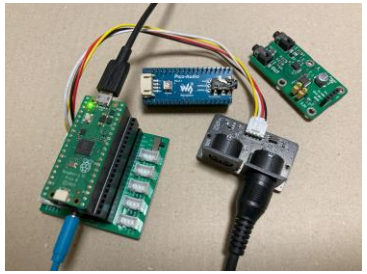
PRA32-U：Arduino UNO R3用シンセサイザーのRaspberry Pi Picoへの移植と改善

石垣 良　2023年8月31日 SWEST25



PRA32-U v0.4（プロトタイプ）仕様

- 4音ポリフォニックモードに対応、コーラスエフェクトを搭載
- Arduino IDEで開発
  - Arduino-Picoを使用
  - C/C++ SDKは開発環境の扱いが難しい
- 制御方法：USB MIDI（またはUART、MIDI）
  - Adafruit TinyUSB、MIDI Libraryを使用
- オーディオ出力：I2S（Inter-IC Sound）48 kHz/26 bit
  - 市販のDAC基板（Pimoroni Pico Audio PackやWaveshare Pico-Audio）を使用
- ソフトウェアライセンス：CC0（フリー）

VRA8-Uからの変更ポイント

- サンプリング周波数：31.25 kHz → 48 kHz
- オーディオ出力：PWM 8/16 bit → I2S 16 bit（発音レイテンシ：0.03 ms → 2.7 ms）
- 疑似的でない、ポリフォニックモードに対応
- フィルターのカットオフ周波数変化：5 → 10+ オクターブ
- 機能追加、各パラメータの調整カーブを変更
- 条件分岐を減らして、最大CPU負荷の見積り容易化
- コードの読みやすさを改善（まだまだ「やっつけ」コードが多い…）

最適化のポイント

- **関数 setup1(), loop1()**　信号処理はArduinoランタイムの影響を受けにくいコア1で行う（コア使用率90%弱）。コア0はバスを共有しており、コア1の処理を遅らせる可能性があるので、信号処理では使用しない。
- **関数属性 __attribute__((always_inline))** ほぼ全ての関数を強制的にインライン展開。
- **マクロ __not_in_flash_func()**　信号処理コードをフラッシュメモリでなく、高速アクセス可能なSRAMに配置。エントリ関数だけにこのマクロを付ければ、インライン展開との合わせ技で、ほぼ全ての関数がSRAMに載る。TinyUSBやMIDI Libraryのコードはフラッシュに置かれてしまうが、XIPキャッシュに載ることを期待。

参考文献

- 石垣 良；リアルタイム処理のために軽量化！シンセサイザの製作，ラズベリー・パイPico/Pico W攻略本，2023年，CQ出版社，pp.162-173.
- 石垣 良；音の時間変化に対応したシンセサイザ作り，ラズベリー・パイPico/Pico W攻略本，2023年，CQ出版社，pp.174-183.

作り方