AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE
AGH UNIVERSITY OF SCIENCE
AND TECHNOLOGY

Rust jako nowoczesny język programowania systemów wbudowanych

Dyplomant: Gabriel Górski
Promotor: dr inż. Krzysztof Świentek

Wydział Fizyki i Informatyki Stosowanej

Kraków, 17 listopada 2020 roku

Plan prezentacji

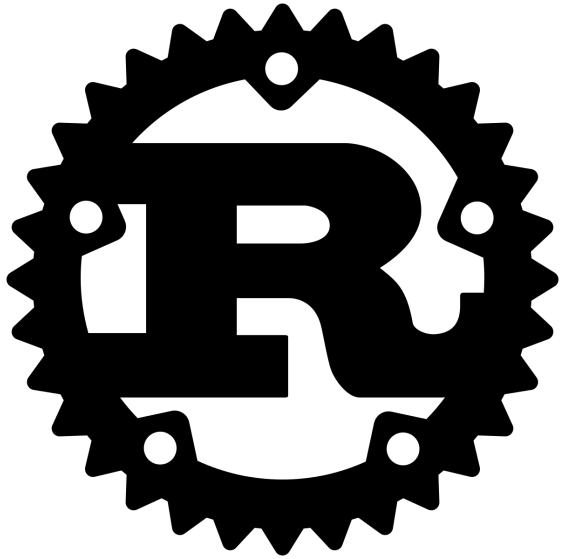
1. Cel i zakres pracy
2. Rust - język programowania & ekosystem
3. Systemy wbudowane
4. Rust a systemy wbudowane
5. Przykładowy projekt
6. Podsumowanie

Cel i zakres pracy

- Zbadanie **użyteczności** języka Rust z naciskiem na dziedzinę **systemów wbudowanych**.
- Analiza **usprawnień** wnoszonych przez język w **odniesieniu** do jego **konkurentów** (C/C++).
- **Praktyczna weryfikacja** ekosystemu języka Rust poprzez stworzenie przykładowego projektu.
- Pokazanie Czytelnikowi pracy podstaw języka i **zmotywowanie Go do dalszej eksploracji tematu**.

Rust - język programowania

- wieloparadygmatowy język programowania
  - nacisk na podejście funkcyjne
- kompilacja AoT **do kodu maszynowego platformy docelowej**; wsparcie dla licznych urządzeń
- unikalne podejście do zarządzania pamięcią
  - **mechanika własności i pożyczania** (brak GC)
    - **gwarancja spójności pamięci; niewyrażalność zachowań niezdefiniowanych**
  - Podczas pisania programu udowadniamy jego poprawność; **kompilacja jest dowodem poprawności**.
- FFI (ang. *Foreign Function Interface*) - interop Rust <-> C, system makr z systemem typów opartym na AST (ang. *Abstract Syntax Tree*), *Unsafe Rust*, iteratory, dopasowanie do wzorca, algebraiczne typy danych, programowanie uogólnione, cechy i więcej.



Rust - język programowania

```rust
fn main() {
    let mut vector = vec![1, 2, 3, 4, 5];

    let const_ref = &vector[0];

    vector.push(6);

    // !!! Co się stało z `const_ref` w przypadku realokacji `vector`?
    borrow_const_ref(const_ref); // Błąd kompilacji
}

fn borrow_const_ref(a: &i32) {}
```

```
error[E0502]: cannot borrow `vector` as mutable because it is also borrowed as immutable
 --> examples/borrowing-4.rs:6:5
  |
4 |     let const_ref = &vector[0];
  |                      ------ immutable borrow occurs here
5 |
6 |     vector.push(6);
  |     ^^^^^^^^^^^^^^ mutable borrow occurs here
...
9 |     borrow_const_ref(const_ref); // Błąd kompilacji
  |                      --------- immutable borrow later used here
```

Rust - język programowania

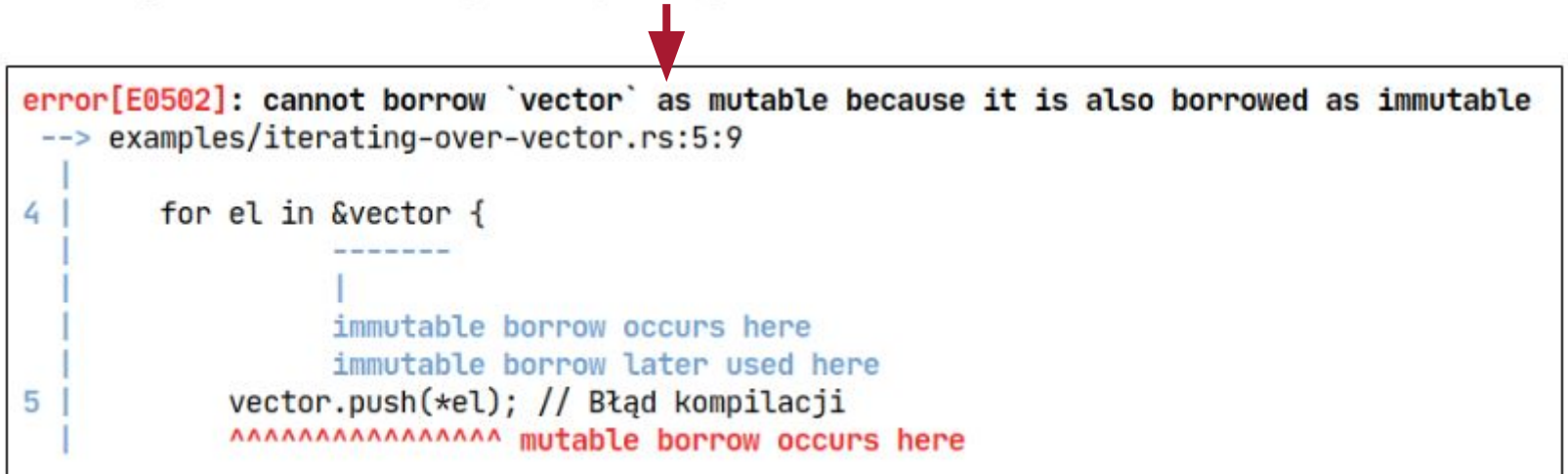
```cpp
#include <vector>
#include <iostream>

int main() {
    std::vector<int> vector = { 1, 2, 3, 4, 5 };

    for(auto& el : vector) {
        vector.push_back(1); // Zachowanie niezdefiniowane!
    }
}
```

```rust
fn main() {
    let mut vector = vec![1, 2, 3, 4, 5];

    for el in &vector {
        vector.push(1); // Błąd kompilacji
    }
}
```



```
error[E0502]: cannot borrow `vector` as mutable because it is also borrowed as immutable
 --> examples/iterating-over-vector.rs:5:9
  |
4 |     for el in &vector {
  |               -------
  |               |
  |               immutable borrow occurs here
  |               immutable borrow later used here
5 |         vector.push(*el); // Błąd kompilacji
  |         ^^^^^^^^^^^^^^^^ mutable borrow occurs here
```

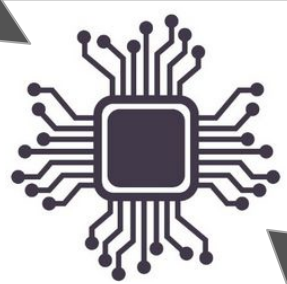
Rust - ekosystem

- *cargo* - standardowy menedżer projektu i zależności
- *rustup* - standardowy menedżer instalacji środowiska
- *rustfmt* - standardowy formater kodu źródłowego
- *rustdoc* - standardowy generator dokumentacji

Systemy wbudowane

Nie ma jednej precyzyjnej definicji systemu wbudowanego; mimo to w bardzo ogólny sposób można powiedzieć, że typowo jest to **zespół sprzętu komputerowego, oprogramowania i innych części, które jako całość mają pełnić z góry zdefiniowaną funkcję**.

[Michael Barr. Programming Embedded Systems in C and C++. O'Reilly Media, 1 edition, 1999, s. 166]

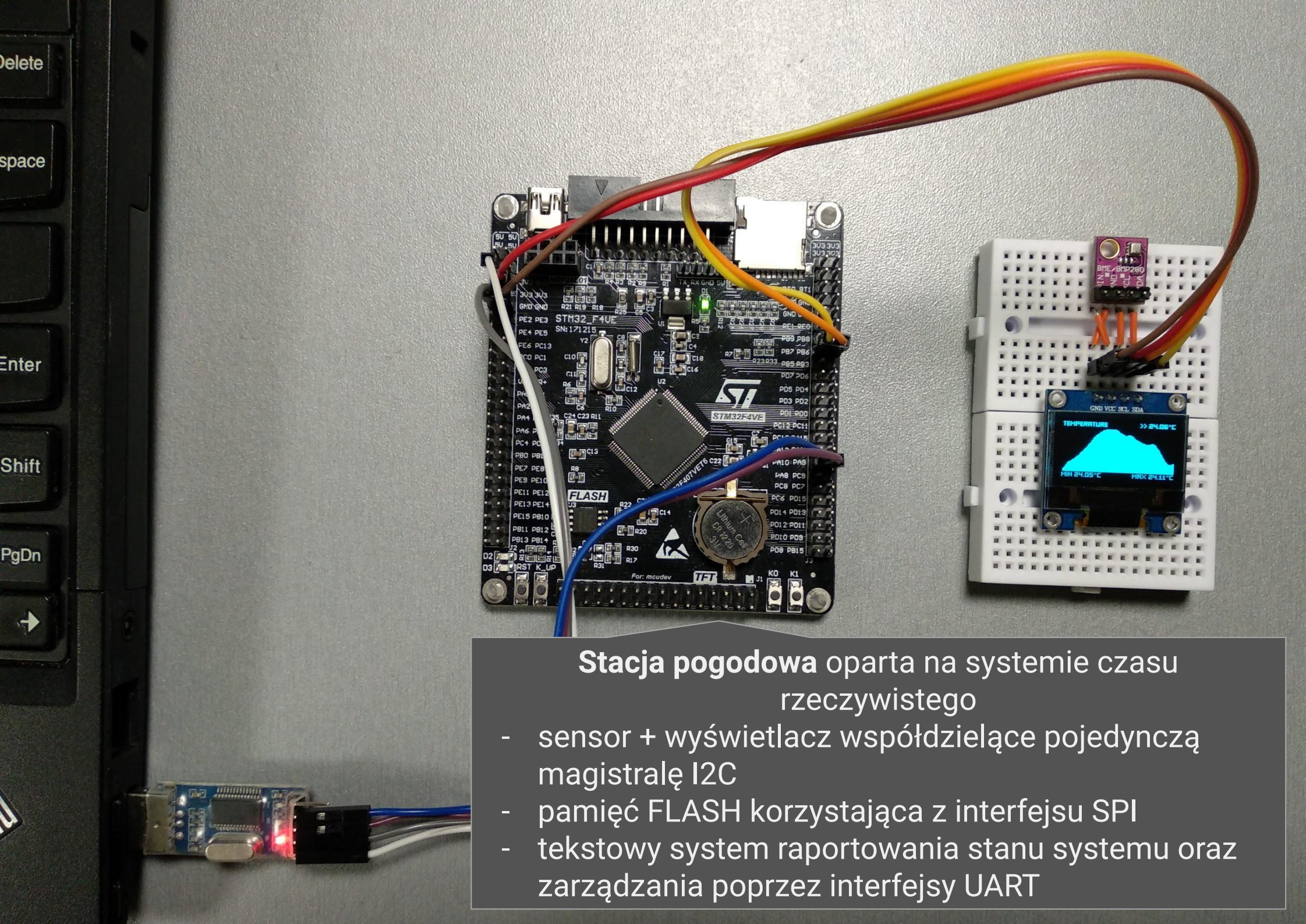


Jednym z elementów odróżniających systemy wbudowane od komputerów osobistych jest właśnie **nacisk na silne ograniczenie wielofunkcyjności i brak możliwości przeprogramowywania przez użytkowników końcowych**.

[Steve Heath. Embedded Systems Design. Newnes, 2 edition, 2003, s. 2]

Rust a systemy wbudowane

- Rust został stworzony z myślą o programowaniu systemowym
  - brak GC
  - brak maszyny wirtualnej; kompilacja do kodu maszynowego
- Żywo rozwijający się **ekosystem abstrakcji, bibliotek i narzędzi**
  - Możność tworzenia sterowników **niezależnych** od wykorzystywanego mikrokontrolera
- System typów i składnia pozwala na **uniemożliwienie wyrażenia niepoprawnej interakcji ze sprzętem**
  - Peryferium jako maszyna stanów
- Aktywna społeczność *Embedded Rust*



Stacja pogodowa oparta na systemie czasu rzeczywistego

- sensor + wyświetlacz współdzielące pojedynczą magistralę I2C
- pamięć FLASH korzystająca z interfejsu SPI
- tekstowy system raportowania stanu systemu oraz zarządzania poprzez interfejsy UART

Podsumowanie

- Rust ma ogromny potencjał **jako język programowania systemowego**; cieszy się rosnącym zainteresowaniem ze strony zarówno programistów, jak i biznesu.
- Wczesna walidacja poprawności oprogramowania znacząco **redukuje ilość błędów**; tym samym owocując **w oszczędności czasu**
- **Ekspresywność** języka pozwala na tworzenie abstrakcji/bibliotek, których **niepoprawne użycie jest niewyrażalne**
- Społeczność ciągle pracuje nad rozszerzaniem inwentarza ekosystemu i **czynienie go przystępniejszym**

Podsumowanie

Rust ma ogromny potencjał **jako język programowania systemowego**; cieszy się rosnącym zainteresowaniem ze strony zarówno programistów, jak i biznesu.

Wczesn… **redukuj… czasu** …**ci**

Ekspres… abstrak… **niewyraz…**

Nawet jeśli Rust nie podbije branży systemów wbudowanych (ani innych), warto go zbadać z powodu narzucanych przez niego **dobrych praktyk programistycznych**, które z pewnością będą pomocne w dowolnym języku systemowym.

Społeczność ciągle pracuje nad rozszerzaniem inwentarza ekosystemu i **czynienie go przystępniejszym**

Dziękuję za uwagę!