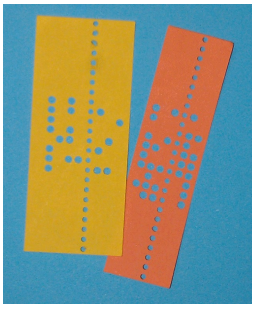


PTAP2DXF - Generate CNC-cut paper tapes from .PTAP or other binaries on a home stencil cutting machine
Version 1.1
Written by Steve Malikoff 2017 in Brisbane Australia.
https://github.com/1944GPW
Last updated 2019-04-19

Uses the free DxfMaker library written by David S Tufts https://twitter.com/davidscotttufts

Introduction

- Do you have a vintage computer and paper tape reader, but no punch?
- Do you need to make some repairs to an existing paper tape?
- Interested in punching other materials but concerned they might damage your punch?
- Are you looking for a quick way to visualise a paper tape (.ptap) file for SIMH?
- Would you like to produce paper tape banners from ASCII text?
- Do you want to make 5-level Baudot-encoded RTTY paper tape?
- Want to make your own custom-punched paper tape?

If your answer is 'yes' to any of these questions then PTAP2DXF is for you.

PTAP2DXF is a small open source command line utility that allows the user to make up to 8-level ASCII paper tapes similar to those punched by an ASR33 Teletype, using a common home CNC stencil cutter from binary images available on the internet, or your own files. For the default 8-level tape it makes, the output is one inch wide with a pattern of up to 8 data mark/space holes across as well as a smaller sprocket feed hole. The rows are spaced at 0.1 inch apart.

Its primary use is to make small tapes for loading software such as Absolute Loaders, MAINDECs (diagnostics) or other programs into vintage computers such as the Digital Equipment Corporation PDP-11, Altair, IMSAI or the like. These images are commonly found with a .PTAP, .PTP,, .TAP, .BIN or other extension and are often used with the SIMH simulator. PTAP2DXF does not actually care about file types as it treats everything as binary. In some situations the binary may first need to be converted using a special utility to a paper tape image.
In addition PTAP2DXF can be used for fun purposes such as punching letters and words along the tape in an upper case 8x8 font, or as a text label punched as a prefixed description at the start of a binary tape. It can make 11/16”-wide 5-level tape for an old Baudot machine. You could even experiment making tapes from various types of plastic film or sheet.

It has plenty of limitations. For one, it's very slow compared to a real paper tape punch and a whole lot more painstaking to produce owing to short lengths and the additional effort of having to join these together. It also needs a stencil cutter with DXF import ability, and some experimentation to get a good result. And it has numerous bugs perhaps to be fixed in a future release.

Requirements

- A Windows, Linux or macOS computer
- The .NET framework runtime package

- For Windows, the .NET framework installed can be as early as 3.0 right through to 4.7 as no special late-version C# constructs have been used.
- For Linux or Macintosh, install the .NET Core runtime from `https://www.microsoft.com/net/core`
- A DXF viewer is highly desirable but not essential, to examine output before cutting. Inkscape is ideal and free to download.
- A CNC home stencil / vinyl cutting machine. I use a 2015 model Silhouette Cameo 2 with a 12-inch x 12-inch sticky cutting mat.
- A sharp(!) cutting blade fitted, best if new. A carbide blade is recommended.
- Studio software for the CNC cutter that can import DXF files, usually supplied with your machine. I use Silhouette Studio Designer Edition V3 that came with the machine.

Using the program

All output shown here has been produced using the standard DOS command shell, the Git bash shell and the Linux Mint shell. After successful compilation *(see code for instructions on how to build for various platforms: Windows, Linux, Mac)* run the program with no arguments to get the usage printed out, which should look similar to the following:

```
C:\Users\you>ptap2dxf.exe
PTAP2DXF - Generate DXF files from teletype punched paper tape binary images, suitable for home CNC
stencil cutting
Written in C# by Steve Malikoff 2017 in Brisbane, Australia. Uses the DxfMaker library written by
David S. Tufts
See https://github.com/1944GPW for more details
Usage:
ptap2dxf [inputfilename.ptap]                        (Input ASCII file to be punched. Same as
                                                      --INPUT="/path/to/inputfile")
         [--ASCII]                                   (Show ASCII character representation for row on
                                                      console output)
         [--BANNERFILE=/path/to/bannerfile]          (Generate uppercase punched banner in 8x8 font
                                                      from ASCII file contents)
         [--BANNERTEXT="YOUR TEXT"]                  (Generate uppercase punched banner in 8x8 font
                                                      from string)
         [--BAUDOT]                                  (convert ASCII characters to Baudot. Forces 5-
                                                      level output)
         [--CONTROL-CHARS]                           (Show control characters on console output)
         [--DRYRUN]                                  (Run everything but do not generate DXF file(s))
         [--FLIP]                                    (Invert bit pattern. Logical NOT)
         [--GAP=n]                                   (Inter-segment gap in mm between each paper
                                                      segment on CNC cutting mat. Default is 0,
                                                      ie. shared edges with no gap)
         [--HELP]                                    (or ? prints this help)
         [--INPUT=/path/to/inputfile]                (.ptap or any binary or ASCII input file.
                                                      Optional switch, does not need to be given
                                                      with filename)
         [--JOINER]                                  (Make adhesive joiners for paper segments)
         [--LEADER=n]                                (Prefix output with blank sprocket punch tape in
                                                      1/10 inch increments eg. 240 is 2 feet.
                                                      aka /HEADER=)
         [--LEVEL=n]                                 (The number of data bits in a row of holes.
                                                      Default is 8 for byte-width ASCII 8-level.
                                                      Use 5 for 5-level)
         [--MARK=c]                                  (Console output character to represent a mark
                                                      (data bit = 1). Default is 'O'
         [--MIRROR]                                  (Reverse the output mark/space bit pattern to
                                                      right-left)
         [--NUMBER=BANNER|LEADER|CODE|TRAILER|ALL]   (NOTE: --N defaults to number the code lines
                                                      only)
         [--OUTPUT=/path/to/outputfile.dxf]          (output DXF file)
         [--PARITY={NONE, EVEN, ODD}]                Parity, if desired. Defaults to NONE

         [--PERDXF=n]                                (Fill CNC cutting mat with this number of 1-inch-
                                                      wide (for 8-level) segment strips across before
                                                      starting another. 5-level = 11/16-inch)
         [--QUIET]                                   (do not write any console output)
```

```
        [--RANGE=n,[L [-p] [+-z]]       (Start generation at byte n and run for following
                                         length L or previous p or prefix/suffix z bytes)
        [--SEGMENT=n]                   (Length in 0.1 inch increments for one vertical-
                                         cut paper strip before generating adjacent
                                         segment)
        [--SPACE=c]                     (Console output character to represent a space
                                         (data bit = 0). Default is ' '
        [--SPROCKET=n]                  (Sprocket feed hole position. Default is 3 for
                                         between 3rd and 4th data bit holes starting from
                                         right)
        [--TEXT="YOUR TEXT"]            (Input text string to be punched, taken from the
                                         command line)
        [--TRAILER=n]                   (Suffix output with blank sprocket punch tape in
                                         0.1 inch increments eg. 120 is 1 foot)
        [--VERSION]                     (Version number)
        [--WAIT]                        (Pause for Enter on console after running)
C:\Users\you>
```

Making a sample tape

Firstly, an example to get a piece of "real" punched paper tape into your hands in a few steps.

This will be to make a tape identical to the sample on the Wikipedia page `https://en.wikipedia.org/wiki/Punched_tape` that shows the word 'Wikipedia' followed by a <CR> and <LF>. You should find this as a tiny 16-byte file '`WikipediaCRLF.ptap`' in the PTAP2DXF Documents folder.

(If not present, go to the marvellous browser-based hex editor at `https://hexed.it/` and do 'New File', set 16 bytes, and enter characters in the right-hand column for 'Wikipedia' Ctrl+M Ctrl+J then export the file, rename to `WikipediaCRLF.ptap` and copy into the current directory)

Run the program, with the file as input:

```
C:\Users\you>ptap2dxf WikipediaCRLF.ptap
+---------+
| O O .OOO|
| OO O.  O|
| OO O. OO|
| OO O.  O|
| OOO .   |
| OO  .O O|
| OO  .O  |
| OO O.  O|
| OO  .  O|
|    O.O O|
|    O. O |
|      .  |
|      .  |
|      .  |
|      .  |
|      .  |
+---------+
Joiner 0000: data byte 00000000  absolute position 00000016
C:\Users\you>
```

Compare this console output with the photo on the Wiki page. Ignore the '`Joiner n: ...`' line for the time being.

The input file is the only argument that does not have to be preceded by a switch, although for consistency's sake the `--input` argument does the same thing.
The console output shows each row of how the paper tape should look. A '**O**' represents a punched hole or binary 1, space a binary zero (this can be changed on the console output) and '**.**' always

represents the sprocket feed hole.
This is the basic output on the console, but it's nice to have a little extra information such as the byte offsets as well as the ASCII printable or control character for each row.

Run it again with

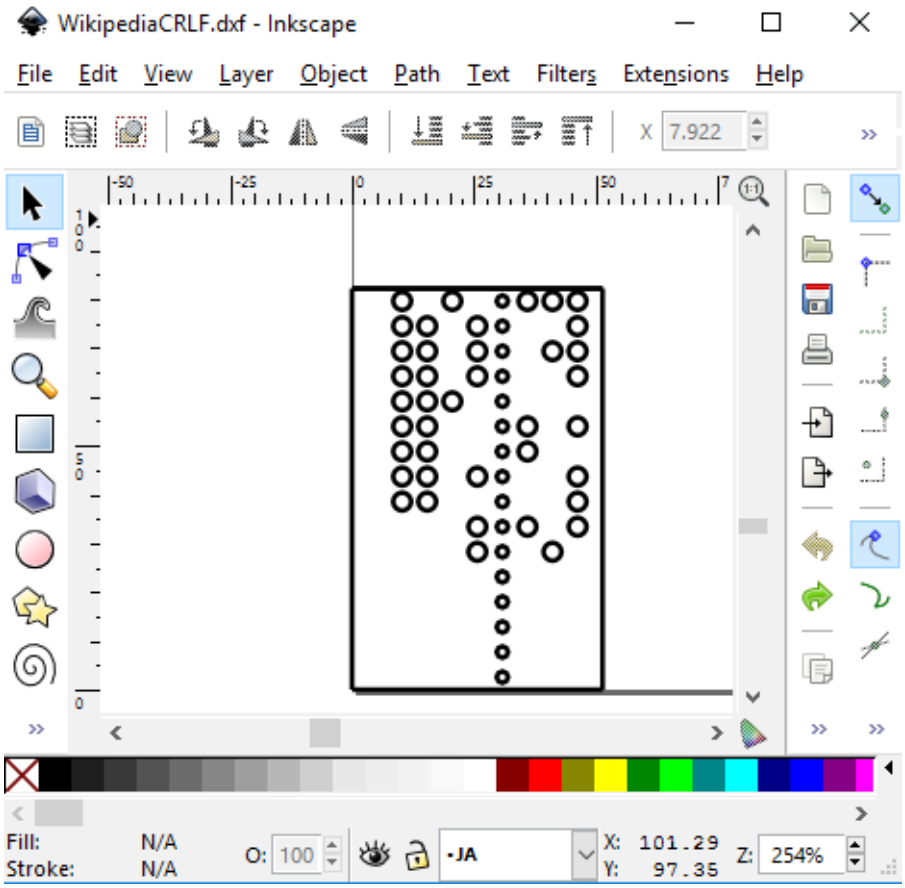
```
C:\Users\you>ptap2dxf WikipediaCRLF.ptap --number --ascii --control
           +---------+
#00000000  | O O .OOO|    W
#00000001  | OO O.  O|    i
#00000002  | OO O. OO|    k
#00000003  | OO O.  O|    i
#00000004  | OOO .   |    p
#00000005  | OO  .O O|    e
#00000006  | OO  .O  |    d
#00000007  | OO O.  O|    i
#00000008  | OO  .  O|    a
#00000009  |    O.O O|    <CR>
#00000010  |    O. O |    <LF>
#00000011  |      .  |    <NUL>
#00000012  |      .  |    <NUL>
#00000013  |      .  |    <NUL>
#00000014  |      .  |    <NUL>
#00000015  |      .  |    <NUL>
           +---------+
Joiner 0000: data byte 00000000  absolute position 00000016
C:\Users\you>
```

Note that even though the #number looks like a line number, it is actually the byte offset from the start of the data rather than line number (which would start at 1) although you can think of it as the line (row) number by mentally adding one.

In addition, you should now find a generated `WikipediaCRLF.dxf` (Drawing Interchange File) output, which is what this program is all about. If no filename output is specified then the same name is used as the input except the filetype becomes '.dxf'.

Here is where a DXF Viewer is handy. I normally use a CAD program to manage DXFs but **Inkscape** is excellent and free to download from `https://inkscape.org/` and I will use it here.

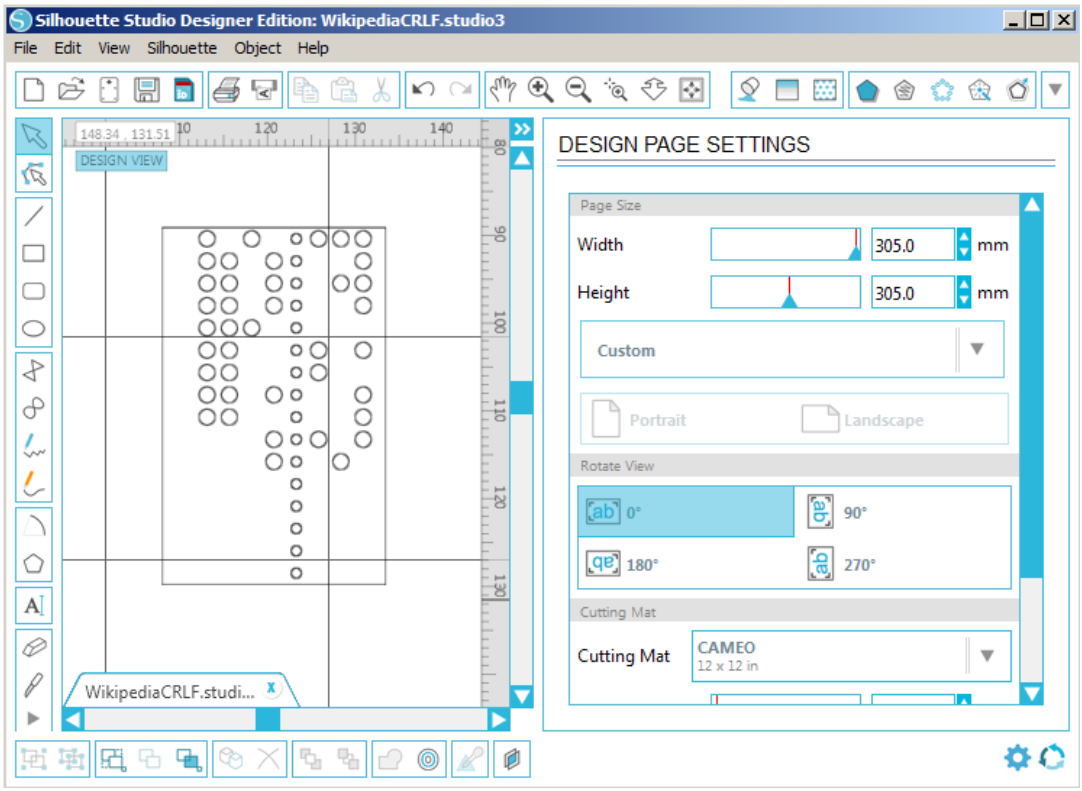
Double-click the `WikipediaCRLF.dxf` file and OK the default scaling dialog, then zoom in to the lower left corner. We should see the following:



Now compare this again with the Wikipedia image.

Producing the cut

Load the DXF into the CNC stencil cutting software. You may need to Select All and drag it to a more convenient position on the cutting mat. In Silhouette Studio Designer V3, it looks like:



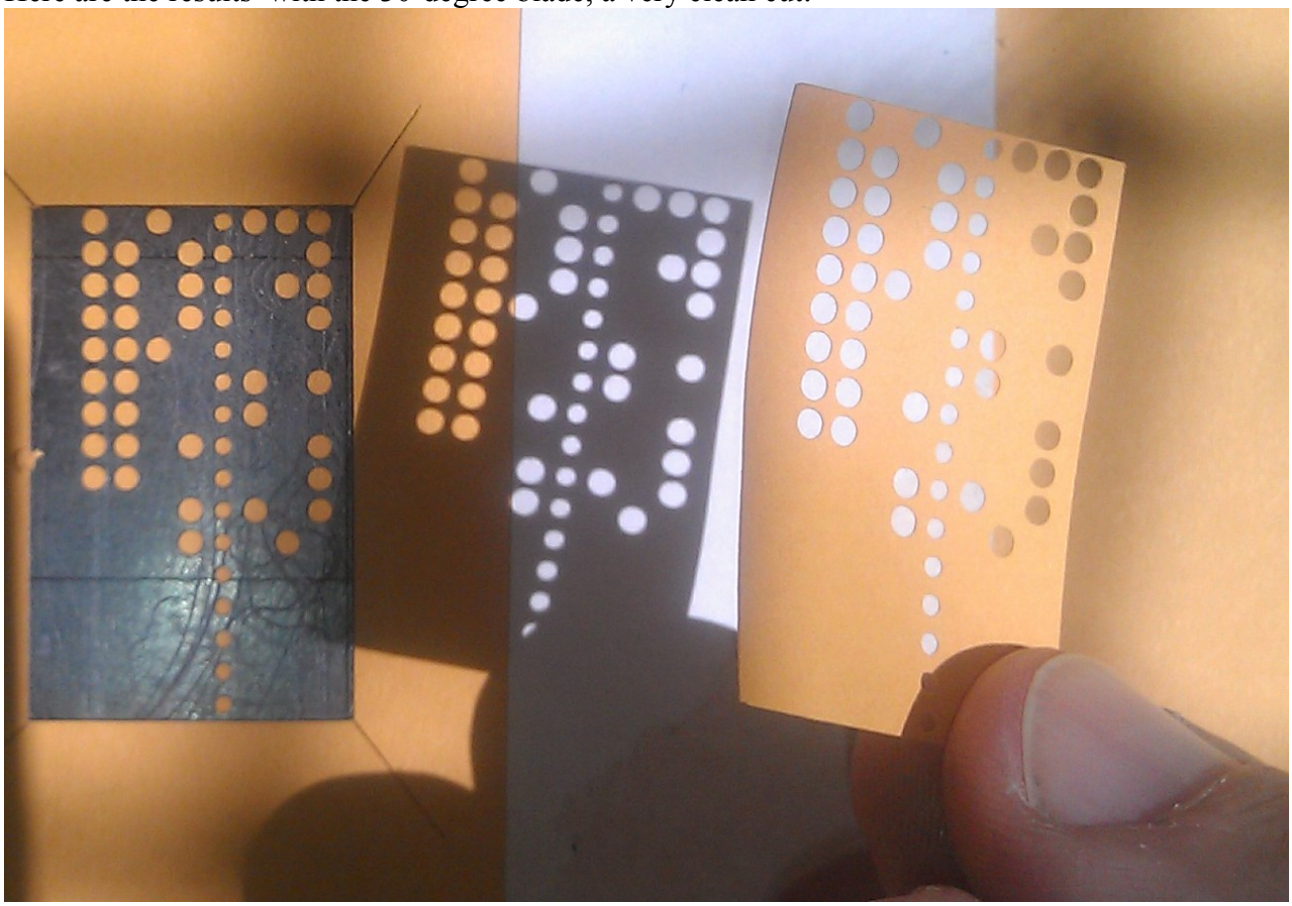
Use a sharp blade. Since this job has many very small diameter holes of which every one needs to be cut properly, it is important to get the correct blade set up with the correct cutting speed and depth, whether to select double-cut and of course suitable paper stock.

Cutting blades usually come in three models, specified by the angle between the paper surface and the cutting edge. They are generally made from tool steel or carbide-tipped. I find the carbide bits to be long-lasting although they are more brittle and the tiny tip of the blade can break off under heavy use.
These cutting blades can be purchased from the machine supplier or second-sourced off eBay. I've bought:

- A general-purpose 45-degree blade (fitted with a red plastic protective cap)
- A more 'pointy' 60-degree blade (blue cap)
- A shallower 30-degree blade for cutting large pieces of vinyl (yellow cap)

I had thought the pointy 60-degree blade would be the best for the many small-diameter holes, but surprisingly have had the best results with the shallow 30-degree bit, which by virtue of the angle is less likely to break than a pointier blade.

Here are the results with the 30-degree blade, a very clean cut:



Leader and trailer

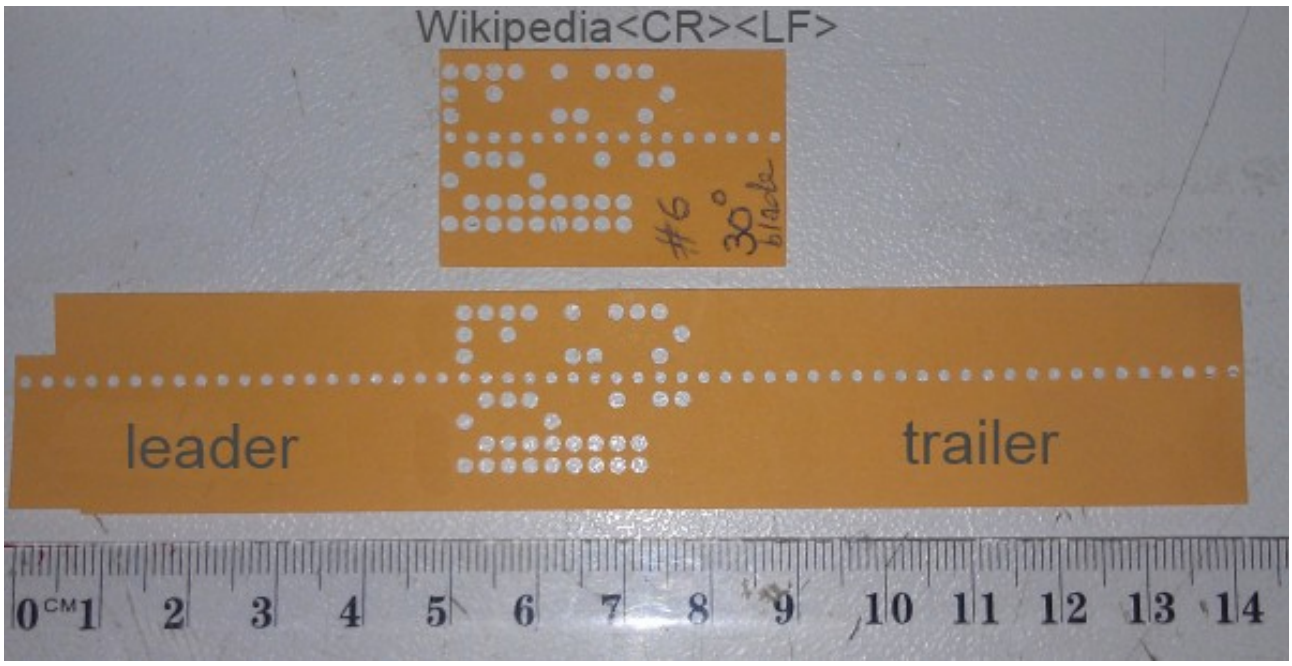
Now the problem with this sample is that there is no leader before the data starts, and only 5 rows of sprocket-feed trailer afterwards. It would be very finicky to load into a paper tape reader device as it stands.
Lets add 2 inches of leader feed and two inches of trailer feed to the output. This is specified in 0.1" increments, ie. Multiples of the normal row spacing:

```
C:\Users\you>ptap2dxf WikipediaCRLF.ptap --number --ascii --control --leader=20
--trailer=20
           +---------+
           |    .    |    <NUL>
           |    .    |    <NUL>
           |    .    |    <NUL>
           |    .    |    <NUL>
           |    .    |    <NUL>
           |    .    |    <NUL>
           |    .    |    <NUL>
           |    .    |    <NUL>
           |    .    |    <NUL>
           |    .    |    <NUL>
           |    .    |    <NUL>
           |    .    |    <NUL>
           |    .    |    <NUL>
```

```
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
#00000000  | O O .OOO|    W
#00000001  | OO O.  O|    i
#00000002  | OO O. OO|    k
#00000003  | OO O.  O|    i
#00000004  | OOO .   |    p
#00000005  | OO  .O O|    e
#00000006  | OO  .O  |    d
#00000007  | OO O.  O|    i
#00000008  | OO  .  O|    a
#00000009  |    O.O O|    <CR>
#00000010  |    O. O |    <LF>
#00000011  |      .   |    <NUL>
#00000012  |      .   |    <NUL>
#00000013  |      .   |    <NUL>
#00000014  |      .   |    <NUL>
#00000015  |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           |      .   |    <NUL>
           +----------+
Joiner 0000: data byte 00000000  absolute position 00000036
C:\Users\you>
```

Now we have two inches of leader and trailer, enough to actually load the tape on say an ASR33 Teletype, a CNC tape reader or manually pull through an optical reader like the vintage Oliver Audio Engineering OP-80A paper tape reader.
The leader also serves as a space to write or paste identifying labels on the tape. Notice also how only the data rows are numbered as the byte offsets for this 16-byte file. The leader and trailer do not have any effect on the data byte numbering by default, only the absolute number position.

Obviously you should look at it in Inkscape but I'll skip the DXF viewing and CNC mat positioning and go straight to the final cut result with the previous example for comparison (ignore the cut-out corner, it was a scrap piece):



Now let's generate a paper tape for the PDP-11 absolute loader, **`DEC-11-L2PC-PO.ptap`** which is 183 bytes long. You can find this tape image under `http://www.avitech.com.au/mm-files/ptb/` if you don't find it included as an example here with PTAP2DXF in the Documentation directory. I've left the leader and trailer off it for brevity below, but will add 10 inches for each for the actual cutting phase.

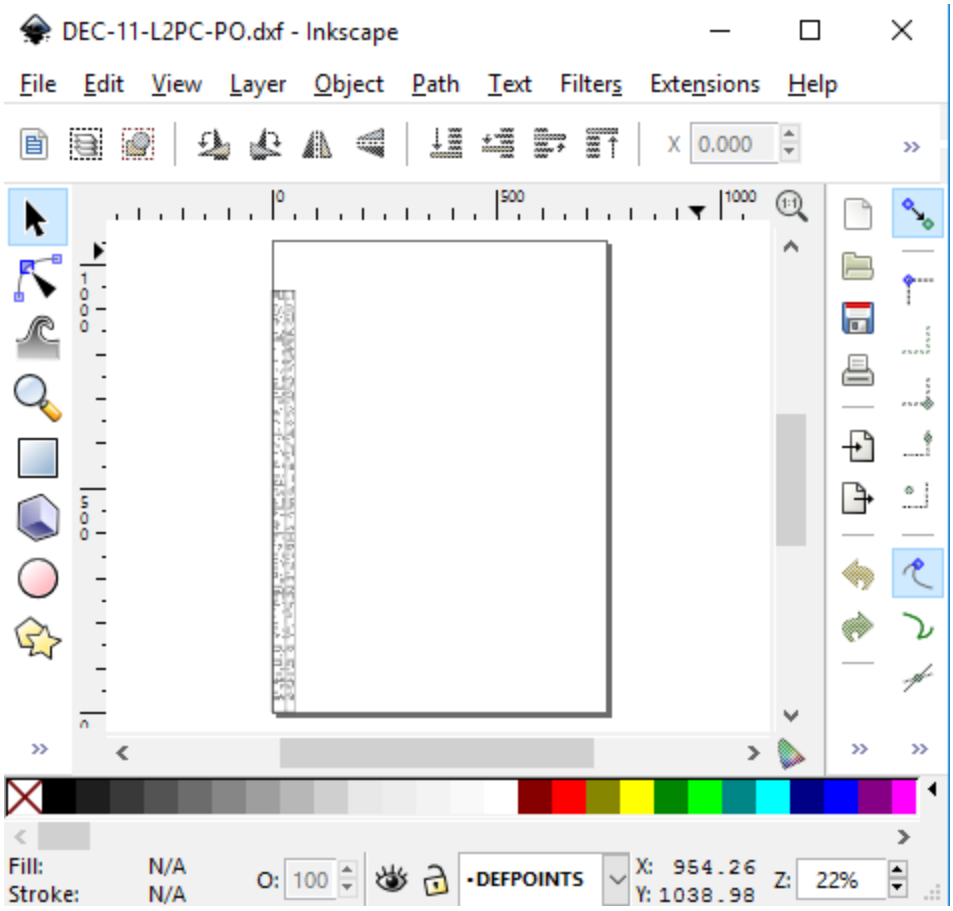
```
C:\Users\you>ptap2dxf DEC-11-L2PC-PO.ptap --number --ascii --control
           +---------+
#00000000  |OOO O.  O|
#00000001  |OOO O.  O|
#00000002  |OOO O.  O|
#00000003  |  OOO.O O|
#00000004  |     .   |    <NUL>
#00000005  |     .   |    <NUL>
#00000006  |OO   .OO |
#00000007  |   O .  O|    <DC1>
#00000008  |O O  .OO |
#00000009  |  O O.  O|    )
#00000010  |OO   .O O|
#00000011  |   O .  O|    <DC1>
#00000012  |OO   .O O|
#00000013  | OO  .O O|    e
#00000014  | O  O. O |    J
#00000015  |     .   |    <NUL>
#00000016  |     .  O|    <SOH>
#00000017  |    O. O |    <LF>
#00000018  |OO  O.OO |
#00000019  |   O .OOO|    <ETB>
#00000020  | OOOO.   |    x
#00000021  |OOOOO.OOO|    RUBOUT
#00000022  |    O.OO |    <SO>
#00000023  |    O.O  |    <FF>
#00000024  |     . O |    <STX>
#00000025  |O    .OOO|
#00000026  |    O.OO |    <SO>
#00000027  |    O. O |    <LF>
#00000028  |     . OO|    <ETX>
#00000029  |     .  O|    <SOH>
```

```
#00000030  |OO  O.OO |
#00000031  |    O.O  |    <FF>
#00000032  |      .  O|    <SOH>
#00000033  |      . O |    <STX>
#00000034  | O  O.OO |    N
#00000035  |   O .   |    <DLE>
#00000036  |      .   |    <NUL>
#00000037  |    O. O |    <LF>
#00000038  |OO  O.O O|
#00000039  |    O.  O|    <HT>
#00000040  |OO   . OO|
#00000041  |O   O. O |
#00000042  |OOOOO.O  |
#00000043  |      . O |    <STX>
#00000044  |OO  O.O O|
#00000045  |    O.  O|    <HT>
#00000046  |OOOO .OOO|
#00000047  |    O.  O|    <HT>
#00000048  |  OOO.O  |
#00000049  |      .   |    <NUL>
#00000050  |      . O |    <STX>
#00000051  |   O .  O|    <DC1>
#00000052  |OO   . O |
#00000053  |OOO  .O O|
#00000054  |      .O  |    <EOT>
#00000055  |      .   |    <NUL>
#00000056  |OO   . O |
#00000057  |  O  .O O|    %
#00000058  |      . O |    <STX>
#00000059  |      .   |    <NUL>
#00000060  |  O  .  O|    !
#00000061  |      . OO|    <ETX>
#00000062  |OOOO .OOO|
#00000063  |    O.  O|    <HT>
#00000064  |  O O.O  |    ,
#00000065  |      .   |    <NUL>
#00000066  |O    .O  |
#00000067  | OO  . OO|    c
#00000068  |      .  O|    <SOH>
#00000069  |   O .  O|    <DC1>
#00000070  |OO  O.O O|
#00000071  |    O.  O|    <HT>
#00000072  |     .O  |    <EOT>
#00000073  |     .O  |    <EOT>
#00000074  |OO   .   |
#00000075  |O   O. OO|
#00000076  |OOO O. OO|
#00000077  |      . OO|    <ETX>
#00000078  |      .   |    <NUL>
#00000079  |      .   |    <NUL>
#00000080  |OOO O.  O|
#00000081  |      .  O|    <SOH>
#00000082  |OO O .  O|
#00000083  |O  O .   |
#00000084  |OOOOO.   |
#00000085  |      .  O|    <SOH>
#00000086  |OO   . OO|
#00000087  |   OO.O O|    <GS>
#00000088  | OO O. O |    j
#00000089  |      .   |    <NUL>
#00000090  |O   O. OO|
#00000091  |O   O. O |
#00000092  |OO  O. OO|
#00000093  |O   O. OO|
#00000094  |OOOOO.OO |
#00000095  |O    .   |
#00000096  |OO   . OO|
#00000097  |O  OO.O  |
```

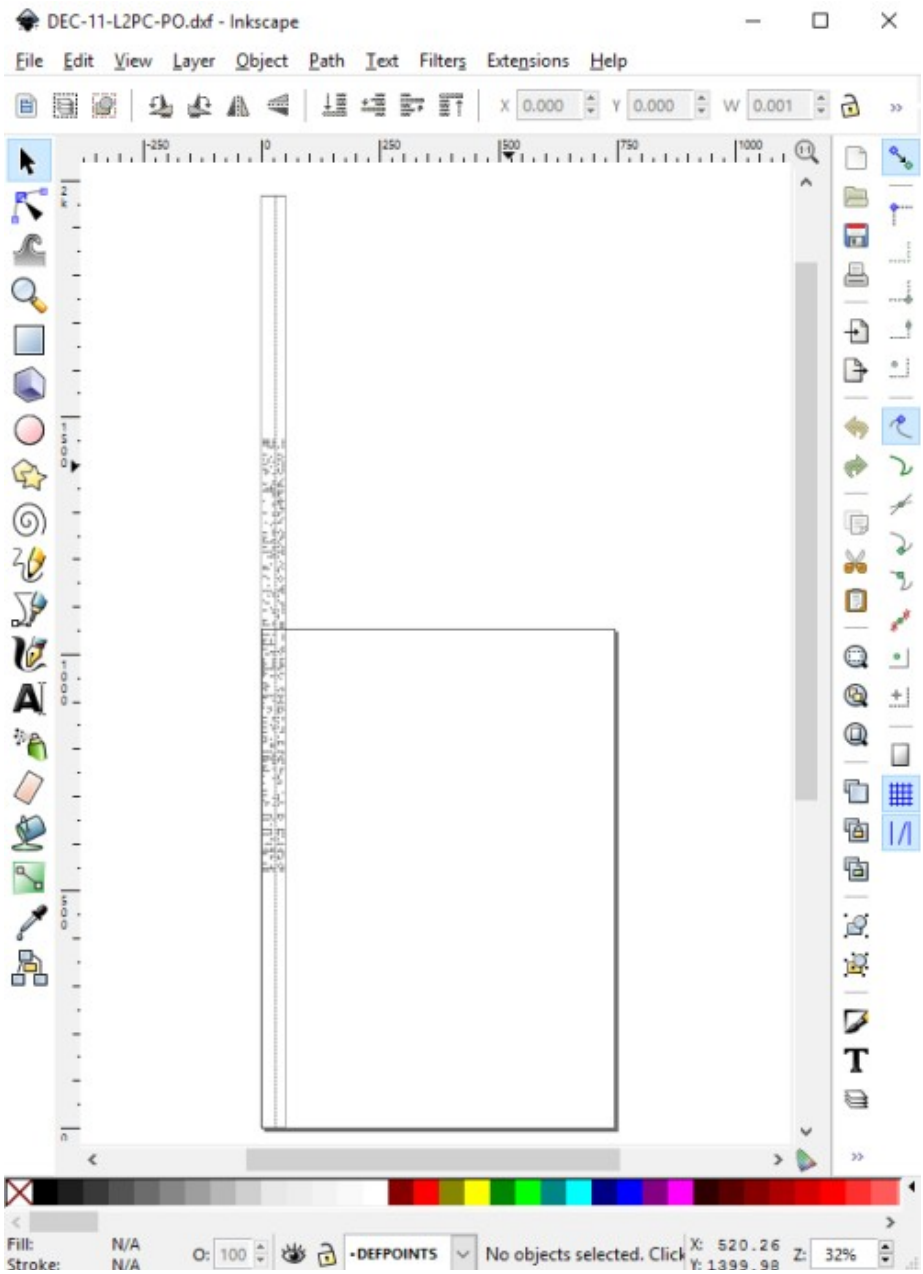
```
#00000098  |      . O |    <STX>
#00000099  |      .   |    <NUL>
#00000100  |OO    .   |
#00000101  | OO   .   |
#00000102  |OO    . OO|
#00000103  | O    .O O|    E
#00000104  |      .   |    <NUL>
#00000105  |OOOOO.OOO|    RUBOUT
#00000106  |OO    . O |
#00000107  |    O. O |    <LF>
#00000108  |O     .OOO|
#00000109  |      .   |    <NUL>
#00000110  |O OO  .OOO|
#00000111  |   O .O O|    <NAK>
#00000112  |  O   .OO |    &
#00000113  |      .   |    <NUL>
#00000114  |OO  O.O O|
#00000115  |    O.  O|    <HT>
#00000116  |OO    .O  |
#00000117  |   O .    |    <DLE>
#00000118  |OO  O.O O|
#00000119  |    O.  O|    <HT>
#00000120  |OO    . OO|
#00000121  |      .   |    <NUL>
#00000122  |OO    .O  |
#00000123  | O O .    |    P
#00000124  |OO    .OOO|
#00000125  |   OO.O O|    <GS>
#00000126  |   OO.    |    <CAN>
#00000127  |      .   |    <NUL>
#00000128  |OOOO .OOO|
#00000129  |    O.  O|    <HT>
#00000130  |OOO O. O |
#00000131  |OOOOO.OOO|    RUBOUT
#00000132  |OO  O.O O|
#00000133  |    O.  O|    <HT>
#00000134  |OO    .   |
#00000135  |O   O. OO|
#00000136  |OOO  . O |
#00000137  |      . O |    <STX>
#00000138  |O     .O  |
#00000139  |    O.O  |    <FF>
#00000140  |      . O |    <STX>
#00000141  |O     .OO |
#00000142  |      .   |    <NUL>
#00000143  |      .   |    <NUL>
#00000144  |OO    .   |
#00000145  |      .  O|    <SOH>
#00000146  |OO    .O  |
#00000147  |    O.O  |    <FF>
#00000148  |O     .O  |
#00000149  | OO   . OO|    c
#00000150  | O  O.O  |    L
#00000151  |      .   |    <NUL>
#00000152  |      .   |    <NUL>
#00000153  |      .   |    <NUL>
#00000154  |OOOO .OOO|
#00000155  |   O .O O|    <NAK>
#00000156  |OOO O. O |
#00000157  |      .   |    <NUL>
#00000158  |   O .    |    <DLE>
#00000159  |      .   |    <NUL>
#00000160  |OOOO .OOO|
#00000161  |   O .O O|    <NAK>
#00000162  |OOOO .O O|
#00000163  |      .  O|    <SOH>
#00000164  |   OO.O  |    <FS>
#00000165  |      .   |    <NUL>
```

```
#00000166  | OOO .OOO|     w
#00000167  |      .  |     <NUL>
#00000168  | O OO. O |     Z
#00000169  |OOOOO.OOO|     RUBOUT
#00000170  |OO    .  O|
#00000171  |    OO.O O|     <GS>
#00000172  |    O .OO |     <SYN>
#00000173  |      .  |     <NUL>
#00000174  |OO    . O |
#00000175  |    O .O O|     <NAK>
#00000176  |OOOOO. OO|
#00000177  |OOO O. OO|
#00000178  |      .  |     <NUL>
#00000179  |      .  |     <NUL>
#00000180  |      .  |     <NUL>
#00000181  |      .  |     <NUL>
#00000182  |      .  |     <NUL>
           +---------+
Joiner 0000: data byte 00000000  absolute position 00000183
C:\Users\you>
```

In Inkscape:



After adding 10 inches of leader feed and 10 inches of trailer feed tape (run the above and add `--leader=100 --trailer=100`), opening the DXF file in Inkscape we should see the following:



Here we have a problem. The paper tape, as it stands, now measures 10" + 18.3" + 10" or a total of 38.3" (973mm) long. Although my stencil cutter could theoretically cut this as one strip, generally that is only done for thicker materials that are self-supporting and can be pushed and pulled by the cutter rollers directly, thus not requiring the square sticky backing mat. As it stands, it would run off the standard 12" x 12" (305mm x 305mm) cutting mat. For thin materials such as paper we will need to use the mat.

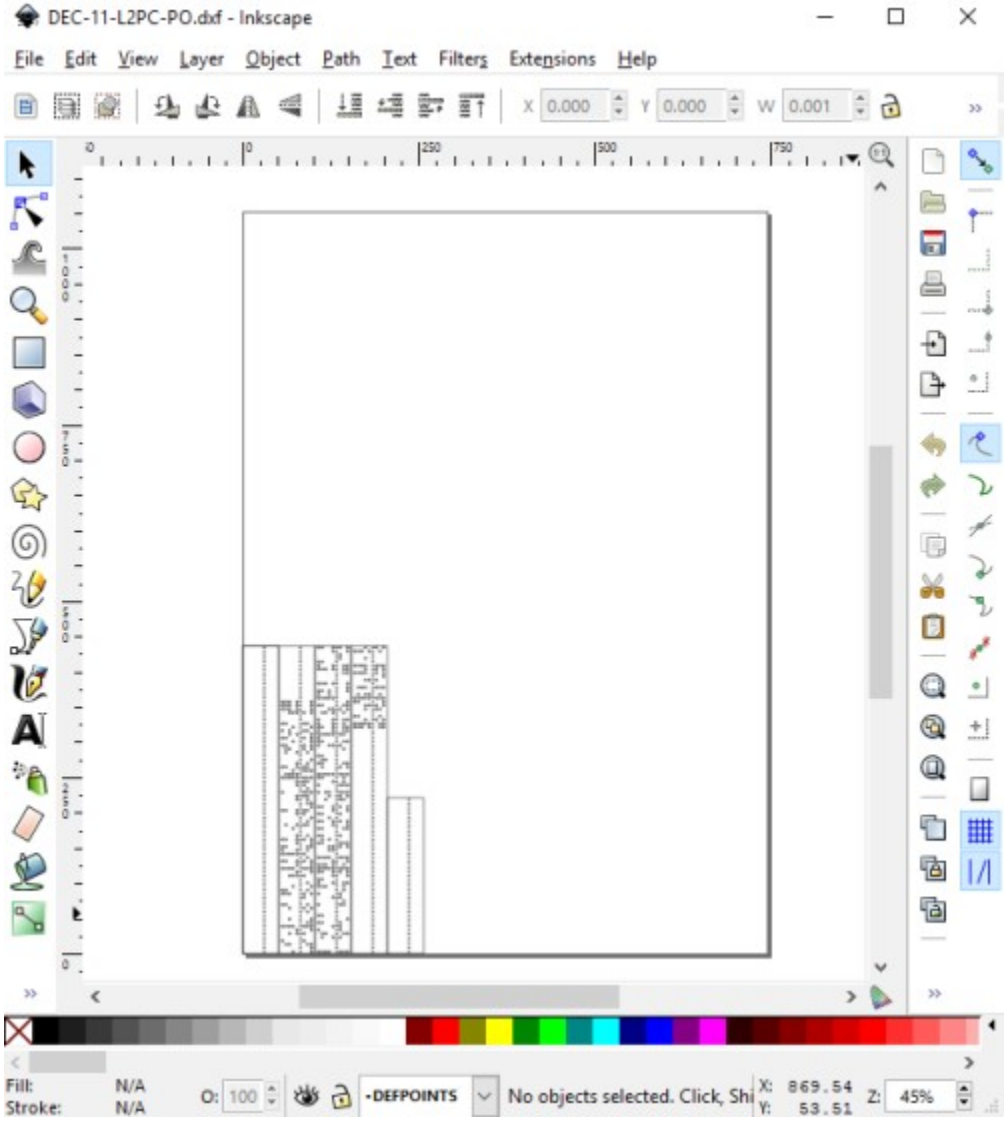
Making larger tapes

We can now apply a switch to break the single tape into segments of an arbitrary maximum length so it will fit on the cutting mat, across from left to right.
Lets say we want it split into 8.5" lengths, the fold size of DEC fanfold paper tape. This would give four large segments and the remaining tail segment about half that length. This segment length corresponds to 85 rows of holes.

Run the following

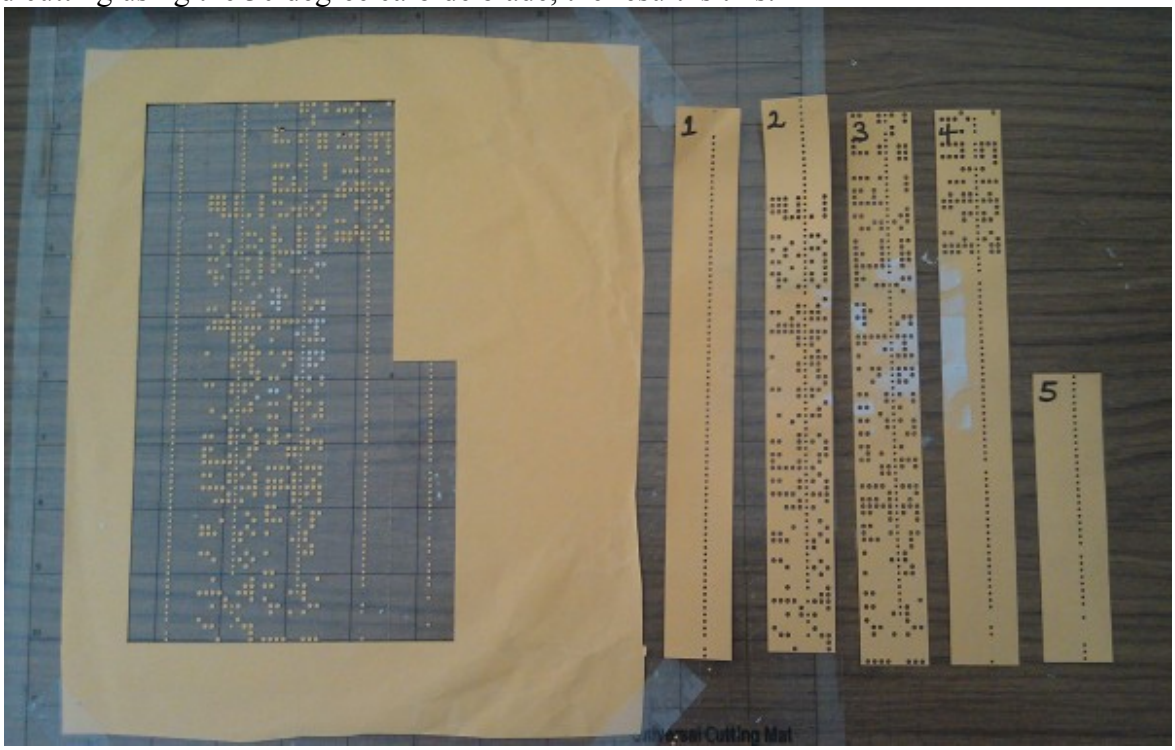
```
C:\Users\you>ptap2dxf DEC-11-L2PC-PO.ptap  --leader=100 --trailer=100 --number
--segment=85
```

And again load the generated DXF into Inkscape:



After setting up a scrap piece of a large yellow business envelope on the sticky mat:

...and cutting using the 30 degree carbide blade, the result is this:

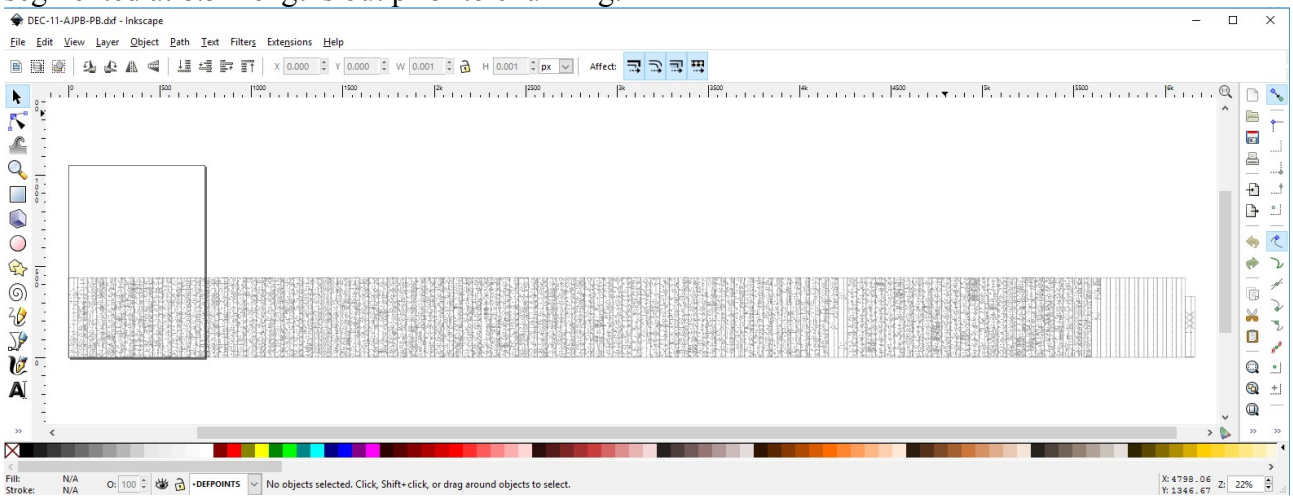


A suggestion is to use a pen to number the leading ends of the segments **before** lifting the segments from the cutting mat.

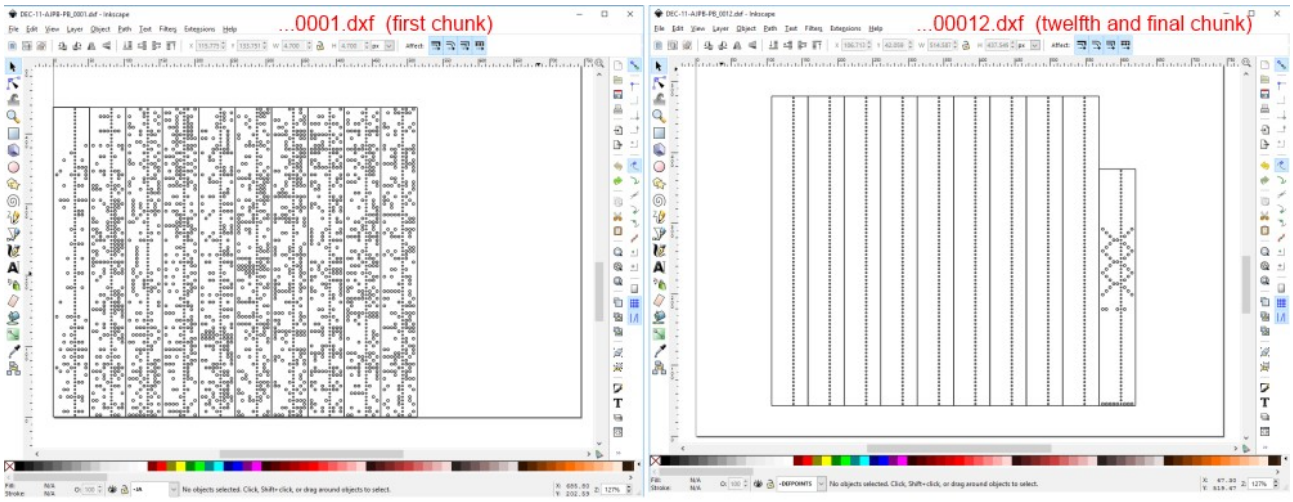
Even larger tapes

For larger tapes that exceed the size of the 12” x 12” cutting mat, the output can be 'chunked' into

any number of separate DXF files using the `-perdxf=n` switch. This limits the number of segments per mat.
For example, here is what PDP-11 BASIC (`DEC-11-AJPB-PB.ptap`) looks like in its entirety, segmented at 8.5” lengths but prior to chunking:



Chunked at `-perdxf=10` the first DXF and last DXF chunks look like the following, and each of the 12 chunks at 10” wide fits easily on a cutting mat. The program auto-suffixes the chunk number ..._nnnn to the filenames:



Joining the tape

Now the problem is, how do we join the segments?
One way would be to simply carefully adhesive-tape (stickytape, or 'Scotch' tape) the ends together and use an X-Acto hobby knife with #11 blade to re-cut the holes by hand, or use one of those rare miniature paper tape hand punches occasionally found on eBay at high prices. However for many segments this would be tedious. Thankfully PTAP2DXF can make adhesive joiners for you.

This is where the last few lines on the console output come into play:

```
----8<----8<----8<----8<----
          |       .     |
          |       .     |
          |       .     |
          |       .     |
          |       .     |
```

```
           |     .   |
           |     .   |
           +---------+
Joiner 0000: data byte 00000000  absolute position -00000015
Joiner 0001: data byte 00000085  absolute position 00000070
Joiner 0002: data byte 00000170  absolute position 00000155
Joiner 0003: data byte 00000255  absolute position 00000240
Joiner 0004: data byte 00000340  absolute position 00000283
```

These lines indicate at what point to produce a tape joiner. Although you could just re-cut a section of the above tape onto adhesive vinyl or contact plastic, the joiner switch adds tabs and alignment indents to the piece so you can hold it on your fingertips for more accurate positioning.
You can specify the lengths of the joiners for what you consider a good enough overlap onto each segment. Say, about 3/4 inch each side or 1.5" overall length, ie. a total of 15 row spacings. Let's bump that to 16 to make 8 holes overlapped onto each segment.

So, let's get to making them. Firstly make a note of the joiner details by copying those last lines to the clipboard and pasting into Notepad.
Then load a suitable sized piece of adhesive vinyl, contact plastic or similar onto the cutting mat. You could also use normal paper and glue the joiners on.
Run PTAP2DXF on the input as follows, specifying an output filename for each joiner so it does not clobber the original tape dxf. You do not need to give the leader or trailer lengths here because the range switch will automatically subtract or add blank rows to accommodate the range:

```
C:\Users\you>ptap2dxf DEC-11-L2PC-PO.ptap  --joiner --range=-15,+-8 --output=j0.dxf
+---------+
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|OOO O.  O| JOINER
|OOO O.  O| JOINER
|OOO O.  O| JOINER
|  OOO.O O| JOINER
|     .   | JOINER
|     .   | JOINER
|OO   .OO | JOINER
+---------+
C:\Users\you>ptap2dxf DEC-11-L2PC-PO.ptap  --joiner --range=70,+-8 --output=j1.dxf
+---------+
|OOOO .OOO| JOINER
|    O.  O| JOINER
```

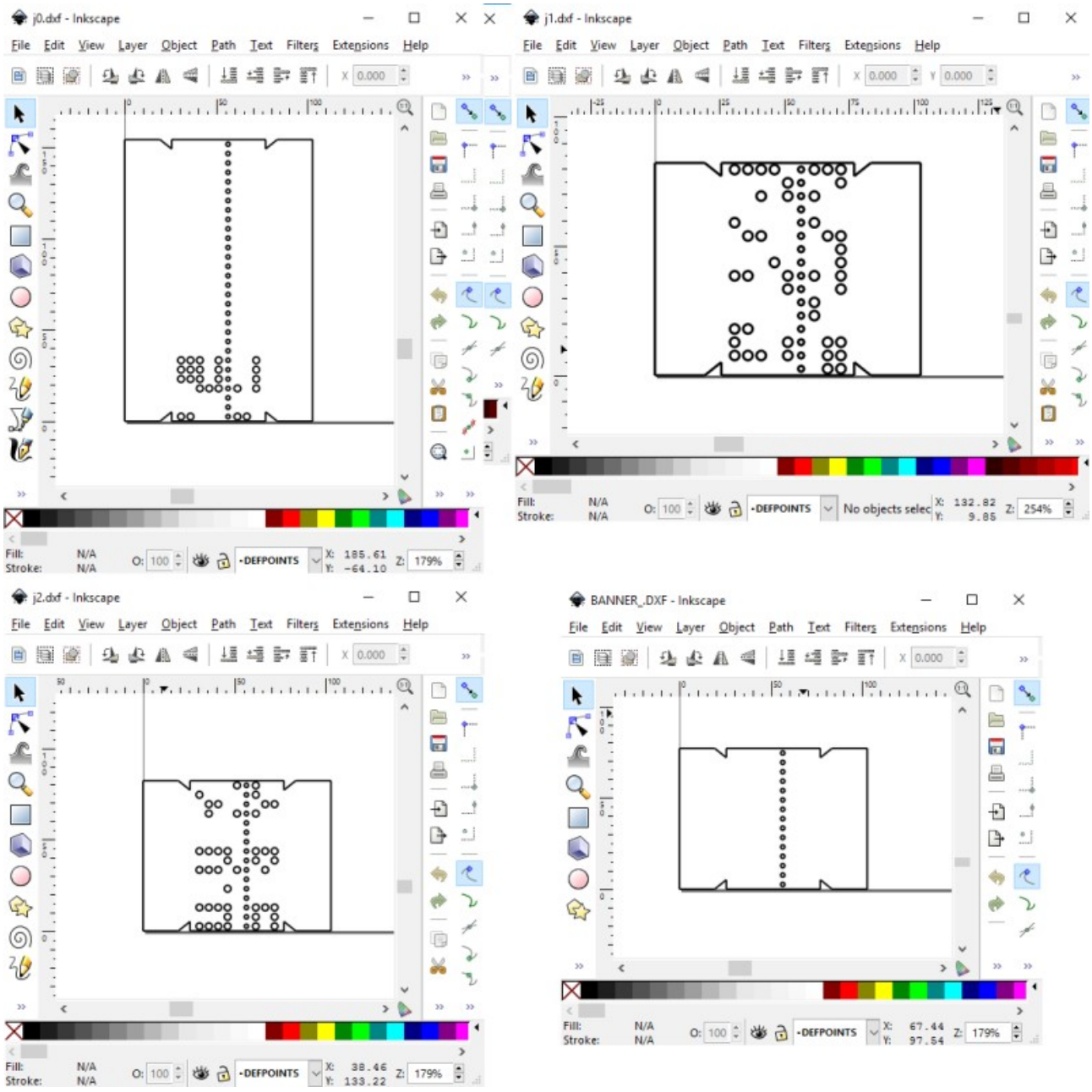
```
|  O O.O  | JOINER
|     .   | JOINER
|O    .O  | JOINER
| OO  . OO| JOINER
|     .  O| JOINER
|   O .  O| JOINER
|OO  O.O O| JOINER
|    O.  O| JOINER
|     .O  | JOINER
|     .O  | JOINER
|OO   .   | JOINER
|O   O. OO| JOINER
|OOO O. OO| JOINER
|     . OO| JOINER
+---------+
C:\Users\you>ptap2dxf DEC-11-L2PC-PO.ptap  --joiner --range=155,+-8 --output=j2.dxf
+---------+
|    O.O  | JOINER
|O    .O  | JOINER
| OO  . OO| JOINER
| O  O.O  | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|OOOO .OOO| JOINER
|   O .O O| JOINER
|OOO O. O | JOINER
|     .   | JOINER
|   O .   | JOINER
|     .   | JOINER
|OOOO .OOO| JOINER
|   O .O O| JOINER
|OOOO .O O| JOINER
+---------+
```

And finally, to produce a blank feed tape joiner consisting only of sprocket holes you can use the leader or trailer argument followed by an output filename:

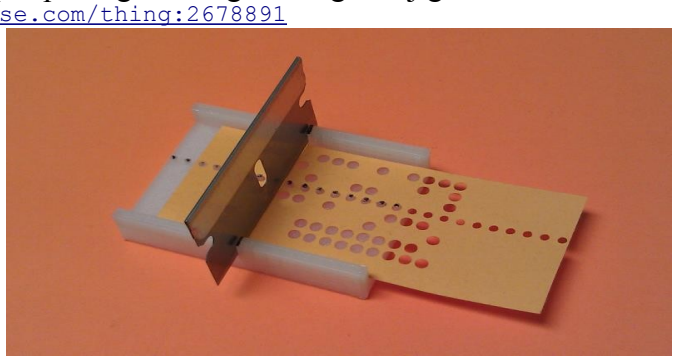
```
C:\Users\you>ptap2dxf --joiner --leader=16 --output=leaderlength16.dxf
+---------+
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
|     .   | JOINER
+---------+
```

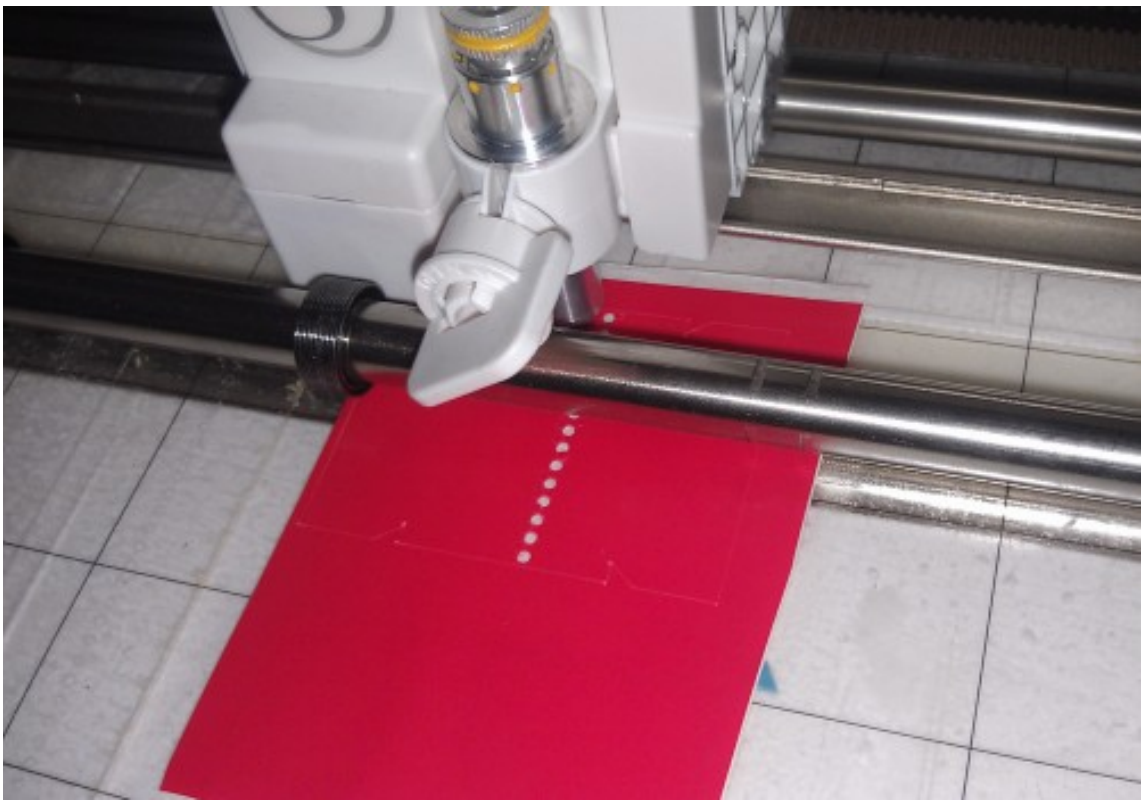
Note that there may be still some calculation bugs in the joiner and range switches, but generally it should produce your required sections with a bit of experimenting.

Loading up the joiners we can see the output features alignment cuts at the top and bottom as well as half-width tabs along each side. These are to allow the adhesive vinyl to be handled with fingertips whilst attempting to stick it accurately onto the segment ends:



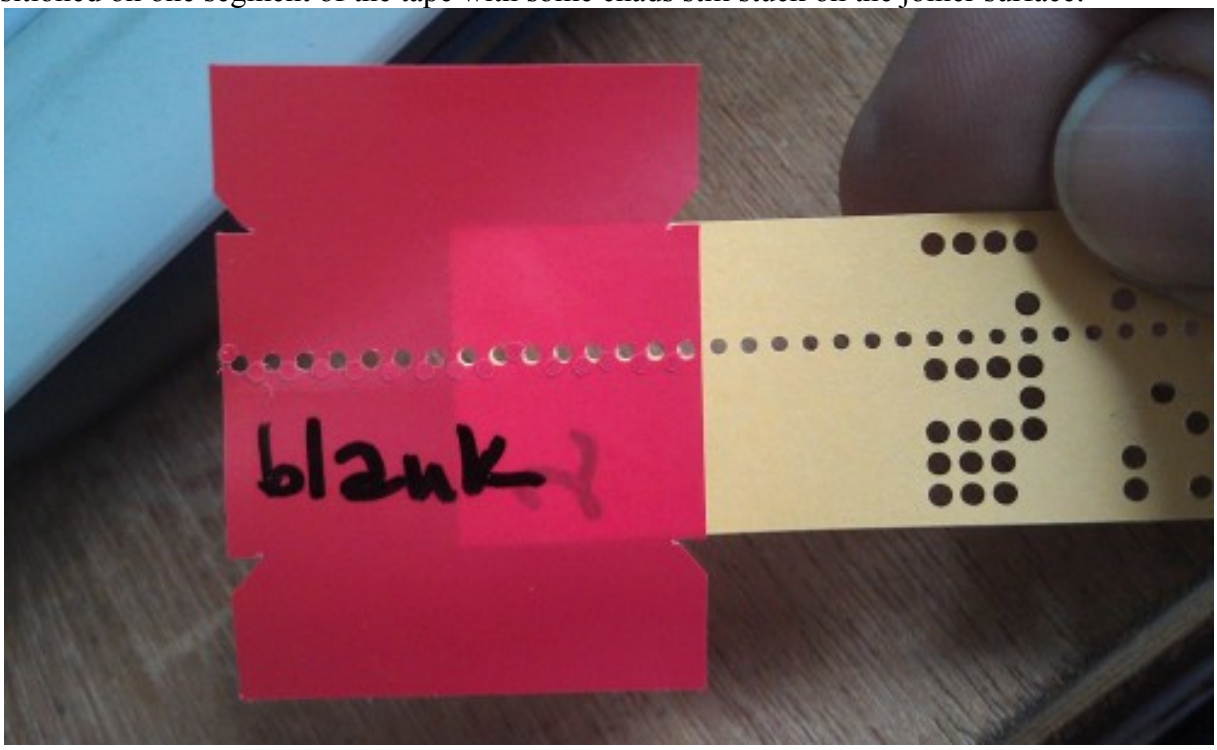
By the way, if you are using PTAP2DXF to make repairs to an existing paper tape you might find my 3D printed paper tape splicing and alignment guide jig useful. It can be found at https://www.thingiverse.com/thing:2678891





Here is some cutting in progress, using red vinyl self-adhesive contact plastic (book covering film):

Positioned on one segment of the tape with some chads still stuck on the joiner surface:



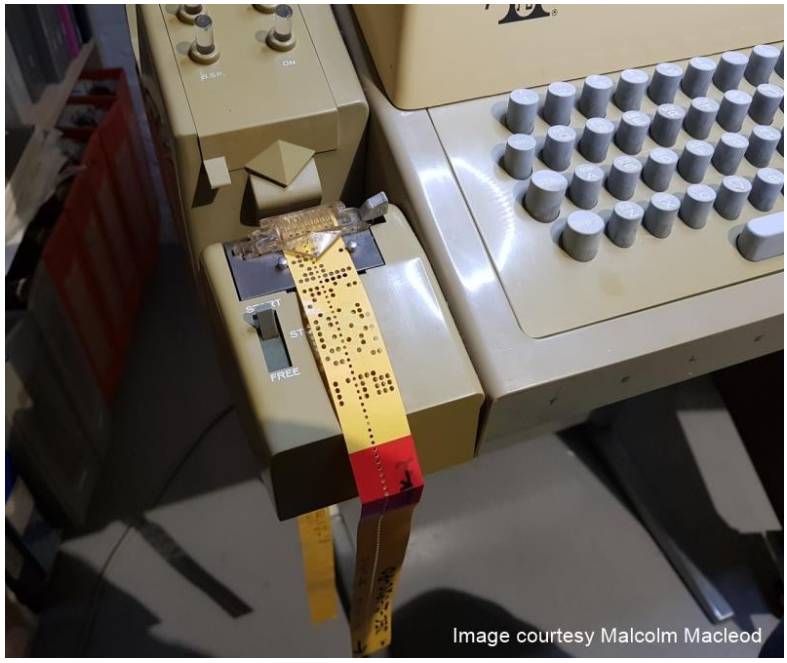
Note that many small chads are produced and will remain stuck to the sticky mat, especially when cutting adhesive vinyl or contact plastic. The use of a credit card or similar plastic 'rewards card' to scrape them off works brilliantly.

Stuck together, laid out and before trimming the joiner tabs:

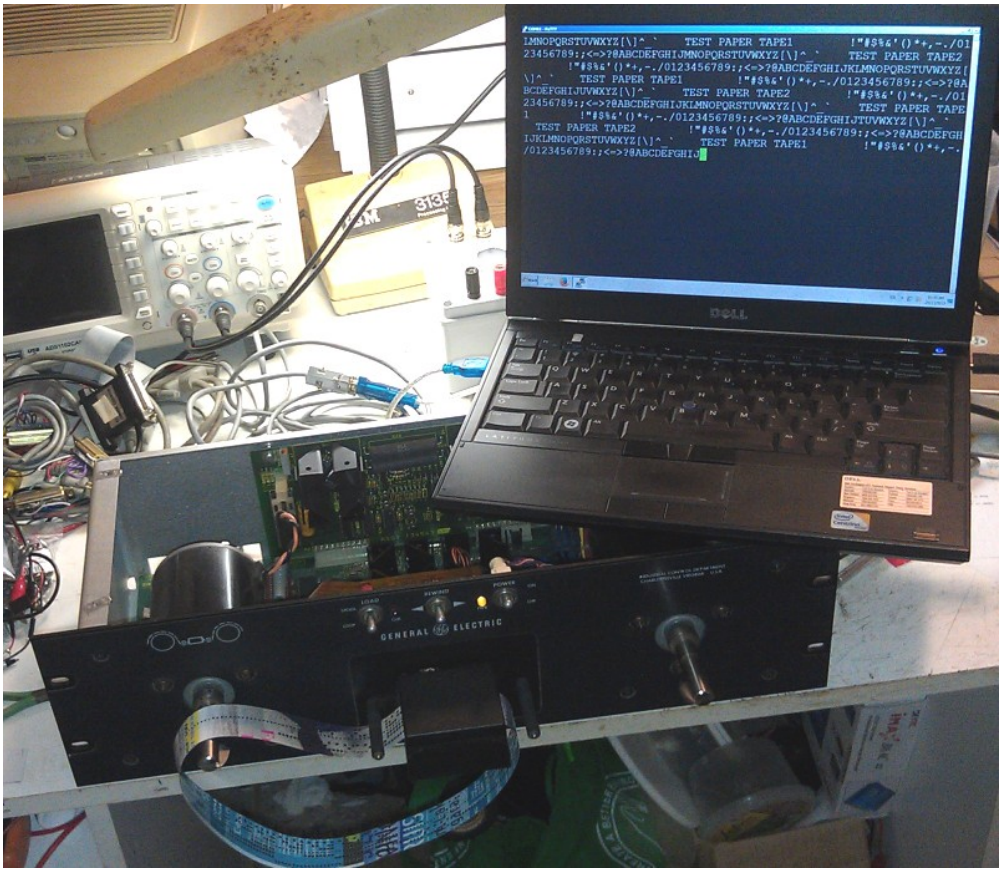
Once the joiner is fixed in place, use sharp scissors to trim the tabs along the edges of the tape, using the indents as the scissor guides. The finished paper tape with all joiners added and trimmed:

Testing

The tape set up for loading on an ASR33 Teletype (courtesy Malcolm Macleod, Avitech):



The following photograph shows running a short loop cut from a glossy magazine cover consisting of printable characters then “TEST PAPER TAPE 1” then another set of printable characters and “TEST PAPER TAPE 2” on an EECO MTS-82 paper tape reader circa 1982, connected through an RS232-to-USB cable to a laptop putty session on COM11:, 300 baud, 8N1 and xon/xoff. The reader was configured to Level 1 paper tape protocol so that run/stop tape control characters did not need to be sent. Toggling the reader's LOAD LOOP switch a few times was sufficient to actuate output:



Producing banners

PTAP2DXF can make paper tape text banners, or you can use this feature to add a text label to the start of a binary file for identification purposes. The text is restricted to an 8x8 pixel upper case only font with some basic punctuation symbols, the available character set alphabet being:

```
!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`
```

The banner text can come directly from the command line or from an ASCII file.

For example, to produce a piece of tape with the letters 'ABC' use

```
C:\Users\you>ptap2dxf --bannertext="abc"
+---------+
|      .  |
| OOOO.OO |
|    O.  O|
|    O.  O|
|    O.  O|
| OOOO.OO |
```

```
|     .   |
|     .   |
| OOOO.OOO|
| O  O.  O|
| O  O.  O|
| O  O.  O|
|  OO .OO |
|     .   |
|     .   |
|  OOO.OO |
| O   .  O|
| O   .  O|
| O   .  O|
|  O  . O |
+---------+
Joiner 0000: data byte 00000000  absolute position -00000001
```

By default a banner DXF output file will be called 'BANNER_(your text).DXF' for instance the previous example becoming 'BANNER_ABC.DXF'.

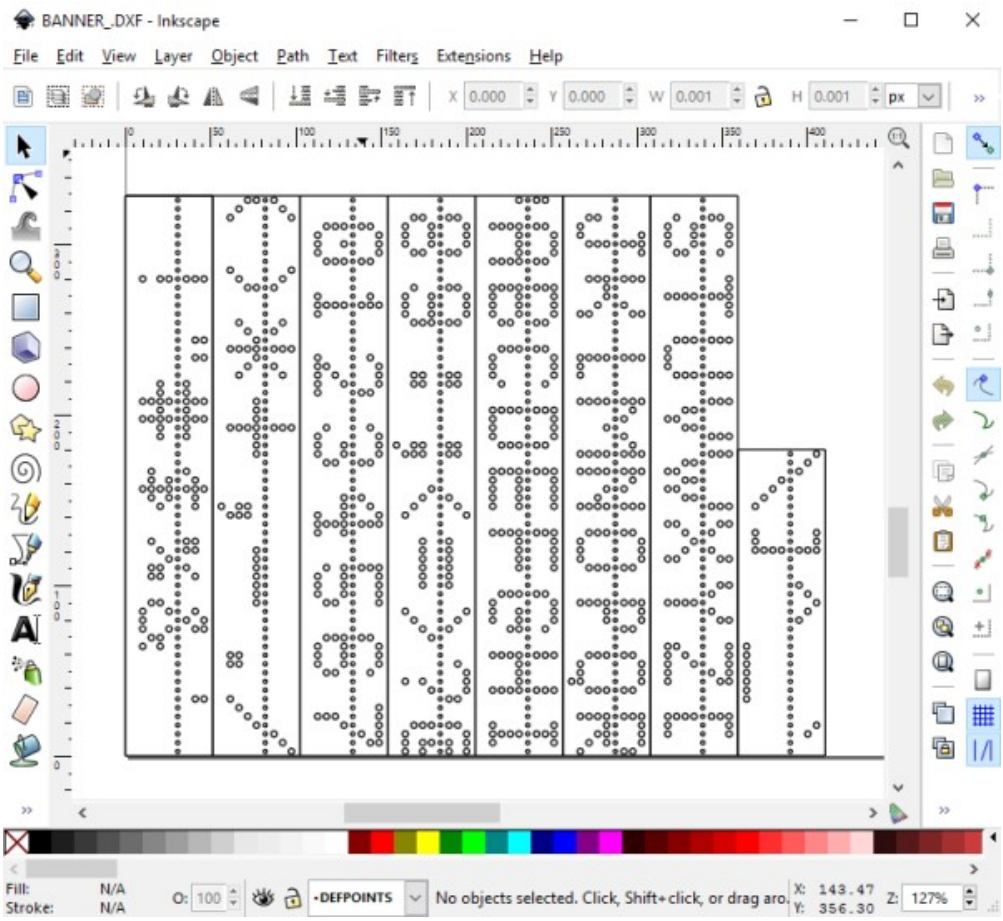
The following adds the banner label 'ABS LDR' to the front of the binary file used in earlier examples:

```
C:\Users\you>ptap2dxf --bannertext="ABS LDR        " DEC-11-L2PC-PO.ptap
```
(output omitted)

Notice the extra 8 spaces in the banner text. This is to produce 64 rows of sprocket feed between the banner and the punched binary code, otherwise the data would start immediately after the 'R'.
Also be aware that the banner character set is for 8-level tape only, and will be truncated if punched at levels less than 8.

For strings including quotes and punctuation, it can be easier to put the text into a textfile and use the `--bannerfile` switch instead of on the command line. For example if the complete alphabet letter string shown above is in the file alphabet.txt:

```
C:\Users\you>ptap2dxf --bannerfile="alphabet.txt" --segment=64
```
Looks like



Other options

The **Mirror** switch (`--mirror` or `/MIRROR`) generates the output with the least significant bit and most significant bit positions reversed, and so for the intervening bits. Its primary use is to produce adhesive joiners that affix to the back of the paper tape segments, if so desired.
Example:

```
C:\Users\you>ptap2dxf --bannertext="abc" --mirror
+---------+
|    .    | MIRROR
| OO.OOOO | MIRROR
|O  .O    | MIRROR
|O  .O    | MIRROR
|O  .O    | MIRROR
| OO.OOOO | MIRROR
|    .    | MIRROR
|    .    | MIRROR
|OOO.OOOO | MIRROR
|O  .O  O | MIRROR
|O  .O  O | MIRROR
|O  .O  O | MIRROR
| OO. OO  | MIRROR
|    .    | MIRROR
|    .    | MIRROR
| OO.OOO  | MIRROR
|O  .   O | MIRROR
|O  .   O | MIRROR
|O  .   O | MIRROR
| O .  O  | MIRROR
+---------+
Joiner 0000: data byte 00000000  absolute position -00000001
```

The **Flip** switch (`--flip` or `/FLIP`) inverts the bits in the output (performs a logical NOT):
Example:

```
C:\Users\you>ptap2dxf --bannertext="abc" --flip
+---------+
|OOOOO.OOO| INVERTED
|O    .  O| INVERTED
|OOOO .OO | INVERTED
|OOOO .OO | INVERTED
|OOOO .OO | INVERTED
|O    .  O| INVERTED
|OOOOO.OOO| INVERTED
|OOOOO.OOO| INVERTED
|O    .   | INVERTED
|O OO .OO | INVERTED
|O OO .OO | INVERTED
|O OO .OO | INVERTED
|OO  O.  O| INVERTED
|OOOOO.OOO| INVERTED
|OOOOO.OOO| INVERTED
|OO   .  O| INVERTED
|O OOO.OO | INVERTED
|O OOO.OO | INVERTED
|O OOO.OO | INVERTED
|OO OO.O O| INVERTED
+---------+
Joiner 0000: data byte 00000000  absolute position -00000001
```

The **Sprocket** switch (`--sprocket=n` or `/SPROCKET=n`) allows a custom (non-standard) positioning of the sprocket-feed hole. By default it is set to 3, or between the 3rd and 4th data bit holes, starting from the right-hand side. Setting sprocket=0 puts the feed hole along the right-hand edge, and sprocket=8 puts them at the left-hand edge. The value of n can be any integer between 0 and the punch level plus one.

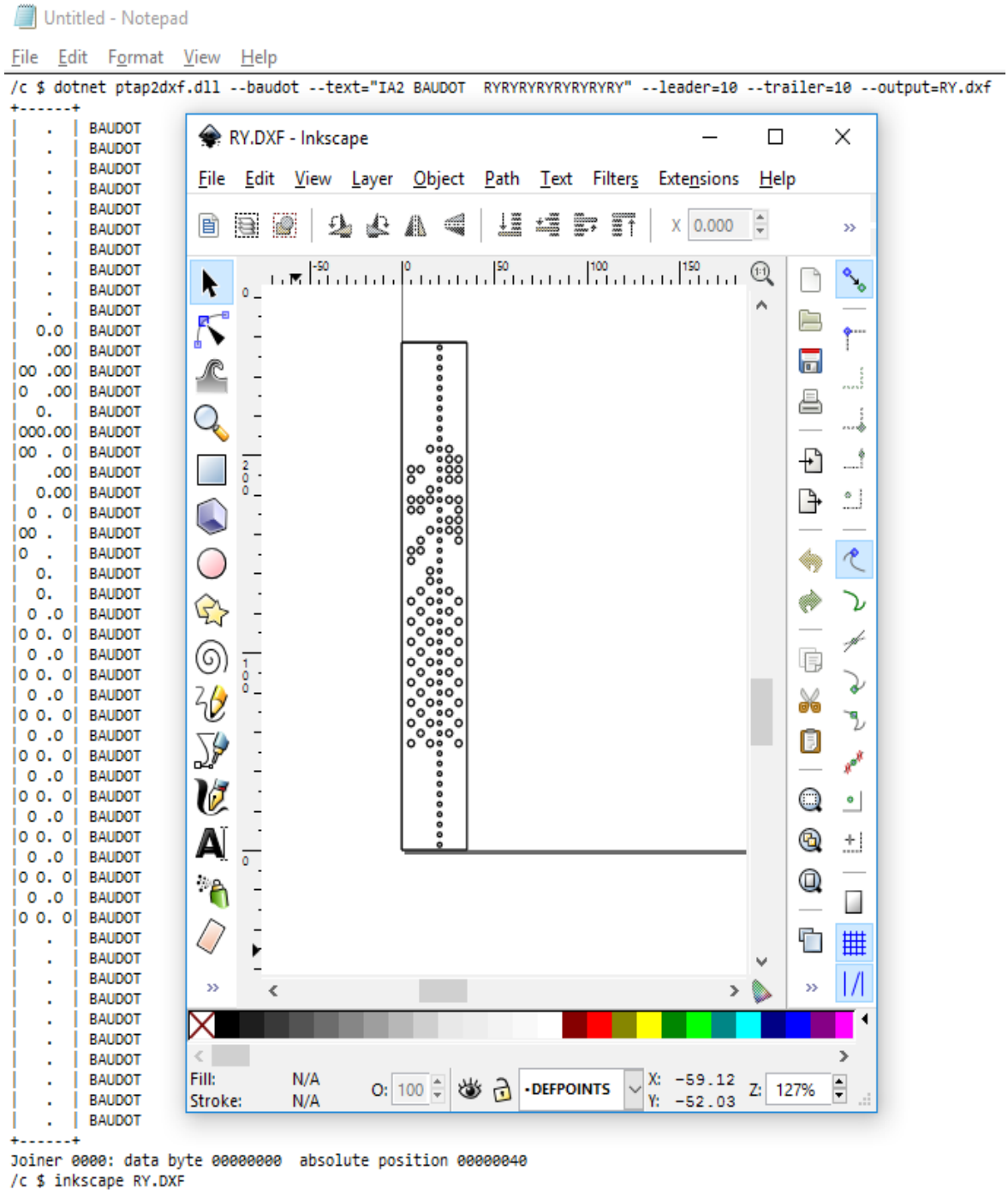
The **Level** switch (`--level=n` or `/LEVEL=n`) enables the production of other level paper tapes. By default the level is 8, or 8-level ASCII output. 5-level tape that is 11/16" wide can be made with --level=5. The encoding is still ASCII with the top 3 bits stripped off. For levels 6,7 and 8 the output is still one inch tape. For levels less than 5, the width is set to (level * 0.1 inch plus an additional 0.1 inch for the sprocket hole and 0.1 inch each side). For --level=5 the sprocket hole is automatically set to between the 2nd and 3rd data bit holes from the right, but can be overridden with the sprocket= switch. Note that this does not force Baudot encoding, use the following switch to do that.

The **Baudot** switch (`--baudot` or `/BAUDOT`) encodes the ASCII bit array into ITA2 Baudot 5-level code. The ASCII input is filtered to only numeric, alpha and a limited punctuation set. The requisite Letter-change (LETR) and Figure-change (FIGS) codes are automatically inserted. Using this flag forces the level to be 5 (ie. 11/16" wide) , so it is not necessary to additionally set `--level=5` as this is implied.

Example:
The RYRY pattern string exercises all the mechanical punch pins in a real punch.

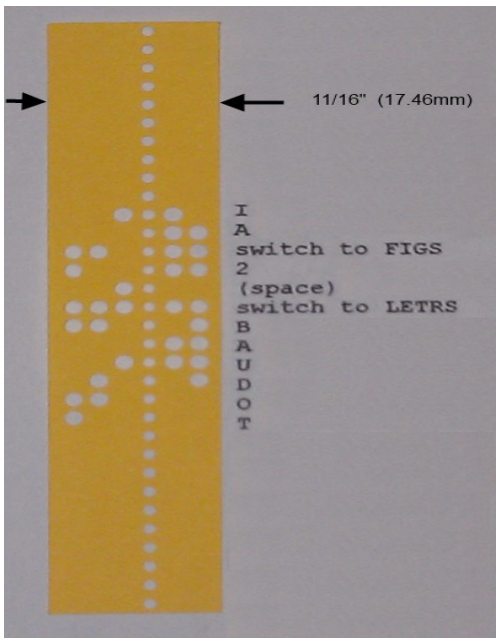
```
/c $ dotnet ptap2dxf.dll –baudot –text="IA2 BAUDOT  RYRYRYRYRYRYRYRY" --leader=10 –
trailer=10 -output=RY.dxf
```

looks like the following (with Inkscape window superimposed on console window):



The different prompt and the use of “dotnet ptap2dxf.dll ...” to run the program comes from running it as a .NET Core application, the same invocation is used on Linux and Mac.

Here is an earlier cut of “IA2 BAUDOT” (without the RYRYRYRYRYRYRYRY). Compare this with the given output shown at https://www.dcode.fr/baudot-code :



The **Mark**=switch (--mark=”c” or /MARK=”c”) sets the console output character for a 1 data bit (“mark”).The default is a capital Oh (“O”) to signify a punched hole. Changing this has no effect on the DXF and will always be a cut hole.

The **Space**=switch (--space=”c” or /SPACE=”c”) sets the console output character for a 0 data bit (“space”).The default is a space (“ ”) to signify a blank (no punched hole). Changing this has no effect on the DXF and will always be a blank space.

For simply checking binary output on the console you might like to try `-mark=”1” --space=”0”` as shown here:

```
/c $ dotnet ptap2dxf.dll --text="ABCD" --mark="1" --space="0"
+---------+
|01000.001|
|01000.010|
|01000.011|
|01000.100|
+---------+
Joiner 0000: data byte 00000000  absolute position 00000004
/c $
```

The **Parity**=switch (--parity=NONE|EVEN|ODD) uses the high bit (leftmost hole) for parity, if desired. The default is NONE.

```
/c $ dotnet ptap2dxf.dll --text="ABCD" --parity=even
+---------+
| O   .  O|
| O   . O |
|OO   . OO|
| O   .O  |
+---------+
Joiner 0000: data byte 00000000  absolute position 00000004
/c $
```

The **Wait** switch (`--wait` or `/WAIT`) simply pauses the program after operation and waits for the Enter key to be pressed before returning to the command shell prompt. The program sets the ERRORLEVEL accordingly, so it can be tested in a batch file.

Possible future developments

- An overlap switch which would repeat the last n bytes (rows) of the previous segment at the top of the next segment allowing the option of an overlapped glue joint instead of having to apply a separate joiner piece.
- Switches to cut paper tape for uncommon formats like Colossus (Baudot with additional start/end message holes), pre-war Creed 9W Morse reperforator, USN Wheatstone tape, ILLIAC, chadless 5-level paper tape etc. etc.
- Production of replica but fully-functional IBM 80-column punched cards, cut from light card stock.