CEC Continuing Education Center

DesignNews

Writing Microcontroller Drivers in Rust

DAY 3 : PAC Deep Dive

Sponsored by

DigiKey

BENINGO EMBEDDED GROUP

informa markets

Webinar Logistics

- Turn on your system sound to hear the streaming presentation.
- If you have technical problems, click “Help” or submit a question asking for assistance.
- Participate in ‘Group Chat’ by maximizing the chat widget in your dock.

THE SPEAKER

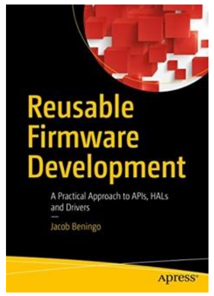
Jacob Beningo

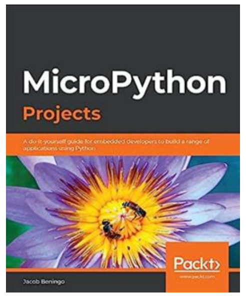
Jacob@beningo.com

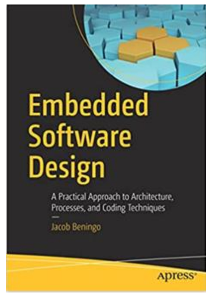
Beningo Embedded Group – CEO / Founder

Focus: Embedded Software Consulting and Training

Help teams deliver higher-quality embedded software faster. We specialize in creating and promoting embedded software excellence in businesses around the world.


Reusable Firmware Development
A Practical Approach to APIs, HALs and Drivers
Jacob Beningo
Apress




Embedded Software Design
A Practical Approach to Architecture, Processes, and Coding Techniques
Jacob Beningo
Apress

Blogs for:

- DesignNews.com
- Embedded.com
- EmbeddedRelated.com
- MLRelated.com

Visit **www.beningo.com** to learn more

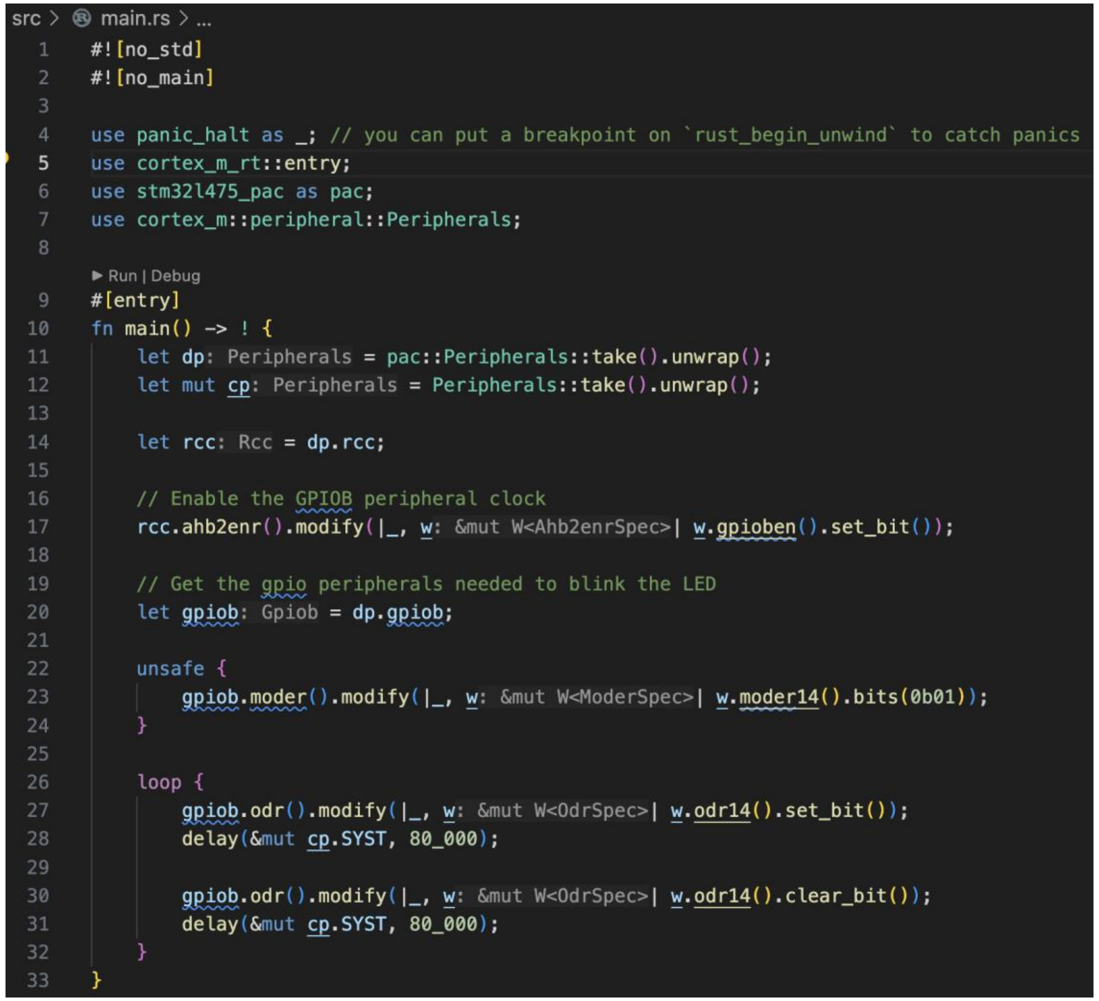
01

Embedded Rust Blinky Application Example

Embedded Rust Blinky Application Example

Hello Blinky!

- Use the peripheral PAC
- Initialize the clock
- Initialize the GPIO connected to an LED
- Loop and toggle the LED



src > main.rs > ...

```rust
#![no_std]
#![no_main]

use panic_halt as _; // you can put a breakpoint on `rust_begin_unwind` to catch panics
use cortex_m_rt::entry;
use stm32l475_pac as pac;
use cortex_m::peripheral::Peripherals;

▶ Run | Debug
#[entry]
fn main() -> ! {
    let dp: Peripherals = pac::Peripherals::take().unwrap();
    let mut cp: Peripherals = Peripherals::take().unwrap();

    let rcc: Rcc = dp.rcc;

    // Enable the GPIOB peripheral clock
    rcc.ahb2enr().modify(|_, w: &mut W<Ahb2enrSpec>| w.gpioben().set_bit());

    // Get the gpio peripherals needed to blink the LED
    let gpiob: Gpiob = dp.gpiob;

    unsafe {
        gpiob.moder().modify(|_, w: &mut W<ModerSpec>| w.moder14().bits(0b01));
    }

    loop {
        gpiob.odr().modify(|_, w: &mut W<OdrSpec>| w.odr14().set_bit());
        delay(&mut cp.SYST, 80_000);

        gpiob.odr().modify(|_, w: &mut W<OdrSpec>| w.odr14().clear_bit());
        delay(&mut cp.SYST, 80_000);
    }
}
```

Blinky TOML

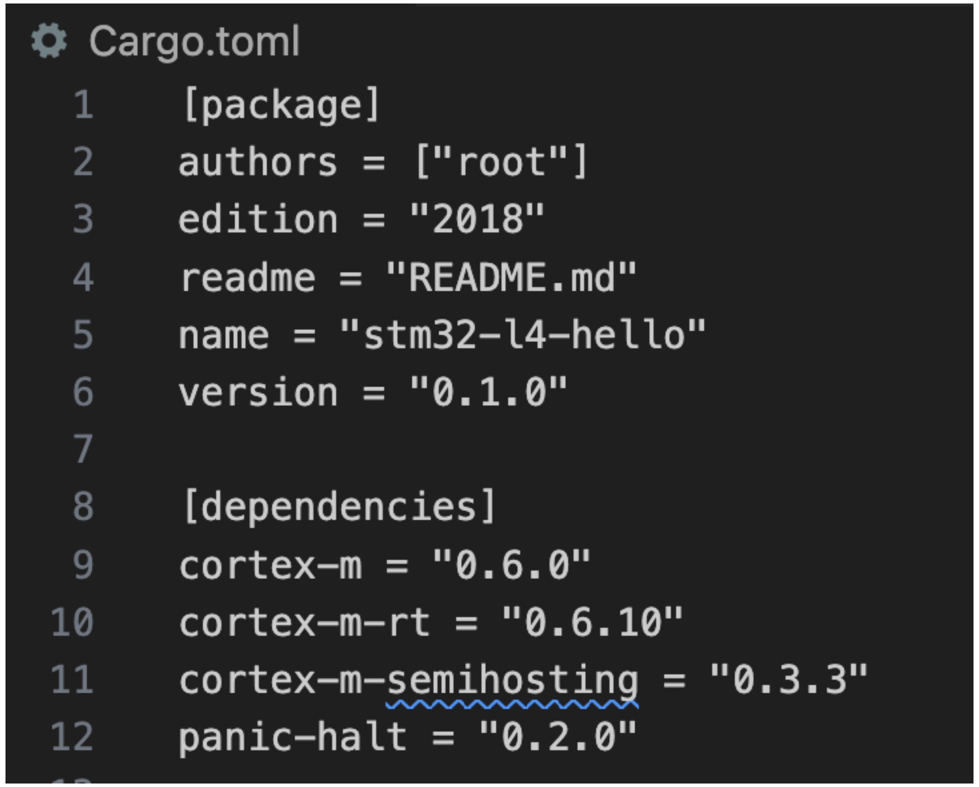
02

Continuing Education Center

BENINGO EMBEDDED GROUP

DigiKey

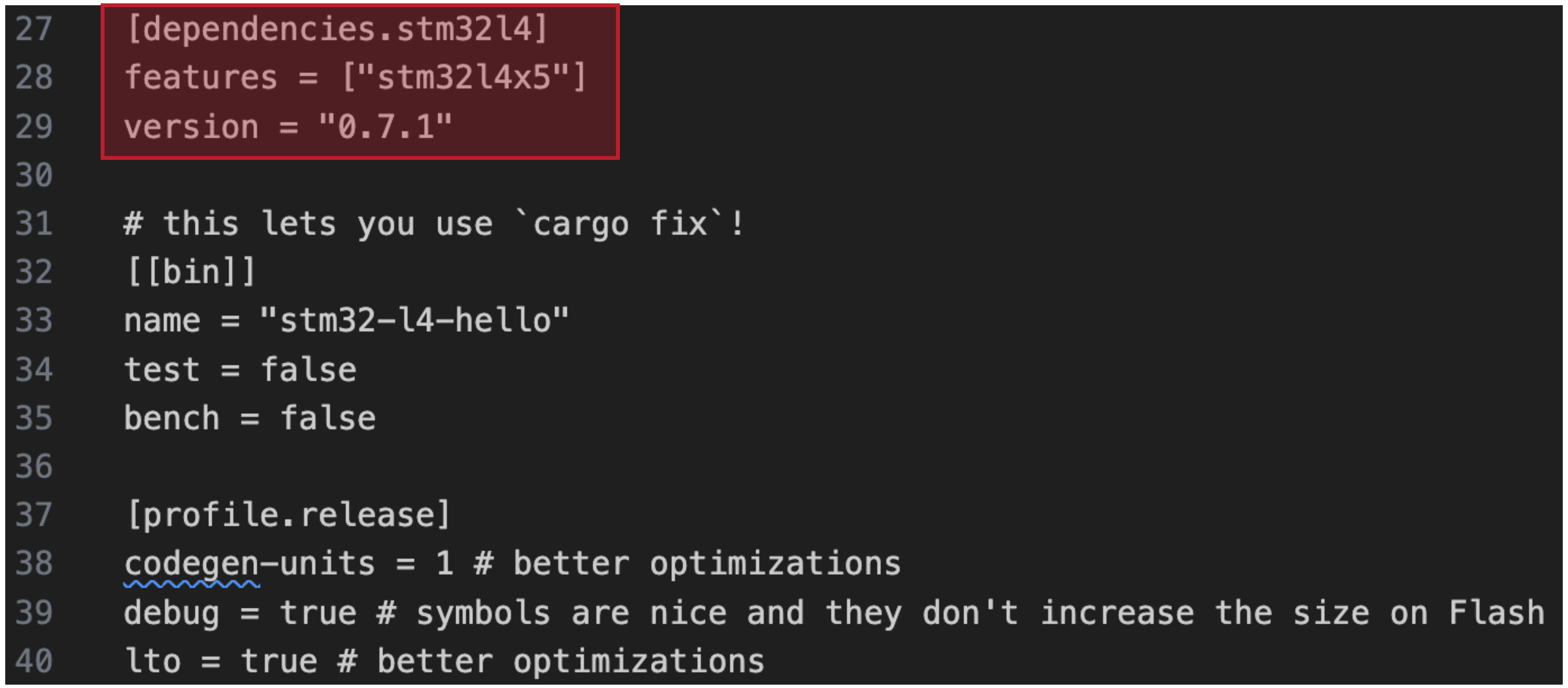
Blinky TOML


Cargo.toml

```toml
[package]
authors = ["root"]
edition = "2018"
readme = "README.md"
name = "stm32-l4-hello"
version = "0.1.0"

[dependencies]
cortex-m = "0.6.0"
cortex-m-rt = "0.6.10"
cortex-m-semihosting = "0.3.3"
panic-halt = "0.2.0"
```

Blinky TOML – Using the Community PAC

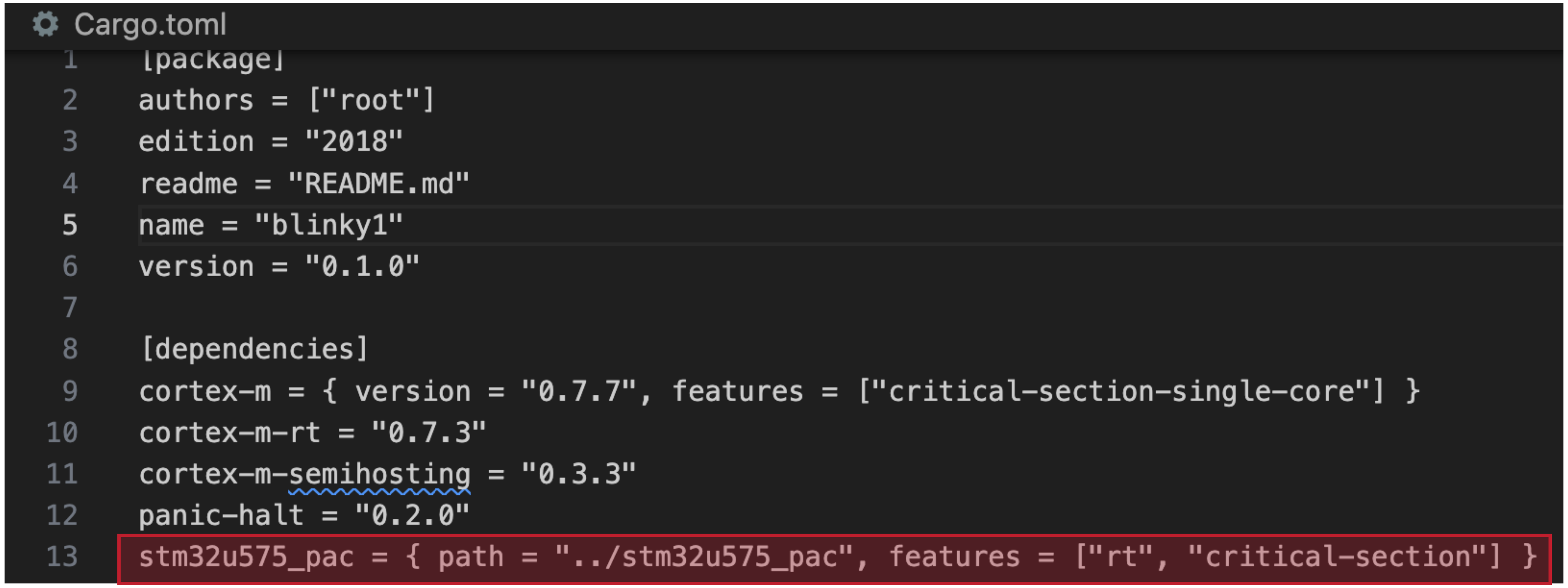


```toml
[dependencies.stm32l4]
features = ["stm32l4x5"]
version = "0.7.1"

# this lets you use `cargo fix`!
[[bin]]
name = "stm32-l4-hello"
test = false
bench = false

[profile.release]
codegen-units = 1 # better optimizations
debug = true # symbols are nice and they don't increase the size on Flash
lto = true # better optimizations
```

Blinky TOML – Using our PAC



```toml
[package]
authors = ["root"]
edition = "2018"
readme = "README.md"
name = "blinky1"
version = "0.1.0"

[dependencies]
cortex-m = { version = "0.7.7", features = ["critical-section-single-core"] }
cortex-m-rt = "0.7.3"
cortex-m-semihosting = "0.3.3"
panic-halt = "0.2.0"
stm32u575_pac = { path = "../stm32u575_pac", features = ["rt", "critical-section"] }
```

Audience POLL Question

How do you use your custom pack?

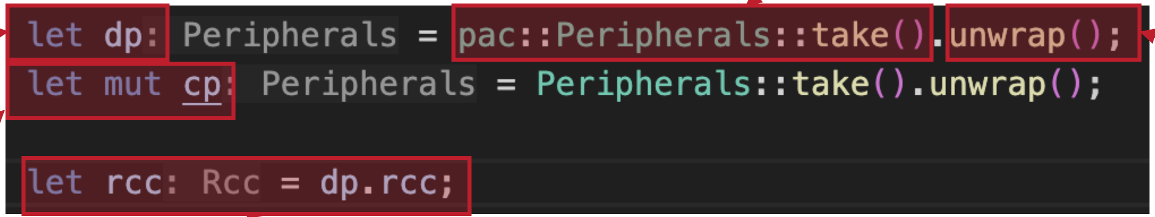
a) Modify the dependencies in Cargo.toml
b) In your code module, use the “use” keyword and your PAC
c) All the above
d) None of the above

03

Embedded Rust Blinky Application Example

Embedded Rust Blinky Application Example

Accessing Peripheral Structures and Fields



```
let dp: Peripherals = pac::Peripherals::take().unwrap();
let mut cp: Peripherals = Peripherals::take().unwrap();

let rcc: Rcc = dp.rcc;
```

Acquire Peripheral Access to Device
(Can only be taken once!)

Holds reference to peripherals available in the PAC.
(Values in dp change, but not dp!

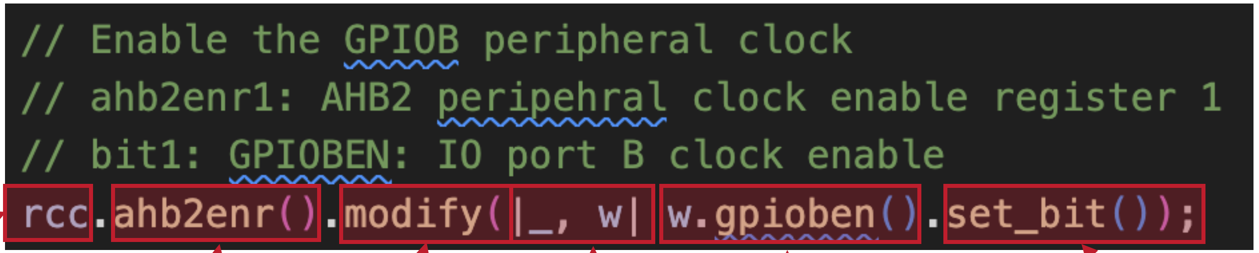
unwrap extracts the value from 'Option' return by take()
- Peripheral Object
- None

Core peripheral reference.
mut so it can be modified later
in the code by passing as reference

Ownership of the rcc field in dp is transferred to rcc.

Embedded Rust Blinky Application Example

Enabling GPIOB Clock



```
// Enable the GPIOB peripheral clock
// ahb2enr1: AHB2 peripehral clock enable register 1
// bit1: GPIOBEN: IO port B clock enable
rcc.ahb2enr().modify(|_, w| w.gpioben().set_bit());
```

Instance of Rcc peripheral

Accesses Enable Register

Access bit in ahb2enr that is gpioben

Set the gpioben bit to 1

Takes a closure (an anonymous function) that specifies how the bits in the register should be changed.

2 Parameter closure:
- _ is the current value of the register
- w is a writable proxy to set or clear bits

CEC Continuing Education Center

BENINGO EMBEDDED GROUP

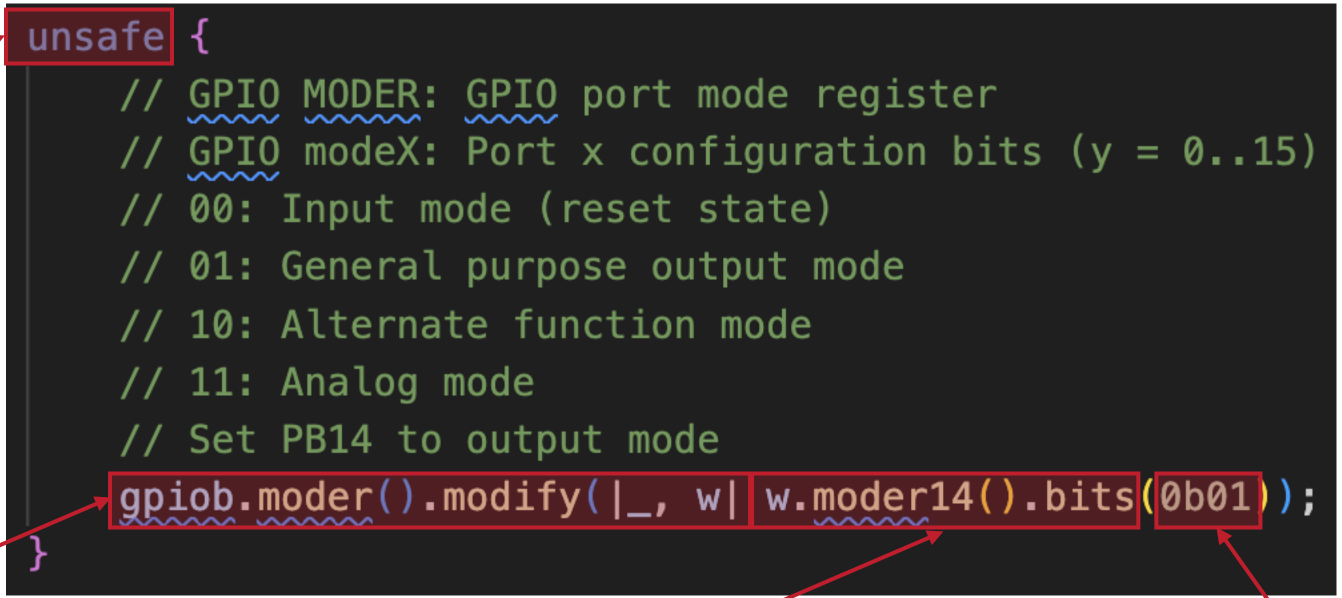
DigiKey

Embedded Rust Blinky Application Example

Setting GPIOB to Output

MODER is a hardware register! Direct manipulation of memory-mapped hardware is inherently unsafe because:

- No bounds checking
- No data race protection
- Unpredictable side effects



```
unsafe {
    // GPIO MODER: GPIO port mode register
    // GPIO modeX: Port x configuration bits (y = 0..15)
    // 00: Input mode (reset state)
    // 01: General purpose output mode
    // 10: Alternate function mode
    // 11: Analog mode
    // Set PB14 to output mode
    gpiob.moder().modify(|_, w| w.moder14().bits(0b01));
}
```

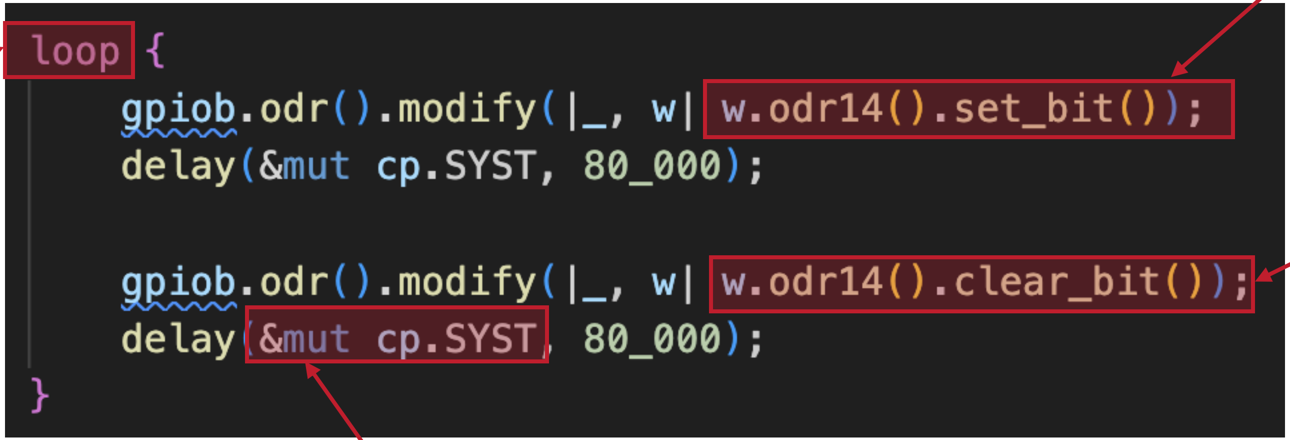
Access MODER with a modify operation

Perform bit operation on mode 14 register bits

Bit pattern to write

Embedded Rust Blinky Application Example

The “main” loop



```
loop {
    gpiob.odr().modify(|_, w| w.odr14().set_bit());
    delay(&mut cp.SYST, 80_000);

    gpiob.odr().modify(|_, w| w.odr14().clear_bit());
    delay(&mut cp.SYST, 80_000);
}
```

Set output data register for pin 14

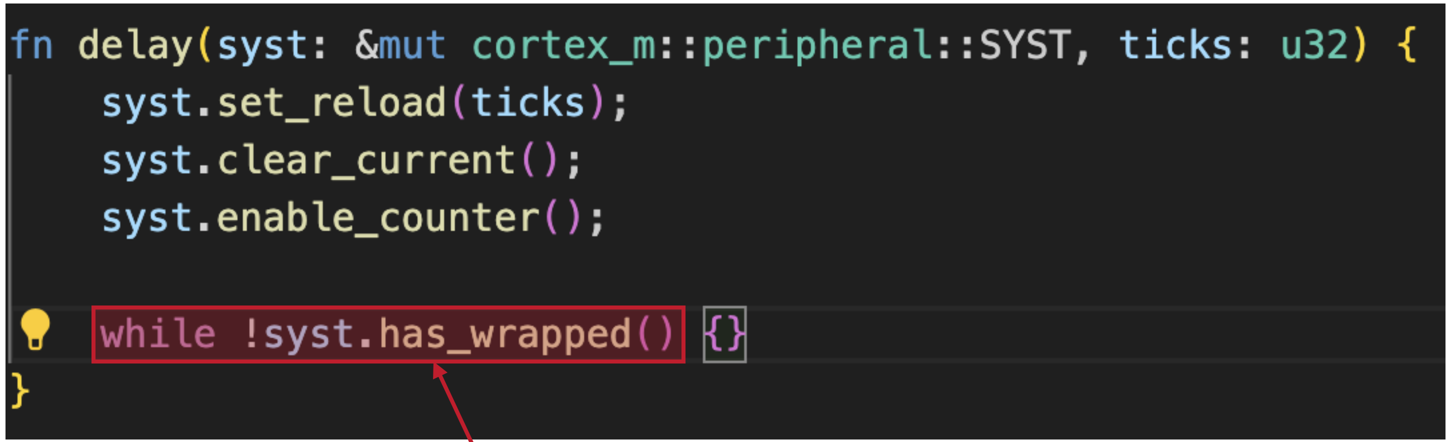
Clear output data register for pin 14

Most Idiomatic infinite loop
- loop { }
- for _ in 0.. { }
- while true { }

Pass reference to a modifiable cp.SYST

Embedded Rust Blinky Application Example

The Delay Function



```rust
fn delay(syst: &mut cortex_m::peripheral::SYST, ticks: u32) {
    syst.set_reload(ticks);
    syst.clear_current();
    syst.enable_counter();

    while !syst.has_wrapped() {}
}
```

Loop until timer expires!

Audience POLL Question

What does _ mean in a closure?

a) Ignore the parameter
b) Clear the register
c) Use the current value of the register
d) None of the above

Next Steps

04

Embedded Rust Docker Container

- https://mailchi.mp/beningo/embedded_rust_docker_container
  - Rust Toolchain
  - Embedded Tools

Beningo Rust Docker Container

CEC Continuing Education Center

BENINGO EMBEDDED GROUP

DigiKey

Additional Resources

Please consider the resources below:

- Jacob's Blogs
- Jacob's CEC courses
- Embedded Software Academy

- Embedded Bytes Newsletter
  - http://bit.ly/1BAHYXm

www.beningo.com

Consulting Coaching Training

CEC Continuing Education Center

DesignNews

Thank You

Sponsored by

DigiKey

BENINGO EMBEDDED GROUP

informa markets