不揮発性メモリ ROMの歴史

マイコンのメモリは、RAMと ROMがあります。RAMは、Ramdom Access Memoryの略で、RAM上の どのアドレスでも自由に読み書き、それも高速にアクセス出来ます。

それに対し、ROMは Read Only Memoryで読み出し専用メモリという事になります。

読み出しは、RAMとほぼ同様に、どのアドレスでも自由に、読み出せます。

それに対し書き込みは、ROMの種類に応じていろいろ制約があります。 初期の UVEPROM は 窓付きROMとも呼ばれ、データを消去する際は 窓部分に、強力な紫外線を当てて消去します。右上画像の 窓付きROM 27C32A(4K byte)は、5V単一で 書き込み出来ます。

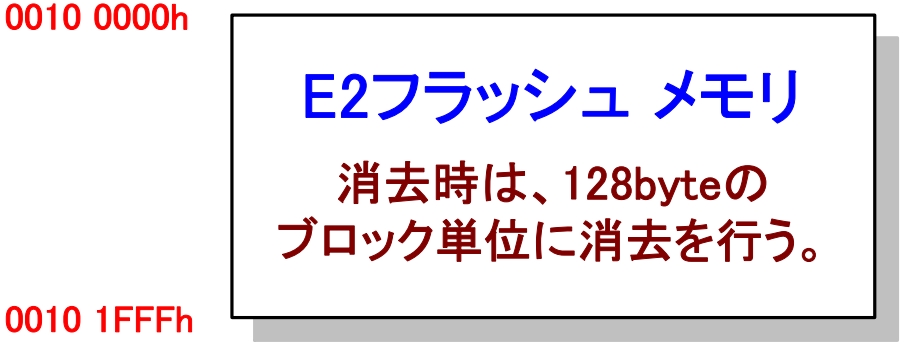
遥か昔の 2708(1Kbyte)などは 書き込み時 12Vと、書き込みパルスが 26V 必要で、書き込み時のシーケンスも ややこしいようです。

余談ですが、私は遥か昔 2732を 仕事で使っていた時期が あります。組み込み用途で Z80マイコンに ROM 4K、RAM 2Kを実装してプログラムを作っていました。 ROMも RAMも容量が小さいのでプログラムは、アセンブラで作る事になります。 左の 24C1024は I2C接続の電気的に消去できる EEPROMです。 容量は 131K byteあります。 2732よりは、だいぶ後の製品と 思います。

昔は CPU、ROM、RAM、周辺回路が 別々のLSIだった。

それが、今の組み込みマイコンは、CPUにROM、RAM、周辺回路が、ワンチップになっているので、便利で制御CPU基板が、コンパクトに作れます。 低消費電力で、実行速度も速くていい事ずくめのようですが、世の中の進歩に合わせ高度な技術が実装されるので、技術者にとっては、理解するのに 骨が折れる要素も増えて来たような気もします。

余談はさておき、RX220の E2フラッシュメモリ（データ格納用）ですが、メモリアドレス配置は 0x0010 0000h ～ 0x0010 1FFFh の 8Kbyteです。 消去時は、先頭アドレスから、128byteサイズのブロック単位で、消去する事になります。 それより細かい単位では消去出来ません。 よって全体が 8Kbyteなので、64ブロックある事になります。



それと 書き込み、読み出し時は、ワード単位（ 2byte単位 ）で、アクセスする必要が ありま す。 よって、読み書きするデータのサイズは偶数 byteにしておいた方がいいです。

ちょっと余談ですが、RX220以前の RXシリーズでは、E2フラッシュメモリの配置アドレスが書き込み時と、読み出し時で異なるという変な仕様になってましたが、RX220は、書き込み時と読み出し時は、同じアドレスです。

因みに フラッシュメモリは、書き込み回数の寿命が ありますね。 USBメモリや、SDカードにも ありますよね。

RX220の プログラムコード用フラッシュメモリは、最低 10,000回以上書き換え可能となっています。データ用 E2フラッシュメモリは、最低 100,000回以上書き換え可能と なっています。

同じマイコンを、デバッグで 何回も使い回していると、いずれ忘れた頃に寿命が来る。という事になります。

それとは、別の話ですが、秋月電子のRX220マイコン基板の説明書には、5V電源を供給するように書いてありましたが、データシートを見ると 3.3Vでも動くようなので、試してみたら 3.3Vで 正常に動作しました。

前回の SPIデバイスのMCP23S17、MCP4922、SRAM 23LC512 及びMCP23017 も 3.3Vで、動く事を確認しました。

何故、3.3V動作を確認したかというと 今回使用した I2C接続の 16文字2行の OLED表示器の電源電圧が、3.3Vだったからです。

それとデバッグ時に気付いたのですが、Renesas Flash Programmerにて プログラムを書き込むと、E2データフラッシュも消える事を確認しました。

プログラム書き込み後、必要であれば、パラメータデータも、プログラム実行時に、パソコンから マイコンに転送して、E2フラッシュに書き込む事になります。

では、今回の E2データフラッシュの アクセスプログラムの概要を 次ページから説明します。

今回の E2データフラッシュアクセス処理

今回の E2データフラッシュアクセス処理は、簡易版になります。

フラッシュメモリに格納するデータは、用途として 半固定的な プログラムの設定を行うパラメータ的なデータと考えます。

よって、E2データフラッシュ先頭に、１個パラメータデータを格納する 単純な用途を想定しています。 そうする事により、E2フラッシュの書き込みアドレスや、消去ブロックの切れ目をサブルーチンを 使う方が、意識しないで使えるからです。 サブルーチンは ３本です。

```
void  e2f_init( void );
                  // E2フラッシュ初期化
void  e2f_write( void *data, int siz );
                  // E2フラッシュ書き込み
void  e2f_read( void *data, int siz );
                  // E2フラッシュ読み出し
```

書き込むパラメータデータは、構造体変数にしておくと便利です。

```
//  テストパラメータ データ構造体
typedef struct {
      Uchar id, ver;   // データ確認用
      char  tx0[18];   // OLED表示文字列１行目
      char  tx1[18];   // OLED表示文字列２行目
      Ushort buf[256]; // 2バイト整数配列
} TEST_PARAM_DATA;

  TEST_PARAM_DATA  Tpm;  // 構造体 変数宣言

// E2フラッシュ初期化
    e2f_init();

// E2フラッシュへ  書き込み
e2f_write( &Tpm, sizeof( Tpm ) );

// E2フラッシュから、読み込み
e2f_read( &Tpm, sizeof( Tpm ) );
```

これ以上、簡単な使い方はないと思います。

e2f_write関数、e2f_read関数の引数部分ですが
(void *data, int siz);
に、なっています。 void *data は、任意に宣言した構造体変数の アドレスが 都度キャストしなくても渡せるように void * としました。 型は、分からなくても 全体のサイズも、渡しているので、書き込むバイト数は 分かります。 行儀の悪い Cプログラミングかもしれませんが、ご容赦下さい。

関数の中身の説明は省略します。
中身は、殆どが E2フラッシュ周辺回路レジスタの設定なので、ソースを見ても何やっているのか分からないと思います。 RX220のデータシートと にらめっこする必要があります。

あとは、動作確認の動画の前に、動作確認のプログラムの動作シーケンスを、簡単に説明しておきます。
E2フラッシュに書き込むデータは、前ページの TEST_PARAM_DATA Tpm の 構造体変数を使用します。

パラメータ データ初期化は

```
Tpm.id = '@';       // パラメータ識別子
Tpm.ver = 10;       // バージョン番号
str_copy( &Tpm.tx0, "2022-10-05" );
str_copy( &Tpm.tx1, "Test Project." );
for( i=0; i<256; i++ )
{
  j = ~i;  j = j << 8;  j |= i;
  // 変数 j の 下位8bit = 0～255の値
  // 上位8bit = 下位バイトのビット反転値
  Tpm.buf[i] = j;
}
```

上記のように行います。メンバー変数 id と ver と 256個の配列変数には 16 bit の全てのビットを 設定するデータを格納しています。 E2フラッシュから Tpm 構造体変数を読み出したとき、上記設定値が、入っているかチェックを行います。 メンバー変数 tx0 と tx1 は、チェックから、外れています。 電源ON直後の処理で、E2フラッシュから Tpmを 読み出し内容のチェックを行います。 データ化けが確認されたら、テラターム側に、"* Invalid Parameter."を 表示します。

起動時、パラメータデータを正常に読み込めたら
"* The parameter is normal."
を、テラターム側に送信し、パラメータ内の tx0 とtx1 文字列を 16文字2行の OLED表示器に表示します。 この tx0 と tx1 の文字列は、テラターム側で入力して、Tpm変数に設定して、Tpm変数を E2フラッシュに書き込む操作が出来ます。
操作メニュー表示は、 "1: Pm Clear , 2: Pm Initial , 3: Pm Text In , 4: Pm OLED Disp , 9: Pm Write" があります。番号で選択します。 テストでは、

① ２ の Pm Initial を 行う。

② ３ の OLED表示の 2行の文字列 入力を 行う。

③ ４ で OLEDに正常に表示されるか確認する。

④ ９ で E2データフラッシュに Pm を 書き込む。

⑤ 一旦、マイコンの電源を切り、再度 電源ONで OLEDに、設定した文字列が表示される事を確認する。 正常に表示されたら、OKです。
これにより、電源を切っても E2フラッシュによって パラメータを、保持されている事になります。

では、確認動画をごらん下さい。