組み込みRustのライブラリ

便利クレート探偵団

第4回 メモリ使用量や最悪実行時間が見積もれるheapless

中林 智之

リスト1 静的変数としてメモリを割り当てる例

```
use heapless::Vec;

fn main() {
    let mut xs: Vec<u8, 4> = Vec::new();
    // 省略

    static mut XS: Vec<u8, 4> = Vec::new();
    // 省略
}
```

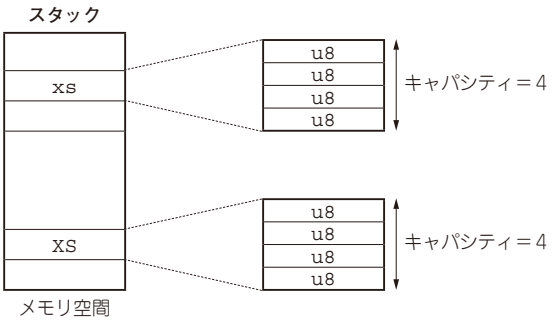

図1 メモリ割り当ての例

Rustは組み込みで使えるプログラミング言語として注目されています．本連載ではそんなRustの組み込み開発で役立つライブラリ（クレート）を紹介します．

PCなどで使うRustは，メモリ・アロケータ（C言語で言うmallocなどのメモリ確保API）が使えるので，一般的なコレクション（可変長配列や双方向リストなど）が使えます．しかしマイコンをターゲットとした組み込みRust（no_std環境）では，そのままではメモリ・アロケータを使えないため，コレクションも使えません．

そこで今回はメモリ・アロケータなしでも便利なコレクションが使えるheapless(1)について解説します．

静的メモリ割り当てに向いているheapless

前回はメモリ・アロケータとしてembedded-allocを組み込み，動的メモリ確保をできるようにすることでallocのコレクションを使う手順を紹介しました．

今回紹介するheaplessはメモリを動的に確保しないため，より静的なメモリ割り当てに向いたクレートです．標準ライブラリの動的メモリ確保を行うstd::Vecのようなコレクションと比較すると次の利点があります．

1. 実行前にメモリ使用量が見積もりやすい
2. 最悪実行時間が見積もりやすい
3. メモリ不足の処理が明示的に行える

それぞれのメリットを詳しく見ていきます．

● メリット①：実行前にメモリ使用量が見積もりやすい

heaplessはメモリに（型パラメータNのキャパシティで指定する）固定サイズのメモリを確保して，その容量内で要素数を増減できるコレクションを提供します．

メモリはスタック内または静的変数として確保します注1．例えば，**リスト1**の例でxsは通常のローカル変数同様スタックに，XSは静的変数としてメモリを割り当てています．

これを図示すると**図1**のようになります．スタック内または静的変数として，u8が最大4つ格納できるメモリ領域を確保します．一度メモリを確保した後は，4つまでなら要素数を増減できる可変長配列のように扱うことができます．つまり，heaplessで利用するメモリは，スタックに積まれる関数内のローカル変数，または静的変数となります．

これらは静的にメモリ使用量が見積もり可能であるため注2，プログラムを実行しなくてもメモリ使用量の最悪ケースを見積もることができます．特にRAMが少ないマイコンにとっては意図しないサイズの動的メモリ確保でメモリ不足に陥ることがないため，あり

注1：ヒープ領域に確保することもできますが，固定の領域を割り当てるので意味がありません．

注2：関数ポインタや再帰関数を使わない限り．

第1回 マイコンなどリソースに制約があるデバイスを対象としたロギング・フレームワーク（2024年3月号）
第2回 便利機能満載の汎用デバッグ・ツールキットprobe-rs（2024年4月号）