

MEMORANDUM

SUBJECT: PRINT I - An Interpretive Routine for Printing Numerical
Output from TX-0

DATE: 9 February 1960

General Description

PRINT I is available either as a subroutine to be converted, or as a utility tape which can be put in memory at a pre-specified location. It is 746₈ registers long, and with it a programmer can print out several kinds of numbers, in various formats, as desired.

Interpretive Routines

A standard TX-0 program contains a series of instructions stored in memory. The computer is designed to interpret these instructions in various ways. "Add x", for example, tells the computer to add the contents of x to whatever number is in the accumulator.

An interpretive routine is quite similar in design. You still have a series of orders stored in memory, but these orders are never placed in the memory buffer register. Instead, they are examined by another program, step by step, and the end result is that the computer seems to obey an entirely different set of instructions from those it is accustomed to seeing. Terms such as "cla", "tra x", and "pna" are no longer carried out, but terms such as "pcm", "pfl n", "plv x" are interpreted properly.

How To Use The Program

Entering the program is accomplished by typing the macro instruction "print". This occupies only two registers and is defined elsewhere in this memo. After typing "print", you may now use a new set of symbols, describing the manner in which you want data printed out. When you have finished using the print command, the pseudo-instruction "plv x" will cause control of the computer to be transferred unconditionally to register x, with a cleared accumulator.

Orders Available in PRINT I

After typing "print" followed by a carriage return, you may call for any of the following orders:

psd x
to y
(print Space suppressed
decimal)

The contents of the bank of registers from x to y will be printed out, horizontally, just as UT-3 does, except that initial zeros will be suppressed with spaces, and the numbers will be signed decimal.

Initial zeros are always suppressed by all these orders, but you have a choice of how you want this done.

pnd x to y (print <u>no</u> space suppressed <u>decimal</u>)	These orders print x to y as decimal numbers, but the numbers will be typed immediately, and initial zeros will be ignored completely rather than suppressed with spaces.
pso x to y	print <u>spaced</u> suppressed (signed) <u>octal</u>
pno x to y	print <u>no</u> space suppressed (signed) <u>octal</u>
psu x to y	print <u>spaced</u> suppressed <u>unsigned</u> octal
pnu x to y	print <u>no</u> space suppressed <u>unsigned</u> octal
psg x to y	print <u>space</u> suppressed unsigned decimal
png x to y	print <u>no</u> space suppressed unsigned decimal
pfr x to y	print, as decimal signed fractions, x to y, assuming the binary point to the right of bit zero. Bit zero is the sign bit, and the sign is changed by complementing the whole number. Normally, six figures to the right of the decimal point are typed. The last figure is not rounded off. That is, .3584289 would appear as .358428 to six place accuracy.

Format

When a command such as

```
pso x
to y
```

is given, the flexowriter does the following:

1. Prints carriage return.
2. Print $x \mid \cdot \rightarrow n_1 \rightarrow n_2 \rightarrow n_3 \dots \rightarrow n_8$
3. Does another carriage return, prints
 $(x+8) \mid \rightarrow n_9 \rightarrow n_{10} \rightarrow n_{11} \dots$ etc.
4. When the last register is printed out, a carriage return is made and PRINT I looks at the next order, the one after "to y".

The format can be modified in many ways. Above, 10_8 columns were printed out. The order

pcm n (print column)

changes the counter criteria, and now only n columns will be printed. This will remain so until another pcm order is provided, or until the PRINT I program is read in again, in which case 10_8 columns will be printed. Do not let n be greater than 10_8 .

The order

pbp (print by pass)

will by pass the program section that types signatures on the format. If this order is given, the very next time a format is specified there will be no register identification at the left hand side. The normal format program is automatically restored when this particular format has been completely printed.

If one wishes to by pass all of the format, the following procedure is used:

pso x

pso y

to z

The important order here is pso x. It is not followed by a "to y" order, but rather by an entirely new order. Whenever a printout is requested, PRINT I looks at the next order to see if it is an instruction. If anything but a "to y" order is in the next register (in this case, pso y is contained there), the print subroutine will immediately type out just the number without any carriage returns or tabs before or after. In the above case, the contents of x will be typed as a signed, space zero-suppressed octal number.

Other Orders

psf n specifies the number of significant figures used in fractions. n must be an octal number. This order permanently changes the program when given, until, of course, another psf order is given. If the psf instruction is not used, six figures will be typed.

noz is operate zero. This order does nothing, it is merely a space filler to enable one to modify his program by inserting or removing orders.

pfl n (n is an octal number) n flexo triplets will be typed.
The triplets must be stored immediately following the pfl order:

pfl 4
flexo thi
flexo s i
flexo s i
flexo t.

yields: this is it.

when the program is run.

Errors

Once PRINT I is properly entered it has several features built in so that it cannot destroy itself if given an illegal order (for example, cle). If an illegal order is given, the flexowriter will type ipi, (illegal print instruction) and the computer will stop. The accumulator will contain, as an add instruction, the register which contained the bad order. Pressing restart will cause the interpretive routine to skip the illegal instruction and look at the next one.

Summary of Orders Available

decimal:	signed	pad (spaces for initial zeros)
		pnd (nothing for initial zeros)
	unsigned	psg (space)
		pnd (nothing)
octal:	signed	pso (ditto)
		pno
	unsigned	psu
		pnu
fraction:		pfr (decimal fractions)
significant figures		
in fraction		psf n (n is octal)
do nothing		poz
flexo characters		pfl n (n is octal)
		followed by the flexo triplets
column modify		pcm n (octal n)
by pass address		
signature		pbp
leave subroutine		plv x (computer control goes to x)

Setting Up The Program

PRINT I is available as a binary tape, occupying registers 30_8 to 776_8 . One should make sure that his program does not use these registers. PRINT I must be read in along with the user's program.

In typing the macro tape one should type the following at the beginning:

```
define
    print
    llr .
    tra 30
    terminate
    psd=20000
    pnd=40000
    pso=60000
    pnc=100000
    psu=120000
    pnu=140000
    pfl=160000
    pbf=200000
    psg=220000
    png=240000
    pfr=260000
    olv=300000
    pcm=320000
    poz=340000
    psf=360000
    to=0
```

The programmer may now use any or all of the print orders, and may call for "print" as many times as he wishes.

PRINT I is also available as a subroutine which the user may convert and store wherever he chooses. The subroutine defines the following symbols:

```
define print
    llr .
    tra pit
    terminate,
```

all the orders (such as pso, pnd) defined above.

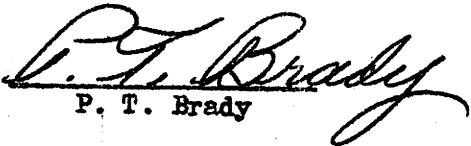
and also:

pit (starting address)
pie (finishing address).

Pit and pie are the only flads used in the program, and since PRINT I defines so few symbols, it may be more convenient for the programmer to use the English tape. "Print" is the only macro instruction defined, "constants" is used once. "And" is defined as opr 305. This will be changed when the new computer code is installed. The English tape contains its own title and ends with the order "start pit". It does not define a starting address, and it cannot be the very first tape of a series of tapes converted. The reader is urged to use the macro memo to see how to convert more than one tape on one run of MACRO II.

Attached is an example of a program written to use PRINT I. Registers 4000 to 4033 contain arbitrary numbers to be printed out.

Signed


P. T. Brady

FIB/dbh

Approved


J. B. Dennis

ADDITIONS TO THE PRINT I MEMORANDUM OF 3 MARCH 1960

P. Brady

Since the first writing of the memo on PRINT I, the program has been revised and contains a few new orders. The following orders may now be used:

psb n

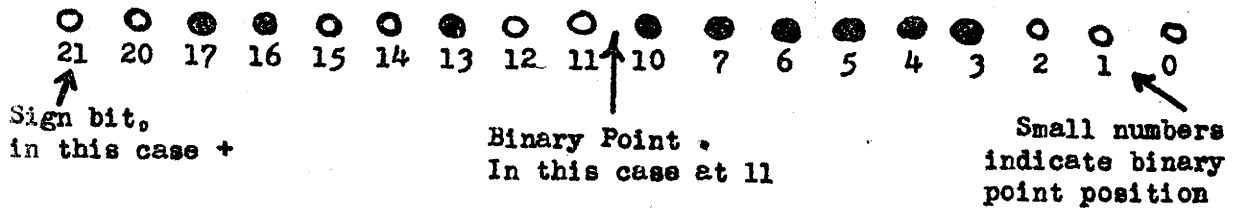
pms x

to y

pmn x

to y

These orders print mixed numbers. A mixed number is formed by setting a binary point in a register, calling everything to the left, except bit zero, the integer, and everything to the right a fraction.



The number shown in the example is read as follows:

Left of the binary point is

144 (octal) = 100 (decimal)

Right is 770 (octal) =

.5
.25
.125
.0625
.03125
+ .015625
.984375

The total number is

100.984375, decimal.

Complementing the entire number changes its sign.

The order psb n sets the binary point. The programmer must indicate this point before giving either of the other two orders. If he does not, PRINT I will assume the point to be at 11.

psb set binary point at n
pms print mixed, suppressed with spaces
pmn print mixed, suppressed with nothing
pms will also align decimal points in a column.

The binary point is restricted to lie within the range $21 \geq n \geq 0$. A very wrong answer will result if n goes out of this range.

In printing mixed numbers, only four columns (or less) may be printed. The order "pcm $0 < n \leq 4$ " should be given before a pms or pmn order. If not, the computer will halt on an in-out stop. You may restore eight columns afterwards if you wish.

If you have forgotten to do this, and you are using the binary tape, stop the computer and store "3" in register 176.

The PRINT I program itself is a little longer after revision. In its binary form it occupies 30_8 to 1020_8 . The English tape varies in length because of the way the constants table is assembled, but should not exceed 1000_8 registers.

The English tape no longer defines anything! A separate table consisting of definitions is available in the same box as the Binary tape and may be used with either the English or binary tape.

The following procedures are recommended when converting:

Using binary tape:

1. Read in a title tape of your program.
2. Read in "Print define".
3. Read in your own tape.
4. Be sure your own tape defines pit=30 and does not occupy registers 30 to 1020, inclusive.

Using English tape:

1. Read in a title tape of your program.
2. Read in "Print define".
3. If you read in "PRINT I English" now, your title tape must contain a starting address. If your own tape is read in now, followed by "PRINT I English", this is not necessary.
4. The English tape defines "pit" for you. It also defines "pie" as the last address.

The additional commands are:

psb = 400000
pms = 420000
pmn = 440000

In programming with PRINT I, you may program a series of orders such as:


```
pfl n
(n registers of flexo)
pcm 3
pso 4000
to
5000
png 3000
to
3017
```

When this is put into operation, you may suddenly decide that you are not really interested in the contents of 4000 to 5000 and you would prefer to skip to the pso order.

This is accomplished by pushing test, with tbr set to "trn pit+6" (trn 36 if using binary tape). If the flexo types "ipi" when you do this, ignore it and push either restart, or test, again.

You may think of a series of print orders as a stack of records on a record changer. They must be played in order, and you cannot skip any, but you may push "reject" whenever you wish.

This will not work if you are in the "pfl" mode. It is very important that you do not try this if the typewriter is printing flexo characters.

It is immaterial to this program whether "type in" is on or off.

Here are the contents of 4000 to 4034.

```
4000| 1
      12
      777777
      776545
      53
      -53
      -773
      545
4010| 0
      -0
      1
      -1
      3333
      776545
      1
      1
decimal 12
        10
        19
        1
        0
        -0
        87989
        128921
octal   200000 | fraction, .5
4030| 100000 | .25
      -20000 | -.5
      -100000 | -.25
```

Assume you want these numbers printed out in a variety of ways. A possible format would be as entered from your program, as follows.

```
      XXX
      XXX
      XXX
print
      pfl 5
      flexo reg
      flexo ist
      flexo 40
      flexo 00
      flexo =
      pnd 4000 | here it will print the contents of 4000 as a decimal
                | number suppressed with no spaces. Since this pnd
                | order is immediately followed by another order, no
                | format will appear.

      psd 4000
      to 4000
      pso 4000
      to 4017
      pno 4000
      to 4017
      poz | will do nothing here
      pfl 4
      flexo che
      flexo ck
      flexo pt
```

```

flexo 1
pfr 4030
to 4034
pbp
pno 4000
to 4017

```

| by pass the format signature

| notice that the signatures will be printed after this instruction, as this part of the program is automatically restored.

```

pcm 4
pso 4000
to 4017
psf 3
pfr 4030
to 4034
cla

```

| this is an illegal order. Restart must be pressed to continue the program.

```

pcm 10
psf 6
pfr 4030
to 4034
plv .+1
xxx
xxx
xxx

```

| leaves the interpretive routine

Here are the results from the on-line printer.

retist 4000 = 1

00	1											
4000	1	12	-	0	-	1232	53	-	53	-	773	545
4010	0	0		1	-	1	3333	-	1232		1	1
4000	1	12		-777777	-	-1232	53		-53		-773	545
4010	0	-0		1	-	-1	3333		-1232		1	1
check pt	1											
4030	.500000	.250000		-.500000	-	-.250000	.015373					
	1	12		-0	-	-1232	53		-53		-773	545
	0	-0		1	-	-1	3333		-1232		1	1
4000	1	12		0	-	1232						
4004	53	-	53	-	773		545					
4010	0	-	0		1	-	1					
4014	3333	-	1232		1		1					
4030	.500	.250		-.500	-	-.250						
4034	.015											
ip1												
4030	.500000	.250000		-.500000	-	-.250000	.015373					

(Program is finished.)