# AIAA SPACE FLIGHT TESTING CONFERENCE

COCOA BEACH, FLORIDA

MARCH 18-20, 1963

Consper

,3-102

#### **ADVANCED MYSTIC:**

## A COMPILER FOR MANAGEMENT CONTROL OF COMPUTER PROGRAMMING

By

R. G. Kelly Technitrol Engineering Corporation Philadelphia, Pennsylvania 63102

First publication rights reserved by AMERICAN INSTITUTE OF AERONAUTICS AND ASTRONAUTICS, 500 Fifth Ave., New York 36, N. T. Abstracts may be published without permission if credit is given to the author and to AIAA.

The Mystic<sup>1</sup> compiler is a simplified coding technique which was adapted to the IBM 650, 704, ERA 1103 AF calculators and which is compatible with the logic of most large-scale computing machines. Advanced Mystic<sup>2</sup> is a modification and extension of that technique currently in use on the IBM 7090. It is designed for applications requiring great input/output versatility, high compiling and operating speed and large storage capacity. The system uses a functional macro-operator structure, is fitted with a unique program interrupt feature and, though designed primarily for scientific computation, is nevertheless adaptable to a wide variety of applications including data reduction, commercial data processing, and analogue computer simulation.

Use of the Mystic system makes possible effective management of computer programming. The entire problem-programming process is clarified and simplified to where progress can be measured precisely, status can be assessed accurately, and time requirements can be estimated reliably. Such control can be achieved only with the use of a rigorously functional programming language and a program-building system which applies proven mass-production techniques to computer programming. Advanced Mystic is just such a programming system.

The computational process is determined by sequences of operators, called commands, which are in the form of multiple address statements. High operating speed results from the buffering of all input and output, from the careful selection of machine language configurations to represent each command and from the automatic deletion of superfluous instructions. The use of only one number notation contributes to the simplicity and effectiveness of the system. To simplify program checking, an automatic "pathfinder" is built into the system, and records the identity of the current command sequence at all times.

Advanced Mystic divides the memory of the machine into two classes: a fixed program area for the compiled program, and a variable operating area for inputs, computations, and results. The command structure controls the assignment of machine storage to the two classes. All but a few of the available storage registers are directly addressable in the system, and access is provided to all input and output units except system tapes.

In the variable operating area all numbers, special symbols, letters of the alphabet, and the ten digits when used as symbols are carried in normalized floating point form, i.e., an eight-digit mantissa and a two-digit exponent, each with sign,  $\pm xxxxxxx \pm xx$ .

<sup>1</sup> Gorman, T. P., and Kelly, R. G., Mystic: An Automatic Coder for the IBM 650, IBM 704, and ERA 1103-AF, ASTIA Document No. 134274, AFMTC-TN-58-2, Feb. 1958.

 $^2$  Developed jointly by T. P. Gorman and R. G. Kelly at the Goddard Space Flight Center, NASA.

- 1 -

Input and output operate in two modes. In all cases, transmission to and from input or output devices is of integers only. In the alphabetic mode, up to four coded characters can be so transmitted to or from a single word and in the numeric mode up to eight digits plus sign. Pre-written conversion-ofbase programs are used to prepare the needed integers from actual machine numbers and to prepare the needed numbers from the input integers.

The commands recognized by Advanced Mystic fall naturally into two groups - those effective during compilation of the program and those effective during execution of the program. The entire command repertoire is shown on pages 3, 4, and 5. The total number of commands recognized by the system is 22 with 6 of these being effective during compilation and 16 during execution. Those controlling program execution are sub-divided into four functional groups: Mathematical, Clerical, Logical, and Input/Output. Each command is described in detail below. A command sequence is a logical unit formed by a collection of commands to be executed in a given order. The commands are punched one to a card and the given order is that in which the cards are introduced into the machine. A single alphabetic character is used to identify each command and the Greek letters,  $\alpha$ ,  $\beta$ ,  $\gamma$ ,  $\delta$ ,  $\tau$ , are used to specify machine locations, the contents of such locations, actual numerical values or alphabetic characters depending upon the particular command.

#### ADVANCED MYSTIC COMMANDS

### **DURING COMPILATION OF PROGRAM**

	1. Begin	Set a begin point in the program sequence so that the program can start at this point on transferring to location $\alpha$ .			
	2. Key	Key, or relocate, all commands following this command (except Q), by an amount $\alpha$ .			
	3. Origin $\ldots \ldots O[\alpha] \ldots \ldots$	Locate the compilation of the machine lan- guage code starting in location $\alpha$ .			
	4. Cue.,	Fix the contents of location $\alpha$ to the contents of location $\beta$ .			
	5. Value $V[\alpha, n]$	Set n, a normalized floating point number, into location $\alpha$ .			
	6. Word $\ldots \ldots W[\alpha, w] \ldots$	Set w, a four character word into location $\alpha$ .			
DURING EXECUTION OF PROGRAM					
Mathematical	1. Add $\ldots \ldots \ldots A[\alpha, \beta, \gamma] \ldots$	Add the contents of locations $\beta$ and $\gamma$ and place the sum into location $\alpha$ .			
	2. Divide $D[\alpha, \beta, \gamma]$	Divide the contents of location $\beta$ by the con- tents of location $\gamma$ and place the quotient into location $\alpha$ .			
	3. Multiply $M[\alpha, \beta, \gamma]$	Multiply the contents of locations $\beta$ and $\gamma$ and place the product into location $\alpha$ .			
	4. Subtract $S[\alpha, \beta, \gamma]$	Subtract the contents of location $\gamma$ from the contents of location $\beta$ and place the remainder into location $\alpha$ .			
Clerical	1. Get	Get into location $\alpha$ the contents of the loca- tion specified by the number $\beta$ plus the con- tents of location $\gamma$ .			
	2. Hold H[ $\alpha, \beta, \gamma$ ]	Hold the contents of location $\gamma$ in the location specified by the number $\alpha$ plus the contents of location $\beta$ .			

- 3 -

		and the second	< .
Logical	3. Initialize $I[\alpha, n]$	Initialize location $\alpha$ to the value n, a normalized floating point number.	•
	4. Replace $R[\alpha,\beta]$	Replace the contents of location $\alpha$ by the contents of location $\beta$ .	
	(1. Compare C[α, β, γ, δ]	Compare the contents of location $\alpha$ with the contents of location $\beta$ . If $(\alpha) > (\beta)$ transfer to location $\gamma$ , if $(\alpha) < (\beta)$ transfer to location $\delta$ , if $(\alpha) = (\beta)$ continue with next instruction.	
	2. End $E[\alpha]$	Exit from the current instruction sequence by transferring to location $\alpha$ .	
	3. Function $\mathbf{F}[\alpha,\beta,\gamma]$	This instruction enables one to transfer to a function and, after its execution, continue to the next instruction. Normally, $\gamma$ is the input to the function, $\beta$ is the location of the Begin instruction for the function, and $\alpha$ is the location to receive the output.	
	4. Jump J[ $\alpha$ ]	Jump to the program tape designated by the contents of location $\alpha$ .	,
	5. Execute $\ldots \mathbf{X}[\alpha] \ldots \ldots$	Perform the machine-language instruction contained in cell $\alpha$ .	;
Imput/Output	Load In $\ldots \ldots L$ Put Out $\ldots p$ [ $\alpha, \beta, \gamma, \delta_1$ .	$ \delta_{18}, \tau_1 \tau_{18}$ ] These commands controlingut to core from cards or tapes, and output from core to cards, tapes or printer. Tapes may be in bcd or binary format.	1
	1. $\alpha$ specifies the first of a consecutivity mitted.	utive set of cells to be occupied or trans-	
	2. $\beta$ specifies the cell whose content the case of tapes, the number of in binary. Input Mode: If $(\beta) = 0$ space n records. Output Mode: rewind tape.	ints control the number of cards, or, in records in bcd or the number of words ), back-space one file; if $(\beta) = -n$ , back- If $(\beta) = 0$ , write end of file; if $(\beta) = -1$ ,	
	3. $\gamma$ specifies the type of input or of T for tape with A, B, C in B indicates binary, and a blank,	output, C for cards, P for print, and adicating tape units 1, 2, 3 also, bcd.	

4.  $\delta_1$  . . .  $\delta_{18}$  Each  $\delta$  is a two-digit number indicating the number of columns in a field on card. Up to 18 distinct fields can be specified.

5.  $\tau_1 \cdot \cdot \cdot \tau_{18}$  Corresponding to each  $\delta$ ,  $\tau$ , an alphabetic character is used to designate whether the field is A (alphabetic with a maximum of 4 characters), N (numeric with a maximum of 9 positions including sign), or S (a field to be skipped with a maximum of 15 positions).

Input/Output

Title... T [Alphanumeric characters from column 2 to column 72]... This command will produce the specified alphanumeric characters into positions 1 - 71 of an output card, a printed line or a bcd record.

5