第5章 リンカ・ローダやアセンブリ記述はどうなるか

マイコンの下回りもRustで書く

中林 智之

マイコンの下回りのプログラムもRustで書く方法

ここまでは，用意されたスタート用パッケージ（quickstartクレート）を使いました．しかし，これでは本当にRustがハードウェアを直接制御しているのか，分かりません．こっそりC言語が下で動いているのではないかと疑われても仕方ありません．

そこで，スクラッチで，Rustによるファームウェアを作る方法も紹介しておきます．リファレンス・ボードにはnRF52840-DK（**写真1**，**写真2**）を使用し，ボード上のLED1を点灯します．nRF52840はARM Cortex-M4Fのコアが搭載されていいます．nRF52*xxx*系のSoCは何も設定しなくてもGPIOが使用できるため，今回のようにLEDを光らせるためだけの最小限の実機で動くサンプルを見せるのにうってつけです．

Cortex-M系のコアは，同じ要領でブートできるため，慣れ親しんだボードでも同様のことが可能です．

手順

● 作るもの

新しいクレート（Rustのパッケージ）を作成して，次の2つを作っていきます．

- リンカ・スクリプト
- Rustソース・ファイル

LEDを点灯するファームウェアは，ベアメタル・アプリケーションなので，リンカ・スクリプトが必要です．ただ，当然と言えば当然なのですが，リンカ・スクリプトはC言語でベアメタル・アプリケーションを作る場合と全く変わりません．ここで強調したいのは，Rustコンパイラがリンカ・スクリプトのような低レベルの仕組みを扱える，ということです．正確にはRustツールチェーンとしてインストールされるLLDがリンクを担当します．

LEDの点灯が動いた後は，開発を便利に進めるためにCargoの設定ファイルを作成します．

新しいクレートを作成し，その中にファイルを作成していきます．

```
$ cargo new nrf_blink↵
$ cd nrf_blink↵
```

作るもの1：リンカ・スクリプト

まず，リンカ・スクリプトから作成していきましょう．クレートのルート・ディレクトリに`linker.ld`を**リスト1**の内容で作成します．

後のRustソースコードとも関連する部分があるため，少しこのリンカ・スクリプトを解説します．

1行目～6行目でnRF52840のメモリ・レイアウトを定義しています．nRF52840には1024Kバイト（1Mバイト）のROMと256KバイトのRAMが搭載されています．

8行目～22行目は`RESET_VECTOR`（リセット・ベクタ）に関する記述です．リンカ・スクリプトに`reset`というシンボルの関数がエントリ・ポイントであることを教え，`RESET_VECTOR`シンボルを強制的にバイナリに埋め込みます．この`reset`と`RESET_VECTOR`はRustソースコードで作成します．

ARM Cortex-Mのリセット・シーケンスは，0番地（正確にはコード・メモリ領域の先頭）にスタック・

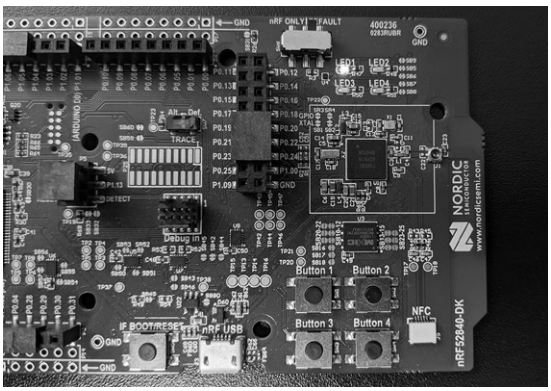

写真1　Cortex-M4Fマイコンにスクラッチで Rustプログラムを書いてLチカさせる