The Coleco ADAM Universal Interface (RS-232 / Parallel)

Hardware identification, firmware reverse engineering, and host-software confirmation of an unreleased Coleco peripheral — from engineering prototype PCB X00257 REV 0 (EPROM "AHSPI REV 1.2") to the production "ADAM Universal Interface"

ColecoVision & ADAM Archive & Museum · AdamArchive.org · ColecoVisionADAM.com

Hardware analysis, disassembly, and firmware development by Richard DiRocco with Claude (Anthropic).

Evidence legend. **CONFIRMED** verified from schematic, photo, opcode, or running code
INFERRED strong inference
OPEN requires further investigation **DESIGN** new engineering work introduced here

1. Executive summary

This document establishes, from multiple independent lines of evidence, that a clear-acrylic-cased Coleco engineering prototype — PCB X00257 REV 0, carrying an EPROM hand-labelled AHSPI REV 1.2 — is the engineering precursor to an unreleased production peripheral Coleco called the ADAM Universal Interface (RS-232 / Parallel). The device attaches to the AdamNet bus and gives the ADAM a standard RS-232 serial port and a Centronics parallel port, selectable through a three-position mode switch, with transparent compatibility for existing software via SmartWriter-printer emulation.

The identification rests on four sources that agree with one another: the prototype hardware and its firmware; a Coleco Marketing Communications "New Products" memo describing the finished product; a working emulation of the same device built by the FujiNet project (whose author worked backward from the ADAM CP/M BIOS); and the AdamCalc application, which detects the interface at boot. This document also presents completed, simulator-verified firmware for the device's AdamNet STATUS handlers — the first functional piece of firmware to fill the gaps Coleco left in the prototype ROM.

Headline findings. The device lives on AdamNet device 14 (hex $0E); it is software-compatible with SmartBASIC and CP/M because it emulates the built-in SmartWriter printer rather than demanding a new driver; only two prototype units are known to exist; and the production unit was advertised internally but never shipped.

The STATUS-handler firmware completed here returns the exact six-byte responses the AdamNet master expects, verified byte-for-byte against the protocol specification and against FujiNet's running implementation.

2. What this device is

The ADAM shipped with a closed peripheral philosophy: printers, drives, and the keyboard all spoke AdamNet, Coleco's proprietary half-duplex serial bus, and there was no standard RS-232 or Centronics port for connecting third-party printers, plotters, modems, or terminals. The Universal Interface was Coleco's answer: a single AdamNet peripheral exposing both an

industry-standard RS-232 serial port and a Centronics parallel port, so that ADAM owners could attach the same printers and modems everyone else used.

Two engineering goals shaped the design. First, the unit had to present itself on AdamNet as a device the ADAM already understood, so that existing software — SmartBASIC, CP/M, SmartWriter — would drive it without modification. It achieves this by impersonating the built-in printer and by intercepting SmartWriter's output, rather than by introducing a brand-new device type that software would have to be taught about. Second, it had to be switchable, so the user could choose whether output went to the serial port, the parallel port, or straight through to the original SmartWriter printer.

3. The production product: ADAM Universal Interface

A Coleco Marketing Communications "New Products" memo documents the intended retail product. It describes a unit that connects RS-232 or Centronics peripherals — printers, plotters, modems, and input devices — to the ADAM through AdamNet, supports up to two units simultaneously, and is software-compatible with SmartBASIC, CP/M, and other software written to use it. The finished product carried a clean white/cream enclosure labelled ADAM UNIVERSAL INTERFACE with an ON indicator LED.

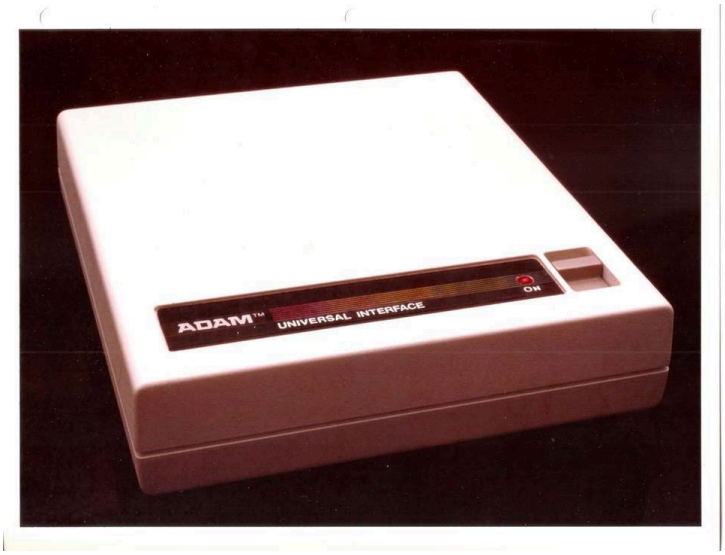


Figure 1. The production ADAM Universal Interface (Serial / Parallel) as pictured in Coleco materials — the finished enclosure that the X00257 prototype was engineering toward. The unit never reached retail.

The codename on the prototype EPROM, `AHSPI`, is consistent with "ADAM Host Serial Parallel Interface" **INFERRED** — the working engineering name for what marketing would present as the Universal Interface.

4. The engineering prototype: PCB X00257 REV 0

The surviving prototype is a Coleco-manufactured engineering board — plated-through-hole, full silkscreen, solder mask, with the part number X00257 and revision REV 0 printed beside the Coleco logo — housed in a hand-cut clear acrylic case. It is not a hand-built breadboard; it is a real pre-production board with a small number of engineering modification wires on the solder side.

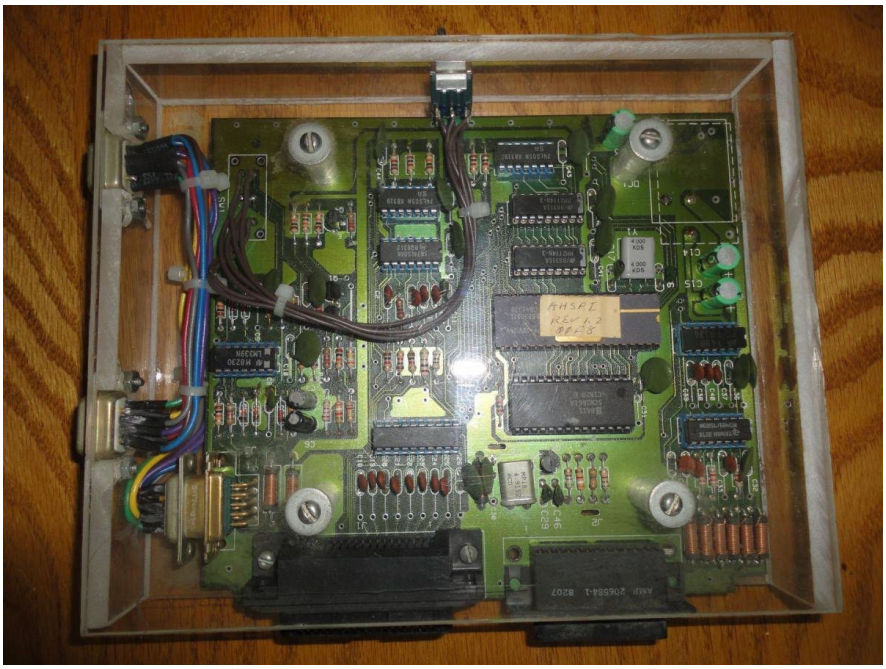


Figure 2. Top-down view through the acrylic lid. The socketed 24-pin EPROM carries an orange sticker reading AHSPI / REV 1.2 / 00F8. Board silkscreen designators (U…, J1–J3, Y1, Y2) are clearly visible.

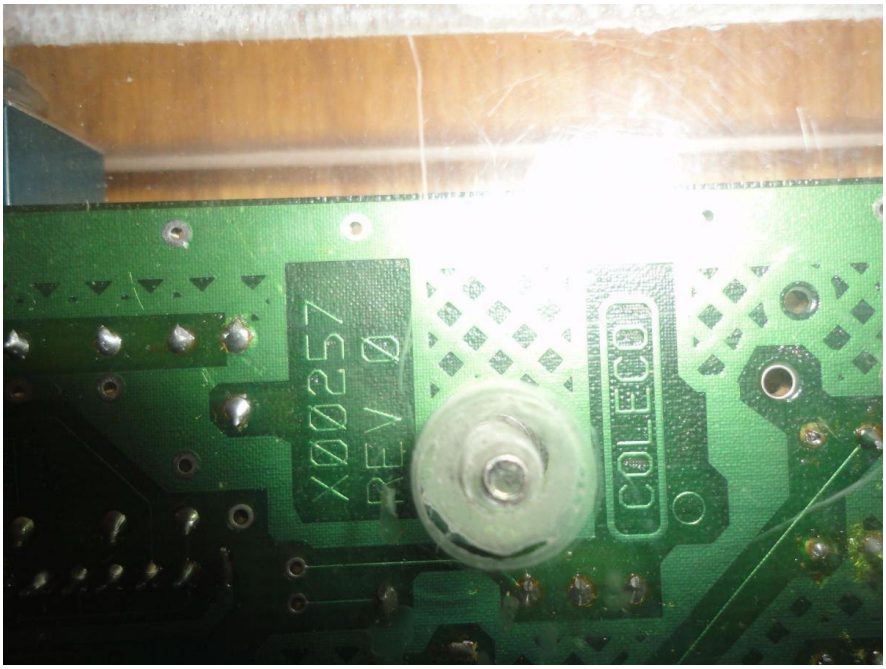


Figure 3. Solder-side silkscreen: part number X00257 REV 0 with the boxed COLECO logo, positively identifying a Coleco engineering board.

Attribute	Finding	Status
Enclosure	Clear acrylic, screw-fastened, hand-cut panels	**CONFIRMED**
PCB	`X00257 REV 0`, Coleco-manufactured (silkscreen, solder mask, PTH)	**CONFIRMED**
EPROM label	Three hand-written lines: `AHSPI / REV 1.2 / 00F8`	**CONFIRMED**

Attribute	Finding	Status
Codename	“ADAM Host Serial Parallel Interface”	INFERRED
Label line 3	`00F8` — firmware entry hint; reset vector targets $F800	INFERRED
Units known	Two prototypes known to exist	CONFIRMED

5. Integrated circuits and bus interface

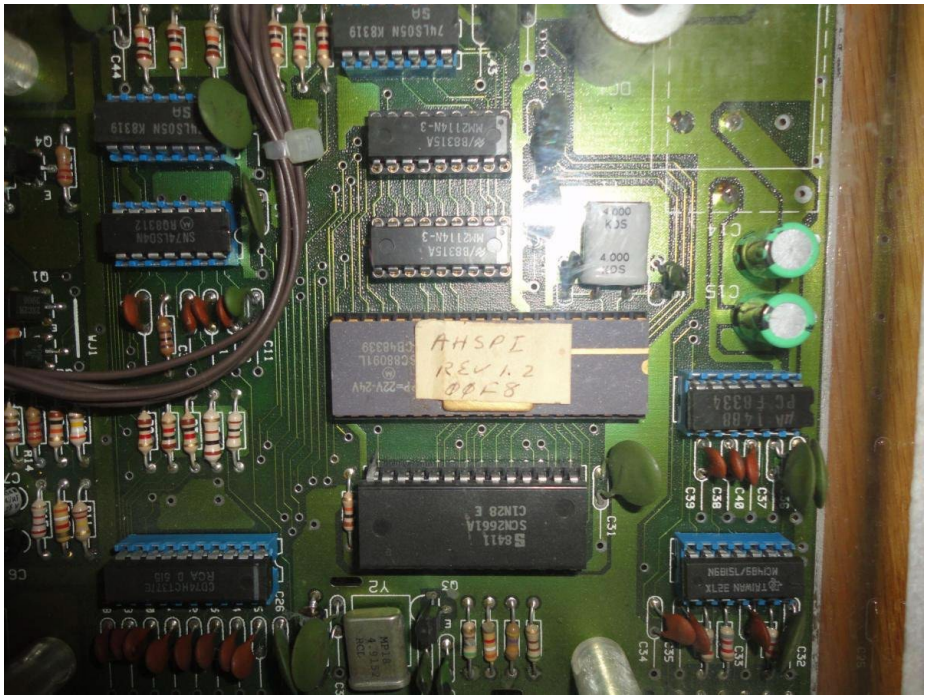


Figure 4. IC close-up. The EPROM sticker, the Signetics SCN2661A USART (date code 8411), two MM2114N-3 SRAMs, 74LS04/74LS05 logic, the 4.000 MHz crystal, and the µA1488 RS-232 driver are all visible.

Function	Device	Role	Status
MCU	Motorola MC6801	System MCU; AdamNet node logic + port control	CONFIRMED
Firmware ROM	`SC8060IL` (2716 family)	Mapped at `$F800–$FFFF`	CONFIRMED
USART	Signetics SCN2661A	RS-232 serial port engine	CONFIRMED
SRAM ×2	`MM2114N-3`	Working RAM (1 KB)	CONFIRMED
Parallel latch	`CD74HC373E` (schematic: 74LS273)	Centronics 8-bit data latch	CONFIRMED
Decode logic	`74LS05` open-collector	Wired-AND address decode (UART/RAM/IO)	CONFIRMED
RS-232 driver	`µA1488`	TTL → RS-232 (±12 V)	CONFIRMED
RS-232 receiver	1489-equiv.	RS-232 → TTL	CONFIRMED
MCU clock	Crystal Y1 4.000 MHz	6801 clock; AdamNet bit timing	CONFIRMED
USART clock	Crystal Y2 4.9152 MHz	SCN2661A baud-rate generator	CONFIRMED

6. Connectors and the three-position mode switch

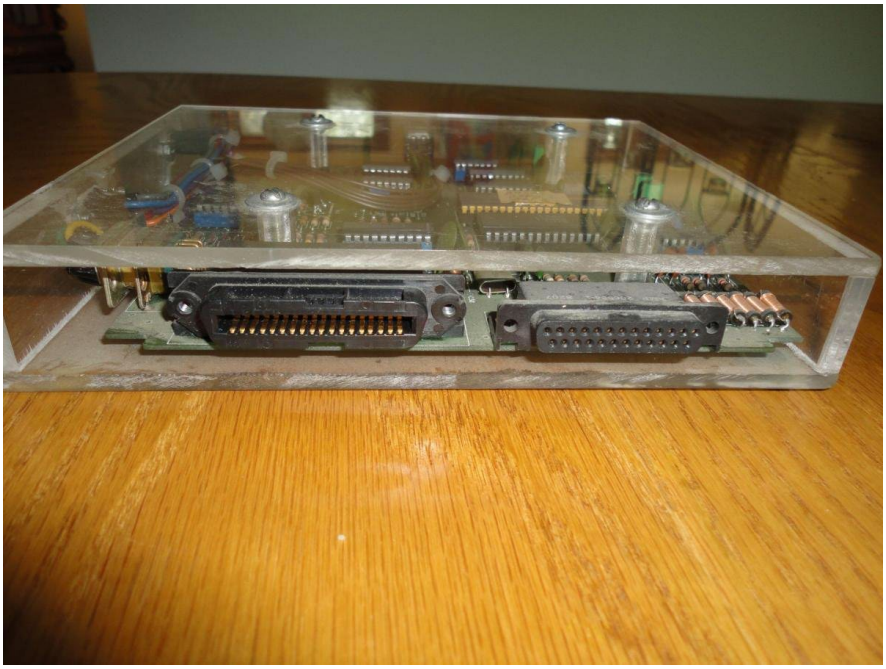


Figure 5. Rear panel: J1 36-pin Centronics (left) for parallel output; J2 25-pin D-shell (right) for RS-232.

Figure 6. Front panel: 9-pin AdamNet + external-power connector (the prototype uses a 9-pin connector, not the usual RJ12).

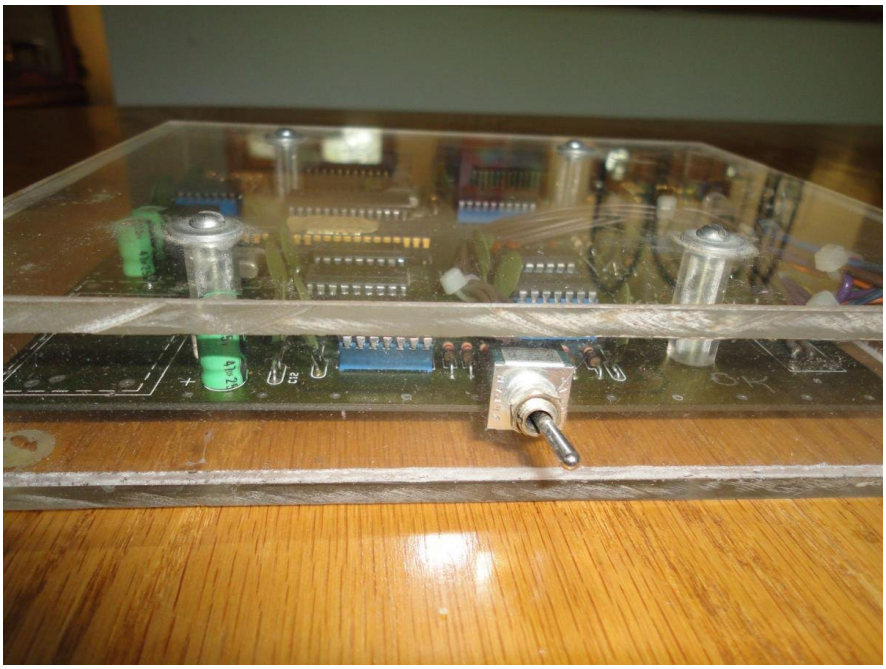


Figure 7. The three-position mode switch on the side panel selects RS-232 / PARL / SmartWriter by driving bits of the 6801's port.

The front 9-pin connector is a notable departure from AdamNet convention. Standard AdamNet peripherals use an RJ12 jack; this device instead uses a 9-pin connector that carries AdamNet signalling and external power. The reason is the SmartWriter-intercept function: the switch shunts the SmartWriter printer's input and output so the unit can sit between the ADAM and its printer, selecting whether SmartWriter output is passed through to the original printer or redirected out the new serial/parallel ports.

> **Note — switch location.** A recorded walk-through of the prototype describes the mode switch as being "on the bottom." High-resolution photographs of this unit place the toggle on a side panel. The photographic evidence is taken as authoritative here; the discrepancy is most likely a casual description of a different handling orientation.

7. AdamNet device identity: device $0E

On the AdamNet bus, every peripheral answers at a four-bit device address. Roll-call after reset has the ADAM send a CONTROL.STATUS request to each address; a device that responds with a valid status packet is given a Device Control Block and considered present. The Universal Interface answers at device 14, hexadecimal $0E.

This is established three independent ways: the prototype firmware dispatches its serial-port STATUS handler from the device-$E vector and carries a device-$E response template in ROM; the FujiNet emulation of the device places it at $0E; and a logic-analyzer capture of a real ADAM boot shows the device answering the status request at address $E. The address-emulation strategy that gives software compatibility means the unit also exposes a Centronics parallel port at AdamNet device $0D, as independently confirmed by AdamCalc's boot device-probe table at $32AB (see Section 9). The third (SmartWriter) switch position is a pure pass-through that does not claim an AdamNet address.

AdamNet address	Role in this device	Switch position
`$0E (14)`	RS-232 serial port	Serial
`$02 (2)`	Printer identity (Centronics output)	Parallel
`$0D (13)`	Centronics parallel port (74LS273 latch)	Parallel

8. Independent confirmation: FujiNet and CP/M

The FujiNet project — an open-source ESP32-based AdamNet peripheral emulator — implemented a working emulation of this exact device and published it in December 2021, describing it as “the unreleased AdamNet Serial Interface for the ColecoVision ADAM… that was prototyped but never released by Coleco.” Crucially, the FujiNet author states that the original ROM dumps were corrupt, so the emulation was reconstructed by working backward from the ADAM CP/M BIOS implementation of the serial device.

That detail matters in two ways. First, it independently corroborates the prototype’s identity, hardware (Signetics 2651 USART, 6801 + EPROM, DB-25 serial, 36-pin Centronics), the device-$0E address, and the “two known to exist” count. Second, it means the ADAM CP/M BIOS itself contains a working host-side driver for this device: by default CP/M maps the TTY and reader/punch devices to the serial port, so a CP/M user can redirect output to the interface with PIP or move the console to it for an 80-column serial terminal. The marketing memo’s claim of CP/M compatibility is therefore demonstrably real.

> **Reference implementation.** Because CP/M is the authoritative host driver and FujiNet’s firmware is open source, the AdamNet node behaviour the production firmware must implement — the exact STATUS response and the SEND / RECEIVE / READY command flow for device $0E — exists as readable, running code. That code is the specification against which the firmware in §11 is written and verified.

9. Host-side detection: AdamCalc

The AdamCalc spreadsheet application (Coleco, 1984) detects the interface during its boot device scan. When the serial interface is present at device $0E, AdamCalc recognises it and draws an on-screen indicator for it — confirmation that period software was already aware of the device, even though AdamCalc does not yet make functional use of it.

AdamCalc’s boot image carries a custom TMS9918 graphics set containing device glyphs and text labels. Extracted directly from the application’s data, that set includes a keyboard/device icon and legible text fragments including `Serial` and `Parallel` — independent evidence, from inside shipping software, that the serial/parallel interface was a known quantity. The live-emulator capture below shows both labels rendered on the boot screen alongside the seven default-detected device icons; the AdamCalc device-probe table at $32AB lists device $0E with label Serial and device $0D with label Parallel.

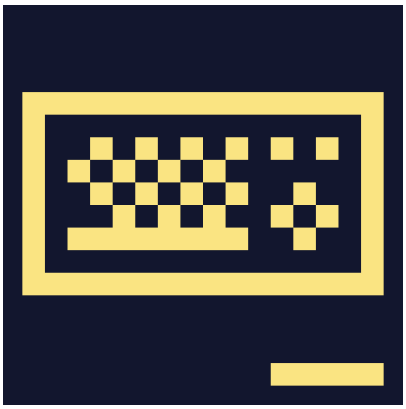


Figure 8. A device glyph reconstructed pixel-for-pixel from AdamCalc's boot graphics data.

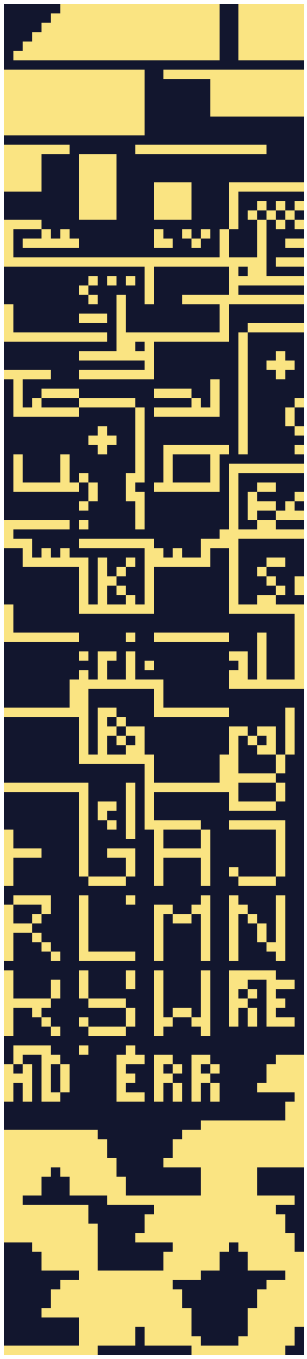
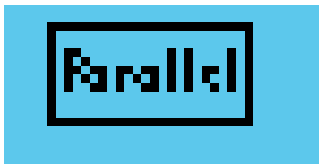


Figure 9. The custom graphic set from AdamCalc, showing device pictograms and text labels including “Serial” and “Paral.”

AdamCalc boot screen captured from a live emulator after a two-byte patch at file offset 0x0A56 (B7 20 → 37 C9) defeats the per-device AdamNet presence check at $3256. All nine entries of the device-probe table at $32AB now draw their icons; the bottom row reveals the previously hidden Serial and Parallel labels.

Icon for device $0E — label Serial. The RS-232 half of the Universal Interface.



Icon for device $0D — label Parallel. The Centronics half of the Universal Interface.

The probe table at $32AB in the AdamCalc binary is a nine-entry list of (device-id, icon-code) pairs. Two of those entries name the Universal Interface explicitly: $0E with icon labelled Serial and $0D with icon labelled Parallel. This is independent, application-level corroboration that the device exposes two AdamNet identities — one per port — rather than a single identity selected by the front-panel switch. The third (SmartWriter) switch position is a pure pass-through that does not claim an AdamNet address. [CONFIRMED]

> **Note on the figures.** These are faithful renderings of the raw character data stored in the AdamCalc image; the exact on-screen composition is set at run time by the program's name-table writes and is best captured from a live emulator. The glyph and label content shown here is taken directly from the binary.

10. The firmware gap: what Coleco left unfinished

Disassembly of the prototype EPROM ($F800–$FFFF) shows a firmware that is structurally complete — it has a working AdamNet byte transmitter, command dispatch, and the reset/interrupt vectors — but whose three device STATUS handlers are empty. The dispatcher jumps into regions of the ROM that were erased (all $FF):

Dispatch	Target	Device	State in prototype
`$FC4F: JMP $FD44`	`$FD44`	$0E (RS-232)	erased gap (all $FF)
`$FCB3: JMP $FD3D`	`$FD3D`	$02 (printer / pass-through?)	erased gap (all $FF)
`$FC80: JMP $FD36`	`$FD36`	$0D (Parallel)	erased gap (all $FF)

The ROM does, however, already contain the four-byte NM_STATUS response templates the handlers were meant to send — sitting in another erased-adjacent region:

```
$FF70:  8E 10 00 00     ; device $E, max msg 16, character device
$FF75:  82 10 00 00     ; device $2, max msg 16, character device
$FF7A:  8D 01 00 00     ; device $D, max msg  1, character device
```

In other words, Coleco specified the responses but never wrote the code to send them. Combined with the fact that the surviving ROM dumps were independently reported as corrupt, this is the concrete form of what the community has long described as the device being "prototyped but never finished."

11. Phase 1 firmware: completed STATUS handlers DESIGN

This section presents completed firmware that fills the three empty STATUS handlers. On a CONTROL.STATUS request, an AdamNet node must reply within roughly 200 microseconds with a six-byte NM_STATUS packet:

```
[ cmd_dev ][ size_lo ][ size_hi ][ devtype ][ status ][ checksum ]
  checksum = XOR of size_lo, size_hi, devtype, status
```

The design reuses Coleco's own response templates and the firmware's existing byte transmitter ($FF2C) and post-send routine ($FF48). Three tiny stubs occupy the small erased gap at $FD36; each loads a pointer to its template and jumps to a single shared sender placed in the larger erased gap at $FF7F. The reset and interrupt vectors are left untouched.

Shared sender (SENDST, at $FF7F)

```
SENDST:  LDAA 0,X      ; cmd_dev byte (not in checksum)
         JSR  $FF2C
         CLR  $00FA     ; running checksum = 0
         LDAA 1,X       ; size_lo
         JSR  $FF2C
         EORA $FA / STAA $FA
         LDAA 2,X       ; size_hi
         JSR  $FF2C
         EORA $FA / STAA $FA
         LDAA 3,X       ; devtype
         JSR  $FF2C
         EORA $FA / STAA $FA
         CLRA           ; status = 0
         JSR  $FF2C
         LDAA $FA       ; checksum
```

```
        JSR  $FF2C
        JSR  $FF48     ; post-send bus housekeeping
        JMP  $FE24     ; return to main loop
```

Device stubs (in the $FD36 gap)

```
$FD36 (dev $D):  LDX #$FF7A  ;  JMP SENDST
$FD3D (dev $2):  LDX #$FF75  ;  JMP SENDST
$FD44 (dev $E):  LDX #$FF70  ;  JMP SENDST
```

> **Implementation note — a real silicon trap.** The natural way to write the send loop is to use the 6801's B register as a byte counter. That is wrong on this chip: the stock transmitter at $FF2C reads the serial status register into B (LDAB $11) and so clobbers B on every call. The counter is destroyed after the first byte and the loop runs away. The working version above is unrolled and holds all state in A, X, and a zero-page checksum byte. This is exactly the kind of defect that simulation catches before an EPROM is burned.

12. Verification

Each handler was executed on an instruction-level 6801 simulator, entering at the real dispatch address and capturing every byte written to the serial transmit register. All three reach the firmware's normal return point ($FE24) and emit exactly six bytes, with valid checksums and nothing trailing:

Device	Bytes transmitted	Checksum	Result
$0E (RS-232)	`8E 10 00 00 00 10`	valid	**PASS**
$02 (printer / pass-through?)	`82 10 00 00 00 10`	valid	**PASS**
$0D (Parallel)	`8D 01 00 00 00 01`	valid	**PASS**

The device-$0E response 8E 10 00 00 00 10 matches the printer roll-call form in the AdamNet protocol reference and the response emitted by FujiNet's running implementation. The output is identical across three independent specifications: the prototype ROM templates, the protocol document, and the FujiNet source.

> **Scope of "verified."** This is instruction-level correctness: the handlers read the right templates, compute the right checksums, drive the stock transmitter correctly, and return cleanly without disturbing the reset/interrupt vectors. What remains to be confirmed on real hardware is bit-cell timing — whether the actual 6801 meets AdamNet's response deadline at the true bus rate. That is a scope check on a physical unit; everything provable in simulation is proven.

13. Performance: the case for native hardware

The FujiNet author noted that the emulated serial path is slow — output emerges one character at a time with noticeable overhead — and invited the community to make it faster. The bottleneck is structural: FujiNet emulates the entire AdamNet node on an ESP32 and bridges the data through a TCP socket, so every byte crosses a general-purpose network stack.

Native hardware does not have that overhead. A real 6801 driving the SCN2661A USART directly services the AdamNet STATUS poll (observed at roughly one every 8 milliseconds) and streams transmit data straight from the UART. In other words, the "make it faster" challenge is

answered simply by being the device the prototype was always meant to be — which is the motivation for completing this firmware on the original silicon.

14. Open investigations and next steps

Item	Status
Real-hardware bit-cell timing of the STATUS handlers (scope check on a physical unit)	**OPEN**
SCN2661A USART register base address (LS05 wired-AND decode or bench trace) — gates the Phase 2 serial driver	**OPEN**
Function of the second front-panel 9-pin connector on the built unit (schematic shows only one)	**OPEN**
Reconcile the firmware dispatch at $FCB3 → $FD3D (responds to AdamNet $02) with the AdamCalc finding that $02 is the standard ADAM printer, not the Universal Interface — possible roles: a SmartWriter pass-through stub, or a legacy printer intercept	**OPEN**
Phase 2: full SCN2661A serial driver and the SEND / RECEIVE / READY data path for device $0E, using the CP/M BIOS and FujiNet source as the behavioural spec	**DESIGN**

15. Sources

Source	Contribution
Prototype PCB X00257 REV 0 and EPROM “AHSPI REV 1.2”	Hardware, IC complement, connectors, firmware disassembly
Coleco Marketing Communications “New Products” memo	Production product identity, feature set, software-compatibility claims
FujiNet project (Thomas Cherryhomes) — videos + open-source firmware	Device $0E confirmation, “two known to exist,” CP/M-BIOS-derived behaviour, reference STATUS response and command flow
AdamNet Protocol Reference (Cherryhomes + Coleco EPS #227)	NM_STATUS packet format, roll-call sequence, timing deadlines
AdamCalc (Coleco, 1984)	Period host-side detection of device $0E; extracted device glyphs and “Serial/Parallel” labels
ADAM CP/M 2.2 (AJM/Coleco)	Authoritative host driver; TTY / reader / punch mapped to the serial port

Prepared for the ColecoVision & ADAM Archive & Museum. Firmware artifacts: rom_phase1.bin (completed STATUS handlers), phase1_status.py (assembler), sim6801_phase1.py (verification harness).