# IBM 1620 Jr.
# Project Status

Report #6 (10/16/2020)


This is the sixth status report on the IBM 1620 Jr. project. An overview of the project is detailed in *IBM 1620 Jr. Project Description, Version 1.4 (9/12/2018)*. 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.

It's been almost two years since the last Status Report and much has been accomplished. A "Punch List" outlines the steps remaining to complete the project and turn it over to the Computer History Museum. The team hopes to finish by Summer 2021.

## General (Team)

The IBM 1620 Jr. Project is a part-time, all-volunteer effort which has a plan, but not a defined schedule. The core team is also geographically dispersed, which complicates joint development a bit. In addition to other interruptions, the team took a 1-year side trip to "productize" the console typewriter into a general-purpose, printing, computer terminal – Cadetwriter. Several vintage computer hobbyists have successfully built their own Cadetwriters.

In the last Status Report an opportunistic goal was to participate in the annual Baseball Hack-A-Thon. The idea was to have a team of baseball enthusiasts hack the historic IBM 1620 Baseball Simulation program to add new features and then run it on the IBM 1620 Jr. The San Francisco regional Hack-a-Thon didn't take place and a team at another site couldn't be formed in time.

The public repository for all project files is at: https://github.com/IBM-1620 The team is releasing all technical material of the project into the public domain under GPLv3. This includes schematics, source code, and related documentation. There is also a public forum for discussing and sharing Cadetwriter information at: https://cadetwriter.slack.com

## 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. It also includes all work to be done on the wooden case supporting the front panel.

Most of the work on the front panel was completed early in the project and has been operating without any problems for over 40 months.

A source of custom metal nameplates was found. Using close-up photographs of the only still running IBM 1620 owned by team member David Wise, a proper nameplate was purchased for the IBM 1620 Jr.

**Console Typewriter** (Joe Fredrick, Steve Casner, Dave Babcock)

The console typewriter sub-project consists of all physical and electrical work done on the console I/O device.

As reported in the last status report, the team demonstrated an operational prototype of the console typewriter at VCF West 2018. It had a v1.6 interface board with v3R4 firmware and a USB interface. The most frequently asked question at the show was: "Can I connect the console typewriter to my XYZ computer?". The team took the question seriously and spent most of the next year upgrading and finalizing the serial interface board and firmware to be a general-purpose terminal named Cadetwriter.

The design of the console typewriter's serial interface board continued to evolve through three production versions:

1.6 The first production board in June 2018. It had a USB interface with an external USB-B type connector and RS-232 serial on internal solder pads. The RS-232 serial port was at TTL levels and required an external level-shifter to produce a standard RS-232 interface. The firmware available at the time supported only the USB interface, although later versions supported both interfaces. It was named the "Wheelwriter USB Interface Board Rev. 1.6".

A detailed pictorial assembly guide was created which describes how to modify a Wheelwriter 1000 typewriter and turn it into a computer terminal for about $150 in parts, plus the typewriter, and 4 hours of work. The board could be ordered from OshPark and several hobbyists bought boards and successfully created their own computer terminals. This board was retired when the Rev. 1.7 board was released, but it is still supported by the latest firmware.

During firmware development the board was found to have a design error. When new firmware is downloaded and the Teensy 3.5 microcontroller is rebooted, the output lines going to the Wheelwriter's logic board are floating and random character(s) are sometimes printed. The solution was to add two pull-down resistor packs to the underside of the interface card.

1.7 The second production board in May 2019. There were no functional changes, but a more robust solution to the floating line boot problem was implemented. The output lines going to the Wheelwriter logic board are now kept actively unasserted until properly initialized and enabled by the firmware via a previously unused GPIO pin. It was named the "Wheelwriter USB Interface Board Rev 1.7".

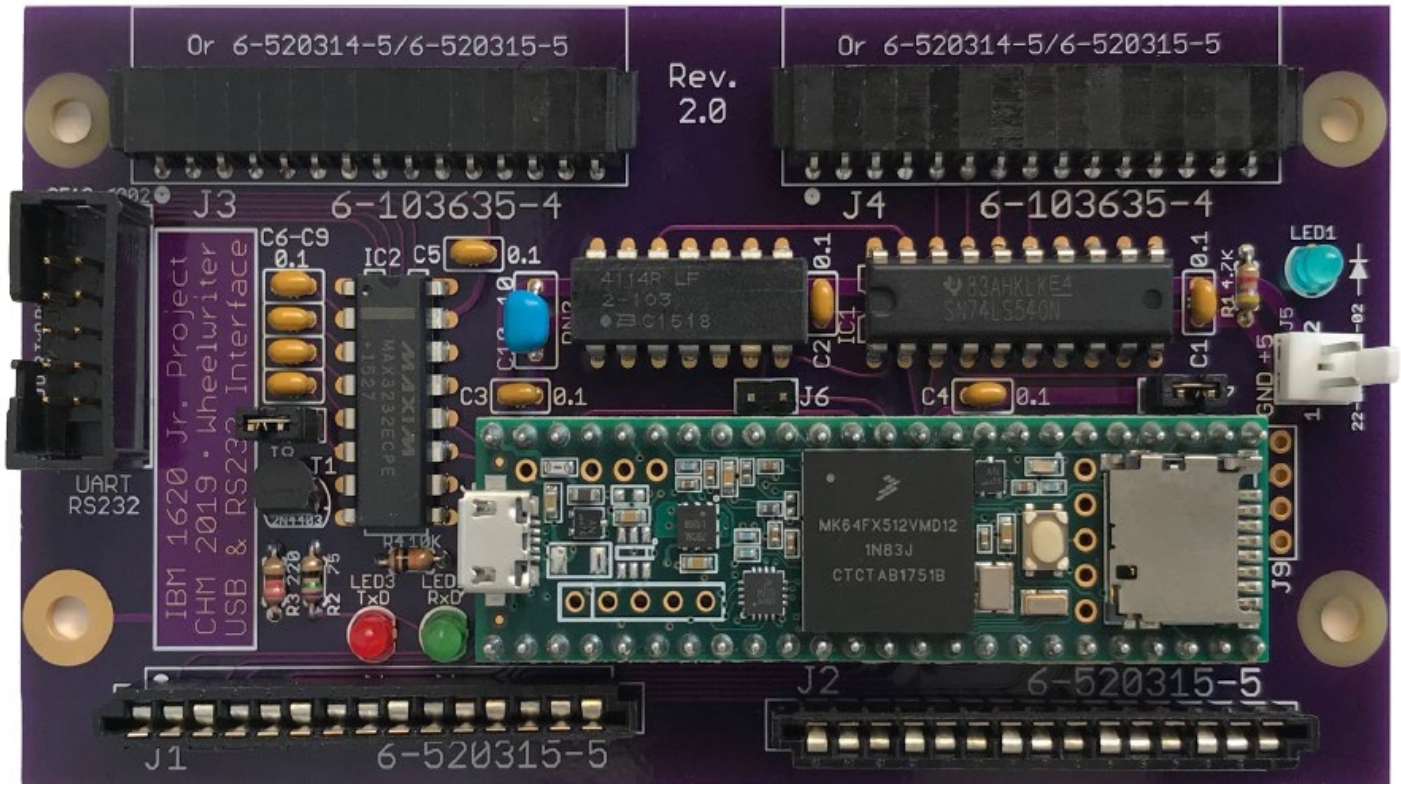
The cable connectors were also improved. Installing the interface board involves moving the two keyboard cables from the typewriter's logic board to the interface board and adding two new cables from the interface board to the typewriter's logic board. When the Rev. 1.6 board was created, proper LIF connectors for the keyboard cables could not be found. So, installation required that two 15-pin connectors on the typewriter's logic board be carefully removed, soldered on the interface board, and replaced by two new latching connectors on the logic board. Unsoldering the connectors requires skill and the modified typewriter logic board is non-standard and cannot be sent in for repair. When the Rev. 1.7 board was designed, compatible LIF connectors were found and installation no longer requires replacing the two connectors on the typewriter's logic board.

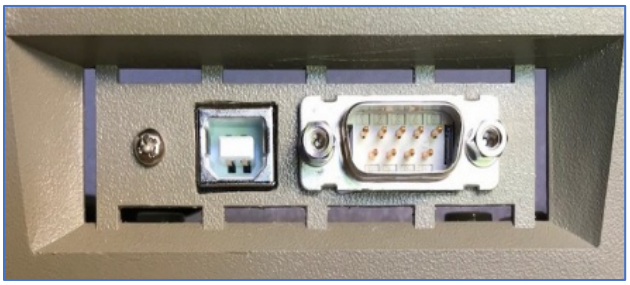
The assembly guide was updated and the new board was uploaded to OshPark. Again, several hobbyists ordered boards and were successful in building terminals. This board was retired when the Rev. 2.0 board was released, but it is still supported by the latest firmware.

2.0 The latest and final production board in July 2019. This update fully supports the RS-232 interface as well as the USB interface. Changing the board layout allowed a MAX3232 RS-232 level shifter IC to be added. Connectors for both USB (USB-B) and RS-232 (DE-9) are installed in the rear of the typewriter. Several Teensy GPIO pins were reassigned, but the firmware can automatically detect the board revision and fully support versions 1.6, 1.7, and 2.0. It was named the "Wheelwriter USB & RS232 Interface Board Rev. 2.0".

The assembly guide (*Cadetwriter: IBM/Lexmark Wheelwriter 1000 Teleprinter Adaptation*) was again updated and is available at: https://github.com/IBM-1620/Cadetwriter/blob/master/docs/adaptation_guide/wheelwriter-adaptation-instructions.pdf

The new circuit board is available on OshPark at: https://oshpark.com/shared_projects/TIkAO4NE Again, several other people have successfully built their own Cadetwriters using this board. In at least three cases, a different version of the Wheelwriter was used. It sometimes requires installing the interface board in a slightly different location to avoid a tilting keyboard and still fit under the print carriage. Later models of the Wheelwriter are capable of printing up to 20cps. In addition, some models have additional keyboard keys and wider carriages. Compile options when building the firmware support these variations.



To mimic the IBM 1620 Console Typewriter, 24 custom keycaps were needed. Likewise, a few replacement keycaps were needed for the ASCII Cadetwriter. The original company that made keycaps for the Wheelwriter typewriters is still in business, selling a variety of keycap sets, and will make custom keycaps for a reasonable price.

Here is an unmodified Wheelwriter 1000 keyboard:


This is the IBM 1620 Jr. Console Typewriter keyboard:

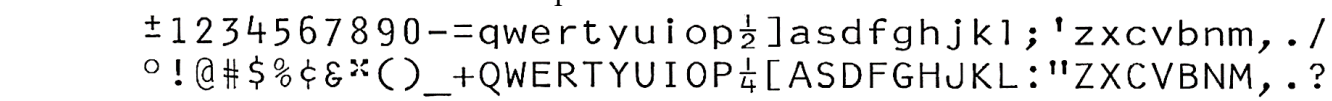

Here is the Cadetwriter keyboard:

Another aspect of the typewriter that was completed in the past 2 years was the printed output. The Wheelwriter typewriter has a "Daisy Wheel" print mechanism with 96 characters on the printwheel. The IBM 1620 requires some special characters (slashed zero, record mark, group mark, flagged digits, release-start, and invalid character) that are not on any printwheel. To support the full ASCII printing character set, Cadetwriter also requires some characters (less than, greater than, back slash, caret, back apostrophe, left brace, vertical bar, right brace, and tilde) not on any printwheel. It is not feasible to create custom printwheels, so two techniques were used to synthesize the missing characters. Note that there are alternate, mostly foreign language printwheels available, but none of them have all of the needed special characters either. The English Artisan 10 printwheel (#1353520) was selected because its font most closely matches the IBM 1620 Console Typewriter's font.

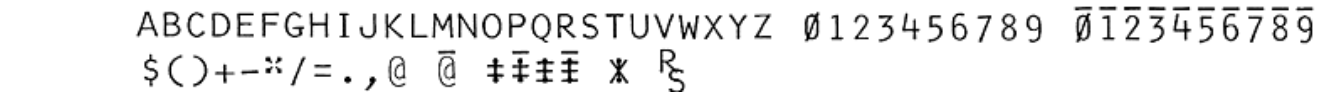
A sequence of overprinted characters and carriage movements were used for the IBM 1620 special characters. For example, when printing a slash zero, a zero is printed, the carriage backs up one character position, and a slash is printed. For a flagged digit, the carriage is rolled down a ½ line, a hyphen is printed, the carriage if rolled back up ½ line & backspaced one character position, and the digit is printed. This slows down the effective print speed, but the output looks similar to the real IBM 1620 Console Typewriter's output. Two factors help to minimize the impact – the infrequent use of these special characters and the IBM 1620's typewriter output rate is 10cps while the Wheelwriter prints at 16cps, allowing it to catch up.

Most of the missing ASCII characters for Cadetwriter can't be produced by just overprinting existing characters. Fortunately, the Wheelwriter has the capability of moving the carriage 1/6th of a character position left & right and 1/8th of a line up & down. Combining these micro carriage movements with printing periods, the missing characters can be created (dubbed "period graphic characters"). This slows down the effective print speed, but the impact is limited due to the relatively low occurrence of these special characters

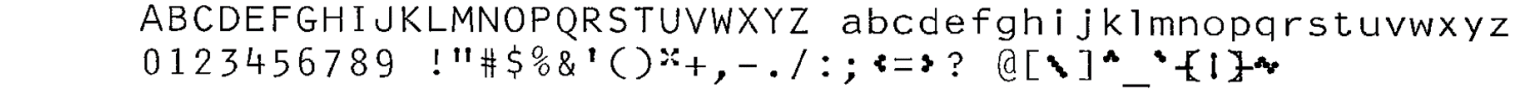
Here is the Wheelwriter's native print character set:




This is the IBM 1620 Jr.'s print character set:

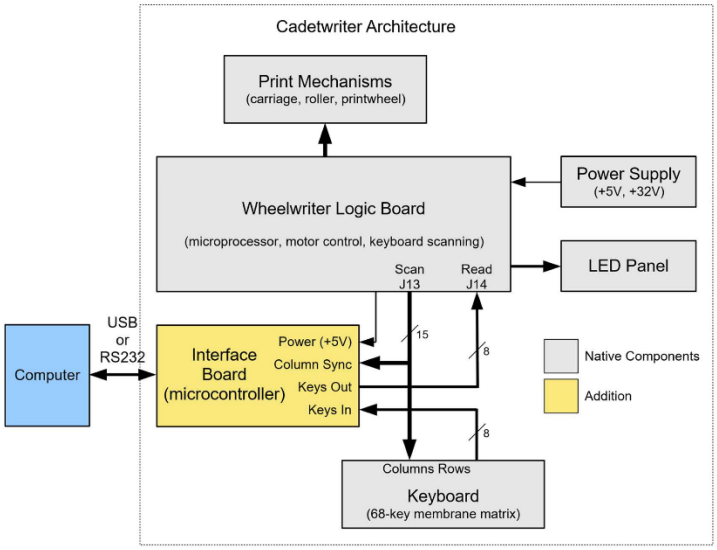



Here is the Cadetwriter's print character set:




The initial firmware for the typewriter began as a diagnostic tool to explore the typewriter's operation and timing. As more was learned, the software kept changing and growing. It reached the point where it was able to control most of the functions of the console typewriter, but with occasional timing and printing problems. It was decided that the best approach would be to start over with a new design that incorporated the learned intricacies of the Wheelwriter and all of the planned additional features.

This is the architecture of Cadetwriter. An early design decision was to interpose an additional circuit board between the keyboard and logic board rather than completely replace the logic board. With no documentation on the print mechanism, reproducing the full printer control by reverse-engineering the logic board would be problematic. It should be much easier to inject fake key presses and let the typewriter's logic board handle all of the printing chores. Reading a scanning keyboard should also be straightforward. Both turned out to be more complex than expected and greatly added to the development time and effort.




The biggest issue was the logic board's complex column scanning. The width of the column scan pulse varies depending on the situation. When there are no changes to the state of the keyboard, the pulse is a consistent width. When a key is first depressed or released, its column pulse is lengthened, most likely to deal with contact bounce. If keys are pressed too quickly, the pulses are longer. If the input buffer becomes full, the typewriter beeps and the current pulse is stretched until the buffer empties. If too many micro carriage movements occur in a short period of time [the exact conditions haven't been determined], then there is more beeping and the column pulse can be seconds long. The 16cps print mechanism can keep up with a fast typist (up to 190 wpm), but not computer output. The columns are scanned in a consistent order except when multiple keys are pressed simultaneously (i.e.: shift A), then previous columns are sometimes re-scanned out-of-order to handle n-key rollover. As a typewriter, the logic board directly reads the keyboard state. As a terminal, the injected keys seen by the logic board are almost always different from what's being typed on the keyboard, so the short-long column pulses are out-of-sync with key presses. In an extreme case, key presses can be lost. The interface board's firmware can detect when these column scan anomalies are occurring, but not when they are about to happen to take corrective action. This complexity is what led to the new firmware implementation.

There have been ten incremental releases of the new firmware over the past 1.5 years. Each one adding support for new features such as the stand-alone typewriter, RS-232 interface, ASCII version (Cadetwriter), slow serial, recognition of ANSI escape sequences, other Wheelwriter models, etc. It is almost 10,000 lines of C code, structured as an Arduino Sketch with multiple interrupt service routines.

The IBM 1620 Jr. Console Typewriter development work was finished in June 2019. Two console typewriters based on Wheelwriter 1000 typewriters are complete with version 1.7 serial interface boards (USB interface), at least version 5R3 of the firmware, and custom keycaps. All of the work since has been to create and support Cadetwriter.

There are several areas of specialization / customization available in Cadetwriter – board detection, emulation selection, defines, data tables, and dynamic settings.

While only the latest Rev. 2.0 serial interface board is currently available, several early adopters and the IBM 1620 Jr. itself, have older production versions. The firmware fully supports all three versions of the board by auto-detecting the board revision and adapting as needed.

The firmware supports four different emulations – stand-alone Wheelwriter typewriter, IBM 1620 Jr. Console Typewriter, ASCII terminal, and an open slot for a hobbyist-defined emulation. Each emulation

has a different keyboard layout, printing character set, special key functions, and communication protocol. These are selected by the placement of two hardware jumpers on the serial interface board:

| J6 Jumper | J7 Jumper | Emulation |
|---|---|---|
| | | IBM 1620 Jr. Console Typewriter |
| | X | Cadetwriter ASCII Terminal |
| X | | <available for future emulation> |
| X | X | Standalone Wheelwriter Typewriter |

Different models of the Wheelwriter typewriter have varying characteristics such as carriage width, print speed, number of keys, placement of keys, etc. Several models are already defined in the code. To add a new model, its characteristics just need to be specified in a set of compile-time "#define FEATURE_xyz" statements and the firmware recompiled.

Much of the logic of the Cadetwriter is contained in FSA data tables. The tables maintain the state of the emulation and define the character set, the keyboard layout, the functioning of each key, how to print each character and its timing, what characters to send, what to do when each character is received, etc. Customization for a new emulation should mostly involve cloning and/or modifying a set of these tables.

Both the IBM 1620 Jr. Console Typewriter and ASCII Cadetwriter emulations have dynamic configuration settings stored in EEPROM that the user can change. Pressing Control-Setup on either machine brings up a setup menu that has options to print the character set, view/change settings, and report/clear errors & warnings. Pressing Control-Shift-Setup resets all setting to their "factory" default.

Here are the IBM 1620 Jr. Console Typewriter settings:

```
---- Cadetwriter: IBM 162Ø Jr. (v5R1Ø) Setup

Command [settings/errors/character set/QUIT]:                    character set

  ABCDEFGHIJKLMNOPQRSTUVWXYZ Ø123456789 Ø̄1̄2̄3̄4̄5̄6̄7̄8̄9̄ $()+-*/=.,@ @̄ ‡‡‡‡ ¤ ₨

Command [settings/errors/character set/QUIT]:                    settings

  record errors [TRUE/false]:                                    true
  record warnings [true/FALSE]:                                  false
  batteries installed [TRUE/false]:                              true
  slash zeroes [TRUE/false]:                                     true
  bold input [true/FALSE]:                                       false
  line offset [Ø-1Ø, 1]:                                         1

  Save settings [yes/NO]?                        no

Command [settings/errors/character set/QUIT]:                    errors

  No errors or warnings.

Command [settings/errors/character set/QUIT]:                    quit

---- End of Setup
```

These are the settings for the ASCII Cadetwriter:

```
Command [settings/errors/character set/QUIT]:                    settings

  record errors [TRUE/false]:                                    true
  record warnings [true/FALSE]:                                  false
  batteries installed [TRUE/false]:                              true
  serial [usb/RS232]:                                            rs232
  duplex [half/FULL]:                                            full
  baud rate [50-230400, 9600]:                                   9600
  dps [7o1/7e1/8N1/8o1/8e1/8n2/8o2/8e2]:                         8n1
  sw flow control [none/XON_XOFF]:                               xon_xoff
  hw flow control [NONE/rts_cts/rtr_cts]:                        none
  uppercase only [true/FALSE]:                                   false
  auto return [TRUE/false]:                                      true
  transmit end-of-line [CR/crlf/lf/lfcr]:                        cr
  receive end-of-line [cr/CRLF/lf/lfcr]:                         crlf
  escape sequences [none/IGNORE]:                                ignore
  line length [80-165, 80]:                                      80
  line offset [1-10, 1]:                                         1

  Save settings [yes/NO]?                          no
```

In preparing for VCF West 2019, a Cadetwriter was connected to a replica of the Altair 8800 [Altair-duino]. Everything worked correctly except there were occasional strings of unexpected characters printed. The emulated computer assumed it was connected to a CRT terminal and the characters were ANSI Escape Sequences. A temporary patch to the firmware eliminated the stray characters. Since the show, code was added to recognize and ignore all generic and vendor-specific escape sequences that adhere to the ANSI standard.

It was discovered at Cadetwriter's debut that the Teensy 3.5 microcontroller being used cannot properly deal with slow baud rates. This is due to its very fast system clock and not enough bits in the internal UART divider to support rates below 1200 baud. Slowing down the system clock was not an option because the firmware would run too slowly for the terminal to function correctly. The specs for Cadetwriter were changed to reflect the supportable speeds.

Paul Williamson, a vintage computer enthusiast who was the first person, besides the team, to build their own Cadetwriter, volunteered to resolve the slow baud rate problem. He had previously helped with testing the Wheelwriter's semi-automatic paper loading feature, column scan timing, and dumping the Wheelwriter ROM. He developed a new support library, SlowSoftSerial, which handles slow baud rate I/O by bit sampling and bit banging triggered by two hardware interval timers. Cadetwriter's firmware automatically uses Paul's library for 600 baud and below and the hardware UART for faster serial rates. Paul also designed handy snap-in paper guides for the Wheelwriter that can be 3D printed. The model is available here: https://www.thingiverse.com/thing:4079509

Chris Coley, another collector, was the first person to build a Cadetwriter using a different model of Wheelwriter (1500). He helped with testing hardware flow control, wide carriage support, and 20cps printing. His Cadetwriter, built from a wide-carriage Wheelwriter 1500, has a tractor feed.

The team was excited to extend the project and create something that is beneficial to the larger community of vintage computer hobbyists. A lot of effort was put into making it easy to replicate and modify. In addition to the detailed assembly guide, a *Cadetwriter User Guide* was written (https://github.com/IBM-1620/Cadetwriter/tree/master/docs/user_guide) and the code contains exhaustive comments. Several people have successfully built about a dozen Cadetwriters and several of them have contributed back to project. Before their demise, The Living Computer Museum expressed interest in building several for

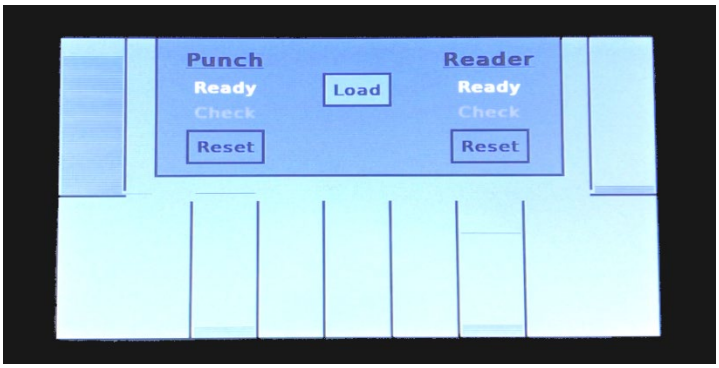
various uses. Cadetwriter was featured by Hack-A-Day: https://hackaday.com/2019/11/14/upgrade-board-turns-typewriter-into-a-teletype/ and was written up in The New Stack: https://thenewstack.io/blast-off-into-the-past-an-apple-1-demo-at-the-vintage-computer-festival-west/

## Card Read-Punch Device (Dave Babcock)

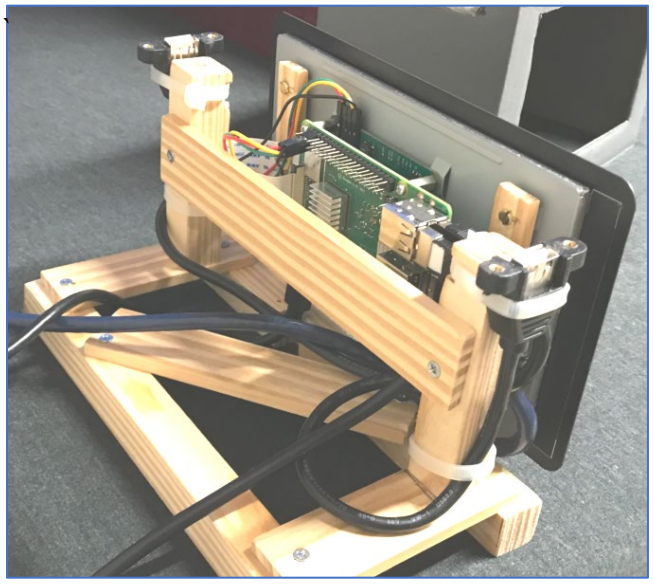
The card read-punch device sub-project consists of emulating the IBM 1622 punched card unit so that IBM 1620 programs can be "read" and "punched". The majority of the IBM 1620 Software Library that the Computer History Museum has is card based.

Development work on the card read-punch device began in mid-May 2020.

A prototype device was built with a Raspberry Pi 3B+, a Raspberry 7" Touch Screen Display, and assorted cables. Code was written to emulate card reading / punching with simple animated graphics and connect with the IBM 1620 Jr. via USB. Code was added to the simulator to implement the punched card I/O instructions and communicate with the card device. The system was tested and debugged using the two IBM card I/O diagnostics, IO02 and IO03 and the SPS assembler.

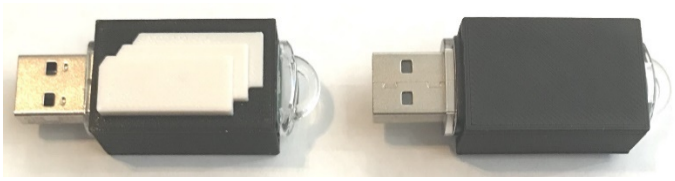



A temporary physical case for the prototype was built in two parts – an inner wooden framework holding all of the hardware and a cardboard shell. It is a 1:7 scale model of the actual IBM 1622 Card Read-Punch. It was complete and functional on June 30th, just in time to be filmed for VCF West 2020.



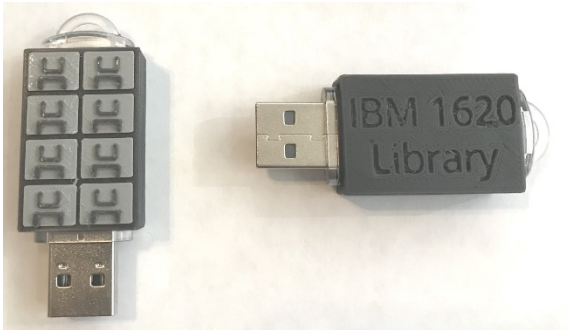

The prototype used the Raspberry 7" Touch Screen Display, but the final version will use a different panel (EVICIV 7" Touch Screen Display). The new screen has almost exactly the same dimensions, but higher resolution (1024 x 600 versus 800 x 480) for better quality animation graphics and built-in stereo speakers for sound effects. Work has begun to design and build the final case.

For the IBM 1620 Jr., individual decks of punched cards will be stored in USB memory sticks. To visually reinforce that, custom shells have been 3D printed that fit over memory sticks:



"read" side "punch" side

Similarly, the entire IBM 1620 Software Library will be stored in a single USB memory stick and the card reader's touch screen will be used to select the specific card deck to be read. A different custom shell will be used with these memory sticks:




"read" side back side (the library memory stick is read-only)

**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.

With the development of the virtual card read-punch device, the card communication protocol, the punched card I/O instructions (RNCD, RACD, WNCD, WACD, DNCD), and the corresponding lights on the front panel (RDR FEED, PCH FEED, LAST CARD, READER NO FEED, PUNCH NO FEED) were implemented in the simulator. While doing this, a few minor errors in the INPUT-OUTPUT front panel lights for typewriter I/O were found and fixed.

The Computer History Museum has a mixed assortment of the official IBM 1620 diagnostics – some on paper tape, some on cards, some in listing form, some missing. Fortunately, ten of the twelve diagnostics that would be apply to the IBM 1620 Jr. are available and usable.

The most critical missing diagnostic is CU06 / DT106 for floating point instructions. That is why volunteer John Bohn offered to write a new, comprehensive floating point instruction test program for the project. He completed its development in January 2019. The program was validated using the SIMH i1620 simulator. The diagnostic has been named FP01 for use with the IBM 1620 Jr.

IBM diagnostic CU02 (Error Check) is a special case. It is designed to force a Check Stop condition for some VRC (parity) errors that can occur in a real IBM 1620. The simulator does not emulate parity, so the CU02 test doesn't apply. However, there are numerous user programming errors which trigger a Check Stop that the simulator detects and are worthy of a diagnostic. So, a new test program, named CS01, was written for the IBM 1620 Jr. that exercises all of the Check Stop conditions in the simulator.

To functionally emulate the IBM 1620's non-volatile core memory, the simulator dumps its "core memory" to a disk file whenever the console POWER switch is turned off and reads it in when the simulator first starts running. Since the simulator is always running when the main power switch [in the back of the case] is on, the contents of "core memory" is not affected by intermediate POWER off's and on's. The IBM 1620 Jr. simulator has a special feature, not available on the real IBM 1620, which leverages this capability. Instead of making use of the saved core memory file, the simulator can preload one of sixteen built-in files. To do this, the console POWER switch must be off, then the user sets the four Program Switches to the selected memory image, and finally they press the RESET button which clears memory and reads in the desired file. Now when the POWER switch is turned on, instead of the previous contents of core memory, a new program has been loaded.

In September 2020, all of the diagnostic programs that apply to the IBM 1620 Jr. were extracted from the Software Library, post-processed, converted if necessary, packaged, and added to the simulator. They are now built into the IBM 1620 Jr. and readily available. They would not need to be run normally unless the simulator is changed or some problem in the CPU, front panel, or peripherals is suspected. A corresponding *IBM 1620 Jr. Diagnostic Manual* with all run instructions, sample output, and IBM documentation was created and is available here: https://github.com/IBM-1620/Junior/blob/master/diagnostics/IBM_1620_Jr_Diagnostic_Manual.pdf

These are the diagnostic programs available:

| PS 1 | PS 2 | PS 3 | PS 4 | Name | Description |
|---|---|---|---|---|---|
| off | off | off | on | CU00 | Console Diagnostic |
| off | off | on | off | CU01 | General Op Codes Diagnostic |
| off | off | on | on | CU05 | Special Instructions Diagnostic |
| off | on | off | off | FP01 | Floating Point Diagnostic |
| off | on | off | on | CU03 | Indirect Addressing Diagnostic |
| off | on | on | off | CS01 | Check Stops Diagnostic |
| off | on | on | on | DX05L | Core Storage 20K Low Diagnostic |
| on | off | off | off | DX05H | Core Storage 20K High Diagnostic |
| on | off | off | on | CU04 | Additional Core Diagnostic |
| on | off | on | off | DX03 | Typewriter Diagnostic |
| on | off | on | on | IO02 | Card I/O Diagnostic |
| on | on | off | off | IO03 | Card I/O Reliability Diagnostic |

More of the IBM 1620's instruction set was implemented since VCF West 2020. Specifically, the divide (LD, LDM, D, DM) and additional (MF, TNF, TNS) instructions and indirect addressing are now fully functional and verified using the IBM diagnostics. The only instructions left to be implemented are the floating point operations (FADD, FSUB, FMUL, FDIV, FSL, FSR, TFL, BTFL).

## **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., assembling the associated documentation, and testing the programs on the IBM 1620 Jr.

Several specific programs have been extracted, post-processed, and run on the IBM 1620 Jr., but the bulk processing of the software library has yet to be done.

**Operation Guide** (Dave Babcock)

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

Work on the Operations Guide has not yet begun.

**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.

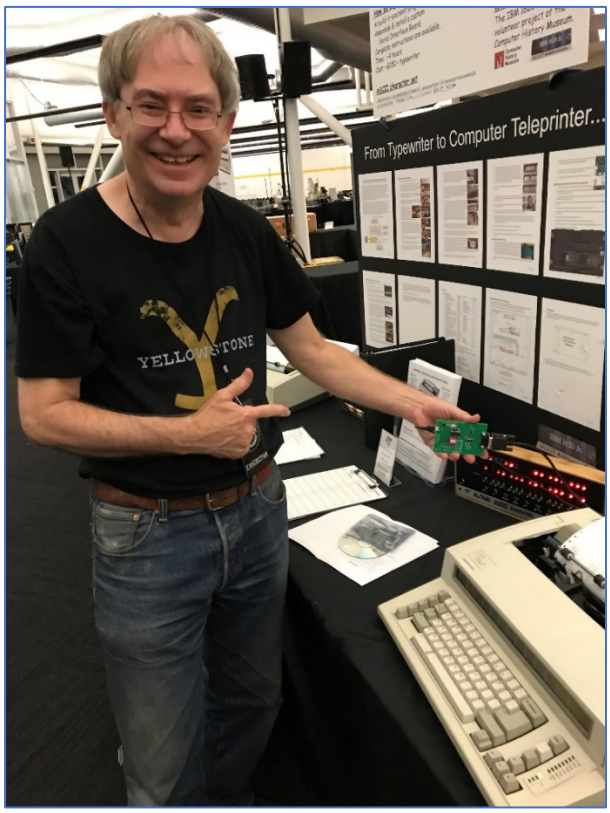
**Vintage Computer Festivals** (Team)

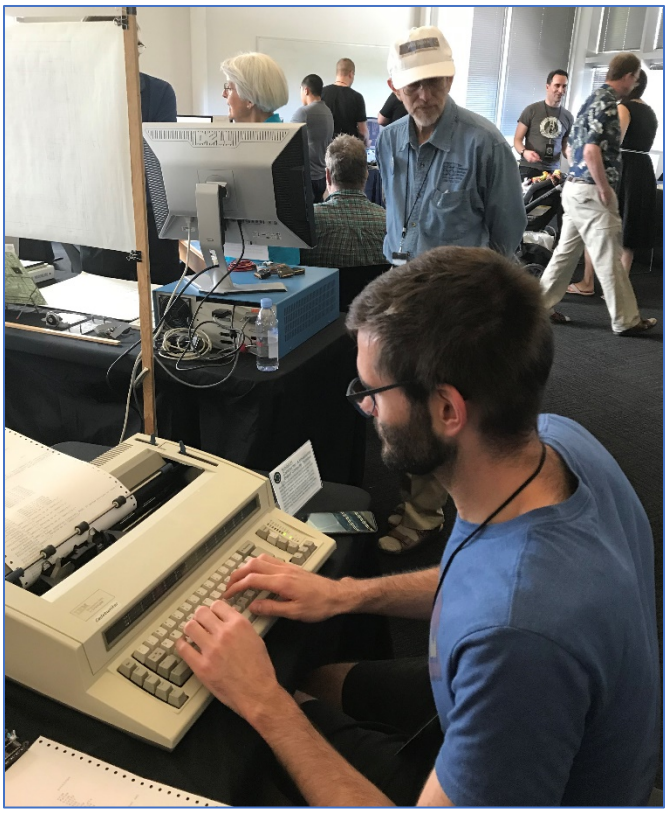
At VCF West 2019, the team had two adjacent booths – one for the IBM 1620 Jr. and one for Cadetwriter.

In the IBM 1620 Jr. booth, the most current version of the simulator, several original IBM 1620 demo programs, and the final console typewriter were featured.

The Cadetwriter booth contained two separate systems, each with their own Cadetwriter. A replica PDP-8/I [Oscar Vermeulen's PiDP-8] used a USB interface to connect to its Cadetwriter. A replica Altai 8800 [Chris Davis' Altair-duino] was connected to its Cadetwriter via an RS232 interface. Both computers were running classic, vintage software which made use of the terminals.


The team made an open offer to all VCF exhibitors – "Bring over your vintage or replica computer [or the Cadetwriter can come to you] and try it out with Cadetwriter." Two people responded to the challenge.



Even before the show opened, Steve Toner brought over his Tiny Basic Computer, literally in his hand. It had an RS-232 interface, but at first it didn't work with Cadetwriter. The problem was quickly isolated. The board's interface was set to 300 baud and that wasn't working correctly in the terminal. The speed of the computer was changed to 9600 baud and everything worked correctly, except for hardware flow control which was implemented wrong in the Tiny Basic Computer. The team was surprised to learn later that it is a known problem with the Teensy 3.5 microcontroller used in Cadetwriter. Low baud rates [600 and below] don't work correctly due to the high microprocessor clock rate. As described above, slow baud rates are now supported in Cadetwriter.

Near the end of the show, Anthony Hoppe of RMG Labs hooked a Cadetwriter to his Asterisk Telephone Switch's computer which was directly behind the Cadetwriter's booth. The RS-232 connection at 9600-8-N-1, worked perfectly the first time.

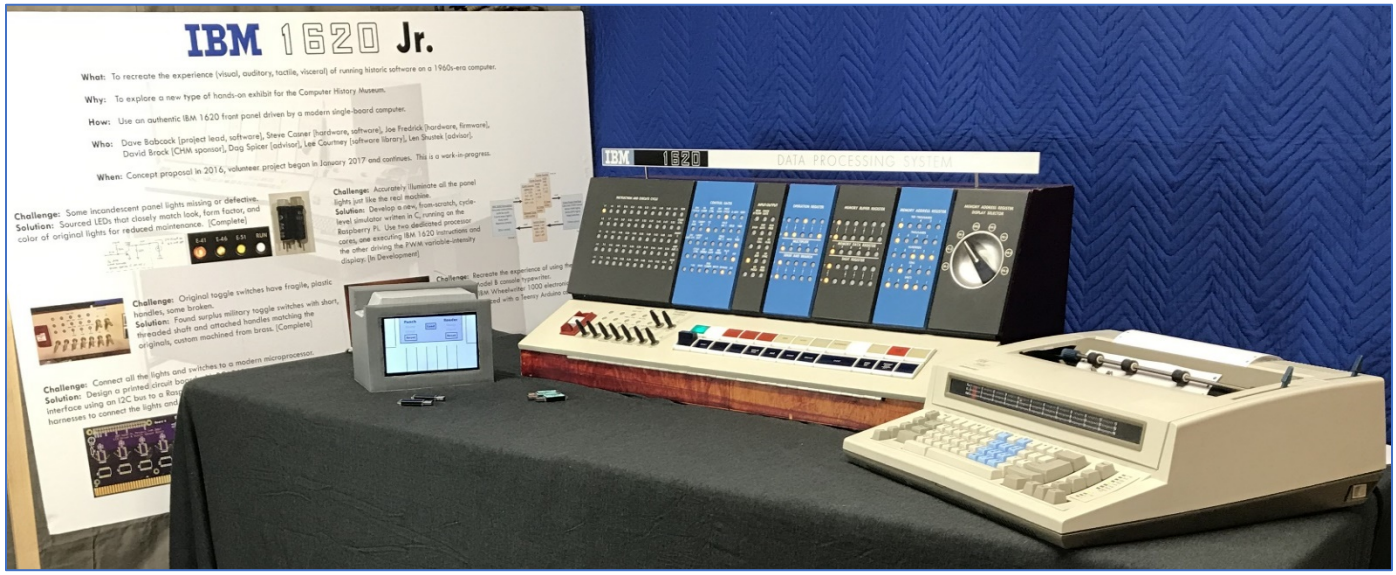

Rather than giving out the usual set of judged category awards, every booth was given a prize in 2019. The IBM 1620 Jr. received the "Best Re-Imagining" award. Cadetwriter won the "Most Original Peripheral" award.


In addition to the two booths, the IBM 1620 Jr. Team also gave a separate presentation on Cadetwriter. Here is the recorded presentation: https://www.youtube.com/watch?v=pNj6Ny_v2FU&t=624s and here are the slides from the presentation: https://github.com/IBM-1620/Cadetwriter/blob/master/docs/presentations/vcf_west_2019_presentation.pdf


Team member Steve Casner did triple duty at the show. He and his wife Karen also maned an adjacent booth with Steve's 1978 Solid State Monopoly Game. Their exhibit won the People's Choice award in addition to the booth prize!

Vintage Computer Festival West 2020 was an all-virtual event. Each exhibitor produced a 5 – 10 minute video of their "booth" and submitted it a month in advance. Then on Saturday, August 1st, each exhibitor had a scheduled 15-minute time slot when their video was played followed by a brief, live Question and Answer period. There was no judging or awards presented this year.

Here is the IBM 1620 Jr. exhibit video: https://www.youtube.com/watch?v=5mGHbALor3k
and this is what the virtual booth looked like:




The IBM 1620 Jr. video was a joint effort by Steve Casner and Dave Babcock. Steve was the director, editor, co-writer, narrator, and host. Dave filled the roles of producer, co-writer, photographer, graphic artist, and demonstrator. Located in the Bay Area, Steve used iMovie to composite all of the pieces of video, still shots, overlays, recorded narration, etc. to create the actual video. He made a "green screen" video recording area in his living room to film his hosting segments. All of the IBM 1620 Jr. hardware is with Dave in Southern California, so he built the "exhibit booth" in his garage to do the filming and static shots with an iPhone. The full demonstration was repeated eight times with each run shot from a different position. Dave wrote the initial script which Steve revised and expanded as the production progressed. Steve created the on-screen stopwatch and directed the "Ken Burns effects". Dave made the graphic slides. There were constant emails and phone calls to coordinate.

Development of the virtual IBM 1622 card read-punch and creation of the video was squeezed into a very short period of time:

- 5/20 -- Development of the card device began, hoping to complete it in time for August VCF.
- 5/24 -- Notified that VCF West 2020 would be virtual on August 1st and that videos needed to be submitted by early July. Team began planning exhibit.
- 6/15 -- Card device software development proceeding well. Planning done and first draft of video script complete. The team committed to VCF that there would be an IBM 1620 Jr. exhibit.
- 6/21 -- Initial exhibit booth built.
- 6/27 -- All software (IBM 1620 simulator, IBM 1622 emulator, SPS assembler) complete and able to successfully run the full demo.
- 6/30 -- Final booth ready and cardboard/wood prototype case for IBM 1622 completed.
- 7/1 -- Production of the video and filming started.
- 7/10 -- Video complete and submitted.

The video tells two stories – what the IBM 1620 Jr. project is and how software was developed in the 1960's – in-between an introduction and final credits in just 9 minutes 6 seconds. The IBM 1620 Jr. segment starts by briefly describing the IBM 1620 while showing archive photos. In the next 2 ½ minutes,

the IBM 1620 Jr. project's goals, description, and progress are presented through narration, graphics, videos, and still shots. The four main components of the project – control console, software simulator, console typewriter, and simulated card reader / punch – are shown and explained. The Cadetwriter side project is also covered. Most of the video (5 ½ minutes) demonstrates what it was like to code, keypunch, assemble, run, and debug a sample program written in the SPS assembly language. This fully functional, recorded live demo is an example of what the Computer History Museum's education department could do with the IBM 1620 Jr. The assemble and run portion of the demo actually took 5 minutes 27 seconds, but the video was sped up during the boring parts with an on-screen stopwatch providing the time reference. The team was told that it was the most thorough and professional looking video of the entire VCF show.

The sample program used in the video was a Power of 2 calculator. Written as part of the IBM 1620 restoration project, it was used to publicly demonstrate the machine in CHM's "Visible Storage". It highlights the IBM 1620's most unique feature – high-precision, variable-length, decimal arithmetic. The user types in a value from 0 to 9999 and the program calculates 2 to that power. In the video, it was used to calculate 2^234, producing a 71-digit result. Two to the 9999$^{th}$ power takes 20 minutes to compute and 5 minutes to print the 3010-digit result:

```
2**9999 = 99753155844Ø3791924418710813417925419117484159430962274260044749264719
4151109733159599808420180972989496655647116045621357782456747068905587968929660 4
8161978927865Ø233968972633826232756330299477602750434590966557712543042303090523
4275453743304481244404524494741900462697081662892531078415473695127845619403261 2
5483219372205233799358134927266114342690808847157887814820381418440380366114267 54
582Ø738Ø919781907294847319497054204802681339105323107136666970182627828247653Ø15
7134Ø1174847ØØ167967158325729648886639832887803Ø8629101570399709908980368912284 1
8811400186514427436259504172322907273252789648007074169608078672940696285476898 8
4559638900413478867837222061531009378918162751364161894635355186901433196515714Ø
6662070081209783584528703070982717116231940062442807365260371599612980589812506 5
4964301208541704038029661600806342461442481279206564220307683694757435571281575 5
5544872757101656910101465820478798232378005202922920783036022481433508257530960 3
1550209321113795433545028730320892847595572802753412562520300375992113094902961 8
5590272223940364531976212741696109913537022365811883804233065168893530199017065 9
8566746827311350281584968727754120890486405491645657201785938762384254928638468 9
6321661079969993844333040418441891901382164138758613682878637239205614719486690 5
4308037116266459874065600098802089140982848737949082265629217067979931392065064 09
2703141738324544345260523790441307911980992885061203522165291537934519659802301 7
0248657829160433605295665045187641170776987269719885762872764525510615547366080 5
3767374128703876369931741492491703784689778233193109372847496395082860518506822 1
6567908607155895699111491922923667220135482091425502536463874182275289317250550 4
2649390619473696434977041717307940352197955949290757288958857180984936406572974 1
8916010407374910859290056945356141254529134087181102887379607088268578438628074 5
2291452496230514315040767791654065050993837928117171769477704587811700422443763 0
8132178432441675973186018864662004722812346162717520033901363691887768820336344 9
3181205187457054833592785253795490501233949400891359629766906412109770141513797 0
4224477507338334194848998443120818156688196951686727900703818370938855527692112 8
6974955509323410984829082574256524711118497385738153457773410884143810018138862 8
8618906826658055984056403963347409436006493218303842758199302673011489357787589 7
3692623184723461543947132974108504025560161182748144084517869560684169196795878 2
0936692525548513580695771979549579907732720866815582846801556112496898499961339 0
8661790115559313222876495678790875040999196181423076249405444801161221810868858 0
9043178507734242029311164896426937811743278220268481311009481785514406180783756 2
7166915163501454883432528427857875275836375944959706485566884507495809065758577 2
0038643252865947787254606165092652423556909157703662026659519231042018210881851 95
5775319894500371426836098140451738987266660234184397934290118976109314560040371 4
0977565897407881222414925923075485244401363736078734406579737520486605754024909 5
2279017084134748935706580316053431957558408871523962983546 88
```