IoTフレームワークで組み込み開発に挑戦

IoT向きモダン言語 Elixir（エリクサー）の研究

第2回 ElixirのIoTフレームワークNerves（ナーブス）とは

髙瀬 英希

もはやC/C++だけでIoTや組み込みのデバイスを開発するのは大変な時代です．本連載では次世代のIoT開発手段として，並行・分散処理向きの軽量・低遅延な関数型のプログラム言語Elixirや，Elixirを使ったIoTフレームワークNerves（ナーブス）を紹介します．

Nervesはラズベリー・パイ（Raspberry Pi）やその他の多くの市販コンピュータ・ボードに対応しており，使いやすいと思います（**写真1**）．

IoTフレームワークNerves（ナーブス）とは

前回はElixir/ErlangをPCにインストールして，複数の方法で実際に動かしてみました．今回はIoTデバイス上で動作する最小構成の実行プラットフォームであり，Elixirによる組み込みIoTソフトウェア開発のためのツール・フレームワークである「Nerves（ナーブス）」を解説します．

メインの開発者は，Justin Schneck，Frank Hunleth，Greg Mefford，Connor Rigby，Jon Carstensという5名のエンジニアです．活発に開発が進められており，本記事執筆時点（2020年4月23日）での最新版はv1.6です（**図1**）(1)．

● 特徴

Nervesには，大きく分けて3つの特徴があります．

1つ目の特徴は，IoTデバイス上で動作する最小構成のブートローダ＋Elixir実行環境＋各種デバイス・ドライバの実行プラットフォームであることです（**図2**）．Nervesの実行基盤にはBuildroot系の組み込みLinuxカーネルを採用しています．このメモリ・フットプリントはたったの12Mバイト程度に収まり，Elixirアプリケーションを含めても数十Mバイトで済みます．Nervesファームウェアは，基本的にmicroSDカードに書き込んで起動することになります．

2つ目の特徴は，Nervesはメモリ・サイズだけでなく，堅牢なシステムを構築できるところにも強みを持つことです．まず，Nervesにはネイティブなエラング（Erlang）VMが移植されています．これによってErlang VMの持つ耐障害性と並列性能，そしてソフト・リアルタイム性をNervesシステムでも享受できます．改めて**図2**に注目すると，ブート領域とルート・ファイル・システムはA/Bの領域として2重化されています．基本は片方の領域でアプリケーションが実行されますが，システム全体での障害が発生した際には，もう片方の領域に実行を移して速やかに障害復旧と再起動が行える仕組みがあります注1．

3つ目の特徴は，冗長になる不要な機能は持たないようにすることで，システムの堅牢性をさらに高めていることです注2．

注1：片方の領域でアプリケーションを実行しながら，もう片方に新しいファームウェアを更新する仕組みもあります．更新後の再起動後には，もう片方の領域のファームウェアを更新することで2重化が保たれます．

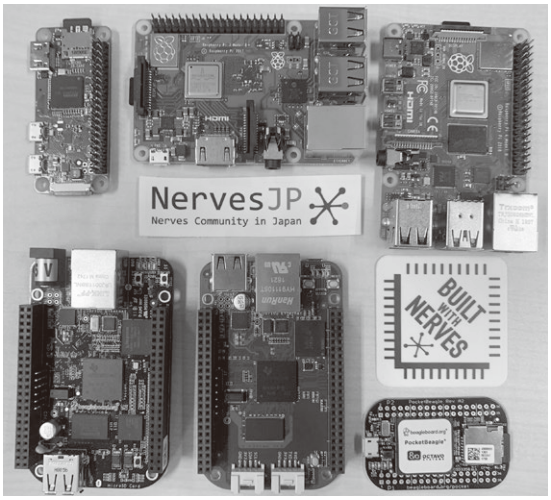

写真1　筆者おすすめのIoTフレームワークNervesはラズパイなどのさまざまなコンピュータ・ボードに対応している
左上から，Raspberry Pi Zero WH，Raspberry Pi 3B＋，Raspbbery Pi 4，BeagleBone Black，BeagleBone Green，PocketBeagle