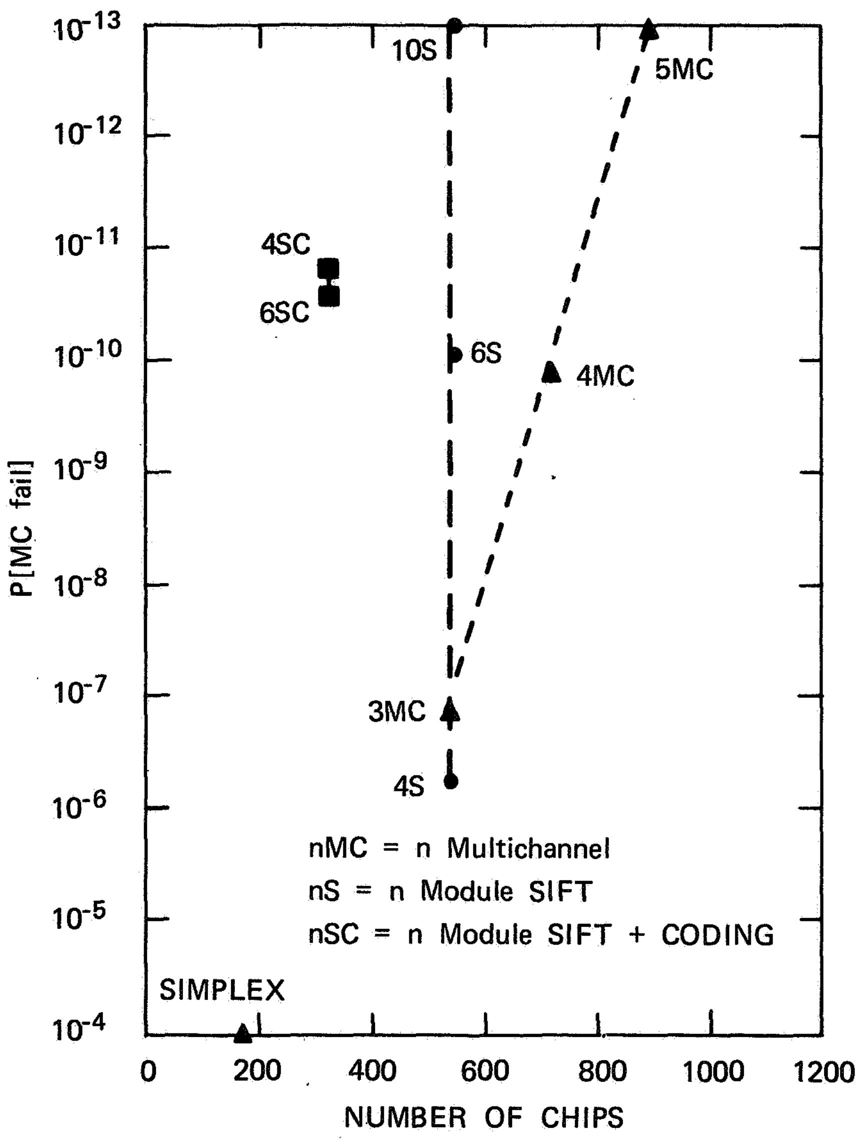
Total chips = 348

P[LC fail] = 0.4×10^{-5}

P[MC fail] = 0.2×10^{-11}

The above analysis is portrayed in figure 1 which shows the relationship between number of chips required and the probability of failure of the most critical tasks.

We conclude that all these architectures are capable of achieving the required reliability given sufficient replication. Using triplication, both of these architectures are capable of achieving a reliability of failure in the region of 10^{-6} to 10^{-7}/hr. Where reliability requirements are more stringent, as in the case for commercial aircraft, the multichannel approach can only achieve sufficient reliability at a cost significantly higher than that achievable by the SIFT architecture. In both cases, the use of coding in memory can



SA-1406-1R

FIGURE 1 PROBABILITY OF FAILURE OF MOST CRITICAL FUNCTIONS P[MC fail] AGAINST NUMBER OF CHIPS

have a significant impact on reliability and cost by handling single-error correction and double-error detection in memory in a very economic manner.

FACTORS INFLUENCING FUTURE COMPUTER ARCHITECTURES

We have discussed in the preceding sections the problems of designing fault-tolerant computers for advanced avionics requirements. We now examine the forces that will influence such computer designs in the future, particularly the period 1980-85. We see three types of influences: changes in requirements, advances in technology, and maturity in this specialized design field.

In looking at requirements we expect to see an increase in the computing load. To a large extent, this will be due to the trend towards aircraft designs that requires substantial real-time control systems for critical functions. Obvious examples are in flutter and attitude control. In addition, the operational modes of commercial aircraft will change. We would expect to see more extensive use of automatic blind-landing systems, collision-avoidance systems, and automatic or semi-automatic route-control systems. In summary, we see a greater requirement due both to more advanced aircraft designs and to a wider range of operational modes.

The most significant development of technology in the late 1970s is expected to be the wide application of large-scale integrated (LSI) technology. This will cause several effects. First, we observe that low-cost production of LSI circuits relies upon large-volume production and therefore there will be a strong incentive to use standard circuits whenever possible. This will greatly influence the type of acceptable computer architectures. Design concepts, such as discussed in the preceding section, are the types that will be favored compared with designs relying upon specialized logic to carry out the various functions associated with fault tolerance.

The second effect will be that the demonstrable inherent reliability of a circuit will be available only on large-production-volume devices. This effect will be another force towards the use of standardized circuits whenever possible. A third effect of LSI development will be the availability of back-up storage units based upon electronic (i.e., non-mechanical) technology. Such developments as bubble or charge-coupled memories potentially can be used to hold data either for later use, or for re-entry into main memory after a memory fault.

The third significant force that will influence future avionics computers stems from the increasing maturity in this field. Most designs in the past were arbitrary designs developed _in vacuo_, i.e., each design effort did not rely upon results of other efforts. There was little that could be taken from one effort to assist another. This is now changing so that the community of fault-tolerant computer designers can borrow from the results of others. Notable examples of this expanding technology base are error correcting/detecting codes, reliable switches, and reliable clocks. We still see deficiencies in the technology base, but expect that with continued research, they will disappear. The most notable present deficiencies are in the field of reliability modeling and in the area of certification. Reliability modeling as an art at present tends only to be able to analyze very idealized systems and must make very simplifying assumptions (e.g., that faults are independent and permanent). We expect that reliability modeling techniques will be developed to the point where more realistic reliability analyses can be carried out. In considering any fault-tolerant architecture, one is faced with the problem of certification of the procedures used for achieving reliability. These procedures may be implemented in either hardware or software, but whichever implementation is used there is a need to prove that the desired reliability characteristics are achieved. The present progress in the field of program proving gives us grounds to believe that formal proofs of fault-tolerant behavior will be possible.

To summarize, we see a strong trend towards the use of LSI circuitry with its attendant reduction in the number of devices, thus greatly improving the intrinsic reliability of computers. In addition, we expect advances in the theory and practice of designing, analyzing and certifying fault-tolerant computers for aircraft control applications.

We see the greatest need for improvement in techniques as:

(a) Structures for logic, systems, and software that provide both high levels of fault tolerance and ease of analysis, without the penalty of gross inefficiency or too inflexible a structure.

(b) Economical and accurate methods for verifying the correctness of system hardware and software with respect to fault tolerance and proper servicing of application programs.

However, there appears to be no fundamental reason why very reliable computers cannot be built within reasonable economic constraints. We would envisage such computers to use more than one technique to achieve adequate reliability. The main techniques would be replication, coding and reconfiguration.

CONCLUSIONS

In some new aircraft types under development there is a need for computational resources to handle very critical functions, indeed, the safety of the aircraft will be dependent on the correct functioning of the computer. In addition, the combination of high reliability and substantial computational load needed for future aircraft makes the use of simple redundant computer configurations impractical.

The present reliability art, together with continually improving technology, promises substantial improvements within the next five years for those aircraft applications with only modest computational loads. However, to meet all the larger set of computational requirements that have been suggested, at the necessary reliability levels, advances in the art of fault tolerant computer design will be required.

REFERENCES

1. Avizienis, A.: The STAR (Self-Testing and Repairing) Computer: An Investigation of the Theory and Practice of Fault-Tolerant Computer Design, IEEE Trans. Comp., C-20, pp. 1312-21 (November 1971).

2. Wang, G. Y.: System Design of a Multiprocessor Organization: Memorandum RC-T-079, NASA Electronics Research Center, Cambridge, Mass. (1969).

3. Design of a Modular Digital Computer System, Phase I Report under Contract NAS8-27926, Hughes Aircraft Company, Fullerton, California (April 1972).

4. Hecht, H. and Fry, L. A.: Fault-Tolerance in the Modular Spacecraft Computer. 6th Annual International Hawaii Conference (January 1973).

5. Wensley, J. H.: SIFT-Software Implemented Fault Tolerance, AFIPS Proc. of the Fall Joint Computer Conf., pp. 243-253 (1972).

6. Wensley, J. H., et al: Fault Tolerant Architectures for an Airborne Digital Computer, Stanford Research Institute, Report of Task I, Contract NAS1-10920 (October 1973).

7. Alonso, R. L., Hopkins, A. L., Jr., and Thaler, H.A.: Design Criteria for a Spacecraft Computer: Spaceborne Multiprocessing Seminar, pp. 23-28, NASA ERC, Boston Museum of Science (October 1966).

8. Alonso, R. L., Hopkins, A. L., Jr., and Thaler, H. A.: A Multiprocessing Structure, Digest of the First Annual IEEE Computer Conf., Chicago, Ill., pp. 56-59 (September 1967).

9. Hopkins, A. L., Jr.: A Fault-Tolerant Information Processing Concept for Space Vehicles, IEEE Trans. Computers, Vol. C-20, pp. 1394-1403 (November 1971).

10. Neumann, P. G., et al: A Study of Fault-Tolerant Computing, Stanford Research Institute, Final Report Contract N00014-72-C-0254 (July 1973).