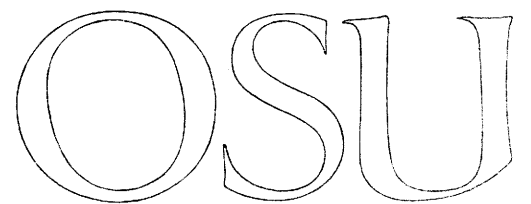


# An Introduction to the ATHENA Computer

by Brian Dumont

May 16, 1969

The logo for Oregon State University, consisting of the letters 'OSU' in a large, outlined, serif font.

**COMPUTER CENTER**

Oregon State University  
Corvallis, Oregon 97331

AN INTRODUCTION TO  
THE ATHENA COMPUTER

cc-69-9

by  
Brian 'EASY' Dumont

May 16, 1969

Computer Center  
Oregon State University  
Corvallis, Oregon 97331

## ABSTRACT

This paper is intended to be an introduction to the use and operation of the ATHENA computer.

## TABLE OF CONTENTS

I.	Abstract	i
II.	The ATHENA's Memory and Various Registers	1.
III.	The ATHENA's Console	3.
IV.	The Instruction Set	6.
V.	How to Use the ATHENA	12.
VI.	A Program to Find Octal Square Roots	14.
VII.	Other Things to Do	17.
VIII.	Further References	18.

## THE ATHENA'S MEMORY AND VARIOUS REGISTERS

The ATHENA has two types of memory. A drum memory is used to hold instructions while a 256 word "scratch pad" memory is used to hold intermediate results of computations.

The drum can hold 8,192 instructions. These instructions are 17-bit words. The drum can be loaded from one of the two paper tape readers or by hand through the Drum Transfer Register, one instruction at a time. Since there is no way (as yet) to generate paper tapes with the correct format, programs must be loaded by hand. The ATHENA fetches an instruction from the drum by finding the instruction location in the Program Address Register. The instruction at this drum address is then loaded into the Program Control Register for execution. The Program Address Register is then incremented so the next instruction may be fetched.

A scratch pad core word is 24 bits in length. Core words are stored from the Accumulator Register and other special registers, while core words are loaded into the Accumulator.

The functions of the various assorted registers are listed below:

PROGRAM ADDRESS REGISTER - This register contains the drum address for the next instruction during program execution. During loading of a program this register contains the address of the instruction being loaded.

PROGRAM CONTROL REGISTER - This register is used for program execution. It contains the next instruction to be executed.

DRUM TRANSFER REGISTER - This register contains the instruction to be loaded into the drum address contained in the Program Address Register.

ACCUMULATOR 1 - This register is used to hold a word that arithmetic and logic operations are to be performed on and then hold the results of these operations. (Note instruction set.) It will usually be referred to as the Accumulator or the A register.

ACCUMULATOR 2 - This is an internal register used to hold intermediate results when arithmetic and logic operations are performed on Accumulator 1.

QUOTIENT REGISTER 1 - This internal register is used as an extension of the Accumulator for multiply, divide and shift commands. It will usually be referred to as the Quotient Register or Q Register. (Note instruction set)

QUOTIENT REGISTER 2 - This internal register, like the Accumulator 2, is used to hold intermediate results of operations on the Q register.

DISPLAY REGISTER - This register is used to hold the quantity to be printed on the Adding Machine Printer.

EXCHANGE REGISTER (X) - This internal register is used to hold the word that is being transferred between a register and another register, core or the drum.

CONSTANTS REGISTER (CR) - This register contains eight 24-bit words that may be changed manually. Note the Enter Parameter instruction.

The following are special purpose registers that were used for missile guidance. Since we are lacking the radar system and missiles, they are of no real use, as of yet.

Steering Register

Acceleration Register

Discrete Register

Vernier Register

## THE ATHENA'S CONSOLE

All the registers mentioned previously are represented by push-button light switches on the upper left hand two-thirds of the console. A light on indicates a bit is in a one state.

These push-button light switches can be used to modify the bits within these registers. The white push-button to the right of all the registers marked CL clears the register. Depressing a push-button light will set the bit in the register it represents and also turns the light on. Thus all the registers may be modified manually by clearing the register and pushing the correct push-button lights representing the desired bits.

On the lower portion of the console there are quite a few push-button lights and switches that are useful. These will be discussed as they appear on the console from right to left.

IGNITION SWITCH - A key is required here. The key may be obtained in the Computer Center, Kidder 130. Turning the key to the right turns the computer power on.

POWER ON - After the ignition key is turned on the Power On switch is depressed. This turns the computer on and the drum begins to rotate. When the Power On switch turns green the drum is finally up to operating speed and the computer is ready to operate.

POWER OFF - This light should be on whenever the motor generator is on. If this is not on the motor generator must be turned on before the ignition switch is of any use. Someone in the NEBULA room should be able to turn the motor generator on if necessary.

READY - This switch is used to ready the computer before executing a program or loading a program. When Ready is pushed all registers are zeroed and an Unconditional Jump to drum address 40 instruction is placed in the Program Control Register. Thus it is practical to begin all programs at drum location 40.

STEP/STOP - The Ready switch must be on for this to work. If the Execute Program switch is on, a single instruction cycle will be generated each time the switch is pressed. If the Manual Load is on the Drum Transfer Register will be loaded into the drum at the address specified by the Program Address Register each time that switch is pressed.

RUN - Depressing the Run switch causes the computer to start executing a program. The Ready switch must be on for this to work.

DISTRIBUTE PULSE RATE - When this button is on, each instruction cycle will generate one timing pulse. This machine is a synchronous machine that requires seven timing pulses per instruction cycle. Thus, each of the individual results of a clock may be viewed from the console by using the Step/Stop button.

DISTRIBUTE CYCLE RATE - When this button is on, each instruction cycle will perform an instruction cycle.

NORMAL RATE - When this button is on, each instruction cycle will perform one instruction.

EXECUTE PROGRAM - When this button is on, a program may be executed from the drum by pressing Ready then Run or Step/Stop to single step through a program.

MANUAL LOAD - When this button is on, a program may be loaded into the drum by pressing Ready, placing the drum address (where the instruction is to be loaded) in the Program Address Register (PAR) and placing the instruction into the Drum Transfer Register (DTR), then punching Step/Start. Note the PAR is then incremented by one so a sequence of instructions may be loaded into the necessary drum positions. Thus, only when the instructions are not consecutive will the PAR need to be modified manually.

TARGET SWITCH NUMBER (TSN) - The target number that is on is used in the coefficient jump instruction (CJ). The target number is used as bits 11-8 of the drum address in the coefficient jump instruction.



MANUAL STOP - If this is on the Manual Stop (MS) instruction will work; otherwise, a No Operation (no op) will occur during this instruction.

TEST JUMP - If this is on the test jump (TJ) instruction will work; otherwise, a no op results.

AUDIBLE ALARM - This turns off the buzzer alarm that results a little while after the Power On switch is turned on.

PARTIAL SYNC - Depressing this causes a Partial Sync pulse. If the computer is waiting in a Wait Partial Instruction (WP) the next instruction will be executed after this pulse.

OSC SPEED - Turning this towards the right increases the speed at which the instructions are executed. A very slow instruction execution speed can result by turning the OSC to the left and turning the Distribute Pulse Rate on.

The following rotary switches should read normal unless a marginal condition is desired.

CORE READ

DRUM TIME

DRUM REAM

## THE INSTRUCTION SET

Before the instruction set is given the notation used will be described.

Octal notation (base eight) will be used to represent the contents of a word. This representation divides a word into groups of three starting from the right. The values of these groups taken in the same order is the octal representation of the word.

For example, the core word containing

```

111110101100011010001000
111 110 101 100 011 010 001 000
 7   6   5   4   3   2   1   0

```

is  $76543210_8$ .

The subscript 8 will be dropped since all numbers pertaining to the ATHENA should be in base 8. Any base 10 numbers used will be subscripted 10.

Just one exception: In designating bit positions, decimal digits are used.

To describe a bit or group of bits the word or register of these bits is subscripted with the bit or bit numbers.

$A_0$  represents the first bit (on right) of the Accumulator.

$Q_{9-0}$  represents bits 0 through 9 of the Q.

Another example would be the instruction

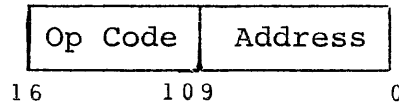
064023

```
00 110 10 0 000 010 111
```

Note: only 17 bits per instruction.

In describing an instruction it is convenient to divide the instruction into two parts. The parts are the address in the instruction bits 9-0 and the operation code (op code) in the instruction bits 16-10. The instruction 064023 has the op code 064 and the address 0023. The address portion of the

instruction will be noted as IA.



The words and symbols used in the instruction set are defined here:

Core (C) - This represents an eight-bit core address to be placed in IA<sub>9-0</sub>.

Shift - This represents a five-bit number that is the number of places to be shifted. This is placed in IA<sub>4-0</sub> of the instruction.

M - This represents the print code to be placed in address bits 2-0. The print codes that are of any use are

2 for decimal  
3 for octal

Note: For printing out in decimal the number must be arranged in a special format. Every four bits are used to represent a decimal number. Thus, to represent 8<sub>10</sub> decimal digits (maximum printed out) 32<sub>10</sub> bits must be used. Therefore, the decimal number printed out is from A<sub>8-Q0</sub>.

ND - The N is the number of the switch register on the constants register. N is a 3-bit quantity placed in address bits 5-3. The D is a 3-bit quantity placed in address bits 2-0 representing the digit position of the constants register to be added to the Accumulator (A). A zero indicates that all eight digits are to be added to the Accumulator.

Drum (D) - This represents a ten-bit drum address place in IA<sub>9-0</sub>.

( ) - Parenthesis around a word means the quantity of the word is used; while no parenthesis indicates the word itself. For instance:

(A) - means the contents of A

(Drum) - means the contents located at the drum address

Drum - means the actual drum address

← - "is replaced by".

(NPAR) - stands for the next drum address that an instruction will be fetched from. A distinction between (PAR) and (NPAR) is used due to the way the multiply, divide and shift instructions work. These commands may take up more than one instruction time. Each additional instruction time increments the (PAR) and inhibits the next instruction from occurring. This continues until the original instruction is done. Thus, the next instruction to be executed is not the normal (PAR)+1.

Unless otherwise noted the next instruction will be from (PAR)+1.

K - The decimal number of places to be shifted.

## THE ATHENA INSTRUCTION SET

<u>Mnemonic</u>	<u>Op Code</u>	<u>Address</u>	<u>Function</u>	<u>Name</u>
AD	064	CORE	$(A) \leftarrow (A) + (C)$	ADD
SB	066	CORE	$(A) \leftarrow (A) - (C)$	SUBTRACT
TP	060	CORE	$(A) \leftarrow (C)$	TRANSMIT POSITIVE
TN	062	CORE	$(A) \leftarrow -(C)$	TRANSMIT NEGATIVE
SA	004	CORE	$(C) \leftarrow (A)$	STORE A
MP	110	CORE	$(AQ) \leftarrow (A) * (C)$ $(NPAR) \leftarrow (PAR) + 15$	MULTIPLY
DV	112	CORE	$(Q) \leftarrow (Q) / (C)$ $(NPAR) \leftarrow (PAR) + 31$ Overflow can be set $(A) \leftarrow \text{Remainder}$	DIVIDE
TQ	102	SHIFT	$(AQ) \leftarrow (AQ)$ Shifted left $(PAR) \leftarrow (PAR) + \left(\frac{K+1}{2}\right)_{10}$	DIVISION SHIFT
LS	104	SHIFT	$(AQ) \leftarrow (AQ)$ Shifted left Overflow on if sign change $(PAR) \leftarrow (PAR) + \left(\frac{K+1}{2}\right)_{10}$	LEFT SHIFT
RS	106	SHIFT	$(AQ) \leftarrow (AQ)$ Shifted right $(A)_{22} \leftarrow \text{Sign of } (A)_{23}$ before shift $(PAR) \leftarrow (PAR) + \left(\frac{K+1}{2}\right)_{10}$	RIGHT SHIFT
AC	15-	DRUM	$(X)_{23-12} \leftarrow (A) + D_{11-0}$ $(A) \leftarrow (X)$	ADD CONSTANT
CA	14-	DRUM	$(X)_{23-12} \leftarrow D_{11-0}$ $(A) \leftarrow (X)$	CONSTANT TO A
CX	12-	DRUM	$(X)_{11-0} \leftarrow D_{11-0}$	CONSTANT TO X
DD	017	M	IF M=3 Print (A) IF M=2 Print $(A_8-Q_0)$	DISPLAY DATA

EP	002	ND	(A) ← (A) + (CR)	ENTER PARAMETER
UJ	20-	DRUM	(PAR) ← D <sub>10-0</sub>	UNCONDITIONAL JUMP
CJ	36- 37-	DRUM <sub>7-0</sub>	(PAR) <sub>11-8</sub> ← TSN (PAR) <sub>7-0</sub> ← D <sub>7-0</sub>	COEFFICIENT JUMP
SJ	22-	DRUM	If (A) is negative (PAR) ← D <sub>10-0</sub> otherwise (PAR) ← (PAR) + 1	SIGN JUMP
ZJ	24-	DRUM	If A is not zero (PAR) ← D <sub>10-0</sub> otherwise (PAR) ← (PAR) + 1	NON-ZERO JUMP
OJ	26-	DRUM	If OVERFLOW on (PAR) ← D <sub>10-0</sub> Turn OVERFLOW off otherwise (PAR) ← (PAR) + 1	OVERFLOW JUMP
TJ	30-	DRUM	If TEST JUMP SWITCH ON (PAR) ← D <sub>10-0</sub> otherwise (PAR) ← (PAR) + 1	TEST JUMP
WC	000	DRUM	Turns OVERFLOW off. If COMPUTATION CYCLE SYNC SIGNAL (PAR) ← D <sub>10-0</sub> otherwise Wait till COMPUTATION CYCLE SYNC SIGNAL	WAIT CYCLE
WP	34-	DRUM	If PARTIAL SYNC SIGNAL on (PAR) ← D <sub>10-0</sub> otherwise Wait till PARTIAL SYNC SIGNAL	
MS	32	DRUM	If MANUAL STOP is on Halt with this instruction remaining in the PAR otherwise (PAR) ← (PAR) + 1	

The following commands are of no real use.

<u>Mnemonic</u>	<u>Op Code</u>	<u>Name</u>
PC	014	Plot Cross Range
PA	015	Plot Altitude
PD	016	Plot Down Range
TS	020	Transmit Steering
TA	022	Transmit Acceleration
TD	024	Transmit Discretes
LV	026	Load V
SI	006	Store Input

## HOW TO USE THE ATHENA

To turn on ATHENA:

- 1) Make sure motor generator is on
- 2) Insert key into ignition switch
- 3) Press Power On button
- 4) Silence alarm by pressing the Disable Alarm button
- 5) Wait till Power On switch turns green

To execute an instruction:

- 1) Turn the Execute Program switch on
- 2) Turn the Ready switch on
- 3) Clear the Program Control Register
- 4) Put the instruction into the Program Control Register.  
Try the instruction 017003
- 5) Press any assorted buttons of the Accumulator
- 6) Press the Step/Stop button

To load a program into the Drum Memory:

- 1) Turn the Manual Load switch on
- 2) Turn the Ready switch on
- 3) Clear the Program Address Register
- 4) Load the starting drum address into the Program Address Register
- 5) Load the starting instruction into the Drum Transfer Register
- 6) Depress the Step/Stop button
- 7) Continue steps 5 and 6 until an instruction is out of sequence then do step 4

To run a program:

- 1) Make sure the Execute Program switch is on
- 2) Press Ready switch
- 3) Modify any registers if necessary
- 4) Turn Test Jump and Manual Stop on if they are to be used
- 5) Press Run to run the program or press Step/Stop to single step through the program



**EXAMPLE PROGRAM:**

Take the square root of a number and print it out.

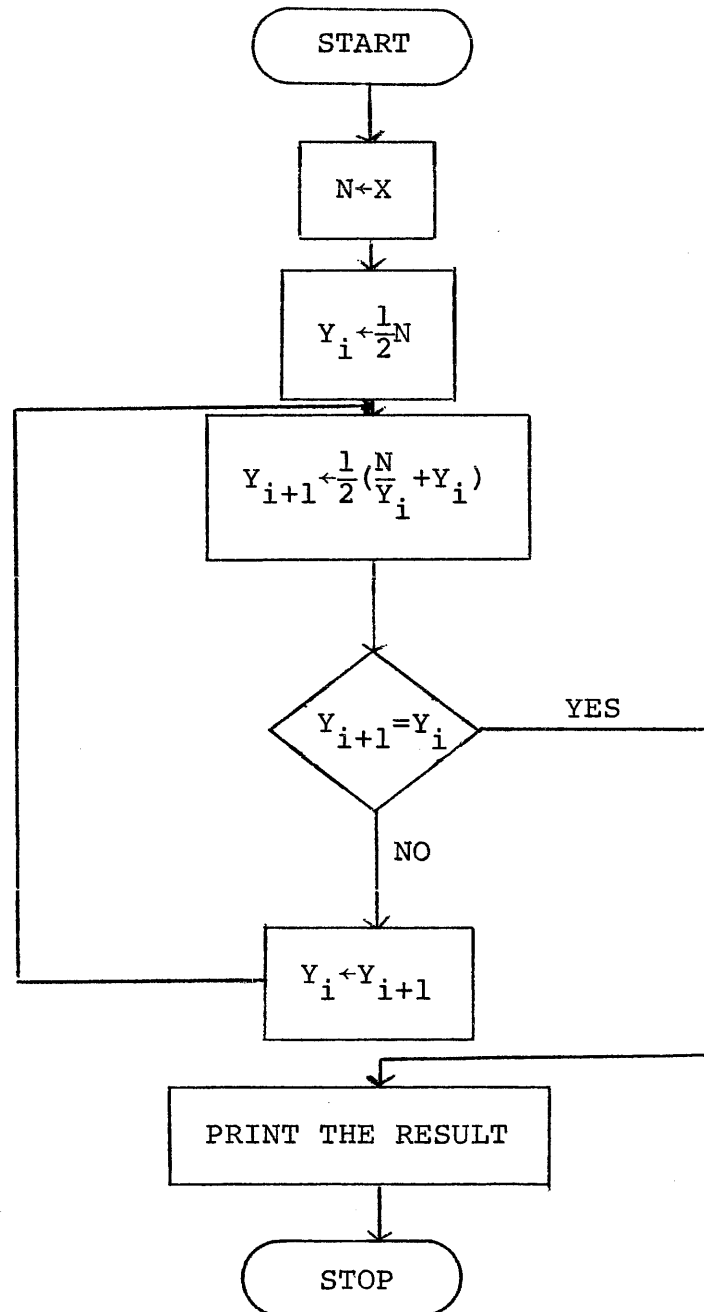
The following algorithm is used:

$$Y_{i+1} = \frac{1}{2} \left( \frac{N}{Y_i} + Y_i \right)$$

Where N is the number of which the square root is to be taken and i goes to K in steps of one. The larger K is the more accurate the answer will be.

In programming this, a simple loop is used. This is shown in the following flowchart.

## Square Root Program Flowchart:



From the flowchart the following program was written.

	Assembly Language Program	PAR	Machine Language Program
	SA N	40	004001
	RS 1	41	106001
	SA YIP1	42	004002
LOOP	TP YIP1	43	060002
	SA YI	44	004003
	TP N	45	060001
	RS 23 <sub>10</sub>	46	106027
		14{	
	DV YI	62	112003
		31{	
	TQ 22 <sub>0</sub>	113	102027
		14{	
	AD YI	127	064003
	RS 1	130	106001
	SA YIP1	131	004002
	SB YI	132	066003
	RS 1	133	106001
	ZJ LOOP	134	240043
	TP YIP1	135	060002
	DD 3	136	017003
	MS 7	137	320007

Note the gaps in the PAR.

To run this program:

- 1) Press Execute Program switch
- 2) Press Ready switch
- 3) Load the Accumulator with the number of which the square root is to be taken
- 4) Press Run

Are your results correct? Do the same with the square root of 121. If your program does not work right the first time you may have to debug the program.

A good procedure for debugging a program is to single step through your program with the Step/Stop switch. After each instruction is executed see if the desired results are in the register and check the Program Control Register to see if the next instruction is correct.

## OTHER THINGS TO DO

For other ideas of what you can do with the computer see the following suggestions:

- 1) Write a binary-to-decimal subroutine
- 2) A NIMS game that uses the A register for its field
- 3) Compute a list of prime numbers
- 4) Fix the drum so it may be written on during program execution
- 5) Interface a Teletype to the ATHENA
- 6) Interface a CRT to the ATHENA
- 7) Convert one of the registers to an index register
- 8) Add a few hundred instructions
- 9) Guide a missile
- 10) Write a Fortran compiler

## FURTHER REFERENCES

- 1) See any of the USAF publications on the ATHENA in the NEBULA room, Kidder 141
- 2) "The ATHENA On Line Assembler" by EASY, A paper describing an assembler for the ATHENA that does not work, as of yet.