Picoの起動処理の流れ

豊山 祐一

ラズベリー・パイPico（以降，Pico）の起動処理について具体的に説明します．また，起動処理を作る上で必要なプログラムのメモリへの配置とリンカ・スクリプト・ファイルについても説明します．

Picoのメモリ・マップ

図1にCortex-M0+の一般的なメモリ・マップと，これに対するPicoのメモリ・マップを示します．

先頭には16Kバイトの ROMがあります．ROMの内容は出荷時に決められており，ユーザは変更できません．このROMに例外ベクタ・テーブルやリセット・ハンドラが存在します．

外部フラッシュ・メモリはQuad SPIインターフェースで接続されていますが，XIP（Execute-In-Place）というハードウェア機能により，通常のROMのようにアドレス空間にマップされてプログラムを実行できます．ここにユーザのプログラムを格納します．

外部フラッシュ・メモリの先頭256バイトは，後述するセカンド・ステージ・ブート・ローダのプログラムを配置することに決められています．さらにその後ろにユーザ定義の例外ベクタ・テーブルと，ユーザのプログラムを配置します．

注意すべきは，ROMにある例外ベクタ・テーブルと外部フラッシュ・メモリの例外ベクタ・テーブルは別物ということです．前者はメーカが作成しユーザからは変更できないものです．後者はユーザが自身のプログラムに応じて作ったものです．

起動処理のステップ

Picoの起動処理の流れを**図2**に示します．

● ROM上のリセット・ハンドラを実行［図2（a）］

RP2040は電源が投入されリセットが発生すると，ROMのリセット・ハンドラが実行されます．Picoが通常モードの場合はファースト・ステージ・ブート・ローダが実行されます．

ファースト・ステージ・ブート・ローダでは，外部フラッシュ・メモリの先頭にあるセカンド・ステージ・ブート・ローダをSRAMに転送して実行します．転送の際にチェックサムの検証も行います．

● XIPの設定［図2（b）］

セカンド・ステージ・ブート・ローダでは，まず外部フラッシュ・メモリ上のプログラムの実行効率を最適にするためにXIPを設定します．セカンド・ステージ・ブート・ローダをSRAMに転送して実行するの

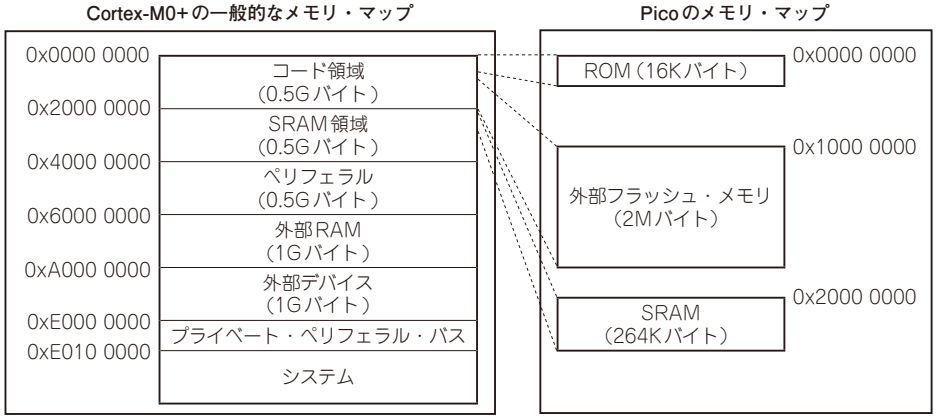

図1
Picoのメモリ・マップ
Cortex-M0+のメモリ空間のうち，Picoではその一部にメモリが搭載されている