前回 060 動画の訂正

前回の動画「060 RX220 SPIアクセス ソフト編」の　4ページ右側の文章に、明らかに訂正した方がいい間違いが、動画投稿後に 見つかりましたので、訂正します。

先頭のコマンドバイトの説明で、「よって コマンドバイトは、」の、次の
「Write時、0x41、Read時は、0x40 になります」
を
「Write時、0x40、Read時は、0x41 になります」
に、訂正します。

申し訳ありませんでした。（v_v;）

MCP23S17の 先頭コマンドバイト、SPIのデバイスであるが、コマンドは、I2C互換のフォーマット。　薄緑の枠の 0100 は固定。
A2、A1,A0は、端子を GNDに接続しているので 0、0、0 に なる。
R/Wは、1 の時 Read ／ 0 の時 Write になる。

0	1	0	0	A2	A1	A0	R/W

よって、Read時は、0x41 ／ Write時は、0x40 に なります。
　それと もう一つ、前回の動画の4ページの説明先頭部分で、「内部に 22個のレジスタ」と書いてましたが、実際は　21個です。
　IOCONというレジスタに 2つアドレスが、割り振ってあるので、アドレスだけ数えると　22個に　なります。

MCP23S17と MCP23017の違い

シリアル通信のホストインターフェースが MCP23S17が SPI で、MCP23017が I2Cです。

元々は、I2Cインタフェースのデバイスとして作られたようで、基本的にバイト単位のコマンド、パラメータのやり取りを行います。

データシートの タイトルが、
MCP23017／MCP23S17
16bit I/O Expander with Serial Interface と と、なっており、ホストインターフェース以外の説明は、共通です。 という事は、このデバイスを制御するコマンド、パラメータの出し方は MCP23017 と MCP23S17 で 共通です。

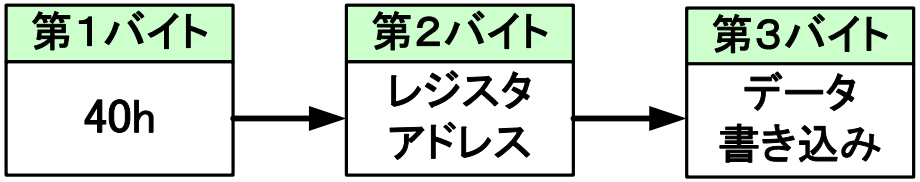
このデバイスで、一つ厄介なのは、 内部レジスタのアドレスが、BANK0 と BANK1 の2つの状態をもっている事です。 この2つの バンクのどちらを選択するかの設定が、先ほど出て来た IOCON レジスタです。 そしてこの IOCONレジスタのアドレスも、BANK0 と BANK1 で 異なります。
BANK0 時の IOCONのアドレス: 0Ah と 0Bh
BANK1 時の IOCONのアドレス: 05h と 15h

つまり、現在どちらの バンクに なっているかが分からないと、バンクを設定する IOCONも アクセス出来ないのです。

前回の動画で、MCP23S17を 動かす事が、出来なかったのは、この BANK0 と BANK1 そして IOCON を よく理解してなかったためです。

MCP23x17の Bank設定コマンド

まず、コマンドの出し方ですが、
3バイト連続して転送します。



それと、電源ON直後の バンク設定が、どうなっているのかを、知る必要があります。

そのためには、電源ON直後の IOCONレジスタの内容を知る必要があります。 データシートの P17に「制御レジスタのまとめ」の表が、有ります。 IOCONの行の 一番右端の POR/RSTの 値を見ます。 これが 電源ON直後のIOCONの値です。0000 0000 となっています。

IOCONレジスタの 各ビット:

b7	b6	b5	b4	b3	b2	b1	b0
BANK	MIRROR	SEQOP	DISSLW	HAEN	ODR	INTPOL	—

上記、IOCONの b7 が、バンクの選択ビットです。 IOCONの 電源ON 初期値が 0000 0000 なので、b7 = 0 で、BANK0 が、選択されています。BANK0 の場合は IOCONのアドレスは 0Ah に なります。BANK0 は、連続して複数バイトデータを書き込むと レジスタアドレスの オートインクリメントが出来るようですが、今回は単純に バイト単位で アクセスしたいので、BANK1 に変更する必要があります。 IOCONの その他のビットは、データシートを参照してください。
では、次のページに BANK1 に変更して、初期化するコマンドを出す手順を、示します。

MCP23x17の Bank設定含む初期化処理

MCP23S17 or MCP23017の 初期化処理を 示します。
まずは、電源 ON 直後の状態で BANK0 である事を 想定して記述します。

① 40h、0Ah、80h　// MCP23S17のモード設定 IOCON で BANK1に切り替え
// この後、 IOCON の アドレスは、05hに 変わっている。
// BANK=1 に 変更した後に、本来の IOCON 設定を行う。
② 40h、05h、BCh　// MCP23S17 IOCON の モード設定

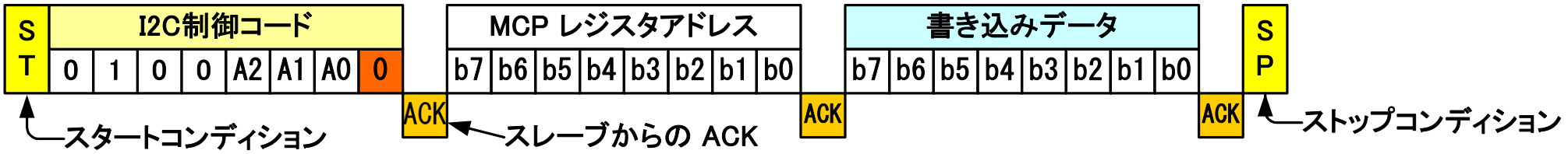
③ 40h、01h、FFh　// GPA 入力極性設定 負論理
④ 40h、00h、FFh　// GPA 方向指定＝入力

⑤ 40h、19h、FFh　// GPB Port Data = FF (LED点灯が負論理のため)
⑥ 40h、10h、00h　// GPB 方向指定＝出力
// ここで、初期化は終了(BANK1 バイトモードで 割り込み機能は 使用してない。)
//----------
⑦ 40h、19h、35h　// GRB LEDポートに 35h を出力
⑧ 41h、09h、Read Data　// GRA ポートから、1バイト読み込み
// 1バイト読み込みは、I2C と SPI とでは、シーケンスが 異なります。
// 次ページで、説明します。

I2Cの 電文シーケンス

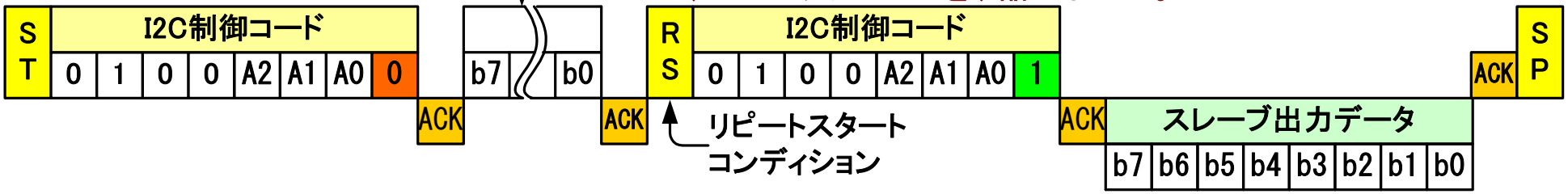
まず、I2Cの レジスタ指定の 1バイト送信と
レジスタ指定の 1バイト受信の例を 示します。

1バイト書き込みシーケンス



1バイト読み出しシーケンス

そのままでは、4Byte入りそうになかった
ので、レジスタアドレスを、縮めました。

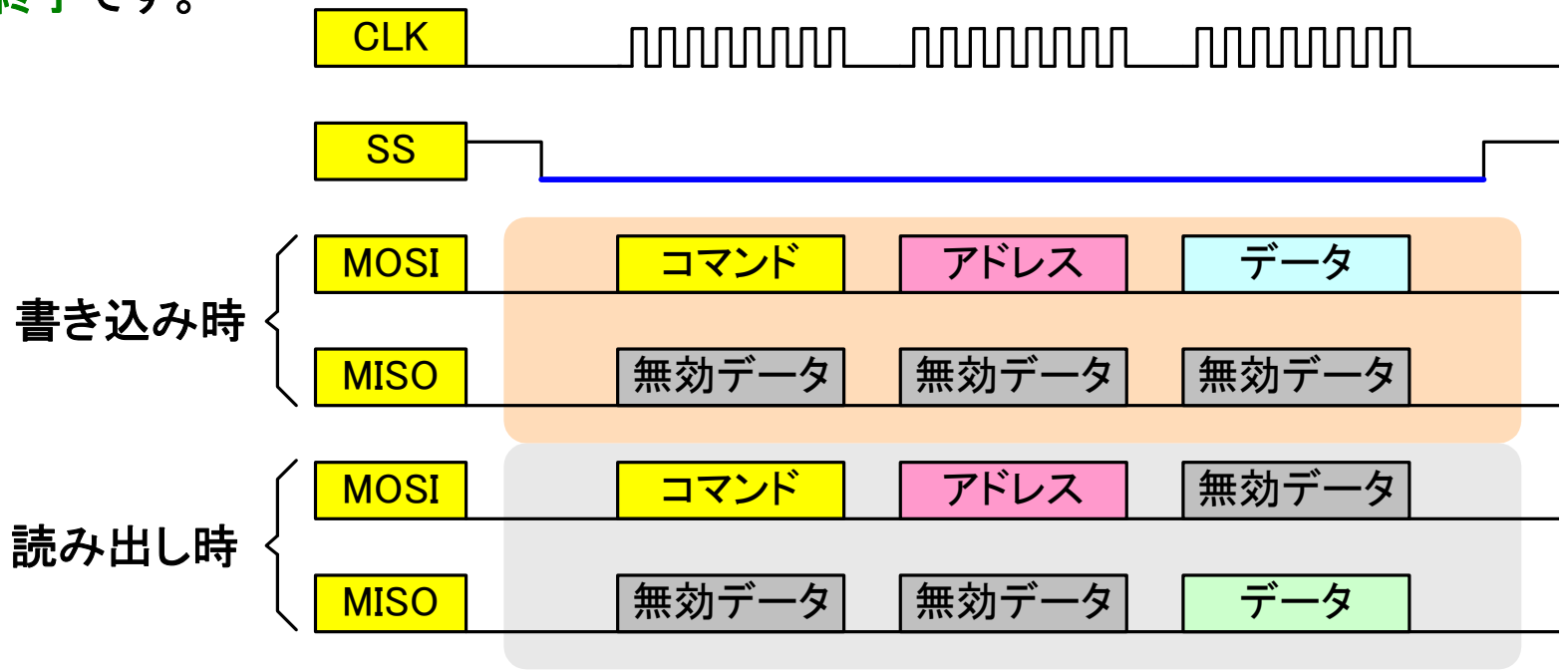


I2Cは、2線式の関係で、データを読み出す場合は、レジスタアドレスまで 転送して、
途中で リピートスタートコンディションを使い、2番目の制御コードで、転送方向を 切り替え、
スレーブのデータを受信します。 よって、読み出し時は 4バイトのデータを やり取りします。

SPIの 電文シーケンス

SPIの方が、書き込み、読み出しともに 3BYTEの 転送でシンプルです。 実は、SPIは、3線式で 常時 全二重通信を行っているので 転送方向 切り替えの必要が 無いのです。

マスタ側で、データを受信したいなら、取り出せばいいし、データは、送ってくるけど必要無いなら空読みすればいい。 という事です。 それと、I2Cには、スタート、ストップコンディションが、ありましたが、SPIの場合は、SS信号が Low になったら 通信の開始で、SS信号が Hi に なったら通信の終了です。

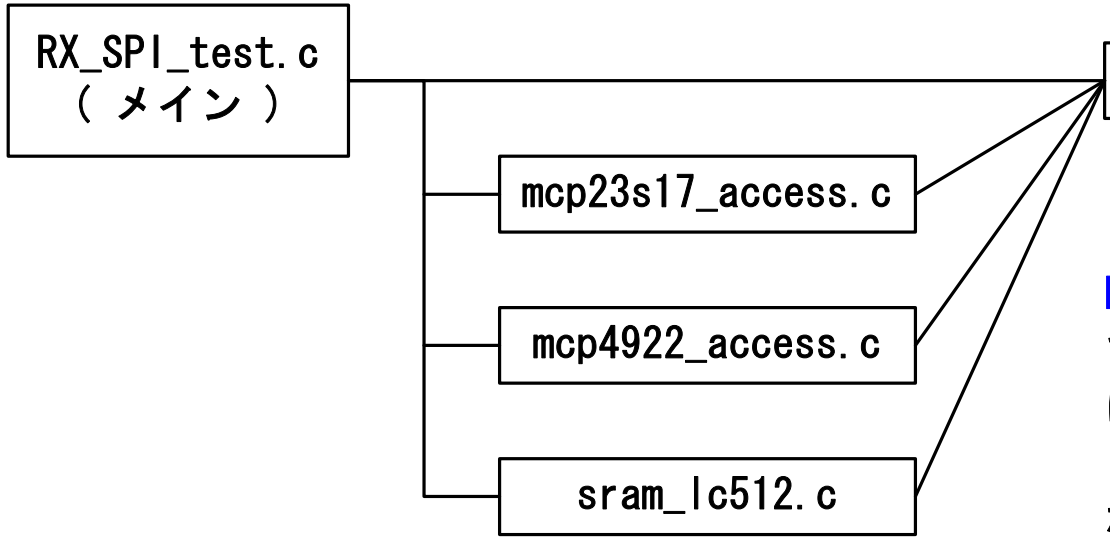


無効データは、スレーブ側が、受信中と判断しているので、スレーブ出力は 0 を出している事が多いです。

マスタが受信する時も、ダミーで 0 を 出力してます。

RSPIと 各SPIデバイスのソフト階層図

今回の HEWプロジェクト内のメインモジュールと
SPIデバイスアクセスに関わるモジュールの階層図です。
他に、シリアル通信と、インターバルタイマを使用してます。
ソースファイルを、参照したい方のために用意しました。



RX22_iocs_rspi.cでは、割り込み処理を３つ登録しています。　割り込み処理内ではRSPI周辺回路のフラグを クリアしてるだけです。 この Project 内の intprg.c も必要になります。

３つの SPIデバイスのうち、MCP23S17だけが、初期化処理を行っています。この初期化処理は、割り込み処理が使える状態で初期化を、行って下さい。
でないと フリーズします。