ワンチップマイコン用ブートローダの製作

電子技術科　浦野　勉
杉山　智聡
吉田　慶一

1 研究目的

最近のワンチップマイコンは，プログラムを記憶するためにフラッシュメモリが搭載されている．マイコンは，このフラッシュメモリに記憶された命令を読み出して実行している．図 1 に示すように、実行用マシン語ファイル（インテルヘキサファイル）は PC 上の統合開発環境により生成され，プログラマと呼ばれる装置によりフラッシュメモリに書き込まれる．

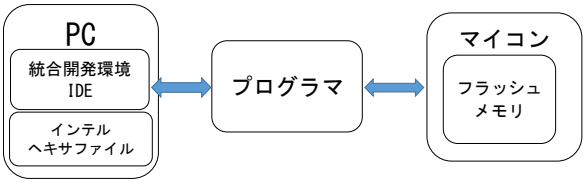

図 1　プログラマを使った構成

一方，図 2 に示すような、マイコンそのものが PC と直接通信してマシン語データを受け取り，自らフラッシュメモリに命令を書き込み(プログラムのロード)，その後実行モードに入るブートローダを用いるという方法もある．ブートローダを使用すると，開発にプログラマが不要になる．(ただし，ブートローダを書き込むためにプログラマが 1 台は必要）これにより，実習時に学生の人数分高価なプログラマが必要であったが，最低 1 台あれば良いことになり，コスト削減となる．

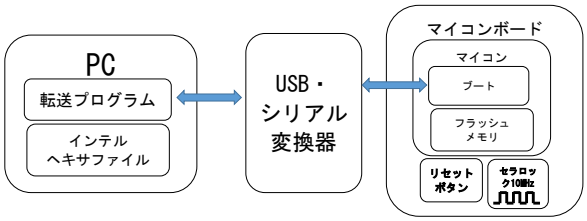

図 2　ブートローダを使った構成

2 製作物の概要

多くのワンチップマイコンは，シリアル通信機能を持っている．そこで，最初に数百円程度の安価で基板実装可能な USB－シリアル通信インターフェースを含むマイコンボードの設計・製作を行った． この構成は，ブートローダ機能だけでなく，プログラム書込み後に，PC 上のアプリケーションとマイコンとの通信手段としても活用でき、デバッグやプログラムの動作の理解に役立つ．

次に，マシン語ファイルのフォーマットであるインテルヘキサファイルを読み込んで，通信用データに変換し送信する，「転送プログラム」を開発した．最後に，マシン語データを受信してフラッシュメモリに書き込みプログラムを起動する「ブートローダプログラム」を作成し，システムを完成させた．

3 システム構成

3.1 ハードウェア

図 2 に示したとおり，システムは，PC，USB シリアル変換器，マイコンボードの 3 つからなる．マイコンは現在実習で使用している MicroChip 社の PIC18F4620[*1] を選定し，外部クロックとして 10MHz セラロックとリセットボタンを設置した．

USB シリアル変換器はマイコンボード側に 5 V 電源を供給するとともに，PC 上に仮想シリアルポートを作成する．ユーザは，PC 上でソースプログラムをコンパイルし，生成したヘキサファイルを転送プログラムに渡す．転送プログラムは，仮想シリアルポート経由でマイコンとシリアル通信を行う．マイコン側のブートローダは，転送プログラムからのデータを受け取り，指定されたアドレスのフラッシュメモリにデータを書き込む．なお，USB・シリアル変換器は秋月電子通商（株）AE-CH340E-TYPEC を用いたが，PC 側のドライバが供給されているものであれば，どのようなものでもよい．

3.2 ブートローダの構成

ブートローダは，ヘキサファイルの機械語データを受け取り，それをプログラム用フラッシュメモリに書き込む．それが終わるとユーザプログラムを起動する．この動作を実現するための 4 つの機能について以下に述べる．

（1）シリアル通信

ボーレート 115200bps，データ 8bit，ストップビット 1bit，パリティ，フロー制御なしの設定で，ごく基本的なデータの読み書き用の関数を作成した．なお，処理が通常完了した後に ACK を返送することで，プログラム転送のタイミングを取っている．

（2）フラッシュメモリ書き込みと起動

マイコンの仕様上，プログラム用フラッシュメモリへの書き込みは，1 ブロック 64 バイト単位で行わなければならない． また，1 ブロックに書き込む前に，そのブロックのデータをクリアしておく必要がある．データを書き込むには，64 バイトの特殊なバッファにデータをコピーし，フラッシュメモリ書き込み命令により 1 ブロックの書き込みが完了する．1 ブロック約 2ms の時間がかかるが，この間は他の処理をしてはならず，割込みも禁止する必要がある．この操作の一部は C 言語では記述できないため，インラインアセンブラで記述した．ユーザプログラムは 0x5000 番地をプログラム開始アドレスに設定したので，ロード終了後に，0x5000 番地へジャンプすれば，ユーザプログラムが実行される．この部分の記述も一部インラインアセンブラを使用した．

（3）割込み処理

このマイコンは，割込みが発生すると 0x0008 番地のルーチンをコールするようになっているが，ユーザプログラムの割込みは 0x5008 番地に設定される．したがって，0x0008 番地には 0x5008 番地にジャンプするプログラムを記述する．

3.3 転送プログラムの構成

このプログラムは，ヘキサファイルからデータを読み込み，ブートローダに対して転送するもので，Python 言語で記述した．

（1）ヘキサファイルの処理

転送機能を実現するために Alexander Belchenko が提供する「Python Intel Hex Library[*2]」を用いた．ファイルの読み込み，使用アドレス範囲の計算，アドレスとデータの対応等転送に必要な処理を非常に簡単に記述することができる．

（2）データの分割と転送

データの通信は USB シリアル変換器による仮想シリアルポートを介して行うので，転送プログラムはシリアルポートに対して送受信を行えばよい．Python 用シリアル通信パッケージとして，Chris Liechti が提供する pySerial[*3] を利用した．これは，通信設定とオープン，データの送受信等を非常に簡単に行うことが出来る．転送データは，64 バイトを 1 ブロックとし，それをシリアル通信で転送する．そして，ACK 応答が確認できたなら次のブロック転送を行う．全てのブロックの転送が終わると，ユーザプログラムが実行される．

3.4 マイコンプログラム用フラッシュメモリの構成

図 3 に，ユーザプログラム転送後のプログラム用フラッシュメモリの状態を示す．今回は，フラッシュメモリの 1/3 をブートローダに，2/3 をユーザプログラムに割り当てることとした為，ユーザプログラムの先頭は 0x5000 番地となった．

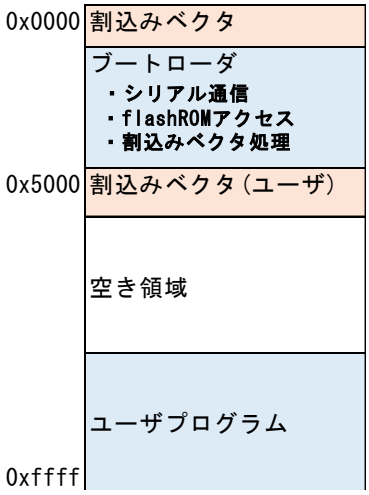

図 3　フラッシュメモリの構成

4 まとめ

PIC マイコンに書き込んだブートローダを使って，ユーザプログラムを書込み，実行できることを確認した．本システムにより書込み装置(プログラマ）を省略することができ，効率的なプログラム開発が可能となった．まだ試行したプログラムが小規模なものなので，もっと大規模で複雑なプログラムでも実行可能か検証した上で，来年度のマイコン実習で活用していきたい．なお，今回は成功しなかったが，ブートローダ領域のルーチンをユーザプログラムに提供できるようになれば，メモリや転送時間の節約となり，より開発効率が高まると考えられる．

また，ブートローダそのものを学習することで，メモリとアドレスの取り扱い，ポインタの取り扱い，インラインアセンブラなど通常の実習では触れないような内容も体験できるので，より高度な制御プログラミングを目指す学習者には良い教材になる．

［参考文献］

(1) Microchip Technology Inc. 2008
PIC18F2525/2620/4525/4620 Data Sheet

(2) Alexander Belchenko 2005-2018
Python IntelHex Library User Manual
https://python-intelhex.readthedocs.io/en/stable/

(3) Chris Liechti 2001-2020 pySerial’s documentation
https://pyserial.readthedocs.io/en/latest/