Computer History Museum

IBM 1620 Jr.
Project Status
Report #4 (4/30/2018)

This is the fourth status report on the IBM 1620 Jr. project. An overview of the project is detailed in *IBM 1620 Jr. Project Description, Version 1.3 (1/12/2018)*. The project is structured as several sub-projects, each with a set of volunteers working on it. Note that this is only a summary of the work done to date. Details are available in email & data files.

General (Team)

After VCF West in August 2017, work on the project switched focus to the Console Typewriter.

The team is committed to demonstrating this year's accomplishments, especially the Console Typewriter, at VCF West 2018 in August. The current plan is to run the historic BBC Baseball Simulation program plus at least one interactive game that makes good use of the Console Typewriter. The simulator will be extended to support console typewriter I/O, all internal options, and the remaining, non-floating-point instructions.

Lee Courtney is not able to contribute to the project due to family and work commitments. Dave Babcock has taken over the Software Library sub-project.

Front Panel (Steve Casner, Joe Fredrick)

The front panel sub-project consists of all physical and electrical work done on the spare IBM 1620 front panel, the Raspberry Pi 3 (RPi), and the interface circuitry connecting the two.

There are a few items remaining to complete the front panel and cabinet:

- Acquire a "new stock" incandescent light and repeat the brightness calibration work. One of the two original manufacturers of bulbs for the IBM 1620, Eldema, still makes the same light (#CF03-CCB-1869). Since incandescent lights can fade due to use and the original lights in the spare front panel being used seem dimmer than the team remembers, it is worth investigating further.
- Label the added indicator lights and switches. This will most likely be done with press-on lettering. Silk screening would require disassembly of the entire front panel – a tedious task now that the panel is completely wired.
- Apply Loctite 271 to the threads of the brass toggle switch handles.
- Repair the lower front right corner of the wooden case.
- Add an internal latch to prevent the front panel from being opened when unattended.
- Add a hinged, lockable, plexiglass back to protect the internals, but allow viewing.
- Add external connectors for power, USB, etc. to the back of the case.
- Add folding handles to the sides of the case, so that it can be carried safely.
- Paint the case metallic gray or add a metal veneer to better match the real IBM 1620 cabinet.
- Make a more permanent "IBM 1620 Data Processing System" sign mounted on the top of the cabinet to replace the temporary foam core version.
- Mount a powered speaker, already purchased, inside the cabinet for playing the machine sounds of the IBM 1620.

Console Device (Dave Babcock, Joe Fredrick, Dag Spicer)

The console device sub-project consists of all hardware and firmware work done on the console I/O device, also known as the Console Typewriter.

Unlike other computers, the IBM 1620's console typewriter was an integral part of the machine, not an [optional] peripheral device. It was not possible to bootstrap programs from paper tape or the disk without it. It was the only means of examining or depositing into core memory. Therefore, a physical teleprinter, not a virtual one, with impact printing must be included to realistically recreate the experience of running the IBM 1620. Balancing the objectives of the IBM 1620 Jr. project, the IBM Wheelwriter 1000 electronic typewriter was chosen. The selection criteria are described in: *IBM 1620 Jr. Console Typewriter Proposal, June 14, 2017*.

It is somewhat fitting that the IBM 1620 Model 1 used IBM's first generation electric typewriter (Executive Model B), the IBM 1620 Model 2 used their second generation electric typewriter (Selectric), and the IBM 1620 Jr. [an IBM 1620 Model 3?] will use their third generation electronic typewriter (Wheelwriter).

The Wheelwriter typewriters had an optional interface which allowed a computer to output to the typewriter, but not use it as an input device. To create a true console typewriter, the project team is designing an interface board which will mount inside the Wheelwriter, electrically interposed between the typewriter's keyboard and motherboard, and communicate with the IBM 1620 Jr. via a bi-directional USB interface. A Teensy 3.5 microcontroller on the interface board will be programmed to make the Wheelwriter operate like the IBM 1620 Console Typewriter.

The interface board will scan the keyboard and process key presses. It will simultaneously generate appropriate [phantom] key presses to the motherboard. The motherboard will still handle the complexities of the print mechanism. The firmware will ignore some key presses or locally echo others and send them to the IBM 1620 Jr., depending on which 1620 input instruction is being executed. [The IBM 1620 Console Typewriter was a half-duplex device.] The simulator will send output character to the interface for printing when executing output instructions. To replicate the printed look of the IBM 1620 special characters, the firmware will use a sequence of over-printed characters, backspacing, and carriage up/down movements.

The work done to-date, in progress, and planned on the Console Typewriter includes:

1. Two Wheelwriter 1000 typewriters were bought for conversion – a primary unit and a spare. A spare keyboard and other supplies (ribbons, continuous form paper, covers) were also purchased. The ribbons and paper are readily available from Staples.

2. The Artisan 10 printwheel was selected as the closest match to the IBM 1620 Console Typewriter's font and three were purchased.

3. A breakout board was made and interposed between the keyboard and motherboard of one of the typewriters for experimentation and prototyping.

4. Dag Spicer located within IBM, borrowed, and scanned a maintenance manual on the typewriter. The project also has the User Manual but have yet to locate a schematic or "principles of operation" manual for the device.

5. Many experiments and measurements were done using the breakout board, test circuitry, and logic analyzers to reverse-engineer the interface between the typewriter's keyboard and motherboard. All of the findings were carefully documented.

6. The keyboard scanning is more complex than expected. The keyboard scan matrix (14 columns x 8 rows) does not map directly to the physical keyboard layout. The motherboard constantly scans the column lines one at a time & out of order and reads the corresponding row lines. In a complete

cycle, each of the 14 column scan pulses is dynamically either 820 usec or 3.68 msec wide, depending on key presses. If any key within a column is in a different state (depressed or released) than it was during the same column's scan in the last cycle, then the column pulse is long. Otherwise, the column pulse is short, which is the usual case. The motherboard samples the row lines in the first 265 usec of a column scan to determine if the scan pulse should be stretched. The row lines of an extended column scan are read a second time to determine their "true" state between 2.16 msec and 3.055 msec into the scan pulse. Selectively wide column scan pulses and delayed row reading is how the motherboard "debounces" key presses and releases.

7. A preliminary, detailed design of the Console Typewriter interface board was done and documented in: *IBM 1620 Jr. Wheelwriter Console Typewriter Technical Specification, Revision 1.0 (12/31/2017)*.

8. The interface board was breadboarded, initial firmware written, and keyboard input / printer output tests were successfully run. In one set of tests, each keyboard key was pressed, locally printed, and sent out via the USB port. In another set of tests, data sent to the Wheelwriter via the USB port was read and printed. A final set of tests did simultaneous keyboard input, USB sending, USB receiving, and printing. Some of the keyboard keys result in multiple printed characters to approximate IBM 1620 special characters.

9. The wire protocol between the simulator and the Console Typewriter was developed and documented in: *IBM 1620 Jr. Console Typewriter Protocol, Version 1.1 (5/19/2018)*.

10. Firmware development continues implementing the communication protocol, supporting IBM 1620-specific functionality, and squashing bugs.

11. A standalone test program in Java is being written to support the Console Typewriter development.

12. Once there is a fully functional [breadboarded] prototype, a circuit board can be laid out, fabricated, assembled, and mounted in the Wheelwriter typewriters.

13. To better match the real IBM 1620 Console Typewriter's keyboard, some of the keycaps on the Wheelwriter typewriters will be relabeled.

14. The input and output speed of the original IBM 1620 Console Typewriter was 10 cps (120 wpm). The IBM 1620 Jr. Console Typewriter interface board will not limit input speed and the simulator will control output speed by sending characters to the typewriter at a 10 cps rate. A typical touch typist averages 40 wpm (3.34 cps), well below the 1620's input limit. The Wheelwriter is capable of printing at 16 cps, so there won't be a problem printing single characters at the 10 cps rate. However, special IBM 1620 characters will require overprinting and carriage motion of 3 - 16 "characters" (flagged group mark), reducing the effective print speed. It is hoped that an average print speed of 10 cps can be maintained, with occasional bursts to 16 cps, for most output instructions' character mixes. The exception might be the Dump Numeric (DNTY) instruction which can print a high volume of zeroes. The IBM 1620 Console Typewriter printed a slashed zero. This requires 3 "characters" to be printed by the Wheelwriter at an aggregate 5.34 cps rate. To deal with this, a "slash zeroes / don't slash zeroes" setting has been defined in the communication protocol, so the simulator can [optionally] select between authentic looking printout and authentic print speed.

15. The IBM 1620 Console Typewriter allowed the operator to mechanically set and clear margins and tabs – the computer was unaware of where these were. Furthermore, the operator could trigger a local tab or carriage return at any time, even when the typewriter was "locked". The 1620 never knew which column the carriage was at and currently the simulator doesn't care either. However, an

aspirational goal is for the simulator to [optionally] display realistic-looking typewriter output on an attached video monitor. It needs to track the current print position to do that. The communication protocol defines a way for the typewriter to notify the simulator of local tabs and returns, but it does not define how to deal with margins and tab stop settings. More planning is needed in this area.

Card Read-Punch Device (Dave Babcock)

The card read-punch device sub-project consists of emulating the physical punched card device in the software simulator using USB memory sticks to represent individual card decks.

Development of the card read-punch device has not begun.

Simulator (Dave Babcock, Steve Casner)

The simulator sub-project consists of developing a new IBM 1620 cycle-level simulator, coded in C, that interfaces with the front panel hardware. It will simulate an IBM 1620 Model 1 Level F punched card system with 60,000 digits of memory and the automatic divide, indirect addressing, MF/TNF/TNS, and floating-point processor options.

The following instructions and options are being implemented for VCF:

- Add/Subtract/Compare sign control
- Indirect Addressing
- Multiply (M – 23), Multiply Immediate (MM – 13)
- Load Dividend (LD – 28), Load Dividend Immediate (LDM – 18)
- Divide (D – 29), Divide Immediate (DM – 19)
- Move Flag (MF – 71)
- Transmit Numeric Strip (TNS – 72)
- Transmit Numeric Fill (TNF – 73)
- Console Typewriter I/O (SPTY – 34.1, RCTY – 34.2, TBTY – 34.8, DNTY – 35.1, RNTY – 36.1, RATY – 37.1, WNTY – 38.1, WATY – 39.1, BI – 46.6/7, BNI – 47.6/7, console INSERT)

In addition to implementing the above items, operating sounds need to be added to simulator. David Wise recorded sounds of his IBM 1620 in operation that the simulator can play back as appropriate. David's machine also answered the open questions about initial, power-up register contents. A few minor changes need to be made to the simulator to match.

The project has IBM diagnostic programs for all of these implementation items, except the Console Typewriter, to validate the simulator.

The simulator currently includes a front panel light/switch/button test as well as two light demos. A set of Console Typewriter tests are being added.

To consolidate all of the different IBM 1620 character codes, a single, "Rosetta Stone" document was produced – *IBM 1620 Character Codes, Version 1.3 (4/20/2018)*.

The following instructions and options will be implemented later in the project:

- Floating Add (FADD – 01)
- Floating Subtract (FSUB – 02)
- Floating Multiple (FMUL – 03)
- Floating Shift Left (FSL – 05)

- Transmit Floating (TFL – 06)
- Branch and Transmit Floating (BTFL – 07)
- Floating Shift Right (FSR – 08)
- Floating Divide (FDIV – 09)
- Punched card I/O (DNCD – 35.4, RNCD – 36.5, RACD – 37.5, WNCD – 38.4, WACD – 39.4, BI – 46.6/7/9, BNI – 47.6/7/9)

Unfortunately, the project does not have IBM diagnostic programs for the floating-point instructions nor any of the I/O devices.

The following instructions and options are not part of the IBM 1620 Jr. project as currently defined, but may be implemented as follow-on project(s):

- Paper tape I/O (DNPT – 35.2, RNPT – 36.3, RAPT – 37.3, WNPT – 38.2, WAPT – 39.2, BI – 46.6/7, BNI – 47.6/7)
- Plotter I/O (DN – 35.2, WN – 38.2, WA – 39.2, BI – 46.7, BNI – 47.7)
- Disk I/O (SK – 34.1, RDGN – 36.0, WDGN – 38.0, CDGN – 36.1, RTGN – 36.4, WTGN – 38.4, CTGN – 36.5, RDN – 36.2, WDN – 38.2, CDN – 36.3, RTN – 36.6, WTN – 38.6, CTN – 36.7, BI – 46.36/37/38/39, BNI – 47.36/37/38/39)
- Printer I/O (DN – 35.9, WN – 38.9, WA – 39.9, BI – 46.25/33/34/35/42, BNI – 47.25/33/34/35/42)

Software Library (Dave Babcock)

The software library sub-project consists of converting all of the museum's IBM 1620 software collection to a format usable by the IBM 1620 Jr. (and the IBM 1620 SIMH simulator), assembling the associated documentation, and testing the programs on the IBM 1620 Jr.

CHM's IBM 1620 software collection falls into these groups:

1. The John Maniotes Collection - four file cabinets of 1620 software on punched cards. This is a large portion of the IBM 1620 General Program Library, IBM Users' Group (COMMON) contributed software, programs written at Purdue, sample data, sample output, etc. The ~2000 card decks were read as part of the original IBM 1620 Restoration Project in a multi-deck, printable-ASCII, column-binary format. The 82 encoded files were never post-processed into an organized, useable format.

2. The David Wise Collection – several dozen binary paper tape images of software for David's IBM 1620. These files do not have to be converted as they are already in a useable format. This is the source of the IBM 1620 Diagnostic programs being used with the IBM 1620 Jr.

3. Miscellaneous card decks from multiple donors. These were read in a single-deck, little-endian numeric, column-binary format. No post-processing of these decks has been done.

4. Other IBM 1620 software from various sources in mixed formats. These need to be dealt with individually and converted to a useable format.

An extraction program, written as part of the IBM 1620 restoration project, was recently updated and used to extract the BBC Baseball Simulation program from the Maniotes collection. The baseball program has been successfully run on SIMH and will be run on the IBM 1620 Jr. once the Console Typewriter is operational.

The extraction program was modified to deal with the format of #3 (miscellaneous card decks) and used to extract several game programs. These programs are being run on SIMH and evaluated for use at VCF West

2018. Also being considered is a poker program contributed by Ronald Rock that he wrote in high school. These programs also need an operational Console Typewriter to run on the IBM 1620 Jr.

Since there won't be support for punched card or paper tape devices in the simulator for VCF, a conversion program was written to translate binary paper tape and SIMH dump format files into the simulator's cmem (core memory) format. The baseball and game programs have been converted.

The work to process the entire software library is still to be done.

Operations Guide (Dave Babcock)

The operations guide sub-project consists of writing a brief document on the operation of the IBM 1620 Jr.

Notes for the Operations Guide are being kept as the simulator is being developed.

Website (Dave Babcock, Team)

The website sub-project consists of selecting, configuring, and populating a dedicated website on the IBM 1620 which will document the IBM 1620, the CHM IBM 1620 restoration project, and the IBM 1620 Jr. project.

Development of the website has not started, but all project emails and documentation are being saved.