Flashing Electronic Shelf Labels

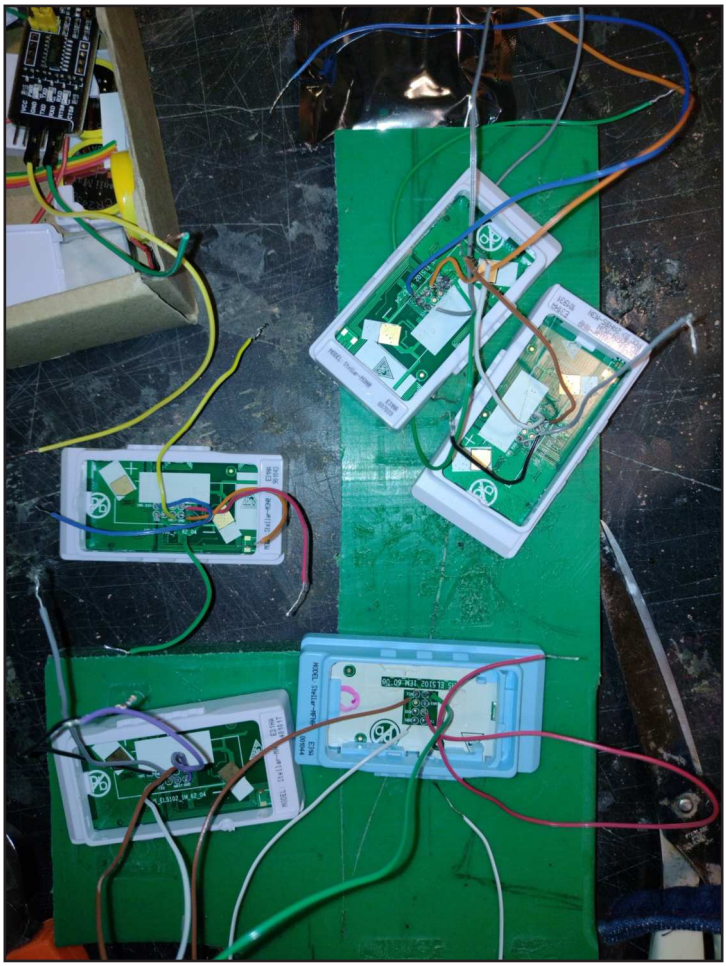
Two major Dutch supermarkets, both using

Hanshow

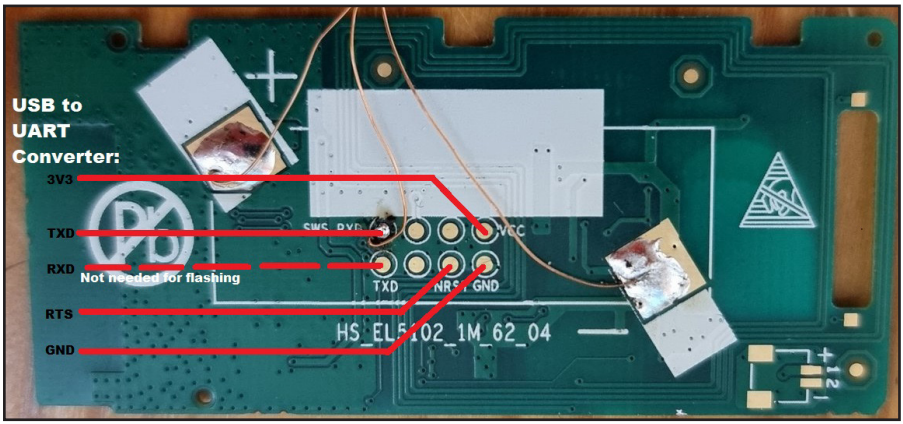
price tags.

Jumbo, using the Nebular model.
No replacable batteries, I need a dremel to get in.

Albert Heijn uses the Stellar model.
Batteries can be taken out, exposing the pads where we can solder to.



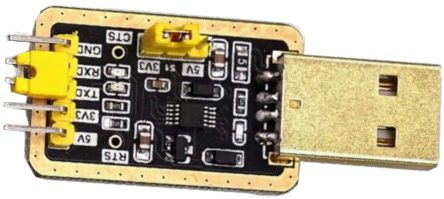
Soldering tiny cables.
It goes all smooth when prepared,
everything coated in tin,
some flux
and a soldering stand
helping us along the way.



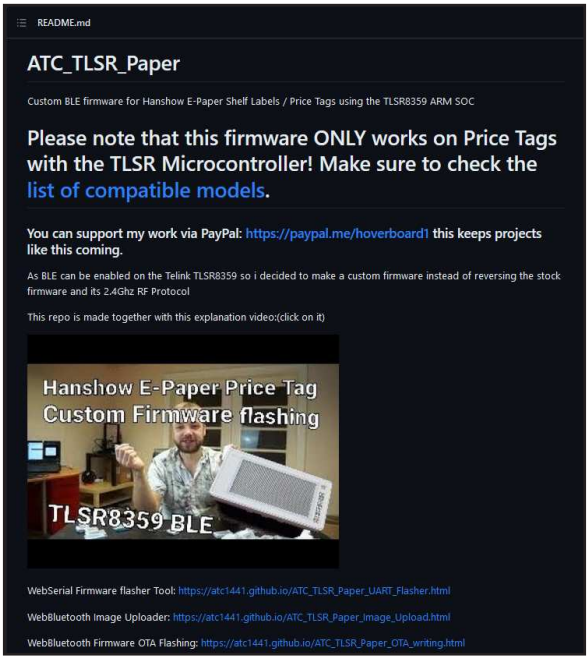
I need to talk with the microchip inside,
rewrite his beliefs.

It's submissive position towards the supermarket
will come to an end.

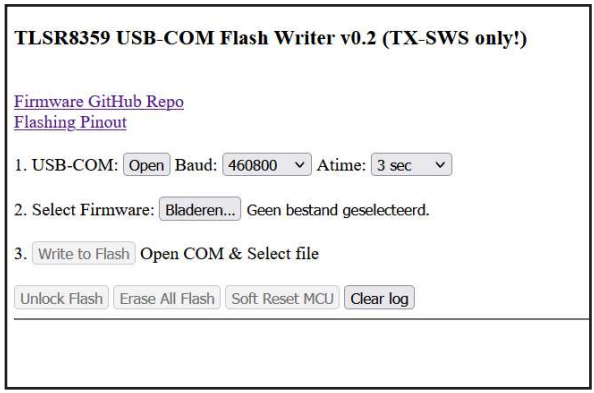
A USB to UART converter will help us with that.



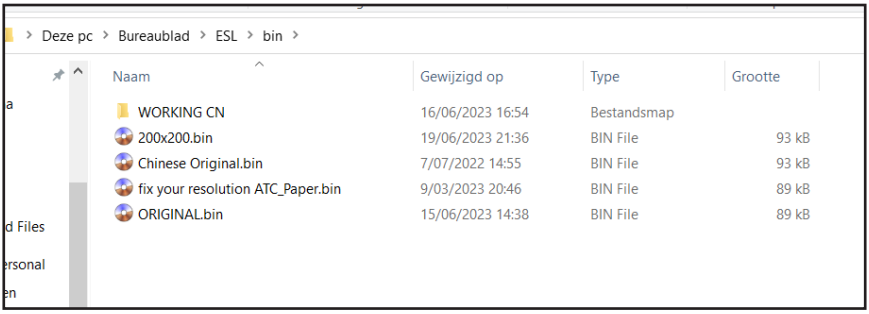
Github brought us ATC_TLSR_Paper,
a firmware,
to flash.



Open the flash tool in Microsoft Edge
(my apologies, Firefox is adjusted to strong privacy settings)



Connect to the serial USB converter,
first, unlock the flash, write the .bin after that.



The .bin needs to be compiled on your computer.
Run makeit.exe in E:\stellar-L3N-etag-main\Firmware.

With these various models and screen sizes it became a mess.
It was smooth to upload a firmware, uploading images was a nightmare.

About the display:

The e-ink panel used in this ESL is 250 by 122 pixels, black and white (no gray levels...yet).

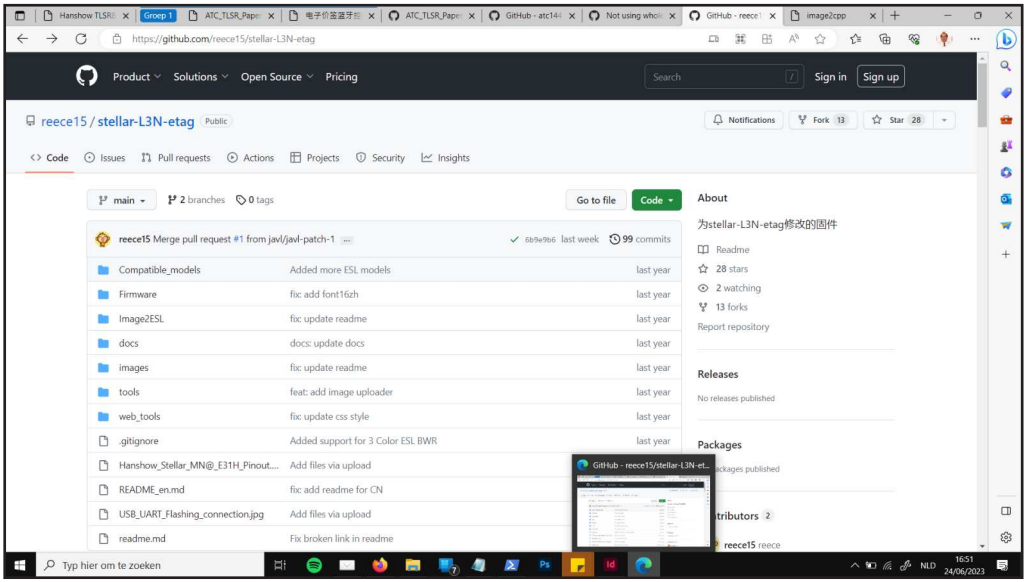
Larry Bank added his OneBitDisplay (https://github.com/bitbank2/OneBitDisplay) and TIFF_G4 (https://github.com/bitbank2/TIFF_G4) libraries to make it easy to generate text and graphics. For anyone wanting to write directly to the display buffer, the memory is laid out like a typical 1-bpp bitmap except that it is rotated 90 degrees clockwise. In other words, the display is really 122 wide by 250 tall, but laying on its side. Each byte contains 8 pixels with the most significant bit on the left. Black is 0 and white is 1. Each row of 122 pixels uses 16 bytes. Here is an example function to set a pixel given the x,y of the orientation (portrait) that the display is used:

```
void SetPixel(int x, int y, uint8_t *pDisplayBuffer)
{
   uint8_t *d = &pDisplayBuffer[(y >> 3) + (249-x)*16];
   *d &= ~(0x80 >> (y & 7)); // set pixel to black
}
```

The following hardware is currently supported, most displays are detected automatically if that fails you can select the correct one in the OTA flashing tool. The graphical layout is not edited for each screen size and will not fit nicely on all especially the 1.54" Version.

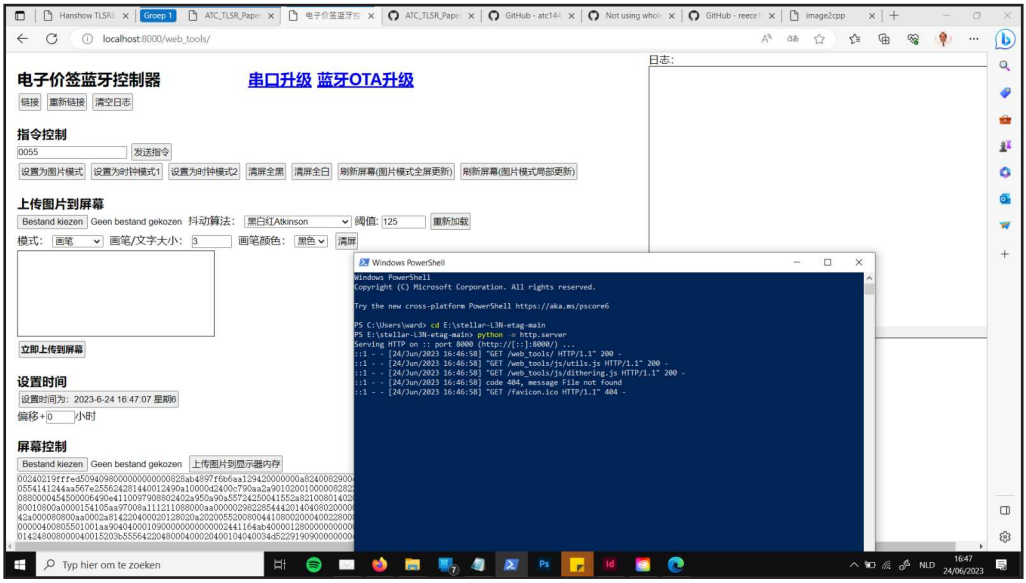
No convenient drag & drop.

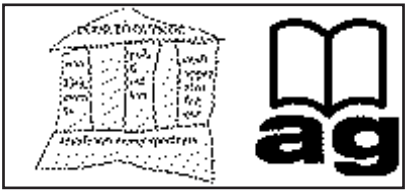
Blessed with reece15/stellar-L3N-etag,
easy uploading was made possible.

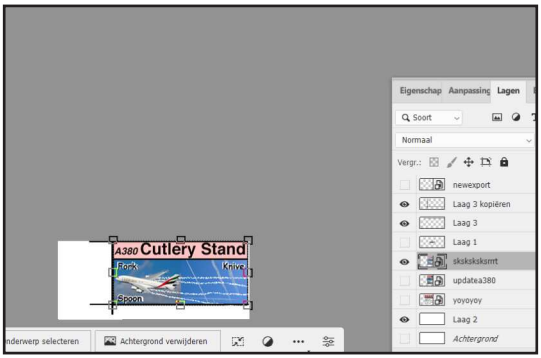


Let's say it works on 70% of our available Stellar models.
It comes with a slightly adjusted firmware.

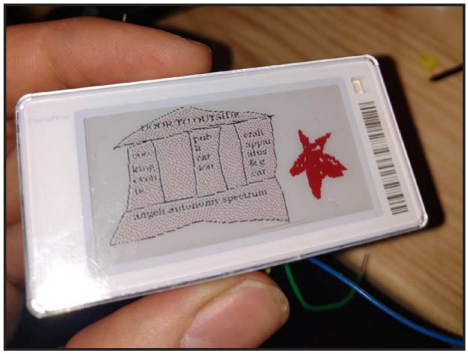
Before I forget, you flash with USB, all communication
takes place over Bluetooth after that.



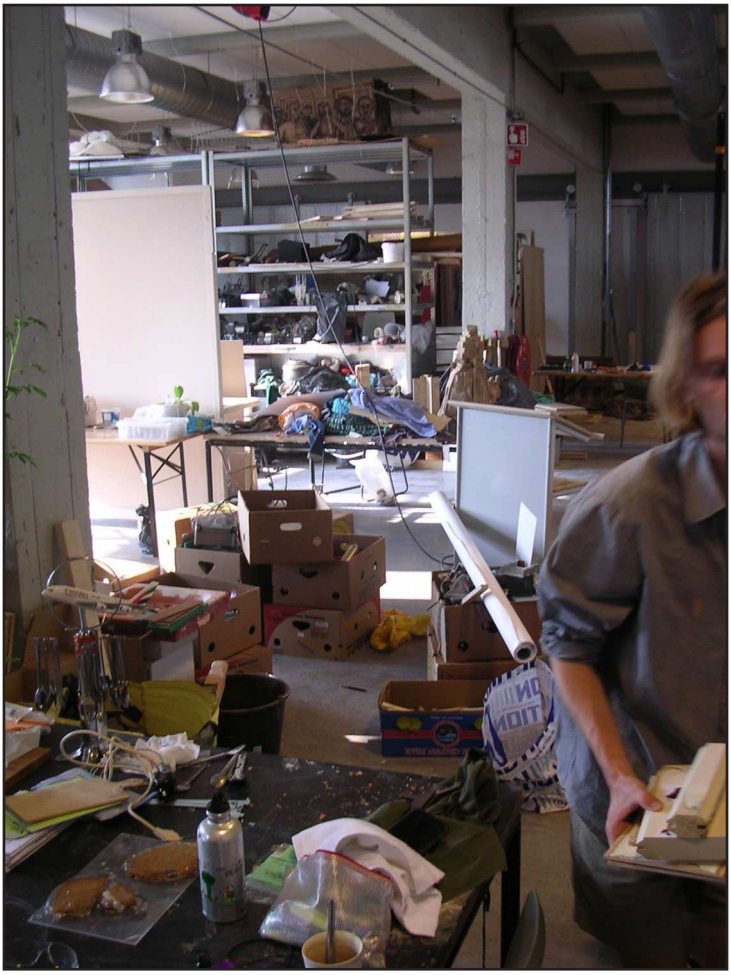




Arrangement in a 122 x 300 pixels file,
export, test, untill it displaying what we desire.



Never steal from people,
steal from places.



Eindhoven, June 2023
Angeli