YEARBOOK

sonicthehedgehog.app

2021

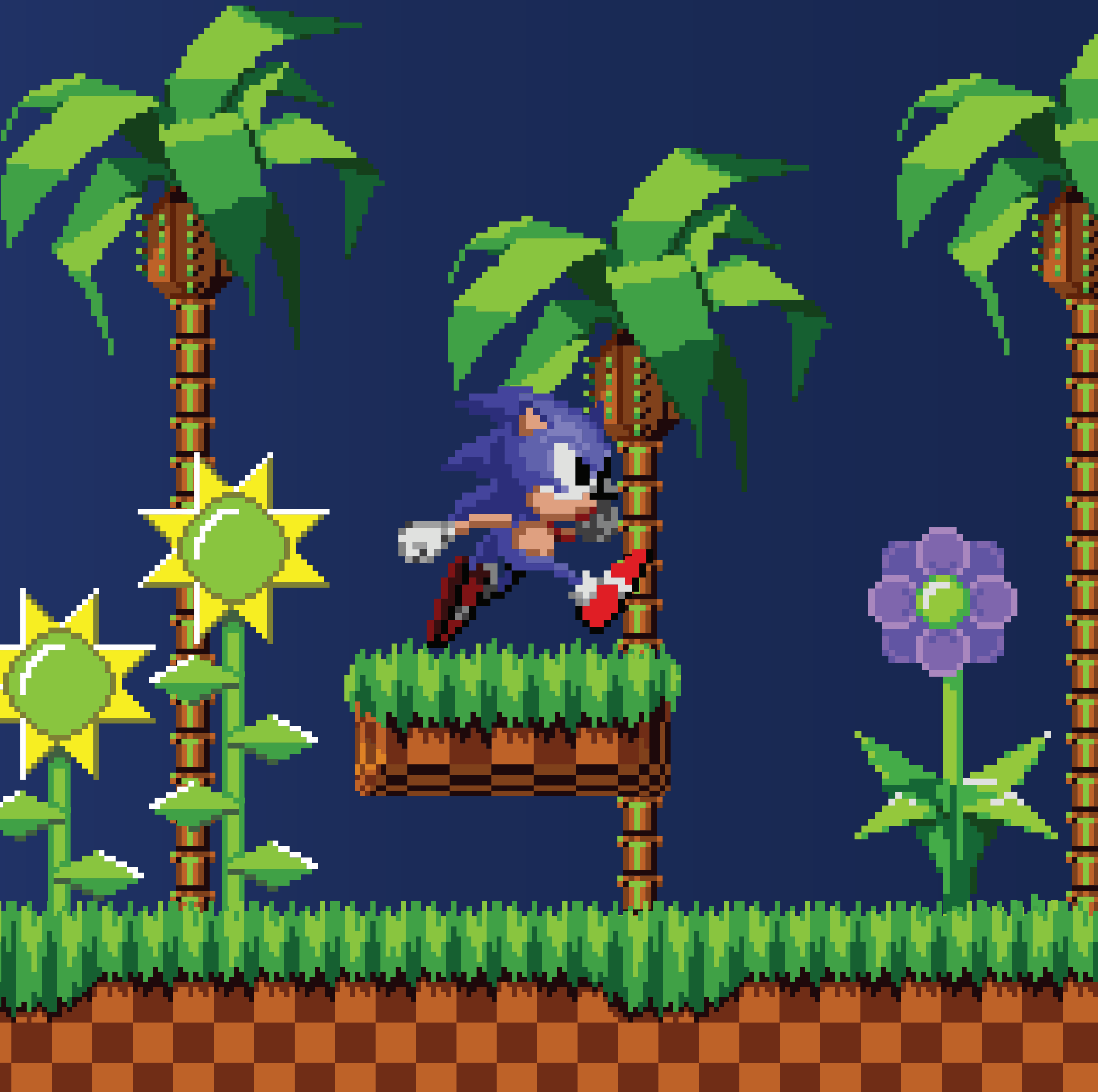
THE SONIC ZONE ソニックゾーン™

SEGA®

30th ANNIVERSARY

SONIC THE HEDGEHOG™

Sonic is steadfast, sincere, and all about speed! He is intuitive, spontaneous, and quick-witted. On top of it all, Sonic cuts a dashing physique, and his charisma is electric!



THE ソニックゾーン SONIC™ ZONE

SONIKKU'S YEARBOOK・ソニックの年鑑 2021

30th ANNIVERSARY
23 JUNE 2021
SEE PAGE 92

CONTENTS | 目次

EDITORIAL

NEWS, MUSIC AND SCREEN

5

TAILS' WORKSHOP

7

AMY'S ART CORNER

SHADOW'S SECRETS

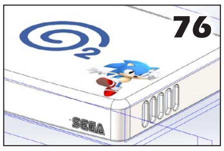
8

64

CREAM & CHEESE TREATS

INSPIRATIONAL ART

83


76

TRUE STORY OF A CONSOLE

SONIC MERCHANDISE SHOWCASE

11

36

WHIMSICAL PONDERINGS

BACKING MY BUDDY!

MISCELLANY

Cover Image

Digital artwork from Sonic X
"Sonic and Amy"
by Sonikku
February 2020

THIS IS AN UNOFFICIAL FAN WORK NOT FOR SALE OR PROFIT. FAIR USE APPLIES

SEGA® A SEGA SAMMY COMPANY · SONIC CHANNEL sonic.sega.jp · IDW® · SONIC TEAM™ · ソニックX™

EVERY ATTEMPT POSSIBLE HAS BEEN MADE TO IDENTIFY, CREDIT AND ACKNOWLEDGE ALL IMAGES, ARTWORK AND COPYRIGHTS IN THIS YEARBOOK. HOWEVER SOMETIMES THINGS FALL BETWEEN THE CRACKS. PLEASE DO LET ME KNOW IF YOU FIND YOUR ART OR TRADEMARK IS NOT CREDITED AND I WILL RECTIFY THE SITUATION.

EDITORIAL

Welcome to the very first yearbook Sonikku has ever published. This is an idea and concept I've mulled over for approximately a decade. With both 2020 and 2021 being worse than a boss fight against Zavok, I had to find things to do to remain sane, and this book was the result. This creative work is a reflection of what I do, things I love, and what these characters mean to me, executed in a magazine format and published digitally.

Each Sonic character represents a particular strength to me and this is how I've styled this publication- for example you will quickly notice Tails heads up all the "tech" stuff, whereas Amy heads up all the arty stuff and so on and so on...

For a little bit of info about me- I have Asperger's syndrome and I have been a Sonic fan for 28 years. I am an IT professional working in the financial sector. Outside of work I enjoy gaming (obviously Sonic) and I am passionate about art, particularly anthropomorphic art or "cartoons".

I have at various times waded into the online Sonic fandom. Older Sonic fans may remember me from the early 2000s when I was active on forums and message boards and had my own Sonic websites. I no longer participate however as I am much older than the current crop of Sonic fans- the generational gap is highly unfortunate.

The original plan was to self-publish this work, but, the closer I got to completion, the more it became apparent that I would need to leverage social media to have any chance of succeeding with this project and realising the eventual aim of licensing and coverage for this territory. I will be clear on this- I detest social media but social media is where Sonic is at these days, so if he is there, I will have to be there too, its mutually exclusive, and in view of the fact that I fully intend to become a licensee at some point in the future, or at the very least an official reseller of merchandise, I need to get my A-game on and I need to ensure the widest possible reach. Hence, social media. Its probably not going to be what I want but I have to play by the society's current rules- as absurd as they may seem to me.

I will try and keep things bilingual here. I am proficient in Japanese, and Sonic is Japanese, so I like to keep things somewhat real and relevant to his origin country.

So, as would be fitting to say in Japanese... 宜しくお願いします (yoroshiku onegaishimasu).
Let's all enjoy Sonic / すべてのソニックを楽しむことができます (subete no sonikku o tanoshimu koto ga dekimasu)

Sonikku(2k)
Cape Town
South Africa
soniczoneyearbook@gmail.com
PGP key fingerprint: 3AA9 12F1 D8E2 BB5F 6896 DA65 FCE0 D3A7 245F D4E5
[All social media accounts were deleted in 2022 because of reasons you can guess]

CAUTION!

Tendency to be brutally honest

Strong opinions, passionate viewpoints and matter-of-fact comments may offend some people.

NEWS, MUSIC AND SCREEN

Release Date:	14 February 2020
Production:	SEGA SAMMY Group Original Film Marza Animation Planet Blur Studio
Budget:	USD 81-95m
Box Office:	USD 306.8m
Cast:	James Marsden Ben Schwartz Tika Sumpter Jim Carrey

SONIC THE HEDGEHOG

SONIC'S FIRST OUTING ON THE *REALLY* BIG SCREEN. WAS IT PASSABLE OR JUST OKAY?

I had known about the script for this film for quite a number of years. It has certainly been passed around, even Sony was going to have a go at it at one point in time.

Eventually in the latter part of the 2010s, Paramount decided to pick it up and run with it, and somehow, in the midst of the controversy in 2019 with the character design that saw Jeff Fowler post a commitment to "fix" the character design on social media, the film seems to have seen the light of day. Like even the worst Sonic game, it has somehow actually made money and they have announced a sequel. It was obvious that there would be one, by the inclusion of Tails in the end credits.

Simon Abrams over at rogerebert.com had a rather scathing review of the film, and I am hard pressed to disagree. The most glaring aspect for me was the facial features, where Sonic is somehow more like Tails with separate eyes instead of the one "single eye" we all know.

He is dressed down to be more like a certain yellow Pokemon because yes, that recent live action film was the yardstick whether we like it or not.

Ultimately my disapproval extends further than just the character's face, other features also irk me- the fact that his arms are the wrong colour #(S&!# Come on guys! In addition to that I already mentioned, its the out-of-character behaviour and words coming from this version of Sonic. The producers and writers seem to be ignorant or totally out of touch with how Sonic and in particular, Dr Eggman face off in the games- their personalities, catchphrases and insults.

The version of Sonic potrayed in the film feels to me like yet another stereotypical adaptation- like the bad anime dubs we all hate, teamed with "Dr Eggman" uttering phraseology and in-jokes from Carrey's previous films (Dumb and Dumber, The Mask). Yuck!

Whilst the supporting cast are not necessarily bad, the most glaring problem here is that it is quite apparent that the folks who made this film either never played the games, or have never really experienced the fandom, or even watched Sonic X (even the shitty 4Kids dubbed version currently on Netflix). This point was highly apparent to me when that first trailer was dropped. What the hell were they thinking?!

The updated character design did not thrill me either, but by that point I had decided it would have to do, nothing we could do further as ranting on social media, as we know, just provides entertainment for trolls and online bullies.

There are two positive aspects to come out of this film though which can't be ignored. The first is that it raises a new awareness about Sonic, particularly for me living in a country where SEGA does very little effort (read: the licensee is lazy and more interested in selling overpriced popcorn and coke at their mediocre movie theatre complexes whilst trying to be the authority on piracy ergo having "torrenters" and "pirates" locked up for filesharing on the internet).

The second is that with awareness comes the marketability of Sonic merchandise, which, as you know, has never really been sold here, ever! Let's hope that one day soon I can walk into a comic book store and buy Sonic merchandise, presently a privilege only available to you if you live in Japan or the United States!

彼女、お借りします

「Kanojo 、Okarishimasu」

Reiji Miyajima's manga gets TV anime adaptation

I first became aware of Kanojo Okarishimasu when I discovered Paolo from Tokyo's YouTube video "Day in the life of a Japanese Manga Creator", roughly two years ago (can't remember). That video gives a never-before-seen view inside the studio of a real mangaka, and specifically Miyajima-san's studio. I was instantly taken aback by this particular mangaka's attention to detail and artistic expression. From that moment on, I began to read the manga, and eagerly awaited the anime which began airing on the 10th of July 2020.

Story-wise, Kanojo Okarishimasu explores the concept of rent-a-girlfriend, something we joked about as long ago as 2003 when I was a young, naive, Japanese language student. At the time there was actually talk of such a service being made a reality- as I recall it was actually available as a kind of virtual girlfriend service. The idea is pretty timeless and Miyajima-san puts together a refreshing take on it- in the form of a tragi-comedy centering around Kazuya Kinoshita, a college student who decides, following a painful breakup, to rent a girlfriend, with intreresting and unexpected results.

Beautifully animated, and faithful to Miyajima-san's character designs the anime was eagerly anticipated and I can see why- its good. The first episode does not disappoint, it follows the same pace as the manga and is fun to watch. Despite all the gloom and doom, 2020's anime season was pretty okay. Don't miss this one!

These two adorable characters, categorised as "yuru-chara" (see box for detailed explanation) represent Sapporo City in Hokkaido.

Yuru-chara are mascots in Japan, each one representing a particular town or region with the purpose of promoting their respective town or prefecture.

The pink bear, named KOAKKUMA was deported from the outer evil planet "Akkuma" to Hokkaido in 2008 because of her goodwill. Her mission is to give smiles to everybody and spread the idea "Smile Change The World". Her counterpart, the purple bear named AKKUMA has come from the outer evil planet "Akkuma" to Hokkaido to chase deported KOAKKUMA.

He is described a lazy bear but actually he is extremely good at playing guitar, and often plays in gigs.

Together, and with other yuru-chara, such as Kaparu and Nyango Star, these two bears have performed all over Japan since 2008, as well as internationally. They are quite active on social media for the most part.

During 2020 and even into 2021, they've been more active than usual, doing live streamed events and competitions in lieu of actual outdoor festivals for obvious reasons. Together the duo have produced two songs with accompanying music videos. They're catchy as heck and pretty cool :)

Official website:
koakkuma.jp

WHAT ARE "YURU-CHARA"?

The term yuru-kyara (ゆるキャラ) is a contraction of ゆるいマスコットキャラクター which means yurui mascot character.

Specifically they are also referred to as gotochi-kyara (ご当地キャラ) which means “local character”. As such they are defined as mascots for the region, town, or prefecture they represent.

The three general rules for a yuru-kyara are:

1. Must convey a strong message of love for the hometown or local region they represent. Therefore they should try to portray some aspect of the place they represent, e.g. local produce, historical figures, legends, and the like.

2. Movements or behaviour should be unique, and unstable or awkward.

3. Must be unsophisticated or laid-back and lovable.

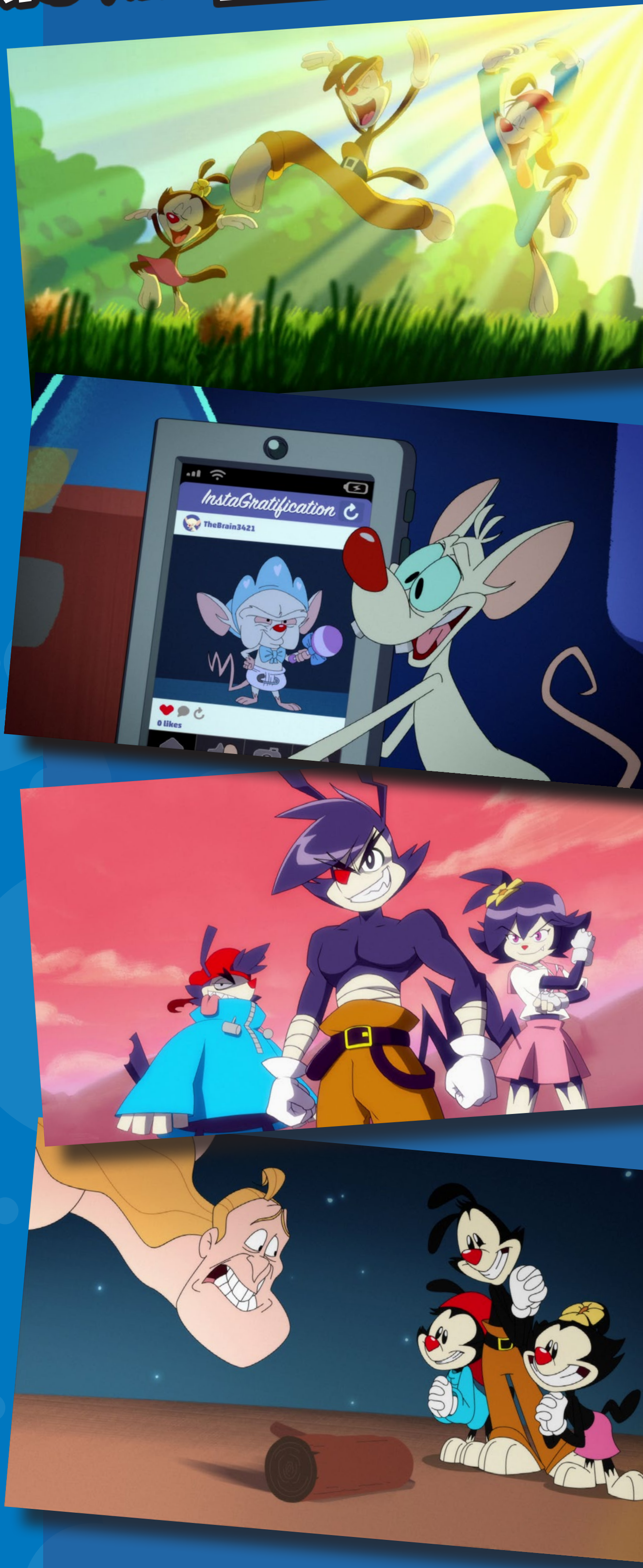
Mascots have an enduring popularity in Japan, linked to historical emotional bonds to non-human characters.

A hulu ORIGINAL
STEVEN SPIELBERG PRESENTS
ANIMANIACS
Zany, Insany and
ALL NEW ANIMANY!

WOWSER, THEY'RE BACK!

(ZANIER, CRAZIER AND WITH *MORE* ATTITUDE)

Without a doubt, one of the best animated shows of the 1990s can only be the Animaniacs. Even today, by today's standards it was cleverly packaged adult humour in the form of animation aimed supposedly at kids.



PROMO IMAGE - WARNER BROTHERS ANIMATION / HULU   SCREEN CAPS FROM SEASON 1 - WARNER BROTHERS / HULU ORIGINAL PRODUCTIONS

In 2020, before all the doom and gloom I came across information that the show was being rebooted by HULU, and while information was relatively hard to come by, it didn't matter because I was not worried. Somehow I knew the reboot would capture the core of the show in the 90s and bring it up to date for the 21st century. Then we had the world go crazy on us which made me forget about this for a while until the end of the year.

Surprisingly quickly, November 2020 rolled around and I was able to watch the first season, and honestly I cannot fault this. Its even better than the original in many aspects- the hidden sexual innuendos and adult jokes are as bold as ever, not forgetting Pinky and The Brain, as witty and calculating on an even grander scale than before. And of course, there's homage paid to the greatness of Japanese anime in one of the first few episodes with the Warners rendered as anime characters complete with power-ups and saiyan modes. That was, in a word, awesome. Even the audio and effects made me wonder what anime shows they were parodying.

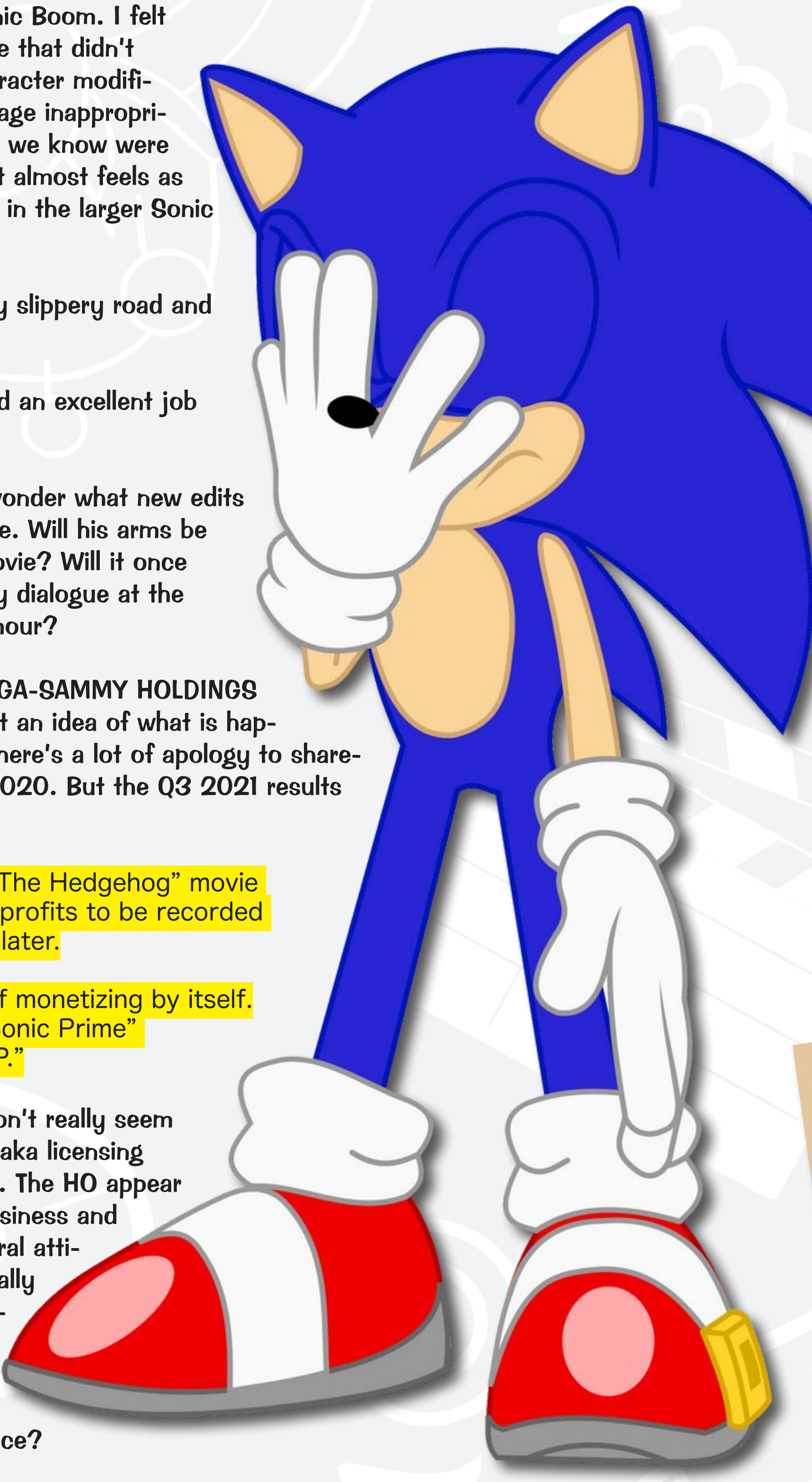
With high resolution, clean vectored artwork, and minor character reworkings that make the characters even more expressive than in 1992, one would be hard pressed to fault this reboot. The only fault I can find however is that the 21st century, as much as we appreciate it and take it for granted, has been introduced into the show with social media influence and cognisance of the snowflake generation, which is rather a damper.

It is officially stated that the Nurse, and other characters such as Minerva Mink have been removed because apparently they're too sexualized for children to handle. Which is really odd because I was a child when this show was on TV and I don't remember or even recall having experienced any negative effects. But hey, you know the rule of the 2020s- if its even slightly likely to cause trouble, just don't do it. The internet (and the world) is full of SJWs ready to burn the place down when their feels are offended.

Despite the single flaw that I've mentioned, I cannot recommend this show enough. You've got to see this. It comes as no surprise then that a third season is in the works! That's all folks!

Here we go again… Another 3DCGI !

NETFLIX SONIC PRIME™



All right, I was not particularly happy with Sonic Boom. I felt that it was a "repackage" for a younger audience that didn't "get" Sonic from the 1990s era. Aside from character modifications that were minor, the whole show just felt age inappropriate to me as an adult. The Sonic Boom games, as we know were mediocre performers (not just in sales) too, and it almost feels as if that incarnation of Sonic was just a passing fad in the larger Sonic ecosystem.

Now it seems Netflix want to go down that already slippery road and come up with their own show…

I don't think NETFLIX do bad work, I think they did an excellent job with other shows, but…

I cannot help but feel ambivalent about this and wonder what new edits this new show will bring to the characters this time. Will his arms be blue? Will he have two separate eyes as in the movie? Will it once again be aimed at 5 year olds? Will we have corny dialogue at the grade 5 level with absolutely no typical Sonic humour?

Before I wrote this opinion, I read through the SEGA-SAMMY HOLDINGS Annual Report for 2020 to Shareholders(1), to get an idea of what is happening at SEGA-SAMMY, and its quite clear that there's a lot of apology to shareholders for the train smash of a year which was 2020. But the Q3 2021 results Q&A(2) are far more revealing and I quote:

> "We do not expect profits from "Sonic The Hedgehog" movie in this current fiscal year, and we expect profits to be recorded in the next fiscal year or later.
>
> About "Sonic Prime", we're not thinking of monetizing by itself. Our stance is to implement the "Sonic Prime" as part of strengthening the IP."

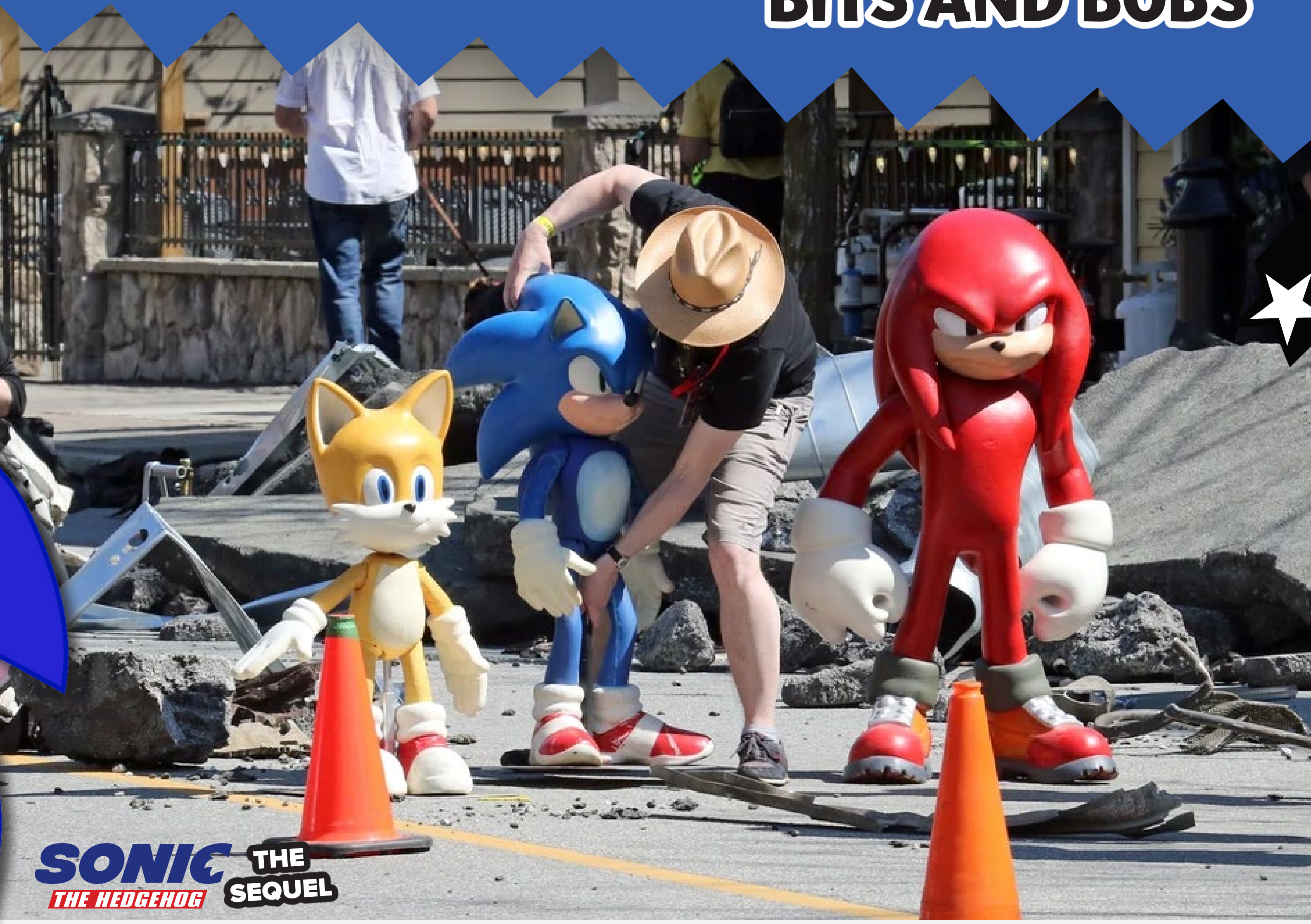
So what does this mean? Simply put, they just don't really seem to care about Sonic- as long as he brings in cash aka licensing fees- anyone can license our blue blur these days. The HO appear always far more concerned with their pachislot business and other game titles. This is consistent with the general attitude and feeling towards Sonic in Japan. Its basically just us Western folk who go nuts over the character and buy the merch and games.

The potential for better games and merchandise is there, but sadly nobody wants to take the chance?

References:
1.https://www.segasammy.co.jp/english/ir/library/pdf/printing_annual/2020/al2020_e_all_print.pdf
2. https://web.archive.org/web/20210226203540/https://www.segasammy.co.jp/english/ir/library/pdf/settlement/2021/3q_main_qa_e_final_.pdf

ART BY sonicdash759

BITS AND BOBS



THE INTERNET IS AWESOME - FOR LEAKED IMAGES THAT IS!

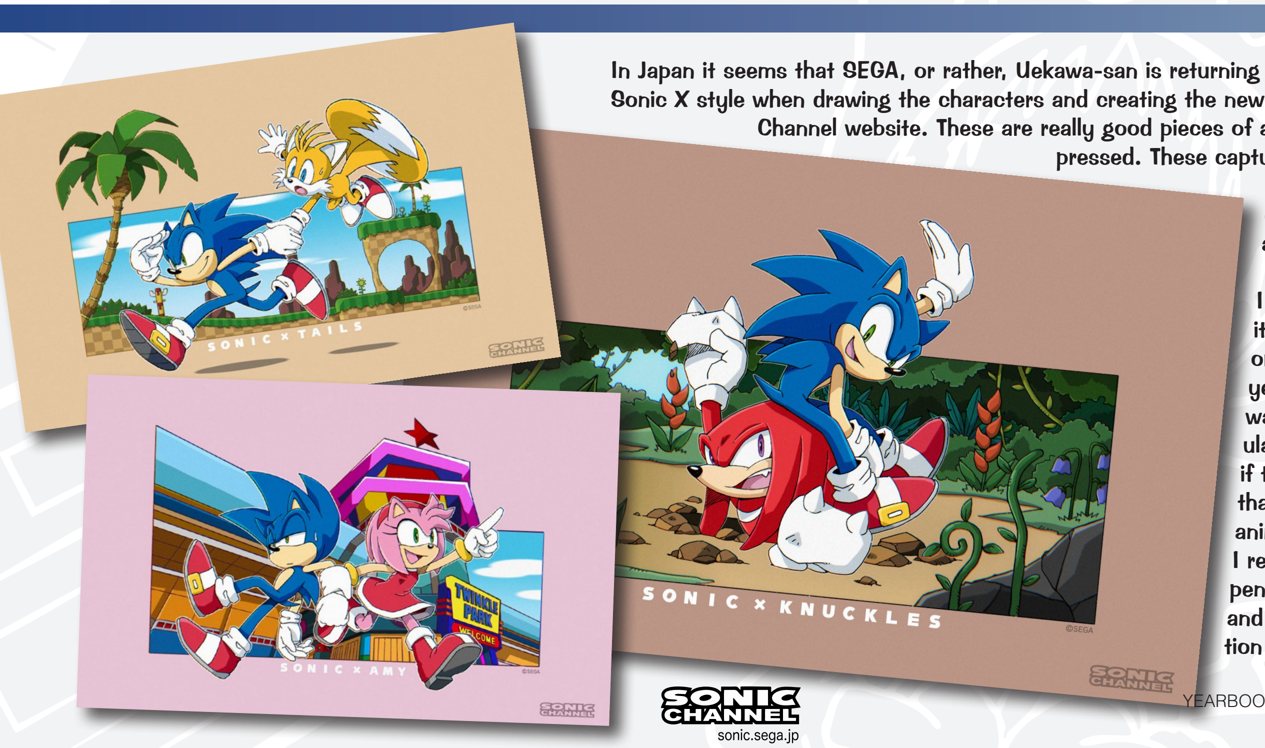
Well, if the above leaked image from the filming of Paramount's SONIC THE HEDGEHOG 2 is anything legit, aside from Sonic's incorrect colour scheme and eyes, it seems the rest of Team Sonic got off with little to no changes. This looks quite passable, and those statues (apparently called "standees") look like they're really well made. I wouldn't mind one of those myself.

According to numerous sources the fiming is complete and the rest of the production work is underway, on schedule for a 2022 release. My questions now shift to more fundamental ones- Where's Amy? Is she going to show up somewhere? Or will be be surprised at the end of the film?

Can you imagine the girl running in with her hammer squealing "Whooooah its my hero, Sonic". That would be quite funny!

In Japan it seems that SEGA, or rather, Uekawa-san is returning to what I suspect is the Sonic X style when drawing the characters and creating the new wallpapers on the Sonic Channel website. These are really good pieces of art, and I am really impressed. These capture much of the nostalgia of the Sonic Adventure era as well as the Sonic X anime.

I personally love this style, it looks just as fresh and original as it did nearly 20 years ago when the anime was at the height of its popularity. I do wonder however if this is a teaser, or a hint that there might be a new anime in the works, although I really doubt that would happen with a new feature film and 3DCGI series in production as I write this.



SONIC CHANNEL
sonic.sega.jp

TAILS' WORKSHOP

Rebuilding the Seagate Central NAS that died (with all my freaking anime on it)

For the last 8 years I have relied on this NAS unit to store my ever-expanding anime collection. This NAS was pretty okay, not very fast but it was fit for purpose. I knew from the beginning it probably ran Linux and I was right about that when one day I managed to log into the OS via SSH (secure shell). Unfortunately this is Seagate at its consumer best- no upgrade path, and indeed, I hadn't the time or interest to poke further. If it ain't broke, don't fix it...

Now this NAS had a USB port and with my Linux knowledge I knew I could format an external hard drive as an ext3 volume and let it be auto-mounted. I did that with great success and it ran for the last three years serving TV programs to the TV. That drive is still working, albeit now mounted on a Linux box (Ubuntu Server 18.04LTS running on a Core i3 HASWELL motherboard recovered from the trash)

So on one fine day in June, I noticed this NAS was struggling- I had a look, at which point I discovered that when shares residing on its internal drive were being accessed, it would choke, freeze, and sometimes reboot. Uh-oh! Not good! Sonic......

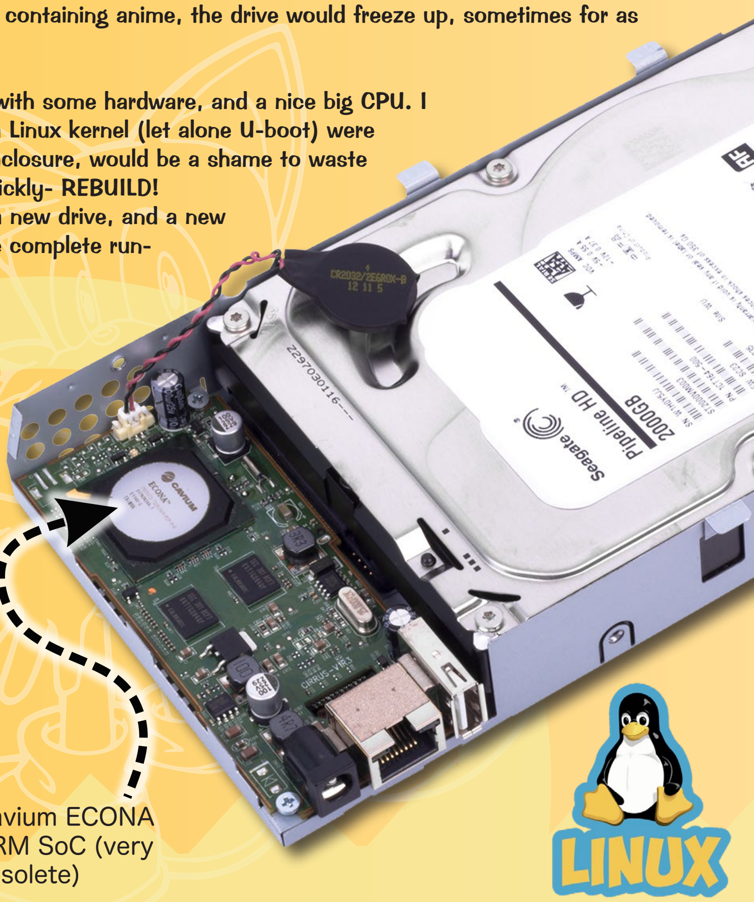
I had a closer look and my worst fears were confirmed, the hard drive was failing! What made this worse was that the actual OS (Linux) was booted from the same drive. In normal (non cost-sensitive) circumstances there would be a large eMMC (flash storage) on the mainboard running the OS so that the unit can be repaired with a new drive. Not in this case alas. I managed to recover 100Gbytes of my anime and other stuff, before it failed for good and the unit refused to boot ever again. Neither disk recovery utilities or any other tool were of any use, every time an attempt was made to read the directories containing anime, the drive would freeze up, sometimes for as long as 30 minutes.

So when I took the unit apart to try and recover the drive I was greeted with some hardware, and a nice big CPU. I quickly found out how obsolete that chip was, and my hopes of buildng a Linux kernel (let alone U-boot) were dashed. I decided to think a little bit further out the box- This is a nice enclosure, would be a shame to waste it, so I thought "hmm... what can we do with this?" The answer came quickly- REBUILD! Like putting a new engine into the Tornado, I decided I was going to put a new drive, and a new CPU in there. Unlike Sonic anime that omits a lot of details, I will give the complete rundown on how I did this.

Brainstorming

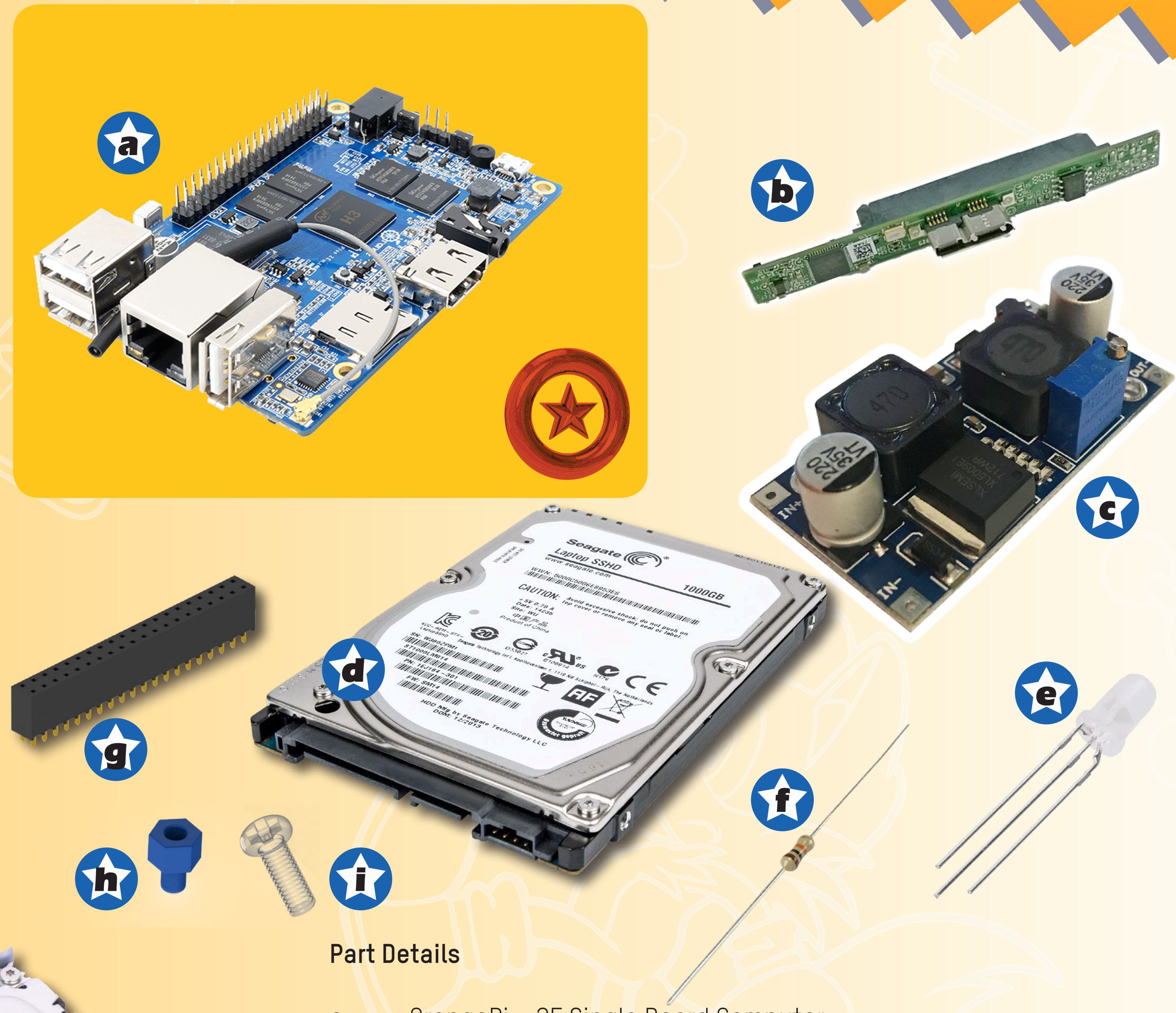
It occurred to me that, I had, in my workshop, an OrangePi +2E SBC, an external hard drive whose drive was dead as a doornail (the USB interface electronics was OK though) and a wall wart with a 5VDC output. I had the idea to put these parts together and do a proof of concept. The results exceeded my expectations, it didn't just work, it became a more than an acceptable solution, and I decided at that point to build all of this into the original housing.

Before we get started with the actual build, let's spend a moment getting familiar with the parts we will be using to make this happen.



Cavium ECONA ARM SoC (very obsolete)

Parts



Part Details

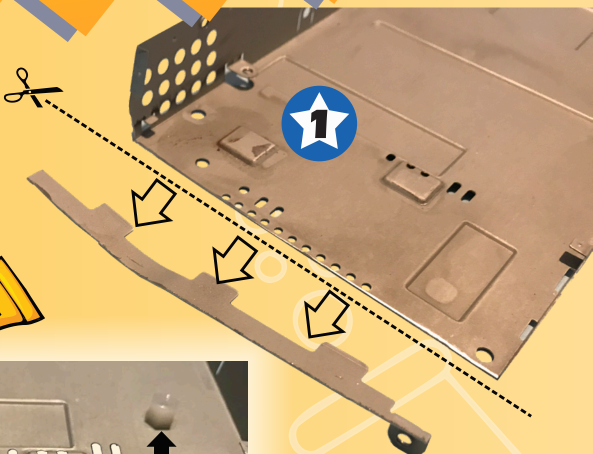
a	-	OrangePi + 2E Single Board Computer
b	-	Controller salvaged from an external Seagate drive
c	-	DC to DC converter adjustable (ACM)
d	-	SATA hard drive or SSD (capacity to suit your requirements)
e	-	Bi-colour LED (5mm typ. 3 pins common cathode)
f	-	Carbon Resistor (180 ohms 1/4 watt) 2 pieces
g	-	DIL header 40 pins 2 rows 2.54mm pitch
h	-	Plastic standoff M3 thread (height 3mm)
i	-	Plastic machine screw (M3 x 8) (trimmed to desired length with knife)

Insides of the Seagate™ Central 2TB NAS (plastic shell un-clipped)

THE GOLD STANDARD

Since writing this article, it has been referenced not only to build new file servers, but repair damaged NAS too. This is a complete reference for mounting Linux filesystems and shares with Samba and Linux.

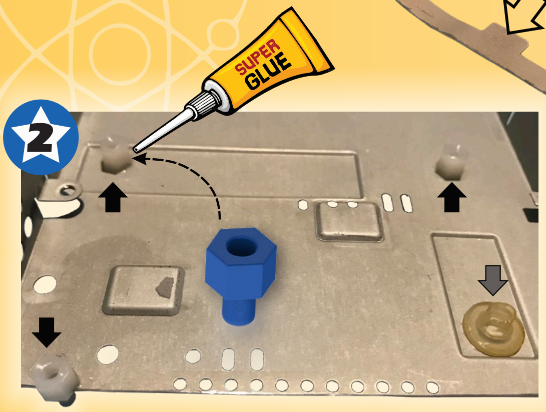
Part I - H/W Build



Disassembly...

I began by removing all of the insides of the NAS unit. The plastics were tricky to get apart but once done it was all good (the trick was to remove the front and back panels)- I could get all the electronics out. I opened the hard drive to see what went wrong, and its quite apparent this drive ingested a foreign object which caused a head crash leaving a mark on the platter (there was an event- something fell on this unit a few months prior).

Once I had the bare metal frame, I began by cutting away the metal to position the SBC* in place. (Figure 1) Once the optimum position was found (so that the Ethernet socket and USB socket lined up with the cutouts in the plastic panel) I drilled holes and inserted threaded plastic standoffs, which were glued in place (figure 2).

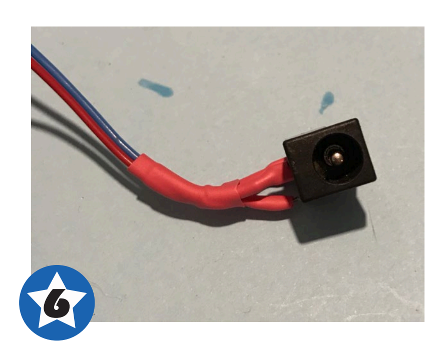


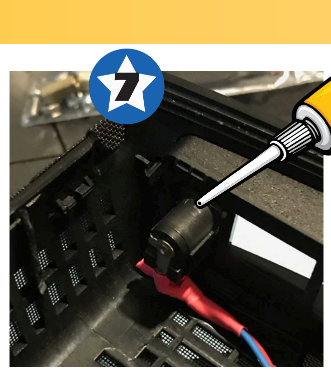
The next step once the SBC was confirmed to fit, was to remove the stacked USB connector (in order to make space for the power jack). The idea is to extend that connector on wires so that it can be located elsewhere in the enclosure (Figure 3).

Next I chose the right edge (or lower edge- depending on how you view it) to mount the laptop hard drive. A simple matter of drilling a few holes and installing it with screws did the trick (figure 4). Because the drive was supported only on one edge I decided to add a support beneath it, by cutting packing foam to size and gluing it in place (figure 5).

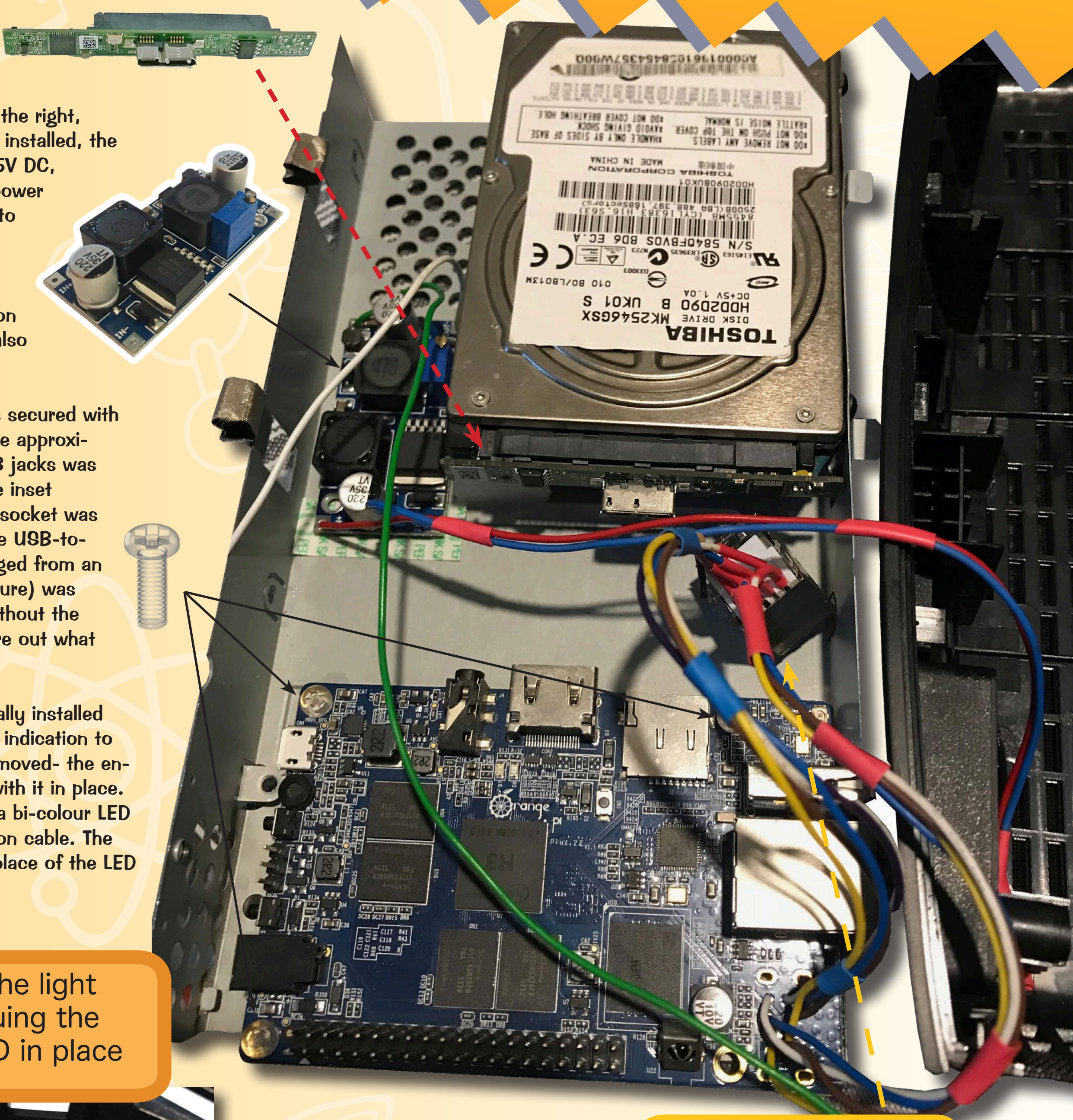
Next up was the power connector. I removed it from the old board, and then attached two lengths of wire to it. Using heat-shrink sleeve here is essential- we don't want any risk of that touching the SBC (figure 6).

Next I glued this connector into the top housing side cover with the pins facing upwards (figure 7 and 8).





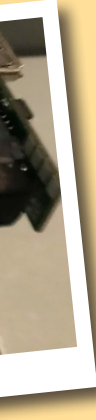
Putting parts together...



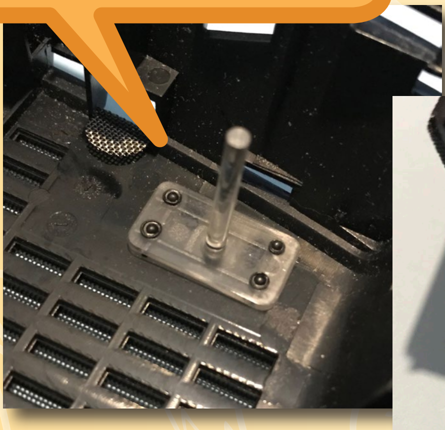
The next stage involved assembling all the various parts into the metal base. As shown on the right, the DC-DC converter was installed, the output was adjusted for 5V DC, and wires attached. The power jack was wired directly into the DC-DC converter. The plastic machine screws were used to secure the SBC in place, the extension of the USB sockets was also performed.

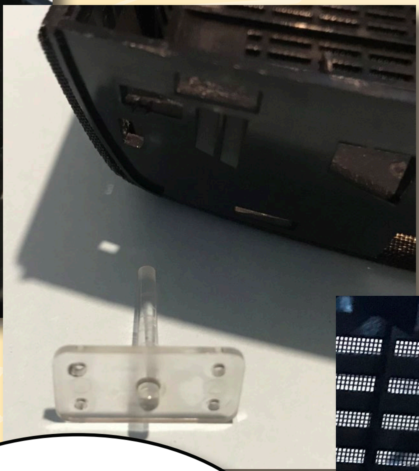
The DC-DC converter was secured with double-sided tape, and the approximate location for the USB jacks was determined as shown. The inset shows details of how the socket was extended off the SBC. The USB-to-SATA controller (we salvaged from an external hard drive enclosure) was fitted to the hard drive without the cable so that I could figure out what goes where.

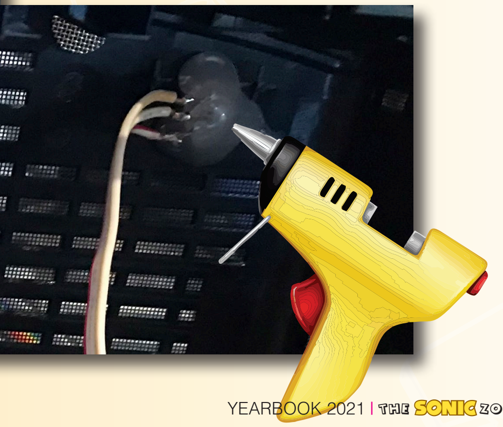
Next, the light pipe originally installed in the housing (for status indication to the outside) had to be removed- the enclosure would not close with it in place. Therefore I chose to use a bi-colour LED soldered onto 3 way ribbon cable. The LED was simply glued in place of the LED light pipe.

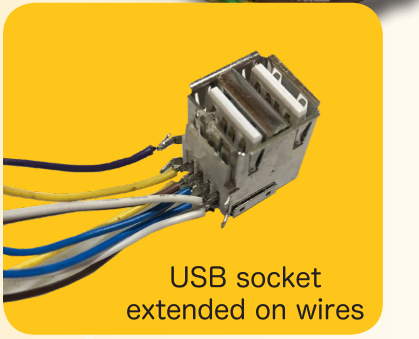


Removing the light pipe and gluing the bi-colour LED in place









USB socket extended on wires

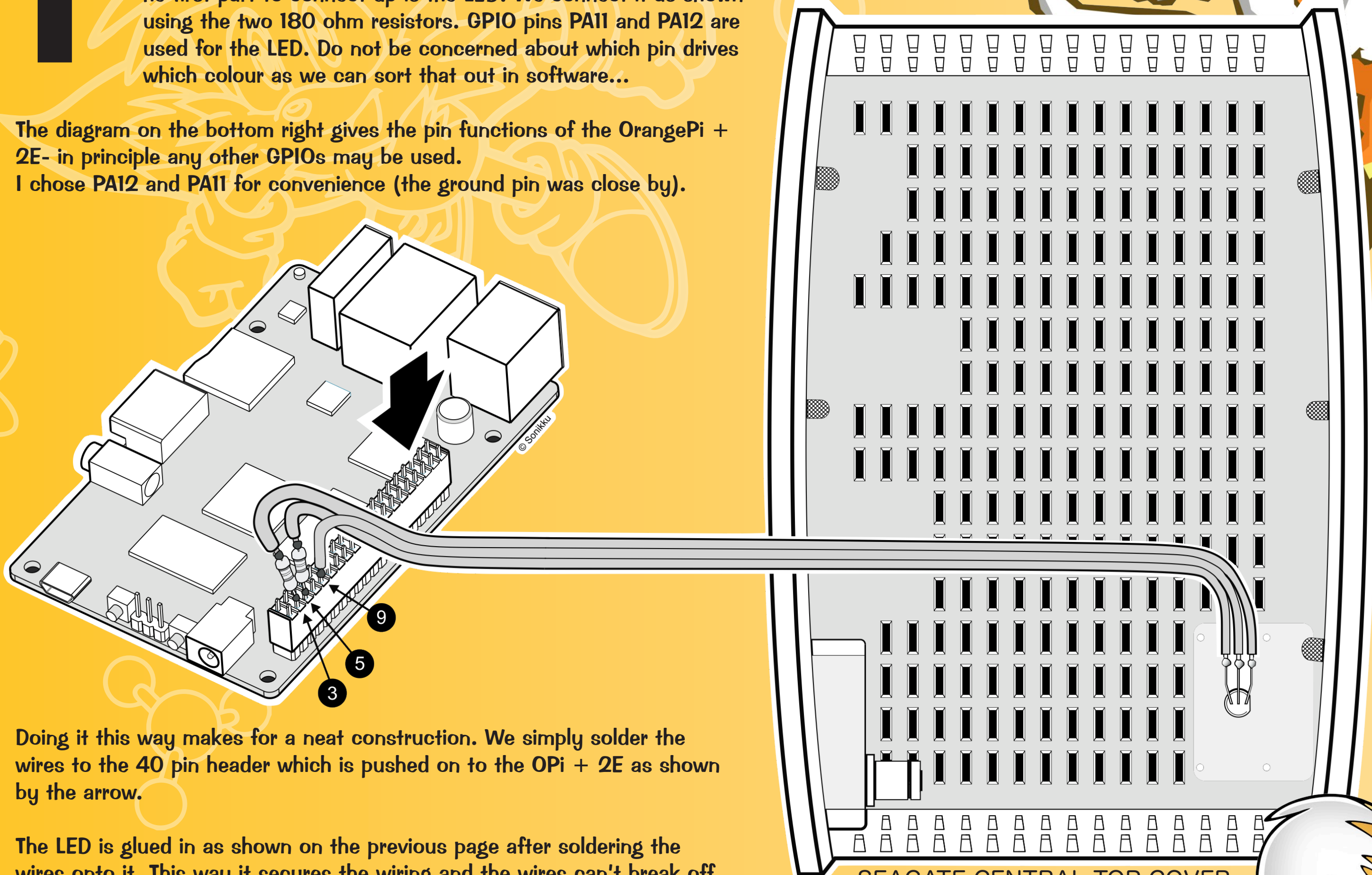
Don't use superglue to affix the LED- cyanocrylate adhesive tends to make the plastic dull and therefore it ruins the UX.

Build...

LED Interconnect

The first part to connect up is the LED. We connect it as shown using the two 180 ohm resistors. GPIO pins PA11 and PA12 are used for the LED. Do not be concerned about which pin drives which colour as we can sort that out in software...

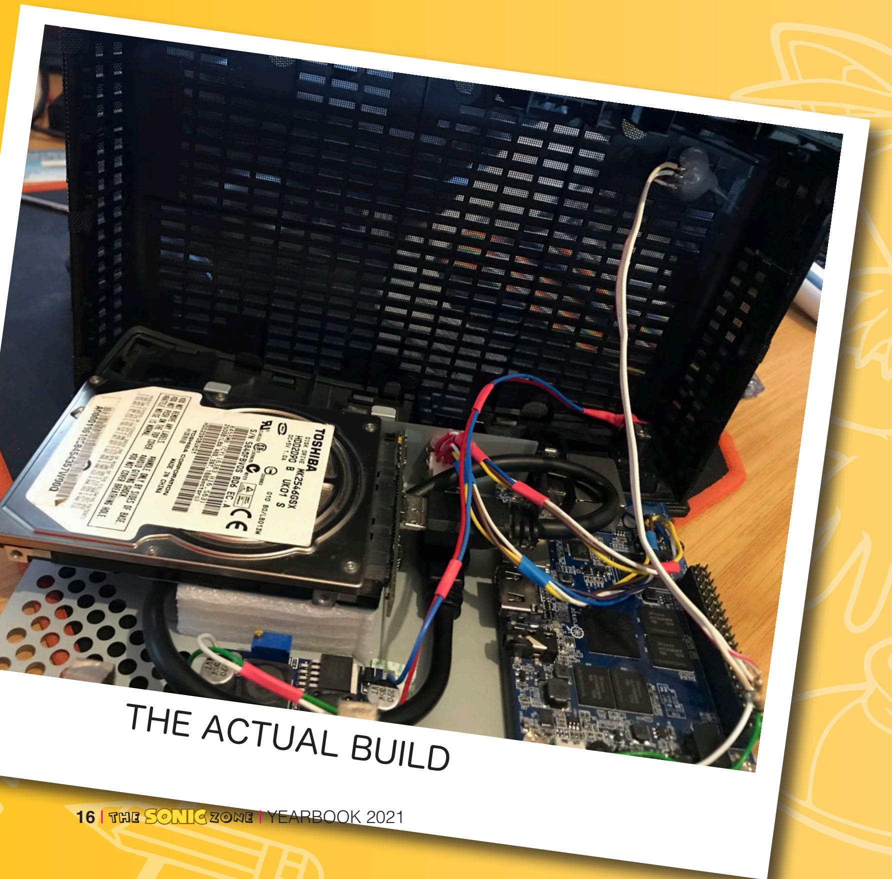
The diagram on the bottom right gives the pin functions of the OrangePi + 2E- in principle any other GPIOs may be used.
I chose PA12 and PA11 for convenience (the ground pin was close by).



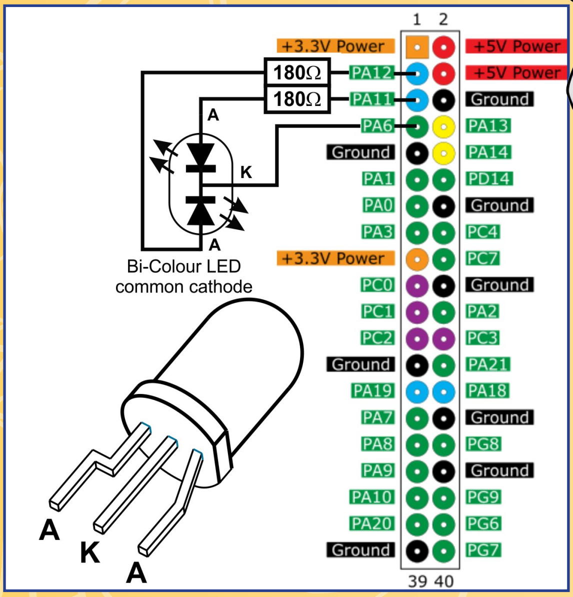
Doing it this way makes for a neat construction. We simply solder the wires to the 40 pin header which is pushed on to the OPi + 2E as shown by the arrow.

The LED is glued in as shown on the previous page after soldering the wires onto it. This way it secures the wiring and the wires can't break off.

SEAGATE CENTRAL TOP COVER



THE ACTUAL BUILD



	1	2	
+3.3V Power			+5V Power
PA12			+5V Power
PA11			Ground
PA6			PA13
Ground			PA14
PA1			PD14
PA0			Ground
PA3			PC4
+3.3V Power			PC7
PC0			Ground
PC1			PA2
PC2			PC3
Ground			PA21
PA19			PA18
PA7			Ground
PA8			PG8
PA9			Ground
PA10			PG9
PA20			PG6
Ground			PG7
	39	40	

OPI+2E PIN DIAGRAM AND LED PIN DESIGNATIONS

Power

The DC jack wires are connected directly to the DC-DC module as shown below (for clarity the LED wiring discussed previously is not shown) The output of the DC-DC module is connected to pins 2, 4 and 6 of the 40 pin connector as shown on the drawing. These pins are adjacent to the ones used for the LED. On the previous page the handy pin diagram also indicates which pins are used for power. Note that the two 5V pins are connected together using a short length of wire.

BEFORE I connected power to the OrangePi it was critical to adjust the DC-DC converter to the correct voltage. Usually these ship from the factory set to 30V DC which is way too high for the SBC. If I connected it without adjustment it would destroy the SBC instantly.

Therefore, the procedure was to unplug the connector from the SBC, connect the 12V AC adapter, confirm 12V was entering the DC-DC converter by measuring the input voltage using a multi-meter and then while measuring the output terminals with the multi-meter, turn the screw on the DC-DC converter's trimpot until the voltage went down, until it approached 5 volts. With a bit of fine adjustment I was able to get the output to be exactly 5.0V.

Once I had confirmed that the voltage was correct, I was able to disconnect the wall adapter, plug the connector into the SBC, and plug the wall adapter into the mains to confirm everything works. I was happy to see the green LED on the OrangePi (the LED near the HDMI connector) turn green. This was a sign of a good boot [by default these boards ship with a Chinese version of Android, so if you connect a monitor or TV you would see the user interface].

Once I had confirmed the board, and electronics was up and running it was safe to mount everything together in the chassis. The OrangePi was mounted using the plastic M3 machine screws as shown in the photograps. The photograph on the previous page also shows how the hard drive was mounted, the SS USB cable was coiled under the drive, the space between the foam packing and the sidewall of the metal chassis being ideal.

Stuff to Download

Before putting the entire unit together and closing it up, it was necessary to install software, particularly the operating system, and confirm it all worked well. For this job, I used the following:

- Micro SD card 4GByte (class 6 or better)
- USB keyboard and mouse
- PC monitor connected via HDMI (any resolution)

The following software is also required (usually I download the latest versions before starting a new project)

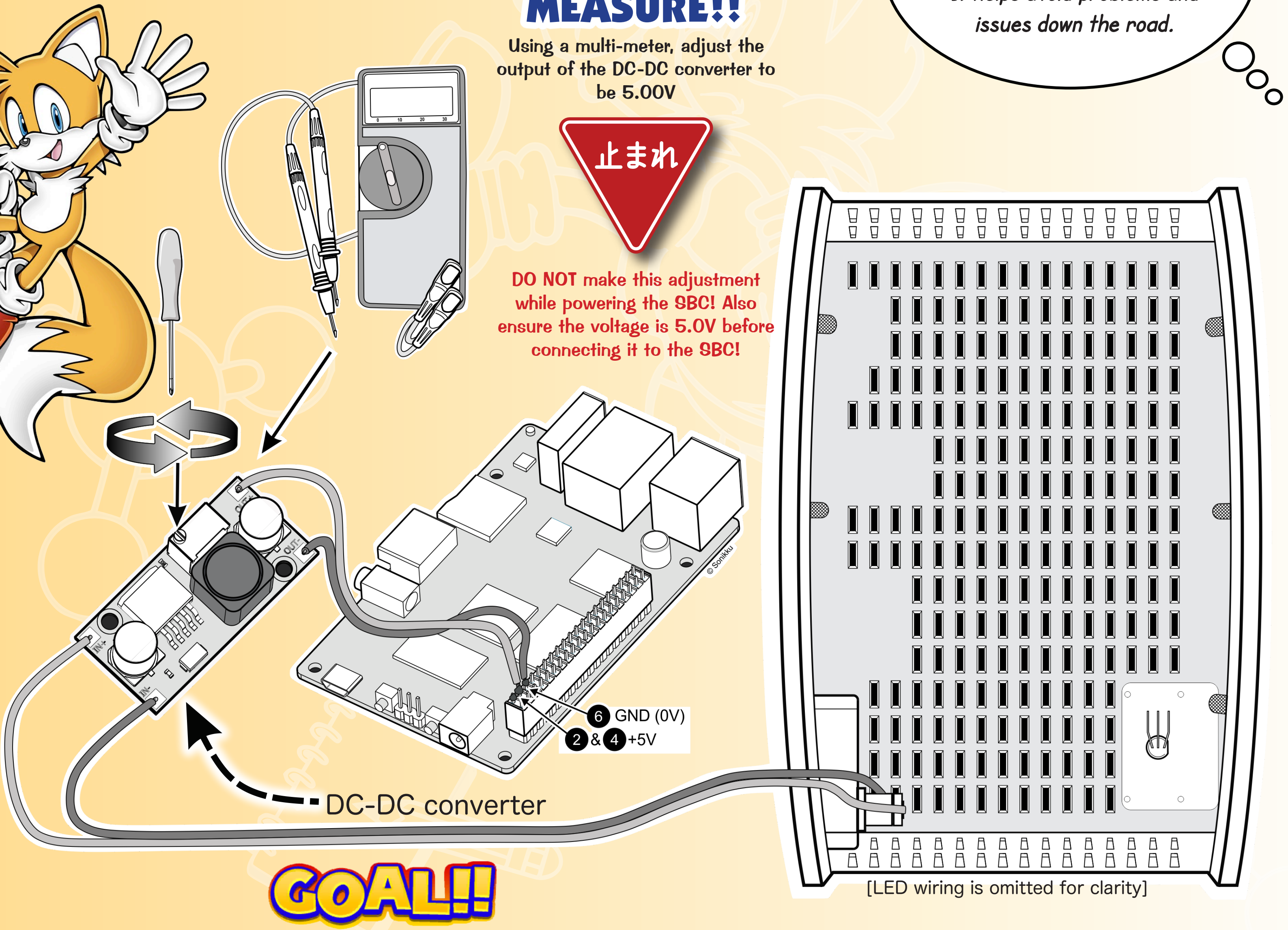
- Balena Etcher
- Armbian for OrangePi + 2E. This is a complete Debian Linux OS. I used Armbian Focal at the time of this writing.
- A good quality SD card reader
- Ethernet cable to connect the OrangePi to your local network.

Always download the latest version of software used before starting a new project. It helps avoid problems and issues down the road.

MEASURE!!

Using a multi-meter, adjust the output of the DC-DC converter to be 5.00V

DO NOT make this adjustment while powering the SBC! Also ensure the voltage is 5.0V before connecting it to the SBC!



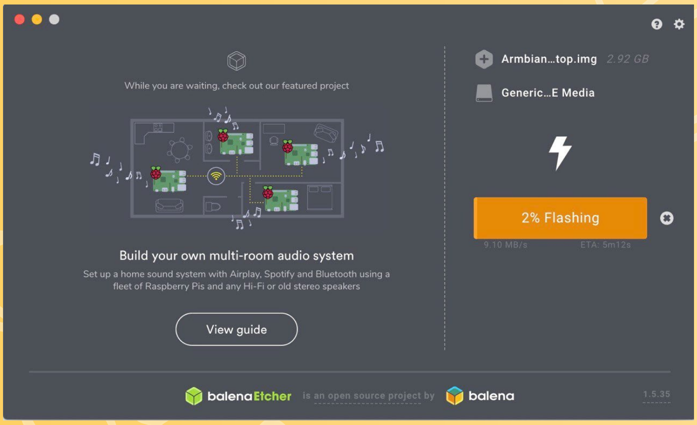
[LED wiring is omitted for clarity]

It’s Linux!

The way I did it here is merely suggestive and was how I got my system working. You are absolutely encouraged to use different software according to your needs and use case!

First things first- SD card

Using Balena Etcher, we write the Armbian image to the SD card from which we will boot the system. Balena will verify the image written so there's no need to worry, just be patient and relax until it is done. Writing speed is highly dependent on the speed of the card, so always use the fastest one you can get...



While you are waiting, check out our featured project

Build your own multi-room audio system

Set up a home sound system with Airplay, Spotify and Bluetooth using a fleet of Raspberry Pis and any Hi-Fi or old stereo speakers

View guide
Armbian...top.img 2.92 GB
Generic...E Media
2% Flashing
9.10 MB/s
ETA: 5m12s
balenaEtcher is an open source project by balena
1.5.35

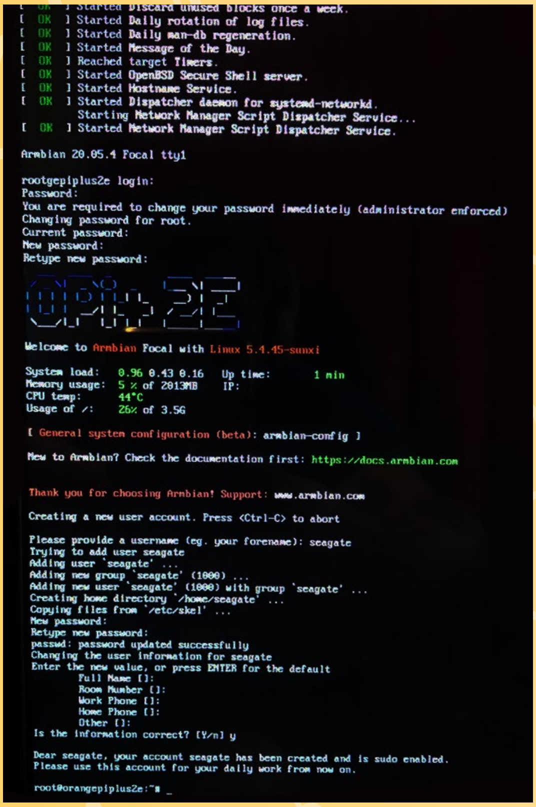
Insert SD card, connect screen, mouse and keyboard and POWER ON!

The OPi + 2E will boot from the SD card you prepared with Balena Etcher. Depending on which version of Armbian you use (there will likely be newer versions by the time this is published), there may be subtle differences but the overall configuration and set-up process is the same. Armbian Linux will boot from the SD card, and when it is finished loading the necessary environment, you will then be required to set a root password ('root' is the super-user account that has full system access). Then you should create the regular user account that you are planning to use to do all the system related tasks using the useradd command:

The Linux command to add a new user is:

```
useradd <username>
```

Because you’re already logged in as root (remember you’ve just set the root password) you are able to easily create this user account. When this completes you will essentially have the root user account and the user account you created. In my case I named the user account ‘seagate’ to match the device I replaced.



```
[  OK  ] Started Discard unused blocks once a week.
[  OK  ] Started Daily rotation of log files.
[  OK  ] Started Daily man-db regeneration.
[  OK  ] Started Message of the Day.
[  OK  ] Reached target Timers.
[  OK  ] Started OpenBSD Secure Shell server.
[  OK  ] Started Hostname Service.
[  OK  ] Started Dispatcher daemon for systemd-networkd.
         Starting Network Manager Script Dispatcher Service...
[  OK  ] Started Network Manager Script Dispatcher Service.

Armbian 20.05.4 Focal tty1

rootgepiplus2e login:
Password:
You are required to change your password immediately (administrator enforced)
Changing password for root.
Current password:
New password:
Retype new password:

 Welcome to Armbian Focal with Linux 5.4.45-sunxi

 System load:   0.96 0.43 0.16   Up time:       1 min
 Memory usage:  5 % of 2013MB    IP:
 CPU temp:      44°C
 Usage of /:    26% of 3.5G

 [ General system configuration (beta): armbian-config ]

 New to Armbian? Check the documentation first: https://docs.armbian.com


 Thank you for choosing Armbian! Support: www.armbian.com

 Creating a new user account. Press <Ctrl-C> to abort

 Please provide a username (eg. your forename): seagate
 Trying to add user seagate
 Adding user `seagate' ...
 Adding new group `seagate' (1000) ...
 Adding new user `seagate' (1000) with group `seagate' ...
 Creating home directory `/home/seagate' ...
 Copying files from `/etc/skel' ...
 New password:
 Retype new password:
 passwd: password updated successfully
 Changing the user information for seagate
 Enter the new value, or press ENTER for the default
         Full Name []:
         Room Number []:
         Work Phone []:
         Home Phone []:
         Other []:
 Is the information correct? [Y/n] y

 Dear seagate, your account seagate has been created and is sudo enabled.
 Please use this account for your daily work from now on.

 root@orangepiplus2e:~# _
```

ARMBIAN FIRST BOOT SCREEN

Get it online & set-up...

Now you need to connect the OrangePi +2E to the internet somehow. To do this I recommend using the Ethernet port and not the wireless interface. In most cases this is as simple as plugging a cable into your router and the other end into the OrangePi. If all goes well you can execute a ping command from the shell:

```
ping google.com
```

Responses from the pings indicate the board has internet access. In this case you can proceed to launch armbian-config. If you are already root then its simply:

```
armbian-config
```

or

```
sudo armbian-config
```

(when not logged on as root)

This will launch a menu-driven, interactive configuration tool with a blue screen. The list of things you need to configure are given in the configuration checklist.

armbian CONFIGURATION CHECKLIST

The following items should be configured (and installed where applicable) using the armbian-config utility (note these changes will be committed to the image running from the SD card):

- Set the hostname under the *Personal Settings* menu (I used 'seagatecentral').
- Set the locale and timezone according to where the system is located, also under the *Personal Settings* menu.
- Under the *Software* menu is an option to install kernel headers. This should be performed.
- Under the *System* menu is an option to install the firmware packages. This should also be performed.
- Additionally, under the *System* menu is an option to configure the hardware. Use this to enable GPIO and I2C, SPI etc (note without this step the LED will not work nor will you be able to control it later from Python).

Reference: https://docs.armbian.com/User-Guide_Armbian-Config/

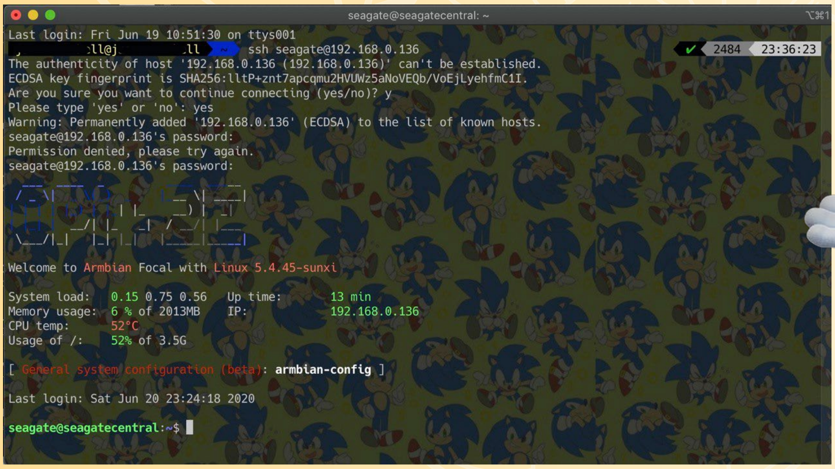
Install Samba... remotely...

The next step is performed while the screen and keyboard are still attached, but we will access the system via the network, using secure shell (SSH). This is a convenient way to ensure the networking is good and to do things from the desktop or laptop. By default the SSH daemon should be running so all you will need to do is obtain the IP address of your board and use that to SSH into it.

On any UNIX-like system (macOS or Linux)

```
ssh <username>@<ip_address>
```

From the terminal (or shell) simply enter the ssh command and fill in the necessary information (username and IP address of the remote host). The ssh command will then connect to your OrangePi as shown...



```
Last login: Fri Jun 19 10:51:30 on ttys001
[redacted] ~  ssh seagate@192.168.0.136                                    ✔  2484  23:36:23
The authenticity of host '192.168.0.136 (192.168.0.136)' can't be established.
ECDSA key fingerprint is SHA256:lltP+znt7apcqmu2HVUWz5aNoVEQb/VoEjLyehfmC1I.
Are you sure you want to continue connecting (yes/no)? y
Please type 'yes' or 'no': yes
Warning: Permanently added '192.168.0.136' (ECDSA) to the list of known hosts.
seagate@192.168.0.136's password:
Permission denied, please try again.
seagate@192.168.0.136's password:

Welcome to Armbian Focal with Linux 5.4.45-sunxi

System load:   0.15 0.75 0.56   Up time:       13 min
Memory usage:  6 % of 2013MB    IP:            192.168.0.136
CPU temp:      52°C
Usage of /:    52% of 3.5G

[ General system configuration (beta): armbian-config ]

Last login: Sat Jun 20 23:24:18 2020

seagate@seagatecentral:~$ 
```

Note 1: local username blanked to protect my privacy.
Note 2: The IP of the device was set by DHCP to 192.168.0.136.
You can change this to a static IP in armbian-config before you get to this point.

Install Samba... remotely... (cont)

Samba (or smbd) for short is a Linux daemon (service) that provides network sharing compatible with Microsoft Windows, Linux and macOS machines.

Samba is installed with the following shell command (you will be asked for your password when using sudo to elevate user privilege)

```
sudo apt-get install samba
```

When the installation is complete, the daemon will be started using the default configuration. It is necessary to then configure Samba. Before we do that, we create a directory where the content shared on the network will reside. This directory is created under the home directory of the user account and will typically be:

/home/seagate/share

To create the directory, simply execute:

```
mkdir /home/seagate/share
```

The commands shown in the information box will ensure Samba is installed correctly. The directory that is required to be created will be a local directory to test the system (don't worry we will add the hard disk soon and map that as a network share) therefore bear in mind that this directory will be limited in storage capacity by the SD card's available space. Usually this works out to be less than a gigabyte (depends on how big the Armbian image is that is running and the size of the SD card).

Configuring Samba...

Installing Samba is not enough to get it running. We need to configure it to work according our needs. The way to do this is to edit its configuration file which is located in:

```
/etc/samba/smb.conf
```

This editing can be performed using one of the inbuilt text editors. GNU Nano is preferable.

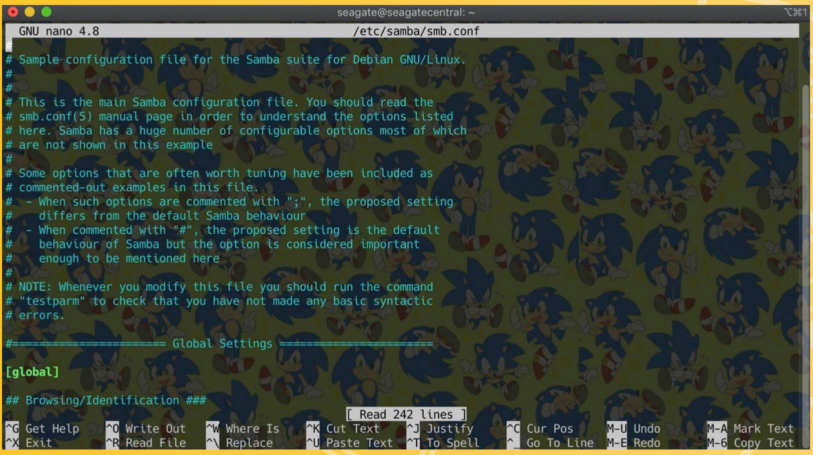
GNU Nano, or nano for short is one of the standard text editors integrated into Linux and most all *nix type operating systems including macOS.

The typical usage to edit a given file is

```
nano <path_to_filename_and_filename>
```

Therefore we edit smb.conf with

```
sudo nano /etc/samba/smb.conf
```

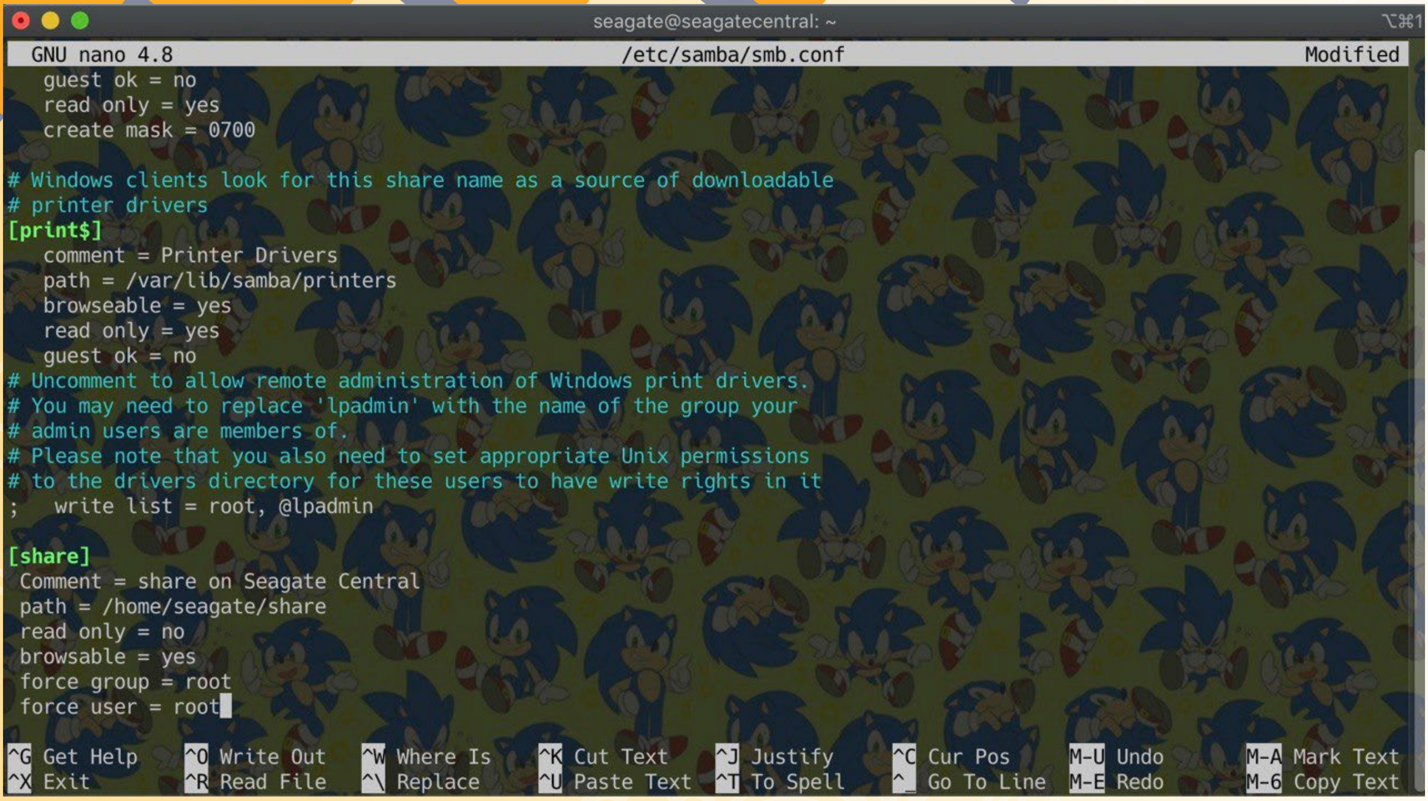


```
seagate@seagatecentral: ~                                        ⌥⌘1
  GNU nano 4.8                        /etc/samba/smb.conf
#
# Sample configuration file for the Samba suite for Debian GNU/Linux.
#
#
# This is the main Samba configuration file. You should read the
# smb.conf(5) manual page in order to understand the options listed
# here. Samba has a huge number of configurable options most of which
# are not shown in this example
#
# Some options that are often worth tuning have been included as
# commented-out examples in this file.
#  - When such options are commented with ";", the proposed setting
#    differs from the default Samba behaviour
#  - When commented with "#", the proposed setting is the default
#    behaviour of Samba but the option is considered important
#    enough to be mentioned here
#
# NOTE: Whenever you modify this file you should run the command
# "testparm" to check that you have not made any basic syntactic
# errors.

#======================= Global Settings =======================

[global]

## Browsing/Identification ###
                              [ Read 242 lines ]
^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos    M-U Undo      M-A Mark Text
^X Exit       ^R Read File  ^\ Replace    ^U Paste Text ^T To Spell   ^_ Go To Line M-E Redo      M-6 Copy Text
```

Launching GNU Nano as per the command provided loads the editor and the smb.conf file is opened, as can be seen in this screenshot the settings are presented in the form of a text file that can be edited.

Commands to the editor are entered by means of Ctrl+ a command key for example Ctrl+X will exit the program.

The majority of Samba's out-the-box or default settings are sufficient for this system, and need not be edited. If you are an advanced user of course then you would probably edit this file according to your needs. However for the novice user, the configuration given in the following text is ideal and perfectly suited for the average home NAS. The relevant section is found towards the end of the smb.conf file, under the section marked as `[print$]`. The following section will show how we make an entry to map the home directory as a network share.



```
  GNU nano 4.8                     /etc/samba/smb.conf                      Modified
   guest ok = no
   read only = yes
   create mask = 0700

# Windows clients look for this share name as a source of downloadable
# printer drivers
[print$]
   comment = Printer Drivers
   path = /var/lib/samba/printers
   browseable = yes
   read only = yes
   guest ok = no
# Uncomment to allow remote administration of Windows print drivers.
# You may need to replace 'lpadmin' with the name of the group your
# admin users are members of.
# Please note that you also need to set appropriate Unix permissions
# to the drivers directory for these users to have write rights in it
;   write list = root, @lpadmin

[share]
 Comment = share on Seagate Central
 path = /home/seagate/share
 read only = no
 browsable = yes
 force group = root
 force user = root

^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos    M-U Undo      M-A Mark Text
^X Exit       ^R Read File  ^\ Replace    ^U Paste Text ^T To Spell   ^_ Go To Line M-E Redo      M-6 Copy Text
```

The above screencap of the smb.conf file being edited shows we added a section under `[share]`. We set the various options such as `path = /home/seagate/share`

As you will recall that was the directory we created in an earlier step. The other options pertain to access rights and are set as shown, for the moment.

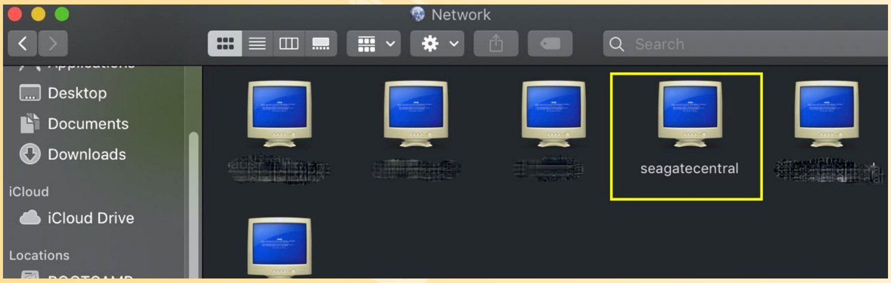
Applying the changes and testing...

When the changes have been made to smb.conf you will need to save them, so simply use Ctrl+X and it will ask you whether you would like to save the changes. Select yes and the changes will be written to the file.

Now, to make these changes effective it becomes necessary to restart the Samba daemon. The Linux command to do this is as follows and entered on the command line:

```
sudo systemctl restart smbd
```

When the command returns, this means Samba has been restarted and you should then see the actual share present on the network on both Windows and macOS (Note it may take a few minutes to appear it all depends on how fast the network is and how quickly the client OS flushes its cache)



What do the options mean?

`read only`

defines if the share is just read only or readable and writable.

`browsable`

defines whether the share can be viewed on Windows machines in My Network Places or not.

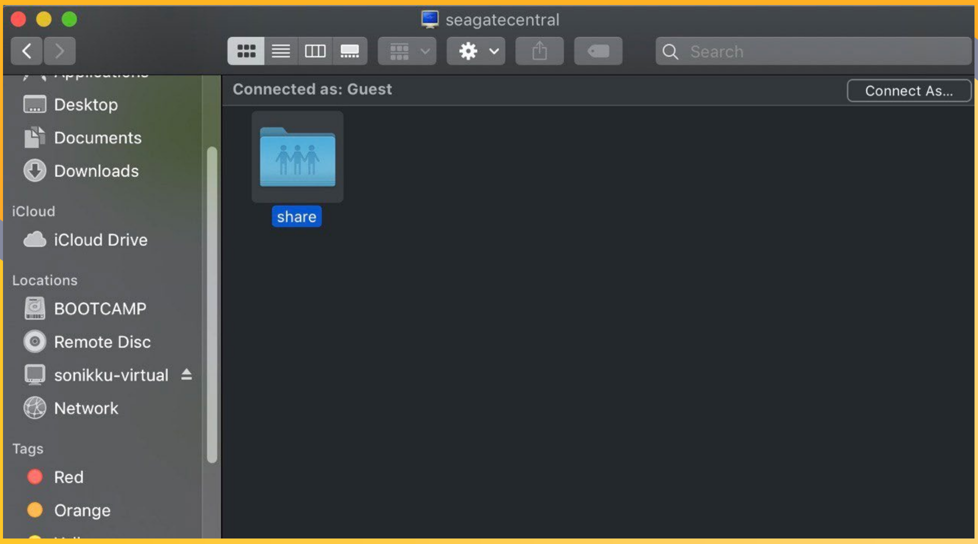
`force group`

This overrides the group policy on the share and directory. By setting it to root we do not need to create users on the system that then require logon credentials on Windows or macOS

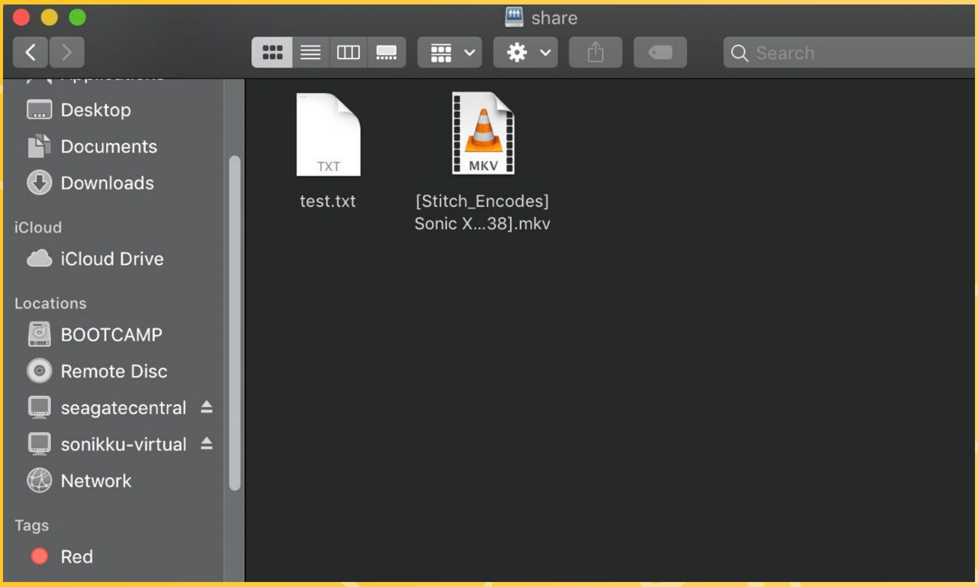
`force user`

Similarly to the above, setting this to root in conjunction with force group = root, effectively grants guests full read/write access to this particular network share. For initial testing this is OK!

Confirming the network share is operational...

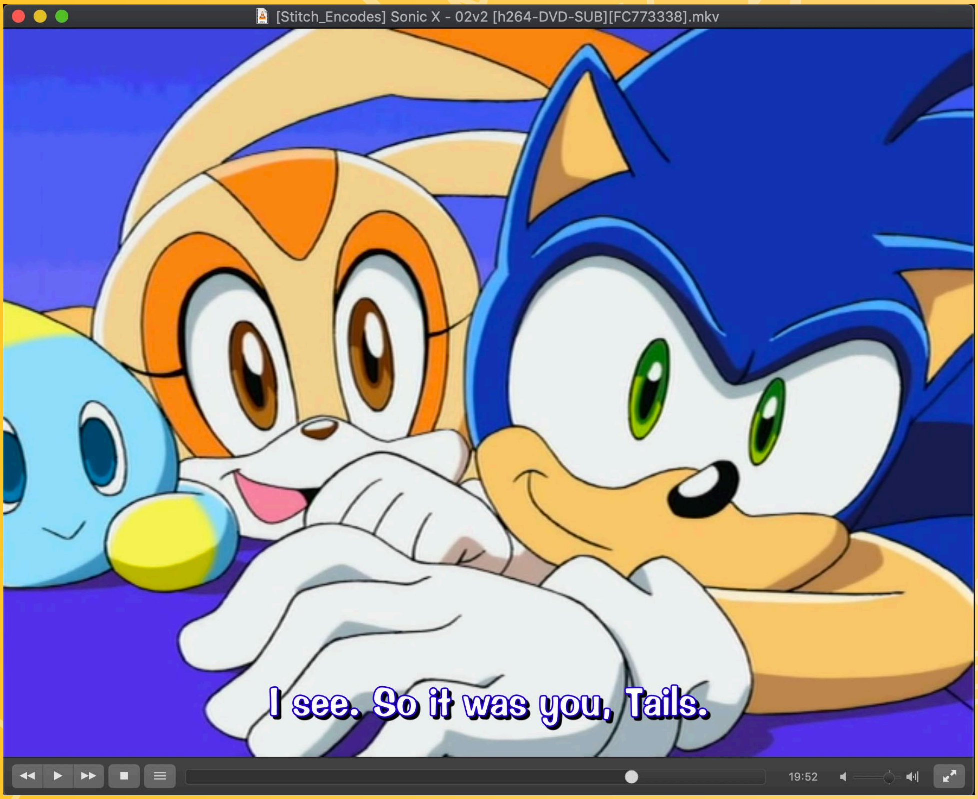


Clicking on the computer labeled `seagatecentral` will bring up a dialog requesting you to enter credentials. Enter `guest` as a username (Windows and macOS) with no password and `/home/seagatecentral/share` appears on the network as a network share.



To test read/write permissions I created (and edited) a simple text file in texedit/Notepad and saved it as `test.txt` - Being able to create, modify and save the file on the network share confirms that Samba is setup correctly and the permissions are OK.

Next, I copied episode 2 of Sonic X to the network share.


[Stitch_Encodes] Sonic X - 02v2 [h264-DVD-SUB][FC773338].mkv
I see. So it was you, Tails.
19:52

Then I used VLC to play the Sonic X episode from the network share. The performance at this point was a bit slow to get going (due to the fact that a relatively slow SD card was being used), however once video playback commenced it worked flawlessly...

GOAL!!

This is the first goal or milestone achieved when constructing a working NAS. If your NAS works as per these examples, congratulations!

You can now proceed to install the OS image on the Orange Pi's eMMC (integrated Multi-Media Card)

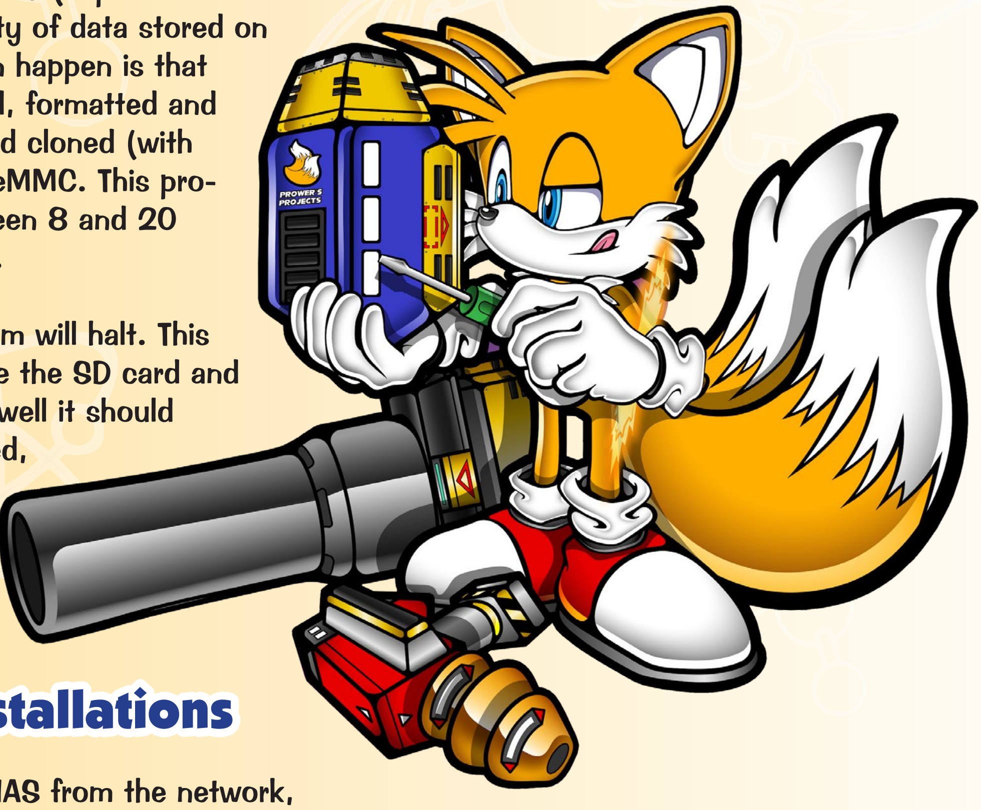
Installation to eMMC

The next phase of construction is to install the image from the SD card to the on-board eMMC. This is accomplished by using the armbian-config utility. This has to be run locally (not via SSH).

DO NOT attempt to run armbian-config via the SSH session. It will not work, and on the slim chance that it does, it will fail during installation leading to data loss!

Under the System menu, is the install menu option. Using this option select the filesystem as F2FS (important to ensure optimal performance and reliability of data stored on the eMMC). The process that will then happen is that generally the eMMC will be partitioned, formatted and then the working image on the SD card cloned (with the correct mount points etc) to the eMMC. This process takes some time, anything between 8 and 20 minutes. Be patient until it completes.

When installation completes the system will halt. This is when you may power down, remove the SD card and power up the board again. If all went well it should boot from the eMMC, and once booted, we again login via SSH and begin to install some cool software...



ART BY SHADOW LIFEMAN

Software Package Installations

Ideally we would like to manage our NAS from the network, the same way the original one was managed with a web interface, and we shouldn't neglect the LED we connected either, so we will begin installing all the software packages required for this. The order of installation is not important, the sequence shown here was how it was done at the time.

The following packages are required (along with necessary kernel headers and libraries) - apt will manage this for you:

- Nginx (pronounced Engine-X)- a lightweight web server
- Python3 (this may already be preinstalled)

To check if Python3 is installed you can run the following command while connected to the NAS via SSH

```
python3
```

The output will be something like

```
Python 3.8.2 (default, Apr 27 2020, 15:53:34)
[GCC 9.3.0] on linux
Type “help”, “copyright”, “credits” or “license” for more information.
```

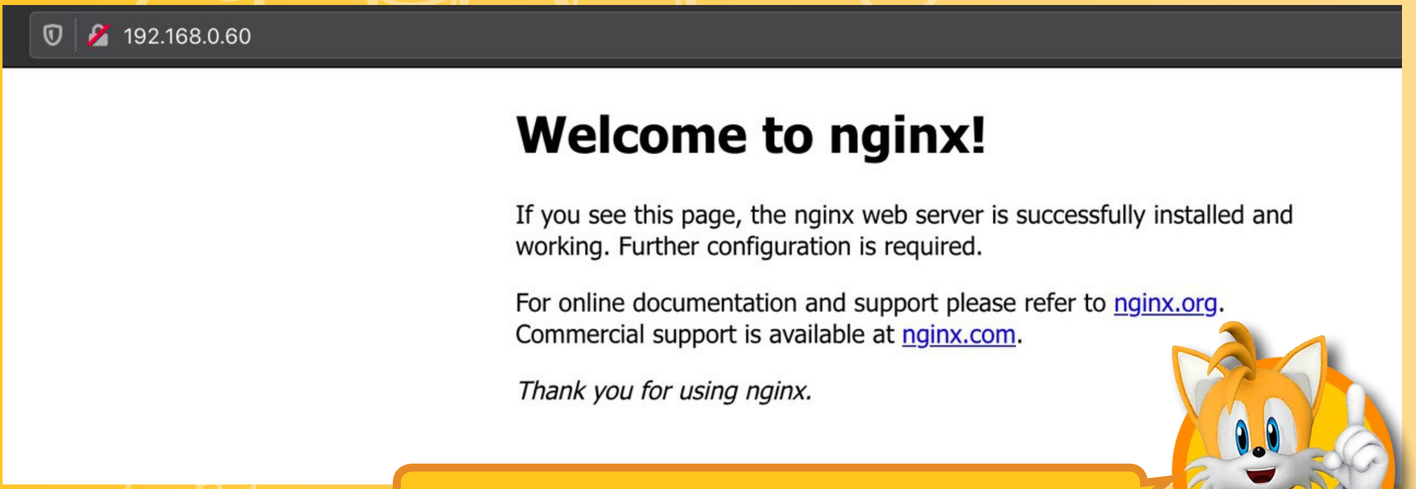
Setting up and Starting nginx...

Installing nginx is as simple as-

```
sudo apt-get install nginx
```

When install completes it will already be running. Therefore the static IP configured for this device will be reachable with a normal web browser

Directly after nginx is installed, you can launch a web browser on your computer, and then access the default web page at the IP address configured for your NAS (I configured it to 192.168.0.60, re-using the allocated IP of the unit that died)



192.168.0.60

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

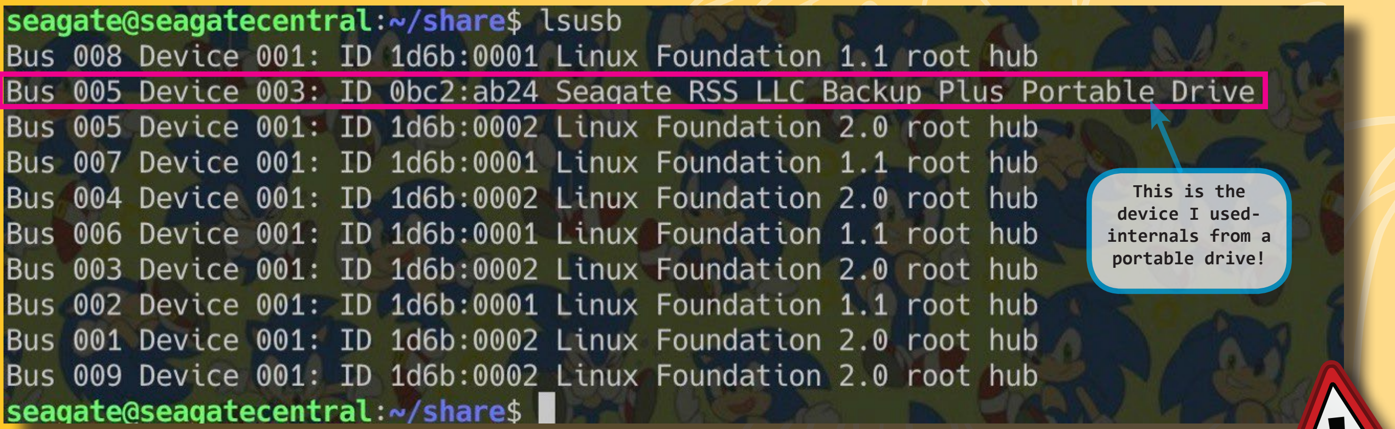
For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Default web page served from the NAS unit after nginx installation

Installing the Hard Disk...

Perhaps the most complex part of this project begins now.
Things can go wrong with this therefore it is important to do a bit of reading online to know how the commands I used operate and their purpose. At this point the hard disk is assumed to be connected to the system as shown in the photos. Now, while logged in via SSH we need to confirm that the USB-to-SATA bridge is operational. We use the `lsusb` command for this.



```
seagate@seagatecentral:~/share$ lsusb
Bus 008 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 005 Device 003: ID 0bc2:ab24 Seagate RSS LLC Backup Plus Portable Drive
Bus 005 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 007 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 004 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 006 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 009 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
seagate@seagatecentral:~/share$
```

This is the device I used- internals from a portable drive!

What does this mean?
This means we've done a scan of the actual hardware plugged in on the USB ports of the OrangePi. That device as shown is clearly the USB-to-SATA bridge that is being detected. The name might be different in your case, including the ID, however this means your hardware is working and you can begin with partitioning and formatting the drive.

Partitioning the Hard Drive...

The next steps are necessary to partition the hard drive.

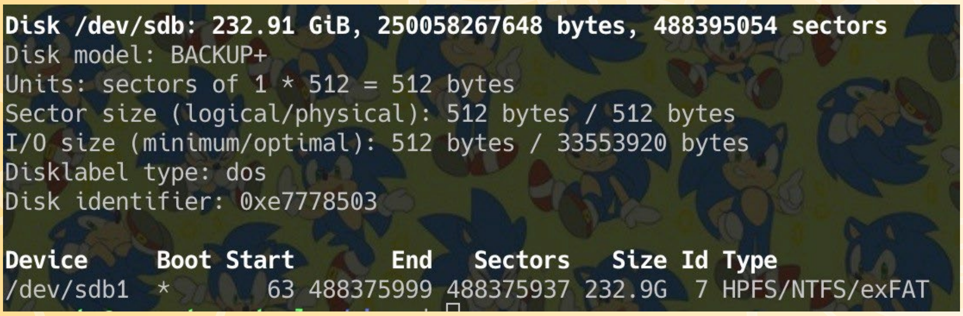
A partition is a space of a predetermined size that can occupy the whole available storage area, or a part of it. Many computer systems including Windows use multiple partitions of varying sizes to create logical drives- often these "additional" partitions will house recovery utilities and the like. Linux, however creates several partitions, each serving a specific purpose and they are not mapped as drives as such. Some Linux partitions are transparent to the user and used as a core part of the operating system.

Now its possible you used a drive salvaged from a dead computer (as I did) or you purchased a new drive. We need to therefore check if there's an existing partition or not. Of course, if the drive was salvaged its likely got some Windows partitions...

To view the partition information of the hard drive, use the following command:

```
sudo fdisk -l
```

This command will produce a list of all the partitions on the system. You will see several, however the one of interest is `/dev/sdb` or similar:



```
Disk /dev/sdb: 232.91 GiB, 250058267648 bytes, 488395054 sectors
Disk model: BACKUP+
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 33553920 bytes
Disklabel type: dos
Disk identifier: 0xe7778503

Device     Boot Start       End   Sectors   Size Id Type
/dev/sdb1  *       63 488375999 488375937 232.9G  7 HPFS/NTFS/exFAT
```

It is quite obvious this drive still has stuff on it, probably Windows 7 as it was salvaged from a dead laptop. Note the following though:

`/dev/sdb` - the device itself aka the hard drive
`/dev/sdb1` - the first partition found- if it exists... Additional partitions will follow as `/dev/sdb2` etc..
(Note these partitions will be absent on a new, blank drive)

Memorize `/dev/sdb` as the drive of interest as we will need this information for the partitioning step.

If you have existing data...

If there are existing partitions on the drive as per the above example, they will need to be removed i.e. deleted. WARNING- This is a destructive operation so ensure that anything important on the drive is backed up! With a new or blank drive, skip the partition deletion steps.

Linux Commands?
Here's a handy-dandy short form guide:

`lsusb`

This command lists all the devices currently enumerated on all the USB buses.

`fdisk -l`

Linux utility for creation and manipulation of partition tables on media (media is any medium that can hold the operating system of the computer). The -l switch tells the program to "list" the partitions system wide i.e. ALL the storage devices. It is the traditional tool used for this purpose but it is not recommended for use on drives larger than 2TB as it uses MBR (Master Boot Record).

`parted`

Linux utility known formally as GNU Parted for creation and manipulation of partition tables similarly to fdisk but is my first choice. However it is recommended to consistently use parted particularly if the target drive is 2TB or greater as you would be using GPT (GUID partitioning tables). Its also easier in my opinion, to use.

`mkfs.ext4 <device>`

This Linux command is used to format partitions on the specified device according to the ext4 filesystem type.

Commands used within parted

`rm`

rm = remove. Similarly to the Linux filesystem command, 'rm' that is used for deleting files, this command is used in parted for deleting partitions. It will prompt for the number of the partition you wish to delete.

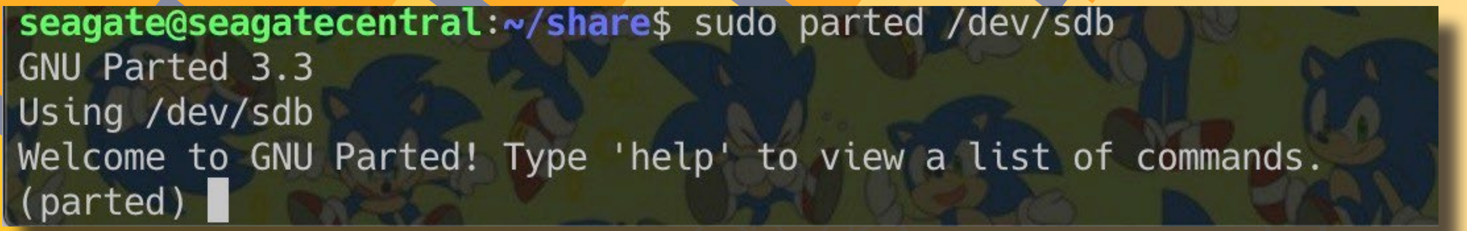
`mklabel <option>`

This command is used to set the partition type ahead of creation.

`mkpart <type> <start> <end>`

This command creates the partition according to the parameters given

Launching GNU Parted...



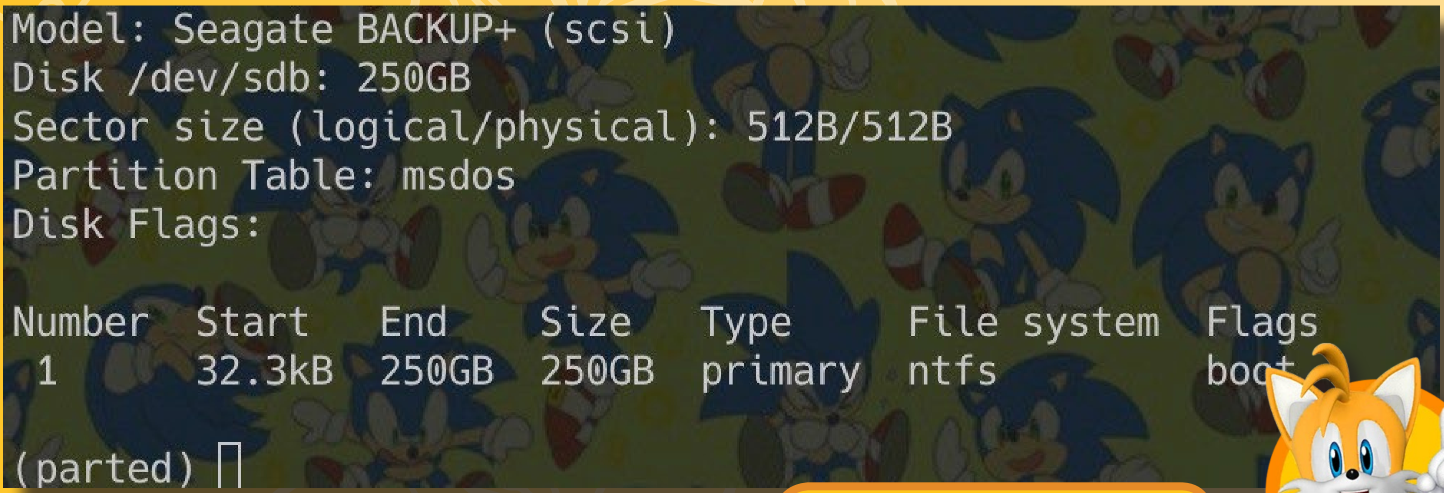
```
seagate@seagatecentral:~/share$ sudo parted /dev/sdb
GNU Parted 3.3
Using /dev/sdb
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) █
```

The cool thing about launching parted is that it has a command prompt similar to fdisk, but its just far more user friendly. The prompt appears at all times as (parted) with a block cursor. Notice the command syntax. That's why we had to remember the drive's detail as being `/dev/sdb` as this is given as an argument to parted.

Where do we go from here?
A. Hard disks with existing partitions

Hard drives that have existing partitions are best repartitioned. The reasons are very technical and long-winded but its always better to delete existing partitions and create new ones.

Delete existing partitions by first identifying them. To do this we enter the `print` command. This drive, fortunately had only one partition on it (the reason- it was a secondary drive in the dead laptop I salvaged it from).
NB: If no partitions are displayed, proceed to 'B'



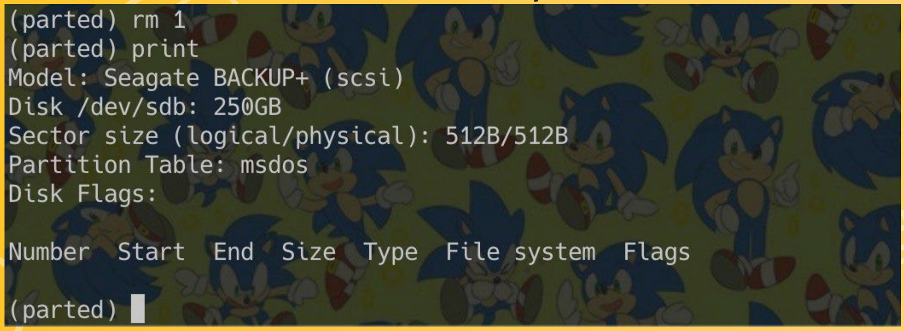
```
Model: Seagate BACKUP+ (scsi)
Disk /dev/sdb: 250GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start   End    Size   Type     File system  Flags
 1      32.3kB  250GB  250GB  primary  ntfs         boot

(parted) ▯
```

There it is... the single
NTFS (Windows) partition

Note the number of this partition, in this case it is the first one found and is numbered '1'. Also note that it spans the entire physical media (250GB drive).
We use the following commands to delete that partition and confirm it has been removed

```
rm 1
print
```

As can be seen from this screenshot, the partition has been removed completely...



```
(parted) rm 1
(parted) print
Model: Seagate BACKUP+ (scsi)
Disk /dev/sdb: 250GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start  End  Size  Type  File system  Flags

(parted) █
```

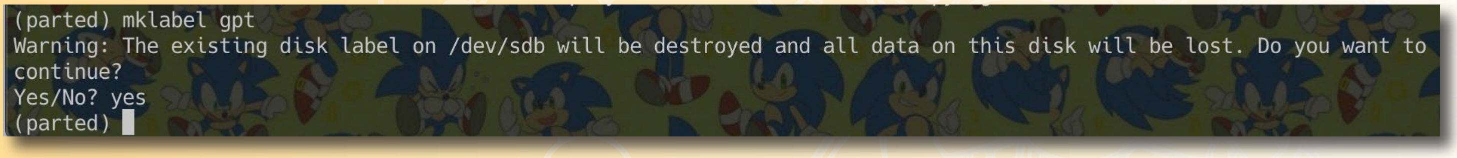
For each partition found, you will need to issue a `rm <number>` command to remove it.

Some drives may have many partitions so use the `rm <number>` command for each one until print reveals no more partitions exist!

B. Hard disks with no existing partitions i.e. new or sanitised drives

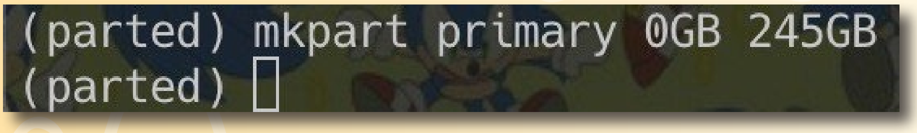
At this point we have a completely blank hard drive. We need to create a partition to hold the filesystem we wish to use to store (and serve) our files over the network.

Note- I am using the entire drive here with a single partition, reserving a small bit of space for overhead. The drive size is 250Gbytes, I will create a partition of 245GB, leaving approximately 3-4GB free to create either a recovery partition if I want to, but in the main I do this for performance reasons.



```
(parted) mklabel gpt
Warning: The existing disk label on /dev/sdb will be destroyed and all data on this disk will be lost. Do you want to
continue?
Yes/No? yes
(parted) █
```

In the above screen cap, I used the mklabel command with gpt as a parameter. This means a new partition will be created using GPT (GUID Partition Table). Then I proceed to create the partition using the mkpart command. As I mentioned I want to use only 245GB of this drive, as this screen cap shows by example:



```
(parted) mkpart primary 0GB 245GB
(parted) ▯
```

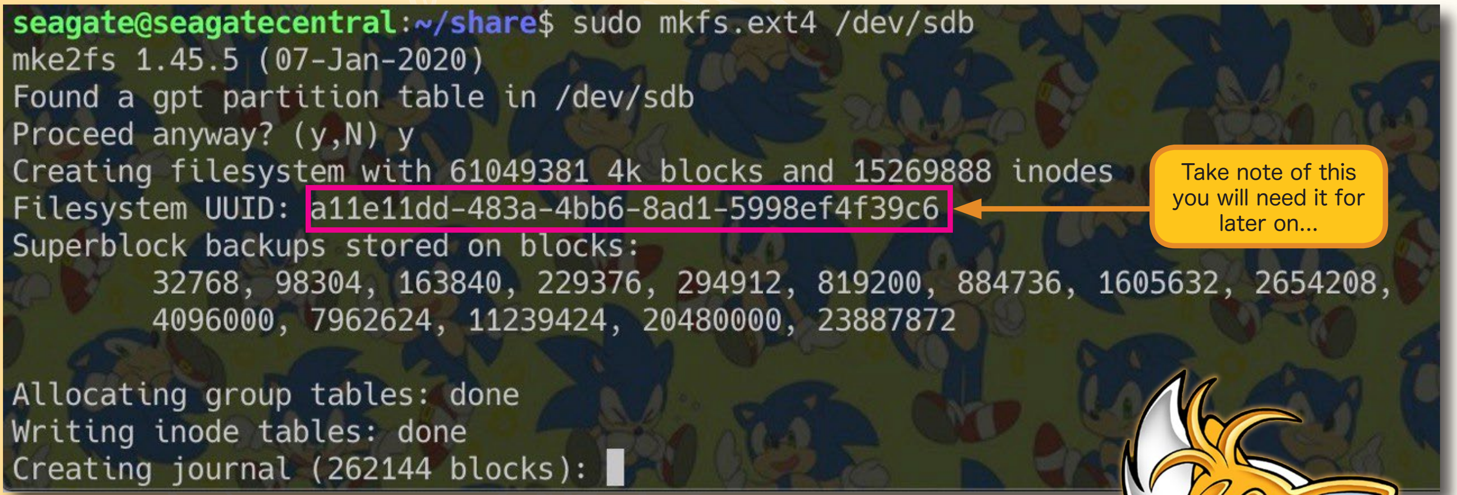
That's the only step required further in parted for blank or new drives. (or any drive where you removed the partitions beforehand). When this step is complete you may exit parted using the quit command.

Formatting the Partition

To make our partition useful, we will need to format it. Now many folks may be familiar with this from their experiences with Microsoft Windows. However we do not use the regular formats (NTFS/FAT32) because not only are they're proprietary, they're awful performers on Linux as I've learnt the hard way.

To ensure top notch performance and the ability to fix things when they go wrong, we will use a default (and widely supported) Linux filesystem known as ext4. (Note that this means the drive will not be readable or writable in macOS* or Windows should you later need to perform data recovery due to hardware failure or other reasons- just bear that in mind.

We format the drive using the standard Linux command mkfs.ext4. As expected this requires privilege escalation so don't forget to pre-pend sudo and enter your user password. Once the command is launched, grab a coffee, and sit back and relax as dpending on the drive speed and size, this could take several minutes, or hours.



```
seagate@seagatecentral:~/share$ sudo mkfs.ext4 /dev/sdb
mke2fs 1.45.5 (07-Jan-2020)
Found a gpt partition table in /dev/sdb
Proceed anyway? (y,N) y
Creating filesystem with 61049381 4k blocks and 15269888 inodes
Filesystem UUID: a11e11dd-483a-4bb6-8ad1-5998ef4f39c6
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
        4096000, 7962624, 11239424, 20480000, 23887872

Allocating group tables: done
Writing inode tables: done
Creating journal (262144 blocks): █
```

Take note of this you will need it for later on...

*An application known as extFS is available for macOS that enables your Macintosh to read and write ext2/3/4 formatted drives - https://www.paragon-software.com/home/extfs-mac/#

Getting the Hard Drive Online...

When the format operation eventually completes, its time to do two things:
1. Create a mount point, and
2. Mount the drive and access it.

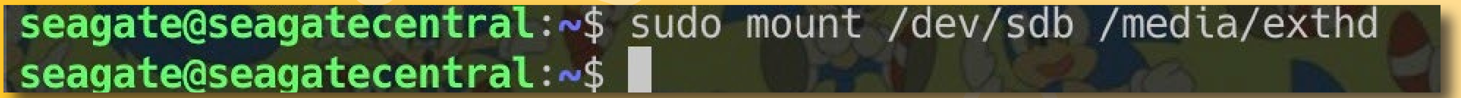
The mount point is a location where the drive's filesystem will appear on the host operating system.
You can choose your own name if you so wish, but my naming convention is used here so that I don't forget years later when I have to change things-

1. The mount point is created with the `mkdir` command with `sudo`:

```
sudo mkdir /media/exthd
```

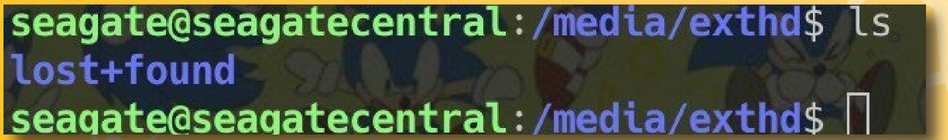
2. The drive is then mounted using the mount command

```
sudo mount /dev/sdb /media/exthd
```



```
seagate@seagatecentral:~$ sudo mount /dev/sdb /media/exthd
seagate@seagatecentral:~$ 
```

There was no error returned when the device was mounted, therefore it was most likely successful. Moreover, when we change to the directory we created for the mount point, we notice we can actually access the drive. The presence of the system reserved directory "lost+found" means our hard disk is now visible in the operating system as any other drive...



```
seagate@seagatecentral:/media/exthd$ ls
lost+found
seagate@seagatecentral:/media/exthd$ 
```

GOAL!!

The hard disk was correctly partitioned, formatted and then became usable when mounted on the OS

Setting up automatic mounting

This NAS would be quite silly if we had to mount the hard disk every time we turned the power off and back on again. The next steps involve:

1. Ensuring the permissions are set correctly for Samba to use this drive

2. Editing `/etc/fstab` so that the hard disk is auto-mounted at every system startup (boot).

First Step - Permissions

We need to ascertain the permissions of `/media/exthd`
This is done with the Linux command

```
ls -ld
```

To correct the permissions of this external hard drive (if we leave it as-is, its root directory will be not accessible by Samba) we need to use the Linux `chown` command to set the permissions.
The permissions returned from the above command should be:

```
drwxr-xr-x 12 root root
```

They will likely be something else, possibly the user account "seagate". To correct this we need to first assume super user or log in as root-
The easiest way is to use the command `sudo su` and enter your password. You will notice the prompt change to `root@seagatecentral`

Now we can change the ownership of `/media/exthd` by using the Linux `chown` command.

```
chown -R root:root /media/exthd
```

This changes the ownership of the hard disk mounted at /media/exthd to the root user AND the root group.

Second Step - Automatic mounting at startup

On Linux and Unix-like systems, the filesystems are configured by use of the fstab (fstab is short for file systems table). Now its quite apparent why we need to add an entry- in order that the hard drive we partitioned and formatted, is mounted automatically at startup and can be used immediately when the system boots up.

At this point we're still elevated to root via a previous `sudo su` command, therefore we can simply edit the fstab using (without sudo-ing):

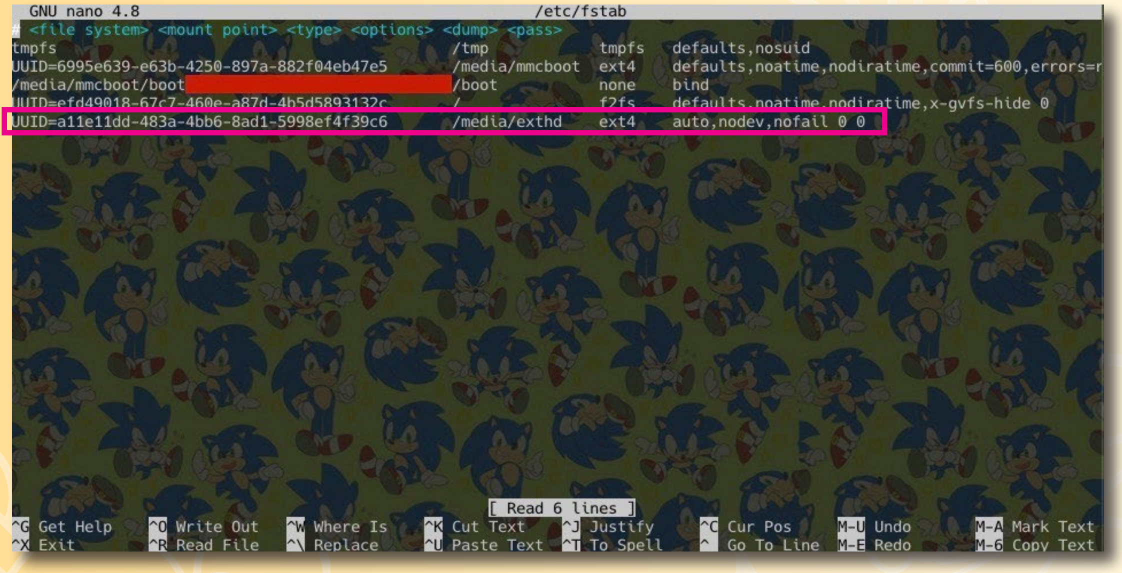
```
nano /etc/fstab
```

This opens up the fstab for configuration

At the bottom of the file, we add a new line, containing the UUID we had to take note of whilst formatting the drive- it's quite apparent we're mapping it to `/media/exthd` the filesystem is ext4, and the parameters are specified as `auto,nodev,nofail 0 0`

Therefore the new entry (as a new line you need to add) is as follows:

```
UUID=<your_UUID_here>       /media/exthd     ext4       auto,nodev,nofail 0 0
```



```
  GNU nano 4.8                              /etc/fstab
# <file system> <mount point> <type> <options> <dump> <pass>
tmpfs                                    /tmp            tmpfs    defaults,nosuid
UUID=6995e639-e63b-4250-897a-882f04eb47e5 /media/mmcboot  ext4     defaults,noatime,nodiratime,commit=600,errors=r
/media/mmcboot/boot[redacted]            /boot           none     bind
UUID=efd49018-67c7-460e-a87d-4b5d5893132c /              f2fs     defaults,noatime,nodiratime,x-gvfs-hide 0
UUID=a11e11dd-483a-4bb6-8ad1-5998ef4f39c6 /media/exthd    ext4     auto,nodev,nofail 0 0

                                   [ Read 6 lines ]
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos     M-U Undo       M-A Mark Text
^X Exit        ^R Read File   ^\ Replace     ^U Paste Text  ^T To Spell    ^  Go To Line  M-E Redo       M-6 Copy Text
```

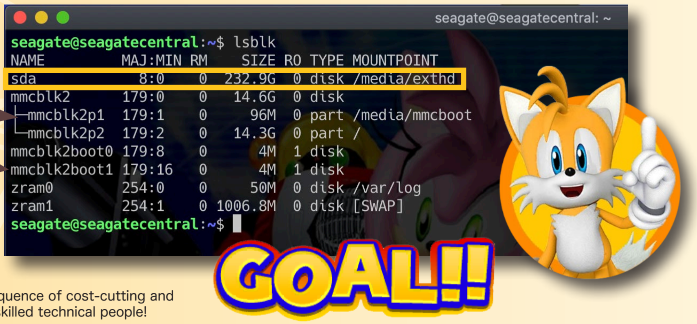
Save fstab and reboot

After you have added that line to fstab, you can save the changes (Ctrl-X and then Y to indicate "yes" to save). Then reboot the system either using a reboot command or by unplugging the power and reconnecting the power.

Once the system has rebooted, the hard drive should then be auto-mounted and you can confirm this by SSHing back into the system and using the command:

```
lsblk
```

You should clearly see the device mounted as `sda` and pointing exactly to where we want it i.e. `/media/exthd`



```
seagate@seagatecentral:~$ lsblk
NAME         MAJ:MIN RM    SIZE RO TYPE MOUNTPOINT
sda            8:0    0  232.9G  0 disk /media/exthd
mmcblk2      179:0    0   14.6G  0 disk
├─mmcblk2p1  179:1    0     96M  0 part /media/mmcboot
└─mmcblk2p2  179:2    0   14.3G  0 part /
mmcblk2boot0 179:8    0      4M  1 disk
mmcblk2boot1 179:16   0      4M  1 disk
zram0        254:0    0     50M  0 disk /var/log
zram1        254:1    0 1006.8M  0 disk [SWAP]
seagate@seagatecentral:~$ 
```

These items are the two partitions of the eMMC and this is where your Armbian installation resides

What do those fstab parameters mean exactly?

`auto`

Automatically detect the filesystem. This is recommended for removable drives (technically our hard disk is a removable drive attached on the USB port).

`nodev`

Very important- this parameter specifies the hard drive cannot be mounted as a special device that could expose system access at the super user level without any password or group permission.

`nofail`

Allows the system to continue booting even if the media has failed or is not present. This was why the original Seagate Central is a stupid design, period!*

Parameters `0 0`

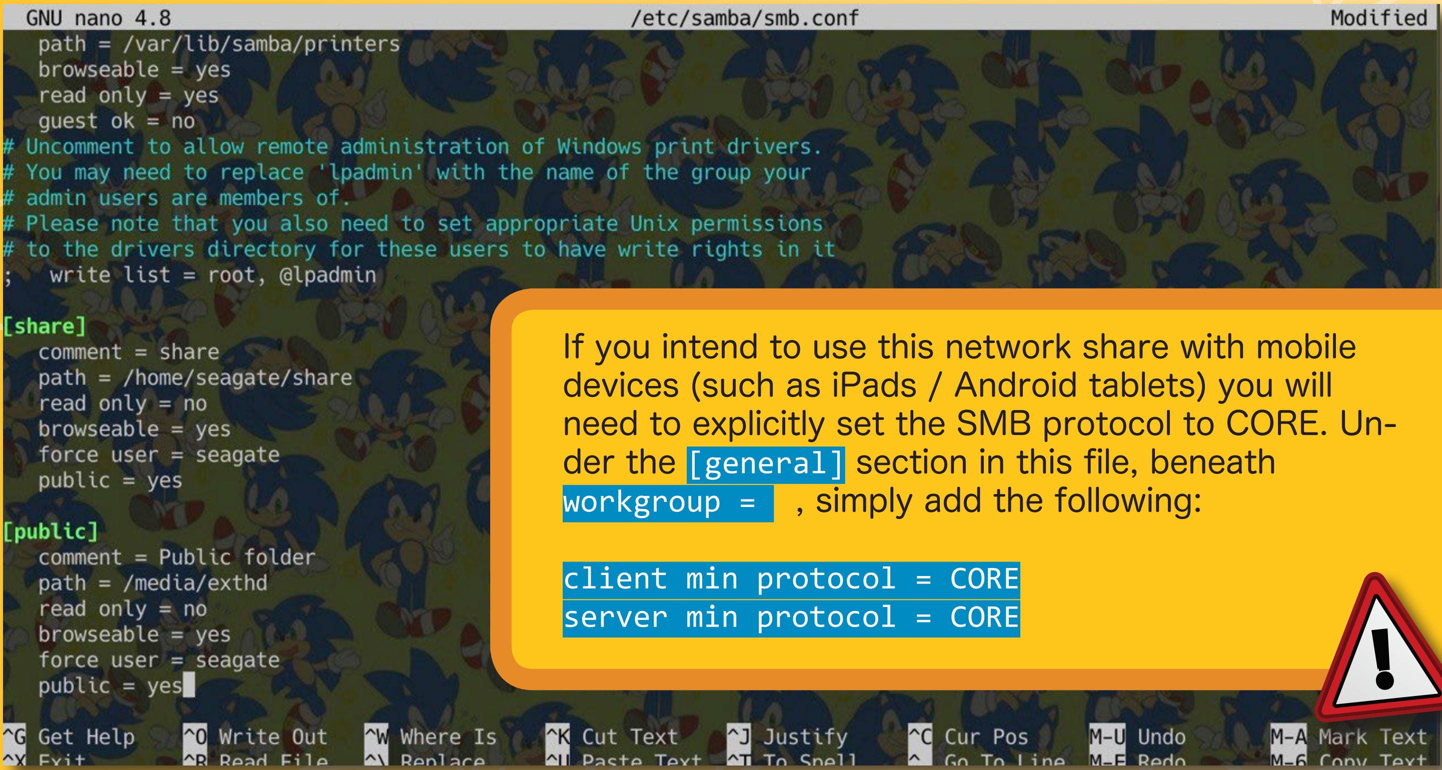
These have special purposes beyond the scope of this article. The defaults of 0 0 should be used. Suffice to say, the second zero simply instructs the filesystem checker (fsck) to skip this device if not present.

*Crappy designs such as the Seagate Central are usually the consequence of cost-cutting and corporate greed i.e. companies run by accountants as opposed to skilled technical people!

Mapping the Hard Drive as a Network Share

At this point we've made serious progress. We now need to make another modification to Samba in order to map this hard drive we installed, to the network. To recap- previously we mounted a directory on the eMMC to the network, which is useful to keep but definitely not where you should store anything unless its very small files. I recommend you use that directory for copying firmware and/or web server updates during development and nothing else.

The following screenshot shows how I edited smb.conf. Notice however the `force user` has been changed to "`seagate`". This is for security reasons and to allow us later to implement access controls. Because the "`seagate`" user has been added to the sudoers group, it will still work like before (may have to refresh your view of the network in Network Places (Windows) / Finder (macOS). Note the additional setting: `public = yes`

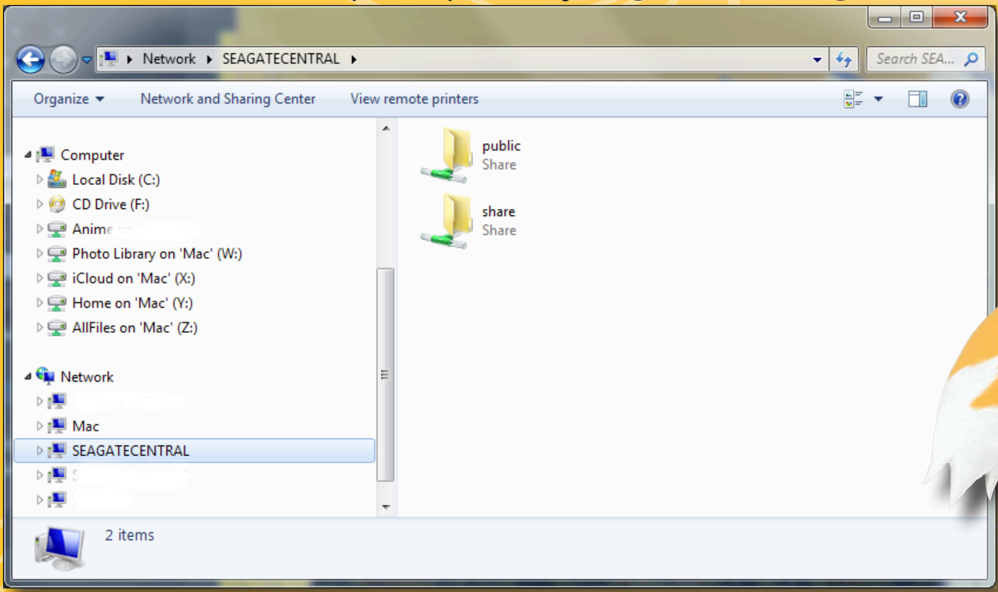


```
  GNU nano 4.8                    /etc/samba/smb.conf                     Modified
   path = /var/lib/samba/printers
   browseable = yes
   read only = yes
   guest ok = no
# Uncomment to allow remote administration of Windows print drivers.
# You may need to replace 'lpadmin' with the name of the group your
# admin users are members of.
# Please note that you also need to set appropriate Unix permissions
# to the drivers directory for these users to have write rights in it
;   write list = root, @lpadmin

[share]
   comment = share
   path = /home/seagate/share
   read only = no
   browseable = yes
   force user = seagate
   public = yes

[public]
   comment = Public folder
   path = /media/exthd
   read only = no
   browseable = yes
   force user = seagate
   public = yes

^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos     M-U Undo       M-A Mark Text
^X Exit        ^R Read File   ^\ Replace     ^U Paste Text  ^T To Spell    ^_ Go To Line  M-E Redo       M-6 Copy Text
```

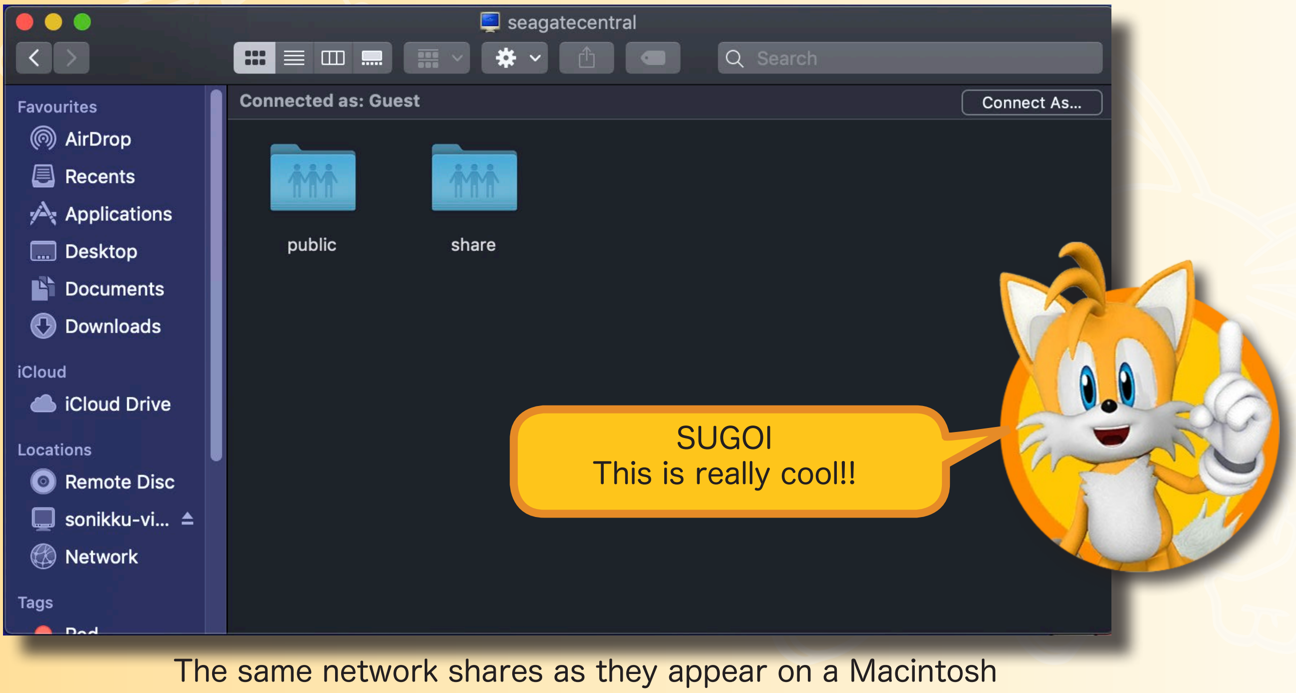
If you intend to use this network share with mobile devices (such as iPads / Android tablets) you will need to explicitly set the SMB protocol to CORE. Under the `[general]` section in this file, beneath `workgroup =` , simply add the following:

```
client min protocol = CORE
server min protocol = CORE
```

From the above example it is clear that `[share]` and `[public]` are the two expected network shares. Save the file and restart Samba as explained previously using the command given earlier (no need to reboot the system).



With Windows 7 this is what we see...



The same network shares as they appear on a Macintosh

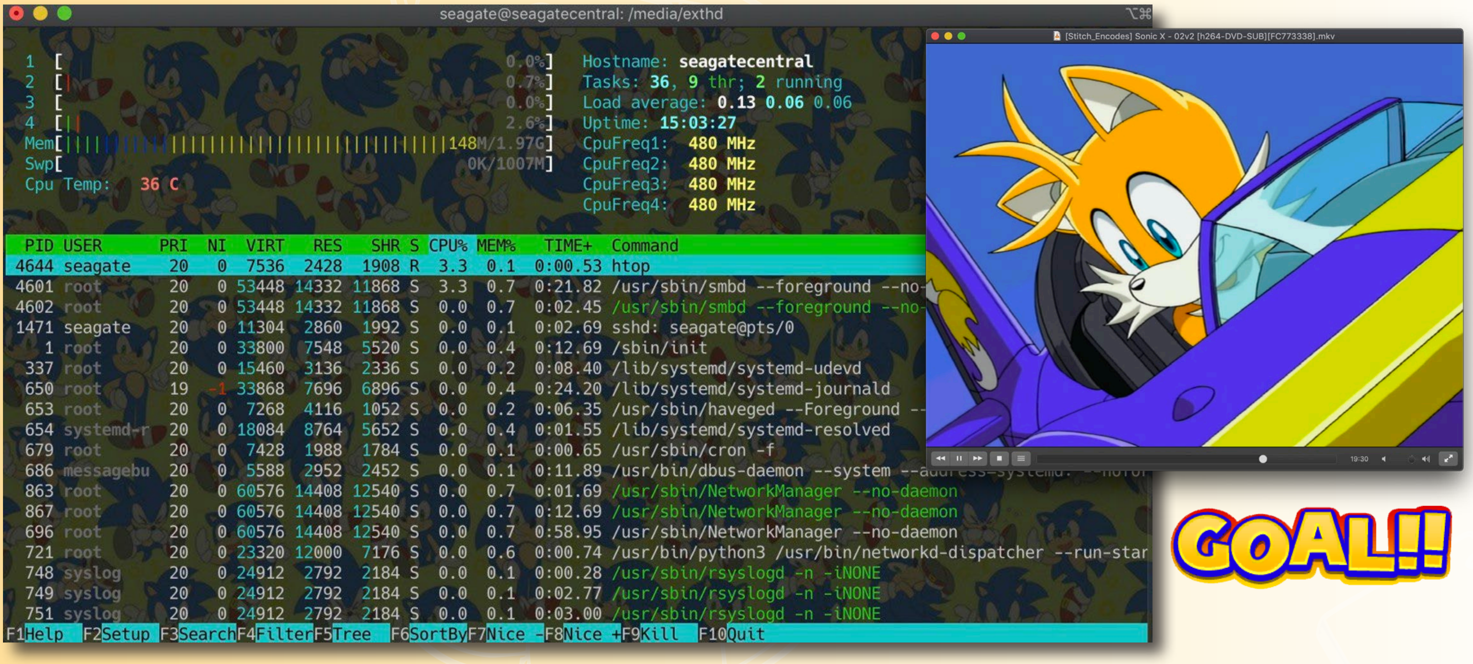
At this point we now have a real, working NAS drive. And we can write files to it and read them. macOS might be a bit iffy but it plays really nicely with Microsoft Windows. Your NAS can be rebooted and it will come back and the shares will be visible. That brings me to the next point...

A word about Security...

It is quite clear we didn't think about securing this device at all... yet!
The reason for this is that security would add another layer of complexity- it would be extremely frustrating to try and figure out what is wrong when it doesn't work like it should and the underlying cause is due to problematic security policies. Linux is widely known for being secure, and adding security concepts such as group and user permissions with authentication could frustrate our attempts at development. Its better get it working, then make it secure. Moreover we're doing active development on the system, so as long as it sits behind a firewall it is reasonably safe from intrusion, unless of course your firewall sucks! That's definitely a debate for another day and beyond the scope of what we're trying to do.

And Performance?

Running the Linux utility `htop` via SSH, while streaming an episode of Sonic X, revealed that the system isn't even breaking out a sweat, even the CPU cores were throttled back to 480MHz, the base rate


seagate@seagatecentral: /media/exthd
[Stitch_Encodes] Sonic X - 02v2 [h264-DVD-SUB][FC773338].mkv
19:30
GOAL!!

```
  1  [                                    0.0%]   Hostname: seagatecentral
  2  [|                                   0.7%]   Tasks: 36, 9 thr; 2 running
  3  [                                    0.0%]   Load average: 0.13 0.06 0.06
  4  [||                                  2.6%]   Uptime: 15:03:27
  Mem[||||||||||||||||||||||||||||||||148M/1.97G]   CpuFreq1:  480 MHz
  Swp[                                 0K/1007M]   CpuFreq2:  480 MHz
  Cpu Temp:    36 C                                CpuFreq3:  480 MHz
                                                   CpuFreq4:  480 MHz

  PID USER       PRI  NI  VIRT   RES   SHR S CPU% MEM%   TIME+  Command
 4644 seagate     20   0  7536  2428  1908 R  3.3  0.1  0:00.53 htop
 4601 root        20   0 53448 14332 11868 S  3.3  0.7  0:21.82 /usr/sbin/smbd --foreground --no-
 4602 root        20   0 53448 14332 11868 S  0.0  0.7  0:02.45 /usr/sbin/smbd --foreground --no-
 1471 seagate     20   0 11304  2860  1992 S  0.0  0.1  0:02.69 sshd: seagate@pts/0
    1 root        20   0 33800  7548  5520 S  0.0  0.4  0:12.69 /sbin/init
  337 root        20   0 15460  3136  2336 S  0.0  0.2  0:08.40 /lib/systemd/systemd-udevd
  650 root        19  -1 33868  7696  6896 S  0.0  0.4  0:24.20 /lib/systemd/systemd-journald
  653 root        20   0  7268  4116  1052 S  0.0  0.2  0:06.35 /usr/sbin/haveged --Foreground --
  654 systemd-r   20   0 18084  8764  5652 S  0.0  0.4  0:01.55 /lib/systemd/systemd-resolved
  679 root        20   0  7428  1988  1784 S  0.0  0.1  0:00.65 /usr/sbin/cron -f
  686 messagebu   20   0  5588  2952  2452 S  0.0  0.1  0:11.89 /usr/bin/dbus-daemon --system --address=systemd: --nofor
  863 root        20   0 60576 14408 12540 S  0.0  0.7  0:01.69 /usr/sbin/NetworkManager --no-daemon
  867 root        20   0 60576 14408 12540 S  0.0  0.7  0:12.69 /usr/sbin/NetworkManager --no-daemon
  696 root        20   0 60576 14408 12540 S  0.0  0.7  0:58.95 /usr/sbin/NetworkManager --no-daemon
  721 root        20   0 23320 12000  7176 S  0.0  0.6  0:00.74 /usr/bin/python3 /usr/bin/networkd-dispatcher --run-star
  748 syslog      20   0 24912  2792  2184 S  0.0  0.1  0:00.28 /usr/sbin/rsyslogd -n -iNONE
  749 syslog      20   0 24912  2792  2184 S  0.0  0.1  0:02.77 /usr/sbin/rsyslogd -n -iNONE
  751 syslog      20   0 24912  2792  2184 S  0.0  0.1  0:03.00 /usr/sbin/rsyslogd -n -iNONE
F1Help  F2Setup F3SearchF4FilterF5Tree  F6SortByF7Nice -F8Nice +F9Kill  F10Quit
```

Part II - Software Dev

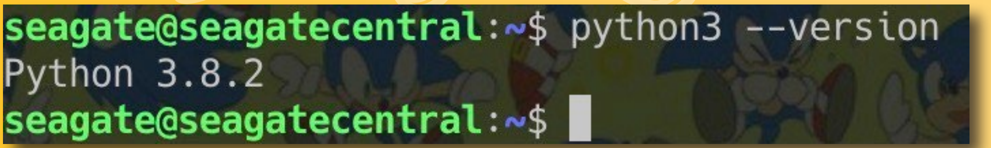
Objectives and Goals

Wow we've made it Sonic!
We have a working device. There are two things I want to do next and they're nice things-

a) Control of the LED indicator
b) A functional web interface similar or more advanced to what the Seagate Central used to have

A. LED Control with Python

We begin development by logging into the NAS via SSH as we have done previously. A quick check is performed on the Python version:

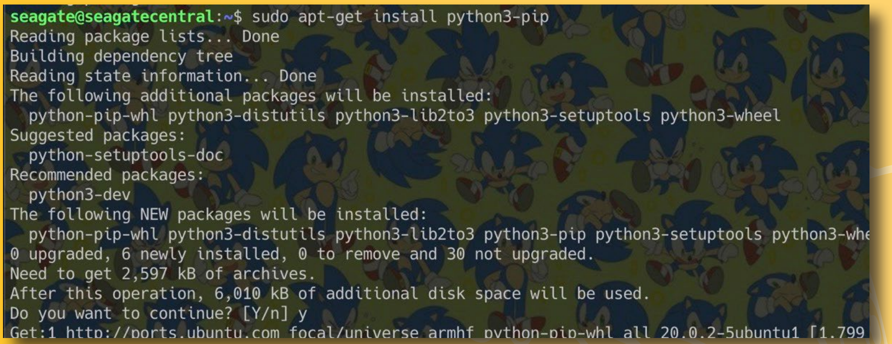


```
seagate@seagatecentral:~$ python3 --version
Python 3.8.2
seagate@seagatecentral:~$ 
```

Next we install a component to complement Python called pip This is a utility to install and manage Python packages. We do this on the command line with:

```
sudo apt-get install python3-pip
```

This command will invoke the installation of all the components necessary for pip to work, most importantly a component called python3-wheel. Your NAS will need to be connected to the internet (or have a route to the internet) to get the necessary...

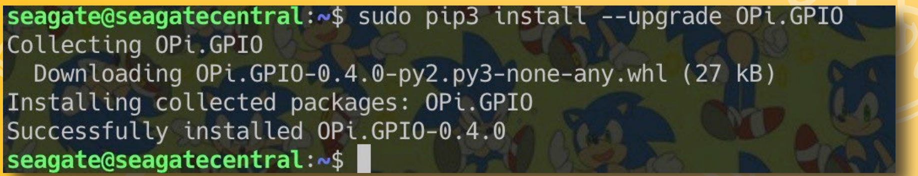


```
seagate@seagatecentral:~$ sudo apt-get install python3-pip
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  python-pip-whl python3-distutils python3-lib2to3 python3-setuptools python3-wheel
Suggested packages:
  python-setuptools-doc
Recommended packages:
  python3-dev
The following NEW packages will be installed:
  python-pip-whl python3-distutils python3-lib2to3 python3-pip python3-setuptools python3-whe
0 upgraded, 6 newly installed, 0 to remove and 30 not upgraded.
Need to get 2,597 kB of archives.
After this operation, 6,010 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ports.ubuntu.com focal/universe armhf python-pip-whl all 20.0.2-5ubuntu1 [1,799
```

With pip installed we can install the package (or library) to control the LED via GPIO on the OrangePi + 2E. We run the command:

```
sudo pip3 install --upgrade Opi.GPIO
```

It should run, and then indicate successful installation. If it doesn't there's a problem so please check the version and/or if there is a successor package (I did this in 2020, things could have changed in the meantime)



```
seagate@seagatecentral:~$ sudo pip3 install --upgrade OPi.GPIO
Collecting OPi.GPIO
  Downloading OPi.GPIO-0.4.0-py2.py3-none-any.whl (27 kB)
Installing collected packages: OPi.GPIO
Successfully installed OPi.GPIO-0.4.0
seagate@seagatecentral:~$ 
```

What is pip?

pip is a package-management system written in Python used to install and manage software packages. It connects to an online repository of public packages, called the Python Package Index

Grab the Source Code

The good thing is, it isn't the 80s anymore so we would never expect you to re-type source code from a page in a book. There is a much better way and its far more modern. All the projects in this book will use this same method of sharing a source code repository- yes you've probably guessed by now we all use GitHub!

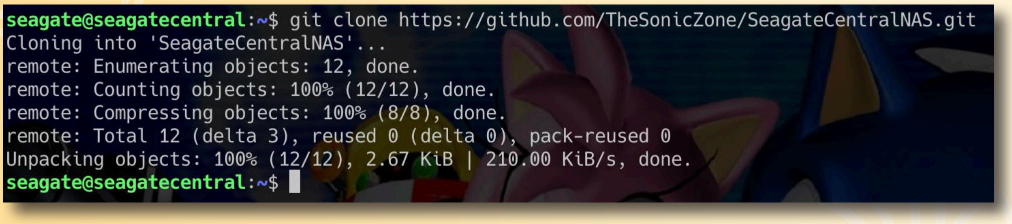
To grab the current LED control application, simply perform a `git clone` from our source repository:

```
https://github.com/TheSonicZone/SeagateCentralNAS.git
```

Ideally you will perform the clone to your home directory, and a folder will be created. Within the folder you will see the source code containing some files, most importantly ledcontrol.py

So, inside your home directory /home/seagatecentral , execute a git clone using the command:

```
git clone https://github.com/TheSonicZone/SeagateCentralNAS.git
```



```
seagate@seagatecentral:~$ git clone https://github.com/TheSonicZone/SeagateCentralNAS.git
Cloning into 'SeagateCentralNAS'...
remote: Enumerating objects: 12, done.
remote: Counting objects: 100% (12/12), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 12 (delta 3), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (12/12), 2.67 KiB | 210.00 KiB/s, done.
seagate@seagatecentral:~$ 
```

When this operation completes, you will notice that your home directory now contains a folder called SeagateCentralNAS, and inside that directory reside the source code files for the LED control part of this project.

The LED control is implemented as a daemon (service) which means later we will start it with the system at boot time. More about that later. First its important to check functionality.

In the directory you will have ledtest.py. The current version is a "snapshot" which means it was a version just prior to me adding networking and other features.

What is Github

Github is a provider of hosting for software development and version control using Git, in the cloud and allowing anyone across the world to collaborate on software projects.

Hardware Test- LED working?

To check that the LED is working correctly, you can run `ledtest.py` on the command line. Before we do this, however, note that the LED may be amber already, this is especially true from the moment the GPIO is activiated in armbian-config. If this is NOT the case, you need to either re-check armbian-config, or check your connections to the LED. But in any case, run the test program first as that will conclusively prove it either way:
On the command line, enter:

```
sudo python3 ledtest.py
```

Note the importance of privilege escalation using `sudo`
This is needed because as with most modern OS, direct access to I/O devices is privileged. If you don't use sudo, the program will emit `permission denied` errors whilst running and do absolutely nothing.

The program runs, and if you wired the LED correctly, it will flash on and off, either green or red.
If it flashes red, simply swap the two wires attached to PA12 and PA11 (or you can swap them in the source code).

Simply enter Ctrl+C to stop the program from running and the LED will either be on or off (depending where the program stopped).

Success! You have gained control of the LED via GPIO

GOAL!!

The ledcontrol.py Daemon (service)

Okay Sonic, time to explain how it works.

The daemon (system service in Linux) is a process that runs in the background, like a device driver, except it is usually started with the system and terminated with the system shutting down.

The idea is to implement a daemon that listens on the network interface of this NAS, and allows the LED to be controlled by commands embedded in the URL. In this way, we can control the LED from code within web pages, or indeed by other services (and applications).

This also means, of course, that you can control the LED directly by using the IP address of the NAS, and the appropriate command. Some would say this is bad, well, the original product was easy to get at via SSH before it died, so there's no real point in expending a lot of security on a simple LED, if someone's gonna mess with it, it will be fixed again by the controlling application!

Think of it as a website- when you visit it with your browser, although a blank page is displayed, stuff happens to the LED.

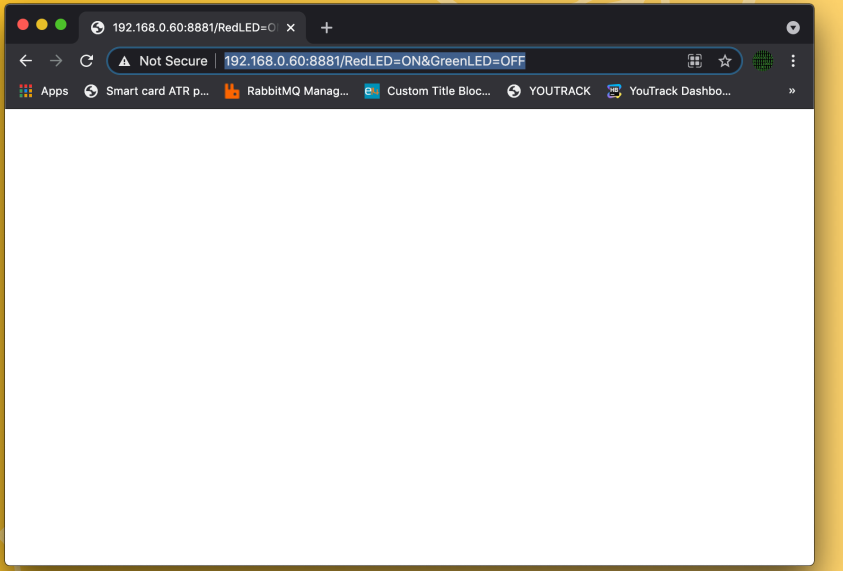
Therefore, in the URL bar of the browser we enter a command like this:

```
http://192.168.0.60:8881/RedLED=ON&GreenLED=ON
```

And the LED turns orange because both RED and GREEN are on.

Think of it as a website- when you visit it with your browser, although a blank page is displayed, stuff happens to the LED.

The example shown here features the Google Chrome browser. Notice the page indicator says "Not Secure", well that's because we did a normal HTTP request and nothing we're doing here is encrypted. Nor does it have to be for an internal application like this whose sole purpose in life is to control an LED. As is obvious the example URL here turns the LED solid red....



192.168.0.60:8881/RedLED=O

Not Secure | 192.168.0.60:8881/RedLED=ON&GreenLED=OFF

Apps · Smart card ATR p... · RabbitMQ Manag... · Custom Title Bloc... · YOUTRACK · YouTrack Dashbo...

Under the hood of ledcontrol.py...

Looking at the source code for ledcontrol.py you will notice that it is designed to constantly listen for HTTP requests incoming on TCP port 8881.

The commands allow for the respective LED colour to be turned on, turned off, slow flash, and fast flash. I found that GPIO requests are expensive in terms of CPU usage, therefore the code is written in a way that might seem cumbersome- but believe me its necessary as we want to keep CPU usage as low as possible. Remember this is a NAS, ans we want to reserve all the resources for serving files.

A thread is utilized, running at 1mS intervals, and only if an update to the LED is required i.e. it needs to be turned on or off, is a GPIO operation utilised. This keeps the CPU usage down to 0.5% of the overall system.

A normal socket listener as used in countless Python examples is used here- the incoming HTTP requests are parsed, and the daemon responds with a HTTP 200 OK message to ensure the browser sees the request as compete and does not sit waiting for a response.

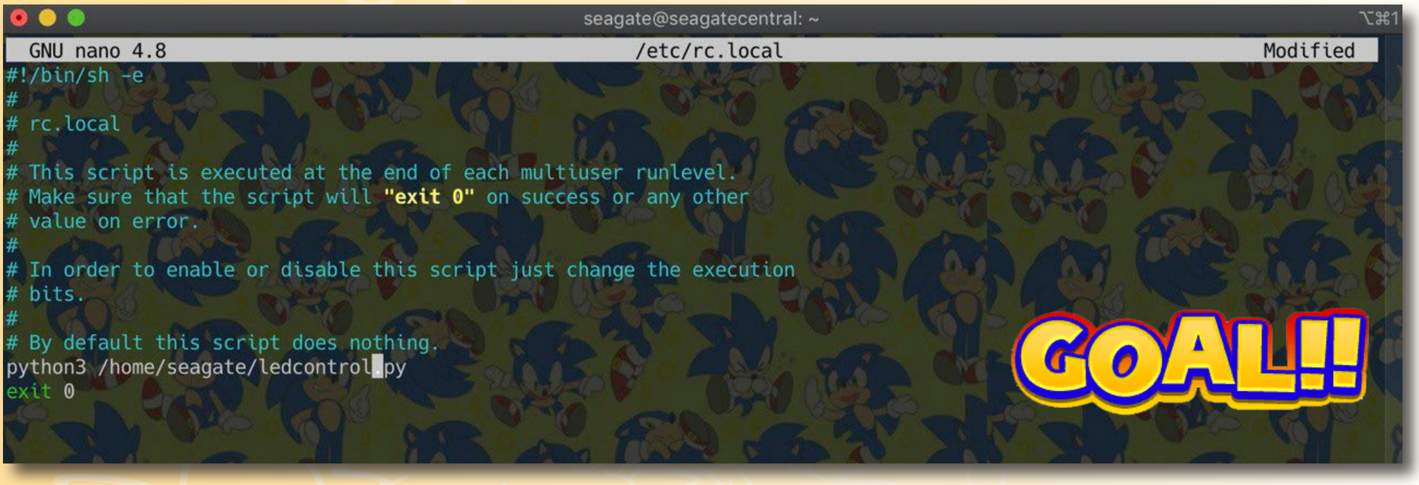
Starting it up with the system...

Letting the daemon start up automatically with the system is as simple as editing the file `rc.local` located in `/etc/` of the Linux filesystem. Before we do this, we copy `ledcontrol.py` from the git directory to `/home/seagate/`

Then, all we do is add the lines:

```
python3 /home/seagate/ledcontrol.py
exit 0
```

Save and close the file, reboot, and the LED should be controllable from a browser at the IP address assigned to the NAS (in my case 192.168.0.60) on port 8881



```
  GNU nano 4.8                        /etc/rc.local                        Modified
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.
python3 /home/seagate/ledcontrol.py
exit 0
```

Next Steps (later)...

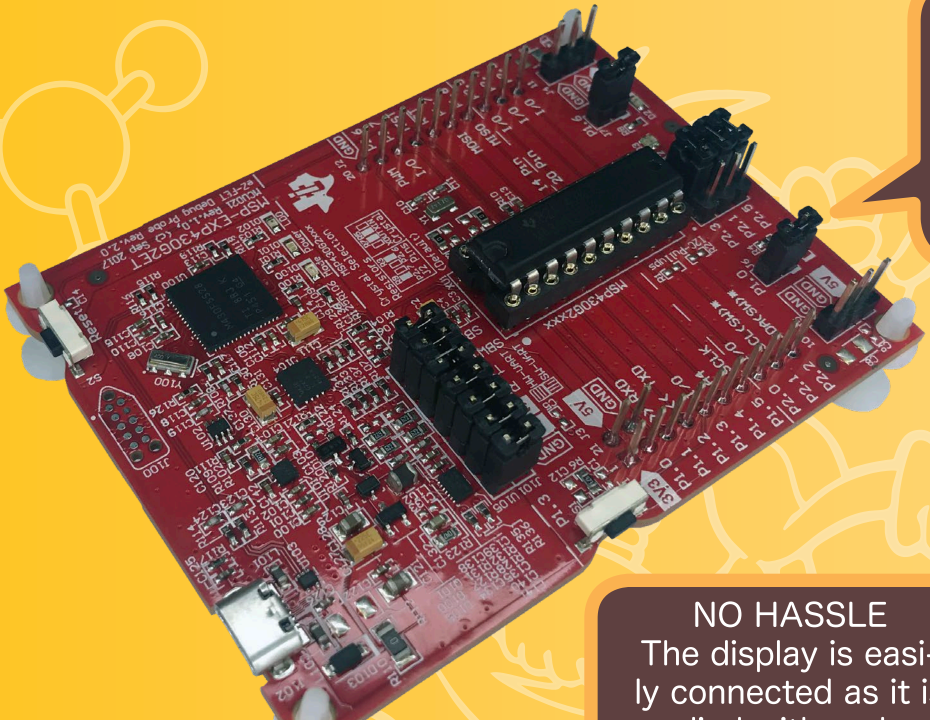
This is as unfortunately as far as we got with it folks. The world went crazy on us, and not the kind of crazy that Doctor Eggman could come up with, even if he tried.

We found that, when the system is powered up from the wall adapter, the inrush current is so high, the wall wart cannot cope with the peak current demand at startup so it begins to oscillate, powering up and down in an endless cycle. This is likely a function of the hard disk drive and may not be an issue if an SSD is used. Nonetheless its not right so we addressed it. At the back of this book, is a solution to the problem, drawn on paper, and built up on stripboard. It solved the problem perfectly.

Tinkering with things

Adventures with e-ink

We've all seen those cool e-readers from Amazon, the Kindle™ with its display that holds an image even when its totally devoid of any power whatsoever. Well, that is known as an electrophoretic display, or more commonly, "e-ink"- short for "electronic ink". Nowadays this display is referred to more commonly as "e-Paper". Recently these have become available to anyone, and now its time to have an adventure...

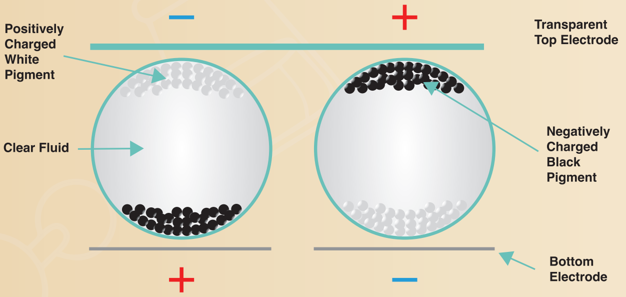


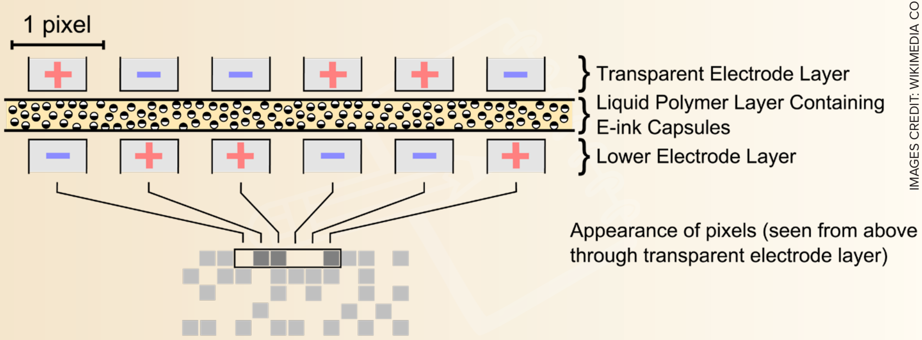
ITS COMPLETELY COMPLETE
The evaluation board is supplied complete with a USB cable so you needn't worry about that, nor getting the right processor chip. It's all in the box!

NO HASSLE
The display is easily connected as it is supplied with a plug-in harness!

HOW DOES IT WORK?

Similar to regular LCDs, these displays utilise transparent electrode layers. However, the liquid crystal is replaced by a liquid polymer packed with ink capsules. The light polarizers are also absent. The orientation of these ink capsules, that is whether they are oriented up or down then determines whether a pixel appears black or white. A capsule has two polar ends- one black, the other white. The applied electric charge will cause the ink capsules to rotate in the liquid polymer such that they will polarize towards the charge on the electrodes. Because the action is electromechanical, the display retains the image in the absence of power! This is also the reason why these displays require a whole process to clear and load images and this process is extremely slow compared to other types of display.


Positively Charged White Pigment
Clear Fluid
−
+
Transparent Top Electrode
Negatively Charged Black Pigment
+
−
Bottom Electrode


1 pixel
+ − − + + −
Transparent Electrode Layer
Liquid Polymer Layer Containing E-ink Capsules
− + + − − +
Lower Electrode Layer
Appearance of pixels (seen from above through transparent electrode layer)

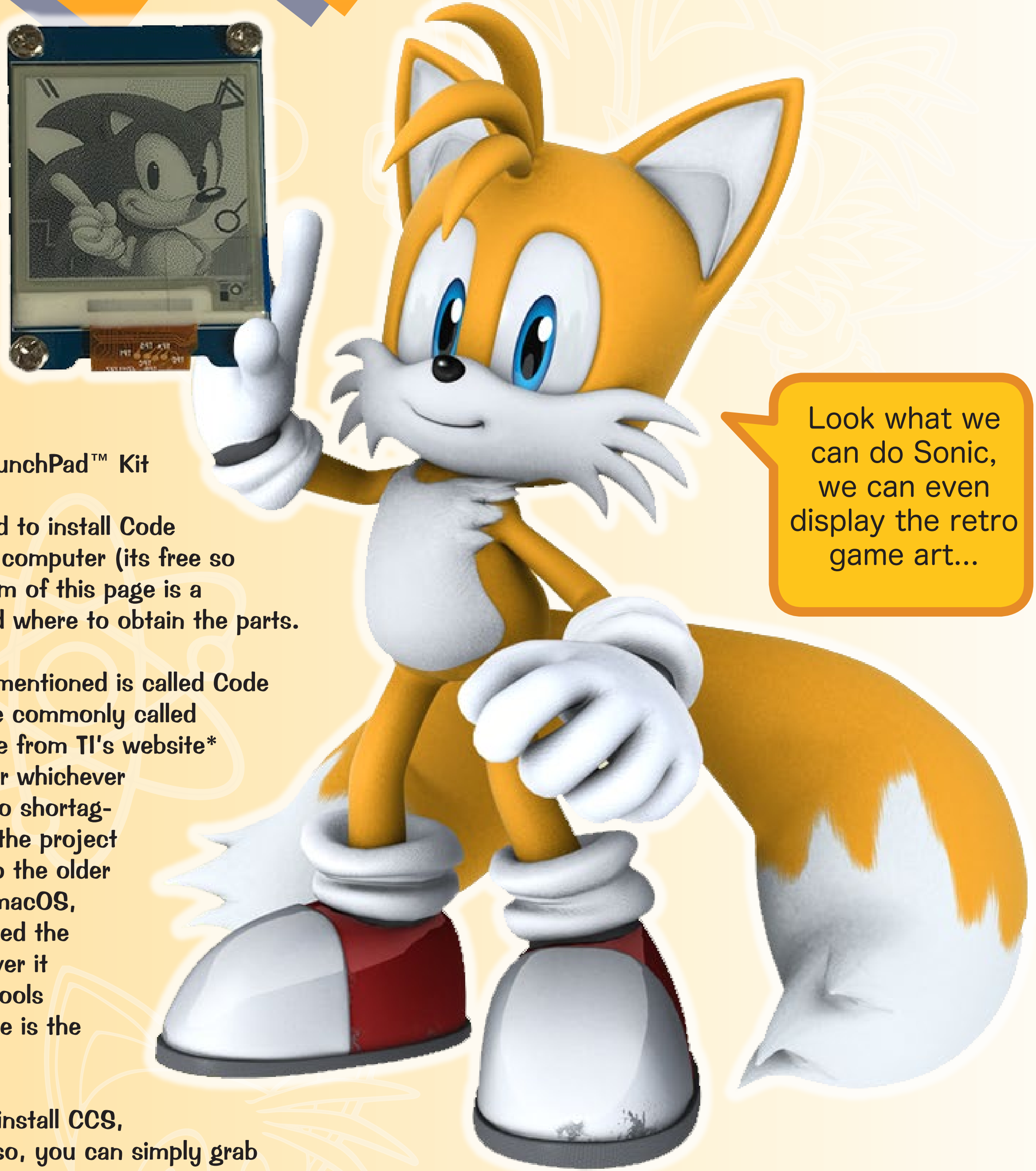
IMAGES CREDIT: WIKIMEDIA COMMONS

Let's Go!!

All right, the first thing to do obviously is to gather all the parts. This comes down to two basic items you would need to get:

- A display module
- A MSP-EXP430G2ET LaunchPad™ Kit

In addition you would need to install Code Composer Studio on your computer (its free so don't worry). At the bottom of this page is a handy-dandy parts list and where to obtain the parts.


Look what we can do Sonic, we can even display the retro game art...

The programming IDE, as mentioned is called Code Composer Studio (or more commonly called CCS). This is downloadable from TI's website* and can be downloaded for whichever computer you have. Due to shortages of TI dev kits I started the project on a Windows™ PC due to the older board not functioning on macOS, but, when the boards arrived the issue was resolved. However it should not matter as the tools are the same, and the code is the same regardless...

The first logical step is to install CCS, and when you have done so, you can simply grab the latest version of this project's source code by means of a git pull from the repository:

```
https://github.com/TheSonicZone/e-Ink
```

Then, all you need to do is "import" the project into Code Composer Studio from the location where you performed the git clone to, and it should compile when you click the "hammer" icon with no issue...

Parts and Software

IT LaunchPad MSP-EXP430G2ET from TI Webstore https://www.ti.com/tool/MSP-EXP430G2ET
MSP430G2553 PDIP 20 (supplied with kit but you can get more from us or TI Webstore)
WAVESHARE 1.54" E-PAPER (robotics.org.za Product Code W12955)
* Software- Code Composer Studio https://www.ti.com/tool/CCSTUDIO
GitKraken (or your preferred Git version control client)
Adobe Photoshop (or any image editor that can process the .iff format)

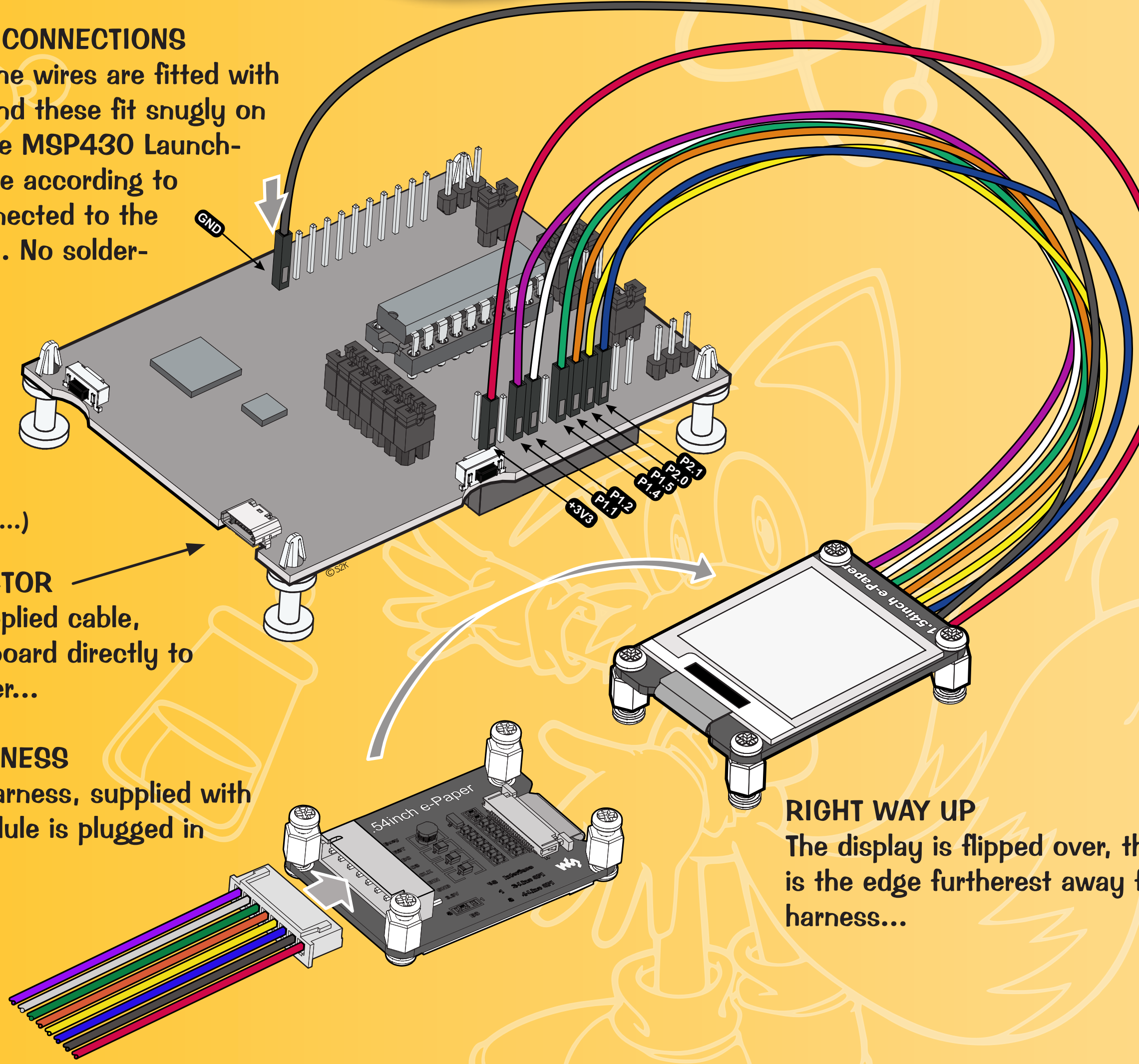
HARDWARE SUPPLIED BY: MICRO ROBOTICS robotics.org.za Texas Instruments ti.com

Putting it together...

LAUNCH YOUR DESIGN.
TI LaunchPad
TI LaunchPad™ kit with MSP430™ MCU
Microcontroller development kit for rapid prototyping featuring the MSP430G2553 microcontroller
Part no. MSP-EXP430G2ET
TEXAS INSTRUMENTS
START
Evaluation kit user guide
Thank you for your purchase of this evaluation kit.
Please read and follow all instructions and warnings in the evaluation kit user guide.
1. Visit TI.com to download the user guide, software and other resources.

UNPACKING THE BOX...

The USB cable (USB-A to micro USB-B) is supplied and is found by lifting out the red cardboard tray in the box...



MAKING THE CONNECTIONS

The ends of the wires are fitted with receptacles and these fit snugly on the pins of the MSP430 Launch-Pad. Each wire according to colour is connected to the pins indicated. No soldering required!

(See table for pin functions...)

USB CONNECTOR

Using the supplied cable, connect the board directly to your computer...

PLUG-IN HARNESS

The plug-in harness, supplied with the e-ink module is plugged in as shown...

RIGHT WAY UP

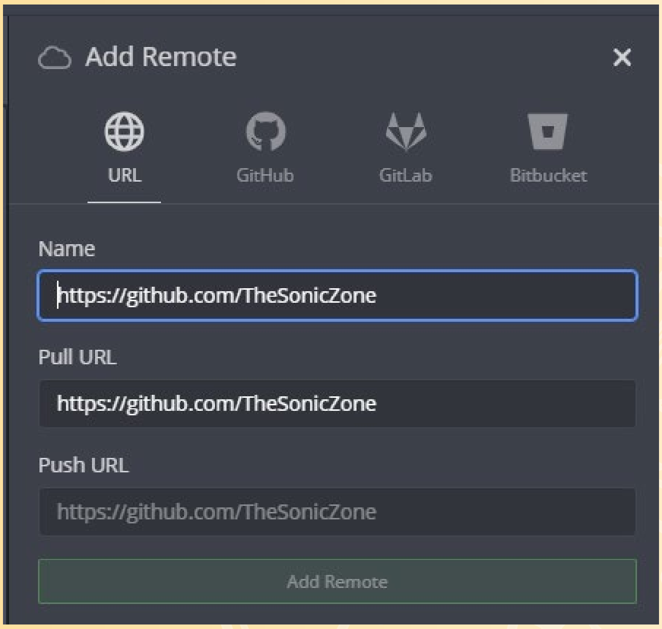
The display is flipped over, the bottom edge is the edge furtherest away from the wiring harness...

Getting the project repository cloned...

I am going to be honest here, I am not a big fan of GitKraken because things tend to get a bit confusing when there's branches and the merging of same. But for this purpose, it sure beats anything else, so we will use it to get the source code checked out onto our machine. On page 37 the Github repository was given and we will use that now to get the project cloned to our local machine. Before you do this, please ensure that Code Composer Studio 10.x is installed and you know where its workspace directory is located- you will need the information about the workspace location as a destination for clone operation.

Note the following:
The repository is read-only to the general public at this time. If you would like to contribute please contact us and we will consider your request for write access. This will be handled on a case-by-case basis only.

First add a remote repo by using the Add Remote feature in GitKraken.

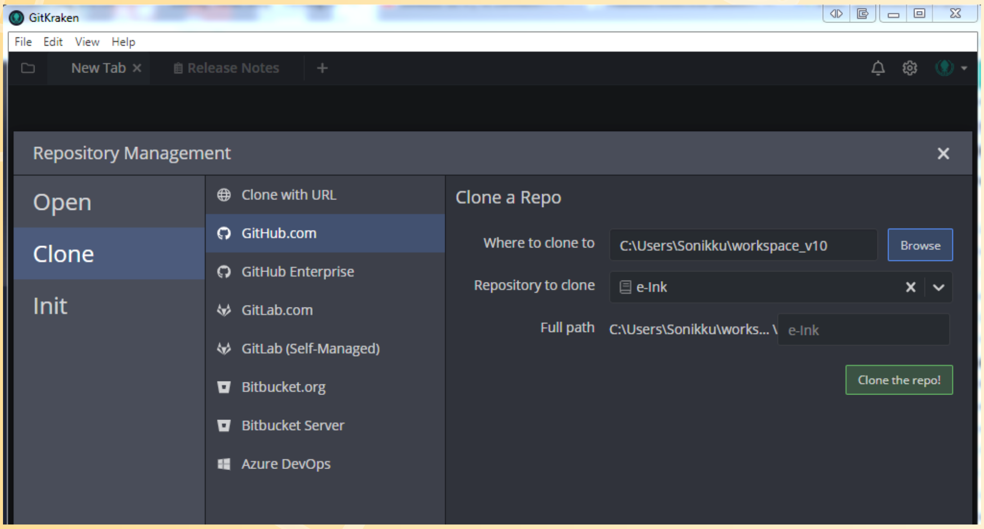


Name: https://github.com/TheSonicZone

Pull URL: https://github.com/TheSonicZone

Push URL: https://github.com/TheSonicZone

Note- in this case the button is greyed out because we already have access to the repository as we're the owners, but on your desktop it will allow you to perform the Add Remote operation. When the above step is complete, you can begin to clone to local i.e. your machine. Fill in the path to the CCS workspace directrory as shown below. The Repository to clone box will allow you to select between various projects we offer, select the e-Ink project and hit Clone The Repo...



Where to clone to: C:\Users\Sonikku\workspace_v10

Repository to clone: e-Ink

Full path: C:\Users\Sonikku\works... \ e-Ink

PROWER'S PROJECTS

You know what its like out there, you see all this cool gadgetry and then you see those magazines with deep pockets publish cool stuff that you thought about nearly a decade earlier. So what to do?

These are what PROWER'S PROJECTS are- all the really freaking cool stuff, without the price tag or that elitist attitude of engineering journals. Ideas and concepts that have been in my head for 5 or more years.

No expensive op-amps, no esoteric mathematics, no long-winded explanations of the merits of the design- just high tech at low cost, accessible to children and adults alike, and all open source, always!

Take what you need, share and most importantly, have fun doing it!

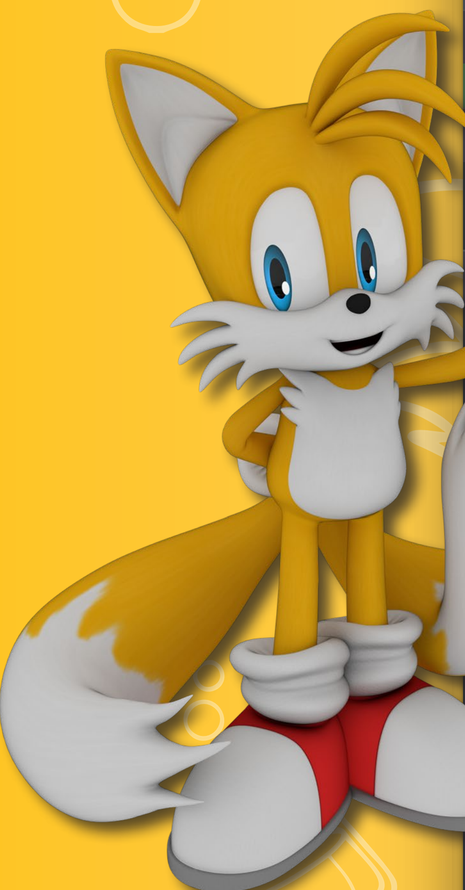
Project Import and Compilation

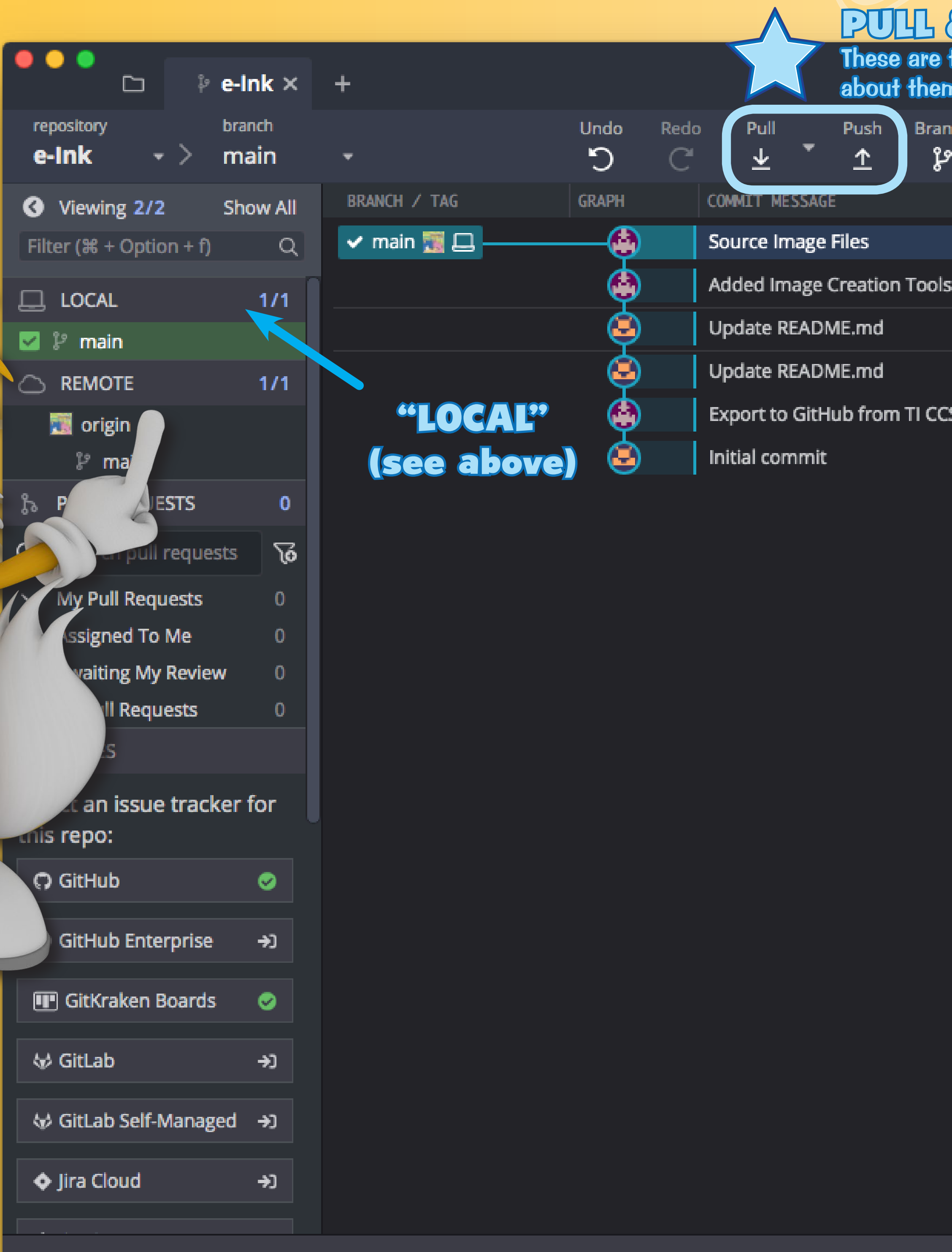
CHECKING THINGS AFTER CLONE OPERATION...

When the clone operation is complete you will notice that you can see all the commits to version control for this project that were done by us. This means the clone was successful and the local and remote are perfectly in sync. The benefit is obvious- if we improve or otherwise update the software all you need to do is a pull operation and your local copy will be updated with the improvements/changes we made...

LOCAL- on your machine's drive.

REMOTE- in the cloud i.e. github.com





PULL &
These are t
about then

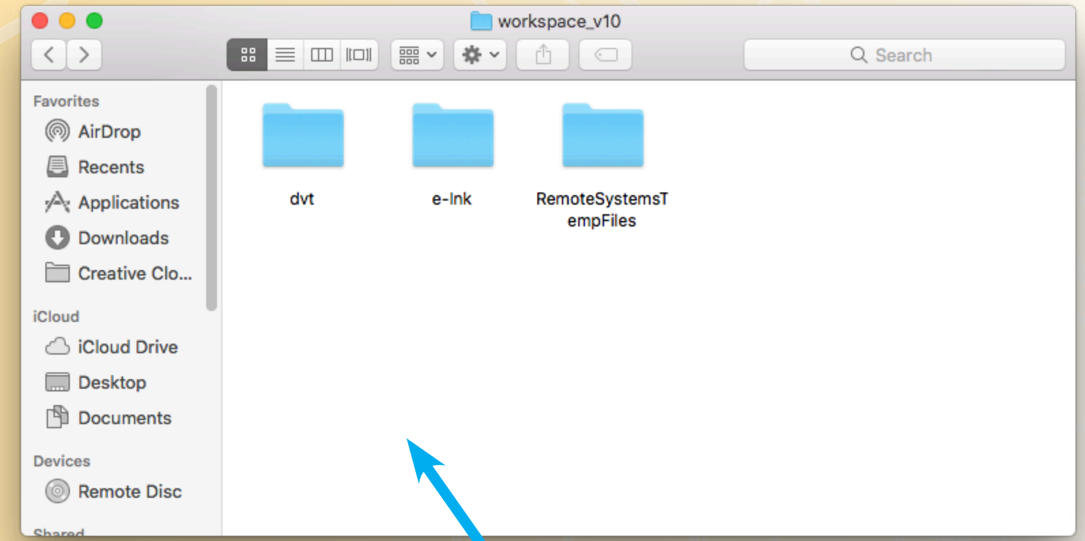
“LOCAL” (see above)

Source Image Files
Added Image Creation Tools
Update README.md
Update README.md
Export to GitHub from TI CC
Initial commit

Shown here is the workspace folder for CCS, and the folder "e-ink" is the local copy of the repository, the arrow refers to which in Gitkraken...

macOS: Users\username\workspace_v10

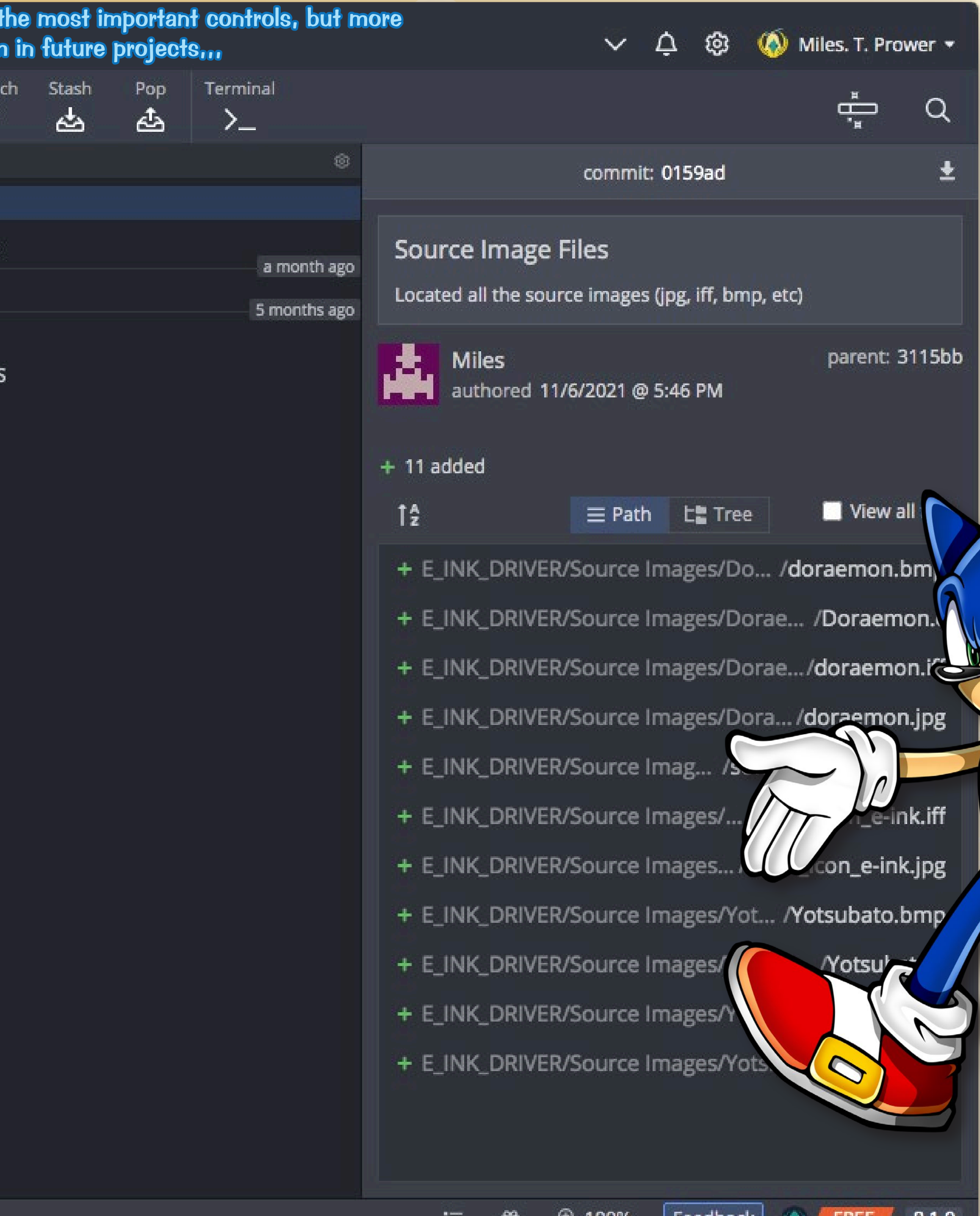
Windows: C:\Users\username\workspace_V10



This is "LOCAL"

& PUSH

the most important controls, but more
n in future projects,,,



commit: 0159ad

Source Image Files

Located all the source images (jpg, iff, bmp, etc)

Miles
authored 11/6/2021 @ 5:46 PM
parent: 3115bb

+ 11 added

```
+ E_INK_DRIVER/Source Images/Do... /doraemon.bm
+ E_INK_DRIVER/Source Images/Dorae... /Doraemon.
+ E_INK_DRIVER/Source Images/Dorae.../doraemon.i
+ E_INK_DRIVER/Source Images/Dora... /doraemon.jpg
+ E_INK_DRIVER/Source Imag... /s
+ E_INK_DRIVER/Source Images/... e-ink.iff
+ E_INK_DRIVER/Source Images... con_e-ink.jpg
+ E_INK_DRIVER/Source Images/Yot... /Yotsubato.bmp
+ E_INK_DRIVER/Source Images/ /Yotsu
+ E_INK_DRIVER/Source Images/Y
+ E_INK_DRIVER/Source Images/Yots
```

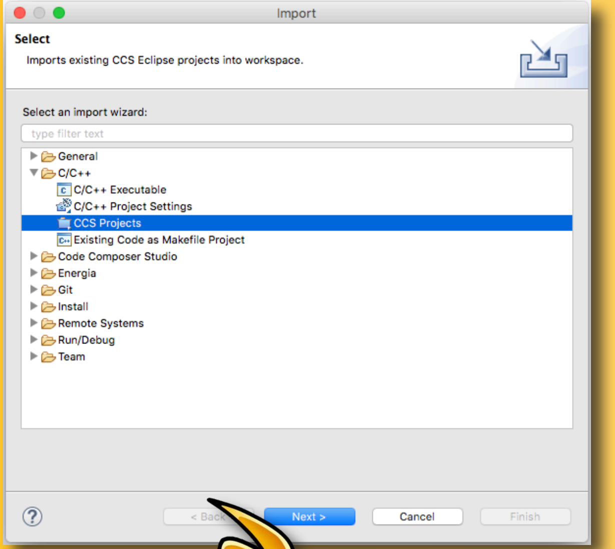
When you perform a commit, all the changed, added or deleted items are listed here...

Project Import and Compilation

IMPORT PROJECT INTO CODE COMPOSER STUDIO

Launch CCS (Code Composer Studio) Windows or macOS, the software is identical. It will appear blank because we haven't imported any projects yet. And that is the next step. The following steps show how to import the project we cloned from GitHub using GitKraken...

Using File -> Import we will have access to the Import Wizard. Select the CCS Projects option and click Next to launch the Import CCS Projects utility


Import

Select

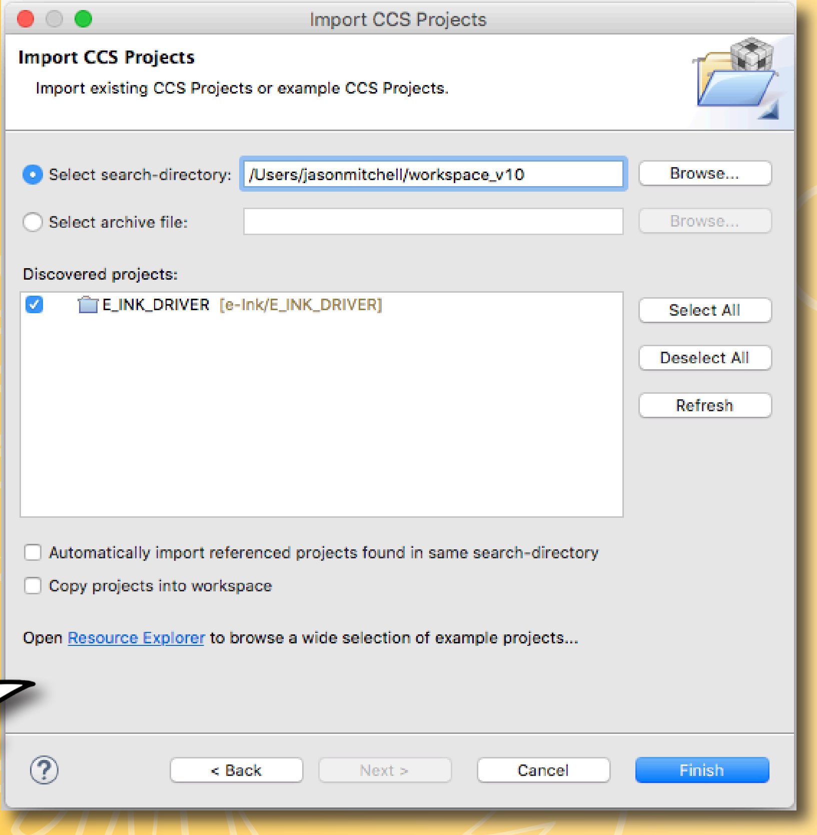
Imports existing CCS Eclipse projects into workspace.

Select an import wizard:

type filter text

- ▶ General
- ▼ C/C++
  - C/C++ Executable
  - C/C++ Project Settings
  - CCS Projects
  - Existing Code as Makefile Project
- ▶ Code Composer Studio
- ▶ Energia
- ▶ Git
- ▶ Install
- ▶ Remote Systems
- ▶ Run/Debug
- ▶ Team

When the Import CCS Projects is displayed, you simply click on Browse, and it will automatically locate the workspace, and if by magic, the project will appear in the Discovered Projects window. All you then need to do is click Finish and the e-ink project will appear in Code Composer Studio


Import CCS Projects

Import CCS Projects

Import existing CCS Projects or example CCS Projects.

● Select search-directory: /Users/jasonmitchell/workspace_v10

○ Select archive file:

Discovered projects:

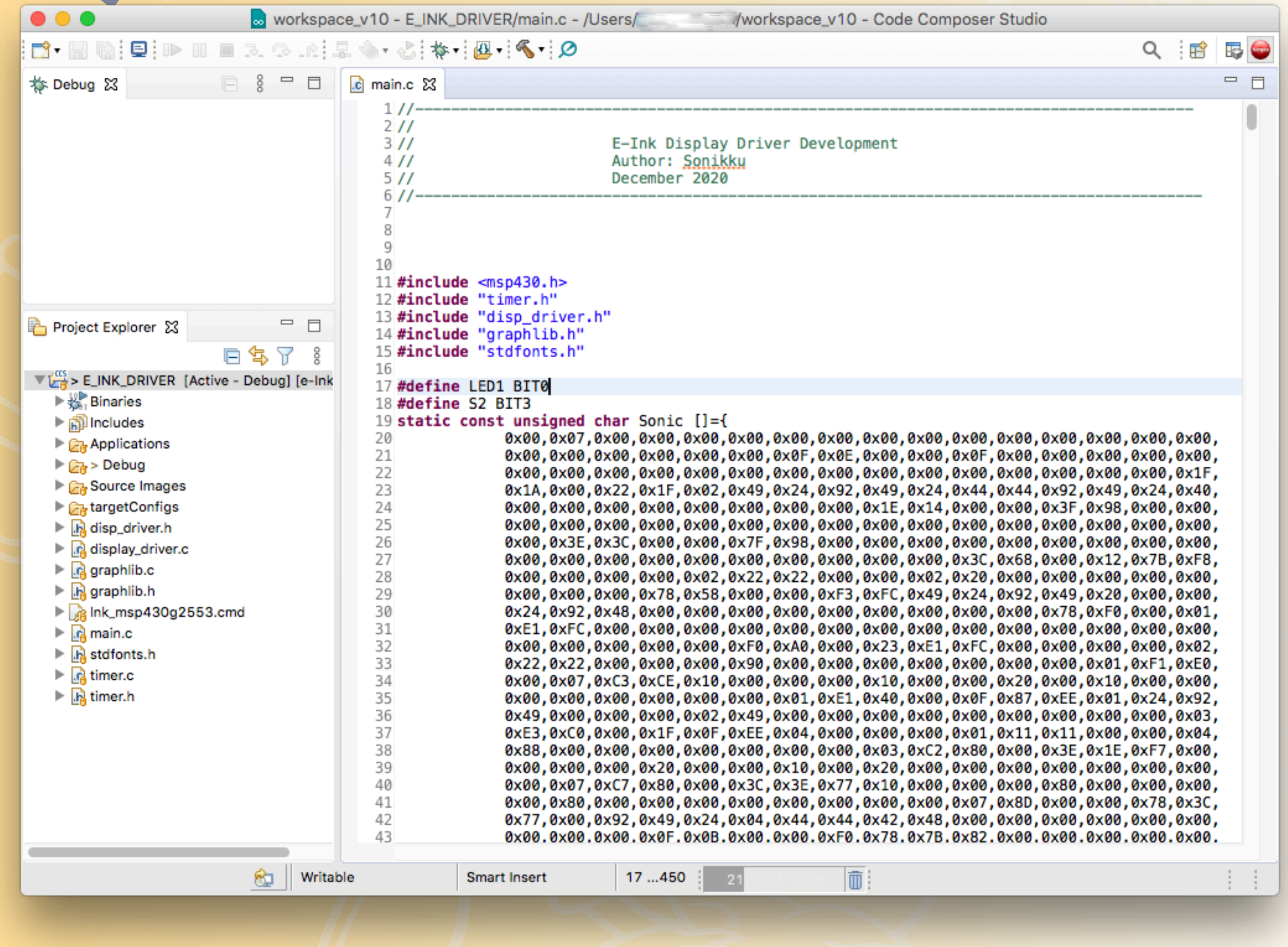
☑ E_INK_DRIVER [e-Ink/E_INK_DRIVER]

□ Automatically import referenced projects found in same search-directory

□ Copy projects into workspace

Open Resource Explorer to browse a wide selection of example projects...

TAILS ART BY ZOIBY @ DEVIANTART



```
//------------------------------------------------------------------------------------------------------
//
//                          E-Ink Display Driver Development
//                          Author: Sonikku
//                          December 2020
//-------------------------------------------------------------------------------------------------------



#include <msp430.h>
#include "timer.h"
#include "disp_driver.h"
#include "graphlib.h"
#include "stdfonts.h"

#define LED1 BIT0
#define S2 BIT3
static const unsigned char Sonic []={
                0x00,0x07,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
                0x00,0x00,0x00,0x00,0x00,0x00,0x0F,0x0E,0x00,0x00,0x0F,0x00,0x00,0x00,0x00,0x00,
                0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x1F,
                0x1A,0x00,0x22,0x1F,0x02,0x49,0x24,0x92,0x49,0x24,0x44,0x44,0x92,0x49,0x24,0x40,
                0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x1E,0x14,0x00,0x00,0x3F,0x98,0x00,0x00,
                0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
                0x00,0x3E,0x3C,0x00,0x00,0x7F,0x98,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
                0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x3C,0x68,0x00,0x12,0x7B,0xF8,
                0x00,0x00,0x00,0x00,0x02,0x22,0x22,0x00,0x00,0x02,0x20,0x00,0x00,0x00,0x00,0x00,
                0x00,0x00,0x00,0x78,0x58,0x00,0x00,0xF3,0xFC,0x49,0x24,0x92,0x49,0x20,0x00,0x00,
                0x24,0x92,0x48,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x78,0xF0,0x00,0x01,
                0xE1,0xFC,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
                0x00,0x00,0x00,0x00,0x00,0xF0,0xA0,0x00,0x23,0xE1,0xFC,0x00,0x00,0x00,0x00,0x02,
                0x22,0x22,0x00,0x00,0x00,0x90,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x01,0xF1,0xE0,
                0x00,0x07,0xC3,0xCE,0x10,0x00,0x00,0x00,0x10,0x00,0x00,0x20,0x00,0x10,0x00,0x00,
                0x00,0x00,0x00,0x00,0x00,0x00,0x01,0xE1,0x40,0x00,0x0F,0x87,0xEE,0x01,0x24,0x92,
                0x49,0x00,0x00,0x00,0x02,0x49,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,
                0xE3,0xC0,0x00,0x1F,0x0F,0xEE,0x04,0x00,0x00,0x00,0x01,0x11,0x11,0x00,0x00,0x04,
                0x88,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,0xC2,0x80,0x00,0x3E,0x1E,0xF7,0x00,
                0x00,0x00,0x00,0x20,0x00,0x00,0x10,0x00,0x20,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
                0x00,0x07,0xC7,0x80,0x00,0x3C,0x3E,0x77,0x10,0x00,0x00,0x00,0x80,0x00,0x00,0x00,
                0x00,0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x07,0x8D,0x00,0x00,0x78,0x3C,
                0x77,0x00,0x92,0x49,0x24,0x04,0x44,0x44,0x42,0x48,0x00,0x00,0x00,0x00,0x00,0x00,
                0x00,0x00,0x00,0x0F,0x0B,0x00,0x00,0xF0,0x78,0x7B,0x82,0x00,0x00,0x00,0x00,0x00,
```

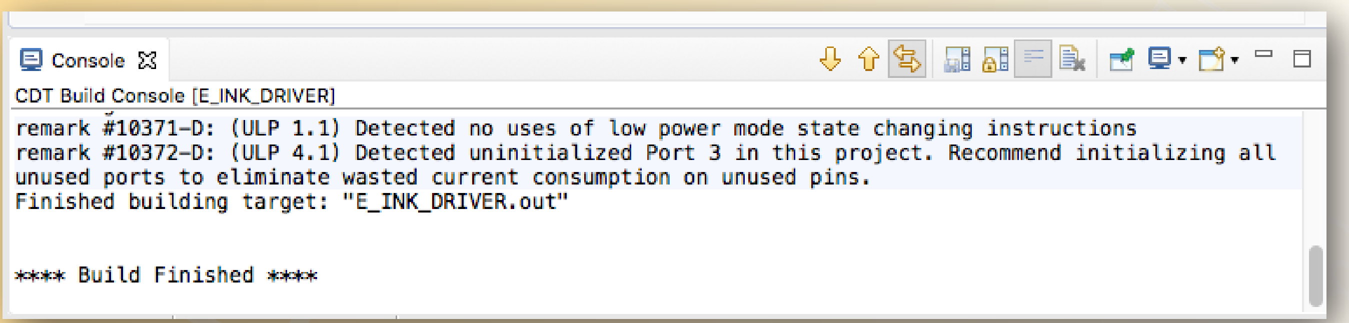
IMPORT SUCCESS - PROJECT SOURCE CODE IS VISIBLE

LET'S COMPILE THE CODE

We are now going to make sure everything is working like it should by compiling the code.
NOTE- Do not edit any of the code at this stage, we will show you how to change the images later on.

Click on the HAMMER icon near the top of the window. This kicks off the compiling, assembling, and linking process of the C code (nothing you need to worry about the project is all set-up for you)

When the build process is kicked off, a console window appears at the bottom, and if all goes well you should see no errors and "Build Finished" message. This means you compiled the code successfully. Awesome!!



```
remark #10371-D: (ULP 1.1) Detected no uses of low power mode state changing instructions
remark #10372-D: (ULP 4.1) Detected uninitialized Port 3 in this project. Recommend initializing all
unused ports to eliminate wasted current consumption on unused pins.
Finished building target: "E_INK_DRIVER.out"


**** Build Finished ****
```

Programming the Board...

CONNECT IT TO YOUR COMPUTER

The MSP430 boards usually ship with a demo program already loaded in the factory. So as a result when you connect it to your computer you may see LEDs flash, but in most cases, you should see a yellow and green LED solidly lit on the board.

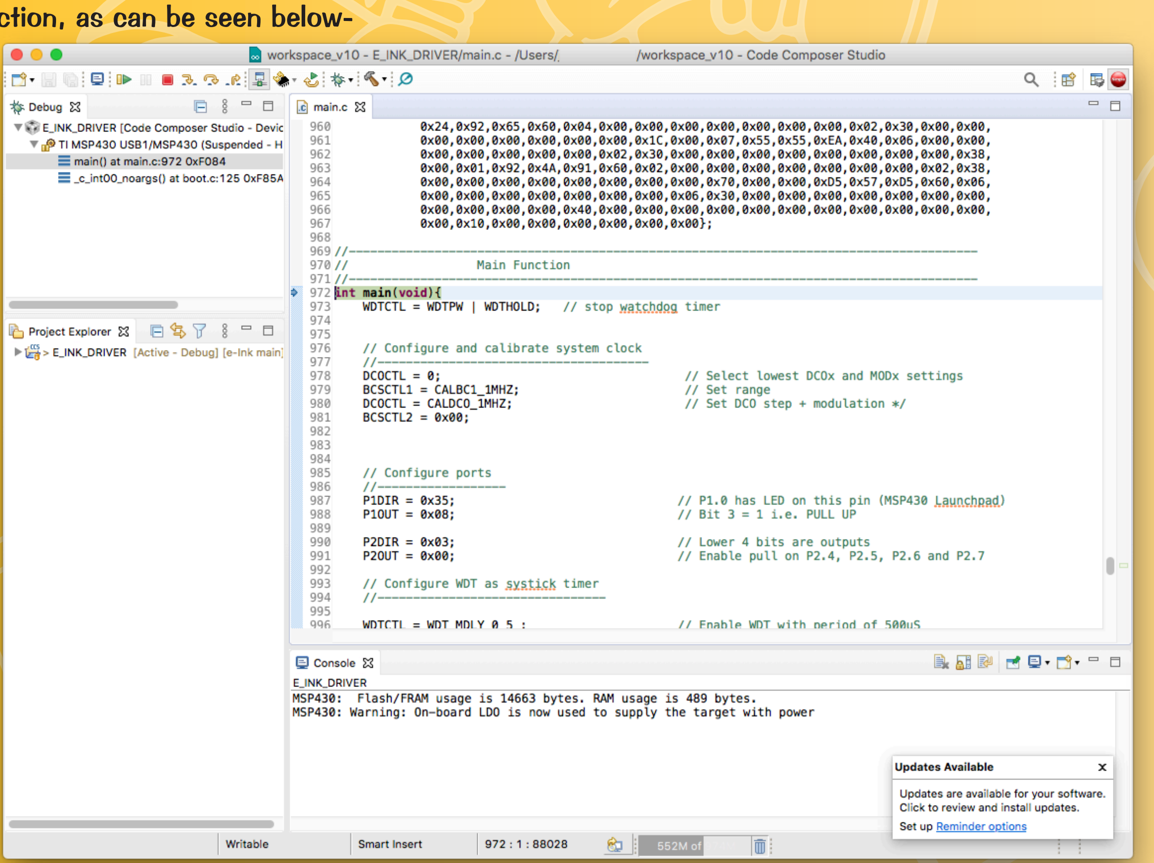
Note- the board is plugged in after CCS is launched.

We will now proceed to program the application as provided from GitHub into the board. This will determine whether there is any problem (for example display faulty or incorrectly connected) and whether any other kind of problem exists. With the board plugged in, we click the DEBUG button as shown...

If you see a pop-up window titled "ULP Advisor" (which would be displayed as we are not taking into account the Ultra-Low-Power features of the processor), simply check the checkbox "Do not show this again" and click Proceed

NOTE- the very first time you do this, it may take a long time to get going. Also note that on older versions of macOS there are known issues with the debugger not working. See the Troubleshooting page for steps to solve this.

If all goes well, the board will be in DEBUGGING MODE, and the program will hit the automatic breakpoint on the `main()` function, as can be seen below-



workspace_v10 - E_INK_DRIVER/main.c - /Users/ /workspace_v10 - Code Composer Studio
Debug
main.c
Project Explorer
Console

E_INK_DRIVER [Code Composer Studio - Devic
TI MSP430 USB1/MSP430 (Suspended - H
main() at main.c:972 0xF084
_c_int00_noargs() at boot.c:125 0xF85A

E_INK_DRIVER [Active - Debug] [e-Ink main]

```
            0x24,0x92,0x65,0x60,0x04,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x02,0x30,0x00,0x00,
            0x00,0x00,0x00,0x00,0x00,0x00,0x1C,0x00,0x07,0x55,0x55,0xEA,0x40,0x06,0x00,0x00,
            0x00,0x00,0x00,0x00,0x00,0x02,0x30,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x38,
            0x00,0x01,0x92,0x4A,0x91,0x60,0x02,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x02,0x38,
            0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x70,0x00,0x00,0xD5,0x57,0xD5,0x60,0x06,
            0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x06,0x30,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
            0x00,0x00,0x00,0x00,0x40,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
            0x00,0x10,0x00,0x00,0x00,0x00,0x00,0x00};

//-------------------------------------------------------------------------------------
//                    Main Function
//-------------------------------------------------------------------------------------
int main(void){
    WDTCTL = WDTPW | WDTHOLD;   // stop watchdog timer


    // Configure and calibrate system clock
    //-----------------------------------
    DCOCTL = 0;                                      // Select lowest DCOx and MODx settings
    BCSCTL1 = CALBC1_1MHZ;                           // Set range
    DCOCTL = CALDCO_1MHZ;                            // Set DCO step + modulation */
    BCSCTL2 = 0x00;



    // Configure ports
    //----------------
    P1DIR = 0x35;                                   // P1.0 has LED on this pin (MSP430 Launchpad)
    P1OUT = 0x08;                                   // Bit 3 = 1 i.e. PULL UP

    P2DIR = 0x03;                                   // Lower 4 bits are outputs
    P2OUT = 0x00;                                   // Enable pull on P2.4, P2.5, P2.6 and P2.7

    // Configure WDT as systick timer
    //----------------------------

    WDTCTL = WDT_MDLY_0_5 ;                         // Enable WDT with period of 500uS
```

E_INK_DRIVER

```
MSP430:  Flash/FRAM usage is 14663 bytes. RAM usage is 489 bytes.
MSP430: Warning: On-board LDO is now used to supply the target with power
```

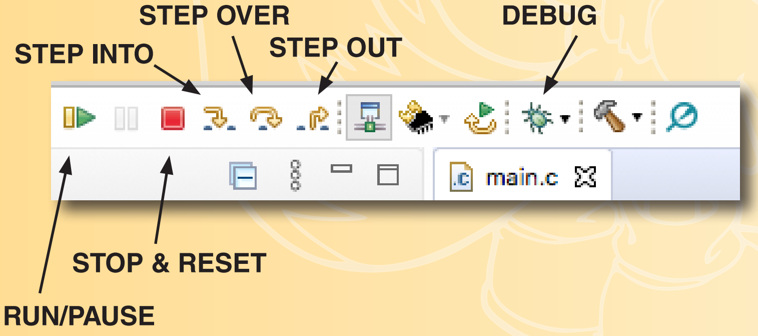
Updates Available ×

Updates are available for your software. Click to review and install updates.

Set up Reminder options

Writable | Smart Insert | 972 : 1 : 88028 | 552M of

DEBUGGER CONTROLS



USB DRIVERS?

macOS
Usually the drivers for the MSP430 are installed at the same time the software is installed, no steps are necessary it just usually works.

Microsoft Windows
The first time you connect this board you will probably see notifications about installing device drivers. This should only happen once (and may re-occur if you use a different USB port or a port on a USB hub). Regardless the drivers are present, the OS simply has to "find" them and install them.

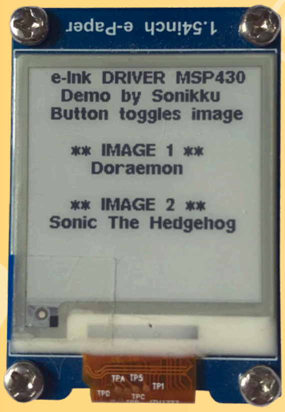
Programming the Board...

LET'S PLAY

Now comes the moment of fun, we simply click the PLAY/PAUSE button on the debugger controls.
Now, assuming all the electrical connections are solid, within seconds you will notice the e-ink screen change, flicker and then display the startup screen text as shown below. Two images are referenced, one of Doraemon and the other of Sonic.

This is text, not an image. There are embedded fonts in the source code (more about this later)

Note: if you do not see this, re-check your connections to the e-ink display or try another e-ink display- they are somewhat fragile items!



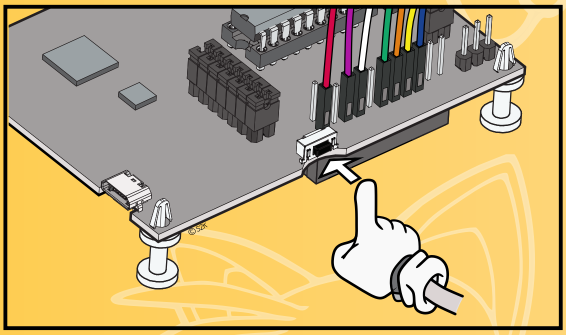
e-Ink DRIVER MSP430
Demo by Sonikku
Button toggles image

** IMAGE 1 **
Doraemon

** IMAGE 2 **
Sonic The Hedgehog

PRESS THE BUTTON...

On the left hand side of the board, is a push button. This button is attached to a GPIO pin (P1.3) and the software polls it. When you press it with the start up screen displayed, the screen will go blank, and the first image will be loaded. During loading a green LED at the bottom of the board will light up, indicating that processing is taking place. Note that this is an e-ink display, it will take a good few seconds to display the image (the image is in fact drawn twice) and its slow due to how the display works.



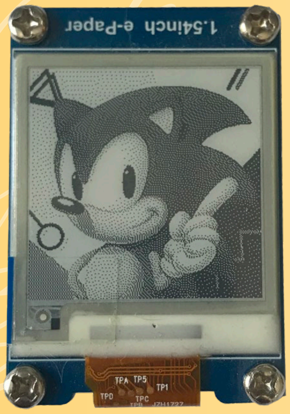
And then you will see Fujiko.F.Fujio's DORAEMON

DORAEMON © FUJIKO.F.FUJIO 藤子·F·不二雄

PRESS THE BUTTON, AGAIN...

Pressing the button again, then replaces the image with that of SONIC from the 1991 Genesis cartridge branding, as it appeared just over thirty years ago...

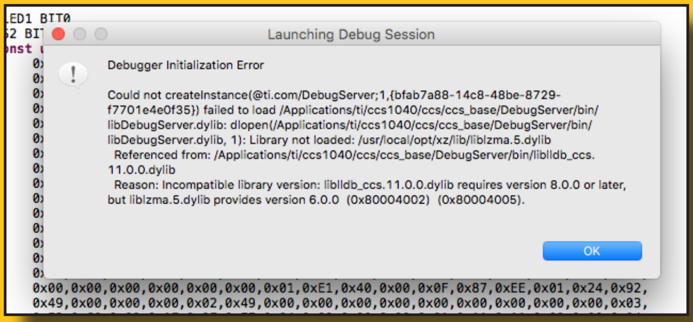
Pressing the button therefore alternates between the two images. Note that if the board is reset, it will default to the very first (text) screen.



TROUBLESHOOTING macOS ISSUES WITH CCS

Debugger Won't Work

When clicking Debug, the following error message is displayed:



Launching Debug Session

Debugger Initialization Error

Could not createInstance(@ti.com/DebugServer;1,{bfab7a88-14c8-48be-8729-f7701e4e0f35}) failed to load /Applications/ti/ccs1040/ccs/ccs_base/DebugServer/bin/libDebugServer.dylib: dlopen(/Applications/ti/ccs1040/ccs/ccs_base/DebugServer/bin/libDebugServer.dylib, 1): Library not loaded: /usr/local/opt/xz/lib/liblzma.5.dylib
 Referenced from: /Applications/ti/ccs1040/ccs/ccs_base/DebugServer/bin/liblldb_ccs.11.0.0.dylib
 Reason: Incompatible library version: liblldb_ccs.11.0.0.dylib requires version 8.0.0 or later, but liblzma.5.dylib provides version 6.0.0 (0x80004002) (0x80004005).

```
0x00,0x00,0x00,0x00,0x00,0x00,0x01,0xE1,0x40,0x00,0x0F,0x87,0xEE,0x01,0x24,0x92,
0x49,0x00,0x00,0x00,0x02,0x49,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x03,
```

The complaint relates to liblzma being an incompatible version. This often happens on older versions of macOS (High Sierra being a common culprit- also seen on Yosemite and Catalina by the way...)

Solution

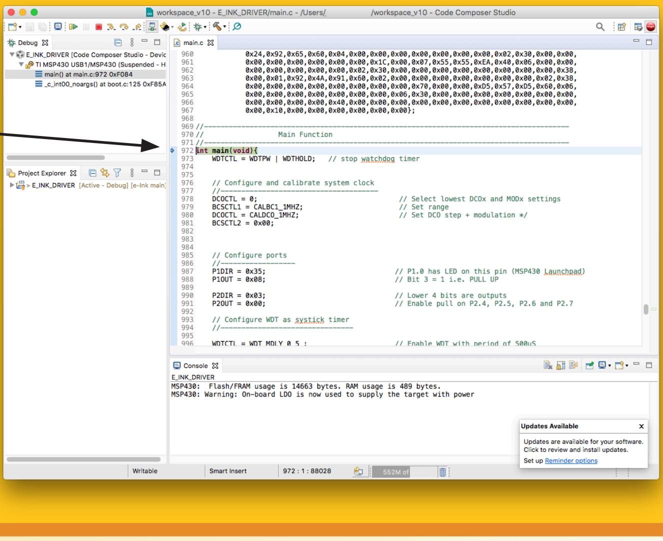
Assuming you have iTerm2 installed with HomeBrew, you need to execute the following commands in the shell.

```
sudo chown -R $(whoami) /usr/local/share/man/man8
sudo chmod u+w /usr/local/share/man/man8
```

followed by...

```
brew install xz
```

You **MUST** then restart Code Composer Studio (and preferably macOS to be on the safe side) Once restarted, clicking Debug should run with no hitches (the first time though might take a bit long, you will see LEDs flash on the board) and it will eventually hit the automatic breakpoint on the main() function as can be seen here... **This means it successfully programmed the board!**



workspace_v10 - E_INK_DRIVER/main.c - /Users/ /workspace_v10 - Code Composer Studio
Debug
E_INK_DRIVER [Code Composer Studio - Devic
TI MSP430 USB1/MSP430 (Suspended - H
main() at main.c:972 0xF084
_c_int00_noargs() at boot.c:125 0xF85A
Project Explorer
> E_INK_DRIVER [Active - Debug] [e-Ink main]
main.c
Console
E_INK_DRIVER
Writable
Smart Insert
972 : 1 : 88028
552M of

```
            0x24,0x92,0x65,0x60,0x04,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x02,0x30,0x00,0x00,
            0x00,0x00,0x00,0x00,0x00,0x00,0x1C,0x00,0x07,0x55,0x55,0xEA,0x40,0x06,0x00,0x00,
            0x00,0x00,0x00,0x00,0x00,0x02,0x30,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x38,
            0x00,0x01,0x92,0x4A,0x91,0x60,0x02,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x02,0x38,
            0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x70,0x00,0x00,0xD5,0x57,0xD5,0x60,0x06,
            0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x06,0x30,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
            0x00,0x00,0x00,0x00,0x40,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,
            0x00,0x10,0x00,0x00,0x00,0x00,0x00,0x00};

//---------------------------------------------------------------------------
//                 Main Function
//---------------------------------------------------------------------------
int main(void){
    WDTCTL = WDTPW | WDTHOLD;   // stop watchdog timer


    // Configure and calibrate system clock
    //------------------------------------
    DCOCTL = 0;                                 // Select lowest DCOx and MODx settings
    BCSCTL1 = CALBC1_1MHZ;                      // Set range
    DCOCTL = CALDCO_1MHZ;                       // Set DCO step + modulation */
    BCSCTL2 = 0x00;



    // Configure ports
    //------------------
    P1DIR = 0x35;                               // P1.0 has LED on this pin (MSP430 Launchpad)
    P1OUT = 0x08;                               // Bit 3 = 1 i.e. PULL UP

    P2DIR = 0x03;                               // Lower 4 bits are outputs
    P2OUT = 0x00;                               // Enable pull on P2.4, P2.5, P2.6 and P2.7

    // Configure WDT as systick timer
    //------------------------------

    WDTCTL = WDT_MDLY_0_5 ;                     // Enable WDT with period of 500uS
```

```
MSP430:  Flash/FRAM usage is 14663 bytes. RAM usage is 489 bytes.
MSP430: Warning: On-board LDO is now used to supply the target with power
```

Updates Available

Updates are available for your software. Click to review and install updates.

Set up Reminder options

Breakpoint Stop

A ***breakpoint*** is a programmed waypoint in a computer program- when it is reached, execution stops. Breakpoints are set by developers to debug their code, useful for locating bugs in their code, and sometimes to perform tests. Effective debugging with breakpoints leads to reliabilty!

Making it your own...

WRITING TEXT

The project source includes three bitmapped fonts in fixed sizes that have been developed over two decades by Sonic and Tails in numerous embedded projects. These fonts are found in the file `stdfonts.h`

The text to be written can be placed on the screen at any desired location as long as such a location falls within a row of 8 pixels.

FEATURES & LIMITATIONS

SCREEN AREA: 200 x 200 pixels
bitmap 1bpp
FONTS: Regular 5x7 dot matrix
Akzidenz Grotesk 16 x 12
Tiny 5x5 dot matrix
IMAGE STORAGE: Max 2 images
(limited by flash memory size
of the MSP430G2553)

Text and graphics functions are provided by `graphlib.c`, which in turn interfaces to the display driver consisting of `disp_driver.c` and `disp_driver.h`

Due to the way that the e-ink works, it is necessary to use three function calls to do the work of placing text. These two functions are called one after the other and this means unfortunately that it is not possible to leverage the standard C library `printf()` due to code space constraints (and the complexity of the e-ink).

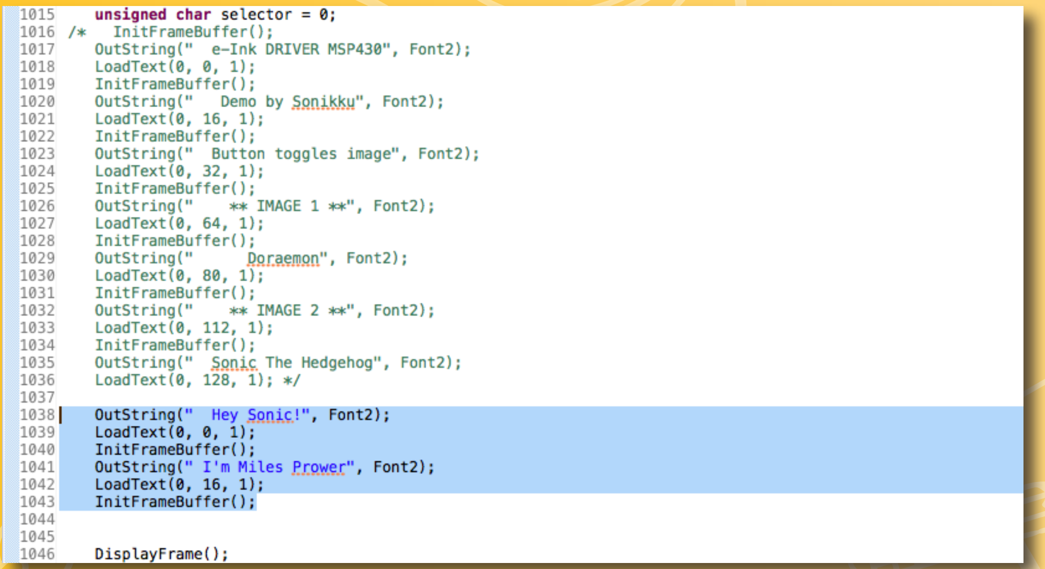
The two functions are...

```
OutString(String Array, Font)
LoadText (x position, y position, Height in units of 8 pixels)
```

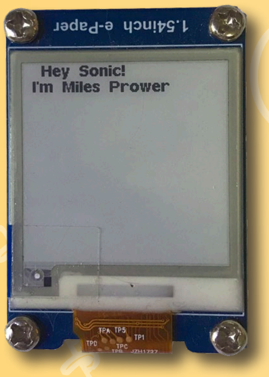
Then, `InitFrameBuffer()` is called to update the changes to the buffer.
When all text is loaded, finally `DisplayFrame()` is called and the entire matrix is written twice to ensure crisp, smooth, dark images

EXAMPLE: MODIFY STARTUP SCREEN

So to demonstrate how easy it is to put our own bespoke text on the screen, we do the following:
We comment out the existing code using `/*` and `*/` and then we add some new lines of code as shown:



```
    unsigned char selector = 0;
/*    InitFrameBuffer();
    OutString("  e-Ink DRIVER MSP430", Font2);
    LoadText(0, 0, 1);
    InitFrameBuffer();
    OutString("   Demo by Sonikku", Font2);
    LoadText(0, 16, 1);
    InitFrameBuffer();
    OutString("  Button toggles image", Font2);
    LoadText(0, 32, 1);
    InitFrameBuffer();
    OutString("    ** IMAGE 1 **", Font2);
    LoadText(0, 64, 1);
    InitFrameBuffer();
    OutString("      Doraemon", Font2);
    LoadText(0, 80, 1);
    InitFrameBuffer();
    OutString("    ** IMAGE 2 **", Font2);
    LoadText(0, 112, 1);
    InitFrameBuffer();
    OutString("  Sonic The Hedgehog", Font2);
    LoadText(0, 128, 1); */

    OutString("  Hey Sonic!", Font2);
    LoadText(0, 0, 1);
    InitFrameBuffer();
    OutString(" I'm Miles Prower", Font2);
    LoadText(0, 16, 1);
    InitFrameBuffer();


    DisplayFrame();
```



LOADING IMAGES

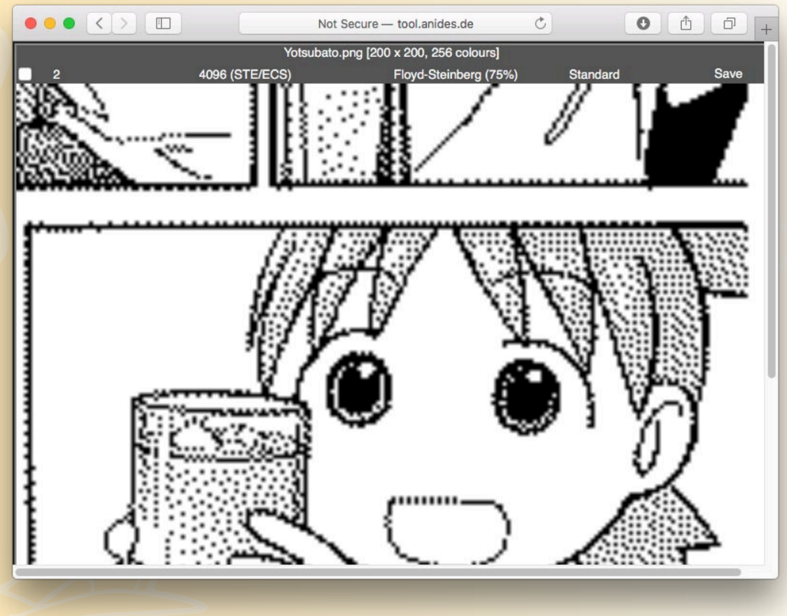
The task of displaying images is somewhat more complex. Images need to be converted from colour to 1 bit per pixel bitmaps and then converted into a C array which is in turn pasted into the source code.
Provided within the project source are three sample images shown alongside and can be found in the "Source Images" directory...

IMAGE CREDITS: YOTSUBATO © KIYOHIKO AZUMA - SONIC © SEGA - DORAEMON © FUJIKO-F-FUJIO 藤子・F・不二雄

EXAMPLE: PREPARE IMAGE, INSERT

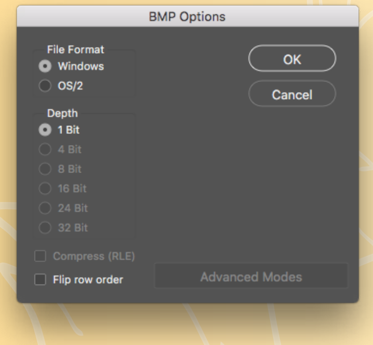
For our example we will use the image of Yotsuba screen-capped directly from the digital version of the Yotsubato manga. We use Photoshop (or any decent image editor) to resize the image (grayscale or colour) to 200 x 200 pixels.
In the source directory we have already done this for you.
Next we have to convert this grayscale image to a good 1bpp using a good halftoning application and to do this we visit the website tool.anides.de (if you know how to get good results in Photoshop, this step is optional). Simply close the default image that appears and then drag and drop the image (.png) of Yotsuba onto the web page.

On the left menu, change the option from Spectrum 512/4k to 2.
On the third menu from the left, change the option from None, to an option where the image looks the best. Usually Floyd-Steinberg (75%) yields the best result, but play around until it looks best. Then select Save and it will save the result to a .iff file.



Not Secure — tool.anides.de
Yotsubato.png [200 x 200, 256 colours]
2
4096 (STE/ECS)
Floyd-Steinberg (75%)
Standard
Save

Open the .iff file in Adobe Photoshop, and confirm that the image looks nicely halftoned as shown in the screen capture. Next you will need to save the file as a Windows bitmap (.bmp). Ensure you set the settings exactly as shown here when the save options are presented-



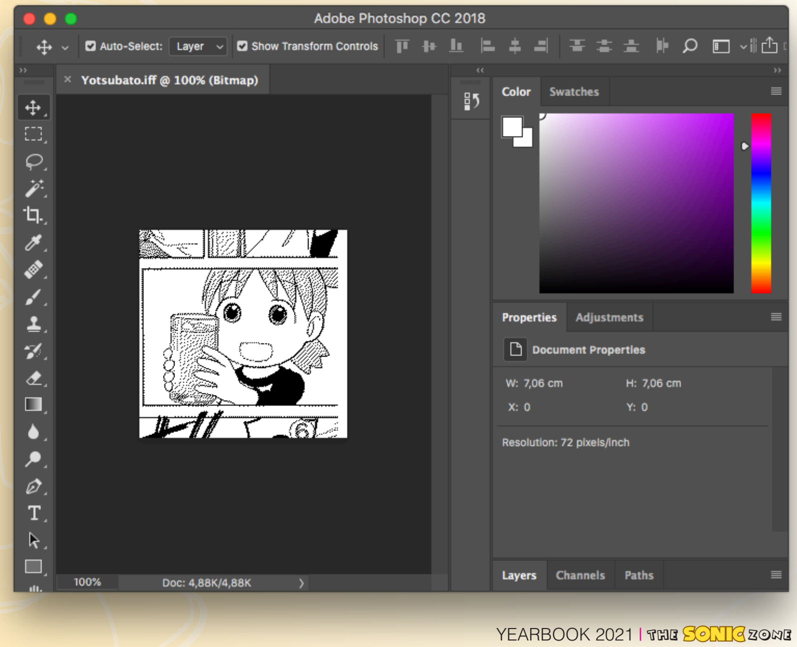
BMP Options

File Format
● Windows
○ OS/2

Depth
● 1 Bit
○ 4 Bit
○ 8 Bit
○ 16 Bit
○ 24 Bit
○ 32 Bit

□ Compress (RLE)
□ Flip row order

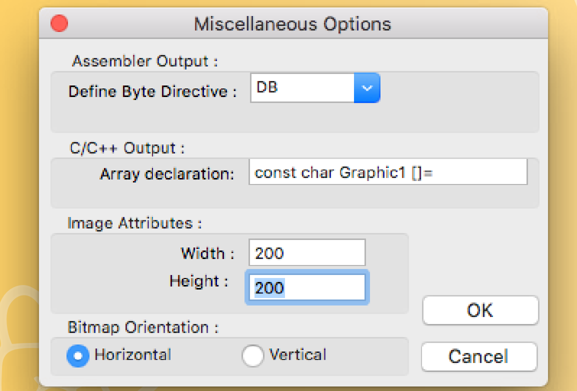
Next, we will need to convert the image to a C array in order to insert it into the source code. In the project is a directory called "Applications". Within this directory is a tool called LCDImager. A Windows version and a macOS version is provided.
The Windows version may be run standalone, by double-clicking on the executable.



Making it your own...

LOADING IMAGES (cont.)

For the macOS version however, the application needs to be installed. Mount and open the .dmg image and the copy LCDImager to the Applications folder. It then appears as an icon in the Launchpad.
Launch LCDImager and then proceed to set it up by using the Options drop-down menu i.e. [Options -> Miscellaneous]. Set the parameters as shown:



Miscellaneous Options

Assembler Output :

Define Byte Directive : DB

C/C++ Output :

Array declaration: const char Graphic1 []=

Image Attributes :

Width : 200

Height : 200

Bitmap Orientation :

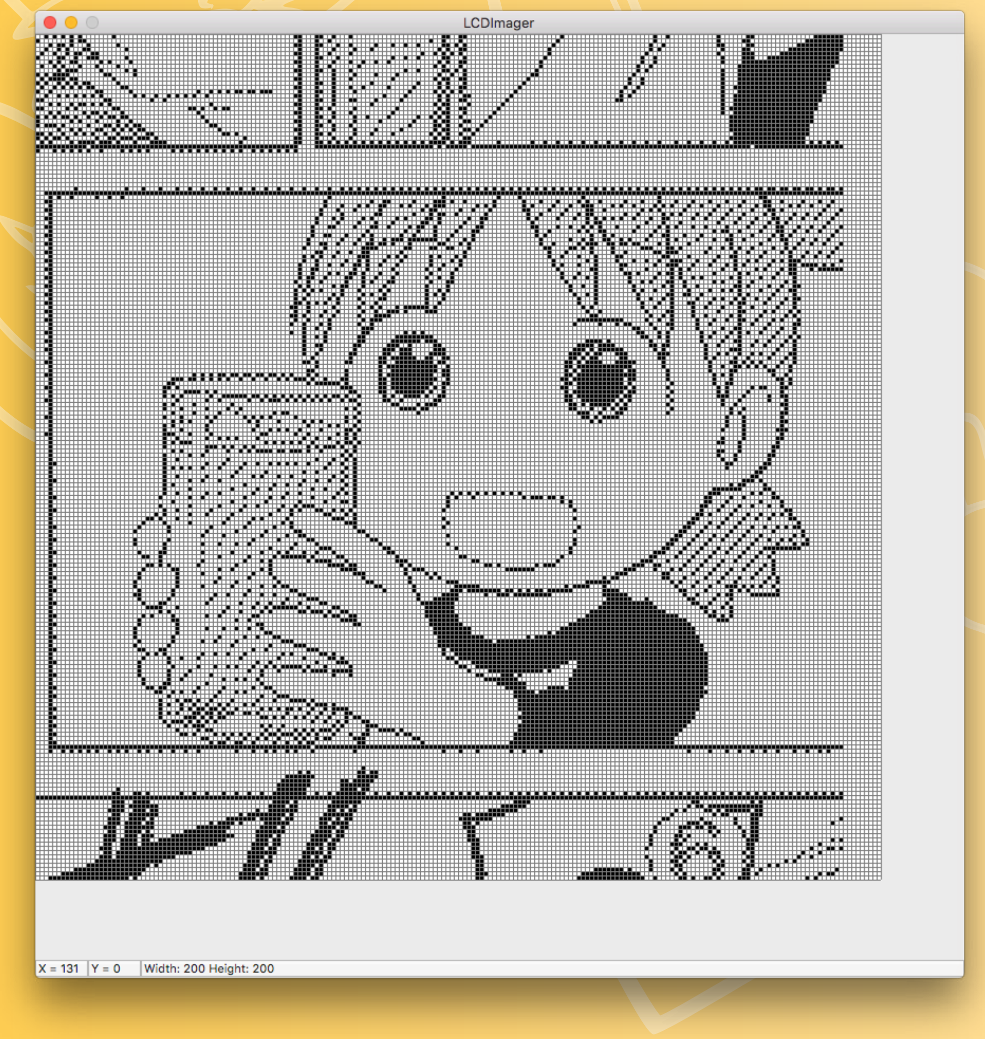
● Horizontal ○ Vertical

Set the image attributes to match the e-ink display at 200 x 200 pixels.

VERY IMPORTANT- set the Bitmap Orientation to HORIZONTAL (this is because the e-ink display writes bytes horizontally as opposed to the more common way of doing it vertically with regular graphic LCD displays.

Then we drag the corners of the application window until the entire dot matrix area representing the image is visible. We then select File -> Open and we open the bitmap we saved in Photoshop.

The program will then accurately display what can be expected to be seen on the e-ink display:



X = 131 | Y = 0 | Width: 200 Height: 200

Use File -> Save As to access the save menu. Give it a file name, and select the format as "C Source (*.c) and set the destination to a location OUTSIDE the project, preferably your home directory. This is done to prevent Code Composer Studio from trying to compile the code in the .c file which only contains the array and no headers, which leads to an (obvious) problem.

The program will display "Data Saved" when it is done. The program may then be closed, and the .c file you saved must now be opened in a suitable editor. Sublime Text is highly recommended for this task...

With the .c file open in Sublime text, select and copy (Command + C), EVERYTHING from the line starting with `const char Graphic1 []={` right until the closing brace }

In Code Composer Studio, near the top of main.c, find the declared arrays for the images, and comment out the image for Doraemon using `/* */`

Beneath the array declared for Sonic, paste the text copied from Sublime Text. Replace `const char` with `static const unsigned char` and replace `Graphic1` with `Yotsubato`.

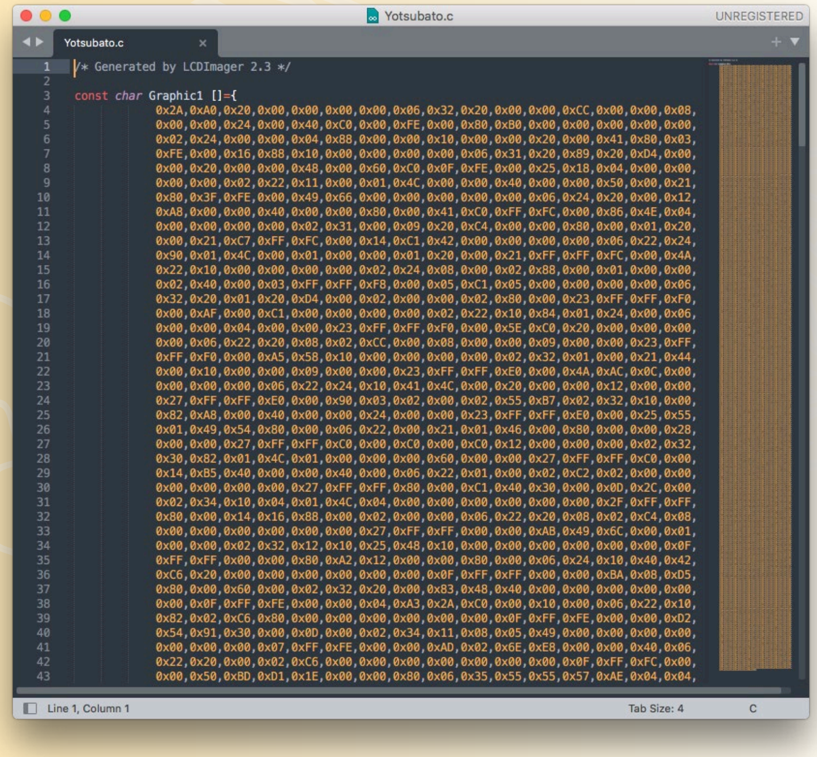
Locate the reference to the array in the `for(;;)` loop in main.c where:

```
case 0x00:
    LoadBitmap(Doraemon);
```

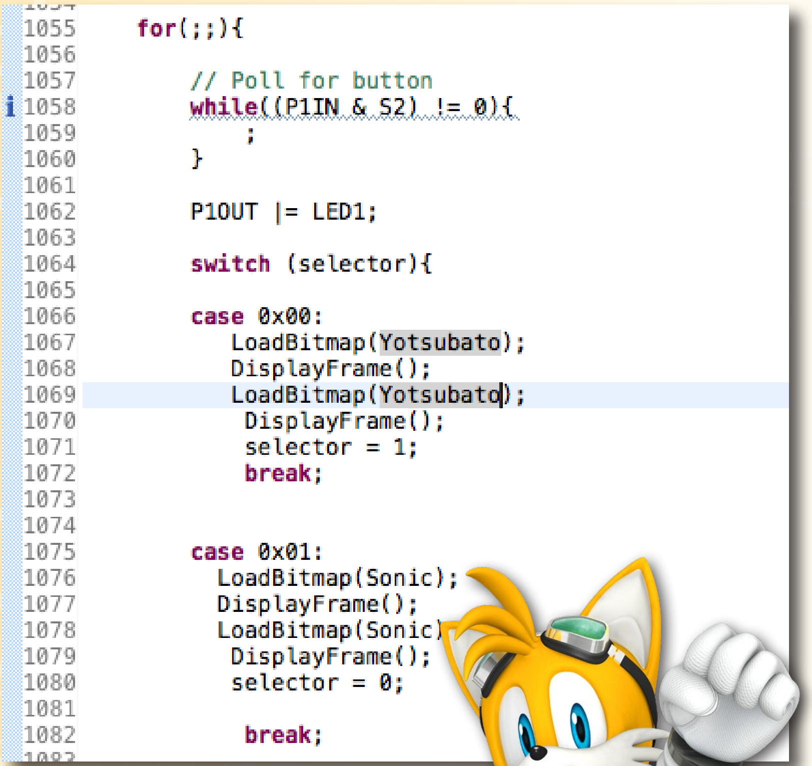
```
Change the reference to be
    LoadBitmap(Yotsubato);
```

The source code should look then as shown to the right. Click on Compile (hammer icon) and there should be no errors. Proceed to debug the project as before.

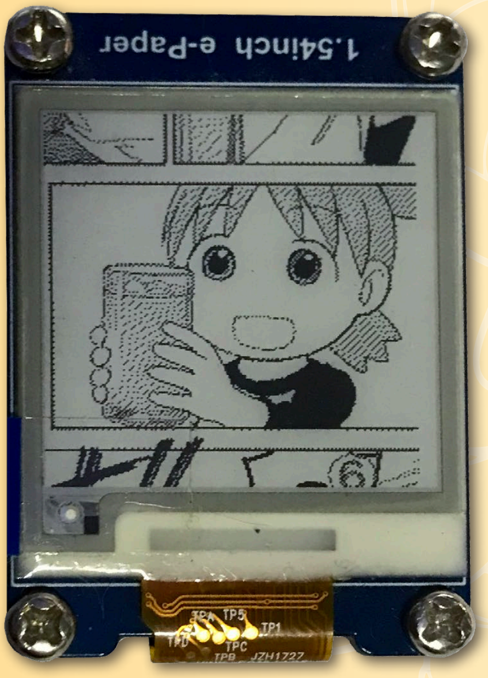
If everything went well, the first image loaded will no longer be that of Doraemon, but rather that of Yotsuba from the manga.



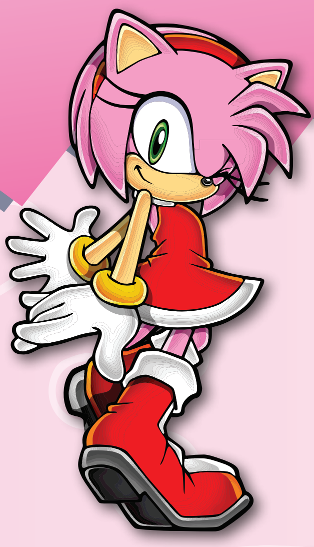
```
/* Generated by LCDImager 2.3 */

const char Graphic1 []={
			0x2A,0xA0,0x20,0x00,0x00,0x00,0x00,0x06,0x32,0x20,0x00,0x00,0xCC,0x00,0x00,0x08,
			0x00,0x00,0x24,0x00,0x40,0xC0,0x00,0xFE,0x00,0x80,0xB0,0x00,0x00,0x00,0x00,0x00,
			0x02,0x24,0x00,0x00,0x04,0x88,0x00,0x00,0x10,0x00,0x00,0x20,0x00,0x41,0x80,0x03,
			0xFE,0x00,0x16,0x88,0x10,0x00,0x00,0x00,0x00,0x06,0x31,0x20,0x89,0x20,0xD4,0x00,
			0x00,0x20,0x00,0x00,0x48,0x00,0x60,0xC0,0x0F,0xFE,0x00,0x25,0x18,0x04,0x00,0x00,
			0x00,0x00,0x02,0x22,0x11,0x00,0x01,0x4C,0x00,0x00,0x40,0x00,0x00,0x50,0x00,0x21,
			0x80,0x3F,0xFE,0x00,0x49,0x66,0x00,0x00,0x00,0x00,0x00,0x06,0x24,0x20,0x00,0x12,
			0xA8,0x00,0x00,0x40,0x00,0x00,0x80,0x00,0x41,0xC0,0xFF,0xFC,0x00,0x86,0x4E,0x04,
			0x00,0x00,0x00,0x00,0x02,0x31,0x00,0x09,0x20,0xC4,0x00,0x00,0x80,0x00,0x01,0x20,
			0x00,0x21,0xC7,0xFF,0xFC,0x00,0x14,0xC1,0x42,0x00,0x00,0x00,0x00,0x06,0x22,0x24,
			0x90,0x01,0x4C,0x00,0x01,0x00,0x00,0x01,0x20,0x00,0x21,0xFF,0xFF,0xFC,0x00,0x4A,
			0x22,0x10,0x00,0x00,0x00,0x00,0x02,0x24,0x08,0x00,0x02,0x88,0x00,0x01,0x00,0x00,
			0x02,0x40,0x00,0x03,0xFF,0xFF,0xF8,0x00,0x05,0xC1,0x05,0x00,0x00,0x00,0x00,0x06,
			0x32,0x20,0x01,0x20,0xD4,0x00,0x02,0x00,0x00,0x02,0x80,0x00,0x23,0xFF,0xFF,0xF0,
			0x00,0xAF,0x00,0xC1,0x00,0x00,0x00,0x00,0x02,0x22,0x10,0x84,0x01,0x24,0x00,0x06,
			0x00,0x00,0x04,0x00,0x00,0x23,0xFF,0xFF,0xF0,0x00,0x5E,0xC0,0x20,0x00,0x00,0x00,
			0x00,0x06,0x22,0x20,0x08,0x02,0xCC,0x00,0x08,0x00,0x00,0x09,0x00,0x00,0x23,0xFF,
			0xFF,0xF0,0x00,0xA5,0x58,0x10,0x00,0x00,0x00,0x00,0x02,0x32,0x01,0x00,0x21,0x44,
			0x00,0x10,0x00,0x00,0x09,0x00,0x00,0x23,0xFF,0xFF,0xE0,0x00,0x4A,0xAC,0x0C,0x00,
			0x00,0x00,0x00,0x06,0x22,0x24,0x10,0x41,0x4C,0x00,0x20,0x00,0x00,0x12,0x00,0x00,
			0x27,0xFF,0xFF,0xE0,0x00,0x90,0x03,0x02,0x00,0x02,0x55,0xB7,0x02,0x32,0x10,0x00,
			0x82,0xA8,0x00,0x40,0x00,0x00,0x24,0x00,0x00,0x23,0xFF,0xFF,0xE0,0x00,0x25,0x55,
			0x01,0x49,0x54,0x80,0x00,0x06,0x22,0x00,0x21,0x01,0x46,0x00,0x80,0x00,0x00,0x28,
			0x00,0x00,0x27,0xFF,0xFF,0xC0,0x00,0xC0,0x00,0xC0,0x12,0x00,0x00,0x00,0x02,0x32,
			0x30,0x82,0x01,0x4C,0x01,0x00,0x00,0x00,0x60,0x00,0x00,0x27,0xFF,0xFF,0xC0,0x00,
			0x14,0xB5,0x40,0x00,0x00,0x40,0x00,0x06,0x22,0x01,0x00,0x02,0xC2,0x02,0x00,0x00,
			0x00,0x00,0x00,0x00,0x27,0xFF,0xFF,0x80,0x00,0xC1,0x40,0x30,0x00,0x0D,0x2C,0x00,
			0x02,0x34,0x10,0x04,0x01,0x4C,0x04,0x00,0x00,0x00,0x00,0x00,0x00,0x2F,0xFF,0xFF,
			0x80,0x00,0x14,0x16,0x88,0x00,0x02,0x00,0x00,0x06,0x22,0x20,0x08,0x02,0xC4,0x08,
			0x00,0x00,0x00,0x00,0x00,0x00,0x27,0xFF,0xFF,0x00,0x00,0xAB,0x49,0x6C,0x00,0x01,
			0x00,0x00,0x02,0x32,0x12,0x10,0x25,0x48,0x10,0x00,0x00,0x00,0x00,0x00,0x00,0x0F,
			0xFF,0xFF,0x00,0x00,0x80,0xA2,0x12,0x00,0x00,0x80,0x00,0x06,0x24,0x10,0x40,0x42,
			0xC6,0x20,0x00,0x00,0x00,0x00,0x00,0x00,0x0F,0xFF,0xFF,0x00,0x00,0xBA,0x08,0xD5,
			0x80,0x00,0x60,0x00,0x02,0x32,0x20,0x00,0x83,0x48,0x40,0x00,0x00,0x00,0x00,0x00,
			0x00,0x0F,0xFF,0xFE,0x00,0x00,0x04,0xA3,0x2A,0xC0,0x00,0x10,0x00,0x06,0x22,0x10,
			0x82,0x02,0xC6,0x80,0x00,0x00,0x00,0x00,0x00,0x00,0x0F,0xFF,0xFE,0x00,0x00,0xD2,
			0x54,0x91,0x30,0x00,0x0D,0x00,0x02,0x34,0x11,0x08,0x05,0x49,0x00,0x00,0x00,0x00,
			0x00,0x00,0x00,0x07,0xFF,0xFE,0x00,0x00,0xAD,0x02,0x6E,0xE8,0x00,0x00,0x40,0x06,
			0x22,0x20,0x00,0x02,0xC6,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x0F,0xFF,0xFC,0x00,
			0x00,0x50,0xBD,0xD1,0x1E,0x00,0x00,0x80,0x06,0x35,0x55,0x55,0x57,0xAE,0x04,0x04,
```



```
    for(;;){

        // Poll for button
        while((P1IN & S2) != 0){
            ;
        }

        P1OUT |= LED1;

        switch (selector){

        case 0x00:
           LoadBitmap(Yotsubato);
           DisplayFrame();
           LoadBitmap(Yotsubato);
            DisplayFrame();
            selector = 1;
            break;


        case 0x01:
          LoadBitmap(Sonic);
          DisplayFrame();
          LoadBitmap(Sonic);
           DisplayFrame();
           selector = 0;

            break;
```



Yeaaaahhhh Sonic
We did it!
We made it work!!



Showcasing the best Sonic fan art

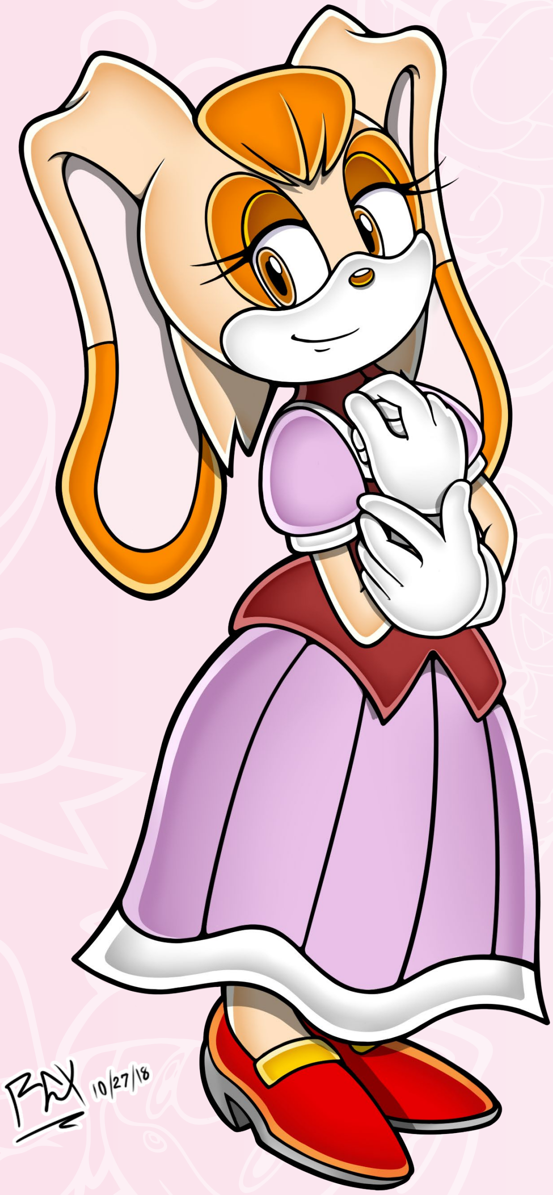
AMY'S ART CORNER

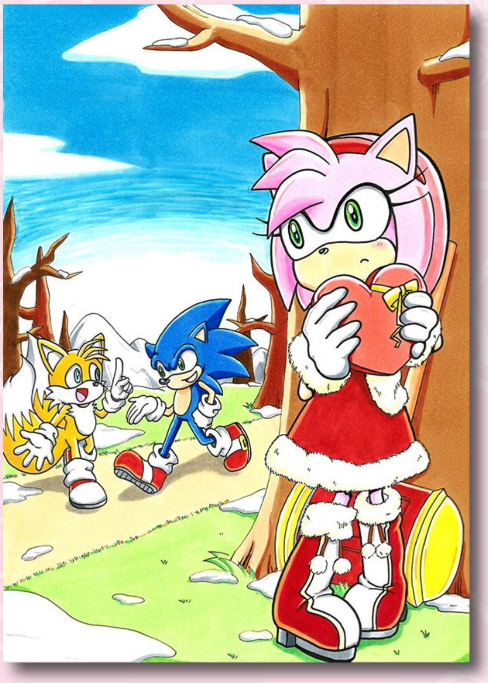
IS MY BOYFRIEND CUTE?

Each year I showcase the artworks created by Sonic fans all over the world

This is a place to give these artists a little love, so if you really like these, write me!

Enjoy :)
Amy xoxo





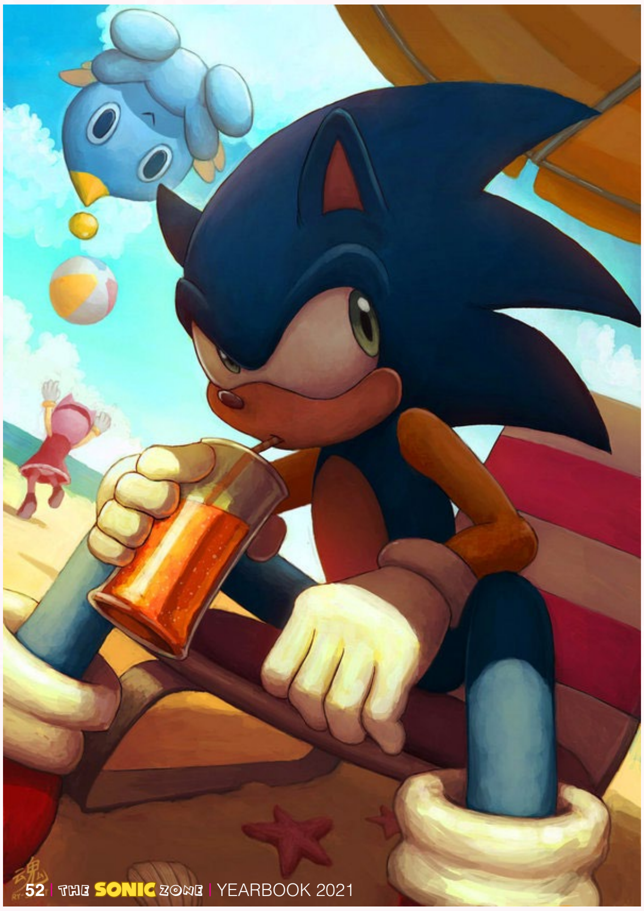
CLOCKWISE FROM RIGHT

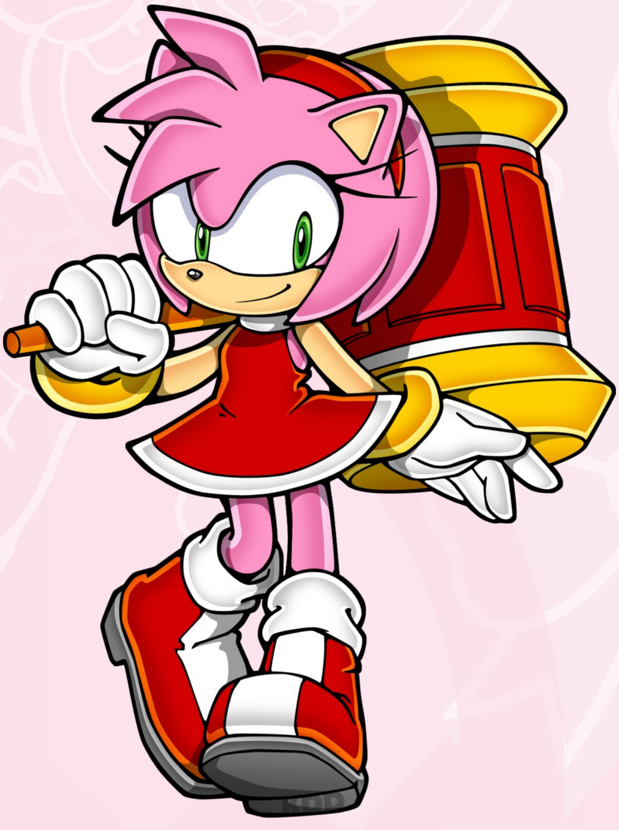
Amy's Valentine
Artist: Speedy
[Sonic Channel - sonic.sega.jp]

Vanilla The Rabbit
Artist: RGX

Who Dat? Amy Rose
Artist: ketrindarkdragon

Taking a Break
Artist: Ry-Spirit





1 - Artist: Jared
Origins Tattoo
Flint, MA, USA

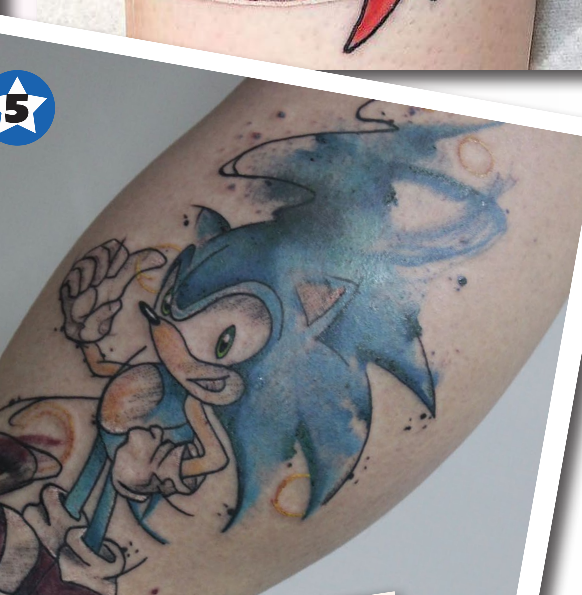
2 - Artist: Jose Lizemore
Hoodoo Tattoos
Cape Town, ZA

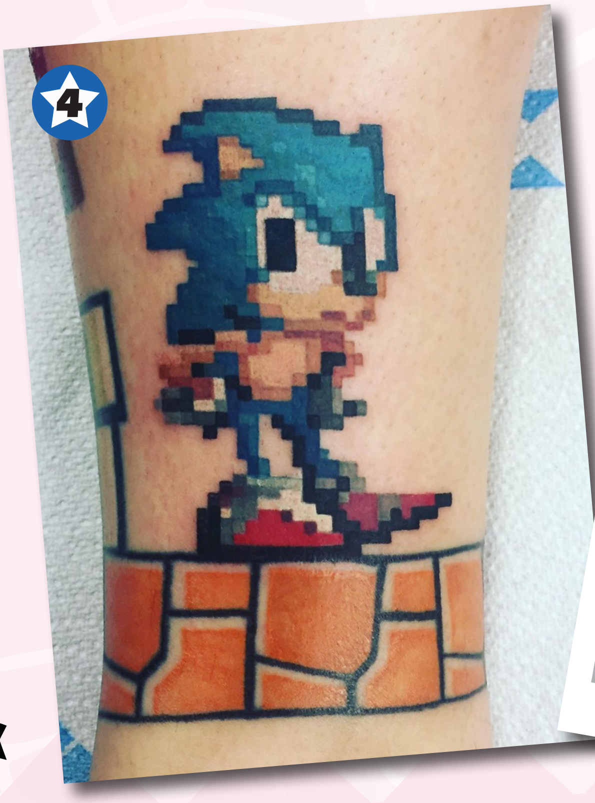
3 - Artist: Chris Hill
Dragon Tattoo Family
Belfast, Ireland

4 - Artist: Sabstar
Gun And Pedal
United Kingdom

5 - Artist: Amanda Barosso
Oca Tattoo
Brazil

6 - Artist: Yeray Perez
Barcelona, Spain





RESPECT ARTIST & CLIENT
TATTOO

WARNING
Criticism and/or critique about these works will not be tolerated! Respect people's reasons for being inked- its usually very personal! This is in line with standard tattoo etiquette which applies to the ink community at large. Reader discretion is advised!
Posting any of this content to Reddit or social media for the purposes of ridicule, or meme, or disparaging discourse is prohibited!

INTRUSION INTO EVERYDAY ITEMS

SHADOW'S SECRETS

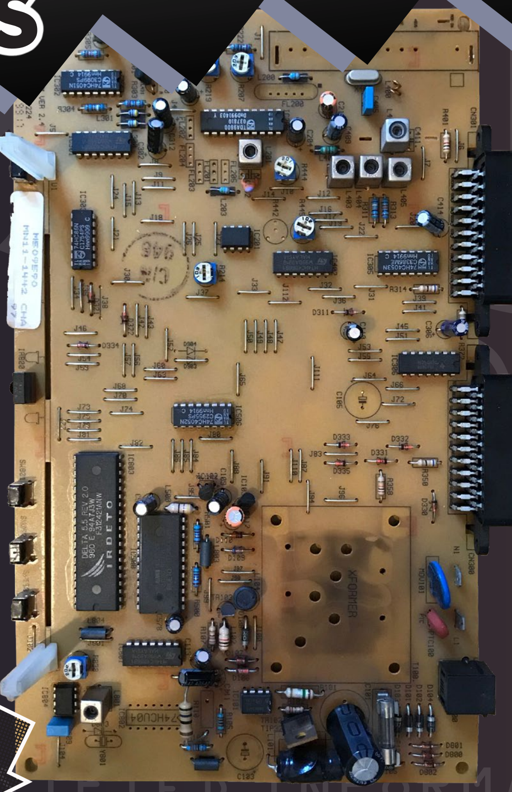
ANY USE FOR THIS KIND OF E-WASTE?

We have the remains here, of a Comcrypt 4000 set-top-box, used many years ago for analog Pay-TV.

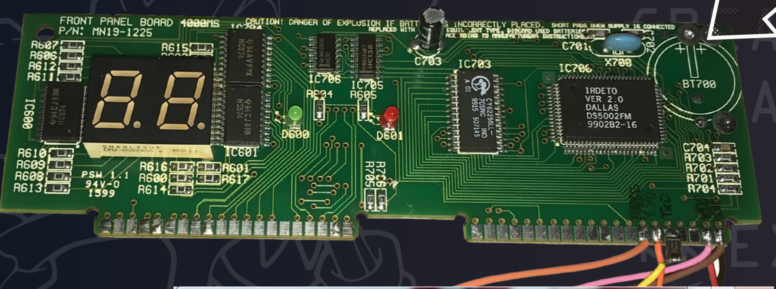
Most of the useful bits were removed by Tails for tinkering, so before the rest of it gets thrown in the bin, we need to evaluate if there's any use for the microprocessor on the display panel. It is makred as a DS5002, which we know is a 'soft' microcontroller a derivative of the Intel 8051- an industry standard device. The 'IRDETO' blurb is a concern though- probably means its impenetrable and therefore worthless.

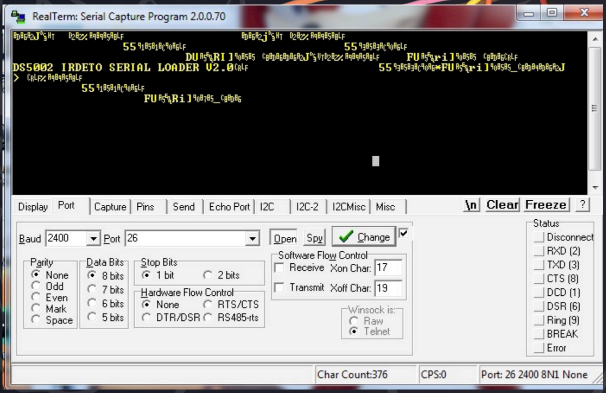
So, as most of us know, this microprocessor has a bootloader integrated into itself, which communicates with a computer on a serial port- the bootloader is used to load the application software and a few other functions.

So let's hook up an off-the-shelf USB-to-serial cable and see what we can do here...



We remove the front panel board, and attach wires from the usb-to-serial cable to the serial port on the MCU, the PROG pin is wired to a toggle switch (not shown) using info from DS5002 user guide. POWER IS ALSO SUPPLIED FROM USB (5V)





RealTerm: Serial Capture Program 2.0.0.70

```
BDBGA2J°SHT   D2B2% A9B9A5ABLF                      BDBGA2j°SHT  D2B2% A9B9B5ABLF
                 55 9185818C90A6LF                                55 9385838C90A6LF
                            DU A5%RI ]908585  CBBDB6BDB6A2J°SVTD2B2% A9B9A5ABLF      FU A5%ri ]908585  CBBDB6CRLF
DS5002  IRDETO  SERIAL  LOADER  V2.0CRLF                                    55 9385838C90A6*FU A5%ri ]908585_CBBDB4BDB6A2J
> CRLF% A9B9A5ABLF
            55 9185818C90A6LF
                      FU A5%Ri ]908785_CBBDB6
```

Display | Port | Capture | Pins | Send | Echo Port | I2C | I2C-2 | I2CMisc | Misc

Baud 2400 Port 26

Parity: ● None ○ Odd ○ Even ○ Mark ○ Space

Data Bits: ● 8 bits ○ 7 bits ○ 6 bits ○ 5 bits

Stop Bits: ● 1 bit ○ 2 bits

Hardware Flow Control: ● None ○ RTS/CTS ○ DTR/DSR ○ RS485-rts

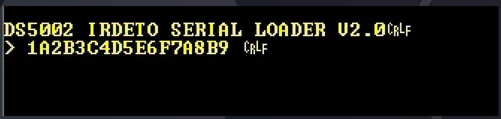
Software Flow Control: □ Receive Xon Char: 17 □ Transmit Xoff Char: 19

Winsock is: ○ Raw ● Telnet

Char Count:376 | CPS:0 | Port: 26 2400 8N1 None

By mucking around with the baudrate and toggling the PROG pin repeatedly we eventually get the loader.

Now before you get exicted and happy, don't bother because it does NOT respond to ANY commands- in fact it stops responding if you try and send any of the published loader commands. After some fuzzing I found that it expects an 8-byte crypto key in hexadecimal format:



```
DS5002  IRDETO  SERIAL  LOADER  V2.0CRLF
> 1A2B3C4D5E6F7A8B9 CRLF
```

So there you have it, IRDETO had their own loader custom made into that chip and it expects an unlocking key. Since we don't have the time or cash to expend on attacking the key, the whole board is worthless, and should be thrown away!

IRDETO- RESPECT MAH AUTHORITAH, ()*&#!

"We are on a mission. A mission to secure the world's digital platforms." - Irdeto website.

What a mission statement- kind of like saying everything out there is JUNK and needs fixing.

This company has a fine tradition of flexing its legal muscle to put itself above the law, particularly in markets (countries) where constitutions and people's rights are not particularly forthcoming. The stance- having people locked up for hacking their crap, its an attitude that began with their Dutch founder Pieter Den Toonder (even his own children hate him openly)!

As far as the science of crypto is concerned, they're not very good at it, as nearly all their products are repeatedly being compromised. Given that, it is not recommended that you attempt to hack their stuff. IRDETO is a particularly nasty company that will have you imprisioned or worse!

Their crappy DRM for games, called DENUVO is also used very close to home- Sonic Mania Plus [1] is a notable case where this rubbish leads to game performance issues.

FUN FACT 1- This divide-and-conquer mentality extends right across even into the parent conglomerate NASPERS- the same company that owns Media24, that in turn owns the much-hated Pay TV company Multichoice/DSTV.

FUN FACT 2- They cannot tolerate ANY form of criticism of their products and/or their company.

References:

1. https://www.extremetech.com/gaming/282924-denuvo-really-does-cripple-pc-gaming-performance

LEGAL DISCLAIMER - READ THIS FINE PRINT!

The ideas, concepts, techniques presented here are for illustrative / educational purposes only. Neither the publisher or author take any responsibility for your actions arising out of your use of this information. Do so at your own risk.

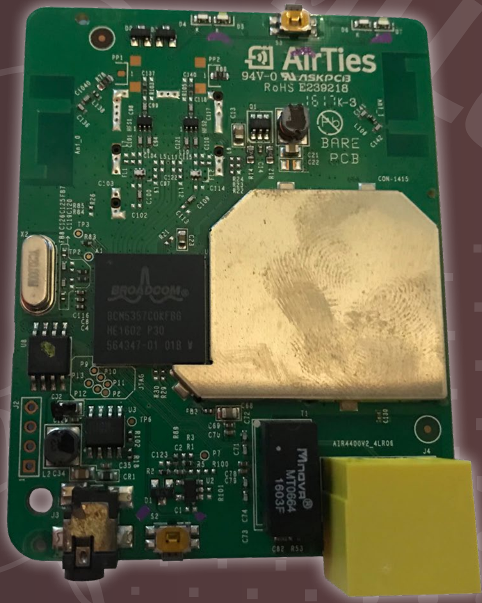
A DUMBED-DOWN WIRELESS EXTENDER

A few years ago you could buy the device shown on the right, that was used to connect a MultiChoice satelite set-top-box known as "Explora" to the internet for the purposes of video-on-demand. The assumption was that most of the morons who susbscribed to that service (you are such a moron if you're willing to pay that kind of price tag for Pay-TV) were not technically savvy enough to know that they could run an Ethernet cable from their DSL, FTTH or LTE router to their set-top-box. This was the solution.

Of course, if you look at the bottom you'll notice that its a "branded" version of an AirTies Air4400. And if the experience we had with the other STB was anything to go by, we know that we need to be wary if anything connected to IRDETO is branded in any way.

Now the reason why I say this piece of crap is dumbed-down is because, early in its life, MultiChoice became aware of this functionality and then gave the middle finger to those who bought these (they are sold below cost- the inkjet cartridge business model) and were using them as WiFi access points*. The issue is, this piece of crap fetches its firmware from a remote server, so you can't really prevent it from upgrading itself, as a few folks learnt.

Now, if you can't use it as a WiFi AP, what else can you do with it? As it turns out, not much at all. They were clever about this- even though the CPU in there is a standard Broadcom BCM5375CO, they paired it with a proprietary WiFi front-end chip. This means there will never be support for this unit in open source, neither in OpenWRT or DD-WRT.

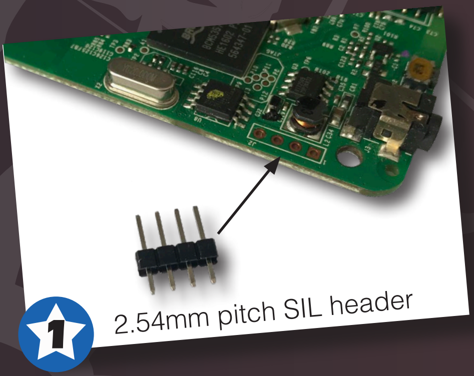


Now, that leaves one potential remaining use, to make this unit be a wireless Ethernet jack. While it is possible to configure it that way and it will work, the damn "calling home" for firmware updates means that, every few days it will lose all the settings you programmed into it. How annoying!

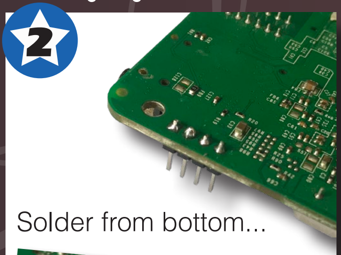
let’s try hack it already...

Instead of throwing these units in the bin, we can delve into what makes them work, and I took two approaches to this. The first was that I noticed a set of pads on the PCB. Tails will tell you immediately that's a serial port, so we go get the same cable we used to try and hack the DS5002 and see what we can find out.

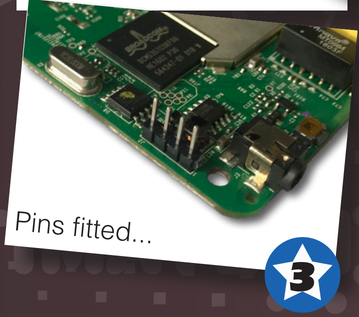
First thing we did was to solder a SIL header on those pads to allow us to easily plug a USB-to-serial cable in there. There are four pins, we only need three so we soldered in a 4 pin header anyways.



2.54mm pitch SIL header



Solder from bottom...



Pins fitted...

BEWARE OF STATIC

We had two of these units to play with and during the writing of this article, one unit inexplicably died. Tails had a look at it and its quite apparent that ESD (good old static electricity) blew the CPU. So be careful, heed the warning given to the right...

CAUTION- STATIC ELECTRICITY

These devices seem unusually prone to ESD (electrostatic discharge) damage. Indeed we lost the first unit we attempted hacking. It just stopped working and now all it does is draw large currents and gets hot.

Wear an earth strap or make sure you work on an anti-static surface

hacking tools required

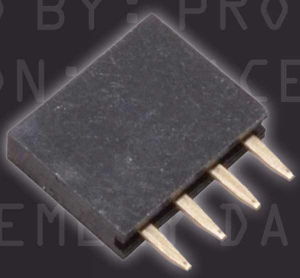
To gain access to the unit, you will need a bit of hardware and a bit of software. We used CoolTerm on macOS and a FTDI TTL-232R-3V3 cable (Windows users can use RealTerm and the same cable)

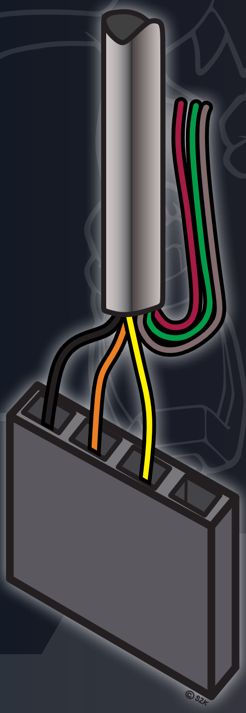
modding the cable

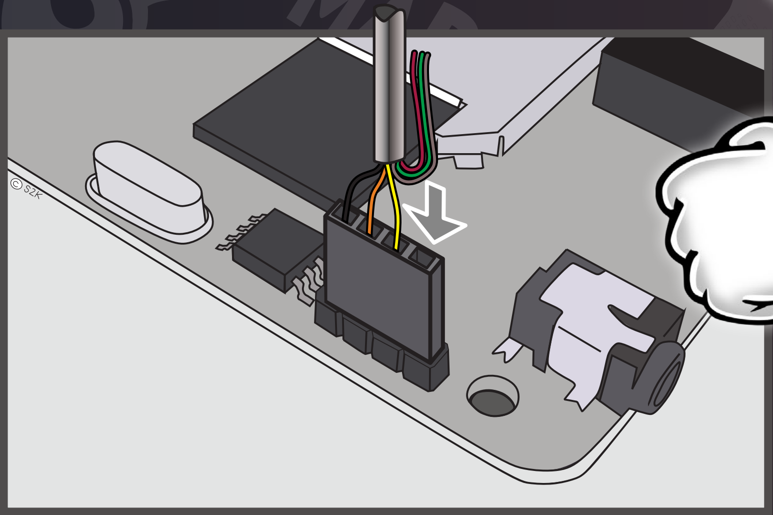
As can be seen in the photograph, the FTDI TTL-232R-3V3 is supplied with a single-in-line (SIL) header already attached.

We do not need most of those signals, we in fact only need three of the wires, black, yellow, and orange. So we cut the header off, separated the un-needed wires and broke out the three wires to a new 4-way header as shown.

FTDI TTL-232R-3V3

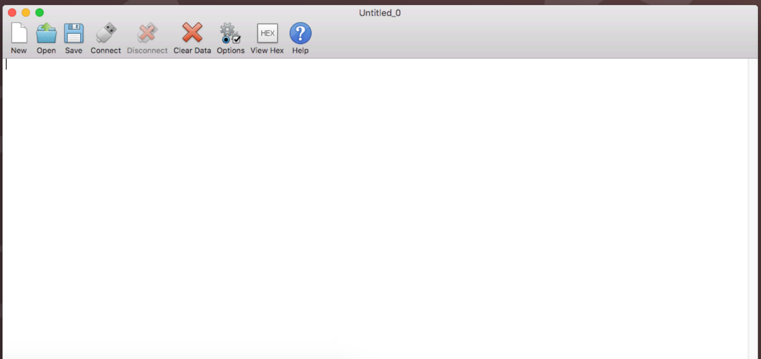

LOW COST FEMALE HEADER 1 x 4

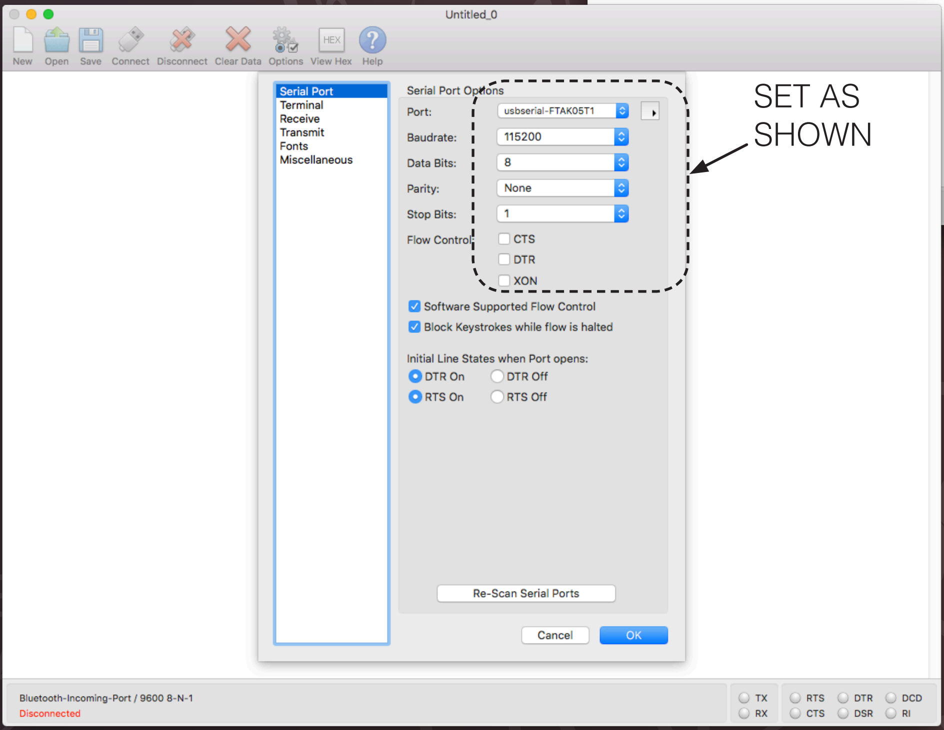

FTDI CABLE BROKEN OUT TO NEW HEADER


NEW HEADER PLUGGED IN

LET'S HACK

From past experience, these on-board serial ports on embedded networking devices are usually initialised to the highest baudrate, commonly 115200bps, and this unit is no exception, so we used CoolTerm on a Mac, with the FTDI cable connected and set it up as shown for 115200bps. Don't forget to click Connect...





Serial Port Options

Port: usbserial-FTAK05T1

Baudrate: 115200

Data Bits: 8

Parity: None

Stop Bits: 1

Flow Control: □ CTS □ DTR □ XON

☑ Software Supported Flow Control

☑ Block Keystrokes while flow is halted

Initial Line States when Port opens:

● DTR On ○ DTR Off

● RTS On ○ RTS Off

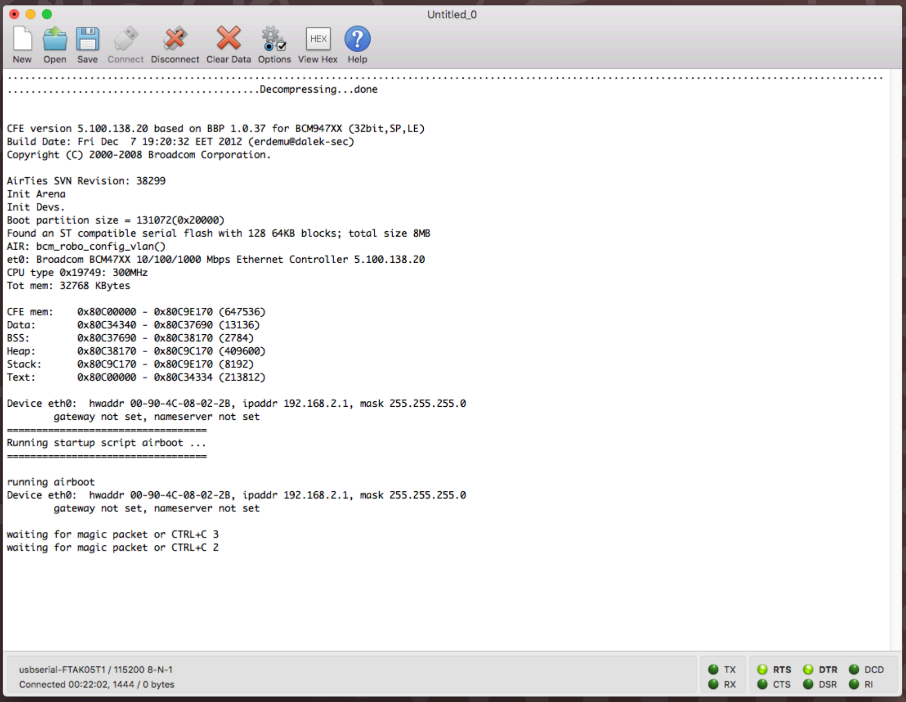
Bluetooth-Incoming-Port / 9600 8-N-1

Disconnected

SET AS SHOWN

WELL... IT'S A CONSOLE...

The moment we power that board up with it connected to CoolTerm, we see lots and lots of interesting stuff, ergo a serial console with active debug output. And here we see the first thing that happens... that's a bootloader, similar to u-boot used for embedded Linux systems...



```
...........................................................................................
............................................Decompressing...done


CFE version 5.100.138.20 based on BBP 1.0.37 for BCM947XX (32bit,SP,LE)
Build Date: Fri Dec  7 19:20:32 EET 2012 (erdemu@dalek-sec)
Copyright (C) 2000-2008 Broadcom Corporation.

AirTies SVN Revision: 38299
Init Arena
Init Devs.
Boot partition size = 131072(0x20000)
Found an ST compatible serial flash with 128 64KB blocks; total size 8MB
AIR: bcm_robo_config_vlan()
et0: Broadcom BCM47XX 10/100/1000 Mbps Ethernet Controller 5.100.138.20
CPU type 0x19749: 300MHz
Tot mem: 32768 KBytes

CFE mem:    0x80C00000 - 0x80C9E170 (647536)
Data:       0x80C34340 - 0x80C37690 (13136)
BSS:        0x80C37690 - 0x80C38170 (2784)
Heap:       0x80C38170 - 0x80C9C170 (409600)
Stack:      0x80C9C170 - 0x80C9E170 (8192)
Text:       0x80C00000 - 0x80C34334 (213812)

Device eth0:  hwaddr 00-90-4C-08-02-2B, ipaddr 192.168.2.1, mask 255.255.255.0
        gateway not set, nameserver not set
==================================
Running startup script airboot ...
==================================

running airboot
Device eth0:  hwaddr 00-90-4C-08-02-2B, ipaddr 192.168.2.1, mask 255.255.255.0
        gateway not set, nameserver not set

waiting for magic packet or CTRL+C 3
waiting for magic packet or CTRL+C 2
```

usbserial-FTAK05T1 / 115200 8-N-1

Connected 00:22:02, 1444 / 0 bytes

INSIDE INFORMATION...

bootloader pwnage...

As you will notice, it gives you a few seconds to either send a magic packet or hit CTRL+C when it boots, so we did the latter, and that gave us some interesting insight...

As with many Linux environments, entering '?' will often return the command list or syntax and this is no exception. Some pretty powerful stuff there, can read the FLASH memory, write it, set environment variables, set the network.

Jusr for fun we ran show devices, and boy, did we see some interesting stuff. It basically displayed the complete structire of the FLASH memory, enough information for any seasoned Linux developer to know precisely what is happening there.

Looking at it quickly, it is clear there's a main image, and a backup image. This makes it pretty useful in that should something go horribly wrong you can recover the unit. It also likely comes into play with firmware upgrades...

Note that this is standard Broadcom fare, and unfortunately at this time I do not have access to those developer tools. If and when the situation changes we will announce it on our site or in the next yearbook!

let it boot, then look around

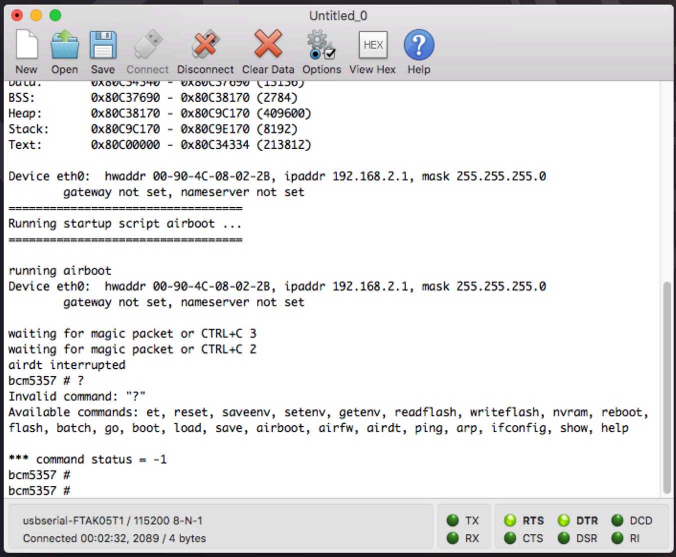
So we turned it off and back on again (unplug and replug the power cable) and we allowed it to do its thing...

When the OS has booted and its internal webserver (this device uses httpd) has started up, you will notice the LED turns white from yellow. This means the unit is operational. You will then see it tries to do two things:

- upgrade the firmware i.e. call home
- get the current time, by doing requests to NTP servers. Of course it will try MonoChoice's MWEB NTP server first, and it will do this for some time, eventually trying even the well known pool.ntp.org

Because we've not connected it to any network, all these NTP requests fail and you will see it keep trying, periodically, sometimes using ntp-ndf.mweb.co.za and pool.ntp.org.

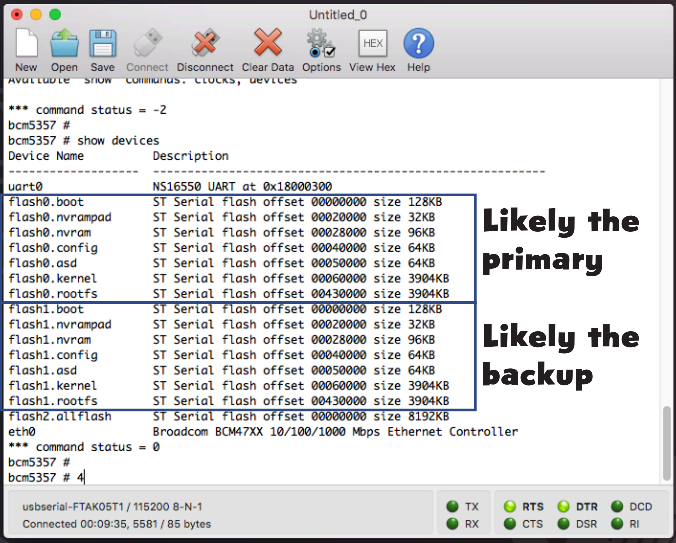
So we do just that, we plug the Ethernet cable into the thing (on the same network as our computer of course)...



```
Data:        0x80C34340 - 0x80C37690 (13136)
BSS:         0x80C37690 - 0x80C38170 (2784)
Heap:        0x80C38170 - 0x80C9C170 (409600)
Stack:       0x80C9C170 - 0x80C9E170 (8192)
Text:        0x80C00000 - 0x80C34334 (213812)

Device eth0:  hwaddr 00-90-4C-08-02-2B, ipaddr 192.168.2.1, mask 255.255.255.0
        gateway not set, nameserver not set
==================================
Running startup script airboot ...
==================================

running airboot
Device eth0:  hwaddr 00-90-4C-08-02-2B, ipaddr 192.168.2.1, mask 255.255.255.0
        gateway not set, nameserver not set

waiting for magic packet or CTRL+C 3
waiting for magic packet or CTRL+C 2
airdt interrupted
bcm5357 # ?
Invalid command: "?"
Available commands: et, reset, saveenv, setenv, getenv, readflash, writeflash, nvram, reboot,
flash, batch, go, boot, load, save, airboot, airfw, airdt, ping, arp, ifconfig, show, help

*** command status = -1
bcm5357 #
bcm5357 #
```

usbserial-FTAK05T1 / 115200 8-N-1
Connected 00:02:32, 2089 / 4 bytes

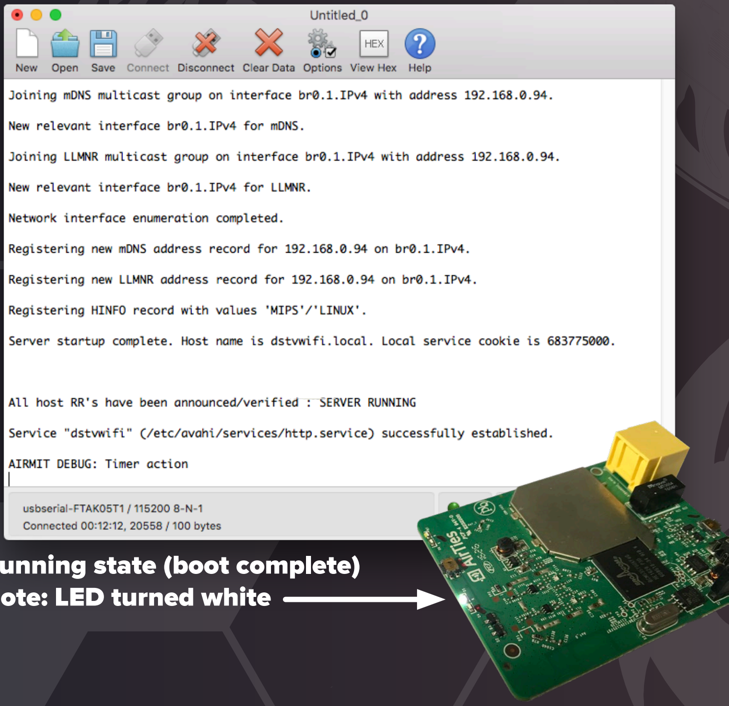
Entry into loader via CTRL+C



```
Available show commands: clocks, devices

*** command status = -2
bcm5357 #
bcm5357 # show devices
Device Name          Description
-------------------  ---------------------------------------------------------------
uart0                NS16550 UART at 0x18000300
flash0.boot          ST Serial flash offset 00000000 size 128KB
flash0.nvrampad      ST Serial flash offset 00020000 size 32KB
flash0.nvram         ST Serial flash offset 00028000 size 96KB
flash0.config        ST Serial flash offset 00040000 size 64KB
flash0.asd           ST Serial flash offset 00050000 size 64KB
flash0.kernel        ST Serial flash offset 00060000 size 3904KB
flash0.rootfs        ST Serial flash offset 00430000 size 3904KB
flash1.boot          ST Serial flash offset 00000000 size 128KB
flash1.nvrampad      ST Serial flash offset 00020000 size 32KB
flash1.nvram         ST Serial flash offset 00028000 size 96KB
flash1.config        ST Serial flash offset 00040000 size 64KB
flash1.asd           ST Serial flash offset 00050000 size 64KB
flash1.kernel        ST Serial flash offset 00060000 size 3904KB
flash1.rootfs        ST Serial flash offset 00430000 size 3904KB
flash2.allflash      ST Serial flash offset 00000000 size 8192KB
eth0                 Broadcom BCM47XX 10/100/1000 Mbps Ethernet Controller
*** command status = 0
bcm5357 #
bcm5357 # 4
```

usbserial-FTAK05T1 / 115200 8-N-1
Connected 00:09:35, 5581 / 85 bytes

show devices COMMAND output



```
Joining mDNS multicast group on interface br0.1.IPv4 with address 192.168.0.94.

New relevant interface br0.1.IPv4 for mDNS.

Joining LLMNR multicast group on interface br0.1.IPv4 with address 192.168.0.94.

New relevant interface br0.1.IPv4 for LLMNR.

Network interface enumeration completed.

Registering new mDNS address record for 192.168.0.94 on br0.1.IPv4.

Registering new LLMNR address record for 192.168.0.94 on br0.1.IPv4.

Registering HINFO record with values 'MIPS'/'LINUX'.

Server startup complete. Host name is dstvwifi.local. Local service cookie is 683775000.



All host RR's have been announced/verified : SERVER RUNNING

Service "dstvwifi" (/etc/avahi/services/http.service) successfully established.

AIRMIT DEBUG: Timer action
```

usbserial-FTAK05T1 / 115200 8-N-1
Connected 00:12:12, 20558 / 100 bytes

Running state (boot complete)
Note: LED turned white →

Connecting to the internal Web Server

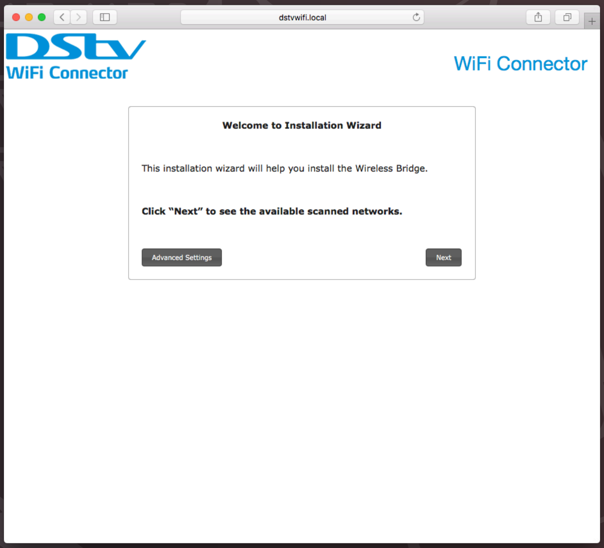
By default, the IP address of this device defaults to 192.168.2.254 as is revealed using the `ifconfig` command. If your computer happens to be on a similar network, that's all good but most often your network will be in the 192.168.0.x or 10.0.0.x range, so to get around that we need to change the IP address. To do that simply enter the command:

```
ifconfig br0.1 192.168.0.94 netmask 
255.255.255.0 up
```

(replace IP and netmask to suit your network).

Now the device will be on the same network as your computer and you can confirm this by pinging the IP address and you should see the responses come back...

Opening up a web browser and entering the URL to `dstvwifi.local` should load the internal web server's page as shown:



Welcome to Installation Wizard

This installation wizard will help you install the Wireless Bridge.

Click "Next" to see the available scanned networks.

making it easier

When that Advanced Settings button is displayed, click it and find the LAN settings, turn off the DHCP server and give the device a static IP on your network so that you don't have to repeatedly execute the ifconfig command every time you hack/reboot the device...

oh that embedded web server?

Didn't forget about that. When the unit is booted, simply hitting the ENTER key returns the command prompt `'#'` when we do a directory listing with the UNIX `ls` command... we see a typical Linux filesystem but that `webs.tar.lzma` file... hmm that's interesting :) To get it off the device we copied that file to the root directory as used by the embedded web server and then downloaded it with wget in a terminal session. Then we unzipped it on the computer. To save you all the effort, you can find our copy of it here:

https://github.com/TheSonicZone/dstvwifi-connector-airties/

```
# ls
bin            license        root           usr            webs-guest
dev            mnt            sbin           var            webs.tar.lzma
etc            proc           sys            webs
lib            ramdisk        tmp            webs-admin
```

HOW WE DID IT

We confirmed the 'webs' directory was actually what was being served to clients on the network accessing the device, so we copied webs.tar.lzma to that directory and downloaded it with `wget` on a Linux or Mac computer (on the Macintosh you will need to install wget in HomeBrew beforehand)
So on the command line of the device i.e. CoolTerm we enter:

`cp webs.tar.lzma /webs` *[This copies that archive to the folder being served to the web browser]*

And then in a terminal on the computer (Linux/macOS):

```
wget http://192.168.0.94/webs.tar.lzma
```

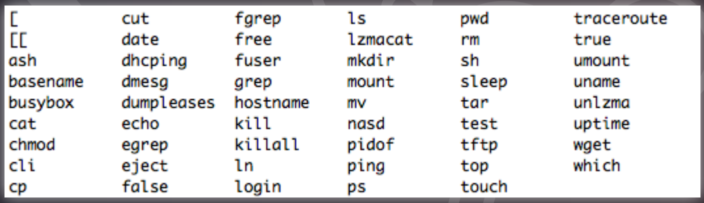
And the file is downloaded into your home directory, where you can unzip it using

```
tar -xvf webs.tar.lzma
```

INSIDE INFORMATION...

playing with the hardware...

Since this is a Linux system, we have freedom to do stuff, like fiddle with the GPIO or devices. So if we look around at what kinds of software is present on the device we find that we have no text editors at all, neither vi or nano. We look in the bin directory and see this is a very minimalist machine indeed... Could be worse, at least we have wget!

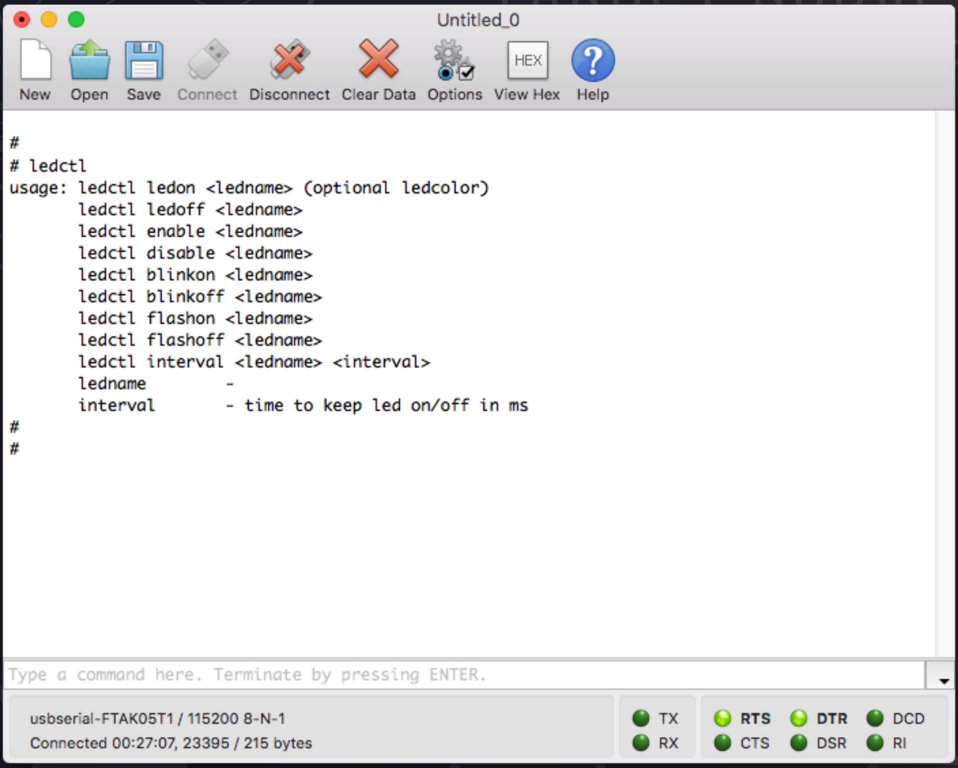


```
[          cut        fgrep      ls         pwd        traceroute
[[         date       free       lzmacat    rm         true
ash        dhcping    fuser      mkdir      sh         umount
basename   dmesg      grep       mount      sleep      uname
busybox    dumpleases hostname   mv         tar        unlzma
cat        echo       kill       nasd       test       uptime
chmod      egrep      killall    pidof      tftp       wget
cli        eject      ln         ping       top        which
cp         false      login      ps         touch
```

So we browse around, particularly in directories such as /usr/bin and we see some interesting executables there. These are of course the very programs that perform the functions to make this a MonoChoice WiFi connector, for example command line programs such as:

`http_upgrade`	[presumably does the call home]
`defaults`	[presumably writes factory defaults]
`ledctl`	[presumably controls LEDs]

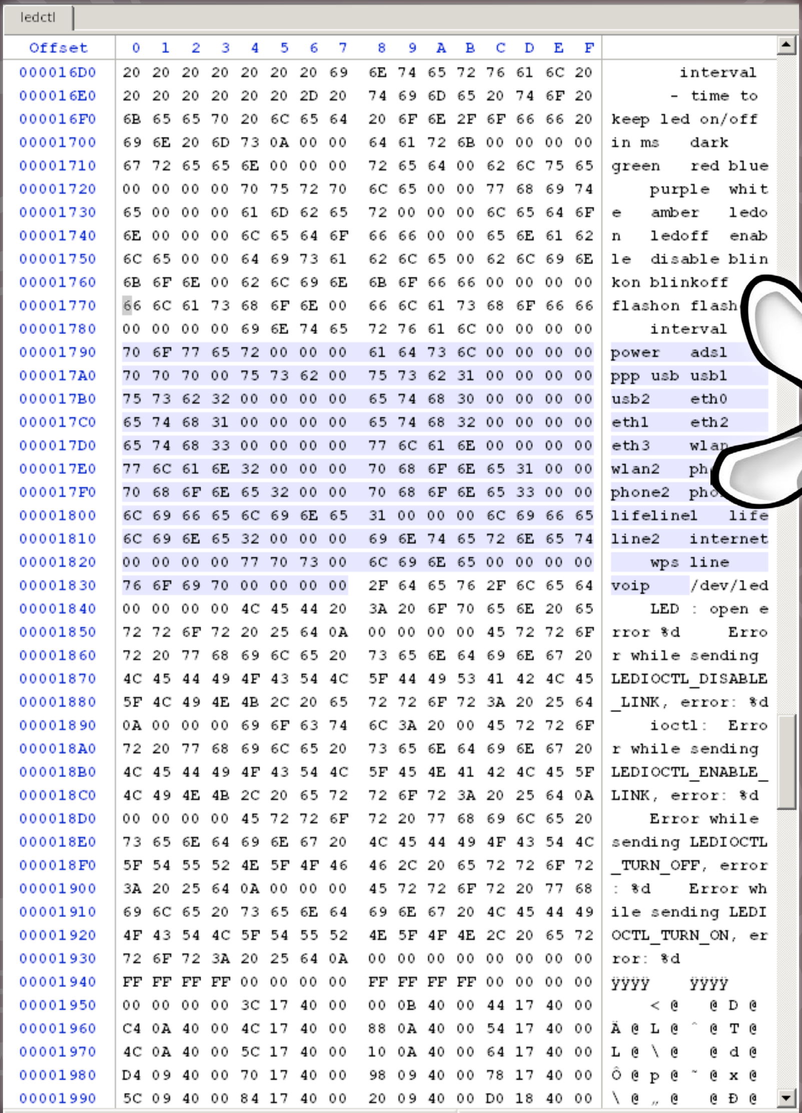
The CLI program ledctl intrigues us most.. so we try and run the thing and see what happens... and aren't we surprised, we've stumbled upon the little program that controls the LEDs on the board.



```
#
# ledctl
usage: ledctl ledon <ledname> (optional ledcolor)
       ledctl ledoff <ledname>
       ledctl enable <ledname>
       ledctl disable <ledname>
       ledctl blinkon <ledname>
       ledctl blinkoff <ledname>
       ledctl flashon <ledname>
       ledctl flashoff <ledname>
       ledctl interval <ledname> <interval>
       ledname         -
       interval        - time to keep led on/off in ms
#
#
```

usbserial-FTAK05T1 / 115200 8-N-1
Connected 00:27:07, 23395 / 215 bytes

The issue now is, we have no idea how to formulate the arguments for that little application to make it do stuff. We tried all kinds of command arguments such as wifiled and power_led and it does nothing. So what we did was we copied the executable off the device to a desktop computer and did a hex dump on the file to see if we could find what arguments it expects. This is hacking in its purest form- When in doubt, don't under-estimate the power of the hex-dump. And indeed we did, find what we were looking for, by opening the ledctl executable in a program called WinHex (run in PlayOnMac)...



```
Offset     0  1  2  3  4  5  6  7   8  9  A  B  C  D  E  F
000016D0  20 20 20 20 20 20 20 69  6E 74 65 72 76 61 6C 20         interval
000016E0  20 20 20 20 20 20 2D 20  74 69 6D 65 20 74 6F 20        - time to
000016F0  6B 65 65 70 20 6C 65 64  20 6F 6E 2F 6F 66 66 20  keep led on/off
00001700  69 6E 20 6D 73 0A 00 00  64 61 72 6B 00 00 00 00  in ms   dark
00001710  67 72 65 65 6E 00 00 00  72 65 64 00 62 6C 75 65  green   red blue
00001720  00 00 00 00 70 75 72 70  6C 65 00 00 77 68 69 74      purple  whit
00001730  65 00 00 00 61 6D 62 65  72 00 00 00 6C 65 64 6F  e   amber   ledo
00001740  6E 00 00 00 6C 65 64 6F  66 66 00 00 65 6E 61 62  n   ledoff  enab
00001750  6C 65 00 00 64 69 73 61  62 6C 65 00 62 6C 69 6E  le  disable blin
00001760  6B 6F 6E 00 62 6C 69 6E  6B 6F 66 66 00 00 00 00  kon blinkoff
00001770  66 6C 61 73 68 6F 6E 00  66 6C 61 73 68 6F 66 66  flashon flash
00001780  00 00 00 00 69 6E 74 65  72 76 61 6C 00 00 00 00      interval
00001790  70 6F 77 65 72 00 00 00  61 64 73 6C 00 00 00 00  power   adsl
000017A0  70 70 70 00 75 73 62 00  75 73 62 31 00 00 00 00  ppp usb usb1
000017B0  75 73 62 32 00 00 00 00  65 74 68 30 00 00 00 00  usb2    eth0
000017C0  65 74 68 31 00 00 00 00  65 74 68 32 00 00 00 00  eth1    eth2
000017D0  65 74 68 33 00 00 00 00  77 6C 61 6E 00 00 00 00  eth3    wlan
000017E0  77 6C 61 6E 32 00 00 00  70 68 6F 6E 65 31 00 00  wlan2   ph
000017F0  70 68 6F 6E 65 32 00 00  70 68 6F 6E 65 33 00 00  phone2  pho
00001800  6C 69 66 65 6C 69 6E 65  31 00 00 00 6C 69 66 65  lifeline1   life
00001810  6C 69 6E 65 32 00 00 00  69 6E 74 65 72 6E 65 74  line2   internet
00001820  00 00 00 00 77 70 73 00  6C 69 6E 65 00 00 00 00      wps line
00001830  76 6F 69 70 00 00 00 00  2F 64 65 76 2F 6C 65 64  voip    /dev/led
00001840  00 00 00 00 4C 45 44 20  3A 20 6F 70 65 6E 20 65      LED : open e
00001850  72 72 6F 72 20 25 64 0A  00 00 00 00 45 72 72 6F  rror %d     Erro
00001860  72 20 77 68 69 6C 65 20  73 65 6E 64 69 6E 67 20  r while sending
00001870  4C 45 44 49 4F 43 54 4C  5F 44 49 53 41 42 4C 45  LEDIOCTL_DISABLE
00001880  5F 4C 49 4E 4B 2C 20 65  72 72 6F 72 3A 20 25 64  _LINK, error: %d
00001890  0A 00 00 00 69 6F 63 74  6C 3A 20 00 45 72 72 6F      ioctl:  Erro
000018A0  72 20 77 68 69 6C 65 20  73 65 6E 64 69 6E 67 20  r while sending
000018B0  4C 45 44 49 4F 43 54 4C  5F 45 4E 41 42 4C 45 5F  LEDIOCTL_ENABLE_
000018C0  4C 49 4E 4B 2C 20 65 72  72 6F 72 3A 20 25 64 0A  LINK, error: %d
000018D0  00 00 00 00 45 72 72 6F  72 20 77 68 69 6C 65 20      Error while
000018E0  73 65 6E 64 69 6E 67 20  4C 45 44 49 4F 43 54 4C  sending LEDIOCTL
000018F0  5F 54 55 52 4E 5F 4F 46  46 2C 20 65 72 72 6F 72  _TURN_OFF, error
00001900  3A 20 25 64 0A 00 00 00  45 72 72 6F 72 20 77 68  : %d    Error wh
00001910  69 6C 65 20 73 65 6E 64  69 6E 67 20 4C 45 44 49  ile sending LEDI
00001920  4F 43 54 4C 5F 54 55 52  4E 5F 4F 4E 2C 20 65 72  OCTL_TURN_ON, er
00001930  72 6F 72 3A 20 25 64 0A  00 00 00 00 00 00 00 00  ror: %d
00001940  FF FF FF FF 00 00 00 00  FF FF FF FF 00 00 00 00  ÿÿÿÿ    ÿÿÿÿ
00001950  00 00 00 00 3C 17 40 00  00 0B 40 00 44 17 40 00      < @   @ D @
00001960  C4 0A 40 00 4C 17 40 00  88 0A 40 00 54 17 40 00  Ä @ L @ ˆ @ T @
00001970  4C 0A 40 00 5C 17 40 00  10 0A 40 00 64 17 40 00  L @ \ @   @ d @
00001980  D4 09 40 00 70 17 40 00  98 09 40 00 78 17 40 00  Ô @ p @ ˜ @ x @
00001990  5C 09 40 00 84 17 40 00  20 09 40 00 D0 18 40 00  \ @ „ @   @ Ð @
```

HEX DUMP OF `ledctl` binary

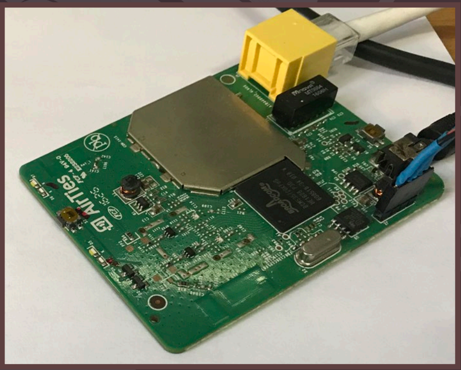
not hard to find...

And with simply browsing the executable in WinHEX we find the names of the LEDs in question (highlighted) and its clear this small application is intended for DSL routers and gateways too..

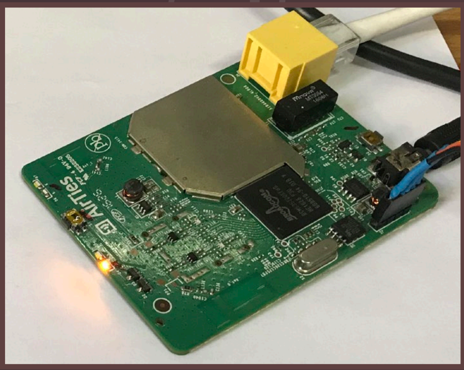
From the above we have numerous LED names, one potentially of use, called `power`. So we compose a command as:

```
ledctl ledoff power
```

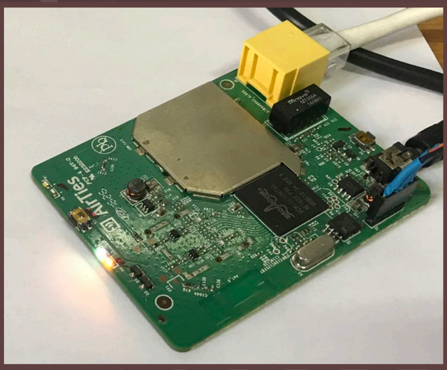
And off goes the WHITE LED on the board. SUCCESS! We have pwned the LEDs on this thing!!
Below are example commands and what happens when executed. Have fun! (Expect more discoveries in next year's yearbook when we get more aggressive with this thing, hopefully we don't kill it!)



`ledctl ledoff power`



`ledctl ledon power`



`ledctl ledon power white`

CREAM & CHEESE TREATS

YUM CAKE

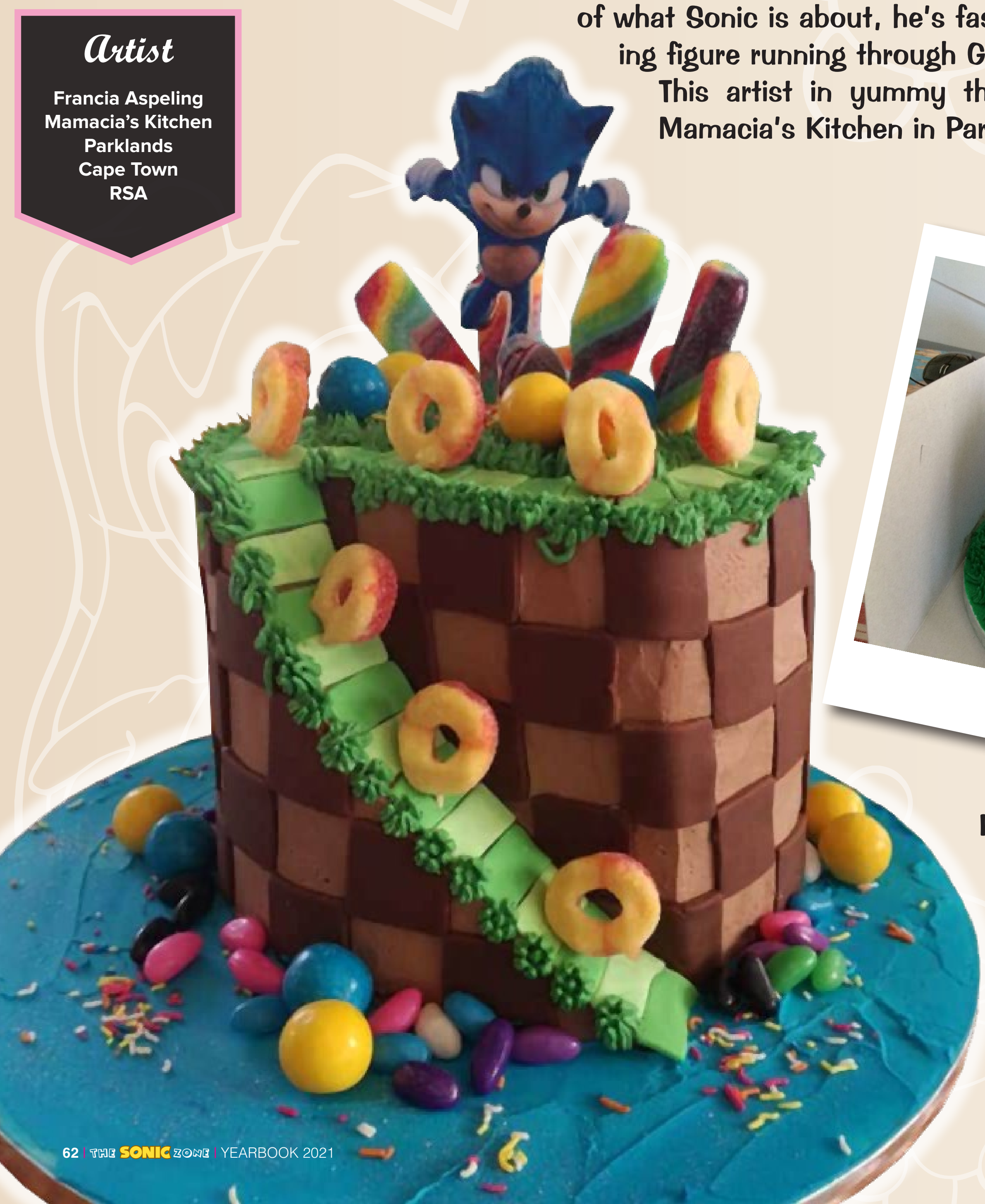
WHO BAKES THESE AWESOME SONIC CAKES?

The cake featured below, was a birthday cake for someone special. The recipient really enjoyed the Sonic The Hedgehog movie and wanted a Sonic cake for his birthday.

This particular cake is what we would call setting the bar pretty high. I've seen a quite a few Sonic cakes from around the world but these cakes are on a next level- they taste yummy too. This cake truly captures the spirit of what Sonic is about, he's fast, and he cuts a dashing figure running through Green Hill Zone. This artist in yummy things, is the owner of Mamacia's Kitchen in Parklands, Cape Town.

Artist

Francia Aspeling
Mamacia's Kitchen
Parklands
Cape Town
RSA



In 2019 Mamacia baked an epic cake for Sonikku's birthday, one that will always be fondly remembered (see photo above). Yes that is a Nintendo Amiibo!

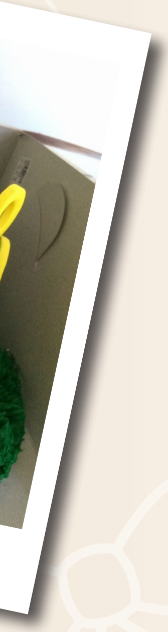
CHILLI DOGS THE WAY SONIC LIKES THEM!

Imagine that, getting to actually make and taste Sonic's favourite meal. Here is a recipe for the real thing...

Stuff you will need:

- Beef Mince (225g)
- Pressed Garlic (1 clove)
- Chilli Powder (2 tsp)
- Tomato Purèe (250g)
- Cinnamon (1/4 tsp)
- Mustard (1 tbsp)
- Brown Sugar (2 tsp)
- Worcestershire Sauce (1 tsp)
- Frankfurters (8)
- Bread Rolls (8)
- Cheddar Cheese (grated)
- Red Onion (chopped)

Photo Credit: Associated Magazines Publishing



Preparation

Prepare a barbecue (charcoal or wood) in the usual manner.

Cook the mince in a medium frying pan on medium heat on a stovetop ensuring it breaks up into tiny pieces. Stir in the pressed garlic, the chilli powder, and the cinnamon and then cook for 1 minute. Add the tomato purèe, mustard, brown sugar and Worcestershire sauce and simmer until thickened.

Grill the 8 frankfurters on the barbecue and place them in the rolls once cooked.

Spoon the chilli on top and sprinkle with grated cheddar and chopped red onion. The onion is optional but adds flavour. As you add the cheese you will notice the magic happen, you will have a real Sonic chilli-dog before you.

Enjoy, this recipe serves 8 friends, enough for everyone including Doctor Eggman. Not sure if he likes hot stuff?

3 DAY
DREAMCON
Attendee
1 2 3
CHAO IN SPACE
EVAN

THE GANG GOING TO DREAMCON

This artwork was created for COVER RE (Convention Edition) of IDW's Sonic The Hedgehog #27

INSPIRATIONAL ART

"DreamCon "- A blast from the Dreamcast Past

Recently I felt that I really miss the Dreamcast era. There was something magical about that time- those two years from late 1999 through to the day when SEGA dropped the bombshell on us in February 2001. Its very hard to explain those years which is why this project exists in the first instance.

Given that I've seen people on the internet do renders and drawings of what they believe a successor console should look like, none of those designs are practical, neither from any form of manufacturing reality or actual engineering standpoint.

This got me thinking for some time, but I really got thinking when I saw the artwork I take pride in featuring here- The little blue Dreamcast Swirl on the con-badges did it!

Many gamers today are unaware of the legacy, or certainly the impact of the SEGA Dreamcast, and take the features of modern consoles for granted- unaware that those very features were Dreamcast innovations. Online gameplay, the controller design, built-in modem...

Indeed at the time, I chose to invest heavily in two units, and not a desktop PC because I knew that nothing, not even the most expensive desktop PC could compare to the graphics performance and overall UX of that small little off-white console. To be very honest I only ever got a desktop PC of comparable performance years later when the costs had dropped sufficiently.

Because I know SEGA won't (and are not in a position to) launch a successor, I feel capable of doing it. I always wanted to create something like this- to leave my own little bit of legacy in the Sonic fandom. It has, admittedly taken me 20 years to acquire the necessary engineering and programming skills but, I can honestly say I have arrived. It is too soon to say too much, but it is all down to the numbers, the financial ones of course. There is a teaser later in this journal, but please be assured this is very real and I take it seriously. Much of what is on the internet is attention seeking. Of course on its own its a very foolhardy proposition as something like this is not profitable, but, if one can do it, why not?

On this note I want to leave you with a very appropriate quote from an interview IDW had with Evan:

"My advice to fans who want to continue learning and creating is to look at the world around them and consider how exploring the wider world can enrich not only their skills, but their understanding and enjoyment of whatever their particular fandom is. It's not a betrayal or a departure; everything you learn can be brought back to what you already love, and make it even better. There's all kinds of wild connections out there, just waiting to be made. Fan art is real art, and you do yourself a disservice when you don't treat it as such"

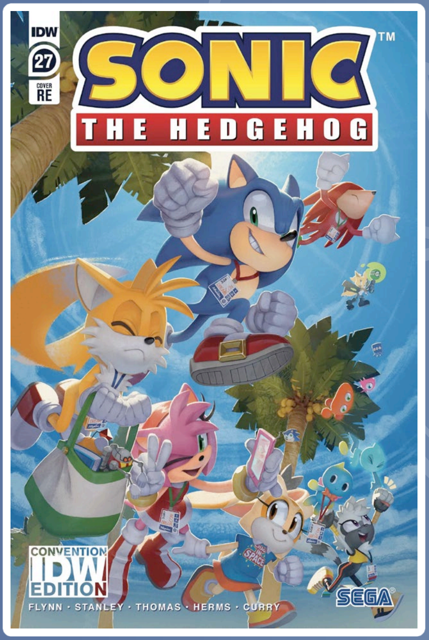
WHO IS EVAN STANLEY?

PHOTO OF EVAN STANLEY AT SONIC REVOLUTION '14 - CREDIT 50 RINGS PHOTOGRAPHY

Admittedly I don't pay too much attention to the artists working on the comics I read (I honestly should, but my time is limited, and I won't bring negativity in about how difficult it is to get IDW's comics in my country).

I recalled the name from the several issues I had read, but when I happened to see the artwork featured alongside, I was impressed enough to dig deeper, and found that she is a really talented artist and one of the right people to carry the franchise into the future along with the current team at IDW. To be honest I was never all that enthused with Archie's Sonic comics!

As I found out, Evan has been a Sonic fan nearly as long as I have. A resident of California, United States, she is a freelance artist, her work features in nearly every IDW Sonic The Hedgehog issue. (I checked and found Evan has been working on the IDW comic from the beginning).



ARTWORK © EVAN STANLEY - IDW PUBLISHING - COMIIC COVER © IDW PUBLISHING - SEGA

#27 - CONVENTION EDITION

The artwork featured was intended for Sonic The Hedgehog #27- a con-exclusive edition available at WonderCon Anaheim 2020. Due to the global "pandemic" however, the event had been canceled or postponed, so instead IDW made the issue available with this cover variant, for purchase, from IDW's website.

As I mentioned in my thoughts about this art, it is honestly inspiring that the Sega Dreamcast is not forgotten, and nor should it be.

The Sega Dreamcast- the little console that could, but was the victim of the usual corporate bullshit that plagues many companies where shareholders and accountants lord over everything and everyone. Trust me, this hedgehog has seen it all, even experienced it!

Fun with 3D Modeling

Making that DreamCon Conference Badge

I have to admit this, I adore con badges. The design, the overall creativity, the look. I cannot explain why, it is just one of those things.

The cover art Evan Stanley did for IDW made a large impression upon me, and the thoughts began collecting in the back of my mind. These thoughts then morphed into the idea of actually doing a design and render for that con badge, as if it would be a real, manufactured item.

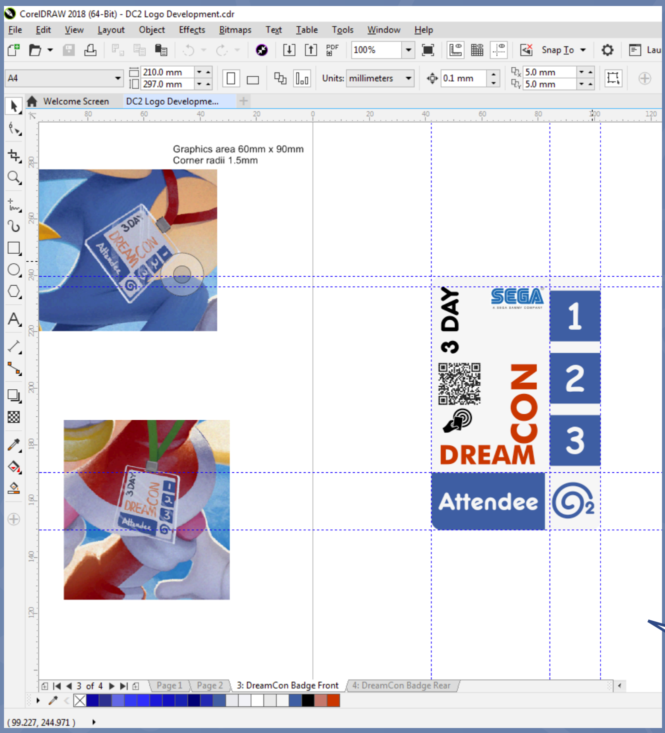
Here's how I did it...

First step- Graphics

Using my favourite application (CorelDRAW) I created the graphics in vector format to get nice crisp clean "print ready" artwork. As far as I could, I used the actual colours Evan used (sampled them in fact and matched them in CMYK). I added a few extras that one would realistically expect to see at a conference- a QR code, and the NFC capability, in addition to info on the back of the badge. I chose not to openly display the attendee's name as these days, with privacy concerns, things are not what they used to be.

When the artwork was finished in CorelDRAW I export both sides independently to high resolution PNG. This is the exact same method I use in my own manga workflow.

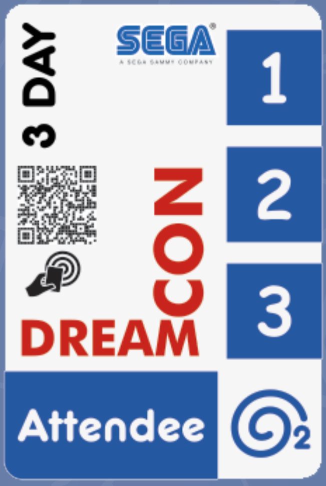
The result was two separate high resolution PNG images (300dpi) which would ultimately be the "textures" used in the 3D model. We can't use the term "texture" in this context, because the size is dependent on the size of the selected face, and this is why SolidWorks refers to these as "decals". Therefore these artworks are applied as decals in the next steps in the process of creating the conference badge in 3D CAD/CAM.


CorelDRAW 2018 (64-Bit) - DC2 Logo Development.cdr

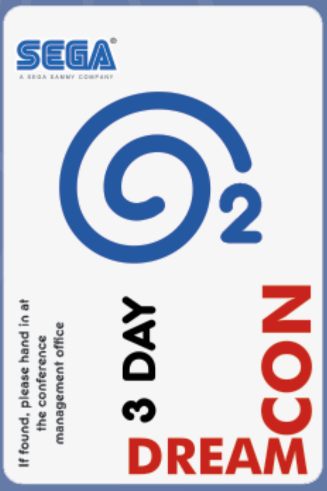
A4 210.0 mm 297.0 mm Units: millimeters 0.1 mm 5.0 mm 5.0 mm

Graphics area 60mm x 90mm
Corner radii 1.5mm

OLD BUT GOLD... CorelDRAW does a great job of preparing the actual printed artwork...



Note the swirl logo. This is a big clue to what awaits... :)

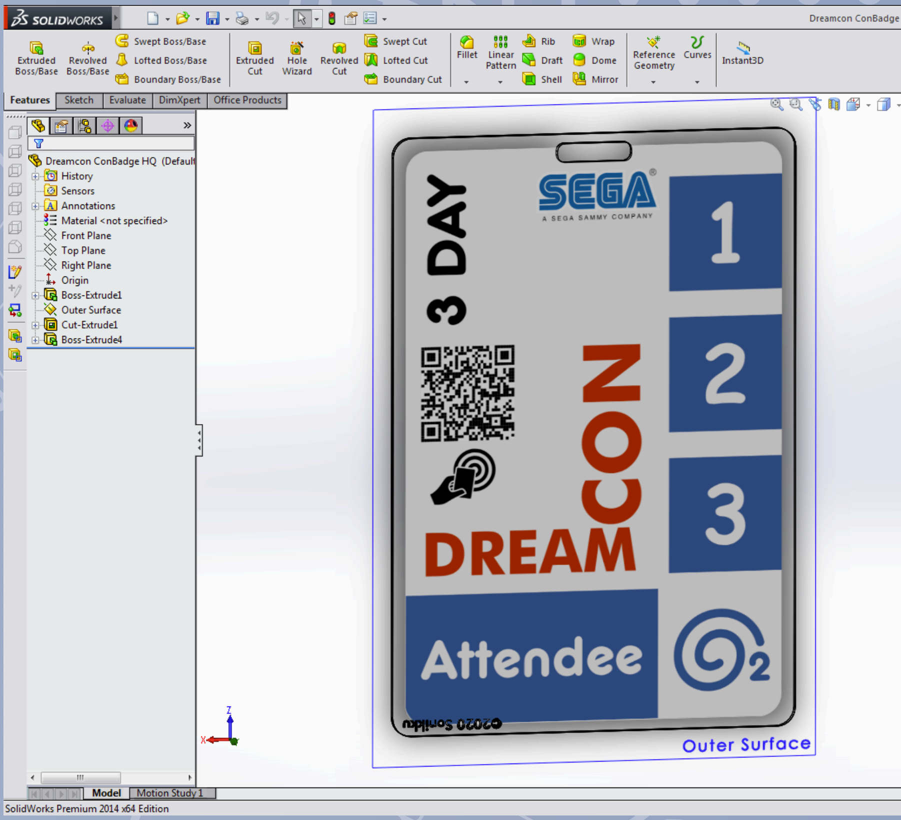

SEGA
A SEGA SAMMY COMPANY
2
If found, please hand in at the conference management office
3 DAY
DREAM CON

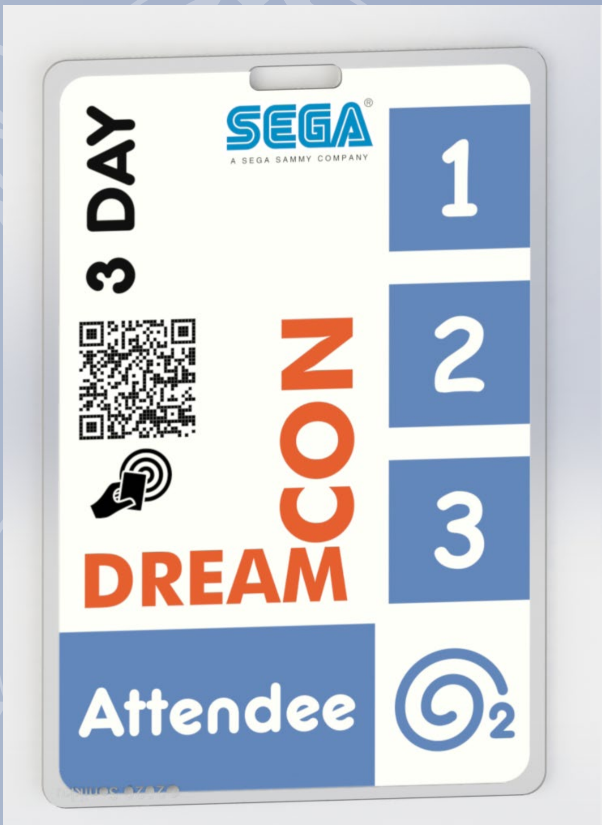
Second Step- Badge Model in SolidWorks

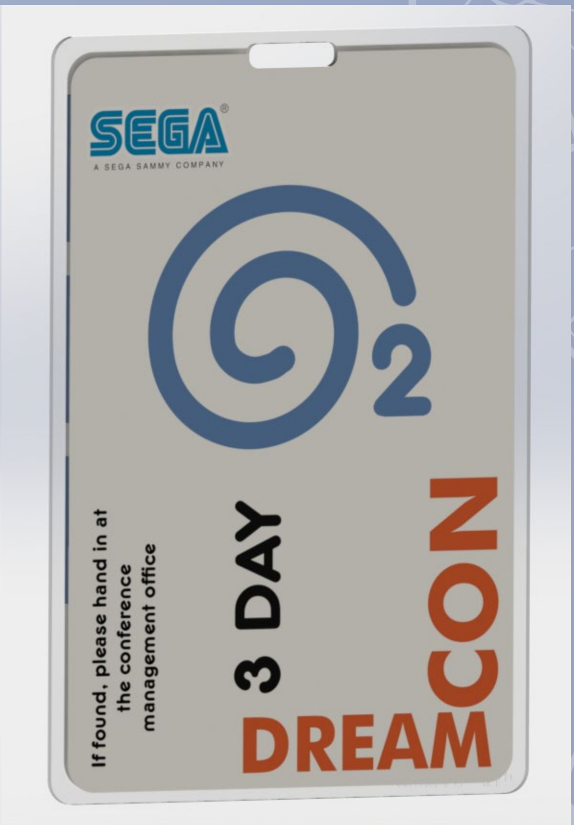
The badge itself, the physical plastic, I designed in SolidWorks. Typical size for the con badge is 65 x 95mm with a rectangular slot for the lanyard. The artwork is 60mm x 90mm, in order to cater for a transparent border indentically to what Evan did in her artwork. Material chosen is clear plastic. The sketches are made, a boss/base feature is extruded and we do this before doing the extruded cut for the lanyard.

Once I was happy with the size, thickness and shape, I applied the artwork from CorelDRAW (the PNG files) as decals on each side. SolidWorks provides handles one can drag to get the size just right and to align them back to back. I performed the extruded cut last, to ensure it clipped the artwork correctly and to get the position right.

There are many multiple ways of doing this, but I did it the way it best made sense to me, and the result was outstanding. The badge looks very real and due to the transparency of the badge, rendering took a long time, typically three hours a side.







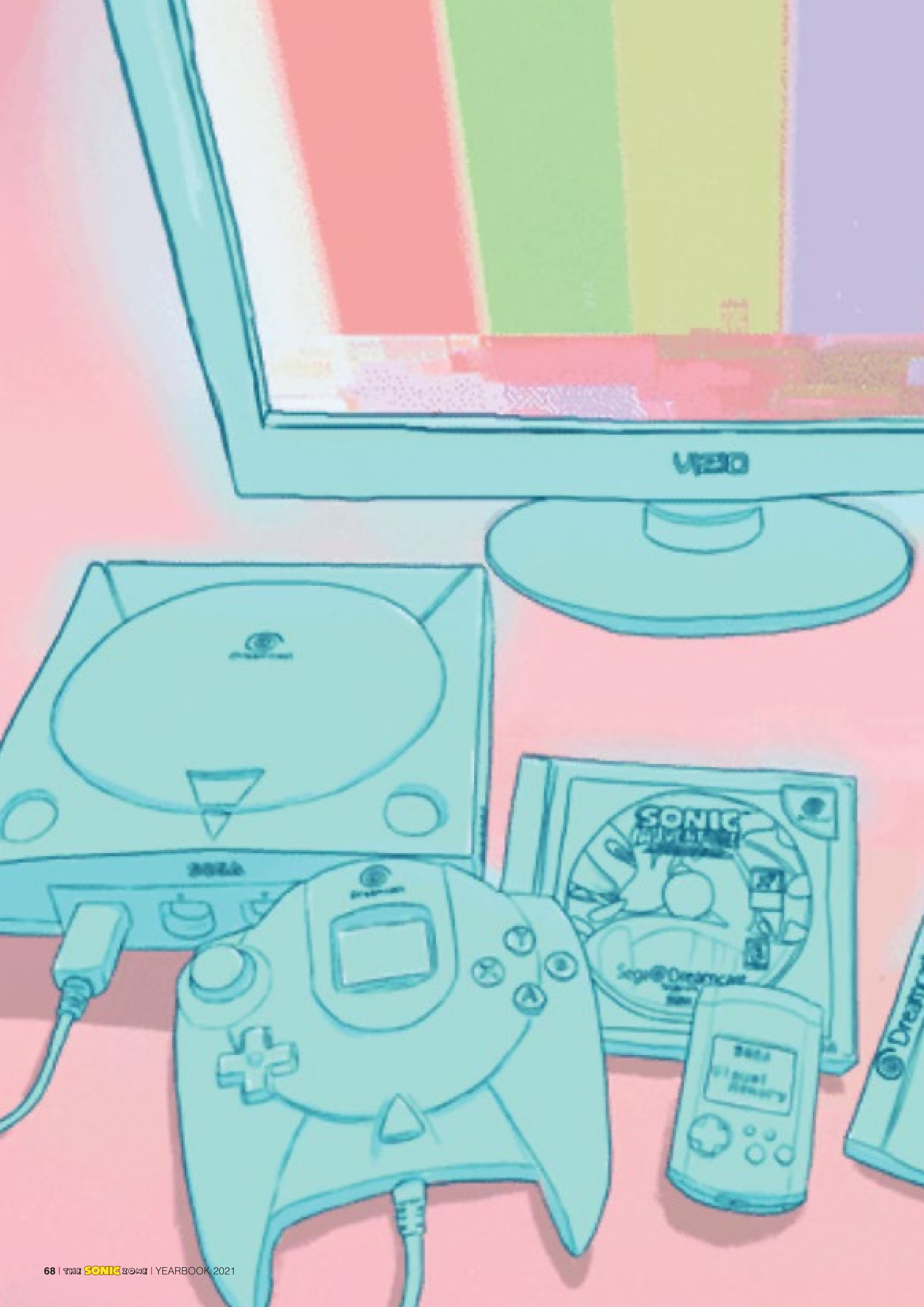
CON BADGE FRONT

CON BADGE BACK

Third Step: Render

Here we have the final rendered front and rear image of the con badge. This looks very real, so real in fact that it had fooled many people who had seen it- asking us when and where this con was taking place. Ah but if only...

Nonetheless this is a good exercise in using SolidWorks for a basic manufactured item. More about how we will use this for real engineering later :)



VIZIO
SEGA
SONIC
Sega®Dreamcast
Visual Memory
Dreamcast

Dreamcast

REKINDLING THE DREAM

September 9th 2019 was a normal day for most people, however for several of us it was exactly twenty years prior that SEGA took their last stand in the hardware business and launched their last console- the Sega Dreamcast. Although the launch at the time was deemed successful, the good turned to bad as the console did not meet sales expectations, leading to SEGA Enterpises (as they were known at the time) abandoning hardware, becoming a software company to avoid bankruptcy.

Let's begin at the beginning

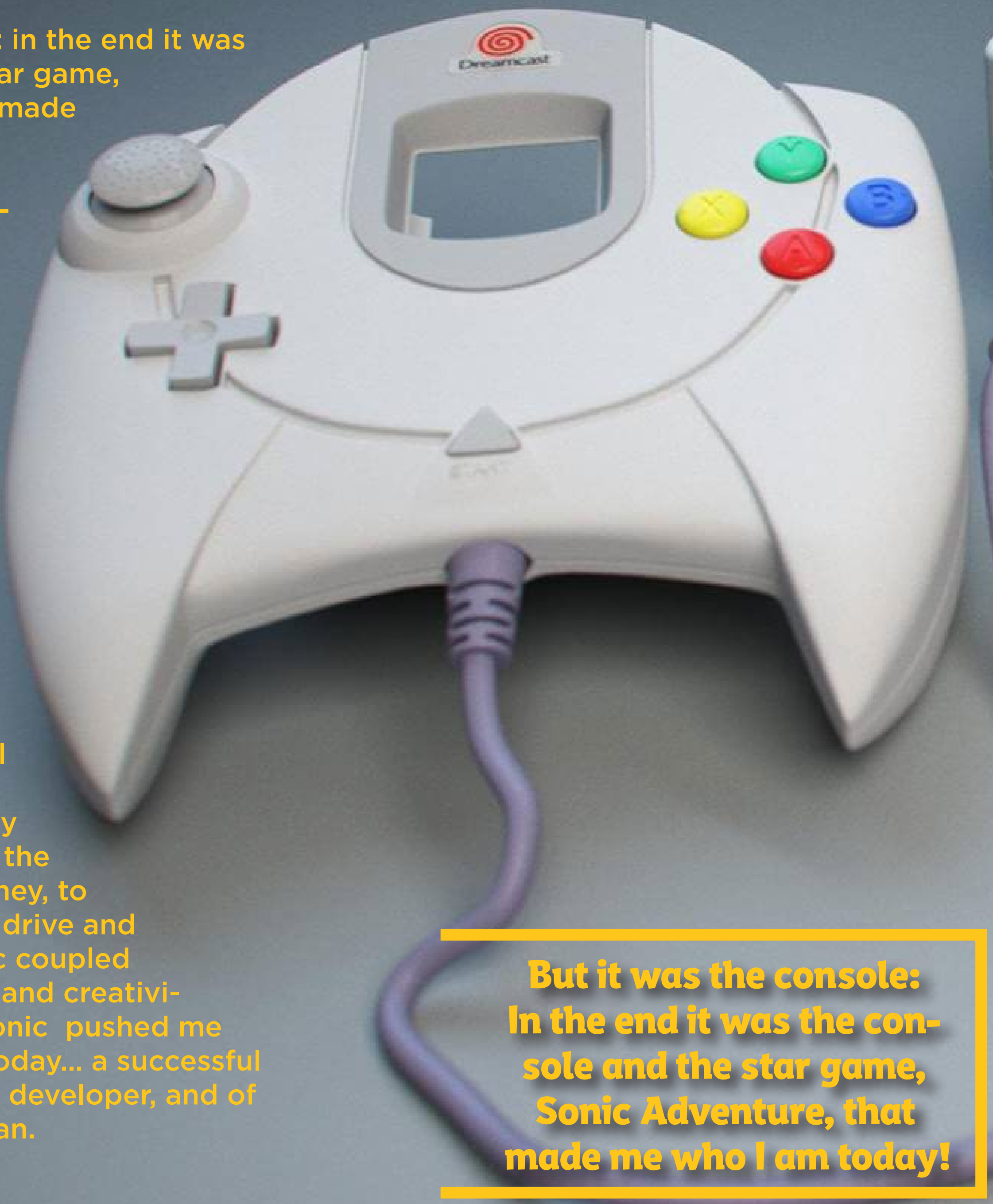
I had to wait until the 5th of May 2000 to take home my Dreamcast, and as Murphy's Law would have it, the copy of Sonic Adventure I had bought with it, was one of the faulty batch that was reported in the gaming press at the time.

Sonic adventure wouldn't load, the console would sit in a perpetual loop trying to read the game disc. Luckily due to changing jobs I had sufficient cash though to purchase a replacement copy of the game a few weeks later, along with a keyboard, an extra Visual Memory Unit (VMU) and copies of the Official Dreamcast Magazine. It was an absolute stroke of luck that I discovered a small gaming shop in Cape Town, known as Subterrania, who were able to get the items for me, at reasonable prices and quickly.

One cold Saturday morning in July 2000 I at last got to play *Sonic Adventure*, without issues, and in progressive video (640x480) on my PC monitor and life would, from that moment on, never be the same again.

I have very fond memories of that time though- it was a time in my life where I ventured out of my bubble and interacted with other people online, mostly in the United States. There was no such thing as social media, nor were websites at the time very sophisticated at all- No Skype, no video-conferencing (the phone companies at the time were still waging war against such applications because it threatened their fat profits), but I used what people overseas were using that flew under the radar nicely: a now-defunct app known as AOL Instant Messenger, and that's how, at 33.6kilobytes per second, with Windows '98 that would crash often, I became immersed in the Sonic fandom.

One thing I can say of the kind of people I met, they were mostly intelligent folk, many of whom went on to become the very people today who pencil the comics, create the games, and manage the marketing- not the zoomers and absolutely useless deadbeats one finds online today. One was able to have a spirited conversation back then with another intelligent human being, online!



But it was the console: in the end it was the console and the star game, *Sonic Adventure*, that made me who I am today!

The game and the console made me want to be an electronic engineer, a software developer, a designer- I wanted to know how the hell they did all that stuff in the small off-white plastic box, how all that magic happened, and how it came to be that it wasn't a bloody IBM computer running crappy Windows. In fact, the small box outperformed any computer we could buy for any sum of cash at the time. So began a journey, to where I am today- the drive and determination of Sonic coupled with burning curiosity and creativity. This console and Sonic pushed me to become who I am today... a successful engineer and software developer, and of course, a huge Sonic fan.

But it was the console: In the end it was the console and the star game, Sonic Adventure, that made me who I am today!

From much joy, to sadness, and the no gaming years

Much of the year 2000 I spent, in my own bubble with the Dreamcast and Sonic fans online, who had introduced me to more than just a world of Sonic, but anime in addition to Japanese culture and things that everyone these days simply take for granted- social skills! Its significant because I was working really crappy jobs at the time, earning minimum wage, but what I was doing, even running a Sonic fan site from this country, kept my mind where it needed to be. I played the game for hours and when I was not playing, I was building my website, interacting with fans, being exposed to the concept of fan-art, and generally just doing things which, at the time were considered taboo in this country- that is, being a complete gamer and geek.

Unfortunately, early in February 2001 I was informed by Subterrrania that SEGA pulled the plug on the Dreamcast. Remember in those years there was no such thing as "news sites" as they are known today, nor things such as "social media". I recall I felt like I was punched in the stomach, because by then I had small successes with doing development on the machine.

As early as December 2000, I had figured out how to do things to the console, such as make the controller vibrate, and if I recall correctly, display messages on the VMU screen, using code in web pages (others in the United States had found my web pages and also had fun with them). And I had succeeded on other fronts too, running small assembler programs loaded via the serial port.

All was not lost though, Sonic Adventure 2 was coming and when I got it, I went through another round of immersion. The game was great and I spent most of that year playing it. I recall how I would wake up at 8AM, eat breakfast, play until 1pm, and then leave for work (I was working shifts at some useless company that I'd rather not mention here).

SEGA's decision though, had offended me, which meant by early 2002 I had moved onto other forms of entertainment, and the friends I had online had also moved on from Sonic- they completed university, got jobs, got married, had children, and so forth. By 2001 I was already watching anime, which at the time was extremely hard to come by in this country so in 2002 I branched out locally and found anime clubs and focused on that extensively for years, to the extent that in 2003 I was formally in university doing Japanese language study part-time in the evenings, and by late 2001 I had gotten into what I wanted to do so badly at the time- I was working as an electronics engineer.

In 2004 I met my wife, got married in 2007, and then focused heavily on my career, which in retrospect was the right thing to do. Sonic had to take a back-seat for a while but, all the skills acquired during those years made it worth putting Sonic in a corner to tap his foot relentessly. During those years I did wonder if I should buy a console again and get the ported (and improved) versions of these Sonic games, but, my focus on my future and my child let me put that on the back burner and for good reason- responsibility. After my son was born, things became more relaxed, and I felt the need to reconnect with my friend the blue hedgehog. At that point I had already had coding experience (had learnt C and C++), my engineering career was already years underway and the time looked right to be able to do stuff, you know, hack games, maybe even make them.

Coming back to Green Hill Zone

Early in 2012 I became aware of new Sonic games with stunning 3D graphics, in Full HD and the console that was considered popular at the time was the XBOX 360. I caved in and bought one that year and bought Sonic Adventure 2 for the console, along with Sonic Unleashed. From that year on, encouraged by my wife, I once again got back into the Sonic fandom and I haven't stopped since. By 2013 I had started with websites once again, the domains of which I own to this day. That year, for the first time I imported my first Sonic figurine, the Good Smile Sonic Nendoroid #214.

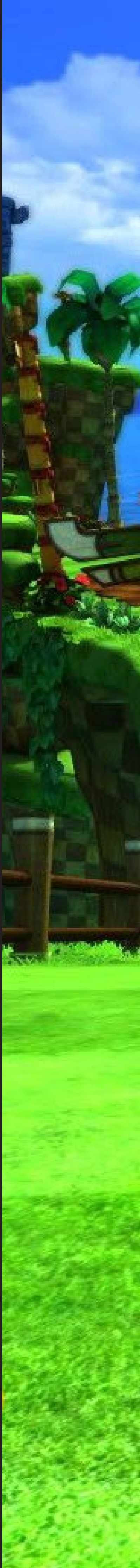
Towards the end of 2013, I had bought Sonic Generations and, needless to say, I still enjoy that game. I had also discovered Blender, and found game assets, in many cases from Sonic Generations, being shared on the internet. The first asset I ever rendered was Tails (and yes it took the whole night and part of the next day on a crappy Pentium 4). It was in that year that I contemplated the reality of what I am doing now, and what you are about to discover next.

Sonic Adventure Trilogy

In common with many older Sonic fans, I always felt that they (Sonic Team) should have released a third instalment of Sonic Adventure. Obviously this has not happened, even though the current head of Sonic Team- Takashi Lizuka, has acknowledged what can be best described as "fan pressure" to do so. It is not likely to happen, and it has to do more with the fact that Yuji Naka is gone forever, and Lizuka has his own way of doing things (as is evident with the number of shitty Sonic games released under his watch). Also, it is a function of SEGA-SAMMY, a very typically Japanese company that actually lives up to common stereotypes about Japanese corporations. Every few years, SEGA undergoes change, and at this writing they've used the "pandemic" to restructure their business once again, but it would seem that they're trying to find their way back into the hardware business.

A new console?

I am old-school, from that school that says "if you want something, do it yourself!". I have seen a number of people online do designs for a successor console but nothing really that is practical or manufacturable. As far back as 2013 I wanted to do what I am doing now, but I had to wait a bit longer to get coding experience, and actual knowledge of how to do such a thing, but, I've arrived and here we are- I am actually doing this.. creating and actually making Dreamcast 2.



FIRST STEPS TO DREAMCAST 2

Even as long ago as 2001, I wanted to conceptualise a console, because it remains an embedded system at heart, and as such is right up my alley. However I would have to wait more than a decade for technology to be generally available to enable me to do this. In part we can thank the explosive technology growth in mobile phones and particularly tablet devices (iPad) for this

I evaluated the situation every year or so, the major stumbling block was that most graphics chipsets were only designed for desktop PCs and for PCIe/AGP busses. In the meantime the video standards changed to digital interconnects and high resolutions (HDMI connections and Full HD resolution respectively). I also did notice the arrival of the Raspberry Pi, but even in those early days I knew the graphics would be a black box- something I've been consistently right about.

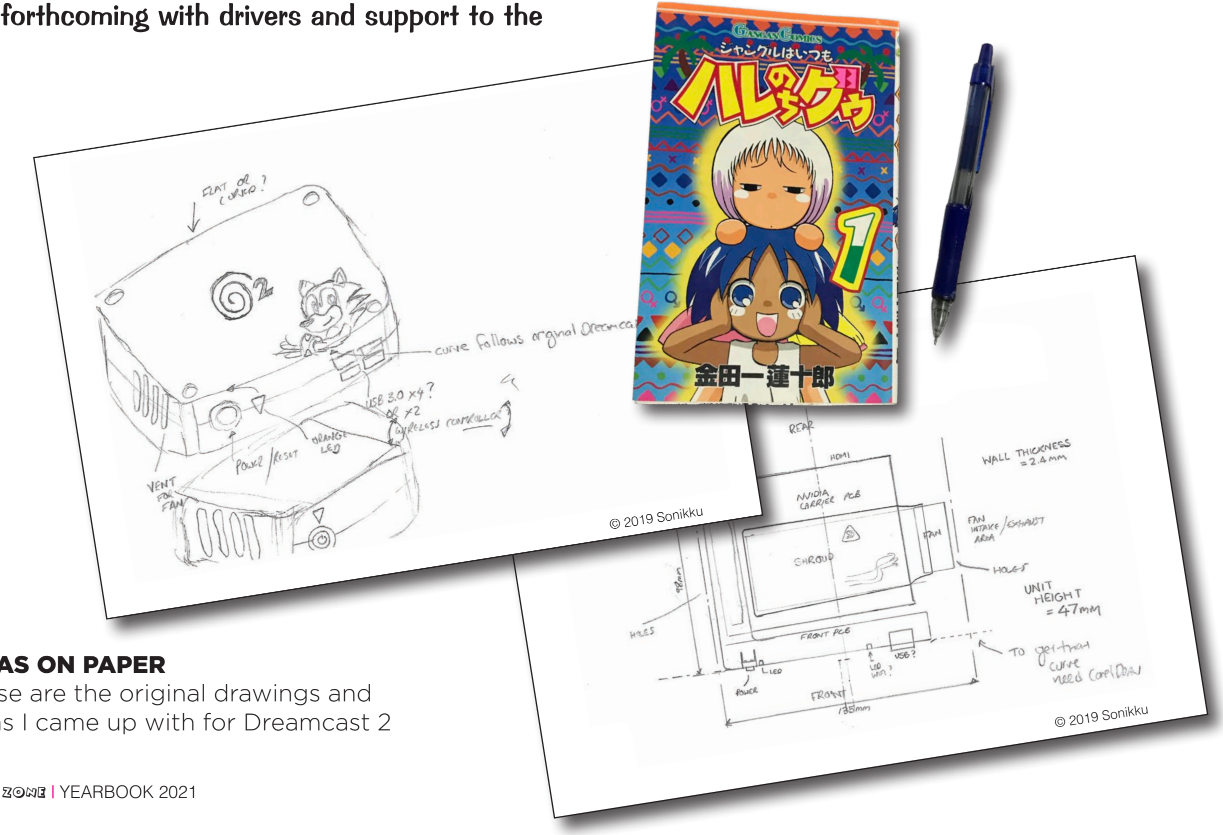
That is not to say I didn't try. I had spent considerable time during 2007 trying to implement some kind of graphics controller in an FPGA. Of course the cost was astronomical and that has been a hugely limiting factor, along with the lack of skills on my part.

During 2013-2014 I noticed that NVIDIA were being more forthcoming with drivers and support to the open source community. At that point I was attempting to hack a RADEON series graphics card to use as a GPU stand-alone but soon realised the infeasibility of that.

Eventually though, NVIDIA began to produce a series of boards aimed at artificial intelligence and robotics- applications requiring machine vision and the like. Finally it seemed I found something that could be used as the base platform of a gaming console. The NVIDIA JETSON NANO development kit caught my eye. What impressed me was that the graphics libraries were available- just a registration away. This meant, for the first time I could attempt a project like this. I was also pleased to learn that the predecessor of this platform, known as TEGRA, was used in the Nintendo Switch- that in itself is huge as far as I was concerned! As I would find out, this board shares some libraries with the Nintendo Switch!

I chose the 2019 release of the JETSON NANO (the version with 4GB memory) but the deciding factor for me was the stated graphics capabilities, which by my reckoning are more than sufficient to run

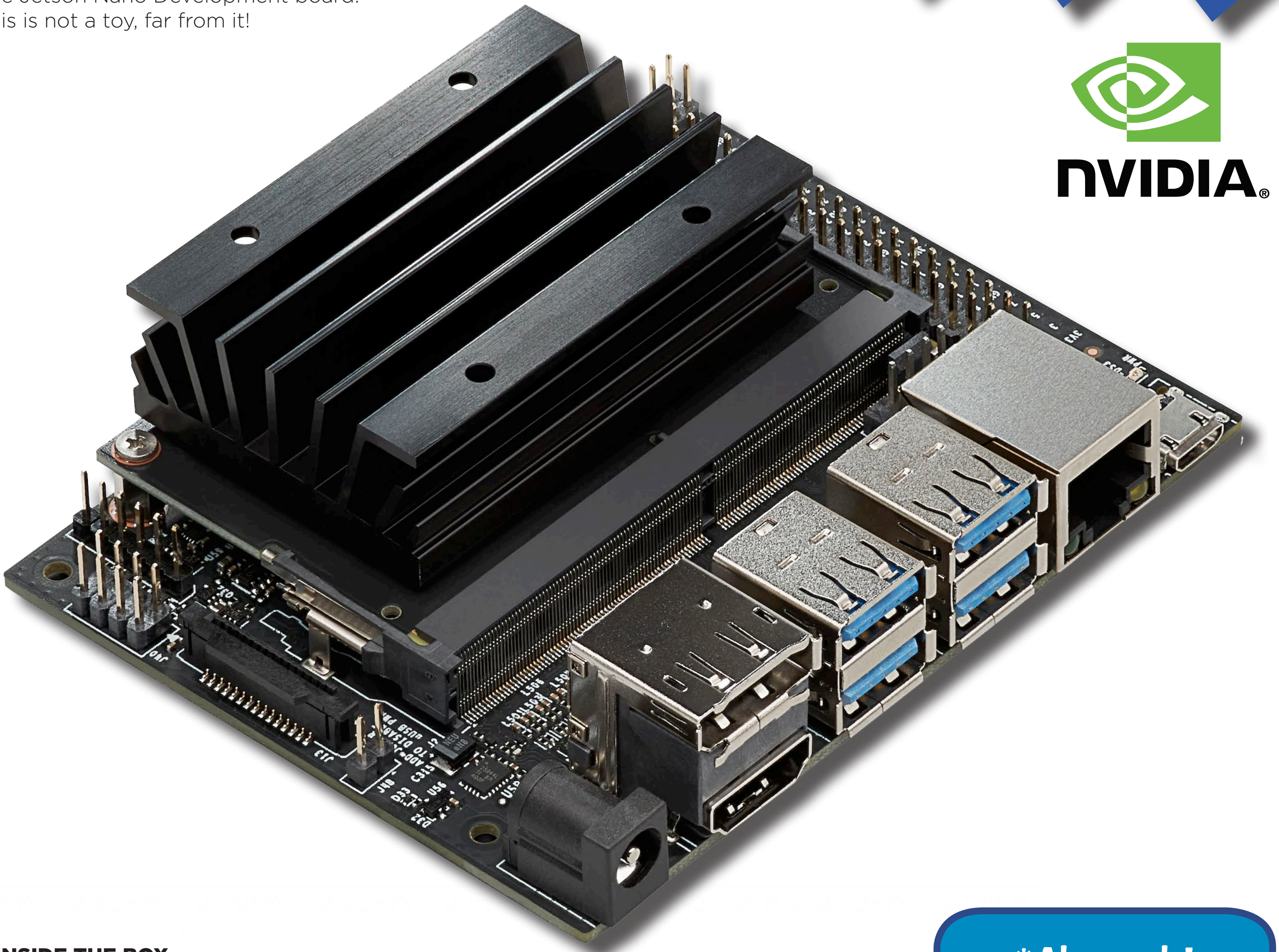
FOR SCALE
This manga gives you an idea of the size of how I drew this out. By the way, that *tankobon* volume was purchased in 2003, even back then I had these ideas, but not the skills...



IDEAS ON PAPER
These are the original drawings and ideas I came up with for Dreamcast 2

MANGA- JUNGLE WA ITSUMO HALE NOCHI GUU by KINDAICHI RENJURO © GANGAN COMICS - PENCIL BY PILOT JAPAN

UP CLOSE AND PERSONAL
The Jetson Nano Development board.
This is not a toy, far from it!



PRODUCT PHOTOGRAPHY SUPPLIED BY NIVIDIA, FRANCHISED DISTRIBUTORS

NVIDIA.

INSIDE THE BOX
Stuff that is inside the box with the board...

HARDWARE SUPPLIED BY:
MICRO ROBOTICS robotics.org.za

*Quote: Sonic Adventure-
1st cutscene

[Continued from page 60]

any Sonic game, even if emulated. I must however mention that this was a significant investment on my part- where I live this board is pretty expensive but the investment was made with the conviction that it would be worth the expense and effort.

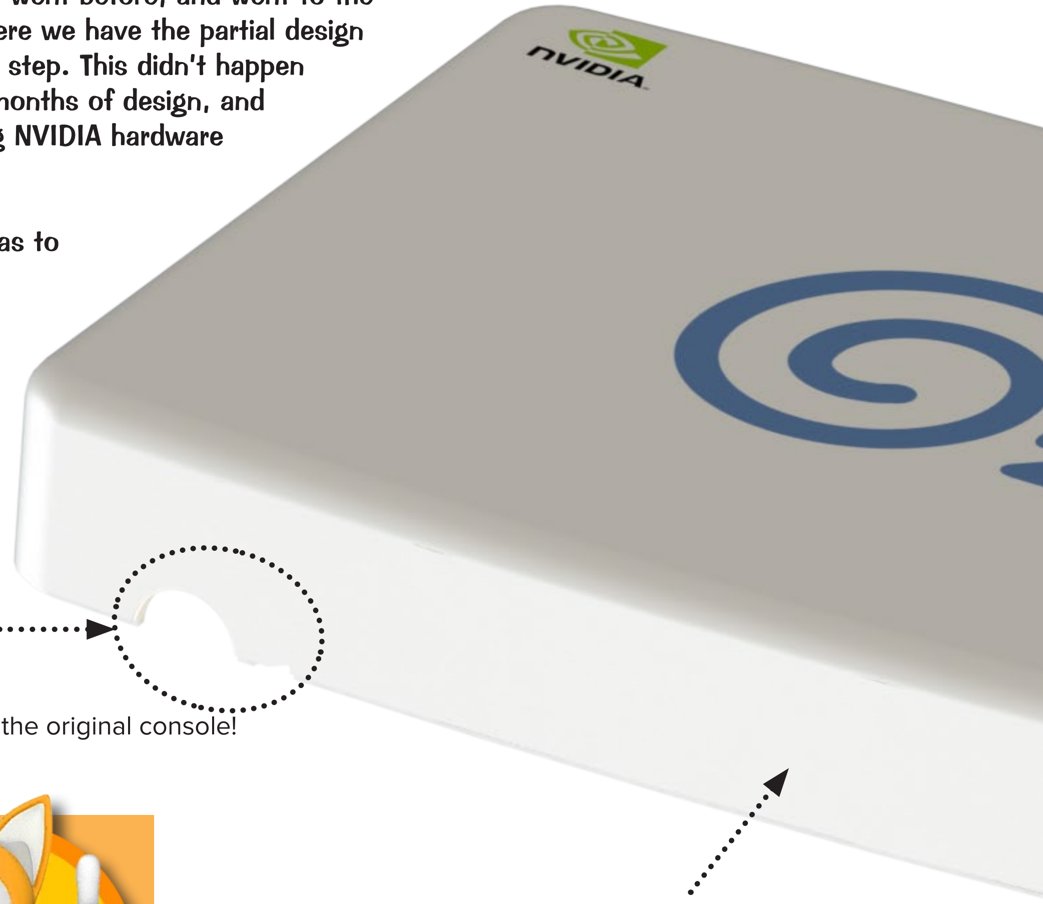
Since I do not have access to the source code of any Sonic games, the most logical way to do a proof of concept such as this would be through emulation and that is why I ensured I chose my hardware wisely. The plan was to port Dolphin Emu to the Jetson NANO, and then run a choice title that runs well in Dolphin- SONIC COLOURS.

PHYSICAL DESIGN

It goes without saying that the physical design of a video game console has to be iconic and specifically appealing. There are other countless rules around this, some of which I will probably never know about. I have seen other's proof of concept console designs (all of them specifically Dreamcast successor console designs) and I don't think any one of them really do any justice.

So what I did was simple. I took what went before, and went to the next logically evolutionary step and here we have the partial design of my vision of that next evolutionary step. This didn't happen overnight either. This is the result of months of design, and keeping in mind we would be installing NVIDIA hardware in that enclosure...

Here it is, complete with annotations as to why I did things a particular way



COMBINING THE OLD AND THE NEW

The power button, this time (not shown in this view) boasts a full RGB LED. In keeping with the original Dreamcast, the power arrow has been retained. During normal operation it shines orange, or blue, depending on your preference or which Dreamcast model you had (US or EU)

The overall curved design is inspired by the original console!

THE OFF WHITE PLASTICS and GREY SEGA LOGO

Call me nostalgic, or plainly stupid but the colour scheme of the original Dreamcast was iconic- well it was to me and still is, and even in this 3D render it still tends to stand the test of time!w

NO OPTICAL DRIVE!

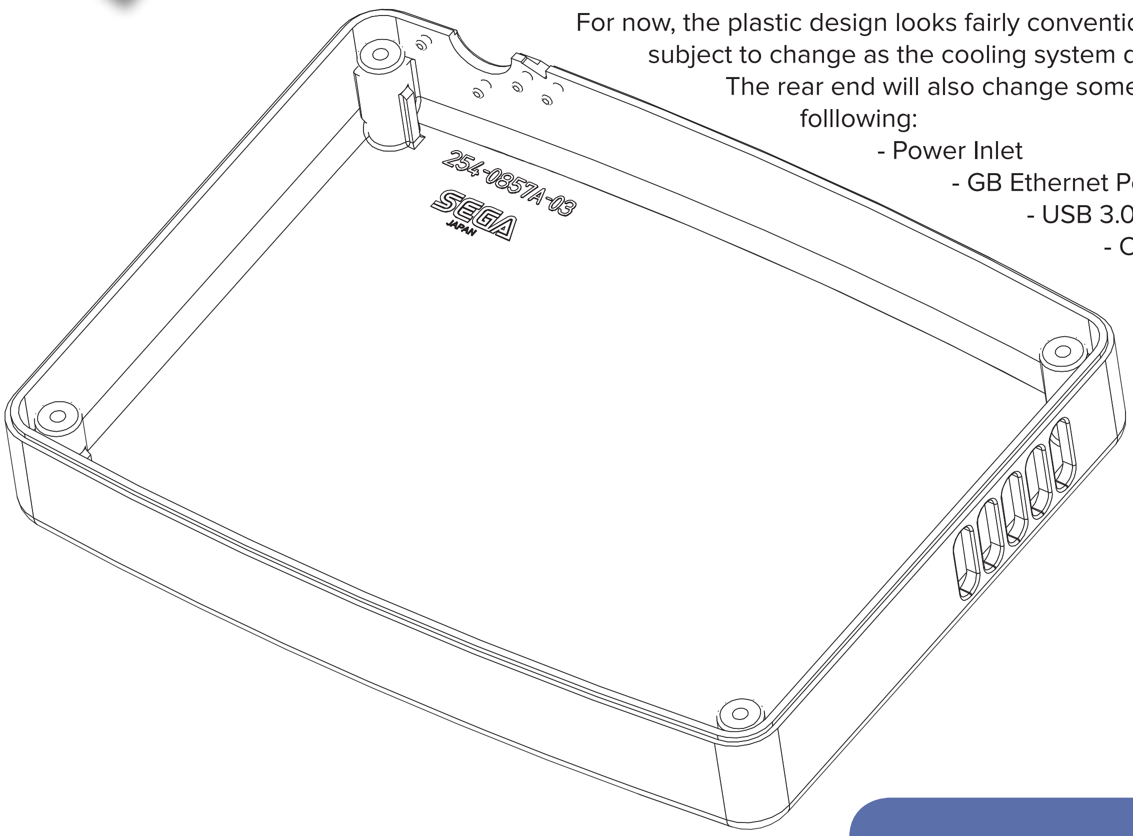
No scratched media, no laser pickup wear issues- your games are stored in the cloud, and downloaded to internal FLASH storage as needed. You may opt for an internal SSD, its your choice.

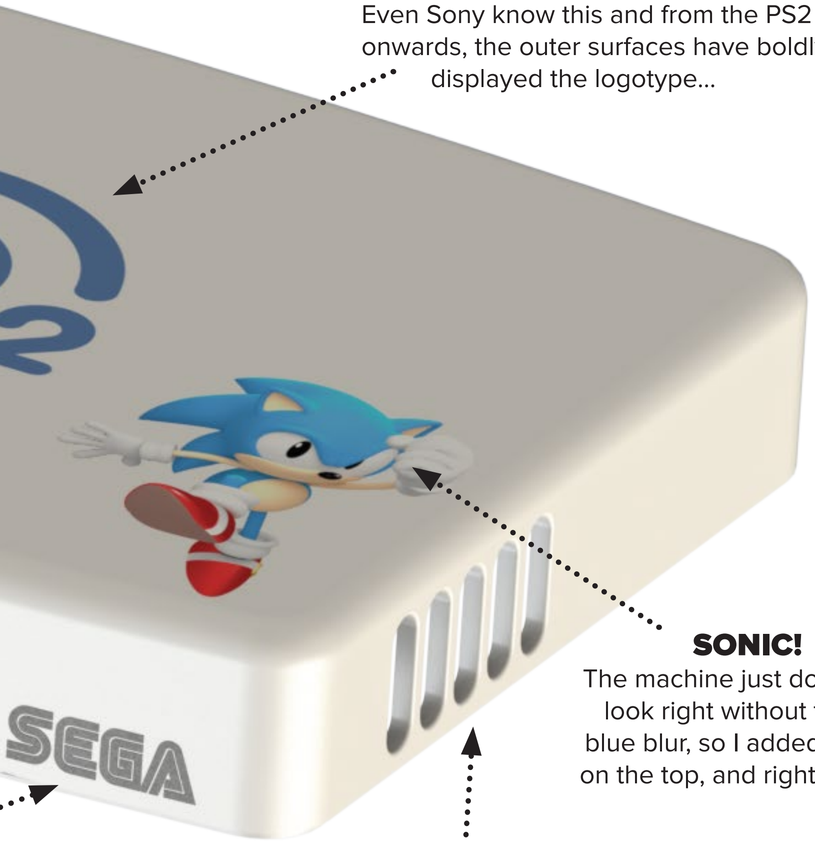
ON THE INSIDE

For now, the plastic design looks fairly conventional. Note however this is subject to change as the cooling system design is implemented! The rear end will also change somewhat to accomodate the folllowing:

- Power Inlet
- GB Ethernet Port
- USB 3.0 Ports
- One HDMI port*
- One DP port*

* Max Res UHD 4K / 2 1080p screens in extend or mirror mode.





BOLD LOGOS ARE IMPORTANT

Even Sony know this and from the PS2 onwards, the outer surfaces have boldly displayed the logotype...

SONIC!

The machine just doesn't look right without the blue blur, so I added him on the top, and rightly so!

THE HEAT HAS TO GO SOMEWHERE...

The original Dreamcast was pretty innovative with its thermal management. Here we are decades later and the same design is used in laptop computers and other devices. So we will direct all the heat out the iconic side vent...

YEAH ITS NOT EVEN CLOSE TO FINISHED BUT WOULD YOU LIKE TO CONTRIBUTE OR HELP?

Due to the turmoil of 2020 and 2021 I didn't quite get as far along with this project as I would have liked.

If you really believe in this project and would like to be involved in some way, please do write to us.

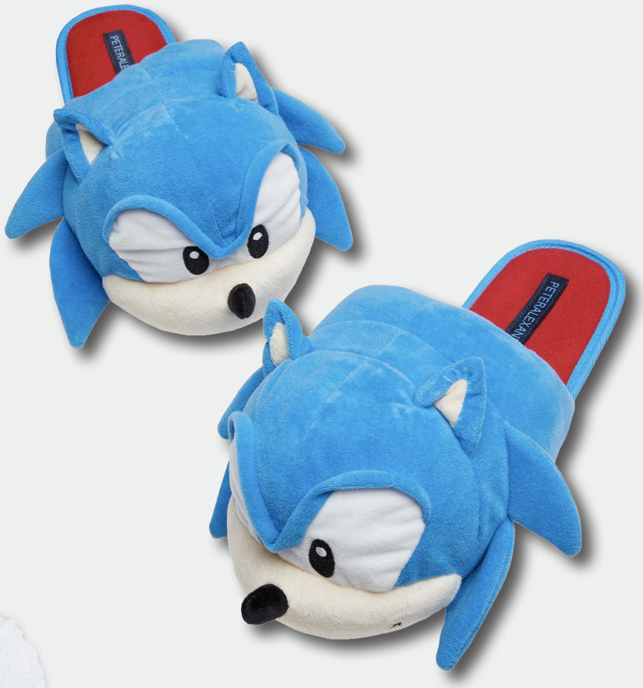
My day job and its attached situations (deadlines, mergers and acquisitions) are mostly the cause. Lots of late nights and burn-out has seen me neglect this. I apologise.

Therefore, if you would like to be involved, please drop me a line (email address given on the editorial page).

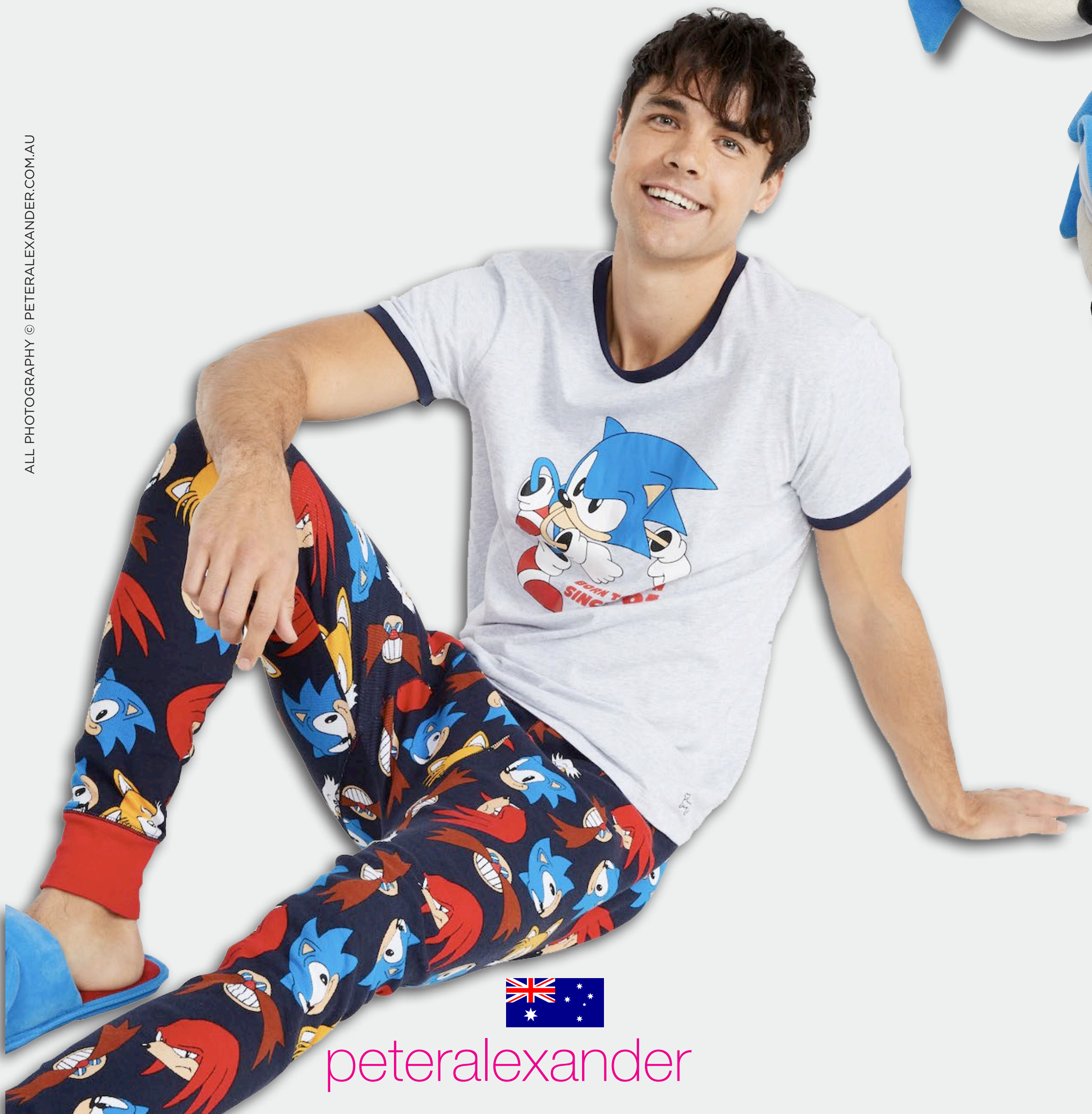
Sonikku

Looking at new and upcoming toys, apparel, and more...

SONIC MERCHANDISE SHOWCASE



ALL PHOTOGRAPHY © PETERALEXANDER.COM.AU



peteralexander

Founded by Australian fashion designer Peter Alexander, the brand is well-known and much loved for its innovative and trendsetting ideas in sleepwear. In 2020 the company licensed Sonic The Hedgehog™ for a range of sleepwear promoted for Father's Day in 2020. Available in Australia and New Zealand.

BORN TO RUN
SINCE '91

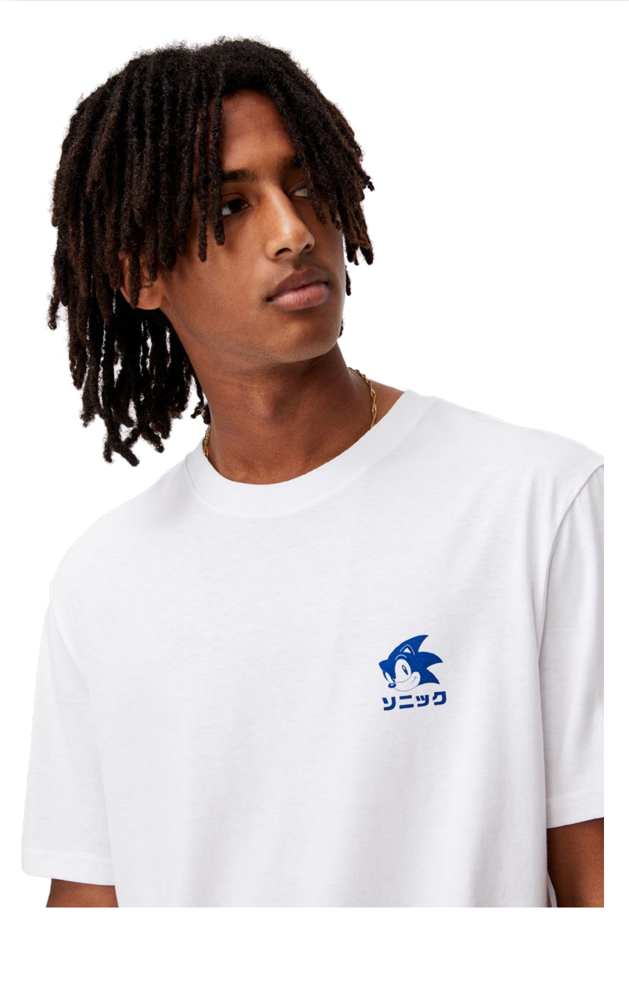
SONIC
THE
HEDGEHOG

peteralexander

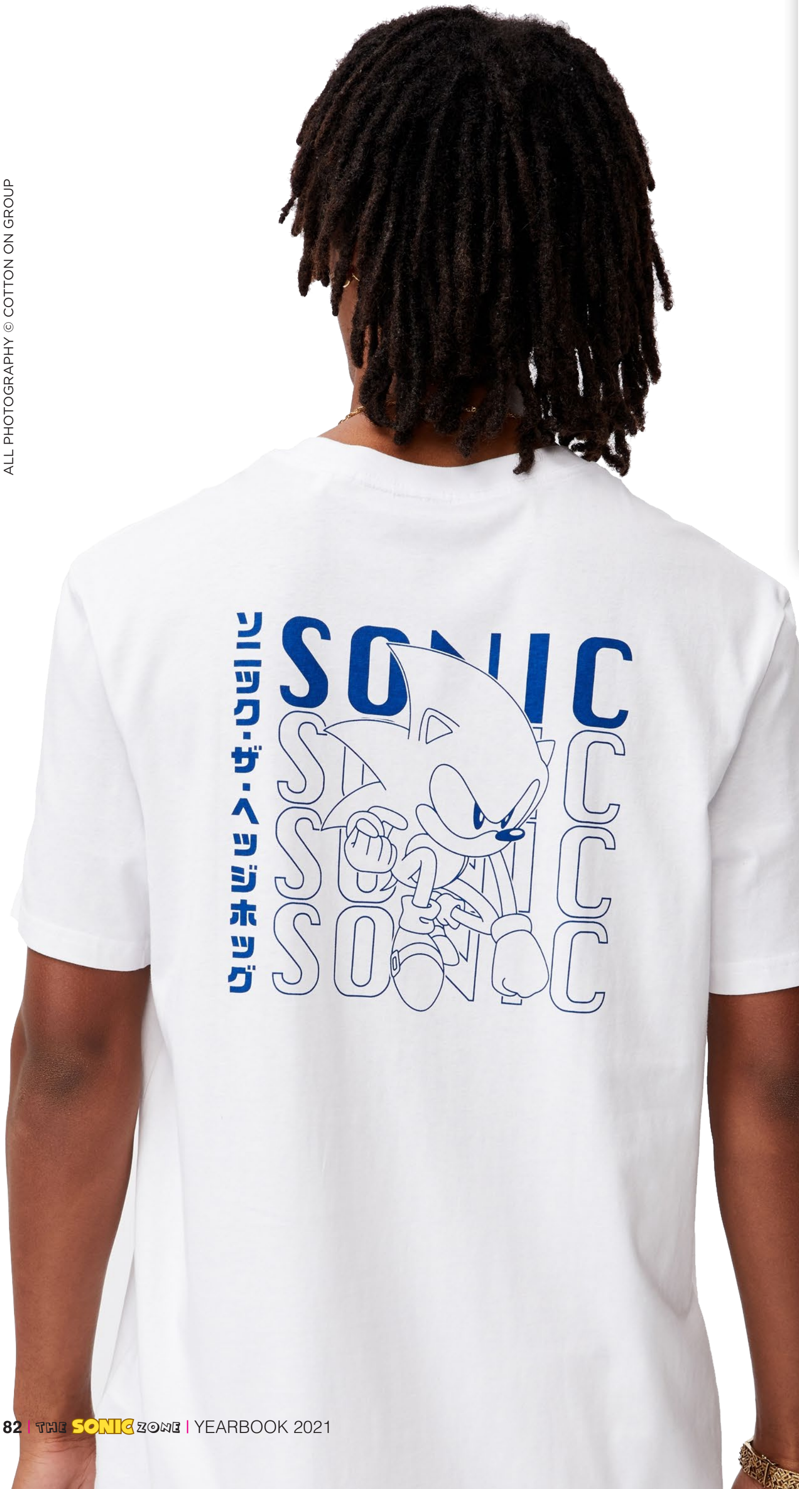
FACTORIE

Regular Gaming Tee

Late 2020 - Early 2021
5297064-02
Iconic Sonic White / Tokyo Sonic



ALL PHOTOGRAPHY © COTTON ON GROUP



COTTON ON GROUP™

Cotton On Group is Australia's largest global retailer, known for its fashion clothing and stationery brands. The Factorie brand is a youth fashion brand and represented globally including South Africa.

SONIC MERCHANDISE SHOWCASE

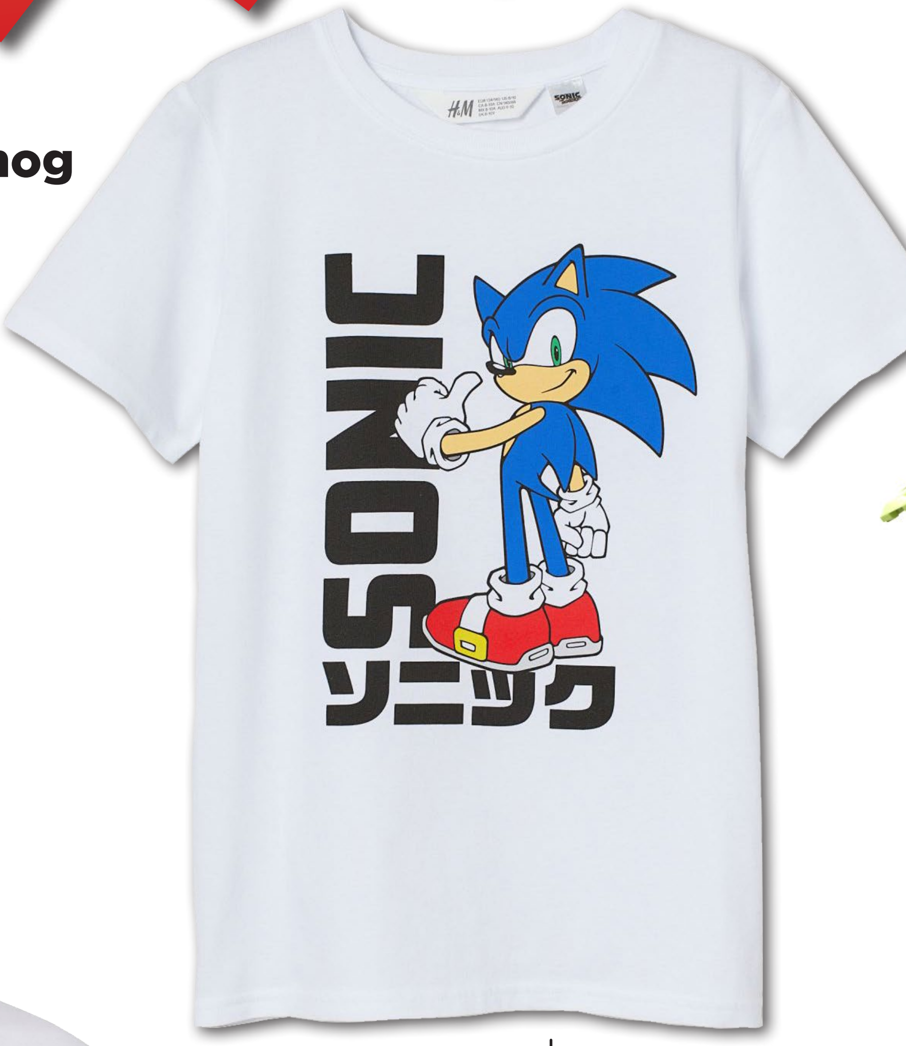
This statue was a GameStop exclusive made by Diamond Select Toys. I noted an early version in 2018 when images surfaced from the 2018 New York Toy Fair. At the time it was "pending approval" from SEGA-SAMMY but in 2019 it was released and sold in GameStop stores in the US as a GameStop exclusive making it slightly more elusive to get. I managed to secure one in 2020, although I had to wait 7 long months for it to reach me.

Only@ GameStop

DIAMOND SELECT TOYS

DIAMOND SELECT TOYS - GALLERY DIORAMA SONIC THE HEDGEHOG™

SONIC MERCHANDISE SHOWCASE



White/Sonic The Hedgehog

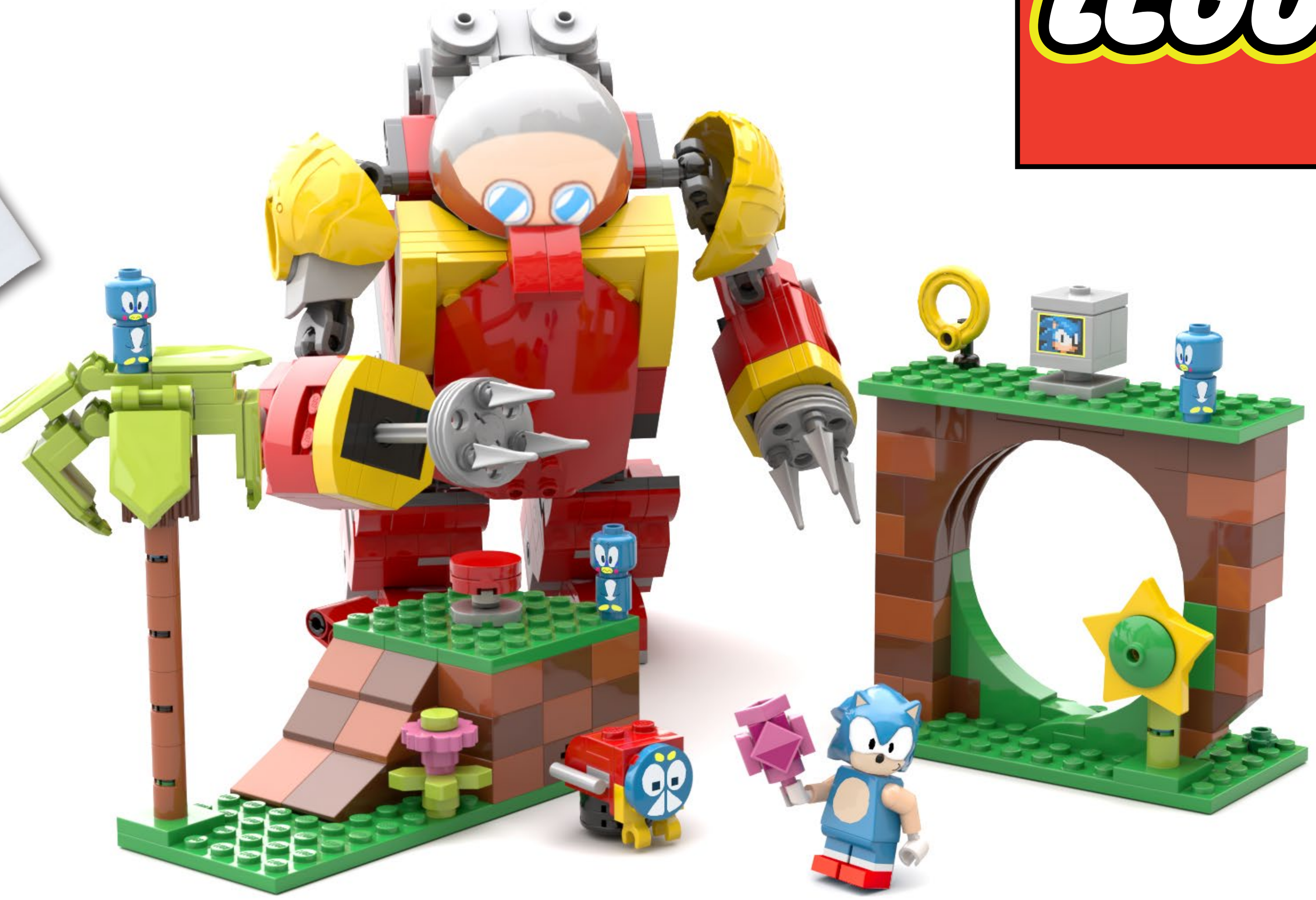
Mid 2021, Launched with other styles on Sonic's 30th Anniversary website sonic30th.com
Article Nr: 0937991007

ALL PHOTOGRAPHY © H&M

Hennes & Mauritz AB is a Swedish multinational clothing-retail company known for its fast-fashion clothing for men, women, teenagers, and children. It is the second-largest global clothing retailer, behind Spain-based Inditex

SONIC MANIA™ GREEN HILL ZONE

Availability: TBA (Assume late 2021 - Early 2022)
Officially announced: 4th Feb 2021*



ALL PHOTOGRAPHY © THE LEGO GROUP

ABOUT THIS SET

This is a concept design submitted by 24 year-old, UK based, LEGO® superfan Viv Grannell. Viv, also a passionate Sonic fan, created her design as a submission to the LEGO Ideas® platform, a LEGO initiative that takes new ideas that have been imagined and voted for by fans and turns them into reality.

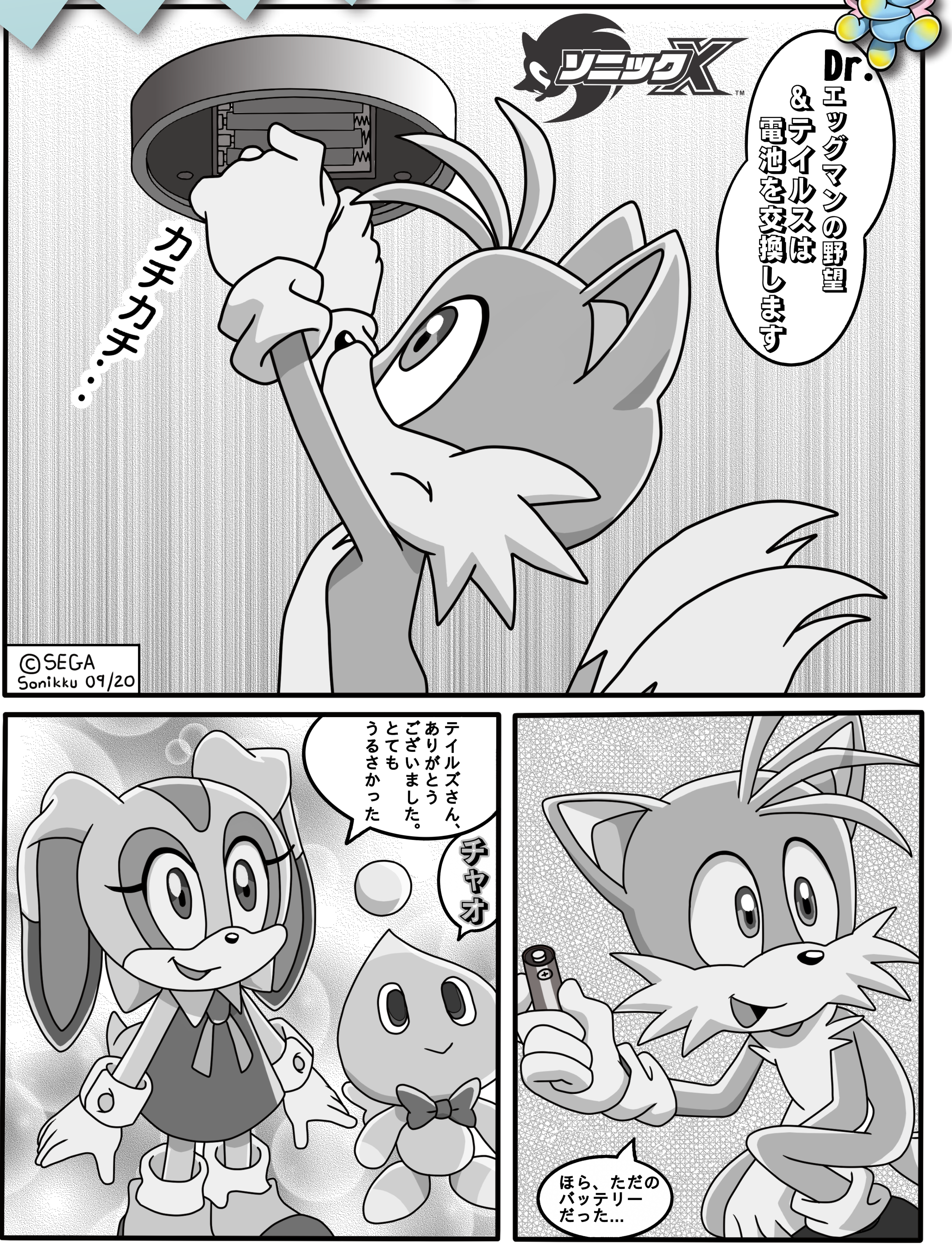
Viv's design was inspired by SEGA's critically acclaimed 2017 platformer experience Sonic Mania™, and features iconic characters and environments from the game's Green Hill Zone level. Once the design was completed, it achieved the prerequisite 10,000 votes from LEGO fans worldwide – before being greenlit to go into production by the LEGO Group.

* https://www.lego.com/en-us/aboutus/news/2021/february/lego-ideas-sonic-mania/

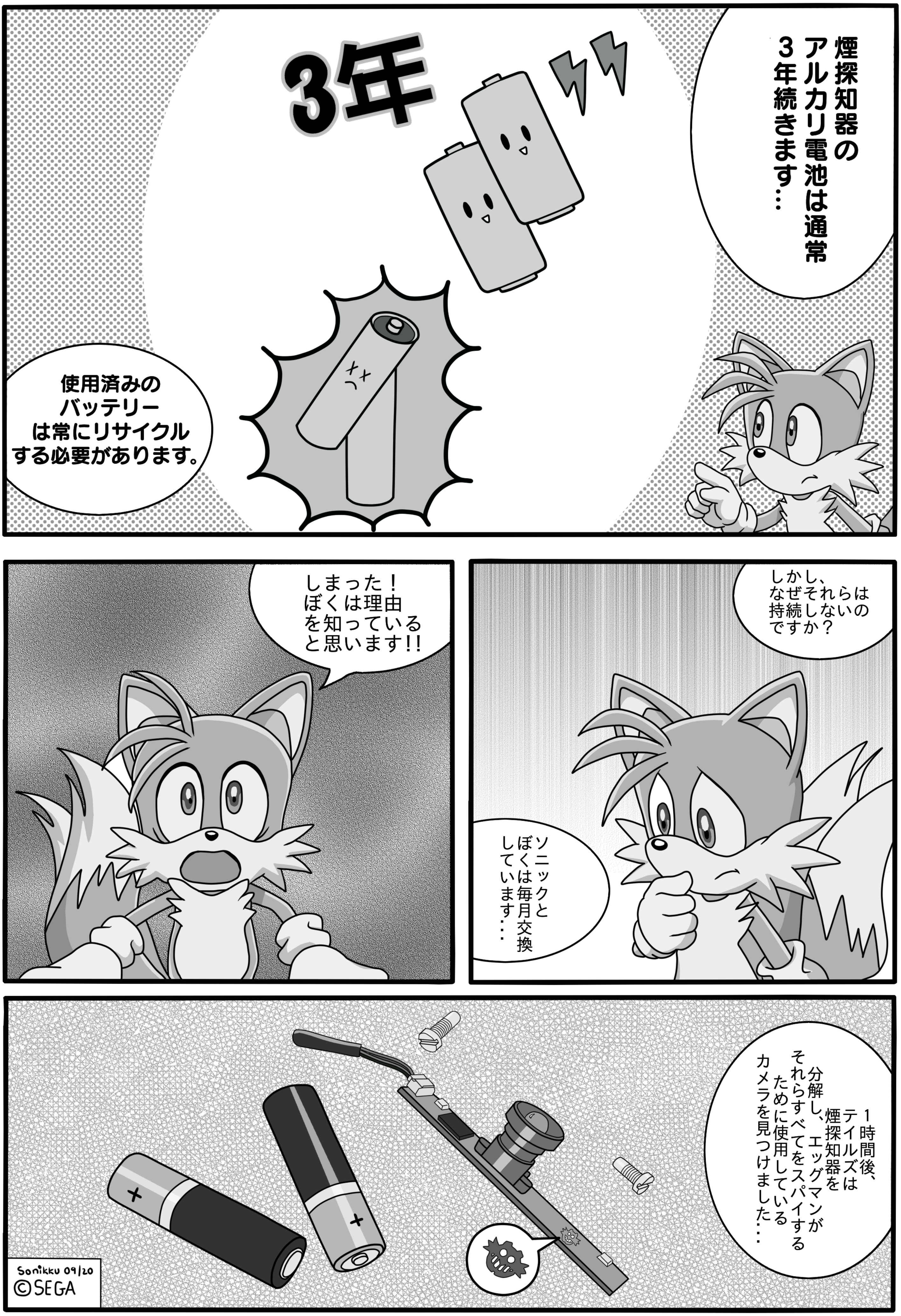
What if Sonic X had manga?

Its a pretty good question...

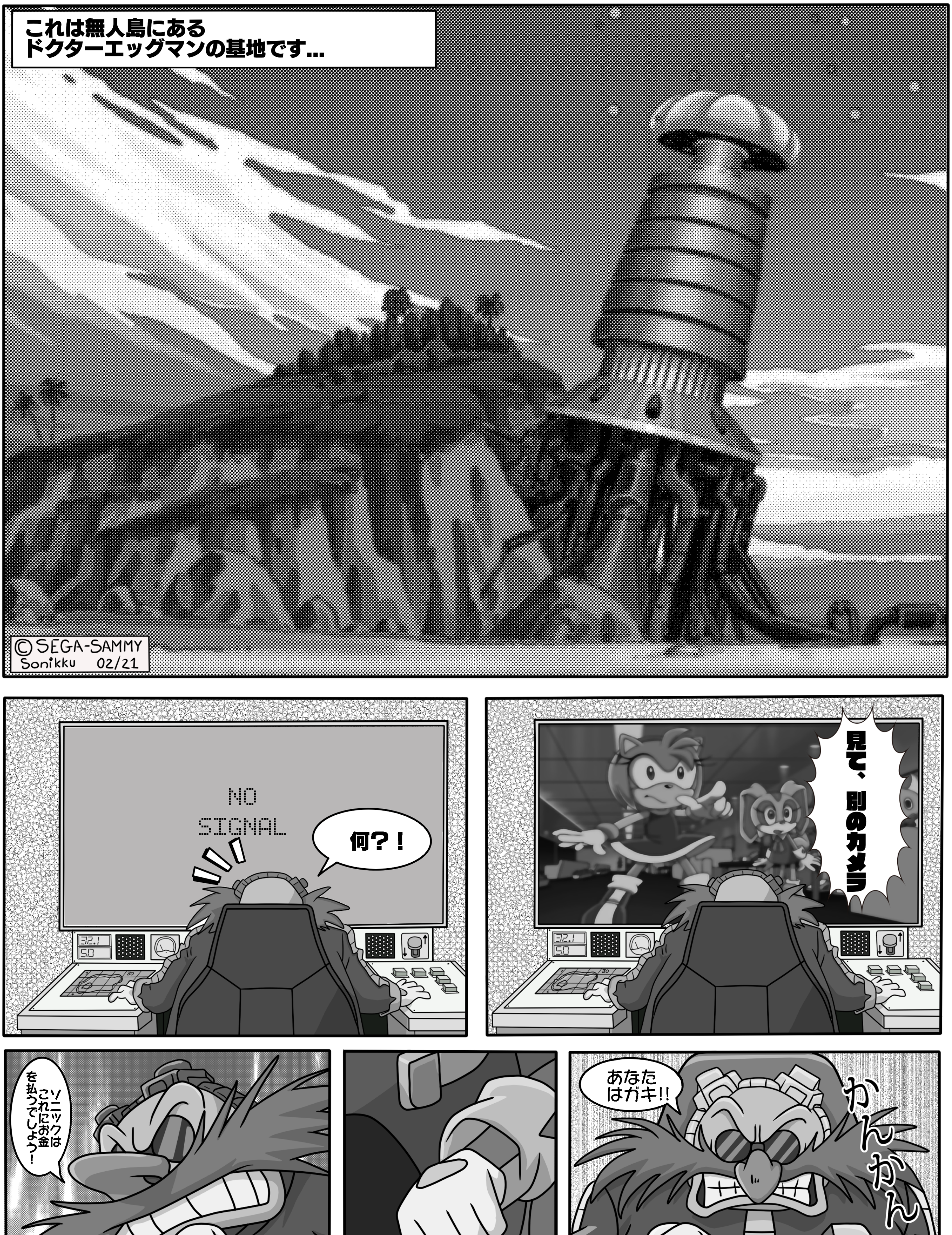
So let's try and see what happens

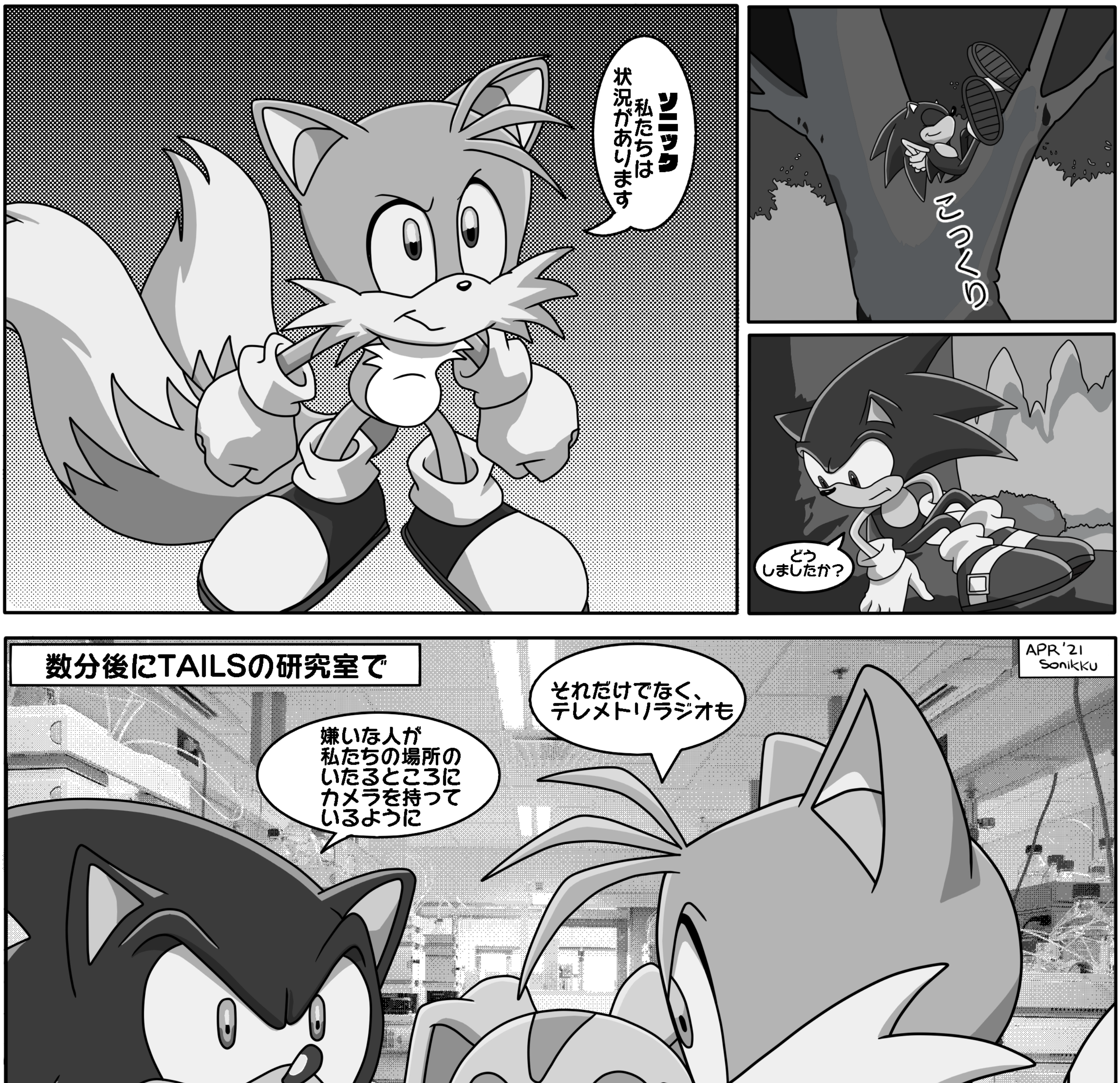

ソニックX
Dr.エッグマンの野望
＆テイルスは電池を交換します
カチカチ・・・
©SEGA
Sonikku 09/20
ほら、ただのバッテリーだった...
テイルズさん、ありがとうございました。とてもうるさかった
チャオ

Author's note - I have no knowledge of any official Sonic X manga, although it is very possible that doujinshi did exist at the time (yes I am aware of the lewd Sonic doujinshi sold at Comikets but that's not valid in this context). Also note that I tailored (didn't blindly trace) the poses from stills taken from the anime and made it more canonical!


煙探知器のアルカリ電池は通常3年続きます…
3年
使用済みのバッテリーは常にリサイクルする必要があります。
しかし、なぜそれらは持続しないのですか？
ソニックとぼくは毎月交換しています…
しまった！ぼくは理由を知っていると思います!!
1時間後、テイルズは煙探知器を分解し、エッグマンがそれらすべてをスパイするために使用しているカメラを見つけました…
Sonikku 09/20
©SEGA

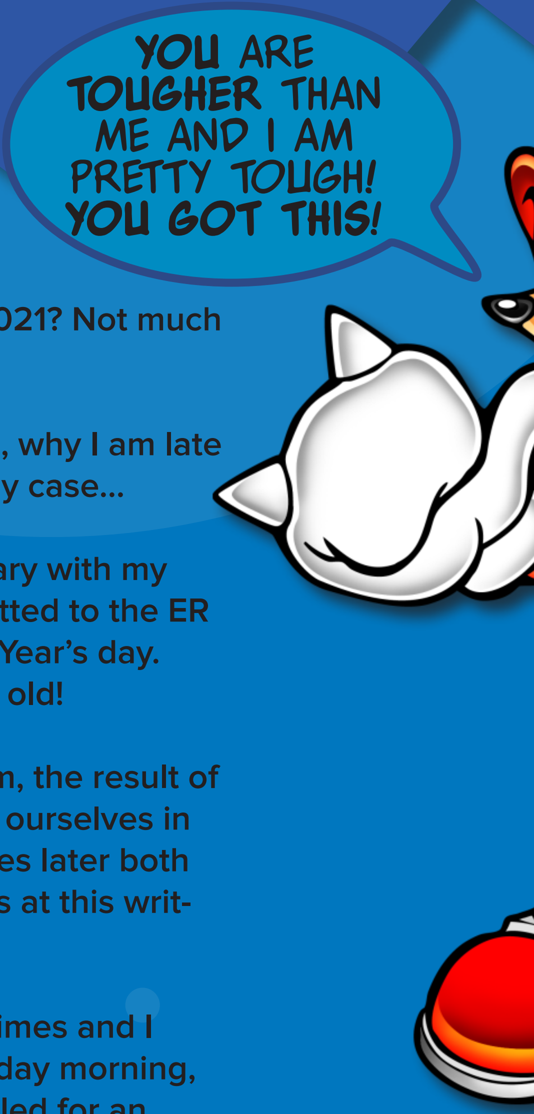
English translated version can be found near the back of this book
英語に翻訳されたバージョンは、この本の後ろの近くにあります。


これは無人島にある
ドクターエッグマンの基地です…
© SEGA-SAMMY
Sonikku 02/21
見て、別のカメラ
NO SIGNAL
何？！
32.1
50
あなたはガキ!!
かんかん
カチー
ソニックはこれにお金を払うでしょう!



That is as far as it goes for 2021. More to follow in next year's yearbook.
それは2021年までです。来年の年鑑でさらに続きます。

BACKING MY BUDDY!



What is there to say about 2021? Not much to say! Not a good year!

Now you know, dear readers, why I am late with this very yearbook in any case...

It started off already in January with my dear friend, Jose being admitted to the ER barely a few days after New Year's day. The year wasn't even 5 days old!

Unable to go to the bathroom, the result of an urethral stricture. We find ourselves in November 2021, two surgeries later both deemed a failure and now, as at this writing, a third about to happen.

Life is unkind at the best of times and I woke up at 2:30 AM on a Friday morning, the 26th of November, the same morning that Jose was scheduled for an assessment for the third surgery- with this very idea. I had tears in my eyes... I had this very spread before you, in my mind's eye, and I grabbed my phone to make sure I made a note!

Life also likes to throw cruel reminders, looking at the date and how perilously close it is to that fateful day in November 1999 when I nearly ended my own life due to depression.

"Sonic? Ah that's for kids!" you might say. I am more convinced than ever before that there is something quite special about Sonic. Its 22 years later and there is still magic happening, the magic that began with me discovering Sonic Adventure so many years ago, a journey that led me to a new life, my marriage, my beatiful child and this wonderful person, a really special friend.

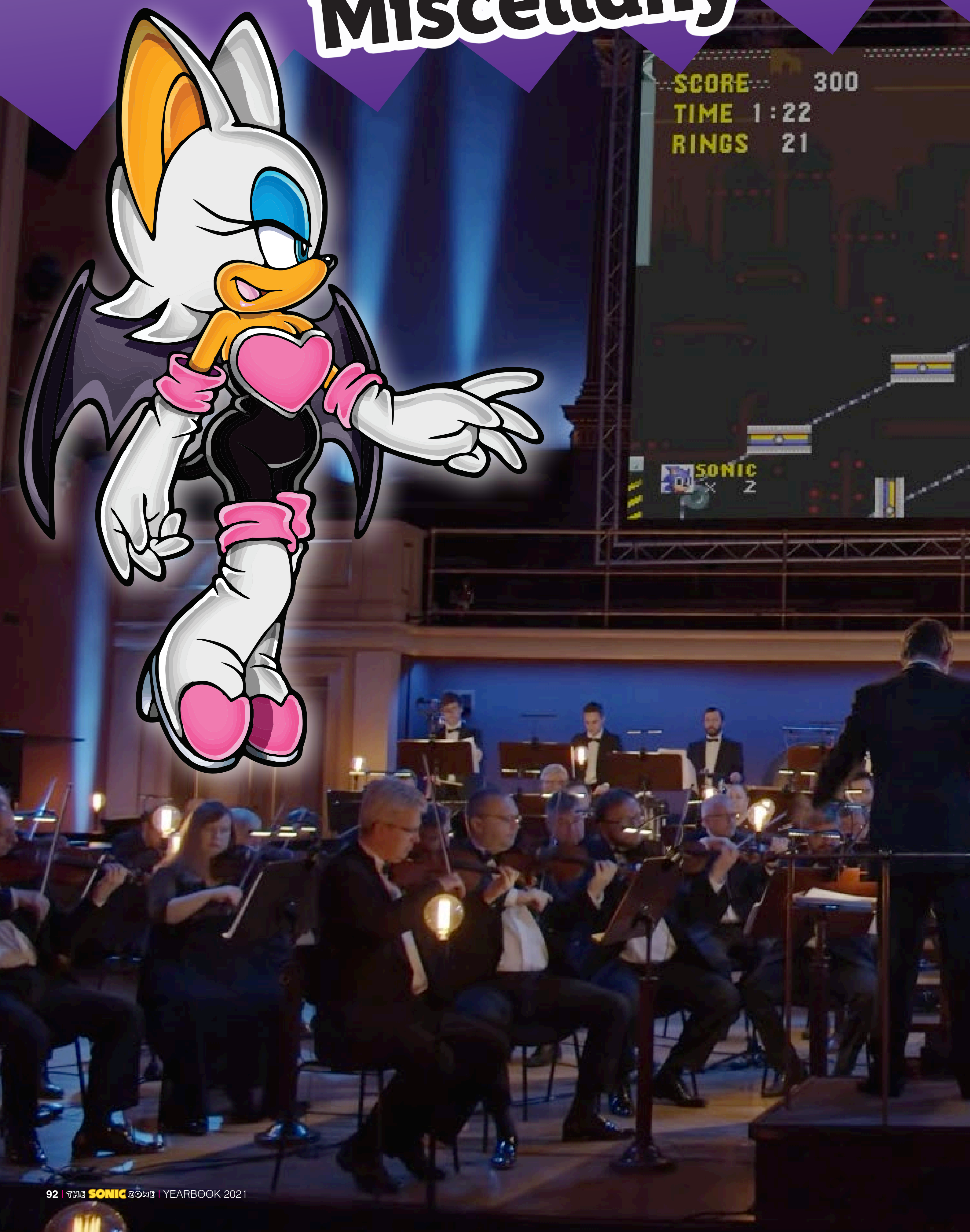
Jose, this is a very personal message to you and I want you to know that you are loved, your family (your actual family and your tattoo family) we all care about you and we want only the best for you.

Sonikku has your back buddy, let's get you better.
Strongs, Godspeed, and we're in this together!

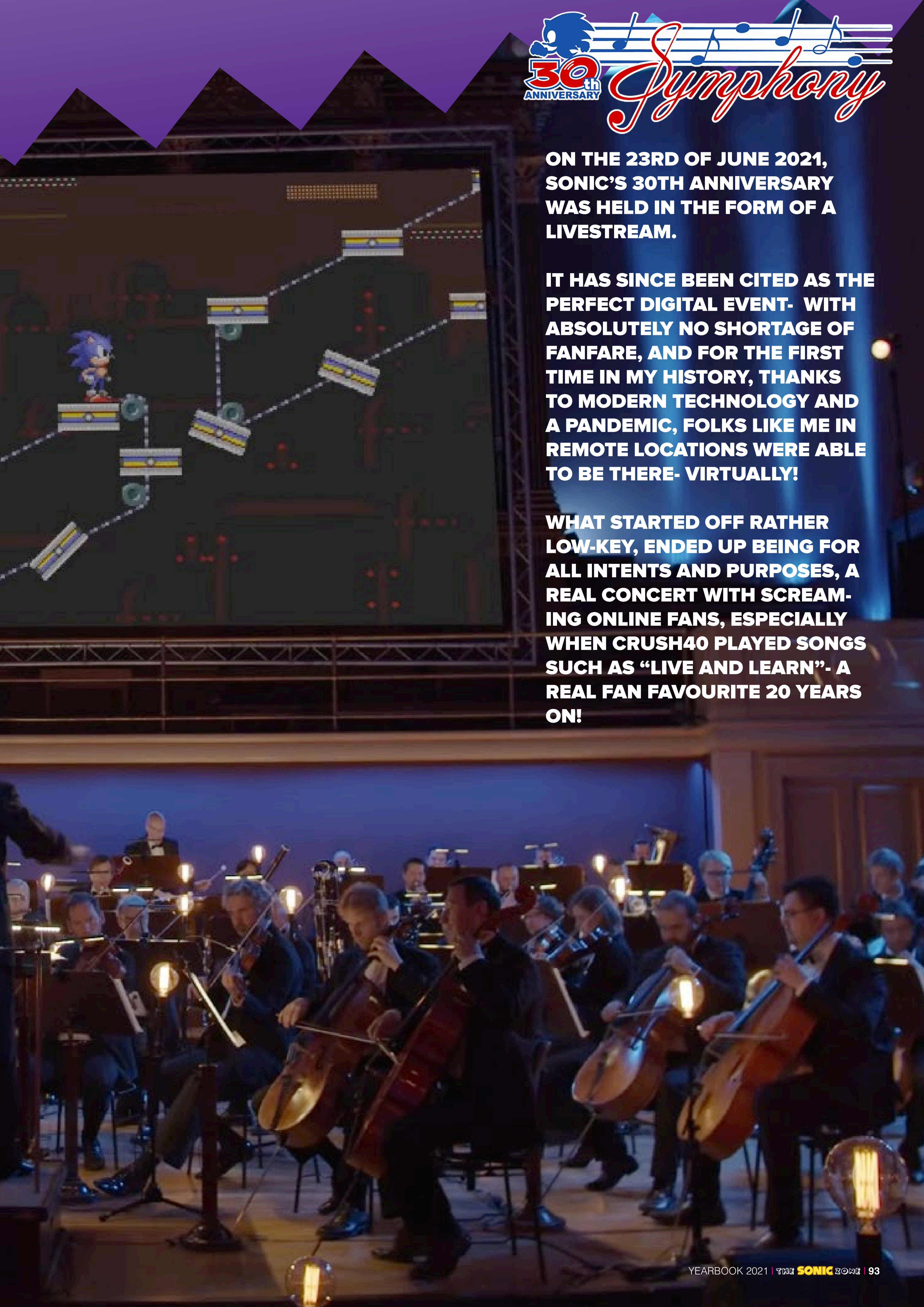
JOSE, YOU'VE GOT THIS ONE, BUDDY!
LL AS U THE S
WE WISH YOU ALL THE BEST ;)
GET WELL SOON DEAR FRIEND
GET WELL SOON UNCLE JOSE FROM ME AND MY MAMA, VANILLA THE RABBIT

Events, Fan Interaction, and more...

Miscellany



30th ANNIVERSARY Symphony



ON THE 23RD OF JUNE 2021, SONIC'S 30TH ANNIVERSARY WAS HELD IN THE FORM OF A LIVESTREAM.

IT HAS SINCE BEEN CITED AS THE PERFECT DIGITAL EVENT- WITH ABSOLUTELY NO SHORTAGE OF FANFARE, AND FOR THE FIRST TIME IN MY HISTORY, THANKS TO MODERN TECHNOLOGY AND A PANDEMIC, FOLKS LIKE ME IN REMOTE LOCATIONS WERE ABLE TO BE THERE- VIRTUALLY!

WHAT STARTED OFF RATHER LOW-KEY, ENDED UP BEING FOR ALL INTENTS AND PURPOSES, A REAL CONCERT WITH SCREAMING ONLINE FANS, ESPECIALLY WHEN CRUSH40 PLAYED SONGS SUCH AS "LIVE AND LEARN"- A REAL FAN FAVOURITE 20 YEARS ON!

HIWATT

When I wanted to write a little bit about Sonic's 30th I thought I would have a lot to write about. It turns out, however, that nothing I can write nor say about it, would do it justice.

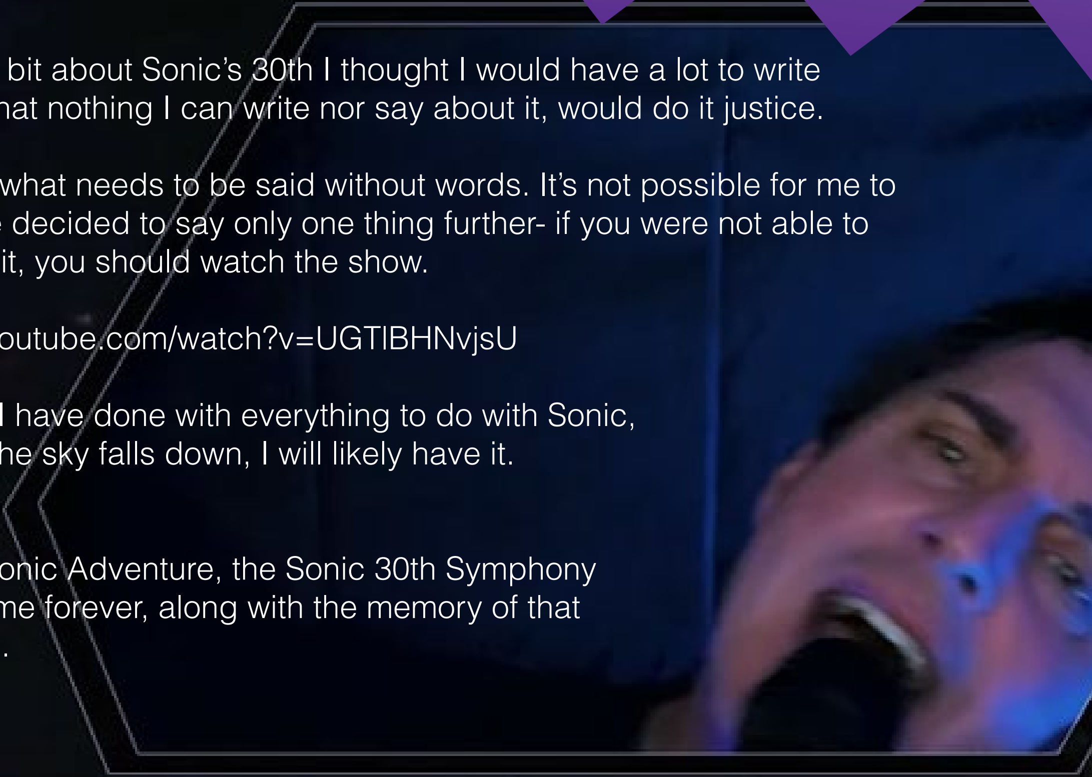
These images here, they say what needs to be said without words. It's not possible for me to say much more, hence I have decided to say only one thing further- if you were not able to see it live or haven't yet seen it, you should watch the show.

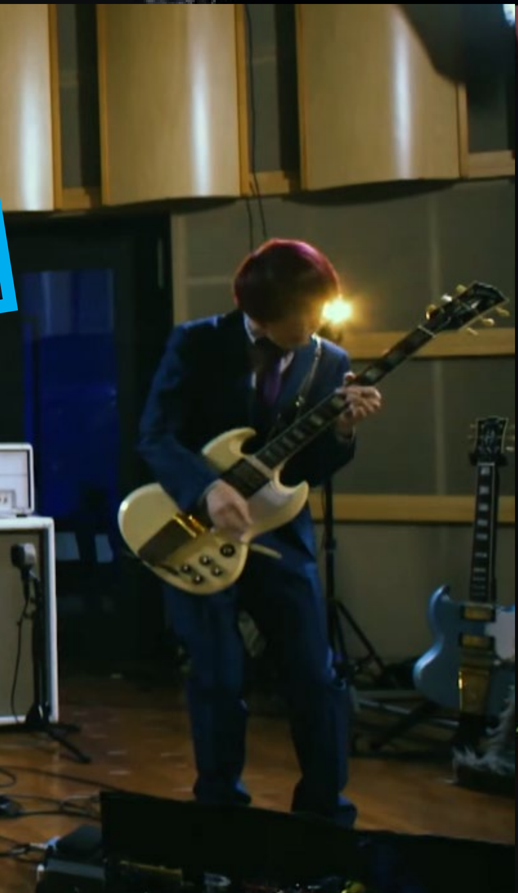
Its on YouTube: https://www.youtube.com/watch?v=UGTIBHNvjsU

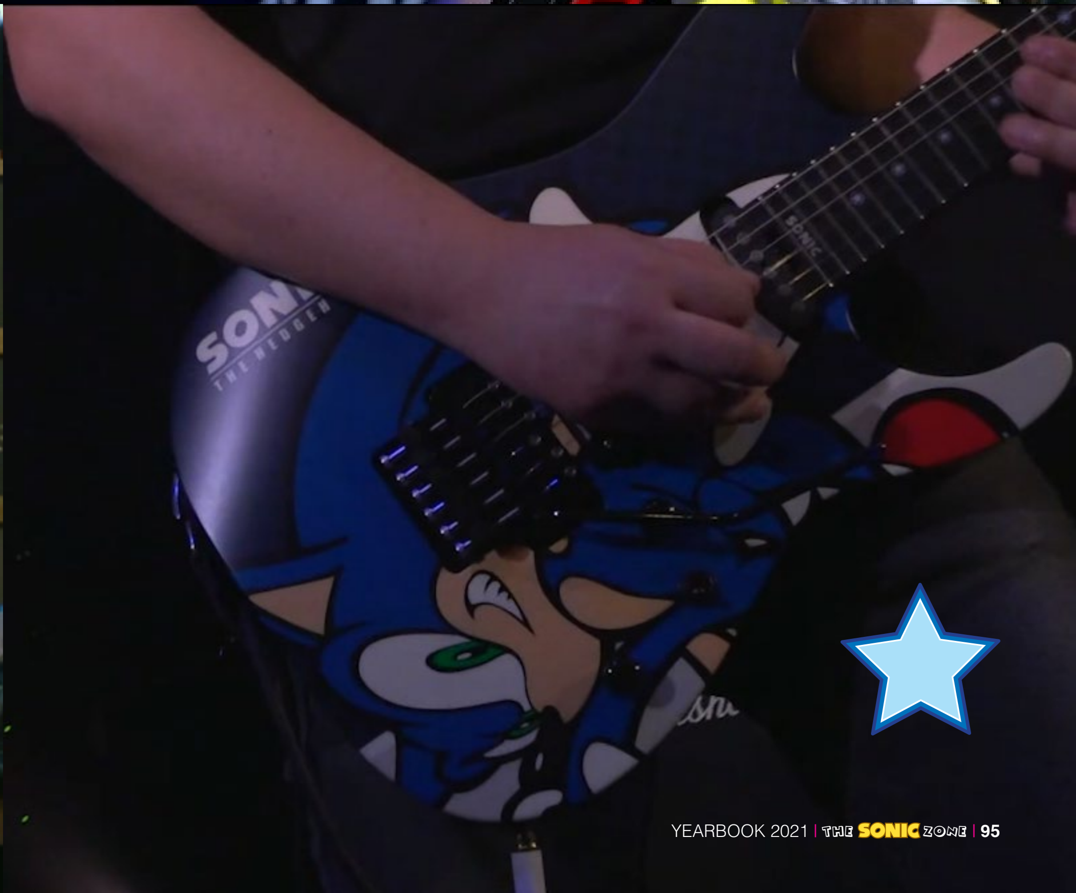
Of course I've archived it, as I have done with everything to do with Sonic, so if YouTube goes under or the sky falls down, I will likely have it.

Without a doubt, along with Sonic Adventure, the Sonic 30th Symphony is a memory that will be with me forever, along with the memory of that fateful day in November 1999.

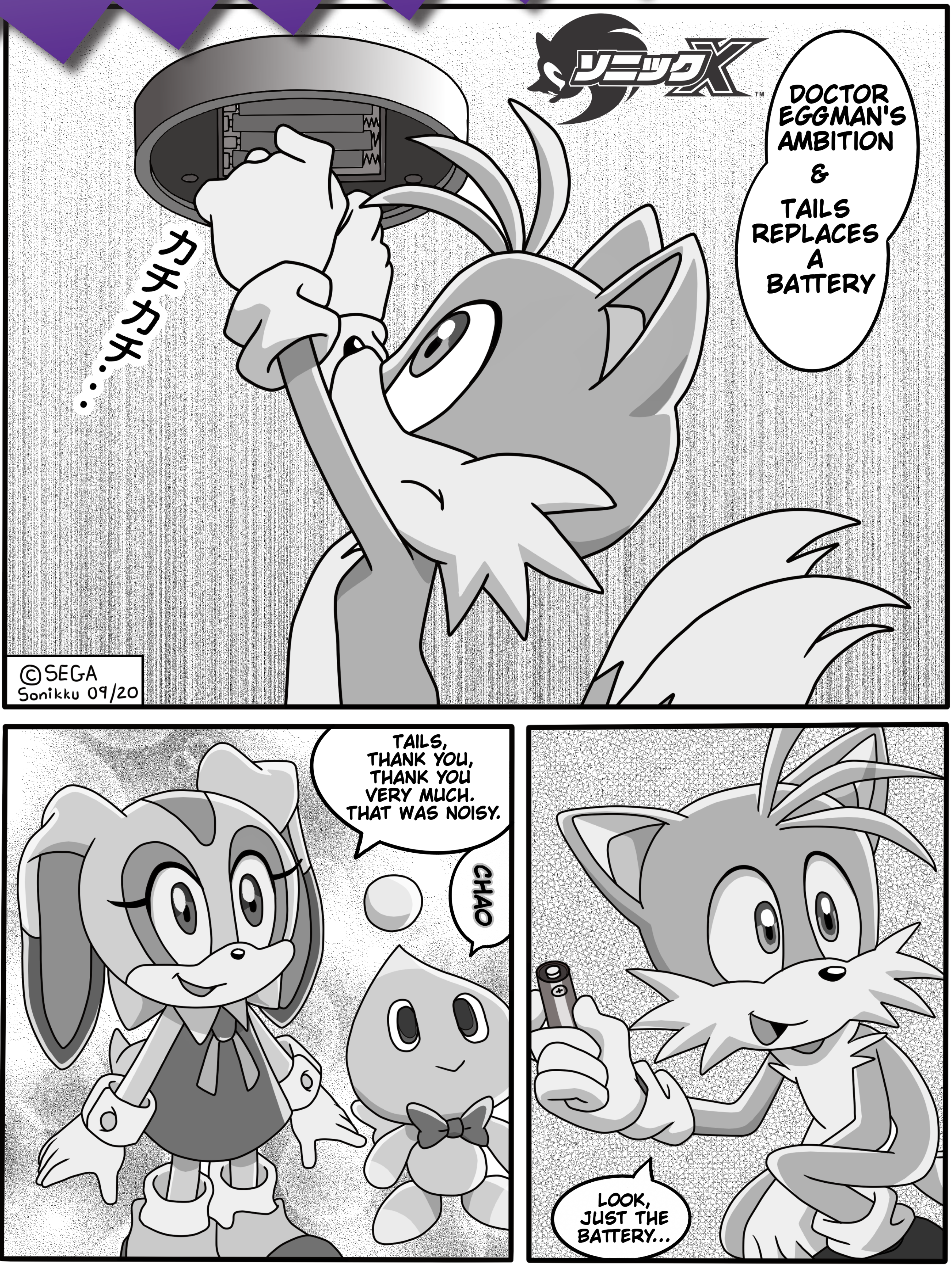
Sonic2k, October 2021

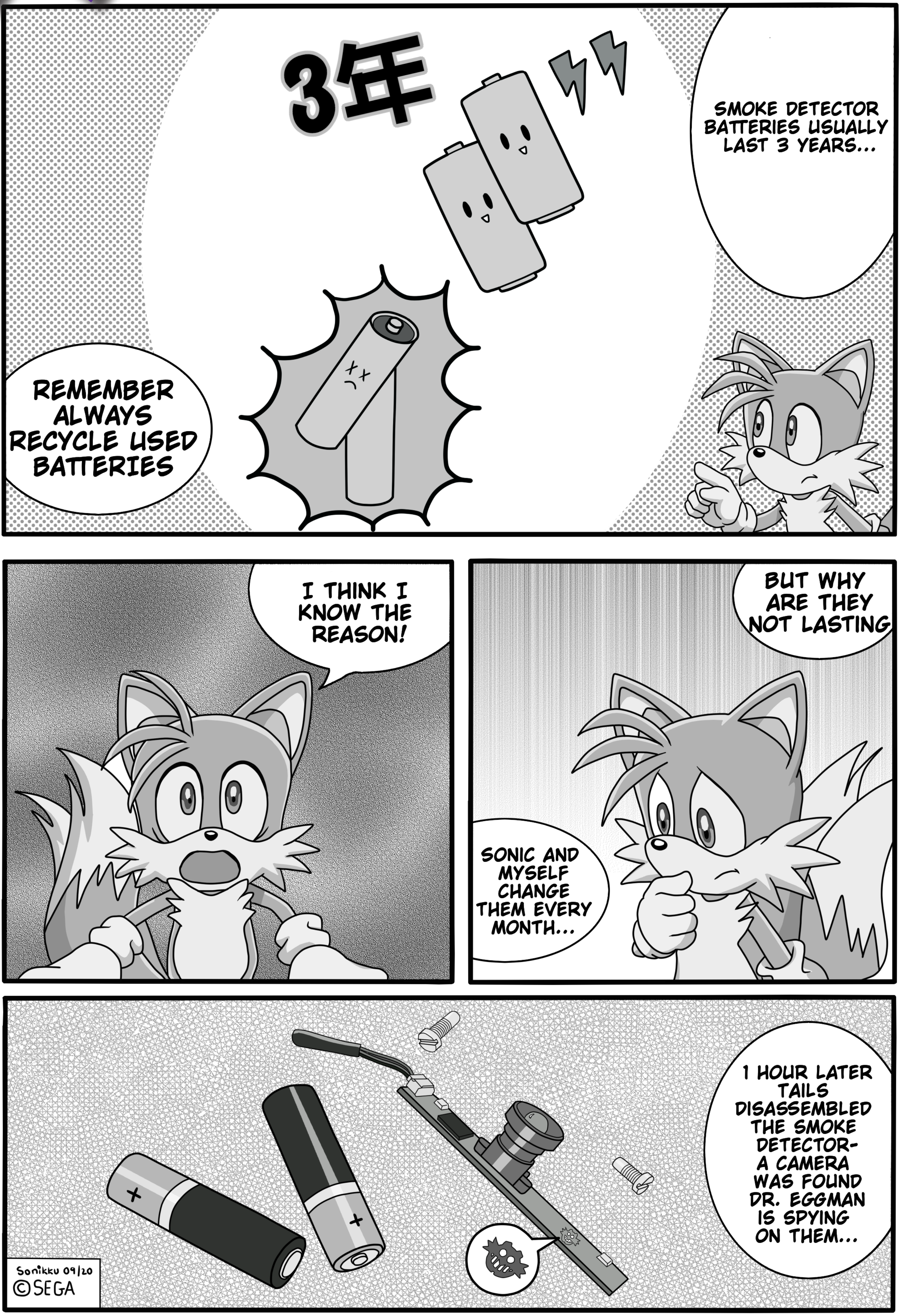


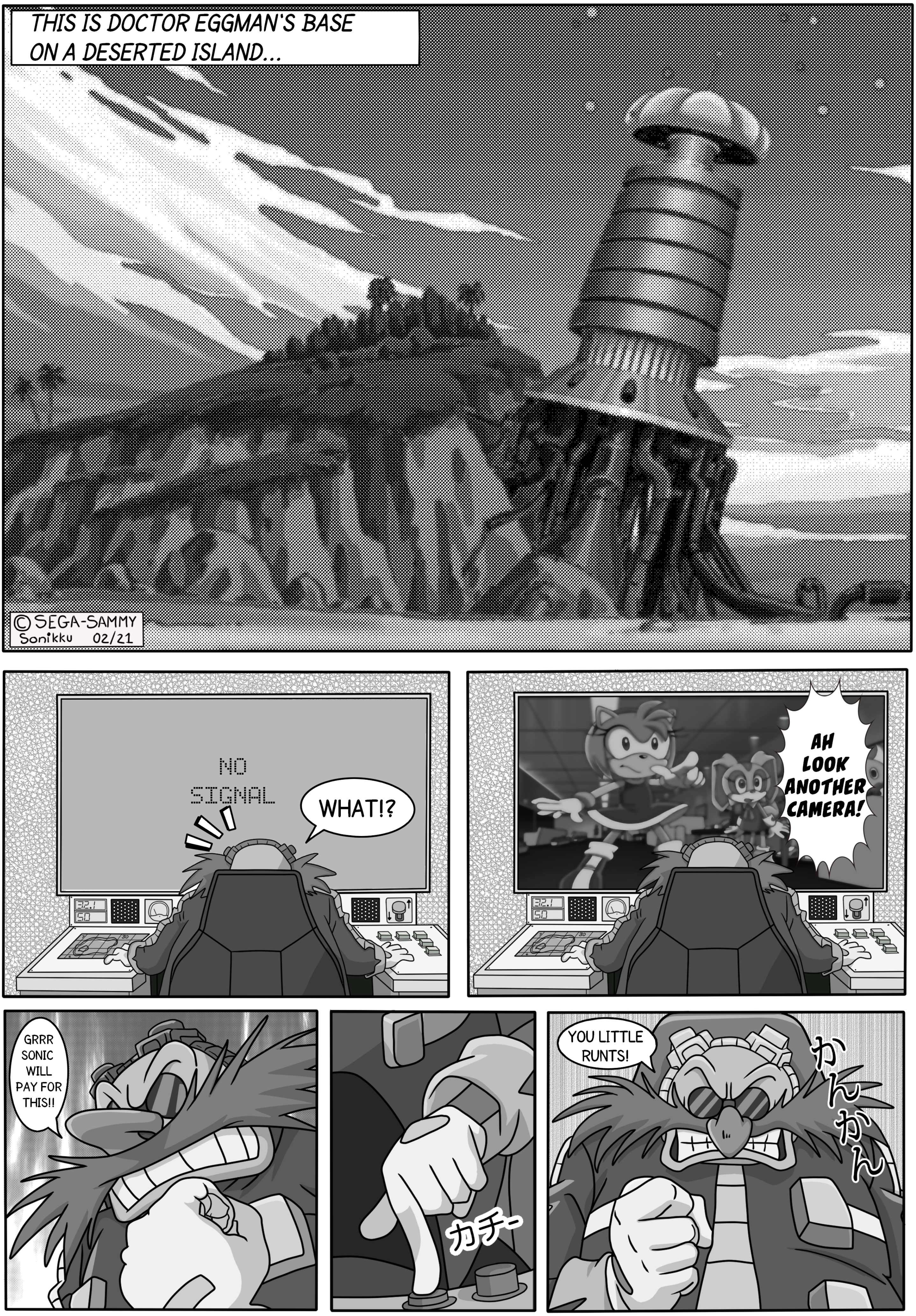


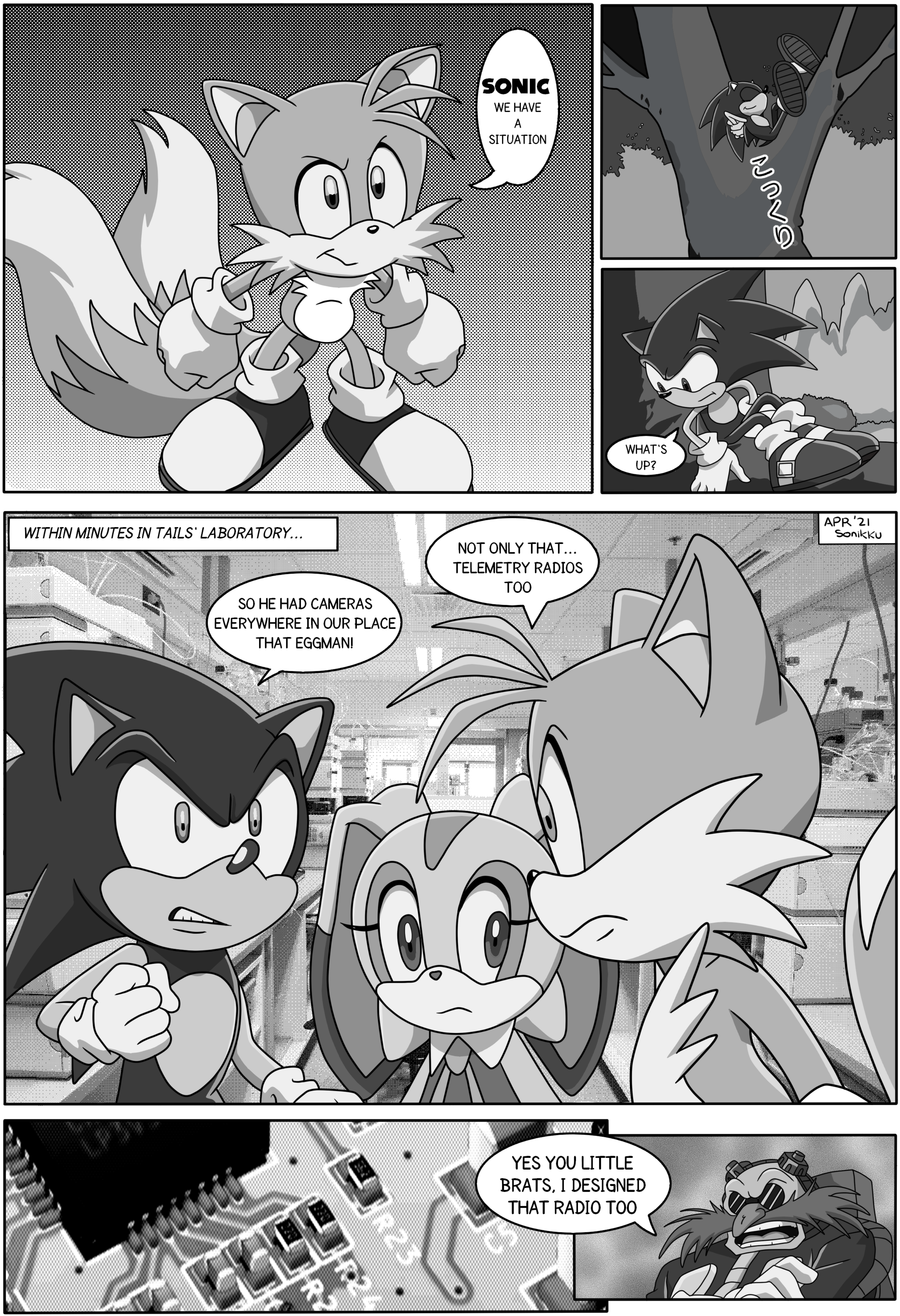


Sonic X manga - English Ver.


ソニックX
DOCTOR EGGMAN'S AMBITION & TAILS REPLACES A BATTERY
カチカチ・・・
©SEGA Sonikku 09/20
LOOK, JUST THE BATTERY...
TAILS, THANK YOU, THANK YOU VERY MUCH. THAT WAS NOISY.
CHAO


SMOKE DETECTOR BATTERIES USUALLY LAST 3 YEARS...
3年
REMEMBER ALWAYS RECYCLE USED BATTERIES
BUT WHY ARE THEY NOT LASTING
SONIC AND MYSELF CHANGE THEM EVERY MONTH...
I THINK I KNOW THE REASON!
1 HOUR LATER TAILS DISASSEMBLED THE SMOKE DETECTOR- A CAMERA WAS FOUND DR. EGGMAN IS SPYING ON THEM...
Sonikku 09/20
©SEGA

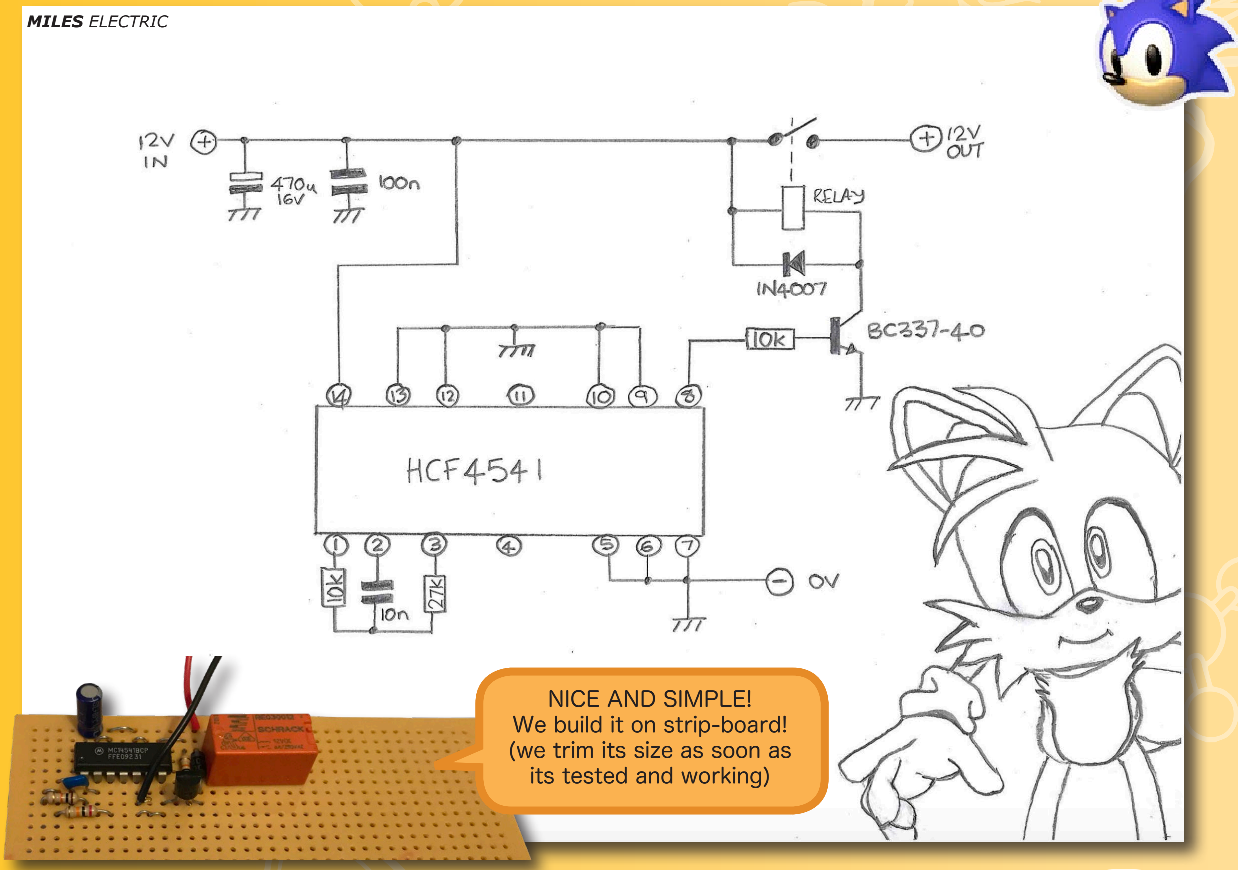

THIS IS DOCTOR EGGMAN'S BASE ON A DESERTED ISLAND...
© SEGA-SAMMY
Sonikku 02/21
NO SIGNAL
WHAT!?
AH LOOK ANOTHER CAMERA!
GRRR SONIC WILL PAY FOR THIS!!
カチー
YOU LITTLE RUNTS!
かんかん


SONIC WE HAVE A SITUATION
むっくり
WHAT'S UP?
WITHIN MINUTES IN TAILS' LABORATORY...
APR '21 Sonikku
SO HE HAD CAMERAS EVERYWHERE IN OUR PLACE THAT EGGMAN!
NOT ONLY THAT... TELEMETRY RADIOS TOO
YES YOU LITTLE BRATS, I DESIGNED THAT RADIO TOO

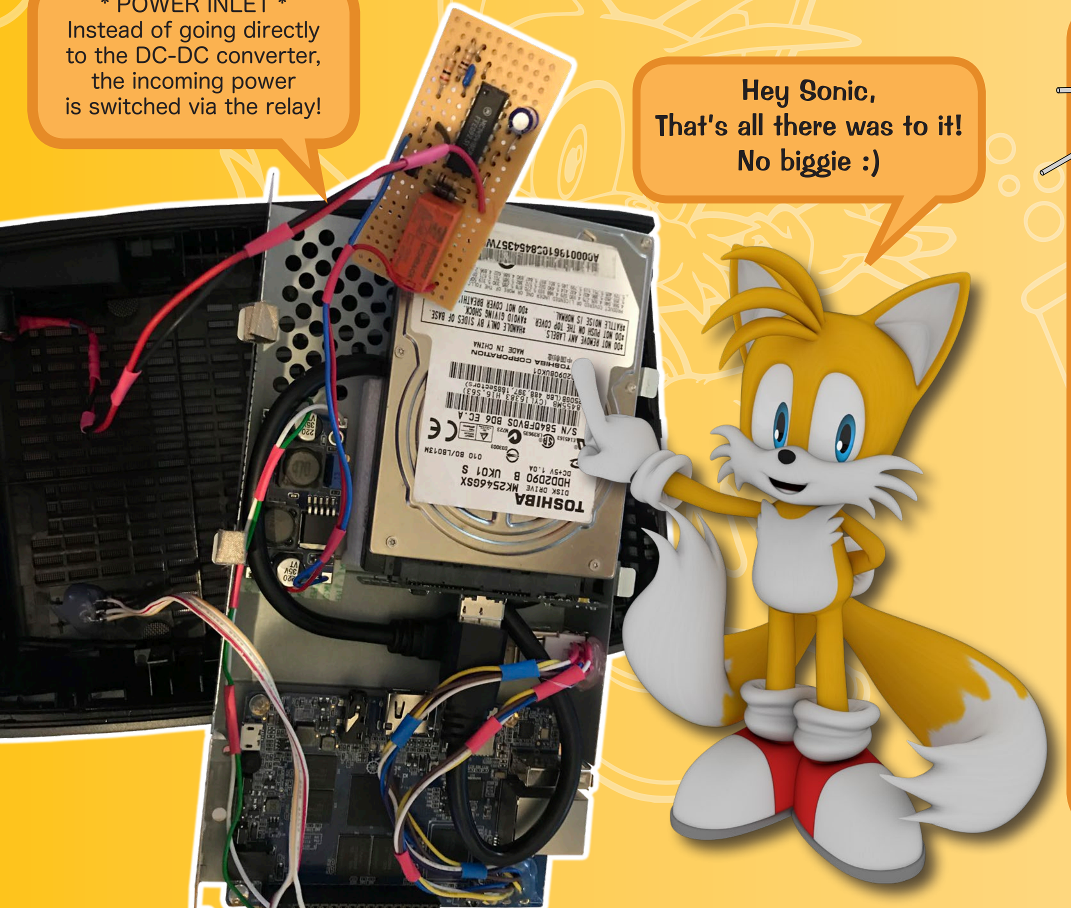
© Pierre Roussel
https://www.artstation.com/pierreroussel

Peak Current Modification for SEAGATE NAS

The following is a modification for the NAS to allow the wall adapter (or wall wart) to cope with high peak current draw at power on. This handy dandy circuit gives the wall adapter enough time to stabilize before connecting the load. This is necessary as some hard drives have large start-up currents. This little circuit will delay switching the power to the rest of the system by a few seconds, enough to give the power supply time to settle and be ready to supply large currents.



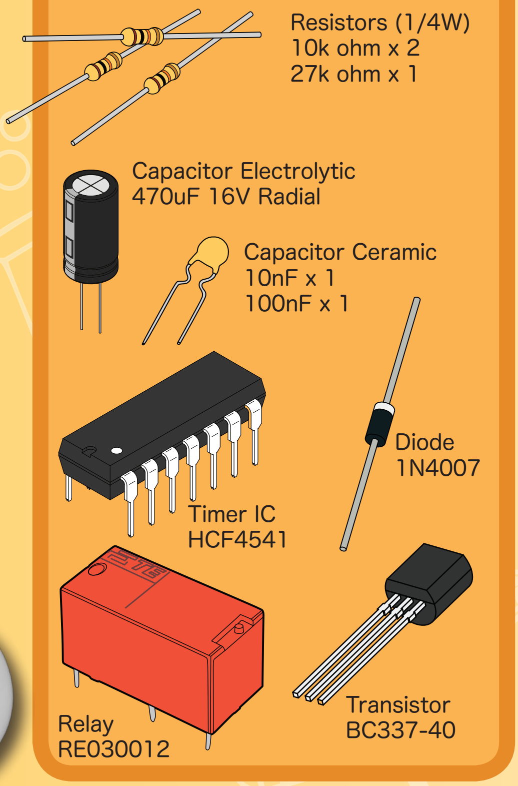
NICE AND SIMPLE!
We build it on strip-board!
(we trim its size as soon as its tested and working)



* POWER INLET *
Instead of going directly to the DC-DC converter, the incoming power is switched via the relay!

Hey Sonic,
That's all there was to it!
No biggie :)

Parts



Resistors (1/4W)
10k ohm x 2
27k ohm x 1

Capacitor Electrolytic
470uF 16V Radial

Capacitor Ceramic
10nF x 1
100nF x 1

Timer IC
HCF4541

Diode
1N4007

Relay
RE030012

Transistor
BC337-40

COLOPHON

Sometimes it takes a global pandemic to get me off my backside to actually bring the ideas that I have had for ten or more years, to fruition!

STUFF THAT HELPED

Lots of Japanese music (YMO, PLASTICS, OFF COURSE); Sonic X; Animaniacs (1992 version and the reboot); Pinky and the Brain; The complete guide to Asperger's Syndrome by Tony Attwood; David Thorne; Sylvester my cat; my dearest family, and certainly last but not least, my employer for not laying me off like so many companies out there have done because their CEOs would rather lay people off than have to sell the Ferrari 'cause of lockdowns hurting profits.

MESSAGE TO SEGA-SAMMY HOLDINGS

Since this is an unofficial fan project I sincerely apologize for shamelessly using SEGA-SAMMY's assets/IP. But SEGA, I am sure you agree, this is proof enough of pure passion for the franchise. You have consistently ignored my requests for licensing/permission because of reasons unknown and I have given up sending letters and emails, I pushed ahead regardless!
それを説明してもらえますか？

SPECIAL THANKS

NVIDIA Corporation (nvidia.com)
Texas Instruments (Rodney Farrow)
Thomas Hancock (Code / Project Review)
Jose Lizemore (Artistic advice and direction)

Micro Robotics supplied many of the gadgets & parts.
Paint Tool SAI was used for the bare manga sketches.
Medibang Paint Pro finished and typeset the manga.
All manga and other drawings created on a WACOM tablet
Github (github.com)
Computers by HP® and Apple®
Monitors by DELL®
Headphones by SkullCandy™
Camera used: Apple iPhone™ 7
Cosmic Comics supplied the Sonic Diorama Statue
Sonic Apparel from FACTORIE, H&M
GNU Linux
OrangePi + 2E [Zhao Yifan]
Printed circuit boards by W.H. CIRCUIT (PTY) LTD
SHEERHOSTING
VOX TELECOM
publuu.com

Tails & Dr. Eggman use MAGNUM™ soldering irons, WIHA® screwdrivers, Hellermann-Tyton® hand tools & cable accessories, Techmet NO CLEAN solder, Techmet liquid flux

THE ソニックゾーン SONIC ZONE™