

NOTICE WARNING CONCERNING COPYRIGHT RESTRICTIONS:

The copyright law of the United States (title 17, U.S. Code) governs the making of photocopies or other reproductions of copyrighted material. Any copying of this document without permission of its author may be prohibited by law.

C.PASTt A FAULT TOLERANT AND SELF TESTING
MICROPROCESSOR

by

M. M. Tsao, A. W. Wilson, R. C. McGarity,
C. J. Tseng* & D. P. Sienorek

DRC-18-41-82

April, 1982

C.fast: A Fault Tolerant and Self Testing Microprocessor

**Michael M. Tsao, Andrew W. WUson, Ralph C. McGarity,
Chia-Jeng Tseng, and Daniel P. Siewiorek**

Department of Electrical Engineering,
and Computer Science Department,
Carnegie -Mellon University
Pittsburgh, Pennsylvania

September, 1981

Key words:

*fault tolerance, self testing, build-in test, Fairchild F8,
single chip system and architecture, reliability, diagnostics,
experience with VLSI*

The paper presented here is an excerpt of a project report "The MPC C.fast micro computer^{1*}, available from the authors. This paper also appears in the *CMU Conference on VLSI Systems and Computations*, held on October 19-21, 1981 in Pittsburgh, Pennsylvania.

While working on this project, the authors were supported by the Defense Advanced Research Agency (DOD), ARPA Order No. 3579, monitored by the Air Force Avionics Laboratory under contract F33615-78-C-1551, and by National Science Foundation grant ENG-78-25755, and by the Carnegie -Mellon University Department of Electrical Engineering.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Agency or the US Government, or other funding agencies.

C.fast: A Fault Tolerant and Self Testing Microprocessor

Michael M. Tsao, Andrew W. Wilson, Ralph C. McGarity,
Chia-Jeng Tseng, and Daniel P. Siewiorek.

Department of Electrical Engineering,
and Computer Science Department,
Carnegie-Mellon University
Pittsburgh, Pennsylvania

During the spring of 1981, the authors were involved in a project to design a single chip fault tolerant microprocessor. The microprocessor chip is now being fabricated by the Multi Project Chip (MPC) facilities. This report presents a brief overview of the chip, examples of the reliability - testability techniques implemented, and some of the trade-off issues resolved during the design process: partitioning of control code into several PLA's, and increase of PLA size and the overall chip size due to testability - reliability constraints.

INTRODUCTION

The C.Fast¹ project attempted to accomplish four goals. The first goal was to provide the authors with experience in designing digital integrated circuits, especially microprocessors. We hoped that this experience would give us a better basis from which to build a Design Automation (DA) system using a hierarchical structured design methodology. A second goal of the project was to explore ways to connect control signals to the data path part in a simple structured way with little random routing. A third goal was to try out some low cost reliability techniques at the IC design level. Two new ideas implemented were parity checking on the the control PLA's and the concept of using the data path to act as a visibility bus for testing purposes. Other reliability techniques were also implemented for the appropriate sections of the chip. A final goal was to produce, as a by product of the design effort, a set of register transfer (RT) level building blocks to be used by our DA programs.

The Fairchild F8 microprocessor [FAIR77] was chosen as the target machine for the following reasons. 1) It represents a "typical" microprocessor architecture of the mid-70's era 2) The original F8 is an n-MOS chip, same as the MPC process. The minimum feature size used is similar to the current MPC process, where the minimum transistor gate area is 5 microns by 5 microns. 3) The complexity of the F8 is not very great, which made reimplementing the Instruction Set Processor (ISP) easier. 4) The original F8 is partitioned

¹.fast, in the PMS notation [SIEW811, stands for Computer: FAult-toferant and Self Testing

in such a way that we could implement the basic CPU chip in less than 24 pins, thus leaving some pins for our testability - reliability portion of the design. 5) As part of earlier research work, we have explored the question of implementing low cost fault tolerant features on an F8 system at the IC level.

CHIP OVERVIEW

The chip can be regarded as consisting of three interrelated sections: the control part, the data part and the reliability part (see Figure 1.) These sections were each under the control of a different designer, though naturally there was considerable consultation between them.

The control is partitioned into two groups of PLA's. Three large PLA's control the instruction execution, provide correct sequencing for the external data bus, and attempt recovery after transient errors. These PLA's broadcast encoded commands on a control bus which traverses the chip in parallel with the data bus. Small decoder PLA's (called Nano PLA's because of their resemblance to techniques used in nanocoding) produce the actual control signals for the data path elements using the broadcast microinstructions as inputs. This partitioning has produced a smaller and faster control section than would have been produced by a more conventional design methodology.

Information about the state of the data part is fed back to the control part through a status bus which is available to all PLA's. The extensive use of buses is intended to reduce random routing and is partly motivated by our Design Automation research. The use of a command bus allows easy testing since it can be made readable and writable through the visibility bus and the I/O bus. It also provides a convenient way for the Retry PLA to take over the data path control when it attempts instruction retry.

The data part is similar to the Cal Tech OM-2 data path chip [MEAD80]. Its similarities include a two phase clocking scheme, precharging of buses, use of a precharged Manchester carry chain in the ALU, and interleaved data buses. It is different from the OM-2 in that all of the storage elements are static, although the latches driving the ALU are dynamic. Also several of the elements have been reworked so that there is a uniform control, power and ground scheme. One of the buses (called the B bus) is divided into several sections to increase parallelism. Other changes include passive pullups in some places, a different spatial ordering of ALU and Shifter, use of a more specialized shifter, and wider Vdd and ground wires. Embedded within the data path part are fault tolerant devices including a parity checker, parity generator, and shadow registers. Also added are a zero detector and a status register.

The chip is also serving as a test bed for several reliability and testability techniques. Testability is enhanced by allowing access to the internal control bus through the I/O bus via the visibility register. Fault tolerances against transient errors are derived from the pervasive parity checking and built-in retry algorithms. It is also intended that two chips will be used in a duplicate and compare system with one in the standby slave mode, ready to take over if an error occurs in the master.

AREA TRADE-OFFS FOR THE CONTROL PLA'S

The control scheme makes exclusive use of PLA's. Here, the PLA's can be thought of as associative read only memories where, unlike ROMs, only those product terms (P-terms) actually needed are used. Thus a PLA based microprogram can continuously examine the present state of the machine, rather than have it retained implicitly in the

microprogram location counter. The PLA's responsible for generating the Microcode (TIMING and MAIN) have the instruction register and part of the processor state fed directly in, requiring only five bits of recycled state. There is no need for dispatch ROMs or condition code multiplexing as is often used in ROM based designs. This simplifies automatic implementations while still resulting in small size.

Our design utilizes a two level hierarchy of PLA's: a group of decoder PLA's, referred to as nano-PLA's, and microcode generation PLA's. No pipelining has been attempted in this design. The current state of the machine determines the operations performed during the next microcycle. This was done to preserve the sanity of the Microprogrammer (though he went insane anyway). The microprogram output is broadcast to all the nano-PLA's during 4_{\geq} and their outputs are guaranteed stable by the following &y

The microcode generation is broken up into two PLA's, TIMING and MAIN. The TIMING PLA keeps track of the individual instruction sequencing and controls timing of the external processor bus. It also generates the F8 ROMC codes which direct the other elements in an F8 system. The generation of ROMC codes and next state was combined into one PLA since they shared many of the same P-terms. The specific sequence of states produced by the TIMING PLA is determined by the instruction being processed. The MAIN PLA combines the present state with the contents of the instruction register to determine the next micro instruction to be executed. Few of these P-terms overlap with the others, so they were placed in their own PLA. This arrangement required less PLA area than other possibilities, as will be shown later.

When an error is detected during instruction execution, the TIMING and MAIN PLA's freeze their state and the Retry PLA takes over. It issues its own instructions onto the micro instruction bus and attempts to return the system to a known state. The instruction is then retried from the point of failure.

In Figure 2, the area requirements of several different arrangements, of PLA's are presented. These estimates do not include the effects of adding fault tolerance. As can be seen from the table, the present arrangement requires 920,000 X², of which 565,000 l² are in the MAIN and TIMING PLA's. If the MAIN and TIMING PLA's were combined into a single PLA, ten P-terms would be saved, but the total area would increase to 630,000 l². Also the overall system would be slower. Another possible arrangement would combine all PLA's into one giant PLA. This yields an even larger and slower PLA of 1,010,000 l². Thus the particular partitioning chosen appears to be a good one.

DESIGN FOR FAULT TOLERANCE

In order to achieve a more reliable microprocessor system, we decided to use fault *tolerance* techniques, rather than fault *avoidance* techniques [ELKI81]. Since we had no control over the processing and manufacturing aspect of the project, we could not do anything about the fault avoidance area of reliability improvement. Furthermore, we wanted to explore the design space for low cost fault tolerance alternatives. For this chip, we employed a four-tier approach in improving the fault tolerant properties of this microprocessor. The goals were to improve the error detection and error recovery capability of the system.

1. Conservative circuit level designs

A static latch design was used for all the flip-flop implementations, including the scratch pad memory (SPM) and all the temporary buffers. In this manner, one could hold off any one phase of the two-phase clock, thus trading off

system speed for testing observability. Since there were some questions concerning the performance of the on-chip clock input IO pad, supplied by the local MPC symbol library, two 10 pins were used for the two phases of the system clock.

2. Register Transfer (RT) level functional blocks

a. Data path part

Parity was used on all registers and the SPM's for error detection. Shadow copy registers were inserted for the accumulator (ACC) and two other key registers, allowing for instruction retry. The issue of detecting ALU errors was handled at the system level.

b. PLA implementation

Several alternative schemes for implementing a self checking PLA were investigated. The simplest one to implement involved restricting the product terms (P-terms) such that, for any input pattern, one and only one P-term is activated. The "even" and "odd" P-terms were alternately placed next to each other, such that a shorted wire will activate terms from both classes. Additionally, two extra OR array output lines were inserted, one for the "even" P-terms, one for the "odd" P-terms. In this scenario, one checks for the unique activation of these two "parity" lines. There are several more complicated schemes that could have been employed, but all required designing a great deal of additional circuitry.

3. The content of the micro code, (functionality of the control PLA)

On detection of external bus transaction error (e.g. memory read), the entire system does a bus retry. (This may not always be possible for all microprocessor systems. It is dependent upon the types of support chips used in a micro computer system.) On detection of internal errors, such as those reported by the ALU input bus parity checker, the current target machine instruction is retried. This can be done only if there is sufficient redundant information. The goal was to make the fault tolerance aspect of the chip transparent to F8 software. Under this type of fault tolerant design, one can run an existing F8 program unmodified. Figure 3 lists the additional PLA area increases due to this aspect of the fault tolerant design.

4. At the system level

Several possible scenarios were explored for this project. We implemented a master-slave, duplicate and compare system.. In this scenario, the slave microprocessor watches the output of the master in a lock step fashion. At the same time, the on board error detection mechanism is also in operation. For additional error detection capabilities, the slave microprocessor compares the output from the master with its own copy of the "proposed" output pattern. On disagreement, and also on detection of internal error, a system "retry" signal is activated. Both microprocessors perform one retry of the current target machine instruction. At the end of the retry cycle, the system can be in any one of the states shown in Figure 4. With this scheme, total error detection and recovery is not guaranteed. However, the system can recover from certain type of error conditions.

DESIGN FOR TESTABILITY

A traditional microprocessor design can usually be grouped into the data path part and the control part. The data part is more observable from the off chip I/O pins. The control part is somewhat harder to control and even harder to observe. In most microprocessor designs, the only way to determine the proper operation of the control part is by observing the output of the data part. The goal for our testable microprocessor was to design a place where we can put a controllable and observable path (the visibility bus) between the control part and the data part, and route it to the off chip I/O pins.

In the C.fast microprocessor design, the main control PLA generates control information for the "control bus", not for the data part directly. We convert this bus into the "visibility" bus during the test mode. Extra circuitry is used for observing values of the control part, as well as jamming new values onto this visibility ("control") bus. Thus, we increased the observability of the control part and the controllability of the data part.

The controllability of the control part Finite State Machine (FSM) is increased by making it easier to write information into the FSM flip-flops (FFs). During test mode, these FFs can be loaded from the off chip data bus I/O pins via the visibility register and the visibility bus. The operation is very similar to all the scan-set ideas, such as the Level Sensitive Scan Design (LSSD) technique used on some IBM machines [EICH77]. On C.fast, the FF values can be loaded during the test mode write cycles. The microprocessor runs one or more executing cycles. The chip is set back to test mode, and the values stored by the visibility register are read off. One important difference is that the data reading and loading uses the 8-bit parallel IO data bus pins. This design does not use shift-in and shift-out pins as in most scan set-like designs. Furthermore, the pins for controlling the test mode functions could be shared with the pins used for the fault tolerance operations. The extra pins do not visibly impact the total pin count.²

Additionally, portions of the FSM used for fault tolerance operations are highly observable during "normal" operations. The microprocessor can easily be fooled into an error recovery mode where the proper operation of the recovery FSM can be observed. The built-in error detectors also increase the testability of the chip.

CAD Tools

A dozen or so CAD programs were used in designing the chip. Manual layouts were done using the Xerox ICARUS interactive IC layout program, running on the Xerox ALTO's. The CIF files generated from the ICARUS layout files were sent over the Ethernet to the CMU VLSI-VAX. The most interesting path was in generating the CIF files for all the PLA's. The sequence is illustrated in Figure 5. Many programs were invoked in the prescribed sequence. However, the tasks were somewhat simplified by using system comand files. The actual bit patterns for the PLA generator were used to drive the ISPS simulation of the micro machine, providing an independent check on the correctness of the microcode. A UNIX shell program was used to merge all the leaf node CIF files, which included 6 ICARUS files, 10 PLA files, and one small hand typed CIF file, into one unified C.fast CIF file. Because of the size of the entire CIF file, design rule checking of the chip was done in separate chunks, merging only a few leaf node files at a time.

²We ran out of chip real estate in which to place the Retry PLA. However, all the associated control signals were placed. Using the visibility bus, the error recovery procedures can still be tested.

SUMMARY

Work on the C.fast chip was initiated in mid-January, 1981, and completed in mid-June, 1981. Four graduate students were actively involved in the design. Figure 6 shows a checkplot of the C.fast microprocessor, and Figure 7 identifies the major functional areas on the chip. The completed design, excluding the RETRY PLA, contains approximately 13000 transistors, of which the TIMING PLA and the MAIN PLA uses 4300 transistors. The data path part, with a 16 byte SPM section, contains 5600 transistors. The "nano-PLA's" accounts for about 3000 transistors. The chip is approximately 6100 microns by 5800 microns, somewhat big for a simple microprocessor. However, we feel that we have satisfactorily completed the four goals stated at the beginning of this project.

ACKNOWLEDGMENT³

The authors would like to thank Bill Birmingham on the layout of the shifter section, and Dr. Dennis Lunder, of Fairchild Microprocessor Product division, who donated a F387X PEP single board computer system, providing a test vehicle for our finished product. We would also like to thank our colleagues at CMLTs vLsi community. Without their many wonderful CAD programs, we could not have completed this design.

REFERENCES

- [EICH77] E.B. Eichelberger, and T.W. Williams. "A Logic Design Structure for LSI Testing". *Proc. 14th Design Automation Conf.* June 77. pp 462-468.
- [ELKI81] S.A. Elkind. "Reliability and Availability Techniques", in *"The Theory and Practice of Reliable System Design"*, by D.P. Siewiorek, and R.Swarz. Digital Press. 1982.
- [FAIR77] "F8 User's Guide." Fairchild. 1977.
- [MEAD80] C. Mead, and L. Conway. *"Introduction to VLSI Systems"*. Addison - Wesley. 1980.
- [SIEW81] D.P. Siewiorek, C.G. Bell, and A. Newell. *"Computer Structures: Principles and Examples."* McGraw-Hill. 1982.

³The paper presented here is an excerpt of a project report "The MPC C.fast micro computer", available from the authors.

While working on this project, the authors were supported by the Defense Advanced Research Agency (DOD), ARPA Order No. 3579, monitored by the Air Force Avionics Laboratory under contract F33615-78-C-1551, and by National Science Foundation grant ENG-78-25755, and by the Carnegie -Mellon University Department of Electrical Engineering.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Agency or the US Government, or other funding agencies.

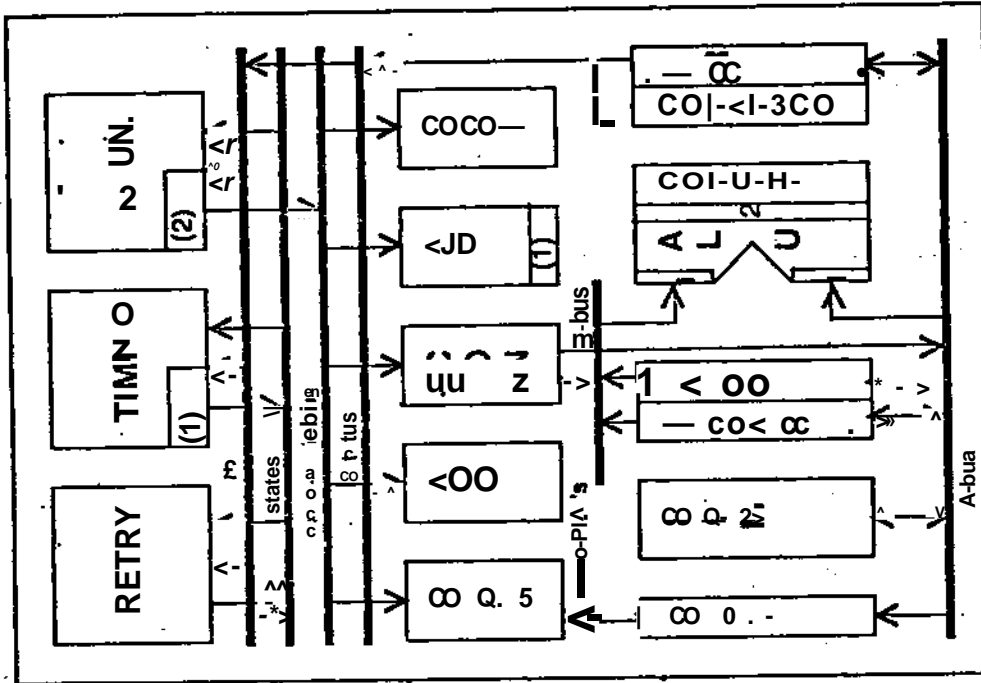


Figure 1a: RT level block diagram of the C.fast microprocessor

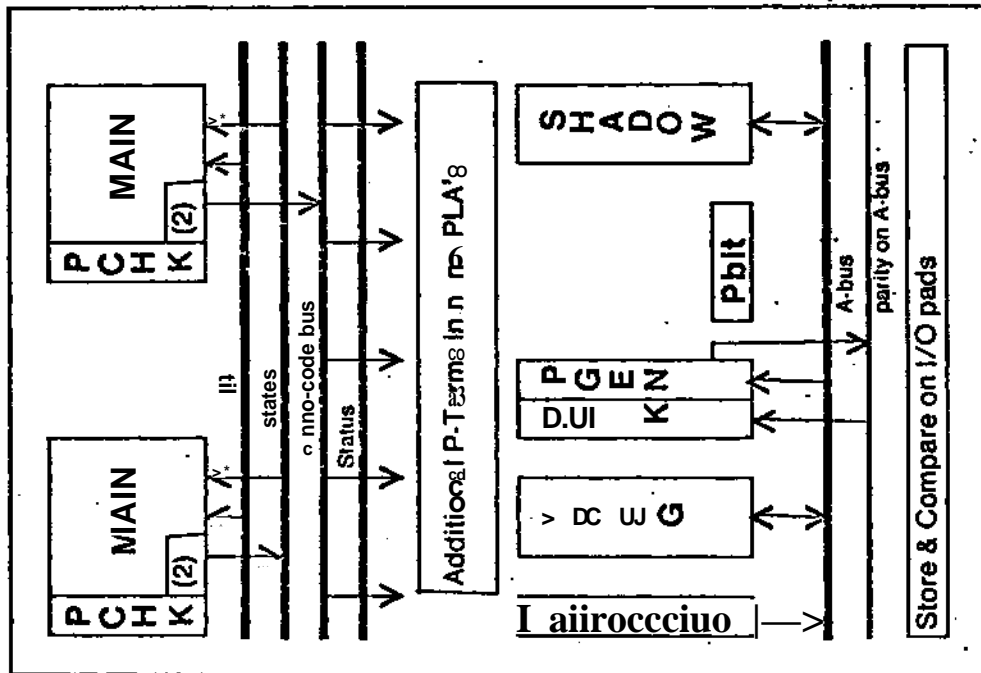


Figure 1 b: Block diagram of the testability-reliability portion of the C.fast

PLA Areas & Timing Delays

Configuration	Inputs	Outputs	P-Terms	Area(KA ²)	Delay (T)
TIMING PLA	14	11	80	251	700
MAIN PLA	14	8	110	314	970
Total PLA	14	19	180	633	2070
SPM PLA	12	33	33	163	140
Hypothetical all-in-one PLA	15	48	181	1010	2150

Comparison of Three Possible Partitions

Present Scheme	920	1010
Combined TIMING & MAIN	990	2210
All-in-one PLA	1200	2280

Figure 2: Table of PLA area trade-offs

PLA'S	Inputs	Outputs	P-Terms	Area Increases (%) over original
TIMING	16	16	116	35 <i>extra states</i>
MAIN	14	U	114	9 <i>extra nano codes</i>
RETRY	12	22	61	... <i>F8 instruction retry</i>
SPM	12	34	35	5
ACC	8	17	23	91 <i>shadow registers</i>
CON	12	11	18	9
ALU	9	13	12	0
SSI	11	14	20	41 <i>shadow status bits</i>
VRE	4	7	8	... <i>visibility registers</i>

Figure 3: Table of PLA area increases due to testability-reliability requirements

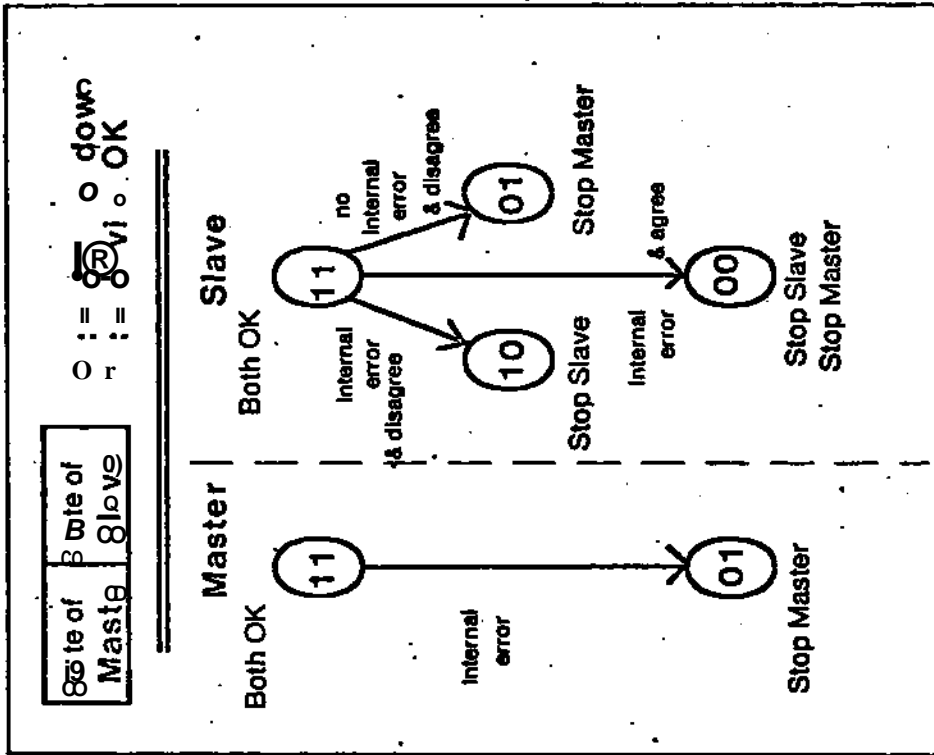


Figure 4: Markov diagram showing the states of a duplicate-match C.fast system

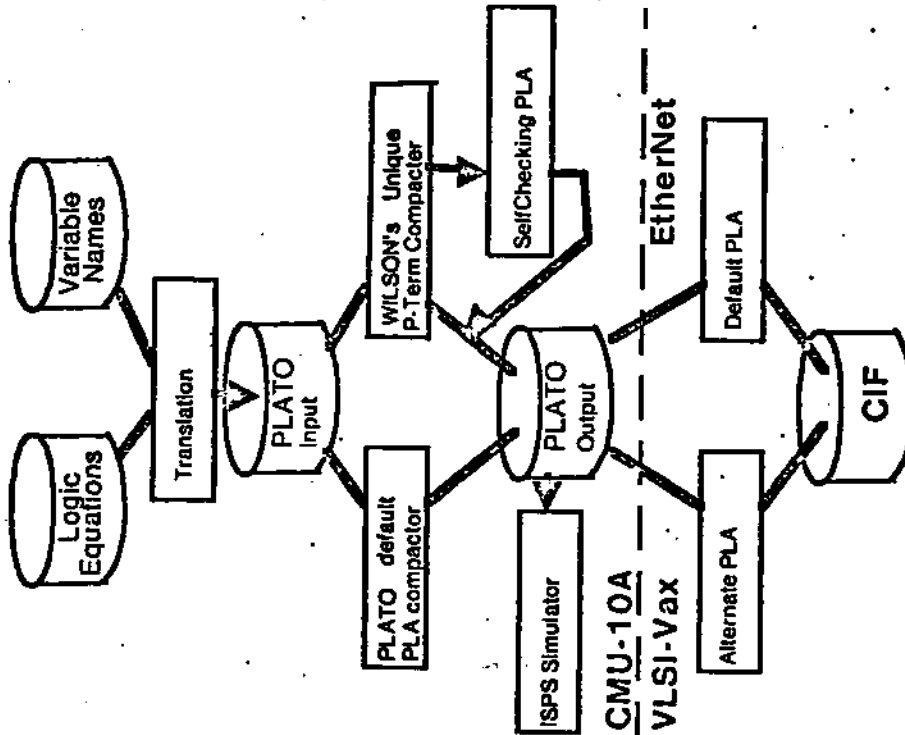


Figure 5: Diagram tracing the design process through various CAD programs

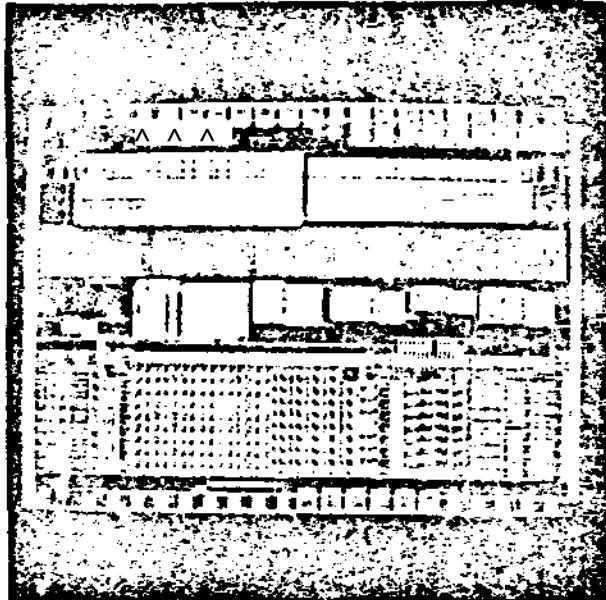


Figure 6: Video monitor display of the CIF check plot for the C.fast chip

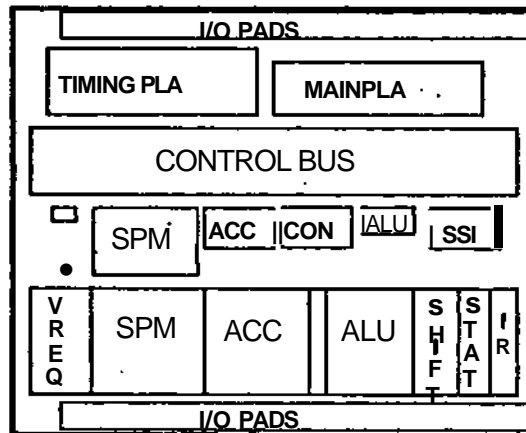


Figure 7: Major functional areas of the C.fast chip