Pyromaniac: Evolution

Adapt, Evolve or Die

Gerph, November 2025

0. Introduction

Introduction

What I'll talk about

I'll be talking about ...

1. Recap on RISC OS Pyromaniac.
2. Towards 64-bit RISC OS.
3. Changing my approach.
4. Conclusion.

Hopefully I won't be *too* technical, but there will be examples of running code.

Introduction

Who am I?

- I'm Charles, but known as Gerph in most things online.
- I worked at RISCOS Ltd and produced RISC OS Select.
- I ported almost all of RISC OS to be 32 bit.
- I've written the only other implementation of RISC OS - RISC OS Pyromaniac - from scratch.
- I'm a strong advocate of taking RISC OS forward, rather than letting it stay stagnant.

1. Recap on RISC OS Pyromaniac

Recap on RISC OS Pyromaniac

What is RISC OS Pyromaniac? (1)

- An implementation of RISC OS.
- Intended for testing, development, debugging and experimentation.
- A system that helps you to try things out.
- Runs on modern systems.
- Written in high level language.

Recap on RISC OS Pyromaniac

What is RISC OS Pyromaniac? (2)

- Easy to run things without hardware or system emulators.
- Provides many debug options to explain what is happening in the system.
- Allows a full disassembly of every instruction.
- High level language means changing behaviour is easier.
- Cloud systems mean that automated testing is possible.
- Use the advanced editors, IDEs and AI to develop your code.

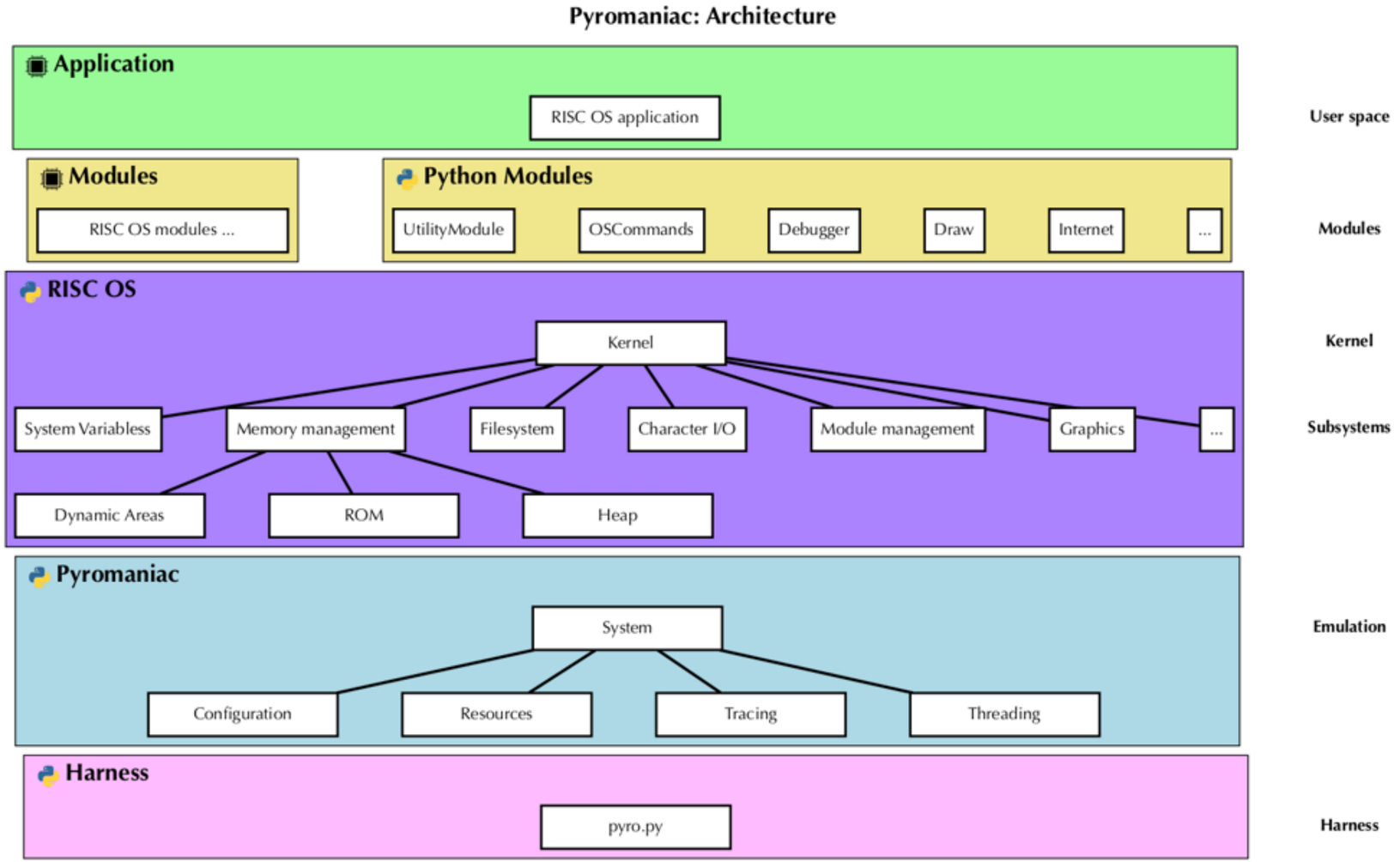
Recap on RISC OS Pyromaniac

What's under the hood? (1)

- RISC OS Pyromaniac implements documented RISC OS interfaces.
- Doesn't implement everything - depends on what you're testing.
- Doesn't have hardware - no memory mapped devices.
- Does implement RISC OS interfaces to drivers.
- OS is written in Python.
- Emulation provided by Unicorn - a QEmu-derived system.

Recap on RISC OS Pyromaniac

What's under the hood? (2)



Recap on RISC OS Pyromaniac

Why do this?

- I enjoy it.
- I want to make it easier to develop software.
- I want to enable others to do things.
- I want to be able to use modern techniques for development.

... and from that, it's grown far beyond my original goals.

RISC OS Pyromaniac demo

2. Towards 64-bit RISC OS

Towards 64-bit RISC OS

What's changed? (1)

Lots of things!
The detail is in the Pyromaniac changelog on the website.
Most of it relates to 64-bit work.

Towards 64-bit RISC OS

What's changed? (2)

Towards 64-bit RISC OS

What driven the changes?

- Testing with a broad selection of applications found bugs.
- Porting old games made improvements to the system.
- Trying out different ways of doing things suggested better debug and implementations.
- Experimenting with architectures produced 64-bit RISC OS.

Towards 64-bit RISC OS

6502 emulation

- I wanted to try out a 6502 emulator - running a system like the BBC Micro.
- Updated an existing open source emulator to use Unicorn-like interfaces.
- Restructured interfaces so that trapping system calls was easier.

I wanted to make it possible to run 6502 emulation inside RISC OS Pyromaniac.

- If you can run 6502 with RISC OS Pyromaniac, anything is possible.
- It's an intermediate step towards AArch64 or x64.

But actually it's a bit too alien to make work.

Towards 64-bit RISC OS

Replacing ARM with AArch64

- Update the disassembler system to support Thumb.
- Then update to support AArch64.
- Introduce DebuggerPlus features - widely used, and we don't want to clash.
- Debugger also supports the exception dump.
- Making the exception dump descriptive allows for other architectures.

Towards 64-bit RISC OS

Describing CPU registers

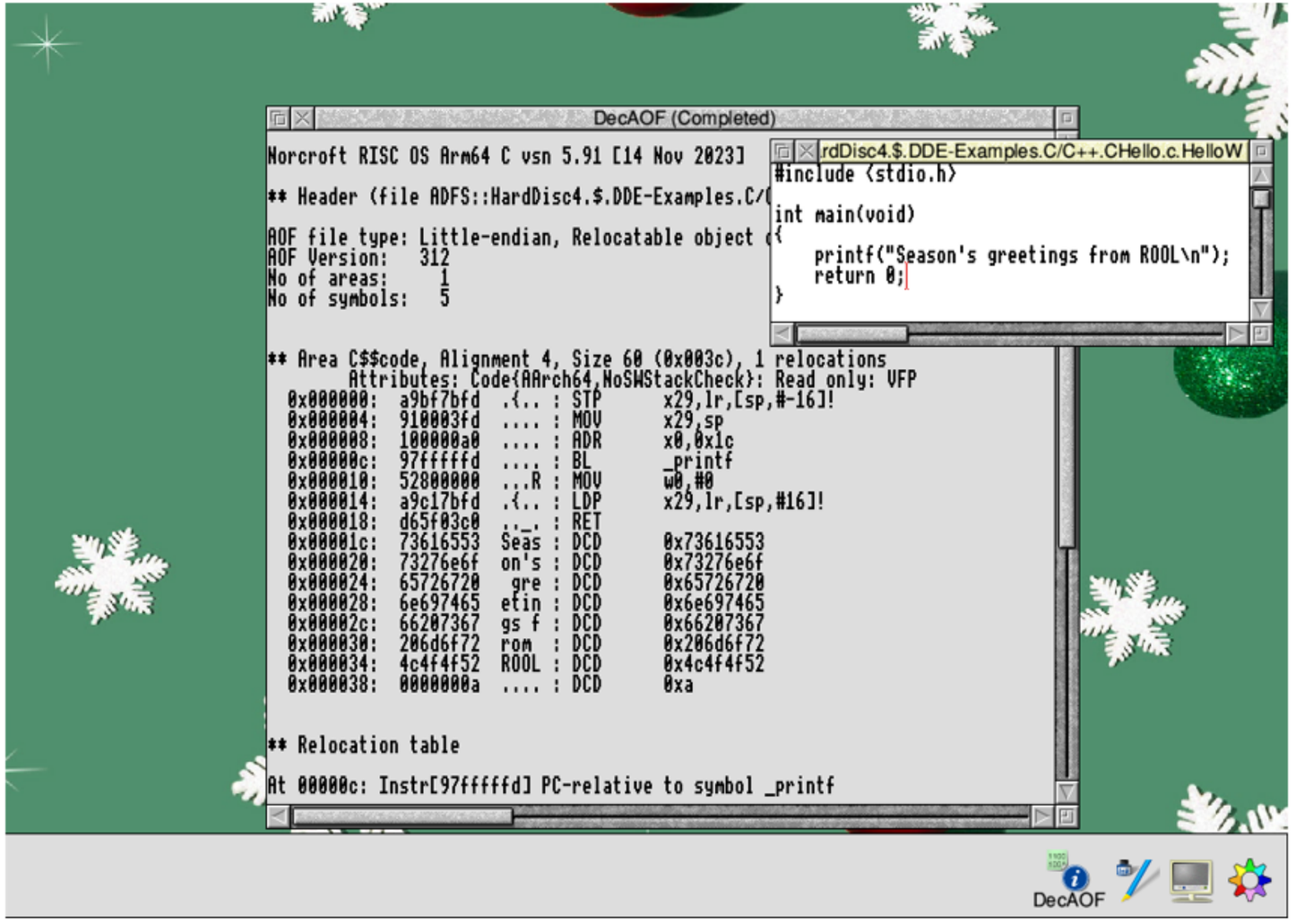
`OS_PlatformFeatures 64` describes the exception dump layout, to address the general problem for the future. The dump layout describes the architecture identifier, the length of instructions and the size of the dump block itself. And then for each register, we describe:

- The register's name.
- How offset the data is stored at.
- How wide the register is in bits.
- The alignment (eg the stack in AArch64 must be aligned to 4 bits).

For flags, we also describe where the flag is in a register, and its meaning.

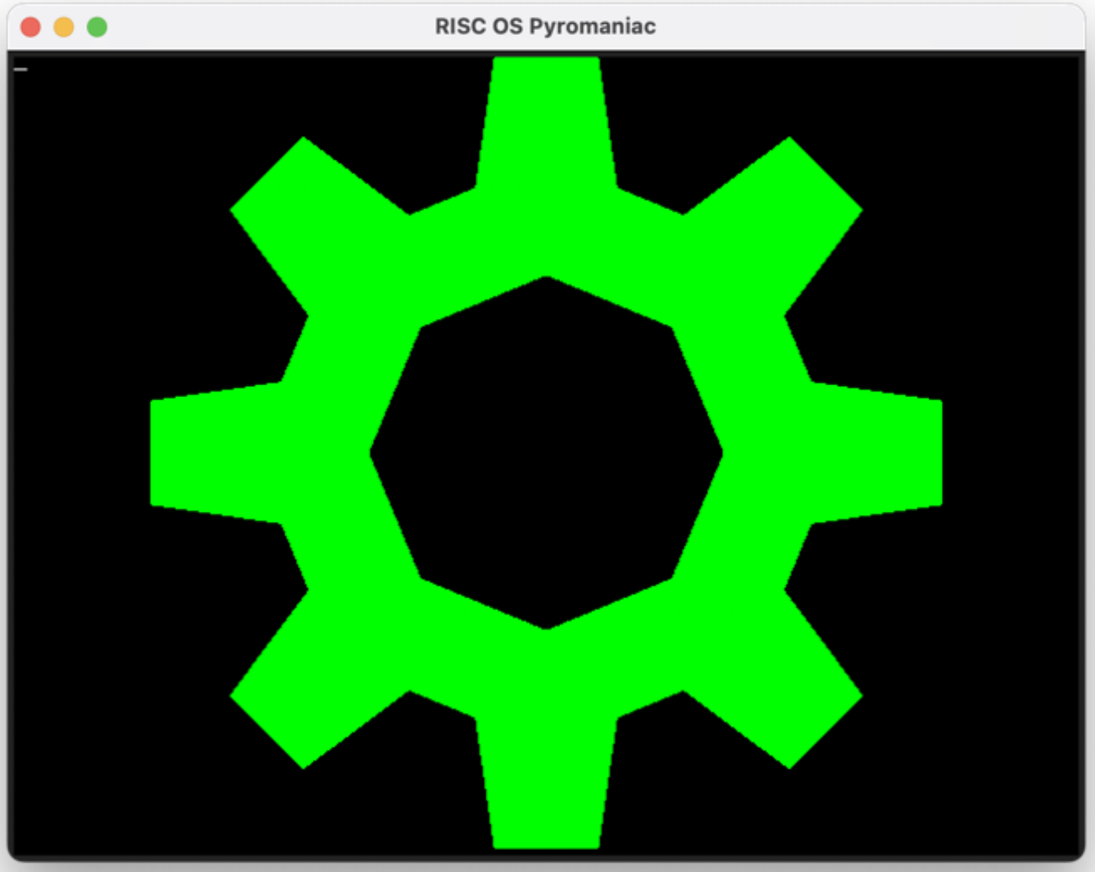
Towards 64-bit RISC OS

Christmas calendars 2023



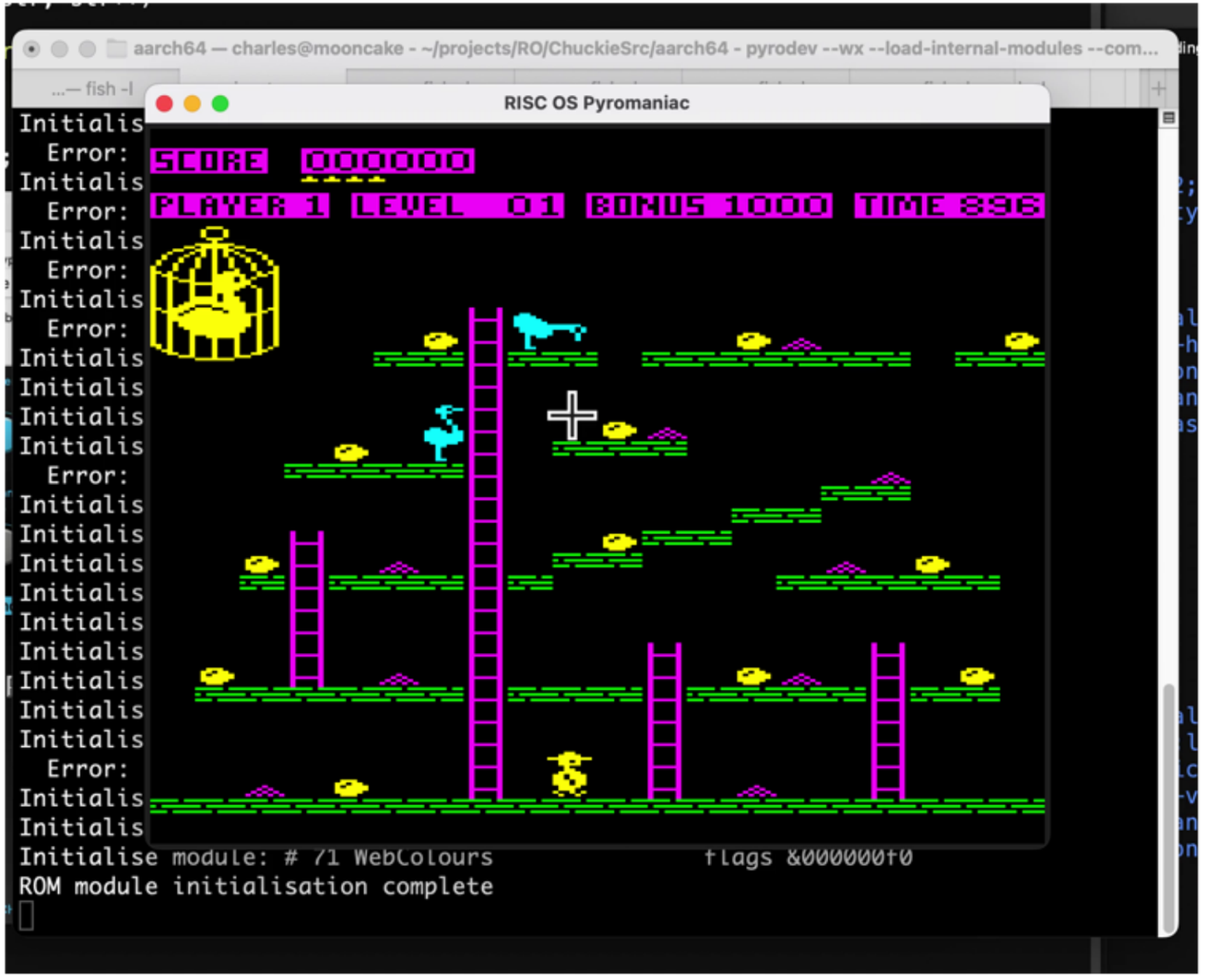
Towards 64-bit RISC OS

The first AArch64 RISC OS program



Towards 64-bit RISC OS

The first AArch64 RISC OS game



Towards 64-bit RISC OS

Presenting 64-bit RISC OS

- Those programs were made open source, on GitHub.
- A C library for 64-bit RISC OS was created on GitHub.
- I gave a presention on what I had done and decisions I had made at ROUGOL in August.
- I encouraged people that this was something we could do.

Presentation resources

Towards 64-bit RISC OS

Creating a build environment

- All the 32-bit tools and libraries were already built into a Docker image for easier use.
- Extending this to 64-bit tools made the system much more useful.
- Took many, many iterations!

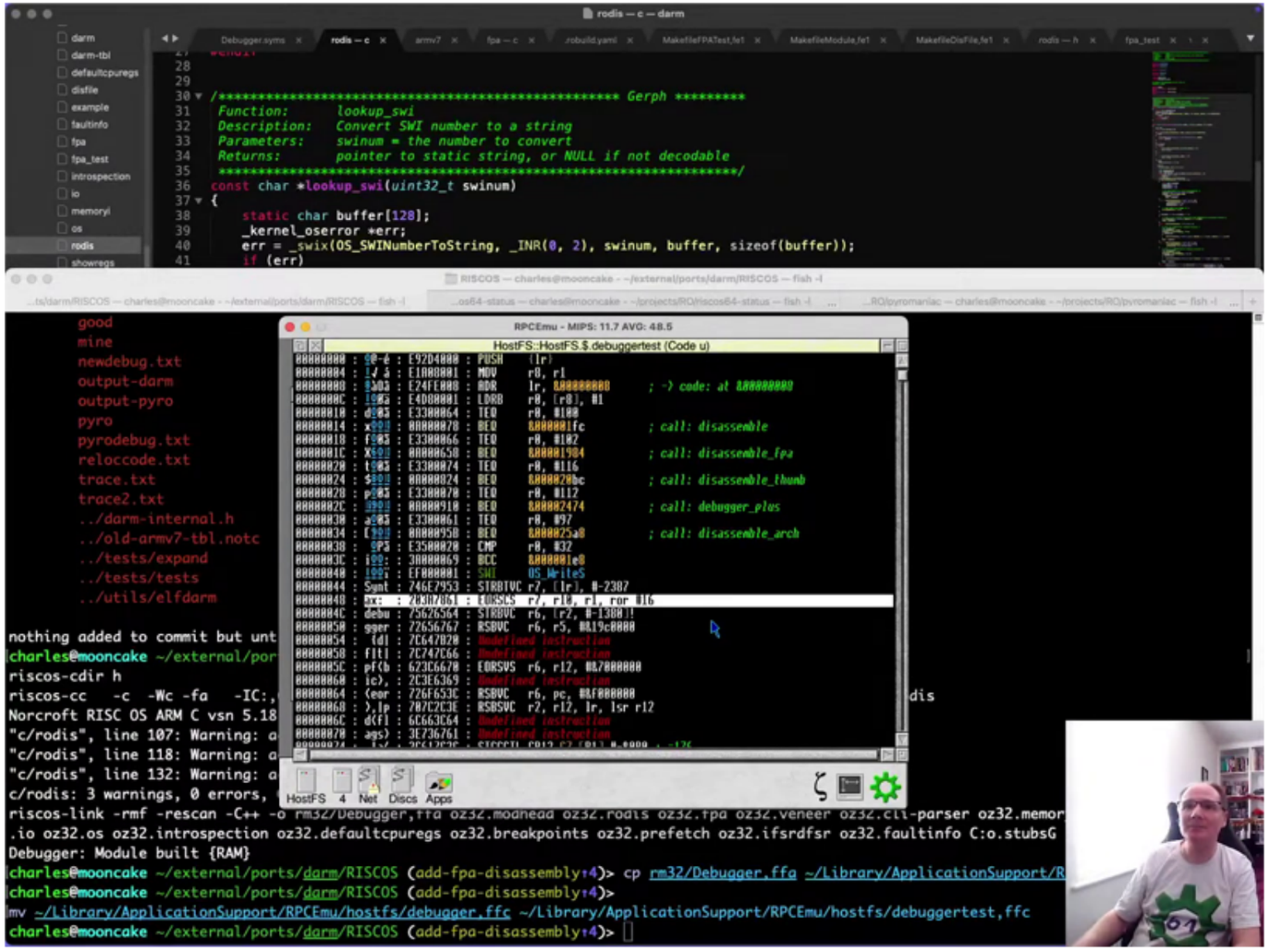
Towards 64-bit RISC OS

Live coding (1)

- Coding on YouTube live streams.
- Partly to improve my skills.
- Showing people how they can do things.
- To give the community more resources.
- Suggested ages ago by someone in a meeting.
- And because Matt Godbolt did it!

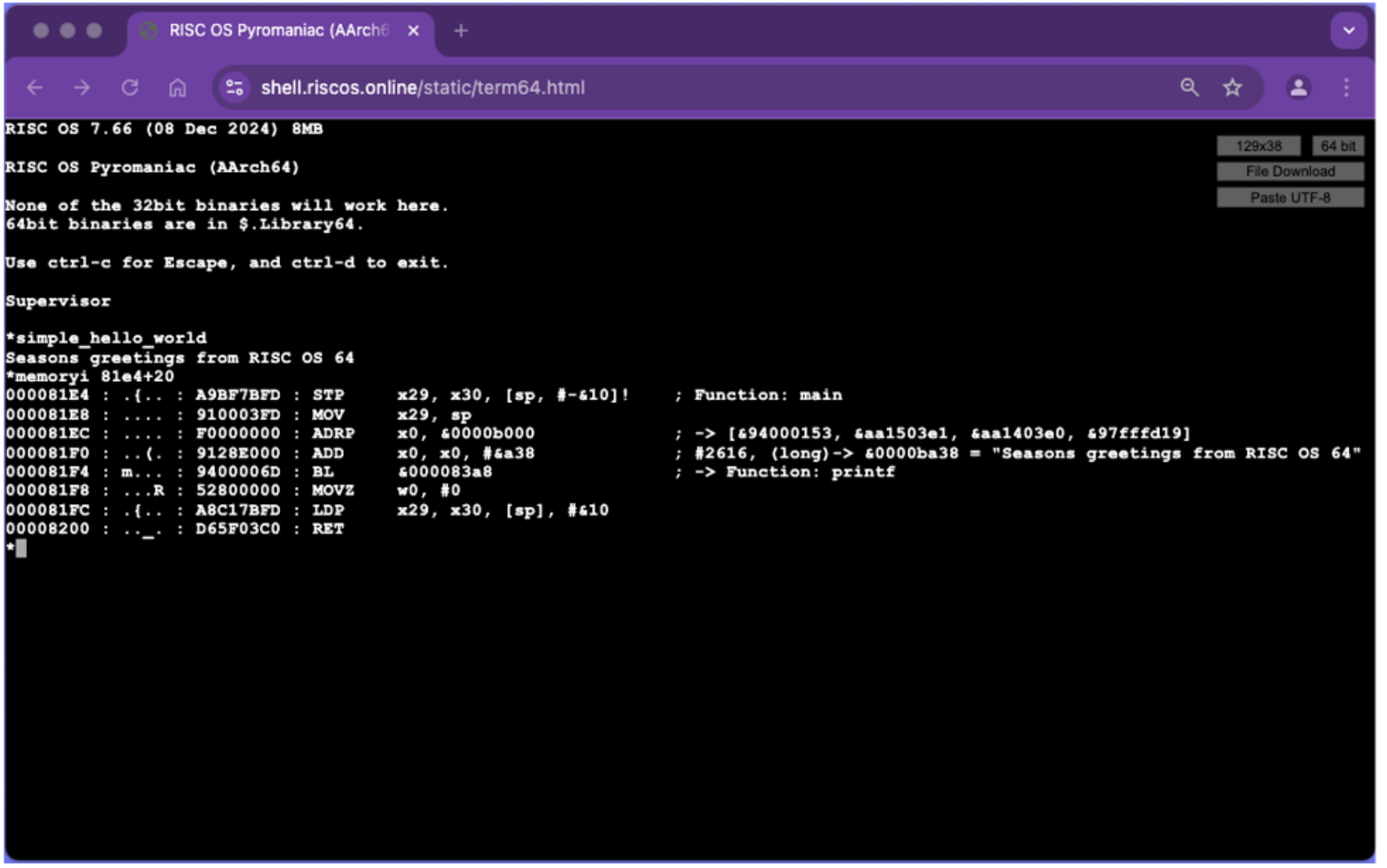
Towards 64-bit RISC OS

Live coding (2)



Towards 64-bit RISC OS

Christmas calendars 2024



Towards 64-bit RISC OS

Moonshots

> *"RISC OS is at risk of being left behind unless we act decisively."*

-- Thus spake ROOL, as they asked for £2.5 million.

Towards 64-bit RISC OS

Moonshots

> *"RISC OS is at risk of being left behind unless we act decisively."*

-- Thus spake ROOL, as they asked for £2.5 million.

- The Moonshots are intended to be larger blocks of work building towards a version of RISC OS that can be moved to 64-bit.
- They were looking for large donators to help with this.

I had done a lot of the basis for developing a 64-bit system, so I made a claim on their bounty.

Towards 64-bit RISC OS

My bounty claim (1)

Why ?

- Partly to raise awareness that I've been doing work towards this for years.
- Partly that if there's some money available, it'd be nice.
- Partly to highlight the amount of benefit it would be to work with others.

And of course, there was the fact that this would be done openly so that the community would see the co-operation.

Towards 64-bit RISC OS

My bounty claim (2)

What was it?

- Software and tools:
  - A 64-bit OS for testing and development.
  - Automation for modern CI development.
  - Tooling to build for 64-bit.
  - Documentation tailored towards documenting variants of the OS.
  - Test suites that exercise the current OS.
- Rationale:
  - Details of how this fits into the development process.
  - Evidence of what's been claimed.
- A claimed value.

3. Changing my approach

Changing my approach

What's going wrong? (1)

- Things hadn't worked out with ROOL (still waiting for them to provide guidance, or... anything really).
- The community hadn't moved to start towards 64-bit.
- My technical things aren't making any impact...

So let's work out how to change the approach...

Changing my approach

What's going wrong? (2)

I can't change ROOL, but why is the community not on board?

- Apathy.
- Lack of direction.
- Waiting for something to happen. Or just for ROOL.
- Scepticism.
- Resignation to their fate.
- No need to change.
- Just wanting to use the system.
- Longing for things from 30 years ago.
- Lack of time and inclination.

Changing my approach

What's going wrong? (3)

What have I been doing?

- Communicating.
- Open sourcing software.
- Demonstrating.
- Presenting and live coding.
- Providing encouragement.

Changing my approach

Make a plan. Execute it. (1)

- There's no written plan for RISC OS Pyromaniac.
- There's only me working on it.
- Having a plan for RISC OS 64 work would hit some of the issues I just mentioned.

Of course, just having a plan doesn't magically make people get on board... but it contributes to the viability of the project.

Changing my approach

Make a plan. Execute it. (2)

A plan will...

- Define what needs to be done.
- Allow problems to be ironed out early.
- Show that things have been thought about.
- Identify deficiencies.
- Provide encouragement to help out.
- Offers guidance for what's needed.
- Give confidence that progress can be made.

... and avoids the current situation where people just hope that something will magically appear.

Changing my approach

Work out what we mean

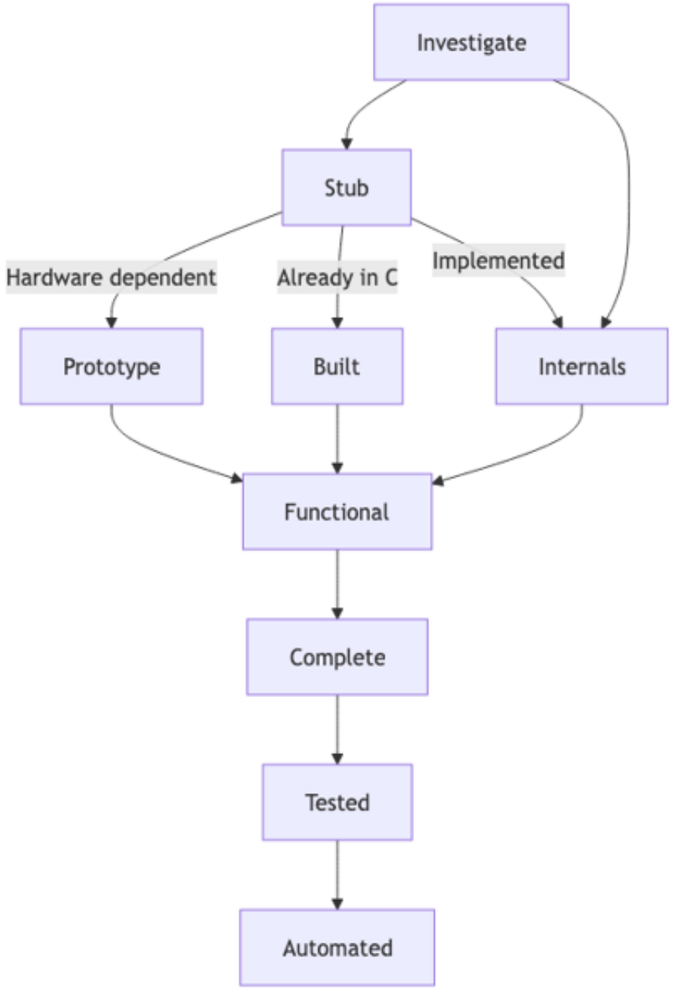
There are different types of components...

- Original assembler components - which need a complete re-write.
- Libraries and prototypes - a prototype that shows how to build a filesystem is more valuable to developers.
- Original C components - which will need updating to be aware of 64-bit.

Each of these types of components will have a different development path, but we can still track them.

Changing my approach

Software Development Lifecycle



RISC OS 64 status tour

Changing my approach

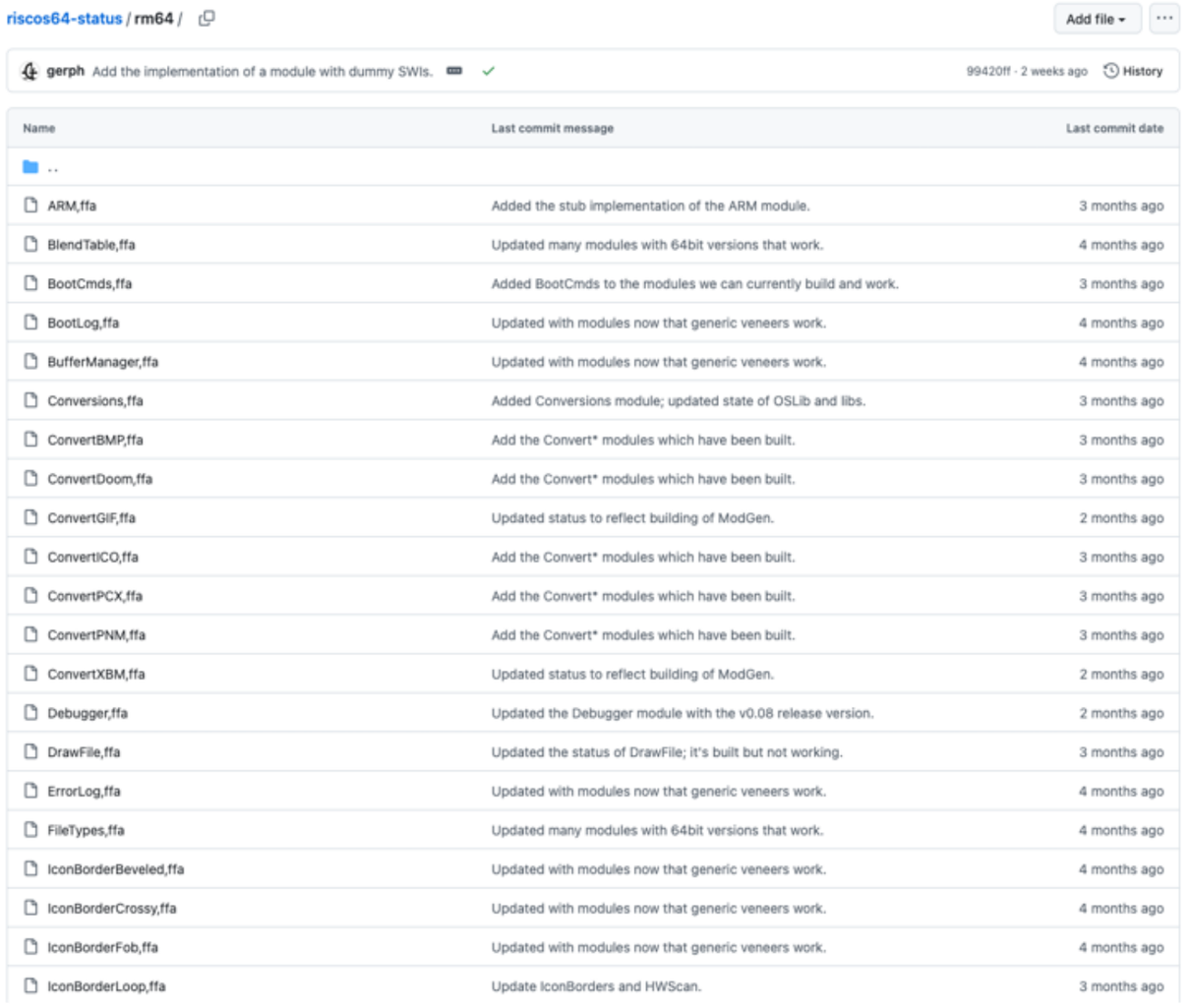
Communicating the status

- The repository is updated regularly.
- Components and phases have more documentation added as I find I need it.
- Status is visible to all.
- Anyone wanting to comment can open issues, or ask questions.

If it's not sufficient, more detail can always be added.

Changing my approach

Testing new components (1)



riscos64-status / rm64 /

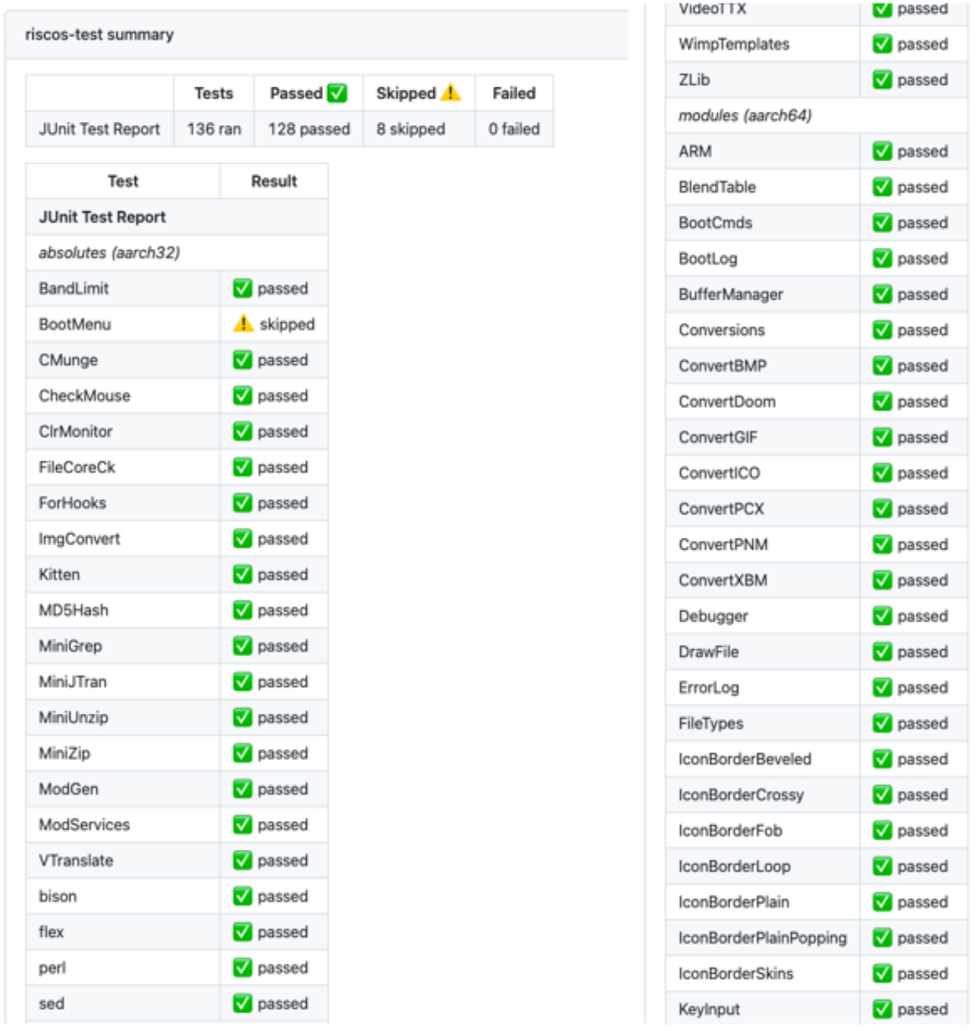
Add file

gerph Add the implementation of a module with dummy SWIs. 99420ff · 2 weeks ago History

Name	Last commit message	Last commit date
..		
ARM,ffa	Added the stub implementation of the ARM module.	3 months ago
BlendTable,ffa	Updated many modules with 64bit versions that work.	4 months ago
BootCmds,ffa	Added BootCmds to the modules we can currently build and work.	3 months ago
BootLog,ffa	Updated with modules now that generic veneers work.	4 months ago
BufferManager,ffa	Updated with modules now that generic veneers work.	4 months ago
Conversions,ffa	Added Conversions module; updated state of OSLib and libs.	3 months ago
ConvertBMP,ffa	Add the Convert* modules which have been built.	3 months ago
ConvertDoom,ffa	Add the Convert* modules which have been built.	3 months ago
ConvertGIF,ffa	Updated status to reflect building of ModGen.	2 months ago
ConvertICO,ffa	Add the Convert* modules which have been built.	3 months ago
ConvertPCX,ffa	Add the Convert* modules which have been built.	3 months ago
ConvertPNM,ffa	Add the Convert* modules which have been built.	3 months ago
ConvertXBM,ffa	Updated status to reflect building of ModGen.	2 months ago
Debugger,ffa	Updated the Debugger module with the v0.08 release version.	2 months ago
DrawFile,ffa	Updated the status of DrawFile; it's built but not working.	3 months ago
ErrorLog,ffa	Updated with modules now that generic veneers work.	4 months ago
FileTypes,ffa	Updated many modules with 64bit versions that work.	4 months ago
IconBorderBeveled,ffa	Updated with modules now that generic veneers work.	4 months ago
IconBorderCrossy,ffa	Updated with modules now that generic veneers work.	4 months ago
IconBorderFob,ffa	Updated with modules now that generic veneers work.	4 months ago
IconBorderLoop,ffa	Update IconBorders and HWScan.	3 months ago

Changing my approach

Testing new components (2)



riscos-test summary

	Tests	Passed ✅	Skipped ⚠️	Failed
JUnit Test Report	136 ran	128 passed	8 skipped	0 failed

Test	Result
JUnit Test Report	
absolutes (aarch32)	
BandLimit	✅ passed
BootMenu	⚠️ skipped
CMunge	✅ passed
CheckMouse	✅ passed
ClrMonitor	✅ passed
FileCoreCk	✅ passed
ForHooks	✅ passed
ImgConvert	✅ passed
Kitten	✅ passed
MD5Hash	✅ passed
MiniGrep	✅ passed
MiniJTran	✅ passed
MiniUnzip	✅ passed
MiniZip	✅ passed
ModGen	✅ passed
ModServices	✅ passed
VTranslate	✅ passed
bison	✅ passed
flex	✅ passed
perl	✅ passed
sed	✅ passed
VideoTTX	✅ passed
WimpTemplates	✅ passed
ZLib	✅ passed
modules (aarch64)	
ARM	✅ passed
BlendTable	✅ passed
BootCmds	✅ passed
BootLog	✅ passed
BufferManager	✅ passed
Conversions	✅ passed
ConvertBMP	✅ passed
ConvertDoom	✅ passed
ConvertGIF	✅ passed
ConvertICO	✅ passed
ConvertPCX	✅ passed
ConvertPNM	✅ passed
ConvertXBM	✅ passed
Debugger	✅ passed
DrawFile	✅ passed
ErrorLog	✅ passed
FileTypes	✅ passed
IconBorderBeveled	✅ passed
IconBorderCrossy	✅ passed
IconBorderFob	✅ passed
IconBorderLoop	✅ passed
IconBorderPlain	✅ passed
IconBorderPlainPopping	✅ passed
IconBorderSkins	✅ passed
KeyInput	✅ passed

Changing my approach

Testing without an OS?

- We have an OS - we have two.
- We have RISC OS Classic - for legacy.
- We have RISC OS Pyromaniac - for 32-bit and 64-bit.
- We have the RISC OS Build service to test things.

This gives us a means to test manually and automatically relatively easily.

Changing my approach

Developing for a new OS?

Developing for a new OS needs tools...

- Norcroft C compiler only targets ARM.
- Better compilers exist which have mass usage and development.
- GCC can be made to work for RISC OS.
- If we remove the RISC OS-specific needs, it's even easier.

The Docker image that I've created allows building of RISC OS components in both 32-bit and 64-bit very simply.

Build environment demonstration

Changing my approach

Native languages

Cross-compiling is good, but we also want languages running in RISC OS...

- **SDL BASIC** - rudimentary port.
- **MyBASIC** - lightweight interpreter port.
- **SED** - it's a language, honest.
- **PocketPython** - Python for small systems.
- **Perl** - yeah, I like Perl.
- **HUProlog** - might as well have a functional language.
- **Lua** - rudimentary port, but surprisingly easy.
- **PicoC** - C interpreter. I've even added `_swix`.
- **Berry** - a small scripting language.
- **TCC** - C compiler, targetting AArch64.

RISC OS 64 language demonstration

4. Conclusion

Conclusion

I've talked about many how the RISC OS Pyromaniac project has evolved.
I've talked about what I'm trying to do to address the issues I see.

- Built up a plan.
- Communicated what's being done.
- Explaining the details.
- Videos to show development and build a community.
- Tools and testing environments for people to use.

And I'm doing this because I hope to encourage others.

Conclusion

Are *you* interested in being involved?

Questions

I'll take any questions that people have.
Slides and Info: **https://presentation.riscos.online/pyromaniac4/**