IBM 1620 Simulator Usage
01-Dec-2008

COPYRIGHT NOTICE

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

This memorandum documents the IBM 1620 simulator. This simulator is based on Geoff Kuenning's 1620 simulator, which is used by permission.

1 Simulator Files

sim/	scp.h
	sim_console.h
	sim_defs.h
	sim_fio.h
	sim_rev.h
	sim_sock.h
	sim_timer.h
	sim_tmxr.h
	scp.c
	sim_console.c
	sim_fio.c
	sim_sock.c
	sim_timer.c
	sim_tmxr.c
sim/i1620/	i1620_defs.h
	i1620_cpu.c
	i1620_fp.c
	i1620_tty.c
	i1620_pt.c
	i1620_cd.c
	i1620_lp.c
	i1620_dp.c
	i1620_sys.c

2 IBM 1620 Features

The IBM 1620 simulator is configured as follows:

```
device names      simulates

CPU               IBM 1620 Model 1 or Model 2 CPU with 20K to 60K memory
                  Model 1 options: indirect addressing, automatic divide,
                   edit instructions, floating point
                  Model 2 options: indexing, binary capability, floating
                   point
TTY               IBM console terminal
PTR               IBM 1621 paper tape reader
PTP               IBM 1624 paper tape punch
CDR,CDP           IBM 1622 card reader/punch
LPT               IBM 1443 line printer
DP                IBM 1311 disk pack with four drives
```

The IBM 1620 simulator implements many unique stop conditions. On almost any kind of error the simulator stops:

- Unimplemented opcode
- Reference to non-existent device

- Invalid digit
- Invalid alphameric character
- Invalid P address digit
- Invalid Q address digit
- Indirect address limit exceeded
- Invalid odd address
- Invalid even address
- Invalid function
- Invalid indicator
- Invalid return address register
- Skip to unpunched carriage control tape channel
- Card reader hopper empty
- Overflow with arithmetic stop switch set
- I/O error with I/O stop switch set
- Invalid disk drive
- Invalid disk sector address
- Invalid disk sector count
- Invalid disk buffer address
- Disk address compare error
- Disk cylinder overflow error
- Disk write check error
- Field exceeds memory
- Record exceeds memory
- Floating point mantissa exceeds maximum length
- Floating point mantissas not the same length
- Floating point exponent check with arithmetic stop switch set
- Floating point exponent missing high flag

The `LOAD` command is used to load a line printer carriage-control tape. The `DUMP` command is not implemented.

2.1 CPU

The CPU options include the CPU model (Model 1 or Model 2), a number of special features, and the size of main memory.

```
SET CPU IA                  enable indirect addressing
SET CPU NOIA                disable indirect addressing
SET CPU EDT                 enable extra editing instructions
SET CPU NOEDT               disable extra editing instructions
SET CPU DIV                 enable divide instructions
SET CPU NODIV               disable divide instructions
SET CPU IDX                 enable indexing
SET CPU NOIDX               disable indexing
SET CPU BIN                 enable binary instructions
SET CPU NOBIN               disable binary instructions
SET CPU FP                  enable floating point instructions
SET CPU NOFP                disable floating point instructions
SET CPU MOD1                set Model 1
SET CPU MOD2                set Model 2
SET CPU 20K                 set memory size = 20K
SET CPU 40K                 set memory size = 40K
SET CPU 60K                 set memory size = 60K
```

Model 1 options include IA, EDT, DIV, and FP; the first three are on by default. Model 2 options include IDX, BIN, and FP; IA, EDT, and DIV are standard on the Model 2.

If memory size is being reduced, and the memory being truncated contains non-zero data, the simulator asks for confirmation. Data in the truncated portion of memory is lost. Initially, the CPU is a Model 1, memory size is 20K, and indirect addressing, editing instructions, and divide are enabled.

Memory is implemented as 5 bit BCD digits, as follows:

```
4     3     2     1     0

flag  8     4     2     1
      <----- digit ----->
```

In BCD, the decimal digits 0-9 are (hex) values 0x0, 0x1, 0x2, 0x3, 0x4, 0x5, 0x6, 0x7, 0x8, 0x9, respectively. 0xA is record mark, 0xC non-punching blank, and 0xF group mark, respectively.

CPU registers include the visible state of the processor. The 1620 has no interrupt system.

name	size	comments
IR1	16	instruction storage address register (PC)
IR2	16	return register
PR1	16	processor register 1
PAR	16	P address register (OR2)
QAR	16	Q address register (OR1)
SS1	1	sense switch 1
SS2	1	sense switch 2
SS3	1	sense switch 3
SS4	1	sense switch 4
HP	1	high/positive indicator
EZ	1	equal/zero indicator
ARCHK	1	arithmetic check (overflow) indicator
EXPCHK	1	exponent check indicator
RDCHK	1	read check indicator
WRCHK	1	write check indicator
ARSTOP	1	arithmetic check stop switch
IOSTOP	1	I/O check stop switch
IND[0:99]	1	indicator array
IAE	1	indirect address enable (Model 2 only)
IDXE	1	indexing enable (Model 2 only)
IDXB	1	indexing band select (Model 2 only)
IR1Q[0:63]	16	IR1 prior to last branch; most recent IR1 change first
WRU	8	nterrupt character

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
```

The maximum length for the history is 65536 entries.

2.2 Console Typewriter (TTY)

The console typewriter (TTY) is a half-duplex console. The typewriter registers are:

name	size	comments
COL	7	current column
TIME	24	polling interval

When the 1620 CPU requests input from the keyboard, a greater than sign (>) is printed. The CPU hangs waiting for input until the return/enter key is pressed. The typewriter has no errors.

2.3 1621 Paper Tape Reader (PTR)

The paper tape reader (PTR) reads data from a disk file. The POS register specifies the number of the next data item to be read. Thus, by changing POS, the user can backspace or advance the reader.

The paper tape reader supports the `BOOT` command. `BOOT PTR` starts the standard paper tape boot sequence at location 0.

The paper tape reader implements these registers:

name	size	comments
POS	32	position in the input file

Error handling is as follows:

error	IOCHK	processed as
not attached	x	set RDCHK indicator, report error, stop
end of file	x	set RDCHK indicator, report error, stop
OS I/O error	x	set RDCHK indicator, report error, stop
parity error	1	set RDCHK indicator, report error, stop
	0	set RDCHK indicator

2.4 1624 Paper Tape Punch (PTP)

The paper tape punch (PTP) writes data to a disk file. The POS register specifies the number of the next data item to be written. Thus, by changing POS, the user can backspace or advance the punch.

The paper tape punch implements these registers:

name	size	comments
POS	32	position in the output file

Error handling is as follows:

error	IOCHK	processed as

```
not attached      x           set WRCHK indicator, report error, stop

OS I/O error      x           set WRCHK indicator, report error, stop

invalid char      1           set WRCHK indicator, report error, stop
                  0           set WRCHK indicator
```

2.5 1622 Card Reader/Punch (CDR, CDP)

The IBM 1622 card/reader punch is simulated as two independent devices: the card reader (CDR) and the card punch (CDP).

The card reader supports the `BOOT` command. `BOOT CDR` starts the standard card boot sequence at location 0.

The card reader reads data from a disk file, while the punch writes data to a disk file. Cards are simulated as ASCII text lines with terminating newlines. For each device, the POS register specifies the number of the next data item to be read or written. Thus, by changing POS, the user can backspace or advance these devices.

The card reader registers are:

```
name              size        comments

LAST              1           last card indicator
POS               32          position in the reader input file
```

The card punch registers are:

```
name              size        comments

POS               32          position in the punch output file
```

Card reader error handling is as follows:

```
error          IOCHK          processed as

end of file       x           set RDCHK indicator, report error, stop

not attached      x           set RDCHK indicator, report error, stop

OS I/O error      x           set RDCHK indicator, report error, stop

invalid char      1           set RDCHK indicator, report error, stop
                  0           set RDCHK indicator
```

Card punch error handling is as follows:

```
error       IOCHK processed as

not attached      x           set WRCHK indicator, report error, stop

OS I/O error      x           set WRCHK indicator, report error, stop

invalid char      1           set WRCHK indicator, report error, stop
                  0           set WRCHK indicator
```

2.6 1443 Line Printer (LPT)

The IBM 1443 line printer (LPT) writes its data, converted to ASCII, to a disk file. The line printer can be programmed with a carriage control tape. The LOAD command loads a new carriage control tape:

```
LOAD <file>                    load carriage control tape file
```

The format of a carriage control tape consists of multiple lines. Each line contains an optional repeat count, enclosed in parentheses, optionally followed by a series of column numbers separated by commas. Column numbers must be between 1 and 12; a column number of zero denotes top of form. The following are all legal carriage control specifications:

```
<blank line>                   no punch
(5)                            5 lines with no punches
1,5,7,8                        columns 1, 5, 7, 8 punched
(10)2                          10 lines with column 2 punched
1,0                            column 1 punched; top of form
```

The default form is 66 lines long, with column 1 and the top of form mark on line 1, and the rest blank.

The line printer registers are:

```
name              size         comments

LBUF[0:119]       7            line buffer
BPTR              7            buffer pointer
PCTL              8            saved print control directive
PRCHK             1            print check indicator
PRCH9             1            channel 9 indicator
PRCH12            1            channel 12 indicator
POS               32           position in the output file
CCT[0:131]        32           carriage control tape array
CCTP              8            carriage control tape pointer
CCTL              8            carriage control tape length (read only)
```

Error handling is as follows:

```
error          IOCHK           processed as

not attached      x            set PRCHK, WRCHK, report error, stop

OS I/O error      x            set PRCHK, WRCHK, report error, stop

invalid char      1            set PRCHK, WRCHK, report error, stop
                  0            set PRCHK, WRCHK
```

2.7 1311 Disk Pack (DP)

The disk pack controller supports 4 drives, numbered 0 through 3. Disk pack options include the ability to enable address writing (formatting).

```
SET DPn ADDROFF                set unit n address enable off
SET DPn ADDRON                 set unit n address enable on
```

Units can also be set ENABLED or DISABLED.

Unlike most simulated disks, the 1311 includes explicit representation for sector addresses. This is to support non-standard formats, such as the inclusion of the drive number in the sector address. As a result, 1311 sectors are 105 digits long: 5 address digits and 100 data digits. If the 1311 has not been formatted, the addresses are zeroes and are synthesized, if needed, based on the sector number.

The disk pack controller implements these registers:

```
name              size        comments

ADCHK             1           address check (compare error) indicator
WLRC              1           wrong length record check indicator
CYLO              1           cylinder overflow check indicator
ERR               1           disk error indicator
DPSTOP            1           disk check stop
```

Error handling is as follows:

```
error          DPCHK          processed as

not attached      x           set ERR indicator, report error, stop
```

1311 data files are buffered in memory; therefore, end of file and OS I/O errors cannot occur.

3 Symbolic Display and Input

The IBM 1620 simulator implements symbolic display and input. Display is controlled by command line switches:

```
-c                display as single character (alphameric for CPU and
                  DP, ASCII for others)
-s                display as flag terminated alphameric string (CPU and
                  DP only)
-m                display instruction mnemonics (CPU and DP only)
-d                display 50 characters per line, with flags denoted by
                  "_" on the line above
```

Input parsing is controlled by the first character typed in or by command line switches:

```
' or -c           character (alphameric for CPU and DP, ASCII for others)
" or -s           alphameric string (CPU and DP only)
alphabetic        instruction mnemonic (CPU and DP only)
numeric           octal number
```

Instruction input is free format and consists of an opcode and up to three operands:

```
op {+/-}ppppp{(idx)},{+-}qqqqq{(idx)},flags
```

The p address and, if present, the q address, are always decimal. A plus sign is ignored; a minus sign denotes indirect addressing (or a negative immediate operand). If indexing is enabled, addresses may be indexed; index registers are decimal numbers between 1 and 7. The flags field is used to set extra flags on the instruction. It consists of digit numbers in ascending order, with no separators. For example,

```
AM -12345(5),67890,110
```

translates into

```
 _ _ ___  _
111234567890
```

The flag over digits 3 and 5 specify the P index register; the flag over digit 6 specifies the P indirect address; the flag over digit 7 marks the end of the immediate Q operand; and the flags over digits 1 and 10 are specified by the third field.

4 Character Sets

The IBM 1620 uses single digits to represent numbers, and pairs of digits to represent characters (alphameric coding). Only a small number of the 256 possible alphameric codings have legitimate values. Further, the translation between alphameric and devices varied from device to device. The simulator implements a code called 1620 ASCII, which allows all 64 possible card codes to be represented by upper case ASCII characters. In addition, lower case alphabetic characters are accepted on input as equivalent to upper case. In the RN column, small "f" denotes the flag bit.

Card code	PT code	RA	RN	LPT WA	ASCII representation
<blank>	C	0	0	blank	blank
1	1	71	1	1	1
2	2	72	2	2	2
3	C21	73	3	3	3
4	4	74	4	4	4
5	C41	75	5	5	5
6	C42	76	6	6	6
7	421	77	7	7	7
8	8	78	8	8	8
9	C81	79	9	9	9
2 + 8	C82	?0A	A	na	^
3 + 8	821	33	B	=	= (or #)
4 + 8	C84	34	C	@	@ (or ')
5 + 8	841	70	0	0	:
6 + 8	842	?0E	E	na	>
7 + 8	C8421	?0F	F	na	{
12	XOC	10	0	+	+ (or &)
12 + 1	XO1	41	1	A	A
12 + 2	XO2	42	2	B	B
12 + 3	XOC21	43	3	C	C
12 + 4	XO4	44	4	D	D
12 + 5	XOC41	45	5	E	E
12 + 6	XOC42	46	6	F	F
12 + 7	XO421	47	7	G	G
12 + 8	XO8	48	8	H	H
12 + 9	XOC81	49	9	I	I
12 + 2 + 8	XOC82	?5A	?f+A	na	?
12 + 3 + 8	XO821	3	?f+B	.	.
12 + 4 + 8	XOC84	4	C	)	)
12 + 5 + 8	XO841	40	0	na	[
12 + 6 + 8	XO842	?5E	?f+E	na	<
12 + 7 + 8	XOC8421	5F	f+F	na	}
11	X	20	f+0	-	-
11 + 1	XC1	51	f+1	J	J
11 + 2	XC2	52	f+2	K	K

11 + 3	X21	53	f+3	L	L
11 + 4	XC4	54	f+4	M	M
11 + 5	X41	55	f+5	N	N
11 + 6	X42	56	f+6	O	O
11 + 7	XC421	57	f+7	P	P
11 + 8	XC8	58	f+8	Q	Q
11 + 9	X81	59	f+9	R	R
11 + 2 + 8	X82	5A	f+A	na	!
11 + 3 + 8	XC821	13	f+B	$	$
11 + 4 + 8	X84	14	f+C	*	*
11 + 5 + 8	XC841	50	f+0	-	]
11 + 6 + 8	XC842	?5E	?f+E	na	;
11 + 7 + 8	X8421	5F	f+F	na	_
0	O	70	0	0	0
0 + 1	OC1	21	1	/	/
0 + 2	OC2	62	2	S	S
0 + 3	O21	63	3	T	T
0 + 4	OC4	64	4	U	U
0 + 5	O41	65	5	V	V
0 + 6	O42	66	6	W	W
0 + 7	OC421	67	7	X	X
0 + 8	OC8	68	8	Y	Y
0 + 9	O81	69	9	Z	Z
0 + 2 + 8	O82	0A	A	na	|
0 + 3 + 8	OC821	23	B	,	,
0 + 4 + 8	O84	24	C	(	((or %)
0 + 5 + 8	OC841	60	0	na	~
0 + 6 + 8	OC842	0E	E	na	\
0 + 7 + 8	O8421	0F	F	na	"
				2	?
				12	!
				22	|
				32	0
				35	:
				36	blank
11 + 0		50		-	`