Rust on the nRF9160

Meeting Embedded
Berlin, November 2019

Jonathan Pallant
Jonathan.Pallant@42technology.com
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nordic nRF9160 SiP</td>
<td>3</td>
</tr>
<tr>
<td>Embedded Rust</td>
<td>8</td>
</tr>
<tr>
<td>Embedded HAL</td>
<td>13</td>
</tr>
<tr>
<td>Creating Safe Wrappers</td>
<td>20</td>
</tr>
<tr>
<td>Our Demo</td>
<td>26</td>
</tr>
</tbody>
</table>
Nordic nRF9160 SiP
The System-in-Package
What do you get?
What do you get (in simple terms)?

- Cortex-M33 @ 64 MHz
- 256 KiB RAM / 1 MiB Flash
- Standard Nordic Peripherals
- GNSS Receiver
- NB-IoT / LTE-M Baseband
- SIM Interface
- Mailbox / IPC
- Power Supply
- RF Frontend
Trust Zone M

- Secure Handler
- Non-Secure Handler
- Secure Thread
- Non-Secure Thread
Embedded Rust
What you need

- Board
- SoC
- CPU Core
- Instruction Set
What we had

- Tiva-C Launchpad
- STM32F4-DISCO
- nRF52-DK
- STM32F7-DISCO
- TM4C123
- STM32F4
- nRF52
- STM32F7
- Cortex-M4
- Cortex-M7

thumbv7em
What we wanted to support

- nRF9160-DK
- nRF9160
- Cortex-M33
- thumbv8m.main
What we did

- nRF9160-DK
- nRF9160
- Cortex-M33
- thumbv8m.main
- nrf9160-dk
- nrf9160-hal
- cortex-m
- rustc + LLVM Backend
Embedded HAL
In a simple world...

- Modem Driver
  - UART Interface

- SD Card Driver
  - SPI Interface

- Accelerometer Driver
  - I²C Interface

- Keypad Driver
  - GPIO Interface
But microcontrollers don’t have a lot in common…
Too many drivers!

- Tiva-C Modem Driver
- Tiva-C UART
- STM32 Modem Driver
- STM32 UART
- nRF52 UART
- nRF52 Modem Driver
- Tiva-C SD Card Driver
- STM32 SD Card Driver
- nRF52 SD Card Driver
- Tiva-C SPI
- STM32 SPI
- nRF52 SPI
- Tiva-C Accel’ Driver
- STM32 Accel’ Driver
- nRF52 Accel’ Driver
- Tiva-C I2C
- STM32 I2C
- nRF52 I2C
- Tiva-C Keypad Driver
- STM32 Keypad Driver
- nRF52 Keypad Driver
- Tiva-C GPIO
- STM32 GPIO
- nRF52 GPIO
Common Peripheral Abstractions

- Modem Driver
- SD Card Driver
- Accelerometer Driver
- Keypad Driver
- UART HAL
- SPI HAL
- I²C HAL
- GPIO HAL
- Tiva-C HAL
- nRF52 HAL
- STM32 HAL
nRF9160 support

- We created nrf9160-hal
  - Upstreamed to nrf-rs
  - Exports parts of nrf52-common-hal

- The nrf52-common-hal uses nrf91 – a Peripheral Access Crate
  - Auto-generated with svd2rust

- Non-Secure now (maybe Secure Mode later)
What does the HAL look like?

```rust
/// Serial interface

use nb;

/// Read half of a serial interface
/// /// Some serial interfaces support different data sizes (8 bits, 9 bits, etc.);
/// /// This can be encoded in this trait via the `Word` type parameter.
pub trait Read<Word> {
    /// Read error
    type Error

    /// Reads a single word from the serial interface
    fn read(&mut self) -> nb::Result<Word, Self::Error>;
}

/// Write half of a serial interface
pub trait Write<Word> {
    /// Write error
    type Error

    /// Writes a single word to the serial interface
    fn write(&mut self, word: Word) -> nb::Result<(), Self::Error>;

    /// Ensures that none of the previously written words are still buffered
    fn flush(&mut self) -> nb::Result<(), Self::Error>;
}
```
Creating Safe Wrappers
Nordic’s API

• Modem access is via proprietary C library
• Has a Berkley-socket style API
  • AT Commands
  • GNSS Data
  • TCP / TLS / UDP / DTLS
  • NRF_AF_INET / NRF_AF_INET6 / NRF_AF_LTE / NRF_AF_LOCAL
• Use integers as the socket ID
• Very easy to leak sockets!
/**
 * @brief Function for creating a socket.
 *
 * @details API to create a socket that can be used for network communication
 * independently of lower protocol layers.
 *
 * @param[in] family The protocol family of the network protocol to use.
 * @param[in] type The protocol type to use for this socket.
 * @param[in] protocol The transport protocol to use for this socket.
 *
 * @return A non-negative socket descriptor on success, or -1 on error.
 */

int nrf_socket(int family, int type, int protocol);
Nordic’s API

```c
int update(nrf_sockaddr_in* p_addr, const void* p_data, size_t data_len) {
    int fd = nrf_socket(NRF_AF_INET, NRF_IPPROTO_TCP, NRF_SOCK_STREAM);
    if (fd < 0) {
        return fd;
    }
    int error = nrf_connect(fd, (void*) p_addr, sizeof(*p_addr));
    if (error < 0) {
        return error;
    }
    error = nrf_send(fd, data, data_len, 0);
    if (error < 0) {
        return error;
    }
    error = nrf_close(fd);
    if (error < 0) {
        return error;
    }
    return 0;
}
```
Our Rust API

```rust
fn update(host: &str, port: u16, data: &[u8]) -> Result<(), Error> {
    let socket = TcpSocket::new()?;
    socket.connect(host, port)?;
    socket.send(data)?;
    ()
}
```
Our Rust API

```rust
fn update(
    host: &str,
    port: u16,
    data: &[u8]
) -> Result<(), Error> {
    let socket = TcpSocket::new()?;
    socket.connect(host, port)?;
    socket.send(data)?;
    ()
}
```
What can it do?

- Simple command-line interface
- You can manually send AT commands to the modem
- You can get a GPS fix
- You can open a TCP socket and send an HTTPS GET request
Where do I get it?

https://github.com/42-technology-ltd
https://crates.io/crates/nrfxlib
42 Technology is a product development and engineering consultancy. We exist to solve difficult technical problems and develop new and exciting products for our clients.

We are a practical and pragmatic group that enjoys hands-on problem solving that gets our clients the answers that they need. Taking the time to understand their problems is therefore a very important part of our development process.

Our clients are experts in their own fields and we complement their deep domain knowledge with our broad experience of technology and innovation, as well as a healthy amount of commercial awareness. They tell us that we are a very agile team, able to respond quickly to their calls for help, and that responsiveness is an important part of 42 Technology’s success.

Our services span front end product and technology strategy, opportunity creation and feasibility analysis, turn-key product development, manufacturing process development, regulatory approval and transfer to manufacture.

The in-house team of engineers, scientists, designers and project managers is supplemented by an extensive network of associates and partner companies, according the specific needs of individual projects, ensuring that we assemble the right project team to deliver the best results to our clients.
## Document Version History

<table>
<thead>
<tr>
<th>Version</th>
<th>Changes from previous issue</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Initial Issue</td>
<td>13 November 2019</td>
</tr>
</tbody>
</table>