The Origins of
Burroughs Extended Algol

Paul Kimpel

2019 UNITE Conference

Session MCP 4059

Wednesday, 2 October, 9:45 a.m.

It All Started in Pasadena…

Who Is this Guy?

Herbert Hoover

31st President of the United States

1874-1964

Herbert Hoover, Jr

1903-1969

- Herbert Hoover, Jr
  - Engineer (Stanford, 1925)
    - Lifelong interest in Radio
    - Built radio guidance network for Western Air Express
  - Entrepreneur
    - Started U.S. Geophysical, 1935 – explore for oil using radio
    - Spun off Consolidated Engineering Corp (CEC), 1937
    - Renamed Consolidated Electrodynamics Corp, 1955

2019 MCP 4059 3

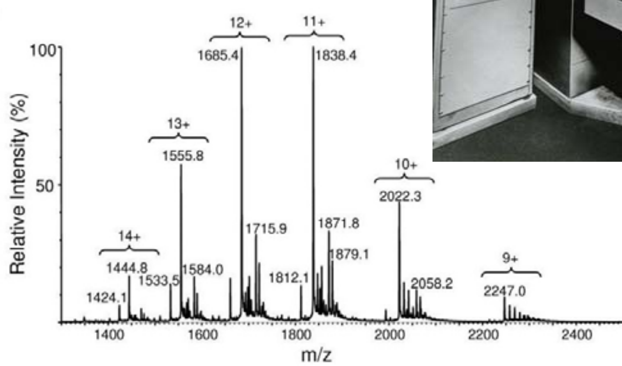
Consolidated Electrodynamics (CEC)

- Instrumentation for seismic exploration
  - Sensors, recorders
  - **Mass spectrometer**, 1942
- Mass spectroscopy analyzes compounds
  - Goal is to determine chemical composition
  - Ionizes a sample – passed through magnetic field
  - Yields a spectrum of mass/charge ratios (m/z)
  - Spectrum analysis requires solving *simultaneous linear equations* (*n* equations with *n* unknowns)
  - It's a *lot* of calculations

2019 MCP 4059 4

Mass Spectroscopy

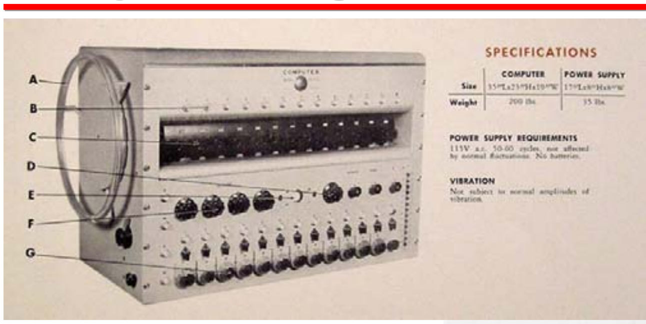
CEC model 21-103
Mass Spectrometer



Typical mass spectrum

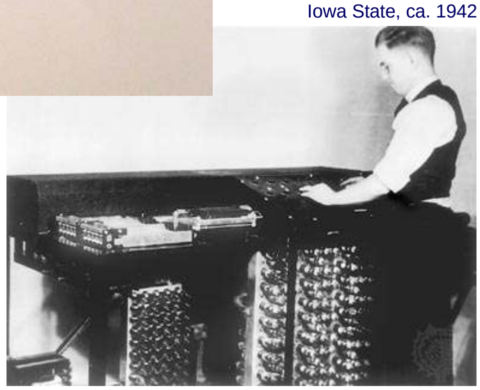
2019 MCP 4059 5

Berry's Analog Solution



CEC Analog Computer for
12 x 12 Equations

Berry at the Atanasoff-
Berry Computer (ABC),
Iowa State, ca. 1942



Clifford Berry

John Atanasoff

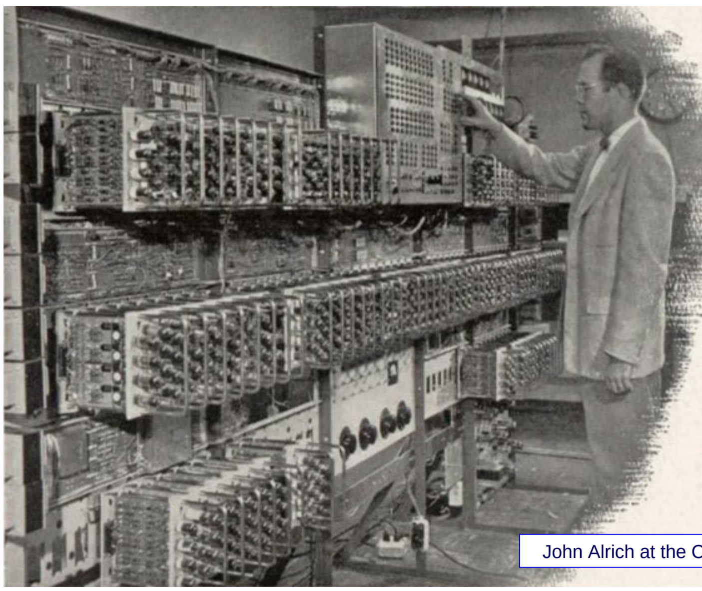
2019 MCP 4059 6

From Analog to Digital

- Analog computer worked, but was insufficient
  - Limited number of equations/unknowns (12 max)
  - Time-consuming, limited precision (~3 digits)
- CEC started researching digital computation
  - Initially intended to design a specialized calculator
  - Assumed 8 digits of precision adequate
  - Discovered customers did not want just a calculator
- CEC altered course to develop a full computer
  - Hired Harry Huskey to teach engineers digital logic
  - Hired Norwegian mathematician Ernst Selmer to design the arithmetic and control logic
  - Resulted in CEC 30-201, 30-202 prototypes (1952-54)

2019 MCP 4059 7

CEC 36-101 "Breadboard" System



John Alrich at the Controls

2019 MCP 4059 8

CEC → ElectroData → Burroughs

- CEC decided computers weren't their thing
  - Very capital-intensive, outside their main business
  - Spun off ElectroData as public corporation (1954)
  - Moved to 460 Sierra Madre Villa in Pasadena, CA
- ElectroData's success
  - Production model "Datatron 203" announced 2/1954
  - Models 204 (mag tape) and 205 ("Cardatron") by 1955
  - For a while, 3rd largest computer manufacturer in U.S.
- Financial pressures became overwhelming
  - Burroughs having trouble entering computer business
  - Offered to buy ElectroData in 1956
  - ElectroData became the "ElectroData Division"

ElectroData Datatron 205 (1955)

Datatron 20x Details

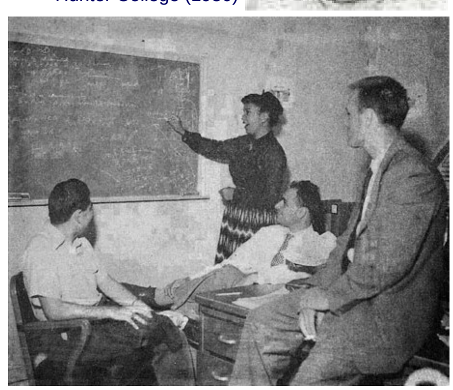
- Vacuum-tube, decimal, drum memory
  - 4000 11-digit words, 8.4ms access time
  - 80 words, 0.84ms access ("high-speed loops")
  - 142.8 KHz clock rate
  - Digit-sequential operation internally
  - First index register in U.S. ("B register")
  - Optional hardware floating-point
- Peripherals
  - 203 – paper tape, Flexowriter typewriter
  - 204 – adds fixed-block, dual-lane magnetic tape
  - 205 – adds Cardatron buffered card interface to IBM tabulating equipment (089, 523, 407)

2019 MCP 4059 11

It Wasn't Just Guys

Sibyl Rock
Mathematician, Analyst, Customer Liaison, Algorithm Designer
UCLA (1931)

Gloria Bullock
Mathematician,
Customer Education
First Datatron Programmer
Hunter College (1950)



2019 MCP 4059 12

A Second Life for the 205

Angry Red Planet, 1959

See: http://starringthecomputer.com/computer.php?c=45

Lost in Space, 1965

2019 MCP 4059 13

Burroughs 220 (1957)

2019 MCP 4059 14

Burroughs 220

- Follow-on to the Datatron 205
  - Larger core memory replaces drum memory
  - Still vacuum-tube, decimal, internally digit-sequential
  - 200KHz clock (up from 143KHz)
- Burroughs trying to make strong showing in both commercial and scientific applications
  - Same 11-digit words, hardware floating-point
  - Sophisticated magnetic tape subsystem
  - Cardatron buffered punched-card interface
- Automatic Programming group in Pasadena
  - Developing assemblers and programming aids
  - Working on IBM-compatible FORTRAN compiler

2019 MCP 4059 15

Enter Algol

Programming Was Hard in the '50s

- Difficult machines, primitive tools
  - Lots of programming in absolute machine code
  - Simple assemblers began to appear
- Most computation was numerical
  - Scientific, engineering, mathematical problems
  - Growing interest in automatically translating standard math notation to computer instructions
    - Short Code, Schmitt & Mauchly (BINAC/Univac I, 1950)
    - AUTOCODE, Glennie (Manchester Mark I, 1952)
    - A-0, Hopper (UNIVAC I, 1952)
    - I.T., Perlis (Purdue University, 1955, Datatron 205)
    - FORTRAN, Backus (IBM, 1957, IBM 704)
  - Growing interest in exchanging programs among different computer systems

2019 MCP 4059 17

The International Algebraic Language

- 1955-1957
  - German GAMM society working on general computing and formula translation
  - Conference in Los Angeles on exchanging computer data and programs
    - ACM, SHARE, USE, DUO
    - Concludes a universal programming language very desirable
- 1958
  - GAMM and ACM meet to exchange proposals
  - Joint session in Zurich to resolve differences
  - Result is "Preliminary Report – International Algebraic Language" (IAL)
  - Becomes known as "Algol-58"

2019 MCP 4059 18

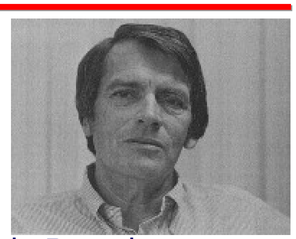
Algol-58 Example

```
procedure Simps (F(), a, b, delta, V);
          begin
     Simps:
          Ibar := V×(b-a);
          n := 1;    h := (b-a)/2;
          J := h×(F(a) + F(b));
     J1:
          S := 0;
          for k := 1 (1) n;
               S := S+F(a + (2×k-1)×h);
          I := J + 4×h×S;
          if (delta < abs(I-Ibar));
               begin
               Ibar := I;
               J := (I+J)/4;    n := 2×n; h := h/2;
               go to J1
               end;
          Simps := I/3;
          return;
          integer (k, n)
          end Simps

area := Simps(poly(), x, x+20, 2₁₀-5, 5₁₀25);
```

2019 MCP 4059 19

Meanwhile, Back in Houston…



- Robert S. ("Bob") Barton
  - 1954 – Takes job with Shell Development Research
  - 1957 – Working with young team on "Shell Assembler" for the 205
  - 1959 – Leaves Shell for Burroughs in Pasadena
- Part of team ("the arthropods") follows Barton
  - Joel Erdwinn
  - Clark Oliphint
  - Dave Dahm (still a summer-student employee)
- Barton heads Automatic Programming group
  - Now responsible for the ambitious IBM-compatible FORTRAN compiler project for the 220

2019 MCP 4059 20

220 BALGOL Compiler

- Barton realizes FORTRAN project is impossible
- Arthropods abandon FORTRAN, start on Algol-58
  - Erdwinn, Dahm
  - Later – Oliphint, Merner, Crowder, Speroni, Knuth
  - Initial compiler released March 1960
- The Burroughs Algebraic Compiler
  - Officially, "BAC-220"
  - Better known as Burroughs Algol, or **BALGOL**
  - Follows Algol-58 more closely than other dialects:
    - JOVIAL (SDC)
    - NELIAC (Naval Electronics Lab, San Diego)
    - MAD (University of Michigan)
    - ALGO (Bendix)

2019 MCP 4059 21

BALGOL Example

```
2 COMMENT SIMPSON-S RULE$
2 PROCEDURE SIMPS(A, B, DELTA, V$$ F())$
2   BEGIN
2   INTEGER K, N$
2   IBAR = V(B-A)$
2   N = 1$
2   H = (B-A)/2$
2   J = H(F(A) + F(B))$
2 J1..
2   S = 0$
2   FOR K = (1, 1, N)$
2     S = S + F(A + (2K-1)H)$
2   I = J + 4H.S$
2   IF DELTA LSS ABS(I-IBAR)$
2     BEGIN
2     IBAR = I$
2     J = (I+J)/4$
2     N = 2N$
2     H = H/2$
2     GO TO J1
2     END$
2   SIMPS() = I/3$
2   RETURN$
2   END SIMPS()$
```

```
2 FUNCTION TORADS(X) = 3.1415926X/180$
2 FUNCTION DARCTAN(X) = 1/(X*2 + 1)$
2 PROCEDURE LOGISTICSIGMOID(X)$
2   BEGIN
2   LOGISTICSIGMOID() = 1/(1 + EXP(-X))$
2   RETURN$
2   END LOGISTICSIGMOID()$
2
2 SUM = SIMPS(TORADS(30.0), TORADS(90.0),
      0.00001, 2.0$$ SIN())$
2 WRITE($$ RESULT, F1)$
2 SUM = SIMPS(0.0, 1.0, 1**-5, 2.0$$
      DARCTAN())$
2 WRITE($$ RESULT, F2)$
2 SUM = SIMPS(0.5, 3.0, 1**-5, 2.0$$
      LOGISTICSIGMOID())$
2 WRITE($$ RESULT, F3)$
2
2 OUTPUT RESULT(SUM)$
2 FORMAT
2    F1(*SINE INTEGRAL =    *,X10.6,W0),
2    F2(*DARCTAN INTEGRAL = *,X10.6,W0),
2    F3(*LOGISTIC INTEGRAL =*,X10.6,W0)$
2 FINISH$
```

2019 MCP 4059 22

BALGOL Features Over Algol-58

- Input-Output
  - Free-field input of numerics, strings
  - INPUT/OUTPUT list declarations for READ/WRITE
  - FORTRAN-like FORMAT declarations for output
- Language features
  - Implied multiplication: **`(X+Y)/2SQRT(Z)`**
  - UNTIL iterative statement
  - OTHERWISE clause for EITHER IF statement
  - Generic type declarations
  - Initialization of arrays
  - Code segmentation with program-controlled overlay
  - MONITOR, TRACE, DUMP diagnostics
- Linkage to machine-language routines

2019 MCP 4059 23

BALGOL Operational Advantages

- Fast, single-pass compiler (mag tape-based)
- Optimized for compile-and-go environment
- Configurable compiler environment
  - Generator program → compiler tape
  - Customize device types and I/O routines
  - Use larger memories (min 5000 words)
  - Augment/replace standard library
- Save and rerun object programs
  - Mag tape
  - Punched cards
  - Paper tape

2019 MCP 4059 24

Impact of BALGOL

- Proved value of compiler operational efficiency
  - Fast, one-pass compilation
  - Compile-and-Go environment
  - Monitoring and debugging aids
- Made the case for *regular use* of higher-level languages over assembly language
- *Customers loved it*
- Convinced Burroughs that Algol was viable
  - Planners believed it would displace FORTRAN
  - Showed that a different architecture was needed
- Provided much basis for design of B5000

2019 MCP 4059 25

Then There Was the *Other* BALGOL…

- Burroughs wanted a compiler like BALGOL for the 205
- They contracted with a Caltech grad student to write a compiler for $5500
- Donald Knuth wrote it *over the summer* in 1960

Donald Knuth

- Knuth continued to consult with Burroughs while at Caltech, until moving to Stanford in 1968
  - Worked on BALGOL (and wrote the comments)
  - Wrote first memory allocation scheme for the B5000

2019 MCP 4059 26

Burroughs B5000 (1962)

2019 MCP 4059 27

Burroughs B5000 / B5500

- Radical departure in hardware architecture
  - Specifically designed for Algol-60
  - Stack-oriented operation, code & data descriptors
  - Hardware support for Call-by-Name ("thunks")
  - Automatic segmentation & overlay ("virtual memory")
  - Multiprogramming & multiprocessing (2 CPUs)
  - Comprehensive operating system (MCP)
  - Programmed exclusively in high-level languages
- Reintroduced as B5500 in 1965
  - Large, fast Head-per-Track disk subsystem
  - Several new instructions, mostly for MCP use
  - Ancestor of B6x00/7x00, A Series, ClearPath MCP

2019 MCP 4059 28

Writing Algol in Algol

- If the compiler is written in itself…
  - How do you compile the compiler?
- B5000 method – *bootstrapping*
  - Defined a temporary implementation language: OSIL
  - OSIL used for B5000 MCP and Algol compiler
  - Assembler-like processor
  - Generated B5000 code, *but ran on the Burroughs 220*
- Wrote two compilers, side-by-side
  - Official one in Algol, then hand-compiled into OSIL
  - Debugged and updated both versions in parallel
  - Once the OSIL version could compile Algol – then the Algol version could compile itself

2019 MCP 4059 29

System Programming in a HLL

- The surprise when Algol could compile Algol…
  - Original compiler was about 8000 lines of Algol
  - OSIL "compile" on the 220 took ***9 HOURS***
  - Algol compile on the B5000 took ***4 minutes***
  - Algol-generated codefile was smaller, too
- Writing Algol in Algol worked so well…
  - Needed to rewrite MCP for the new HPT disk
  - MCP team took the Algol compiler, and…
    - Ripped out all the stuff for storage allocation, I/O, etc.
    - Added a few low-level features for hardware control
  - Called the result **ESPOL**
- MCP systems have never had an assembler

2019 MCP 4059 30

Issues with B5000 / B5500 Algol

- Array rows limited to 1023 words
- Lexical scope addressing
  - Cannot address intermediate nested environments
  - Can address outer-block and local-procedure only
- Character manipulation
  - B5x00 used high-order bit in word as a "flag" (tag)
  - Flag bit indicated control words (descriptors, etc.)
  - Implemented Word-Mode and Character-Mode states
    - Character-Mode originally intended to support COBOL
    - Implemented in Extended Algol as **Stream Procedures**
    - *Ignores flag bits and all memory address protection!*
    - Extremely useful – extremely dangerous
    - Prompted development of **POINTER**s, **SCAN**, **REPLACE**, etc.

Lexical Scoping Example

```
begin comment Knuth's Man-or-Boy Test;
    real procedure A (k, x1, x2, x3, x4, x5);
    value k; integer k, x1, x2, x3, x4, x5;
    begin
        real procedure B;
        begin k:= k - 1;
              B:= A := A (k, B, x1, x2, x3, x4);
        end B;
        if k <= 0 then
            A:= x4 + x5
        else
            B;
    end A;

    file dc (kind=remote, units=characters, maxrecsize=72);
    write (dc, <"Result = ",j11>, A (10, 1, -1, -1, 1, 0));
end.
```

Not allowed on B5000/5500

Note: Result = -67; run with STACK=9000

Stream Procedure Example

```
INTEGER STREAM PROCEDURE GETCHAR(A, OFFSET);
  VALUE OFFSET;
  BEGIN COMMENT
      RETURNS THE CHARACTER CODE AT THE LOCATION OF "A" OFFSET BY
      "OFFSET" CHARACTERS;
  LOCAL REP;                      % HOLDS DIV-64 REPEAT COUNT
  SI:= LOC OFFSET;                % SOURCE IS ADDRESS OF OFFSET WORD
  SI:= SI+6;                      % ADVANCE SOURCE BY 6 CHARACTERS
  DI:= LOC REP;                   % DEST IS ADDRESS OF REP WORD
  DI:= DI+7;                      % ADVANCE DEST BY 7 CHARACTERS
  DS:= CHR;                       % MOVE OFFSET 7TH CHAR TO REP 8TH
  SI:= A;                         % SOURCE IS ADDRESS IN A
  REP(SI:= SI+32; SI:= SI+32);    % SI:= *+ (OFFSET DIV 64)×64
  SI:= SI+OFFSET;                 % ADVANCE SOURCE BY (OFFSET MOD 64)
  DI:= LOC GETCHAR;               % DEST IS ADDRESS OF GETCHAR RESULT
  DS:= 7 LIT "0000000";           % CLEAR HIGH-ORDER 7 CHARS OF RESULT
  DS:= CHR;                       % MOVE SOURCE CHAR TO 8TH OF RESULT
  END GETCHAR;
```

Equivalent in B6700 Algol to: **`REAL(A[OFFSET],1)`**

2019 MCP 4059 33

Burroughs B6500/6700/7700 (1969)

2019 MCP 4059 34

Burroughs B6500/6700/7700

- Solved all major problems with B5500 Algol
  - Lexical scoping (32 "D" address registers, now 16)
  - Moved flag bit to separate 3-bit "tag" field
  - String instructions (**`POINTER`**s, **`SCAN`**, **`REPLACE`**)
  - Longer array rows (orig. 2^{20}, now 2^{32} words)
  - Segmented arrays, resizing of arrays
- Other Algol extensions
  - Powerful and convenient sub-tasking capabilities
  - DOUBLE, COMPLEX data types
  - Eventually – object-oriented structures
- Enhanced over time to become A Series and current ClearPath MCP systems

2019 MCP 4059 35

Extended Algol Anecdotes

The DEFINE

- Richard Waychoff, 1961
  - One of original B5000 Algol compiler authors
  - Was discussing symbol table design with Don Knuth
  - Knuth thought for a minute and said,

    "With that organization of a symbol table, you can allow one symbol to stand for a string of symbols."
- Originally, DEFINE was non-parametric
  - Dave Dahm implemented parametric DEFINEs in the late 1960s

2019 MCP 4059 37

Dollar Cards and Sequence Numbers

- It all started with a bad B5000 card reader…
  - As Algol grew above 2000 cards
  - Compilation from cards became a nightmare
  - So, Waychoff and Bobby Creech went to a bar…
    - Worked out a scheme to keep source on tape
    - Set aside columns 73-80 for sequence numbers
    - Merged tape with correction cards by sequence number
- Created $-card to signal source mode, e.g.,
  - **$ CARD**
  - **$ TAPE**
- Later, more options added
  - Listing control
  - Void cards from tape
  - Resequence, create NEWTAPE, etc.

2019 MCP 4059 38

Percent-Sign Comments

- Standard Algol comments
  - **COMMENT BLAH, BLAH, BLAH ;**
- B5000/5500 Algol
  - Used a Stream Procedure for scanning source
  - Needed an efficient way to detect end-of-card
  - Compiler overwrote column 73 of card image with "**%**"
  - Called the "stoplight" character
  - When "**%**" detected, compiler advanced to next card
- Didn't take long to figure out a free "**%**" anywhere on a card would stop the scan
- Effectively made rest of the card comments

2019 MCP 4059 39

Partial-Word Syntax

- Bit-field manipulation in Algol
  - **X:= A.[30:22];**
  - **Y:= A & B[7:8] & C[39:7:8];**
- Burroughs 220 had a similar feature for digits
  - Some instructions could operate on part of a word
    - Designated as the "sL" field (start-Length)
    - Start with digit "s" and use "L" digits *to the left*
  - Digit numbering: **± 1234 56 7890**
  - Example: **STA WD,63**

A Register: **+7631450822**

sL = 63

Memory at address WD: **+1719825634**

2019 MCP 4059 40

Array Row I/O

- B5000 Algol/MCP did not have array-row I/O
  - FORTRAN-like formatted I/O with lists
  - RELEASE statement for files
    - Buffer-level I/O (somewhat like modern Direct I/O)
    - Buffer only accessible as a Stream Procedure parameter
    - No blocking/unblocking support
    - Inefficient, a pain to use
- B5500 Disk File MCP introduced new I/O scheme
  - Read into and write from Algol array rows:

```
READ (F, 30, A[*]) [EOF];
```

  - Supports blocking and unblocking of records
  - Supports intelligent buffer handling (e.g., read-ahead)
  - Significantly more efficient

Input-Output List Declarations

- Algol formatted I/O has a LIST declaration
  -
```
LIST L1 (A, B, C+2, FOR I:=1 STEP 1
UNTIL N DO [ M[0,I], M[I,0], M[I,I] ]);
WRITE (LINE, FMT, L1);
```
  - Used with formatted READ & WRITE statements
  - On B5000, literal list could not be in the READ/WRITE
- Another carry-over from 220 BALGOL
  - INPUT and OUTPUT declarations
  - Created a co-routine called by the I/O formatters
  -
```
OUTPUT L1 (A, B, C+2, FOR I=(1,1,N) (
     M(0,I), M(I,0), M(I,I) ))$
WRITE ($$ L1, FMT)$
```

References

- ElectroData and 205 History – Tom Sawyer
  - http://www.tjsawyer.com/B205home.php
- B5000 Oral History Transcript (1985) – CBI
  - http://hdl.handle.net/11299/107105
- Stories of the B5000… – Richard Waychoff
  - http://www.ianjoyner.name/Files/Waychoff.pdf
  - https://archive.computerhistory.org/resources/access/text/2016/06/102724640-05-01-acc.pdf [scanned original]
- Algol Source Code and Emulators
  - http://www.phkimpel.us/ElectroData-205/
  - http://www.phkimpel.us/Burroughs-220/
  - http://www.phkimpel.us/B5500/
- This presentation
  - http://www.digm.com/UNITE/2019/

2019 MCP 4059 43

END

The Origins of

Burroughs Extended Algol