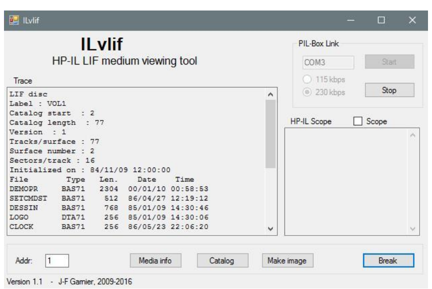
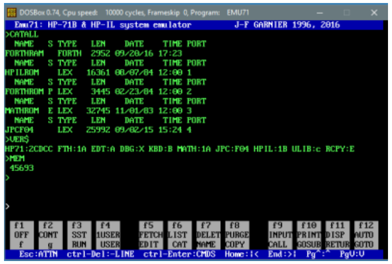
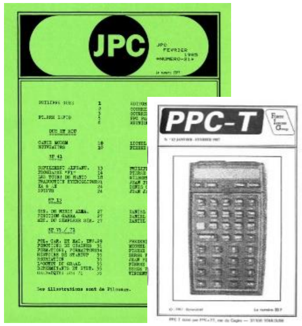
Allschwil 2016

Jean-François Garnier

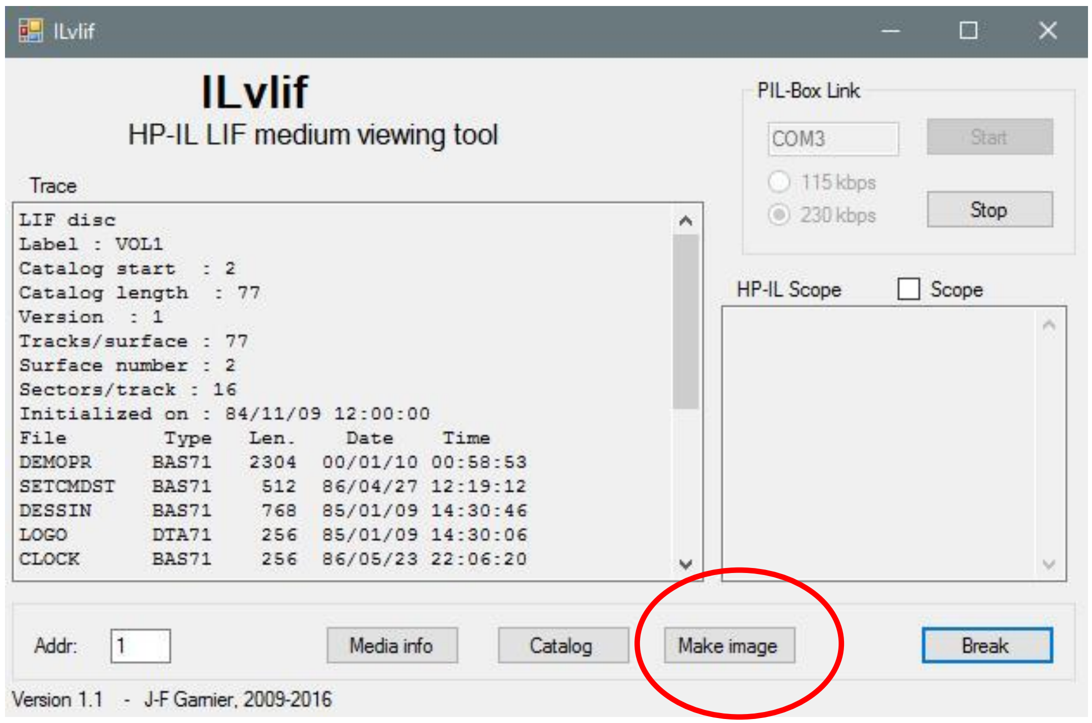
- Making LIF disc image files
- How fast is HP-IL
- Status on Emu71/DOS and scan of French PPC Journals







ILvLIF 2016



Reading LIF disc information and catalog from a PC,
and making LIF disc image !

ILvLIF 2016

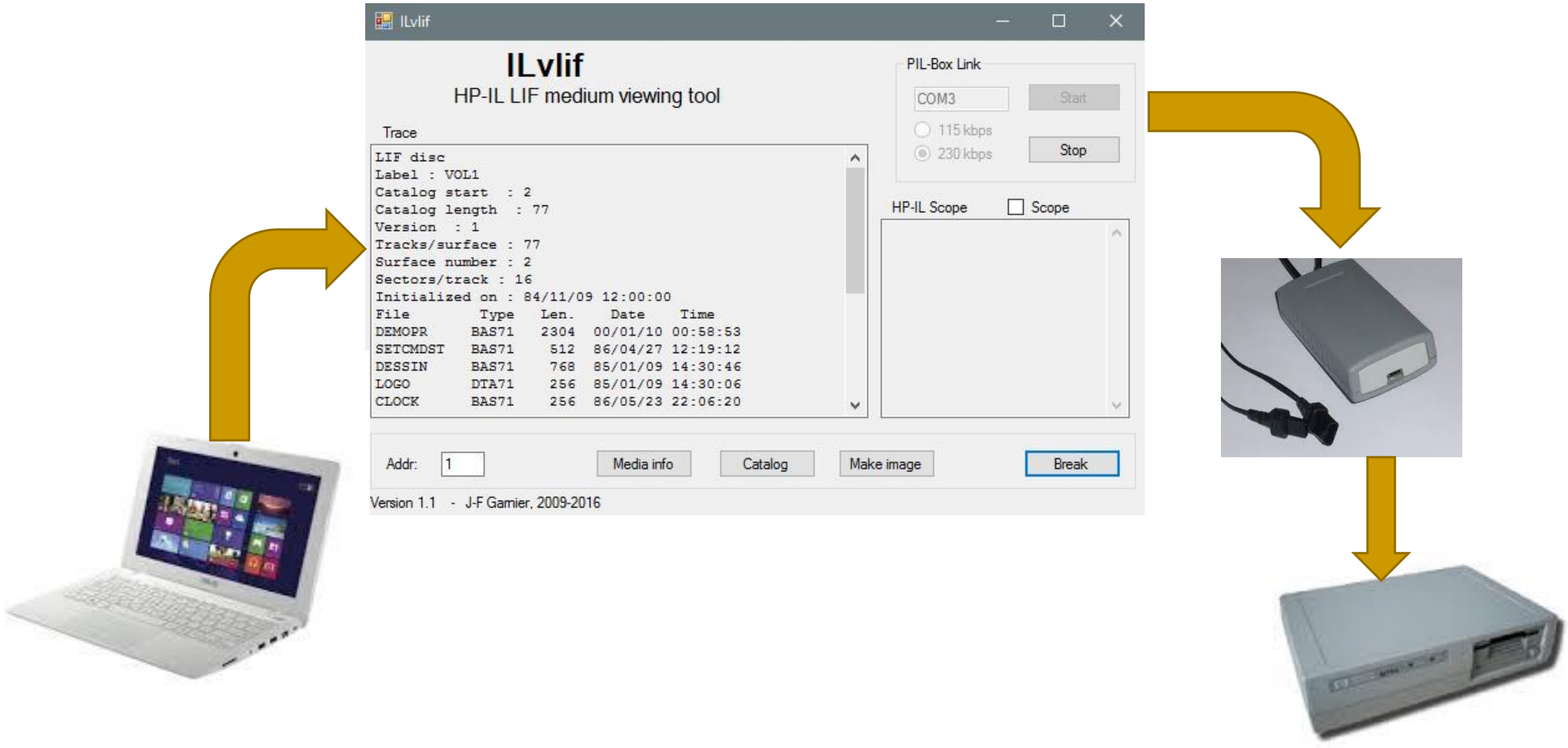
- In the past (1990-200x) I used to use an old DOS computer to access LIF discs from the PC.
I was using VLIF, a personal DOS utility to read disc information and catalog, and create a LIF disc image.
VLIF can still be found in the Emu71/DOS distribution package.

- In 2009, I partially ported my VLIF to Windows, for use with the HP9114 and the PIL-Box.
The resulting ILvLIF was more a demonstrator that a real utility.
Due the limited performance of the PIL-Box transfer speed (at the time), I didn't attempt to implement the disc image creation.

- In 2014-2016, I significantly improved the PIL-Box performance, first with version 1.5, then again with version 2.1.

- So … the 'Make disc image' function is now available!

ILvLIF 2016 - demonstration



ILvLIF 2016 - conclusion

HP9114 + PIL-Box + ILvLIF

Disc image files for ILPer

It's time to save your last LIF discs !

The end…

How fast is HP-IL ?

A discussion of the HP-IL performance,
from specifications to real measurements.

PIL-Box optimization.

How fast is HP-IL : what marketing said.

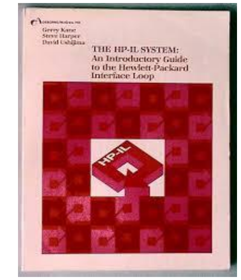
- HP “Marketing” often presented the HP-IL as able to reach a maximum speed of **20 kBytes/s**, much faster than the standard RS232 link of the time (~1 kByte/s at 9600 bps).

- This was based on the ~50us HP-IL frame duration (one byte transmission time), but has nothing to do with operation in real conditions.

- This figure have been used more or less unchanged by several reviews on HP-IL:

Data Transfer Rates

The rate at which data may be transferred over the loop interface is theoretically limited to 20K (20,000) bytes per second. Using equipment currently available (at the time of this printing), speeds of about 2K bytes per second are achievable. This translates to about half of a page of text (on an 8½″ × 11″ piece of paper) per second.



From “The HP-IL system: An Introduction Guide”, Kane, Harper & Ushijima, 1982, p.6

µP-PERIPHERAL CHIPS INTERFACES 93

Figure 5.15 shows a comparison among the HP-IL interface and other well known interfaces in a speed vs power requiments graph[14]. As it can be seen HP-IL is faster than RS-232 and slower than HPIB. However, it is unique in its low power requirements. It is this combination of good speed with very low power that makes HP-IL a suitable interface for portable devices. The most obvious is the portable computer needing battery powered peripherals (printers, mass storage).

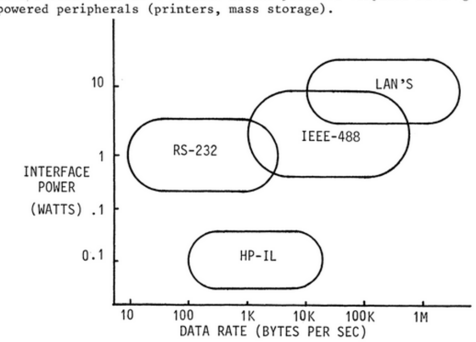
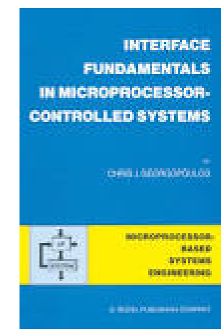


Figure 5.15. HP-IL speed/power comparison.*(After Ref.14)*.



From “Interface Fundamentals in Microprocessor-Controlled Systems”, Chris Georgopoulos, 1985

HPIL is suited to CMOS, battery-powered instruments. HPIL devices transmit data serially in a loop at up to 20 kBytes/sec.

From Digital Design, June 1982

How fast is HP-IL: a better analysis (HP)

- The HP Journal on HP-IL (January 1983) provided a much better analysis:

- Max theoretical performance with the 1LB3 HP-IL chip (software delays not included) :
  1 device: 7.1 kB/s
  3 devices: 6.4 kB /s
  5 devices: 5.5 kB/s

- Real performances with actual controllers:
  HP-85:   3 kB/s
  HP-41C: 200 Bytes/s

(The HP-75C was not mentioned,
and the HP-71B was not yet existing)

How Fast Is the HP-IL?

Determining the data rate for an interface is often difficult and misleading since it is usually dependent on highly variable hardware and software delays in the devices connected to the interface, rather than the maximum specifications of the interface itself. For any statement of data rate to be at all useful, the conditions under which the measurement is made must be carefully and completely specified.

The first important question is, "How fast could the HP-IL be if the devices did not limit the speed?" Assume a loop with only two devices: a controller/talker and a listener. The talker is continuously sending data frames to the listener. Because the first frame bit determines whether or not a frame is a data frame (as opposed to a command or ready frame), the listener will immediately retransmit the frame after only a one-bit delay and simultaneously load the frame into its buffer. Similarly, the talker can begin transmitting its next data frame after receiving only the first bit of the previous data frame. Error checking of the received frame is either not done or is done in parallel with transmission of the following frame.

Under these conditions, a continuous stream of data would fill the loop, limited only by the HP-IL timing specifications. These specifications say that a frame can be sent every 49 μs. The maximum data rate that could possibly be achieved is then slightly more than 20 kilobytes per second (each frame contains one byte of data). Furthermore, the loop could contain as many as ten of these "no-extra-delay" devices before the data rate would begin to suffer.

While such devices could certainly be built with existing technology, the HP-IL interface integrated circuit (see article on page 16) trades off this maximum performance for somewhat lower cost. This leads immediately to the next important question. "How fast would the HP-IL be if it were limited only by the existing interface IC and not by any added software delays?" Once again, the two-device loop example is useful, except that the devices now use the real HP-IL interface IC as opposed to a hypothetical one.

The sequence of events would proceed more or less as follows. After the data byte is written to the talker's HP-IL chip, there is a 4-μs delay before transmission of the bits starts. The frame takes 46 μs to be sent, and then the listener delays 7 μs before passing the byte to its microprocessor. The listener now retransmits the frame, which again has a 4-μs delay followed by a 46-μs transmit time. After the frame is received at the talker, there is a final 34-μs wait while error-checking is done before the following frame can be sent.

The grand total is 141 microseconds per frame, which translates to just over 7 kilobytes per second. Because the active listener does not retransmit the frame before it has been completely received, any additional devices will add to the time and degrade the data rate somewhat. An idle device using the present HP-IL interface IC delays a data frame by 13 μs so that three devices on the loop will reduce the speed to about 6.4 kilobytes per second, four devices will only support 5.9 kilobytes per second, and five will run at 5.5 kilobytes per second.

As the analysis gets closer and closer to the real world, software delays must also be accounted for. With a reasonably fast microprocessor and time-efficient (not necessarily ROM-efficient) code, an extra delay of no more than 50 μs could probably be achieved in each active HP-IL device. At a little less than 250 μs per frame, the two-device loop would have a speed of just over 4 kilobytes per second, three devices would operate at 3.8 kilobytes per second, four at 3.6, and five at 3.5.

Probably the easiest way to determine the data rate of a real system is simply to total the delay times of the individual devices involved. These times must be measured from the end of the received frame to the end of the frame transmitted through the device. Naturally, this value will vary somewhat depending on whether the device is acting as a talker or a listener and what type of data is being transmitted, but an average value is still useful.

For the HP-IL interface integrated circuit without extra software delays, the delay is around 70 μs. When the assumed software delay is added, the number goes to 120 μs. An HP-85 Personal Computer with the I/O ROM and HP-IL interface can achieve a frame delay of roughly 300 μs. The HP 82161A Digital Cassette Drive takes 600 μs per frame for a transfer of less than one 256-byte record, but increases to an average of 2600 μs per frame for very large blocks of data because of the record gaps on the tape. The slowest device is the HP-41C Programmable Calculator with its bit-serial microprocessor at about 5000-μs delay per frame.

A little computation then indicates that the combination of the HP-41C and the cassette drive can achieve a data rate of 175 bytes per second for short transfers and about 130 bytes per second for longer blocks of data. An HP-41C talking to an HP-85 can run at a little less than 190 bytes per second. If a cassette drive were used with an HP-85, the rate would be 350 bytes per second for long transfers, and about 1100 bytes per second for short ones. Two HP-85 Computers could communicate with each other at a little less than 1700 bytes per second across the HP-IL.

Clearly, the HP-IL is fairly slow when compared with the HP-IB. However, the data rate and other features of this interface system are well suited to the primary area of intended application: low-cost, battery-powered, portable systems.

-Steve Harper

From the HP Journal Vol.34, N1, January 1983:
article by Steve Harper.

How fast is HP-IL - specifications and benchmarks

Transfer rate (bytes/sec): 8K (copying to a loop, no devices on loop); 6.4K (copying in a file); 4.5K (OUTPUT statement, no formatting (USING)); 4K (ENTER statement, no formatting (USING) and version 1B of the HP-IL module).

From the HP-71B specifications in the HP Catalog, 1987, p.61

DISC BENCHMARKS

Disc	Speed	Interleave
9114 DS 3.5"	3.5Kb/S	4
9121 SS 3.5"	1.7Kb/S *	?
9122 DS 3.5"	1.7Kb/S *	?
8290x 5.25"	1.7Kb/S *	?
913x hard disc	1.7Kb/S *	NA

* 82169A HP–IL/HPIB Converter speed limit

PPA6E100 HEWLETT PACKARD

From a HP Portable Plus (HP110 Plus) training.
http://www.jeffcalc.hp41.eu/hpplus/files/slides_mass_storage.pdf

- Using the HP-71B (best HP-IL controller) a realistic figure is 4 to 6 kB/s max.
- Using the HP Portable Plus, the performance is limited to the speed of the external devices to about 2-3 kB/s

How fast is HP-IL - measurements

Benchmark on HP-71B, using the sequence:

1 RESTORE IO @ SEND UNL MTA
2 T=TIME @ COPY HPILROM TO :LOOP @ T=TIME-T
3 DISP 16384/T;" bytes/s"

Results:

HP-71B + HP-IL loopback test	8.5 kB/s
HP-71B + 1 device (HP9114A)	8.7 kB/s (yes, faster!)
HP-71B + 3 devices (HP9114A, HP82163B, HP82164A)	8.7 kB/s
HP-71B + 5 devices (. . . + HP3468A, HP5384A)	7.4 kB/s
HP-71B + HPIL/HPIB (HP82169A) mailbox mode	4.3 kB/s
HP-71B + HPIL/HPIB (HP82169A) translator mode	2.6 kB/s
HP-71B + PIL-IO board (v2.0)	3.5 kB/s (no USB)
HP-71B + PIL-Box (v2.1, ILPer stopped/bypass mode)	3.5 kB/s (no USB traffic, as PIL-IO)
HP-71B + PIL-Box (v2.1, 230 kbps, ILPer running)	1.8 kB/s (optimized config.)

These tests reflect only the performance of the transport layer of the HP-IL, there is no data processing in the HP-IL devices.

How fast is HP-IL – PIL-Box optimization

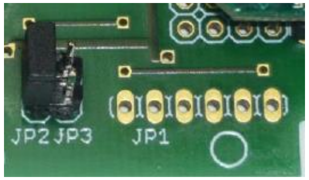
- The bottleneck for the PIL-Box performance is the USB link.
- USB is not efficient for the byte-per-byte communication of the HPIL.
- To get the best result, some optimization is needed.

- Use the benchmark sequence on HP-71B, to test the best configuration:

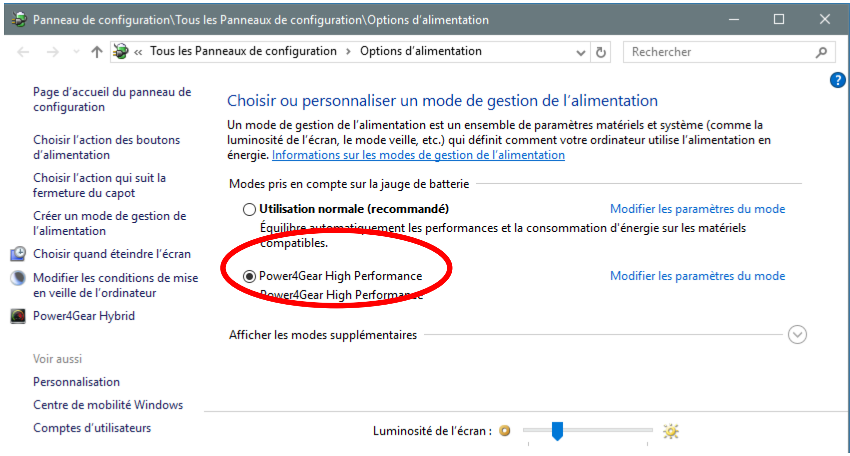
1 RESTORE IO @ SEND UNL MTA
2 T=TIME @ COPY HPILROM TO :LOOP @ T=TIME-T
3 DISP 16384/T;" bytes/s"

- Use the latest PIL-Box firmware (currently v2.1), try to use the 230 kbps com speed (jumper JP2 removed).
- Try the different USB ports.
- USB 3.0 ports may give better results.
- Try to use an external USB hub.
- Make sure the PC power option is using a 'High Performance' mode.

- Even with the best settings, performance is depending on each particular PC.



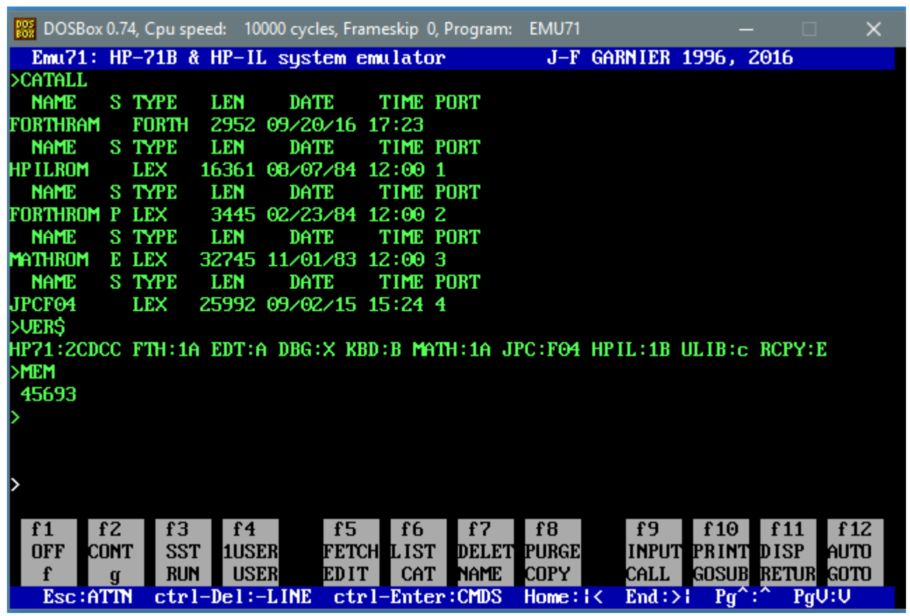
PIL-Box com setting (v2.1):
JP2 installed: 115 kbps
JP2 removed: 230 kbps




KRISTAL S.A.
informatique
HPIL 800
KRISTAL S.A.
informatique
The end…

Emu71/DOS: 20th Anniversary!

Emu71 v1.1 (1996) on a Windows-98 computer



Emu71 v2.45 (2016) in DOSBox on a Windows-10 computer

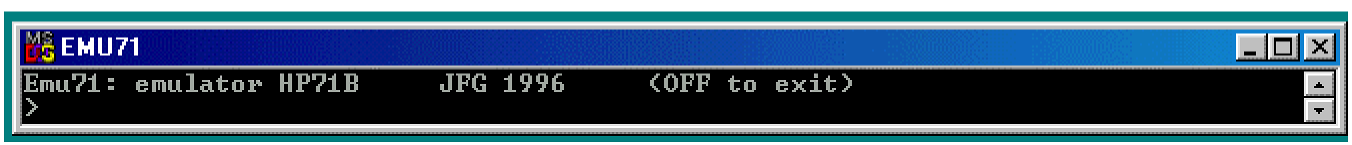
Emu71/DOS history notes

```
; **************************************************************
;              émulation du processeur Saturn par le 8086
;                   J-F Garnier 1993, 2004
; ----------------------------------------------------------------
; Module saturn.asm : définitions des registres, moteur d'exécution
; ----------------------------------------------------------------
; Version 0.0 16/03/93
;   prototype, faisabilité
; Version 1.0 04/11/95
; Version 2.0 07/09/01 extensions pour HP42
; Version 2.1 08/09/02 maj
; Version 3.0 03/10/02 12 bits jump table
```

- The idea of Emu71 started in … 1993
but work really started only on end of 1995

- Motivations:
  - build a HP-71B environment with the ability to run modules that I only owned as ROMCOPY images,
  - have lot of memory (at the time my configuration was a HP-71B + a HHP 32k RAM/EPROM module – not enough!),
  - and possibly be faster that the real thing.

- First stable, usable version in **1996**: single-line display emulation, but already full ROM and RAM module emulation.
About 3x the native HP-71B speed on a 386SX at 25 MHz



The Saturn emulation core

- Hand-coded, highly-optimized assembly code

- Heavy use of tables, macros, unrolled loops to get the highest speed. My PC of the time was a 386SX, I wasn't sure to get at least the HP-71B speed.

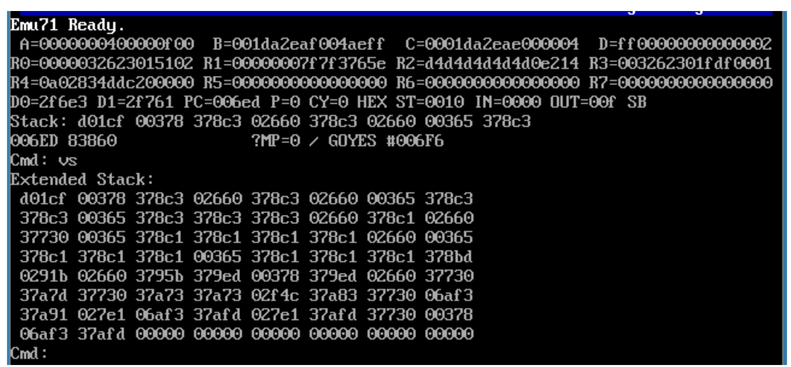
- Use the packed BCD format (2 nibbles per byte) and the Intel x86 native BCD support provided by the DAA/DAS opcodes, for speed (again).

- Emulation core source file (before macro expansions): ~ 200 kB

- Emulation core object code: ~ 30 kB,

- many hours (I didn't count…) of optimizing, testing, benchmarking on different machines and host CPU, …

- bonus:
extra R5-R7 registers and 64-level hardware stack:




```
; fonctions d'addition décimal

addd_byte       macro
                   mov   al,[di]
                   adc   al,[si]
                   daa
                   mov   [di],al
                   inc   si
                   inc   di
                 endm

_addd_hinib:      ;
                mov   al,[di]
                mov   ah,[si]
                mov   dl,al       ; copie
                and   ax,0f0f0h   ; masque
                add   al,ah
                daa
                lahf
                and   dl,0fh
                or    al,dl
                sahf
                mov   [di],al
                ret

_addd_16:       addd_byte
_addd_14:       addd_byte
_addd_12:       addd_byte
_addd_10:       addd_byte
_addd_8:        addd_byte
_addd_6:        addd_byte
_addd_4:        addd_byte
_addd_2:        addd_byte
                update_carry
                ret

_addd_15:       addd_byte
_addd_13:       addd_byte
_addd_11:       addd_byte
_addd_9:        addd_byte
_addd_7:        addd_byte
_addd_5:        addd_byte
_addd_3:        addd_byte
_addd_1:
_addd_lonib:
                mov   al,[di]
                mov   bh,al
                adc   al,[si]
                daa
                update_halfcarry
                mov   ah,bh
                and   ax,0f00fh
                or    al,ah
                mov   [di],al
                ret
```

Emu71/DOS specificities…

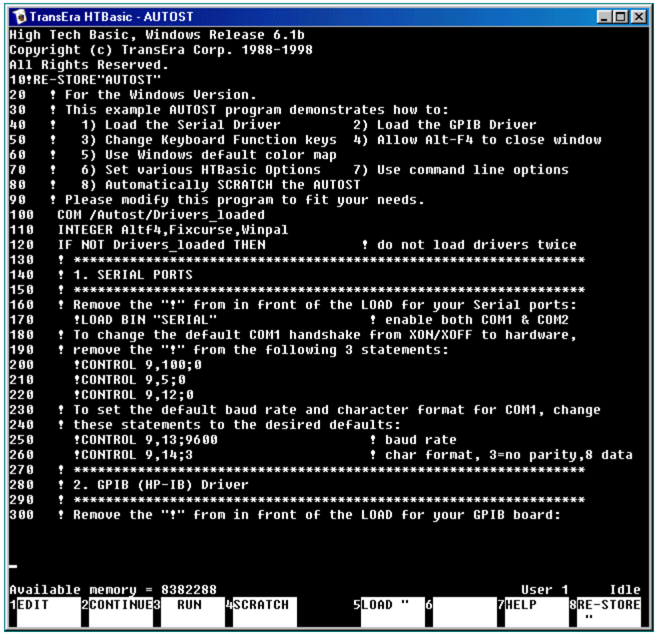
- Since speed was the top priority, some choice have been made since the beginning, especially for the I/O system emulation.

- As a result, Emu71/DOS is not an exact, accurate emulation at register level, especially for the display and keyboard management (the Diagnostic ROM fails). A benefit of this implementation is the 78-char display line instead of the 22-char LCD.

- Emu71/DOS also has some limitations on RAM modules (e.g. no support of 4 kB RAM modules, RAM ports other than port 0 not usable as main memory).

- Emu71/DOS is a text-mode, DOS-style application, and this will not change! The model of the Emu71/DOS user interface was the HP Basic aka RMB running on HP 9000 workstations series 200/300 (still existing as [HT Basic for Windows]).

- Emu71/DOS includes a few special features:
support of the 8 ports (0-7) managed by the HP-71B OS,
support of the DEBUG command of the Forth/Assembler ROM,
support of the EEPROM module type.



HTBasic - a model for the Emu71/DOS user interface

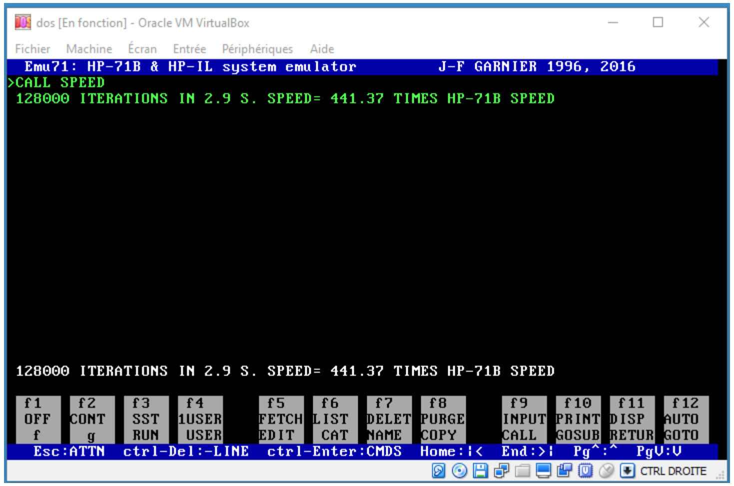
Emu71/DOS status in 2016

- Still usable:

Despite its 16-bit DOS nature, Emu71/DOS can still be run on modern 64-bit OS. DOSBox is the simplest solution. VirtualBox gives higher performance but is more complex.

- Still powerful:

x5 the HP-71B speed with DOSBox (2 emulation layers), x400+ the HP-71B speed with VirtualBox [1].



Emu71/DOS v2.45 in VirtualBox

- The HP-IL board and the PIL-Box are no more supported natively due to the modern OS constraints.

ILPer is now the natural gateway between Emu71/DOS and the real HP-71B.

- Still maintained:

The last update was rev 2.44 on dec. 2014, to support large HP-IL drives up to 16 MB.

- And for the first time, the full Emu71/DOS source files including the Saturn emulation core in assembly language are available (under GNU license).

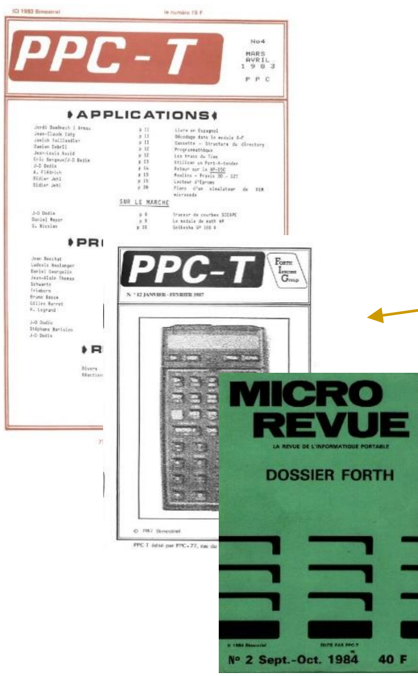
[1] Tested on a Intel core i3 1.8GHz computer.
Benchmark: number of empty loops done in 10s (FOR I=1 TO 1E6 @ NEXT I)

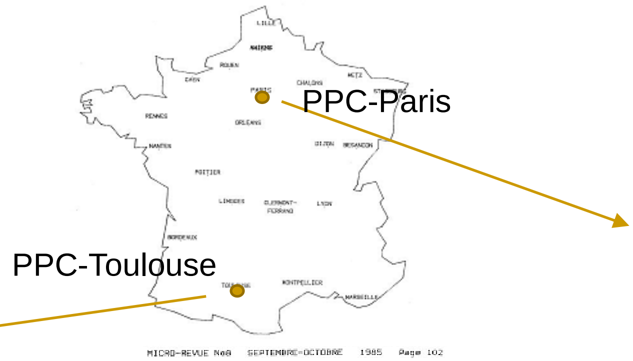
```
1 19171 6F4A                          _sto_C_D0_n:      exec_mem store C D0 n
2 19172 6F4A  BB 217Cr                                mov   bx, offset _regD0
2 19173 6F4D  BF 2116r                                mov   di, offset _regC
2 19174 6F50  E9 FD1D                                 jmp   exec_store_n
1 19175 6F53                          _lod_A_D1_n:      exec_mem load A D1 n
2 19176 6F53  BB 2180r                                mov   bx, offset _regD1
2 19177 6F56  BF 2102r                                mov   di, offset _regA
2 19178 6F59  E9 F6F6                                 jmp   exec_load_n
1 19179 6F5C                          _lod_C_D1_n:      exec_mem load C D1 n
2 19180 6F5C  BB 2180r                                mov   bx, offset _regD1
2 19
2 19
1 19183 6F65                          _sto_A_D1_n:      exec_mem store A D1 n
2 19184 6F65  BB 2180r                                mov   bx, offset _regD1
2 19185 6F68  BF 2102r                                mov   di, offset _regA
2 19186 6F6B  E9 FD02                                 jmp   exec_store_n
1 19187 6F6E                          _sto_C_D1_n:      exec_mem store C D1 n
2 19188 6F6E  BB 2180r                                mov   bx, offset _regD1
2 19189 6F71  BF 2116r                                mov   di, offset _regC
2 19190 6F74  E9 FCF9                                 jmp   exec_store_n
  19191
  19192
  19193                                               end
Turbo Assembler      Version 4.1      23/07/07 16:19:23      Page 338
Symbol Table
```

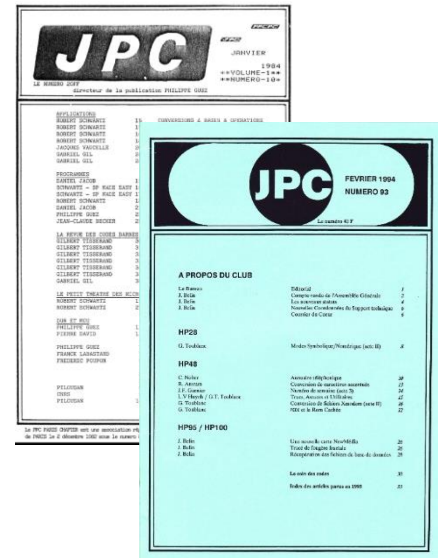
The Emu71/DOS page: http://www.jeffcalc.hp41.eu/emu71/index.html

The end…

The French HP User Club Journals (1982-1994)



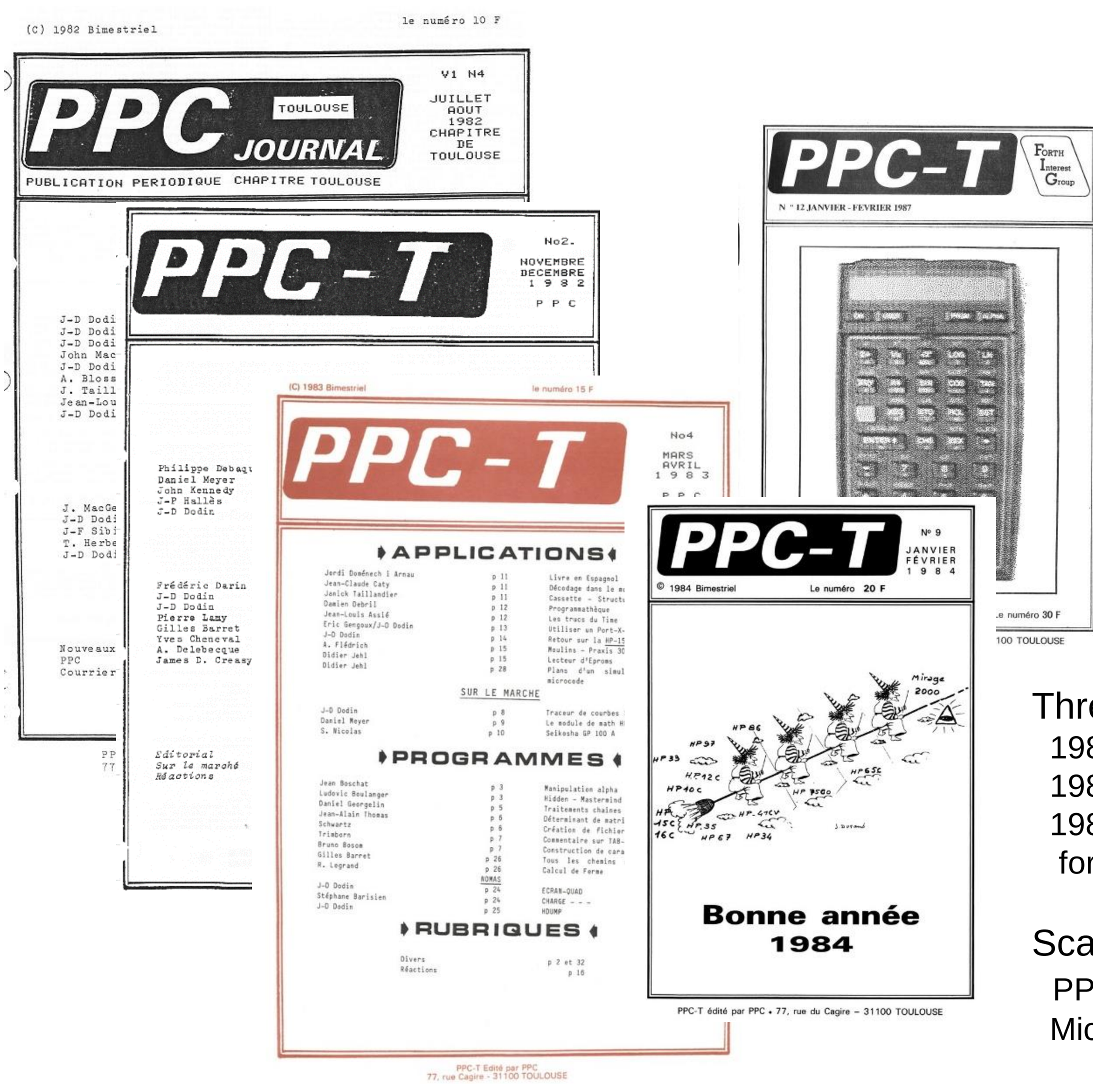




The French PPC Chapters
and the journals:
PPC-T
Micro-Revue
JPC

A few individuals started to scan the French PPC Journals.
This document is a status of the material already available.

Journals of PPC-Toulouse



Three periods over 6 years:

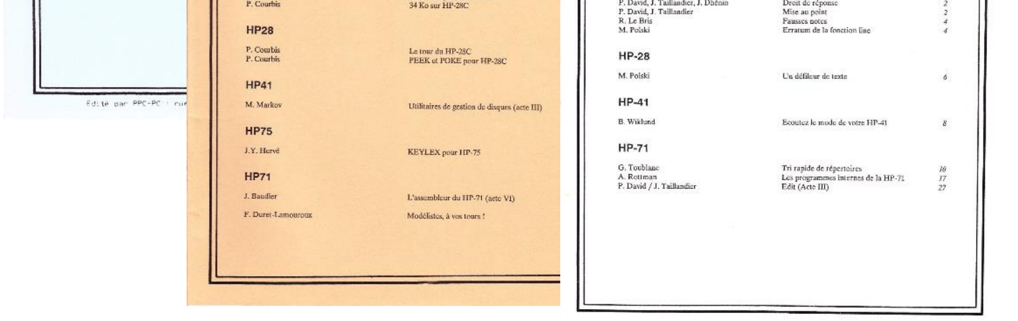
- 1982-1984: PPC-T, V1N1 to V1N4, then N1 to N11
- 1984-1986: Micro-Revue, N1 to N15
- 1987: last 3 issues of PPC-T, N12 to N14
- for a total of 33 issues

Scan status:

- PPC-T almost done [1],
- Micro-Revue only partially scanned (8 over 15) [2]

[1] Scanned by hpmaniac (http://hp41.net/forum/viewtopic.php?f=7&t=344) and myself
[2] Scanned by rogeroge (http://www.silicium.org/forum/viewtopic.php?f=35&t=38925)

JPC: Journal of PPC-Paris



Over 12 years and three periods:

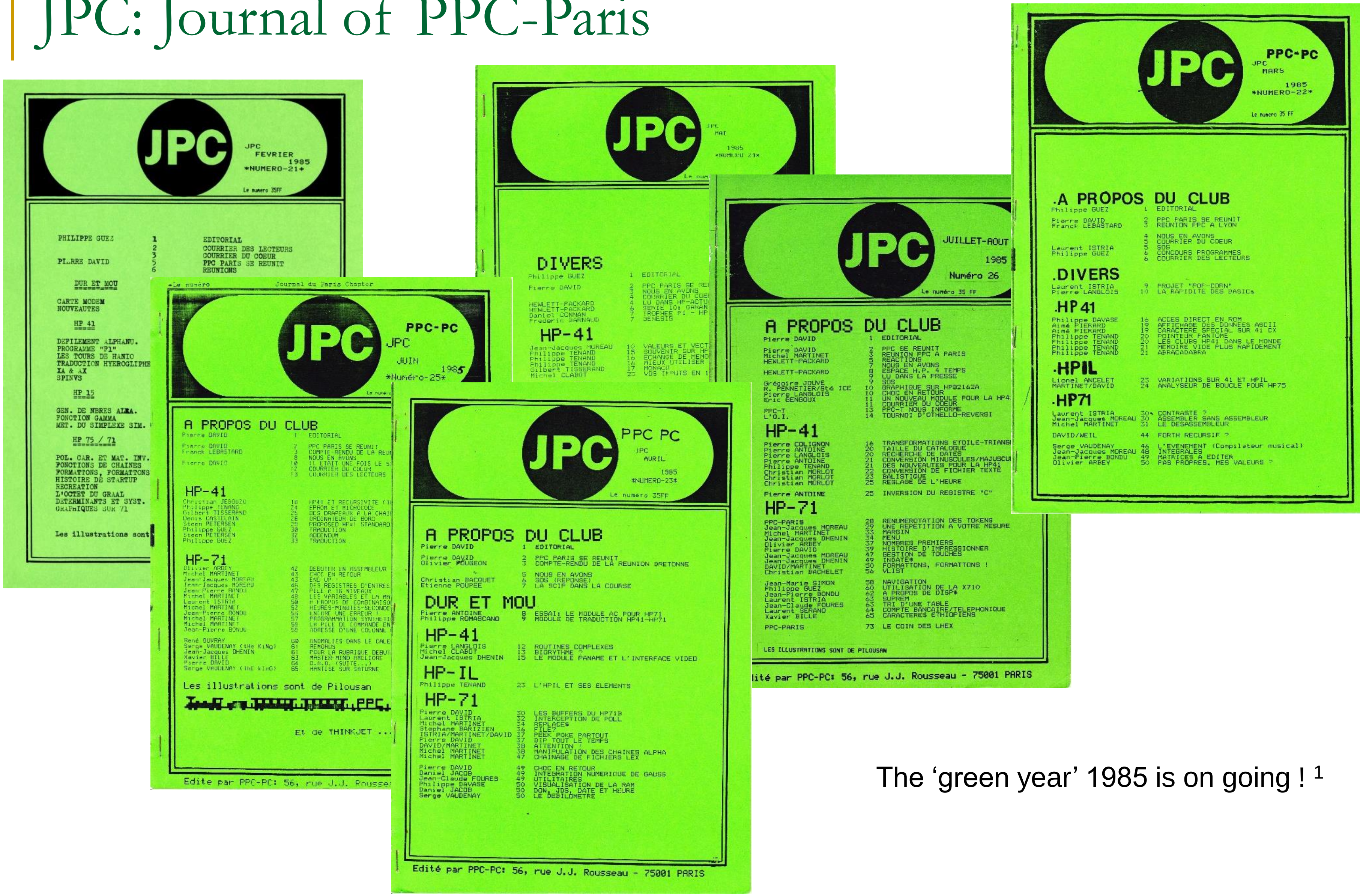
1983-1989: N0 to N70, ‘classic period’, mainly 41C/71B/75C,
1990-1991: N71 to N75 , low activity period, lower JPC quality
1992-1994: N76 to N99, ‘new period’, mainly 48S/G and 95LX
for a total of 96 issues (some merged multiple numbers)

Scan status:

1983-1984 done [1], score to date: 22 over 96…

[1] Scanned by hpmaniac (http://hp41.net/forum/viewtopic.php?f=4&t=179)

JPC: Journal of PPC-Paris



The ‘green year’ 1985 is on going ! [1]

[1] See http://www.hp41.net/forum/viewtopic.php?f=6&t=355

For more information on the PPC-T and JPC journals, see my french resource page http://www.jeffcalc.hp41.eu/divers/index.html