Burroughs B5500 Simulator Usage

01-May-2016

COPYRIGHT NOTICE

The following copyright notice applies to both the SIMH source and binary:

Original code published in 1993-2007, written by Robert M Supnik

Copyright (c) 1993-2016, Robert M Supnik

Burroughs B5500 simulator written by Richard Cornwell

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL ROBERT M SUPNIK OR RICHARD CORNWELL BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of Robert M Supnik or Richard Cornwell shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from both Robert M Supnik and Richard Cornwell.

Table of Contents

[1. Simulator Files 3](#__RefHeading___Toc1281_273995359)

[2. B5500 Features 4](#__RefHeading___Toc1283_273995359)

[2.1 CPU 4](#__RefHeading___Toc1285_273995359)

[2.2 I/O Channels (IO0..3) 6](#__RefHeading___Toc1287_273995359)

[2.3 I/O Devices 6](#__RefHeading___Toc1289_273995359)

[2.3.1 Console Station (CON) 6](#__RefHeading___Toc1291_273995359)

[2.3.2 B124 Card Reader (CR) 7](#__RefHeading___Toc1293_273995359)

[2.3.3 B303 Card Punch (CP) 7](#__RefHeading___Toc4484_273995359)

[2.3.4 B320 Line Printer (LP) 8](#__RefHeading___Toc4486_273995359)

[2.3.5 B422 Magnetic Tape (MT) 8](#__RefHeading___Toc4488_273995359)

[2.3.6 B5470 Disk Controller (DK) and B471 Electronic Storage Unit (ESU) 8](#__RefHeading___Toc4490_273995359)

[2.3.7 B430 Drum(DR) 9](#__RefHeading___Toc4492_273995359)

[2.3.8 B249 Terminal Controller. 9](#__RefHeading___Toc4500_273995359)

[3 Symbolic Display and Input 10](#__RefHeading___Toc4502_273995359)

[4 Character Codes 11](#__RefHeading___Toc4556_273995359)

This memorandum documents the Burroughs B5500 simulator.

The Burroughs 5500 was a unique machine, first introduced in 1961 as the B5000. Later advanced to the B5500 (1964) adding disks and finally the B5700 (1971) adding solid state drum. It was the first computer to use the stack as it's only means of accessing data. The machine used 48 bit numbers, all of which were considered to be floating point numbers. A computer word could also be used to hold up to 8 6-bit characters. All software for the machine was written in High level language, mostly Algol or Algol derivatives. Fortran and Cobol were also available for programming, but the operating system and all the compilers were written in Algol. There was a special version of Algol called EPSOL which allowed the generation of in-line instruction. Programs written in EPSOL had to be run directly from the hardware as the compiler generated absolute code only. One drawback to the machine was that all programs had to be developed in one language since linking was not part of the system.

The B5500 added a number of instructions to the B5000, including LLL, TUS, FBS. These instructions where to improve the performance of the machine. Also added was the XRT instruction which allowed for a subroutine to access the full PRT table. The B5700 was basically an enhanced B5500 with core memory instead of drum for fast on line storage. MCP used the drum (and later Auxmem) as a cache for frequently used segments.

# 1. Simulator Files

To compile the Burroughs B5500 simulator, you must define USE\_INT64 as part of the compilation command line.

|  |  |  |
| --- | --- | --- |
| ***Subdirectory*** | ***File*** | ***Contains*** |
| **I7000** | b5500\_defs.h | Burroughs B5500 simulator definitions |
|  | b5500\_cpu.c | B5500 CPU, Channel, interface |
|  | b5500\_io.c | B5500 I/O Channel. |
|  | b5500\_sys.c | B5500 System interface |
|  | b5500\_urec.c | B5500 Unit record devices. |
|  | b5500\_dk.c | B5470 Disk controller |
|  | b5500\_dr.c | B430 Drum or B6500 memory module. |
|  | b5500\_dtc.c | B249 Terminal Controller. |
|  | b5500\_mt.c | B422/B423 Magnetic tape controller. |

# 2. B5500 Features

The B5500 simulator is configured as follows:

|  |  |
| --- | --- |
| **Device Name(s)** | **Simulates** |
| **CPU** | B5500 CPU with 32KW of memory |
| **IO0..3** | B5500 I/O Controller |
| **MT0..16** | B422/B423 Magnetic Tape Controller |
| **CR0..1** | B124 Card Reader |
| **CP** | B303 Card Punch |
| **LP** | B320 Line Printer |
| **DK0..1** | B5470 Disk Controller |
| **ESU0..20** | B471 Electronic storage unit |
| **DK** | B430 Drum |
| **CON** | Supervisory Printer |
| **DTC** | B249 Terminal Controller. |

## 2.1 CPU

The CPU options include setting memory size and CPU type.

|  |  |
| --- | --- |
| SET CPU 4K | Sets memory to 4K |
| SET CPU 8K | Sets memory to 8K |
| SET CPU 12K | Sets memory to 12K |
| SET CPU 16K | Sets memory to 16K |
| SET CPU 20K | Sets memory to 20K |
| SET CPU 24K | Sets memory to 24K |
| SET CPU 28K | Sets memory to 28K |
| SET CPU 32K | Sets memory to 32K |
| SET CPU NOIDLE | Disables Idle detection |
| SET CPU IDLE | Enables Idle detection |

The B5500 typically included a second CPU which could be used to handle non I/O based computing. To enable the second CPU:

SET CPU1 ENABLE

This needs to be done before the system is booted, otherwise it will not be detected. Note the second CPU can be disabled laster and the system will continue to run with only one CPU.

CPU registers include the visible state of the processor as well as the control registers for the interrupt system.

| ***Name*** | ***Size*** | ***Comments*** | ***History*** |
| --- | --- | --- | --- |
| C | 15 | Instruction Pointer. |  |
| L | 2 | Syllable Pointer. |  |
| A | 48 | AR Top of stack register. |  |
| B | 48 | BR Second stack register. |  |
| X | 48 | X extension of BR. |  |
| GH | 6 | Combined G and H registers. |  |
| KY | 6 | Combined K and Y registers. |  |
| MA | 15 | Memory address register |  |
| S | 15 | Stack pointer register |  |
| F | 15 | Frame pointer register |  |
| R | 15 | PRT pointer or Tally register |  |
| P | 48 | Last program word read |  |
| T | 12 | Current instruction |  |
| Q | 9 | Error flags. |  |
| AROF | 1 | AR valid flag | A |
| BROF | 1 | BR valid flag | B |
| PROF | 1 | P has valid program word |  |
| TROF | 1 | T has valid instruction |  |
| NCSF | 1 | Normal/Control mode | N |
| SALF | 1 | Subprogram mode | S |
| CWMF | 1 | Character/Word mode | C |
| MSFF | 1 | Mark stack flag mode | M |
| VARF | 1 | Variant flag | V |
| HLTF | 1 | Halt flag. |  |
| IAR | 15 | Interrupt pending register |  |
| TUS | 32 | Peripheral ready status. |  |
| HALT | 1 | Halt processor |  |

The CPU can maintain a history of the most recently executed instructions.

This is controlled by the SET CPU HISTORY and SHOW CPU HISTORY commands:

|  |  |
| --- | --- |
| SET CPU HISTORY | clear history buffer |
| SET CPU HISTORY=0 | disable history |
| SET CPU HISTORY=n | enable history, length = n |
| SHOW CPU HISTORY | print CPU history |
| SHOW CPU HISTORY=n | print first n entries of CPU history |

Instruction tracing shows the CPU flag (0 for P1, 1 for P2), the instruction address CL, the A, B, X, S, F, R, M, GH & KV registers before the instruction execution. The flags are show if they are set. The Q, symbolic instruction and lastly the pending interrupt vector.

## 2.2 I/O Channels (IO0..3)

The B5500 supported up to 4 I/O Channels. These channels were floating and could talk to any device. The simulator allows for any of the channels to be enabled or disable.

Channels have the following registers:

| ***Name*** | ***Size*** | ***Comments*** |
| --- | --- | --- |
| D | 48 | Data control register. |
| CC | 6 | Command Code |
| W | 48 | Data buffer register |

## 2.3 I/O Devices

### 2.3.1 Console Station (CON)

The console station allows for communications with the operating system. The station is half duplex and will either print or accept input. Whenever the computer sends a message it is prefixed with a ‘R’ character. When the station is ready to receive input it prompts with a ‘I’. Input is buffered until the return character is entered. Backspace will remove the last character typed. An <esc> will send an interrupt to the processor to request it read a record from the console. An <esc> while in input mode will cancel input mode and clear any typed message.

### 2.3.2 B124 Card Reader (CR)

The card reader (CR) reads data from a disk file. Cards are simulated as ASCII lines with terminating newlines. Card reader files can either be text (one character per column) or column binary (two characters per column). The file type can be specified with a set command:

|  |  |
| --- | --- |
| SET CR*n* FORMAT=TEXT | Sets ASCII text mode |
| SET CR*n* FORMAT=BINARY | Sets for binary card images. |
| SET CR*n* FORMAT=BCD | Sets for BCD records. |
| SET CR*n* FORMAT=CBN | Sets for column binary BCD records. |
| SET CR*n* FORMAT=AUTO | Automatically determines format. |
| SET CR*n* EOF | Return EOF when deck as been read. |

or in the ATTACH command:

|  |  |
| --- | --- |
| ATTACH CR*n* *<file>* | Attaches a file |
| ATTACH CR*n* -f *<format> <file>* | Attaches a file with the given format. |

The card reader can be booted with the:

|  |  |
| --- | --- |
| BOOT CR0 | Loads first 3 words of card. |

If EOF is set it will be cleared when the card deck has been fully read, and must be reset on each attach command.

The B5500 could support up to 2 card readers. CR1 is by default disabled.

Error handling is as follows:

error processed as

not attached report error and stop

end of file out of cards

### 2.3.3 B303 Card Punch (CP)

The card reader (CP) writes data to a disk file. Cards are simulated as ASCII lines with terminating newlines. Card punch files can either be text (one character per column) or column binary (two characters per column). *(The B5500 only supported text output)*. The file type can be specified with a set command:

|  |  |
| --- | --- |
| SET CP FORMAT=TEXT | Sets ASCII text mode |
| SET CP FORMAT=BINARY | Sets for binary card images. |
| SET CP FORMAT=BCD | Sets for BCD records. |
| SET CP FORMAT=CBN | Sets for column binary BCD records. |
| SET CP FORMAT=AUTO | Automatically determines format. |

or in the ATTACH command:

|  |  |
| --- | --- |
| ATTACH CP <file> | Attaches a file |
| ATTACH CP -f <format> <file> | Attaches a file with the given format. |

Error handling is as follows:

error processed as

not attached report error and stop

OS I/O error report error and stop

### 2.3.4 B320 Line Printer (LP)

The line printer (LP) writes data to a disk file as ASCII text with terminating newlines. Currently set to handle standard signals to control paper advance.

|  |  |
| --- | --- |
| SET LP*n* LINESPERPAGE=*lpp* | Sets number of lines per page on printer. |

The channel skips are set as follows:

|  |  |
| --- | --- |
| 1 | Skip to top of form. |
| 2 | Skip to top of form. |
| 3 | Skip to next even line. |
| 4 | Skip to next odd line. |
| 5 | Skip to middle or top of page. |
| 6 | Skip ¼ of page. |
| 7 | Skip one line. |
| 8 | Skip one line. |
| 9 | Skip one line. |
| 10 | Skip one line. |
| 11 | Skip one line. |
| 12 | Skip to top of page. |

### 2.3.5 B422 Magnetic Tape (MT)

These come in groups of 16 units each. The controller defines which channel the devices will be on.

Each individual tape drive support several options: MT used as an example.

|  |  |
| --- | --- |
| SET MT*n* LOCKED | Sets the mag tape to be read only. |
| SET MT*n* WRITEENABLE | Sets the mag tape to be writable. |

### 2.3.6 B5470 Disk Controller (DK) and B471 Electronic Storage Unit (ESU)

The B5500 could support up to 2 disk controllers. Each of these could be attached to up to 10 Drive units which each could have up to 5 individual drives. The simulator models only full storage units. Disk drives could also be shared by the controllers, this limited the maximum storage units to 10.

Each of the disk controllers can be enabled or disabled. Or they can be set to share units.

|  |  |
| --- | --- |
| SET DK NODFX | Support up to 20 ESU units. |
| SET DK DFX | Supports sharing of ESU units for a maximum of 10 units. |

Each ESU can be attached to a disk file. The simulator supports two types.

|  |  |
| --- | --- |
| SET ESU*n* MODI | Set storage unit to type I which supported up to 48 million characters. |
| SET ESU*n* MODIB | Set storage unit to type IB which supported up to 96 million characters. |

If DFX is enabled, only 10 ESU units maximum should be attached. If DFX is disabled, then unit ESU10 corresponds to the first unit on DK1.

### 2.3.7 B430 Drum(DR)

The B430 Drum addressed up to 32K words and was used by MCP to cache segments. In later models this was changed to a B6500 memory modules to offer a solid state drive.

|  |  |
| --- | --- |
| SET DR DRUM | Set DR to emulate a drum. Buffering to file. |
| SET DR AUXMEM | Set DR to emulate a solid state drive. No file attachment. |

### 2.3.8 B249 Terminal Controller.

The B249 Terminal Controller can support up to 32 lines, with a maximum of 112 characters buffer per line. The default is to support 8 lines. The number of lines is set with the SET LINES command. Note this command must be set before the DTC is attached. And can’t be changed while the DTC is attached.

|  |  |
| --- | --- |
| SET DTC LINES=*n* | Set number of lines to allow, 0-32. |

The buffer size of all lines can be modified based on MCP’s needs and what it is configured to. Note this command must be set before the DTC is attached. And can’t be changed while the DTC is attached.

|  |  |
| --- | --- |
| SET DTC BUFSIZE=*n* | Set the buffer size for all connections. |

The ATTACH command specifies the port to be used for Telnet sessions:

|  |  |
| --- | --- |
| ATTACH COM *<port>* | set up listening port |

where port is a decimal number between 1 and 65535 that is not being used other TCP/IP activities.

Once DTC is attached and the simulator is running, the DTC listens for connections on the specified port. It assumes that any incoming connection is a Telnet connections. The connections remain open until disconnected either by the Telnet client, a SET COM DISCONNECT command,or a DETACH COM command.

|  |  |
| --- | --- |
| SET DTC DISCONNECT=*n* | Disconnect line *n* |

The DTC implements the following special SHOW commands

|  |  |
| --- | --- |
| SHOW DTC CONNECTIONS | Displays current connections to the DTC |
| SHOW DTC STATISTICS | Displays statistics for active connections |
| SHOW DTC LOG | Display logging for all lines. |

Logging can be controlled as follows:

|  |  |
| --- | --- |
| SET DTC LOG=*n*=*filename* | Log output of line *n* to *filename* |
| SET DTC NOLOG | Disable logging and close log file |

When a connection is established to the DTC it supports the following special character functions.’^’ represents Control Character.

|  |  |
| --- | --- |
| ^E | SendENQ, Who are you? |
| ^B | Send STX, Forces special control character into buffer. |
| } | Append character to buffer, and terminate line |
| CR | Terminate line. |
| LF | Terminate line. |
| ~ | Terminate line. |
| ^U | Clear input buffer. |
| BS | Delete last character entered. |
| DEL | Delete last character entered. |
| ? | Set abnormal flag. |

# 3 Symbolic Display and Input

The B5500 simulator implements symbolic display and input. These are controlled by the following switches to the EXAMINE and DEPOSIT commands:

|  |  |
| --- | --- |
| -c | Display/Enter Character mode syllables |
| -w | Display/Enter Word mode syllables. |
| -b | Display/Enter Character data. |
|  | Display/Enter Octal data. |

The symbolic input/display supports 4 formats for instruction display:

* <opcode>
* <opcode> <six bit constant>
* <opcode> <four bit constant>
* <opcode> <10 bit constant>

# 4 Character Codes

| BCI | BCD | ASCII | Card Code |
| --- | --- | --- | --- |
| 00 | 12 | 0 | 10 |
| 01 | 01 | 1 | 1 |
| 02 | 02 | 2 | 2 |
| 03 | 03 | 3 | 3 |
| 04 | 04 | 4 | 4 |
| 05 | 05 | 5 | 5 |
| 06 | 06 | 6 | 6 |
| 07 | 07 | 7 | 7 |
| 10 | 10 | 8 | 8 |
| 11 | 11 | 9 | 9 |
| 12 | 37 | # | 3-8 |
| 13 | 77 | @ | 4-8 |
| 14 | 72 | ? | 2-8 |
| 15 | 15 | : | 5-8 |
| 16 | 16 | > | 10-6-8 |
| 17 | 17 | } | 7-8 |
| 20 | 60 | + | 12-10 |
| 21 | 61 | A | 10-1 |
| 22 | 62 | B | 10-2 |
| 23 | 63 | C | 10-3 |
| 24 | 64 | D | 10-4 |
| 25 | 65 | E | 10-5 |
| 26 | 66 | F | 10-6 |
| 27 | 67 | G | 10-7 |
| 30 | 70 | H | 10-8 |
| 31 | 71 | | | 10-9 |
| 32 | 73 | . | 10-3-8 |
| 33 | 33 | [ | 10-4-8 |
| 34 | 20 | & | 12 |
| 35 | 34 | ( | 10-5-8 |
| 36 | 66 | < | 10-6-8 |
| 37 | 35 | ~ | 10-7-8 |
| 40 | 52 | | | 11-10 |
| 41 | 41 | J | 11-1 |
| 42 | 42 | K | 11-2 |
| 43 | 43 | L | 11-3 |
| 44 | 44 | M | 11-4 |
| 45 | 45 | N | 11-5 |
| 46 | 46 | O | 11-6 |
| 47 | 47 | P | 11-7 |
| 50 | 50 | Q | 11-8 |
| 51 | 51 | R | 11-9 |
| 52 | 53 | $ | 11-3-8 |
| 53 | 54 | \* | 11-4-8 |
| 54 | 40 | - | 11 |
| 55 | 74 | ) | 11-5-8 |
| 56 | 56 | ; | 11-6-8 |
| 57 | 57 | { | 11-7-8 |
| 60 | 00 | blank |  |
| 61 | 21 | / | 10-1 |
| 62 | 22 | S | 12-2 |
| 63 | 23 | T | 12-3 |
| 64 | 24 | U | 12-4 |
| 65 | 25 | V | 12-5 |
| 66 | 26 | W | 12-6 |
| 67 | 27 | X | 12-7 |
| 70 | 30 | Y | 12-8 |
| 71 | 31 | Z | 12-9 |
| 72 | 33 | , | 10-3-8 |
| 73 | 17 | % | 10-4-8 |
| 74 | 52 | ! | 10-2-8 |
| 75 | 13 | = | 10-5-8 |
| 76 | 55 | ] | 10-6-8 |
| 77 | 56 | " | 10-7-8 |