SR-QUAL-66-25
November 25, 1967
30, 1966

GEORGE C. MARSHALL SPACE FLIGHT CENTER
HUNTSVILLE, ALABAMA

THE DEVELOPMENT OF A CHECKOUT LANGUAGE: ATOLL

By

R. C. Balentine

QUALITY AND RELIABILITY ASSURANCE LABORATORY

National Aeronautics and Space Administration

NASA

MSFC - Form 454 (Revised September 1961)

SR-QUAL-66-25
November 30, 1966

THE DEVELOPMENT OF A CHECKOUT LANGUAGE: ATOLL

By
R. C. Balentine

QUALITY AND RELIABILITY ASSURANCE LABORATORY

November 30, 1966 SR-QUAL-66-25

THE DEVELOPMENT OF A CHECKOUT LANGUAGE: ATOLL

By

R. C. Balentine

ABSTRACT

ATOLL was developed to fulfill the requirements for a common computer language that could be used by the test engineers for launch and factory checkout. "ATOLL" is the abbreviated name for Acceptance, Test, Or Launch Language.

COMPUTER SYSTEMS SECTION
VEHICLE SYSTEMS INTEGRATION BRANCH
VEHICLE SYSTEMS CHECKOUT DIVISION

TABLE OF CONTENTS

November 30, 1966 SR-QUAL-66-25

THE DEVELOPMENT OF A CHECKOUT LANGUAGE: ATOLL

SUMMARY

The implementation of a computerized automatic system was begun at MSFC in 1961. This system became operational in late 1962 and it or its successors have been used successfully for the automatic checkout of systems on stages of the Saturn I, Saturn IB, and Saturn V vehicles.

ATOLL (Acceptance, Test, Or Launch Language) was developed for use with the automatic checkout program to provide the test engineers with a common computer language as a working tool with checkout operations.

SECTION I. INTRODUCTION

A. BACKGROUND

In 1961, Marshall Space Flight Center began the task of converting the then existing systems test hardware from a manually controlled system to one controlled by a general purpose digital computer. While a few process control systems using general purpose computers had been developed in this country, implementation of a computerized automatic test control system of any kind was nonexistent. Techniques and hardware had to be developed from scratch. A system was required that operated in real time. Real time is defined as the actual time during which a physical process transpires. For automatic test control, this means the operating time of the Ground Support Equipment (GSE) and stage under test.

Perhaps the greatest obstacle to automatic test control was the deeply ingrained ideas and techniques of manual testing. Many clung to old ideas with the cry, "If it works, never change it." However, based on an assumption that a general purpose computer can function as a test controller, a hardware and software system was developed and delivered in late 1962. This early system was successfully used to automatically control several systems tests on the S-I stage of the Saturn I vehicle.

B. EARLY SYSTEMS SOFTWARE TECHNIQUES AND LANGUAGES

This early system was delivered with inadequate software to permit the sophistication of a smooth continuous flowing operation. Initially, two techniques were developed and implemented for systems checkout. They were repetitive data based systems and sequential discrete operational systems. Both are still in use today for an optimum solution to various types of checkout requirements. However, automatic test control is still not the smooth continuous flowing operation desired.

1. Repetitive Data Based Systems. In the data based system, data blocks or tables containing all required parameters are prepared prior to automatic testing by taking basic data from punched cards and arranging this data in a specified format on magnetic tape.

Small subroutines are written to utilize the data. These subroutines, one for each type of operation, are written separately to keep the overall system modular. The constants, conversion values, test values, tolerances, range, and switch selections form the data input. Thus, by indexing (indirect addressing) a relatively short read and compare routine, the entire test is completed.

2. Sequential Discrete Operation Systems. When attempts to block data input with the data based systems proved to be impractical, a sequential discrete operations system was developed. Since steps are normally not repeated, a different approach was indicated.

a. PTP Form. A tabular format called Procedure Translation Preparation (PTP Form) was developed for the early systems stage electrical networks tests. The "language" was a pseudo symbolic set with single letter designations (e.g. K for relay, X for execute, D for discrete, R for relay, etc.) A rigid and inflexible system evolved. The jargon was not a genuine outgrowth of previous test terminology. Implementation allowed only limited online modifications, and longer tests were artificially blocked to permit offline completion and simplified correction. The language did not meet the total need of the test engineers.

PTP was implemented and used for electrical networks systems checkout of the S-I stage of the Saturn I vehicle.

b. QUEST. Because of noted problems in the PTP format and implementation, the Quality and Reliability Assurance Laboratory (then known as Quality Assurance Division) in early 1963, began the specifications of a language and implementation that would meet the test control requirements of electrical systems test (stage electrical networks). The Quality Electrical Systems Test (QUEST) language specifications were completed in June 1963, to provide the following:

(1) A language for use in automatic testing of stage electrical networks that would be independent of the checkout equipment.

(2) Utilization of the knowledge of the vehicle test engineer for automatic systems test programming.

(3) A single document to serve as a test procedure, test program, and checklist.

The QUEST specifications called for an extensive amount of online (test time) capability by the test engineer to modify test processors and the test sequence (the chief fault of PTP). Fault isolation was limited to a recognition of a fault and identification of the sequence of program operations in which the error occurred. Implementation was to make extensive use of the cathode ray tube display.

Because of efforts to achieve a centerwide test language, the QUEST language was never implemented. However, the development and implementation of the centerwide test language has allowed projection that indicates QUEST would have met its objective quite well.

c. HYLA. Concurrent with Quality and Reliability Assurance Laboratory's efforts with QUEST, the Astrionics Laboratory began the development of the Hybrid Language Assembler (HYLA).

HYLA was a free format assembly language (up to 80 characters and spaces) with special operators (MACRO's) for test functions. The language was oriented toward countdown-clock sequencing with considerable emphasis on DDAS and analog readings. The intent of HYLA was to provide a programmer tool to permit faster development of prelaunch and flight routine

An implementation of HYLA was begun, but abandoned because of efforts to achieve a centerwide test language.

d. STOL. The Douglas Aircraft Company with the aid of Mesa Scientific Corporation (now Planning Research Corp.) developed the Saturn Test Oriented Language (STOL) for use on the Saturn S-IVB stage.

STOL is still in use on the S-IVB stage checkout. Basically, STOL is a quasi-test oriented language. As is the case with HYLA, the language is a free format language (up to 72 characters and spaces). STOL permits the user to initiate discrete signals with four different commands (all function identically). The language is best suited for electrical networks testing. The goals of STOL are similar to those of QUEST in that the test engineer with help can develop his test procedures in STOL. In fact, the developers of the STOL procedures have become STOL programmers.

SECTION II. ATOLL

A. GENERAL

As previously mentioned, the languages QUEST and HYLA were under development in the summer of 1963. Each language was applicable to a limited number of test situations; therefore, a committee (Standard Computer Language Conference) was formed by MSFC Automation Subboard No. 4 to develop a single centerwide test and launch oriented language. The stated goal was: "The common language for launch and factory checkout shall be such that the test engineer can use the language to do his job. The language should be such that the test engineers do not have to know the computer, programming or programming techniques." The language was called the Acceptance, Test, Or Launch Language (ATOLL) suggested by Mr. Richard K. Jenke of Kennedy Space Center and Chairman of the committee.

B. REQUIREMENTS

The committee set forth the basic requirements to be used for development of the test control language. They were:

(1) To provide a language for use in automatic checkout and launch testing of the stage or vehicle completely independent of local consideration (i.e., checkout or launch equipment and location).

(2) To utilize the knowledge of the launch vehicle test engineer for automatic checkout and launch testing.

(3) To establish criteria for the development of a single document to satisfy the following needs.

(a) To serve as a detailed test procedure when automatic checkout and launch testing are conducted.

(b) To serve as a test program for input into the automatic checkout and launch test system.

(c) To serve as a test review and evaluation document for plant representatives and project staffs.

(d) To serve as a checklist for government inspection teams when verifying contractor performance of automatic tests.

C. CRITERIA

The following criteria were established for development of ATOLL:

(1) It should be close enough to test nomenclature to be easily taught and used.

(2) It should be capable of expressing any single test function.

(3) It should be flexible enough to permit a test engineer to easily group a series of simple tests into a comprehensive test procedure expressing more complex test functions.

(4) The format should be directly translatable into the necessary computer operating instructions required to perform the indicated function or test.

(5) It should conform to the general philosophy of automatic checkout and launch testing.

D. BASIC LANGUAGE

To satisfy the stated requirements and criteria, a tabular format for ATOLL was established. Field assignments and addressing schemes were developed. The various fields were step, operator (for the command on operation), condition (to be set, tested, or evaluated),

value, upper limit (or tolerance), lower limit (or tolerance), engineering units, time, and a variable (or address field). Figure 1 is an example of the print format used in ATOLL. Remarks were not to be processed.

A total of fifteen operators were established in the basic language set. Approximately 45 operators have been added to this number, expanding the total list to approximately 60 operators. Of these, only 14 of the original operators and some five others (added for clarity) form the basic language. The additional operators (41) have tended to make ATOLL less of an engineering and more of a programmer's language. This basic group of operators (19), being those normally used by test personnel, are discussed in the following paragraphs.

1. The ANALOG STEP (ALOG) Operator. This operator will cause the application of an analog step value to the addressed device. Recording or storaging cannot be initiated by this operator because a no-go condition cannot be detected.

If a time in milliseconds is stated, the step will be applied for the stated period and then removed (i.e., returned to zero). If no time is stated, the step will be applied for an indefinite period of time (i.e., until another ALOG instruction with the same address is executed).

2. The APPLY (APLY) Operator. This operator will cause the application of an analog signal of selectable amplitude and frequency to the addressed device. Recording or storaging cannot be initiated by this operator because a no-go condition cannot be detected.

3. The BEGIN (BEGN) Operator. This operator will be used to attach a name or label to a subtest procedure only. The name may be any combination of alphanumeric characters. The step and substep fields for this operator will be specified as "ZEROS".

4. The DELAY (DELY) Operator. This operator will cause a delay in the execution of the test statement sequence until the status of the addressed device meets the specified gate condition. The gate condition may or may not have time release as specified in the time field. If no condition or analog value is listed in the fixed field, the operator becomes an unconditional time delay for the time specified in the time field.

ATOLL TEST PROCEDURE FORM

STEP NUMBER		OPERATOR	OPERANDS							REMARKS
			FIXED						VARIABLE	
STEP	SS		CON	VALUE	LOWER LIMIT	UPPER LIMIT	UNITS	TIME		
		NAME							ATOLL SAMPLE TEST	AN EXAMPLE OF AN ATOLL I SEQUENCE
		BEGN							FIRST SEGMENT	FIRST STATEMENT OF THIS SUBPROCEDURE..
0001	10	SCAN							RMT03,RDY	SCAN TO VERIFY THAT ALL DISCRETES ARE IN THE REQUIRED STATE (0) AND THAT INCORRECT STATES ARE OUTPUT ON MAGNETIC TAPE AND CRT DISPLAY.
0002	10	DISO	1						D121	TURN GENERATER NO. 1 ON.
	20	DISI	1						D61,D81	GENERATOR NO. 1 VOLTAGE AND FREQUENCE OK.
	30	DELY						180		DELAY FOR 180 MILLISECONDS.
	40	SCAN							RMT03,RDY	SCAN + RECORD NOGO'S FOR GENERATOR NO. 1.
0003	10	DISO	1						D122	TURN GENERATOR NO. 2 ON.
	20	DISI	1						D62,D82	GENERATOR NO. 2 VOLTAGE AND FREQUENCE OK.
	30	SCAN						180	RMT03,RDY	DELAY 180 MILLISECONDS, THEN SCAN + RECOND
0004	10	SETT							T0001	SET GENERAL TIME CELL NO. 1 TO THE PRESENT
	20	READ		+4.950	-0.050	+0.050	VDC		A303,RMT03,RDY	READ ANALOGUE MEASURMENT 303
0005	10	DISI	0						D337	MULTIPLEXER 3 IS NOT BUSY
	20	SCAN						350	RMT03,RDY	SCAN + RECORD NOGO'S
0006	10	GATE						5000	Toool	GATE ON TIME CELL NO 1
0007	10	DISO	0						D121	TURN GENERATOR NO. 1 OFF
	20	DISI	0						D61,D81	VOLTAGE + FREQUENCE ARE ZERO
	30	SCAN							RMT03,RDY	SCAN + RECORD NOGO'S
		.							.	.
		.							.	.
		.							.	.
0100	10	RETN								END OF THE SUBPROCEDURE
		.							.	.
		.							.	.
		.							.	.
		END								END OF THE ATOLL SAMPLE TEST

VEHICLE STAGE DOES NOT APPLY TEST ATOLL SAMPLE TEST PAGE 1 OF 1

MSFC - Form 127 (Rev. February 1964)

Figure 1. ATOLL Test Procedure Form (Sample)

5. The DISCRETE INPUT (DISI) Operator. This operator will cause a change in the reference profile of the addressed input devices, and in addition, can cause the addressed devices to be sensed and their status checked against the updated reference profile.

6. The DISCRETE OUTPUT (DISO) Operator. This operator will cause a change in the condition of the addressed switching devices. The operator may or may not be time dependent, where time dependency is the duration of the specified condition only. If no time is specified, the condition is absolute. Recording or storaging cannot be initiated by this operator because a no-go condition cannot be detected.

7. The DISPLAY (DPLY) Operator. This operator will cause the operation of the associated display system. As the various contractors and MSFC laboratories have highly dissimilar systems with varying capabilities, the use of the fields shall be as defined by the using organization except that the mnemonic DPLY shall be entered in the operator field.

8. The END (END) Operator. This operator will cause an end of test in a main test procedure only.

9. The EXECUTE (EXEC) Operator. This operator will initiate an unconditional branch to the named operational sequence (i.e., subtest procedure or machine language routine). If the named operational sequence has been previously called, execution commences immediately. Otherwise, the EXECUTE operator will call the named operational sequence and then initiate execution. The return from the named operational sequence will be to the test statement immediately following the execute statement.

10. The TIME GATE (GATE) Operator. This operator will cause a delay in the execution of the test statement sequence until the realtime is equal to or greater than the sum of the addressed time cell and the synchronous time value entered in the time field. If the addressed time cell is a counting device, such as the countdown clock, the date time value will be entered in the value field and the time field will be blank.

11. The NAME (NAME) Operator. This operator will be used to attach a name, or label, to a main test procedure written only in ATOLL test statements. This operator will be the first operator used in any test procedure.

12. The ANALOG RAMP (RAMP) Operator. This operator will cause the application of an analog ramp function to the addressed device. The ramp will be linear with the slope defined by the test statement. Recording or storaging cannot be initated by this operator because a no-go condition cannot be detected.

13. The READ (READ) Operator. This operator will initiate the measurement and tolerance check if required of the addressed DDAS or hardwire input channels.

14. The RETURN (RETN) Operator. This operator will cause the termination of a subtest procedure only and a return to the next higher level of testing.

15. The OUTPUT and RECORD (RECD) Operator. This operator will cause the output or recording of the addressed table or storage group of information which is prestored under the assigned address. Each table or storage group will be output in a predetermined format which is designed to the ATOLL system.

16. The SCAN (SCAN) Operator. This operator will cause a scan or evaluation of all discrete inputs and a check against the reference profile of the discrete conditions. This operator may or may not be time dependent, where time dependency is the time duration between the start of the operator execution and the time the scan is commenced. Because this operator will detect a no-go condition, an online recording location may be supplied.

17. The SEMIAUTOMATIC CONTROL TRANSFER (SEMI) Operator. This operator transfers control of the language system to a semiautomatic control mode, sometimes referred to as the SAC mode, under the direction of the test conductor. The return to the automatic test statement execution mode will be initiated by the test conductor. The implementation of the SEMI operator is a significant portion of the execution program in the test control computer. The use of SEMI using the cathode ray tube display has solved the main fault of PTP, our first "test language". While the implementation has been a function of the stage under test, the SAC mode has permitted individual testing of analog and discrete lines, rescanning of discrete lines, recycling through the last step or block, restarting the test, aborting the test, and similar functions.

18. The SET TIME (SETT) Operator. This operator will set the addressed time cell. If the addressed time cell is a counting device, such as the countdown clock, the value to be set into the cell will be placed in the value field. If the cell is a noncounting device, the system time reference will be stored in the addressed time cell. If required for a systems time reference to the previous step, a "1" shall be put in the condition field, and to reference the next, "0" shall be put in the condition field.

19. The TEST (TEST) Operator. This operator will cause a conditional branch based on the state of the bilevel or analog parameter addressed in the variable field. The state may be stored by specifying the storage address in the variable field. The discrete or analog condition that must be met will be placed in the condition field. The address from which the executive takes the next instruction, if the condition is satisfied, must be listed in the variable field.

Conditions tested with the TEST operator include "on", "off", "less than", "greater than", "equal to", "not equal to", "within tolerances", "not within tolerances", etc.

NOTE: For additional information on these and other operators, see MSFC Drawing 85M06078 "Launch and Checkout Computer Program Configuration and Control Plan", or R-QUAL-PC 65.721 "Survey of ATOLL Operators".

SECTION III. SYSTEMS IMPLEMENTATION

A. GENERAL

To date there have been six implementations of ATOLL on Saturn stages and breadboard. Two research and development facilities also have working systems although both are presently being updated. These implementations are:

(1) Marshall Space Flight Center (R-QUAL-PS) for Saturn I (100 series) Instrument Units.

(2) Marshall Space Flight Center (R-ASTR-E) for the Saturn I Systems Development Breadboard Facility.

(3) International Business Machines for the Saturn IB (200 series), Saturn V (500 series) Instrument Units, and KSC operating system.

(4) The Boeing Company for the S-IC stage test and checkout.

(5) North American Aviation for the S-II stage test and checkout.

(6) The Chrysler Corporation for the S-IB stage electrical networks checkout.

(7) Marshall Space Flight Center (R-QUAL-PS) for the TRAINER computer and simulator (a training aid used by the NASA School of Reliability and Quality Assurance).

(8) Marshall Space Flight Center (R-QUAL-PS) for the Advanced Test and Checkout Method Evaluation Development (ATCOMED) facility.

The Douglas Aircraft Company began an implementation for the S-IVB stage; however, because of schedule consideration ATOLL was abandoned for the inhouse system STOL.

The ATOLL concept has been introduced into the launch operating system by IBM. In that system ATOLL is one of many aids available to the test personnel.

B. TECHNIQUES

All implementations of ATOLL, to date, have been a two pass system that translates ATOLL to an intermediate language used by the online executive system of the test control computer. ATOLL statements are translated into small data blocks in the assembly language of the computer to be used. This assembly source language is then assembled by a second pass into a machine language data block containing a call sequence for the appropriate subroutine. These data blocks are then put on magnetic tape and executed by the test control computer when requested by the test conductor. Figure 2 depicts this flow.

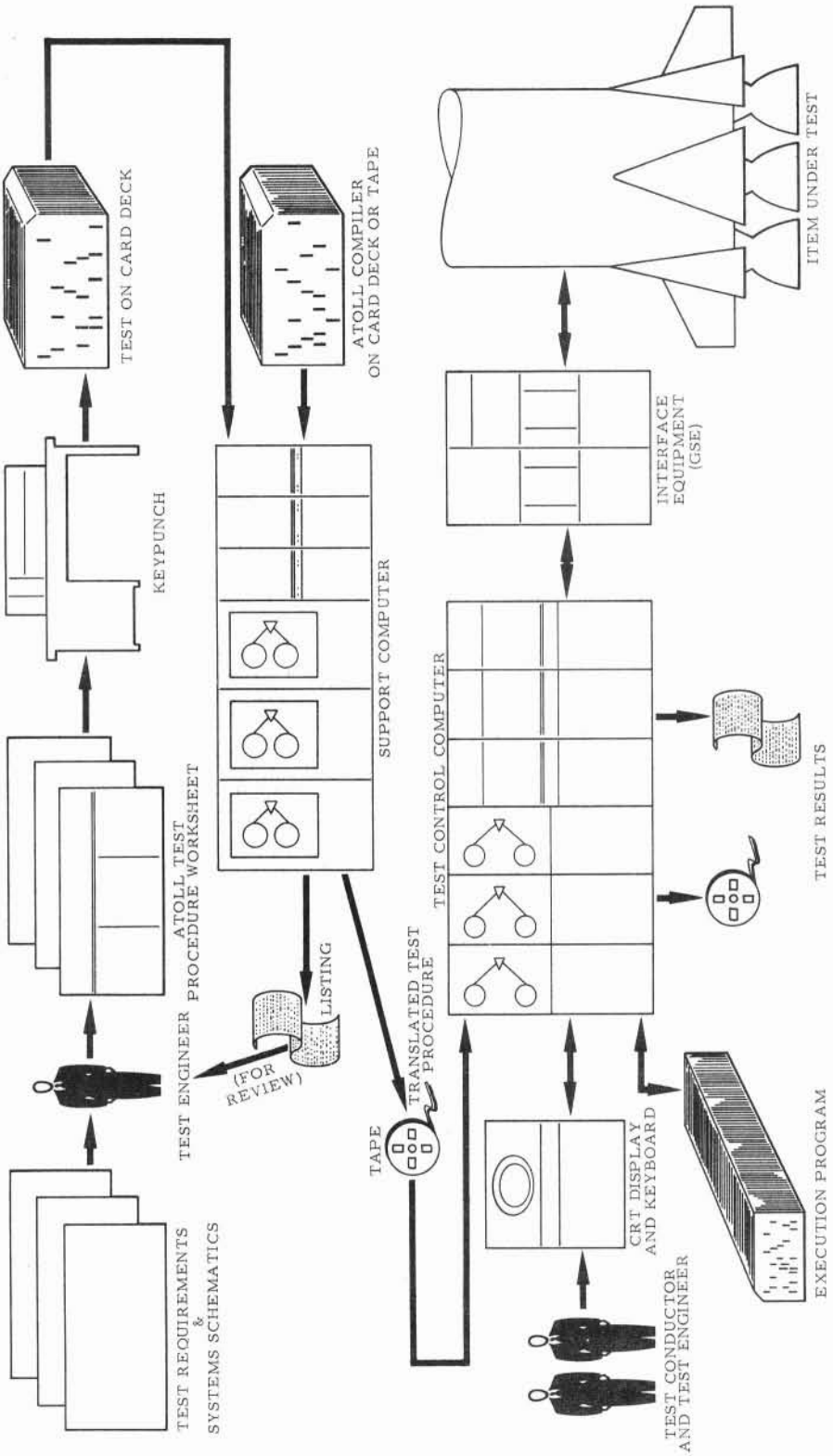

Figure 2. ATOLL Flow Diagram

To assure adequate control of the test procedure during actual testing, a considerable amount of direct control is provided by an extensive set of options on a cathode ray tube display and associated keyboard. These options include the following:

(1) Ability to interrupt the test in progress.

(2) Ability to scan, rescan, and record the discrete (and DDAS Discrete) input lines.

(3) Ability to determine the state of any discrete line or analog signal.

(4) Ability to send out any discrete output or analog stimulation.

(5) Ability to add, delete, or restore specific steps.

(6) Ability to repeat one or more steps.

(7) Ability to obtain considerable information on status, location, and history of a specific failure.

ATOLL is a tool to be used by the test engineer for automatic testing. The language does not verify the test engineer's techniques or verify that a complete test will be performed. The language, like all tools, is no better than the test engineer using it.

C. PROBLEMS AND LIMITATIONS

Several problems and limitations have developed since the introduction and implementation of ATOLL. The original goal was never completely met as the language did not provide the tools to adequately handle self contained data based systems. The basic language has proven to be an effective tool for test engineers interested in stage electrical networks and system integration. These tests are basically sequential, making them really adaptable to ATOLL. Other areas make only limited use of ATOLL for the purpose of interfacing with other systems tests.

Numerous mathematical and computer program control functions have been added. These features made ATOLL more oriented toward the computer programmer. However, ATOLL is not a good programming language. While these additions have improved the language's ability to cope with a greater variety of situations, the solution to the test engineer is poor as it is not written in his language.

Considering these additions, the language is still not well suited for guidance, control, and measuring systems. Symbolic machine language with the ATOLL interface is used.

This inability of ATOLL has been one of the major problems in implementation. Too many users and potential users have suggested and succeeded in introducing additional operators which have only complicated the situation.

While ATOLL is capable of operating under control of the countdown clock system, the language can best serve when operating at equipment speed. The countdown clock was created to slow operations to the reaction speed of man. The digital computer, being hardware equipment with comparable operational speeds, is not bound by this constraint. The degree of automation achieved is dependent upon the amount of effort the computer is allowed to do for the engineer. Being constrained to the countdown clock reduces the level achievable.

Another limitation is the language itself. ATOLL is not a pure engineering language but is a nomenclature and jargon developed around Marshall Space Flight Center's philosophy of automatic test and checkout. Other users differ on terminology, technique and philosophy of stage test and checkout. While ATOLL has improved technical communication between the vendor and the purchaser, the differences in approach have created confusion among some contractors.

D. LANGUAGE EXPANSION

The ATOLL concept permits expansion of the language. As has previously been discussed, the original set of 15 operators was expanded to approximately 60. Of this number, 30 have been used in systems developed by Marshall Space Center. The majority of the additional operators were included at the request of stage contractors to permit development of tests more in keeping with their own test philosophy. Thus, these changes tend to make the language more

complex and less of the pseudo-engineering language originally developed. The mathematical and logically operational changes are of the traditional form in general: Transfer (GOTO), Move data (MOVE), Set index (SETX), Subroutine Call (CALL), Reserve Memory (RESV), Algebraic Computation Mode (ALGE, a limited Fortran), and others.

While these additions have been quite helpful to the individual users, their overall value is difficult to judge. When the test engineer is forced to use and become very familiar with this expanded range of operations, the engineer becomes a full time ATOLL programmer and less of a test engineer.

Operators need to be defined and implemented that will allow the test engineer to solve these data based problems. The solution may be a degeneration to the symbolic source language of the computer being used. This is now being done on the implementation used for stage electrical system test; however, the test engineer must then rely on the computer programmer as he did before the development of ATOLL.

SECTION IV. EXPANDED APPLICATIONS

While ATOLL was originally designed for stage test and checkout, it has proved to be very useful in other research and development applications where sequential testing is required. The language will allow the test engineer to attach his own blackbox in to an interface unit without the preassignment stimuli and response lines. The test procedure or operation sequences are then coded in ATOLL and the test procedure compiled and run.

This technique does not require preassignment of channels or lines. The engineer just makes his assignments at the time of initial hookup. This is the approach being used in the Advanced Test and Checkout Method Evaluation Development (ATCOMED) facility, which a laboratory research and development test station. The system shows great promise as a forerunner of test stations and control computer uses to come.

ATOLL can also be used to verify stimuli and response of components as well as the larger stage systems. The approach is similar to the one discussed in the paragraph above. A component test station using a small general purpose computer is capable of testing even the most sophisticated component subassembly. By assigning ATOLL addresses

to points in the test station, an ATOLL system with a greater degree of test flexibility can then be established.

The uses of ATOLL are limited only by the imagination of the potential user. Some examples of additional areas are aircraft and rocket component testing, electronic assembly checkout, and other special purpose performance testing tasks.

SECTION V. A CONCURRENT DEVELOPMENT: ATOLL II

A. INTRODUCTION

In October of 1964, effort was initiated to develop an advanced ATOLL (later called ATOLL II). The improved language was to incorporate the added experience and hardware designs.

B. APPROACH

To develop and implement the advanced ATOLL (ATOLL II), the following approach was recommended:

(1) Redefinition of ATOLL operators and format as a joint effort of the users and computation personnel.

(2) Proceed with the development of the present ATOLL (ATOLL I) based on the operators and format specified.

(3) The computation personnel and proposed ATOLL II users were to begin immediately with the language development.

(4) Computation personnel were to provide milestones and schedules compatible with ATOLL II users requirements.

C. GOALS

As the original goals of ATOLL were not met by the language, a similar goal was established by the proposed user of ATOLL II: "To allow the test engineer to concern himself mainly with

the technical aspects of the test and equipment under test. The language used to prepare the product should relate directly to the language the engineer uses to describe the item to be tested and testing conditions and actions."

ATOLL II does not comply with this objective. Rather, ATOLL II is a computer programmer oriented language resembling a cross between FORTRAN, ALGOL, and COLBOL. However, as there was no FORTRAN for the RCA 110A computer, the system has real possibilities as a Fortran type compiler for that system.

D. STATUS

At present, the only planned implementation for ATOLL II is for use at the Saturn V launch site. This implementation is now near completion.

SECTION VI. OTHER APPROACHES

As pointed out in the earlier sections, none of the existing systems solve all of the automatic test control problems. ATOLL is only one approach to the overall problem. Several other approaches should be given serious consideration. Some of these will require improved hardware.

A. IMPROVED ATOLL II

The present trend for launch operations is toward ATOLL II. One approach would be to expand the scope of ATOLL II and allow ATOLL I to function as a functional subset. Here the test engineer could be concerned only with the ATOLL I subset. Computer programmers could use the additional capability of ATOLL II to accomplish the required testing of the self contained data based systems.

B. STANDARD TERMINOLOGY SYSTEM

ATOLL was developed by a committee comprised of programmers with test control experience. One of the chief problems was a lack of standard stage terminology. If such a language had existed, it would have been the basis for ATOLL I. When such a terminology does become available, a new language will be required.

C. MODIFIED BASIC COMPUTER LANGUAGE

The basic language of the computer is by far the most efficient way to go if the desire is to have a software routine of minimum size or maximum speed. The time required to generate these routines is also very long. Many of the present assembly techniques allow the introduction of marco operators. ATOLL I or a similar language could be made an integral part of the more flexible assembly language. Again, the test engineer would be concerned chiefly with the ATOLL subset.

D. MODIFIED HARDWARE

Present computer systems for automatic test control require computer software routines to perform the test operations. ATOLL is a collection of computer software routines linked together with a set of the test engineer's written calling sequences. The state-of-the-art now permits the designer of computer hardware to incorporate many of the test control operations into hardware itself. This improved hardware, together with future software, will undoubtedly be the future approach taken in test control.

DISTRIBUTION

R-QUAL-DIR, Mr. Trott/Mr. Tiller
R-ASTR-E, Mr. Fichtner
R-COMP-RD, Dr. Krenn
R-COMP-RDA, Mr. Marion (2)
R-QUAL-OCP, Mr. Krone (3)
R-QUAL-P, Mr. Brooks
R-QUAL-PC, Mr. Batte
R-QUAL-PF, Mr. Kessler
R-QUAL-PI, Mr. Rushing
R-QUAL-PS, Mr. Hall
R-QUAL-PSC, Mr. Funderburk
R-QUAL-PSC, Mr. Balentine (25)
R-TEST-IEW, Mr. Lide (5)
R-TO, Mr. Vedane (2)
MS-D, Mr. Garrett

November 30, 1966 SR-QUAL-66-25

APPROVAL

THE DEVELOPMENT OF A CHECKOUT LANGUAGE: ATOLL

The information in this report has been reviewed for security classification. Review of any information concerning Department of Defense or Atomic Energy Commission programs has been made by the MSFC Security Classification Center. This report, in its entirety, has been determined to be unclassified.

This document has also been reviewed and approved for technical accuracy.

B. J. Funderburk, Chief
Computer Systems Section

L. Hall, Chief
Vehicle Systems Integration Branch

C. O. Brooks, Jr., Chief
Vehicle Systems Checkout Division

D. Grau, Director
Quality and Reliability Assurance Laboratory