Supersonic Retro Development with Docker

FOSDEM – Retrocomputing Devroom
2nd February 2025

by Steven Goodwin

@marquis de geek

https://em.ulat.es/
https://github.com/MarquisdeGeek/ZXDocker
https://github.com/MarquisdeGeek/DragonDocker

Introduction

- About Docker
- ZX Spectrum example
- Building and running
- Developing inside
- Conclusions

Who am I?

The Ego Slide

Who am I?

- General-purpose developer (JavaScript and C/C++/Actionscript/Assembler)
    - Games on PC, online, consoles, mobile
    - Cloud infrastructure with AWS and Azure
    - Education Technology
- Author – six books, including “20 GOTO 10”
- Composer and musician
- Magician
- Retro-computing enthusiast
- Open source advocate
    - FOSDEM
    - Keynote speaker

About Docker

About Docker

- “Docker is a software platform that helps developers build, test, and deploy applications quickly. It's an open-source platform that uses containers to package software into standardized units”

ZX Spectrum example

ZX-ample

ZX-ample - Dockerfile

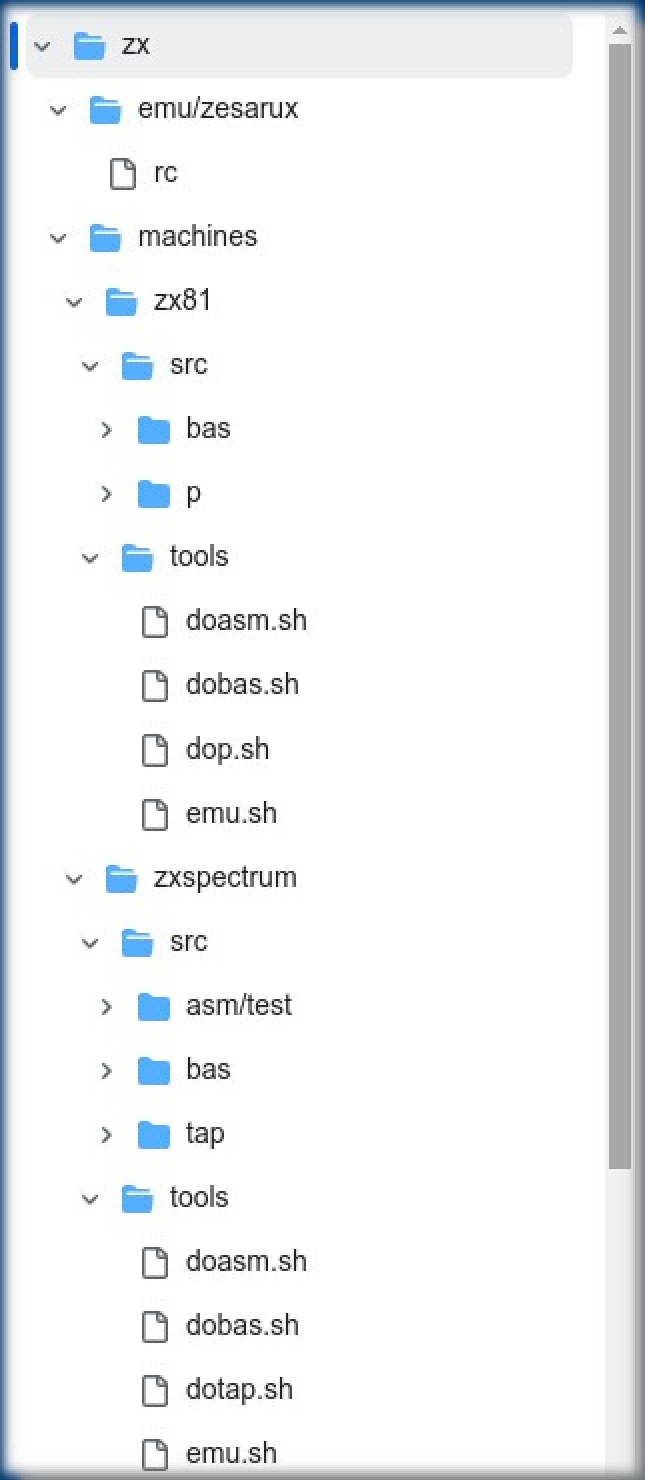
```
FROM ubuntu:22.04
LABEL Version="1.0" \
      Date="2024-Feb-24" \
      Docker_Version="Docker version 25.0.3, build 4debf41" \
      Vendor="em.ulat.es" \
      Maintainer="Steven Goodwin - Marquis de Geek (@marquisdeGeek)" \
      Description="A basic Docker container to compile and use ZX Spectrum tools"

ARG   USER_NAME=user
ARG   USER_PASSWORD=${USER_NAME}
ARG   USER_PATH=/home/${USER_NAME}


# Create a non-root user
RUN useradd -ms /bin/bash $USER_NAME && echo "$USER_NAME:$USER_PASSWORD" | chpasswd && adduser $USER_NAME sudo


# Base OS and tools
RUN apt-get update && \
    apt-get install -y wget unzip git tcc sudo vim && \
    apt-get remove --purge --auto-remove -y
```

ZX-ample – Dockerfile (continued)

```
# The ZX env
USER $USER_NAME
RUN cd $USER_PATH && \
    mkdir -p zx && \
    mkdir -p zx\zx81 && \
    mkdir -p zx\zxspectrum


# Emu: zesarux
USER root
RUN apt-get update && \
    apt-get install -y spectemu-common spectemu-x11 spectrum-roms zmakebas libxxf86vm1 libsdl1.2debian && \
    apt-get remove --purge --auto-remove -y

USER $USER_NAME
RUN cd $USER_PATH/zx && \
    mkdir contrib && cd contrib && \
    wget -O zx.tar.gz https://github.com/chernandezba/zesarux/releases/download/ZEsarUX-X/ZEsarUX_linux-X-ubuntu22_x86_64.tar.gz && \
    tar zfx zx.tar.gz && \
    rm zx.tar.gz

COPY zx/emu/zesarux/rc /$USER_PATH/.zesaruxrc
```

ZX-ample – Dockerfile (the magic bit)

```
USER $USER_NAME
COPY zx/machines/zx81/tools /$USER_PATH/zx81
COPY zx/machines/zxspectrum/tools /$USER_PATH/zxspectrum

WORKDIR $USER_PATH/zx/machines

# Keeps the container open
CMD [ "/bin/bash" ]
```



```
zx
  emu/zesarux
    rc
  machines
    zx81
      src
        bas
        p
      tools
        doasm.sh
        dobas.sh
        dop.sh
        emu.sh
    zxspectrum
      src
        asm/test
        bas
        tap
      tools
        doasm.sh
        dobas.sh
        dotap.sh
        emu.sh
```

How to build a docker image

```
$ ./0-build.sh
```

Or

```
$ docker build -t emulates/zx .
```

How to run a docker image

```
$ ./1-run.sh
```

Or

```
$ docker run -i -e DISPLAY=:0.0 --device /dev/snd -v /dev/snd:/dev/snd  -v /tmp/.X11-unix/:/tmp/.X11-unix/  -v ./zx/machines:/home/$USER_NAME/zx/machines -t emulates/zx /bin/bash
```

When inside a Docker image

```
$ ./zx81/tools/doasm.sh mycode.a
```

- It knows the assembler binary, and flags
- Because it’s assembler, it knows the code is in src/asm
- It builds the code to an appropriate out directory
- It runs the emulator, with the output binary
- So… edit in VSCode/Sublime/vi on the host, and run in Docker

When inside a Docker image

```
$ ./zx81/tools/dobas.sh mycode.bas
```

- Same thing, but convert text files of BASIC code into images

Extras

- Why just one Docker?

Extras

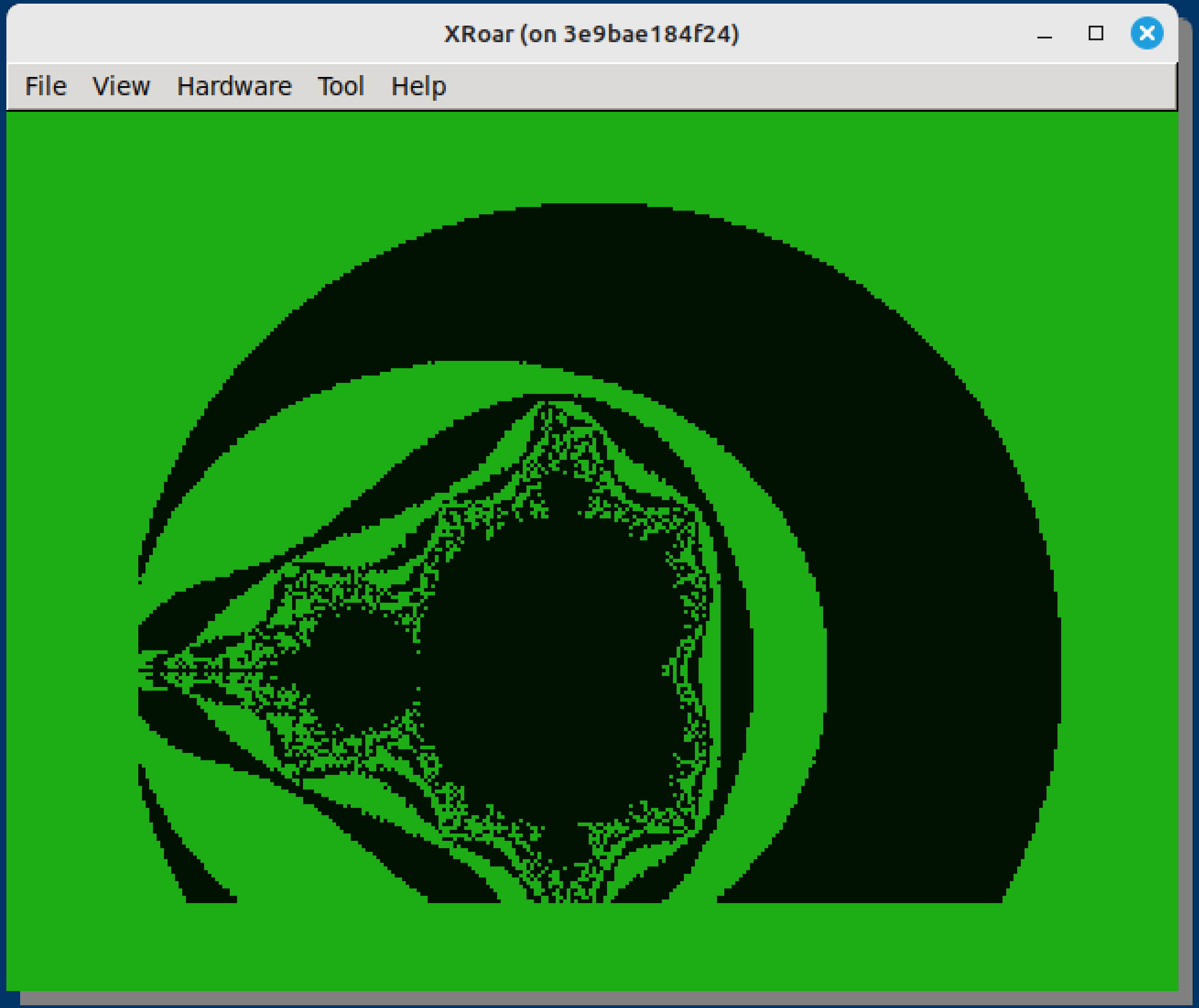
- Why just one Docker?
- Z88dk, a C compiler for the Z80, has a Dockerfile
- a) Edit on host machine
- b) Compiler in the Z88dk container
- c) Copy files from one container to another
- d) Run the emulator tools from the ZXDocker container
- ./xfer.sh script to help

Enter the Dragon

https://github.com/MarquisdeGeek/DragonDocker

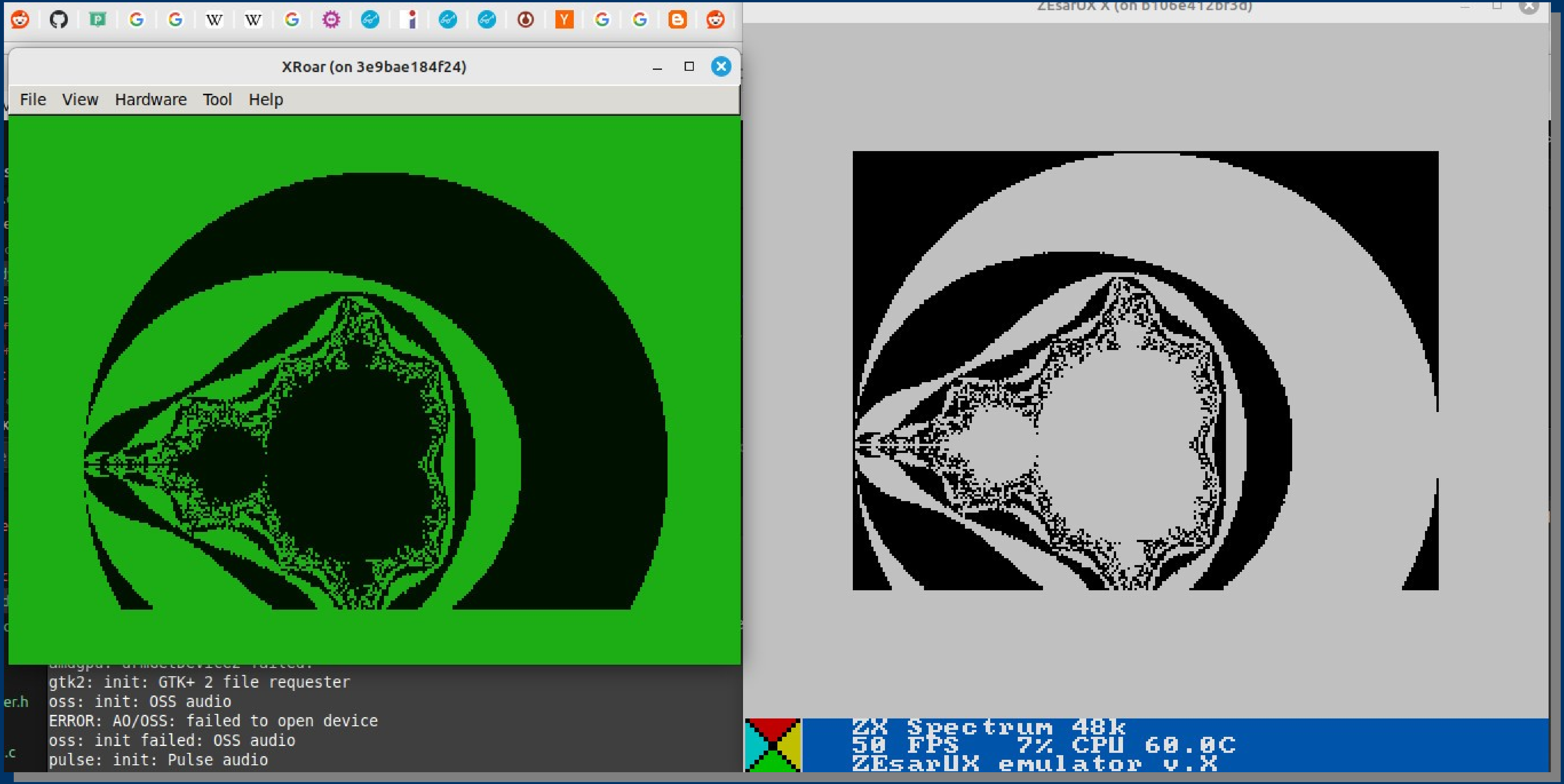
Enter the Dragon

- DragonDocker is the same
- ...but has a C compiler incorporated
- ...and some extra examples



https://github.com/MarquisdeGeek/DragonDocker/tree/main/dragon/src/c/mandelbrot

And finally...



```
amdgpu: drmGetDevice2 failed.
gtk2: init: GTK+ 2 file requester
oss: init: OSS audio
ERROR: AO/OSS: failed to open device
oss: init failed: OSS audio
pulse: init: Pulse audio
```

```
ZX Spectrum 48k
50 FPS    7% CPU 60.0C
ZEsarUX emulator v.X
```

Conclusions

- Docker gives us a fixed environment
- It’s lightweight enough to have individual tools in their own images
- Supersonic turn-around times
- Cross-platform C development, on 8-bit platforms, is now trivial

Any Questions?

@marquis de geek

www.MarquisdeGeek.com

FOSDEM Scorecard:

Attended: 25
Diaries written:20
Talks given: 25 (on 23 different topics)
Devroom Diversity: 12

https://em.ulat.es/
https://github.com/MarquisdeGeek/ZXDocker
https://github.com/MarquisdeGeek/DragonDocker

@marquis de geek

Any Questions?

@marquis de geek

www.MarquisdeGeek.com

FOSDEM Scorecard:

Attended: 25
Diaries written:20
Talks given: 26 (on 24 different topics)
Devroom Diversity: 12

https://em.ulat.es/
https://github.com/MarquisdeGeek/ZXDocker
https://github.com/MarquisdeGeek/DragonDocker

@marquis de geek