

The Yarg Language: Over-engineering a Model Railway Controller

OggCamp 2026

John McAleely, April 26th



transmissionbegins.com

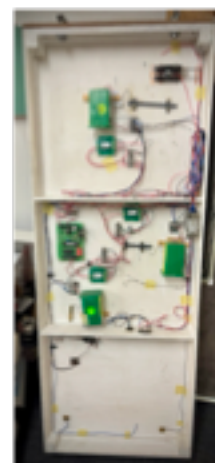


The Problem

- Club exhibition project
- Move trains from track to track
- On a fairly small board
- Other club members will do future work
- Push by hand is an option
- We already have CAN Bus controls for other items...



Above

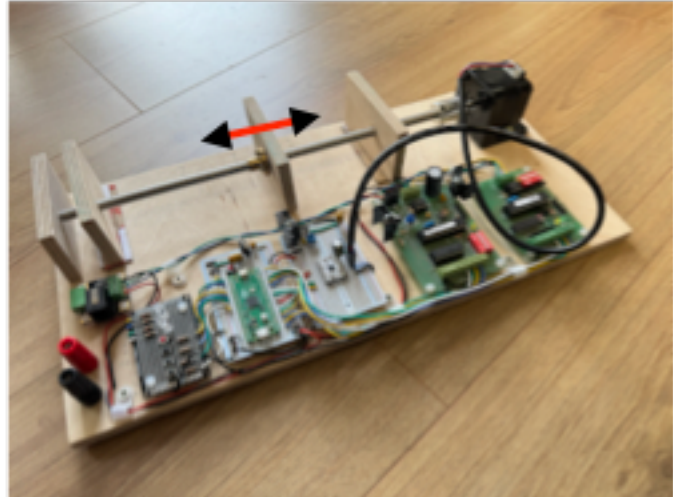


Underneath

Version 1 - the 'simple' solution

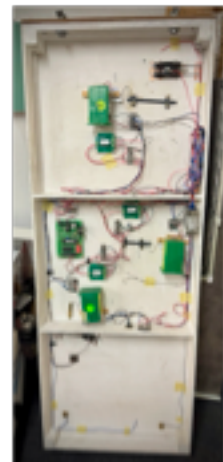
Kit assembly

- Stepper Motor
- 2x CAN BUS PIC modules
- 1x Stepper Driver
- 1x Pico
- Micropython Firmware
- 1x Test I/O board



Issues to address

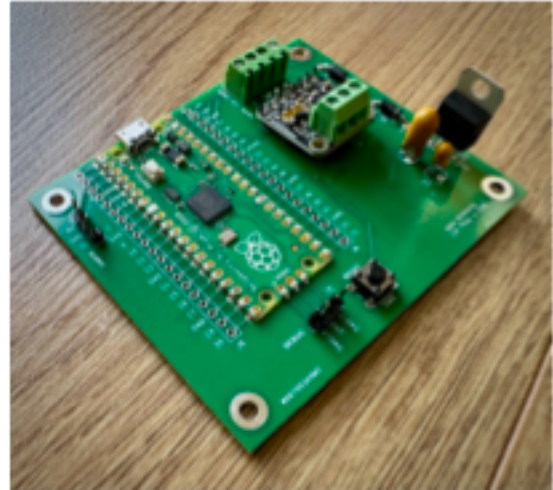
- Size
- Software complexity
- Maintenance and adjustments without me



Version 2 - the Hardware Solution

Space saving with bespoke hardware

- Pico
- CAN transceiver
- Stepper Driver (not shown)
- Pico software required:
 - PIO, Multiple Tasks
 - Use Micropython, or C or...?



Version 2 - the Software Solution: Yarg Language

- A language specifically for Microcontrollers
- Intended to be general purpose, not limited to Model Railways!
- Named after Cornish Yarg Cheese



Yarg Features

- Dynamic
- Interpreted REPL
- Compiles to bytecode VM before execution
- Direct Hardware access:
 - Registers
 - Interrupts
- Multiple core support

Inspirations

- Lua
 - Bytecode VM performance, co-operative multitasking
- Go
 - Types, Pre-emptive multitasking
- MicroPython
 - REPL, ease of use

System Features

- `yarg`
 - Runs on development PC
 - Builds UF2 files, runs test suites
 - In future will optimise using resources of host PC
- `cyarg`
 - C implementation on Pico
 - REPL, interpreter, compiler, VM
 - Bytecode VM designed with real-time in mind
- Public, open-source (MIT) project: github.com/yarg-lang
 - 0.3.0 demo release, main has breaking changes as I speak

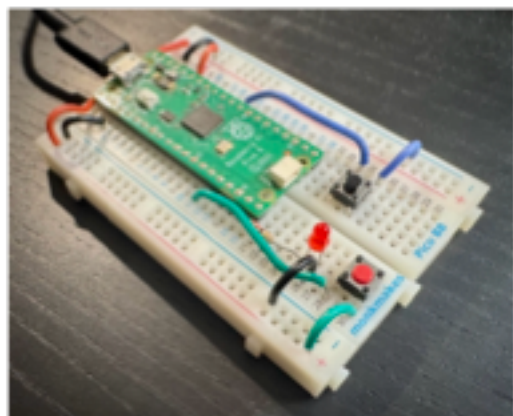
Demo: Hello Button

- Interrupt handlers are functions

```
fun gpio_response() {  
  const events = gpio_events();  
  gpio_acknowledge_irq(gpio, events);  
  share(chan, events);  
}
```

- Registers are structures

```
place struct {  
  struct {  
    uint32 status;  
    uint32 ctrl;  
  }[30] gpio;  
} @x40014000 io_bank0;
```



The Future

- Finish the Stepper Module
- Characters and Strings
- RP2xxx PIO
- Pico 2, Pico W, Pico 2W
 - Including the RISC-V cores
- ESP32



Yarg Lang
yarg-lang.dev



- Newsletter - sign up here: yarg-lang.dev/notes
- Project and sources on GitHub (MIT licence): github.com/yarg-lang
- Local Model Railway Group: southlondonag.org.uk
 - National Organisation: www.scalefour.org
- Questions, ideas and suggestions for what's next are welcome!