NBSIR 74-445

Programming the Moore 5Z From Basic

Philip G. Stein

Institute for Basic Standards
National Bureau of Standards
Washington, D. C. 20234

January 28, 1974

Final Report

Prepared for
Optical Physics Division
Institute for Basic Standards
National Bureau of Standards
Washington, D. C. 20234

NBSIR 74-445

PROGRAMMING THE MOORE 5Z FROM BASIC

Philip G. Stein

Institute for Basic Standards
National Bureau of Standards
Washington, D. C. 20234

January 28, 1974

Final Report

Prepared for
Optical Physics Division
Institute for Basic Standards
National Bureau of Standards
Washington, D. C. 20234

U. S. DEPARTMENT OF COMMERCE, Frederick B. Dent, Secretary

NATIONAL BUREAU OF STANDARDS, Richard W. Roberts, Director

CONTENTS

PROGRAMMING THE MOORE 5Z FROM BASIC

by

Philip G. Stein

This publication assumes that you know almost everything already. This includes BOSS X, Interdata BASIC, and the Moore 5Z.[1] Information about these other things are found in the listed references.

The Moore system is programmed using machine-language subroutines callable from BASIC. The form is:

```
CALL n,A,B,C,D...
```

Where n is an integer that designates the subroutine being called

A is the first argument

B is the second argument

and so on.

n's in the range 1-9 are reserved for machine programming, and 1-7 are currently implemented. All arguments are numeric, global, real numbers, and are passed by name rather than by value. This means that dummy variables may be passed and arguments thereby returned from the machine language. No calls by value are allowed! (This would be like CALL n,A,3.2,C) No expressions are allowed in the argument field of calls, but they may appear in the calculation of subroutine number n.

The description of the operation of each CALL follows.

[1] Certain commercial equipment, instruments, or materials are identified in this paper in order to adequately specify the experimental procedure. In no case does such identification imply recommendation or endorsement by the National Bureau of Standards, nor does it imply that the material or equipment identified is necessarily the best available for the purpose.

CALL 1,A,B(1)   Used to read the encoders and the probe.

A is a flag indicating whether to proceed or to ignore the encoders and return to BASIC. It will be used as a dummy later.

A=0, read encoders

A=1, ignore this call

B is the first element of a (minimum) four-element list where the data will be found upon exit.

B(1) is the current reading of the probe. -2048 is full negative deflection, +2047 is full positive deflection. Zero is center. The probe switch must be on the ACCessory position for this reading to be valid.

B(2) is the current reading of the X encoder. The 10 microinch digit is truncated if the reading is greater than 9.99999 inches (plus or minus), but the decimal point is always in the right place. The sign is handled correctly.

B(3) is the current reading of the Y encoder, as above.

B(4) is the current reading of the Z encoder, as above.

CALL 2,A,B,C,D   Used to operate the sluing motors.

A is a flag indicating whether to run the motors or to stop them.

A=0 stop the sluing motors and return them to manual control

A=1 operate the sluing motors from the computer

B is the velocity for the X sluing motor. Minimum velocity is about B=5 if the axis is stopped, but may be as low as B=1 if it is reduced to that value after the axis is started. Maximum velocity is plus-or-minus 4096, corresponding to full speed of the motor. Any proportional speed in between is possible. Values greater than 4096 are treated as equal to 4096. The sign sense of only the X axis is reversed from the manual direction. Programming x=+4096 causes full speed in the negative X direction.
A programmed speed of zero for any single axis causes that axis to stop and return control to manual even though other axes may be moving.

C is the velocity for the Y sluing motor.

D is the velocity for the Z sluing motor.

CALL 3 General purpose STOP and clear.

This subroutine has no parameters. It stops all machine motion, clears all current stepper counts, returns all controls to manual mode, and then turns on the computer enable circuit.

Note: Computer enable circuit...

In order to prevent accidents when the machine is first turned on or in the event of a power failure, a hardware computer enable circuit has been installed. This circuit is normally turned OFF (preventing the computer from controlling the Moore) when the computer is powered up, when there is a power interruption, when the INT button is pressed on the computer, or when button D is pressed on the Moore operators panel.

The subroutine CALL 3 is the only way to turn ON the computer enable circuit. Call 3 first determines that all computer controlled functions are in their passive or off states so that the Moore will not move unexpectedly when the computer is enabled; then it closes the computer enable circuit.

CALL 4,A,B,C Initializes the real-time clock.

A is a flag equal to 0, 1, or 2.

A=0, turn off the real time clock

A=1, turn on the real time clock

A=2, turn on the real time clock, and display its contents on the front panel lights.

B is a constant which determines the clock rate, in milliseconds. It must be between 1 and 1000. The time-of-day countdown is automatically adjusted to compensate for the rate, so that ticks are always once per second. The clock rate is also used to determine stepping motor speed, and this is not adjusted to anything, but used as is. Clock rates which are not even divisors of 1000 will cause the time-of-day to be incorrect, as they will be divided into 1000 and truncated. Recommended speeds are 20, 25, 40, 50, 100, 200, 250, 500, and 1000 milliseconds.

C sets the initial value of the time-of-day clock, an integer representing seconds elapsed since an arbitrary zero. If this value is changed while the clock is running, an error of as much as one second may occur.

CALL 5,A(1) Reads the status of the system.

A is a three-element list. The first member of the list must be specified in the calling statement, the rest of the members are inferred.

A(1) is a flag on input. If it is zero, only A(1) will be returned on output, leaving the rest of the internal program status undisturbed. If A is non-zero when the subroutine is called, a full status check will be made and A(1) through A(3) will be meaningful.

A(1) is the current reading of the time-of-day clock on output from the subroutine.

A(2) is the emergency status indicator. It is a priority system working in daisy-chain fashion as follows: Many situations in the Moore room can cause emergencies: Limit switches (indicate 0.25 inch of axis travel before shut-down), buttons A, B, C and D on the operators panel, and other (yet to be installed) indicators are among them. Each emergency is given a number, with number 1 being the highest priority. The computer will remember these emergencies, and will queue them for CALL 5. A value of A(2) of 0 means that there are no emergencies in the queue. If A(2) is non-zero, it will be equal to the number of the highest priority emergency in the queue. Subsequent calls to CALL 5 will bring succeeding members of the queue into A(2). Each call to CALL 5 with A(1) non-zero (meaning read full status) removes an emergency from the stack, so if you do one, you had better be prepared to handle the emergency. If it is ignored and CALL 5 is used again, the emergency just removed from the stack is lost. That is why CALL 5 with A(1)=0 is used, to get the time of day without popping the emergency stack.

If a sequence of CALL 5's are being done, and low-priority emergencies are being removed from the stack, the occurrence of a high priority emergency during that time will cause it to be placed on top of the stack, and it will be the next one off. Low priority emergencies remain in the queue and do not get out of turn.

Computer interrupts for emergencies are handled as they happen, and are transferred to the CALL 5 routine at the end of each clock tick. A double interrupt on one line will, therefore, appear as two entries in the queue only if the queue has been emptied in that position before the next tick. That is to say that the hardware will remember two interrupts, but the software will merge the second into the first unless the first is acknowledged by BASIC before the next tick of the clock (programmed rate, not time-of-day clock). The clock must be on for all of this to work.

A(3) is the busy indicator. It can be used to determine whether the hardware stepper controllers are busy, whether the software stepper drivers are finished moving their respective axes, whether the dummy timer is finished, and the position of the spindle gearshift lever. (For more information on the dummy timer, see CALL 7.)

Values of A(3) are determined by summing different numerical integers for each busy.

Dummy timer:	1
Z Stepper :	2
Y Stepper :	4
X stepper :	8
Spindle shift AC :	16
Spindle shift step :	32

Absence of a busy condition results in 0 being added to A(3). Presence of the condition results in the integer from the above table being added to A(3).

CALL 6,A,B,C,D Operates the Step/slue clutches and other functions.

A controls the X axis stepper and clutch.

A=+ or - 1 : Forward or reverse stepping of the X table (respectively) Note: Clutch must be engaged before stepping is ordered, otherwise some steps may be lost.

A=+ or - 2 : CCW or CW stepping of the rotary table (1000 steps/degree) The above modes are for use with CALL 7 setting the computer to count steps whose rate is determined by CALL 4.

A=+ or - 3 : Forward or reverse indexing of the X table using the stepping motor controller to count steps.

A=+ or - 4 : CCW or CW indexing of the rotary table using the stepping motor controller to count steps.

B controls the Y axis stepper and clutch.

B=+ or - 1 : See above for A=1

B=+ or - 2 : See above for A=2

B= 3 or 4, undefined.

C controls the Z axis stepper and clutch.

C=+ or - 1 : See above for A=1.

C=+ or - 2 : See above for A=2.

C=+ or - 3 and 4 See above for A= 3 or 4, but applies to spindle stepper rather than rotary table. (5 steps/degree)

D controls the spindle AC drive.

D=+ or - 1 : CW or CCW sluing of the spindle.

Notes: 1) In all cases, an argument of 0 will stop the appropriate axis or rotation, return the machine to slue mode, and return to manual control. This may be done while continuing to operate other axes.

2) Parameters other than the above numbers will cause the program to execute an illegal instruction. when sending them from BASIC, check their values.

CALL 7,A,B,C,D Used to set stepper distances or counts.

A controls the X axis stepper count. A should be set to a positive integer for number of steps to be stepped. It will remain busy (see CALL 5) until that number of steps has been completed. The count is set to zero by CALL 3, and may be changed to new values, including zero, by subsequent references in CALL 7. The rate for the stepper is set by CALL 4, and the clutch must be activated when appropriate by CALL 6. This count is also used for the rotary table if selected by CALL 6.

B controls the Y axis stepper count in the same way.

C controls the Z axis stepper count in the same way. This count is also used for the spindle stepper if selected by CALL 6.

D controls the dummy timer count in the same way. This looks just like a stepping motor except that nothing moves. Time intervals measured in number of stepper clock cycles may be set up and run, with CALL 5 telling when they have expired.

Concurrency: In the case of CALL 7, stepper increments are put into the software in such a way that the stepping and counting is done concurrently with other BASIC programming. The computer will not wait for the stepping to be completed in this case. If you want to wait, write a loop using CALL 5 to test and see when this has been completed. The same is true of the time-of-day and dummy clocks; they run by stealing time from the running program but the running program does not have to wait. CALL 7 and CALL 4, therefore, exit to BASIC having initiated their actions, but before those actions have taken place.

References

Model 70 Users Manual

Interdata Publication 29-261, 1971

BASIC Language Reference Manual

Interdata Publication 29-338, 1973

BOSS Reference Manual

Interdata Publication 29-292R01, 1973

NBS-114A (REV. 7-73)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET	1. PUBLICATION OR REPORT NO. NBSIR 74-445	2. Gov't Accession No.	3. Recipient's Accession No.

4. TITLE AND SUBTITLE

PROGRAMMING THE MOORE 5Z FROM BASIC

5. Publication Date

6. Performing Organization Code

7. AUTHOR(S)

8. Performing Organ. Report No.
NBSIR 74-445

9. PERFORMING ORGANIZATION NAME AND ADDRESS

NATIONAL BUREAU OF STANDARDS
DEPARTMENT OF COMMERCE
WASHINGTON, D.C. 20234

10. Project/Task/Work Unit No.
2321123

11. Contract/Grant No.

12. Sponsoring Organization Name and Complete Address *(Street, City, State, ZIP)*

SAME

13. Type of Report & Period Covered
Final

14. Sponsoring Agency Code

15. SUPPLEMENTARY NOTES

16. ABSTRACT *(A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here.)*

Instructions for programming motions and data recording from the Moore 5Z measuring machine at the NBS dimensional technology laboratory, using machine-language calls in the BASIC programming language, as implemented on an Interdata model 70 computer.

17. KEY WORDS *(six to twelve entries; alphabetical order; capitalize only the first letter of the first key word unless a proper name; separated by semicolons)*

Computer; dimensions; numerical control; programming.

18. AVAILABILITY

□ Unlimited

☒ For Official Distribution. Do Not Release to NTIS

□ Order From Sup. of Doc., U.S. Government Printing Office
Washington, D.C. 20402, SD Cat. No. C13

□ Order From National Technical Information Service (NTIS)
Springfield, Virginia 22151

19. SECURITY CLASS (THIS REPORT)
UNCLASSIFIED

20. SECURITY CLASS (THIS PAGE)
UNCLASSIFIED

21. NO. OF PAGES
15

22. Price

AMERICAN REVOLUTION BICENTENNIAL 1776-1976