MSX0 キーボード 新プロトコル案

2024 年 5 月 3 日　HRA!

目次

1. はじめに

MSX0stack クラファン版（以降、MSX0 と表記）に付属の FaceII キーボードは、MSX のキーボードとして見た場合に下記のような問題点があります。

1. 入力できないキーがある
2. 従来 MSX では同時に入力できるが FaceII キーボードでは同時に入力できないキーの組み合わせがある
3. FaceII キーボードと FaceII ゲームパッドが排他である

「無ければ作れば良い」という観点で、作ってみることを試みてみましたが、FaceII キーボードと MSX0 との通信プロトコルが FaceII キーボードのハードウェア的な都合に合わせたプロトコルで有り、上記課題を通信プロトコル自体が抱えていることから、MSX0 側の対応を変えない限りは改善が不可能であることがわかりました。

そこで、「必要であれば、必要な人が作れば良い」という入り口を作るために、新しいプロトコルを提案させていただきます。

2. 新プロトコルの方針

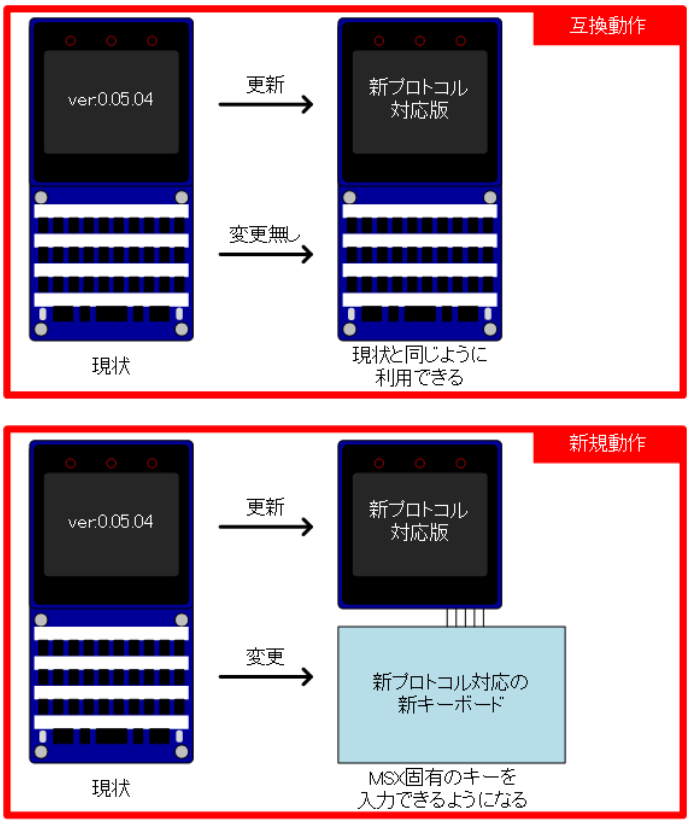
新プロトコルは、下記の方針に基づいて設計されています。

1. 既存プロトコルと互換性を持つ拡張プロトコルである
2. MSX キーマトリクス＋ゲームパッドを通知する
3. 新プロトコル対応デバイスであることを識別できる

2.1. 既存プロトコルと互換性を持つ拡張プロトコルである

既に出回っている FaceII キーボード・FaceII ゲームパッドのファームウェアを更新するのは困難だと考えています。それらを更新せずとも、使い続けられるようにすることで新プロトコルの導入を容易にします。

新プロトコルに対応した新しいキーボード・ゲームパッドを接続した場合に、より良い体験を得られるようにするプロトコルです。



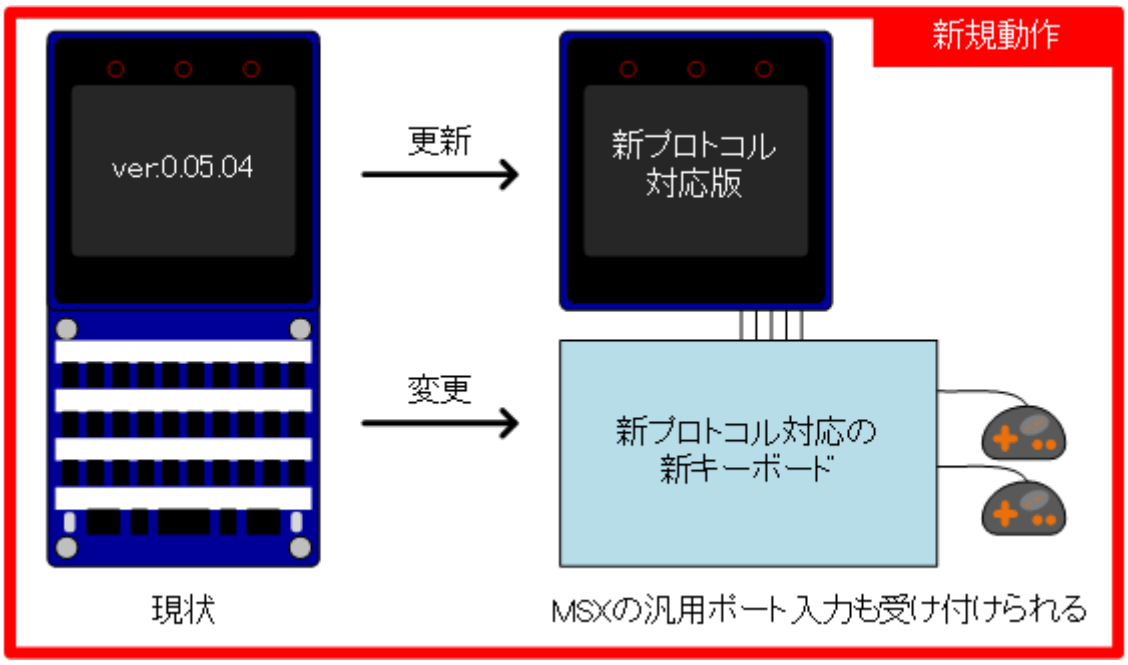
2.2. MSX キーマトリクス＋ゲームパッドの情報を通知する

従来の MSX は、PPI (i8255) に接続されたキーボードのマトリクス配列によって共通になっています。新たに作るキーボードが実際にそのようなマトリクスになっているかどうかは考慮に入れず、プロトコルとしては MSX キーマトリクスをそのまま採用することで下記のようなメリットを享受します。

1. キーボード側にキーさえ搭載すれば、MSX のどのキーでも入力可能になる
2. 海外向け MSX0 を作る際に MSX0 のキーボードエミュレーションが影響を受けずに海外 MSX のキー配列に対応出来る
3. キーボード側で PPI の返す値に加工しているため、MSX0 側のキーボードエミュレーションにかかる CPU 負荷が低減する
4. 複数のキーの組み合わせ入力にも従来 MSX と同等に対応出来る

さらに、入力デバイスはキーボードだけで無く、汎用ポートに繋げた 2 つのゲームパッドもあります。ゲームパッドへの入力のみサポートすることで、スイッチのみのゲームパッド・ジョイスティックなどの入力を、キーボードと同時に受けられるようになります。（汎用ポートからの出力信号はサポートしないので、マウスや JOYMEGA 等への対応は出来ません。）

汎用ポートにボタン等を繋げて MSX0 のコントロール用に使うことも考えると、ゲーム以外への応用も可能なため、IoT 用途でも有益な入力ポートとなります。



2.3. 新プロトコル対応デバイスであることを識別できる

接続しているキーボードは、FaceII キーボードと同じ I2C Address 08h を使います。

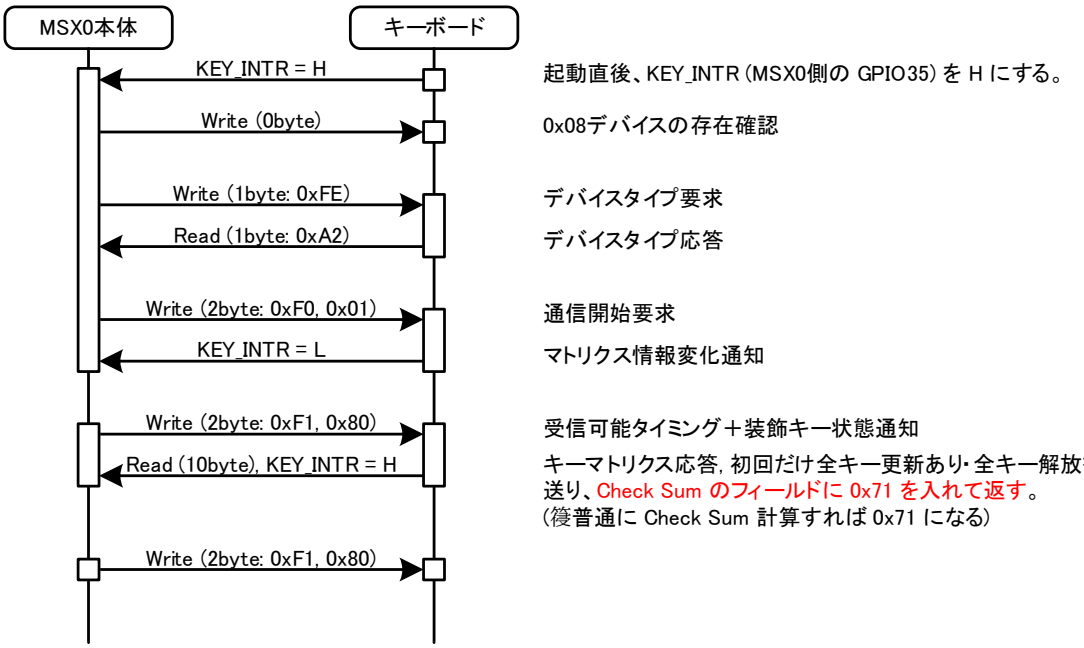
新プロトコルに対応している新キーボードデバイスなのか、あるいは FaceII キーボードや、FaceII キーボード互換キーボードなのか、を識別する手段を用意しています。これにより、新キーボードデバイスの時のにのみ新キーボードデバイス対応ドライバーへ切り替え、そうでなければ FaceII キーボードの既存ドライバーで動作することで、両方に対応出来ます。

3. 現状のプロトコル仕様

ここでは、私の理解が正しいか確認する目的で、現状のプロトコルの仕様について具体的に記載します。

3.1. 制御フロー

現状のキーボード通信は以下のフローで行われます。



デバイスタイプは下記が存在します。

表1 デバイスタイプ一覧

デバイスタイプ応答の番号	デバイスの種類	概要
0x00	Faces テンキー/Faces キーボード	キーマトリクス応答の代わりに、押されたキーコードが送られてくる 割り込み信号のピンアサインが異なり外部 I2C と競合する
0xFF	FaceII ゲームパッド	キーマトリクスが送られてくる
0xA2	FaceII キーボード	キーマトリクスが送られてくる

MSX0 クラファン版に付属しているゲームパッド、キーボードは、FaceII タイプであり、割り込み信号が外部 I2C と競合する問題が解消されている。敢えて古い Faces テンキーや Faces キーボードに対応する必要も無いため、本書では以降、Faces テンキーや Faces キーボードには言及しません。

このプトロコルの良いところとしては、最初にデバイスタイプをチェックするため、新しいデバイスタイプにしてしまえば、実際のキー状態のやりとりに関して、全く異なるフォーマットを採用できる点です。一方で、デバイスタイプを増やしすぎると、MSX0 側に「既に世の中に出回った対応デバイス」の全てのドライバーを内蔵させる必要が出てくるため、むやみに増やすべきでありません。

キーマトリクス応答の 10byte は、下表に示す内容です。+0 には 10byte を示す 0x0A。最後の +9 には +0～+9 を全て足した結果の下位 8bit が 0x00 になるようにするチェックサム値が入っています。

Keyboard → MSX0

	7	6	5	4	3	2	1	0
+0	0	0	0	0	1	0	1	0
+1	更新通知	0	0	0	0	0	Q	W
+2	E	R	T	Y	U	I	O	P
+3	更新通知	0	0	1	0	0	A	S
+4	D	F	G	H	J	K	L	BS
+5	更新通知	0	1	0	0	0	0	Z
+6	X	C	V	B	N	M	$	SPC
+7	更新通知	0	1	1	0	0	0	0
+8	0	0	0	Fn	Sym	Ent	ALT	aA
+9	CSUM							

更新通知と書かれている bit は、その更新通知が含まれる 1byte と、その次の 1byte の中のキー情報（赤文字）のうち、前回から変化があるビットが存在すれば 1 を立てます。変化が無ければ 0 にします。

受信可能タイミング＋装飾キー状態通知 0xF1 は、続く 1byte で装飾キー Fn, Sym の状態を、MSX0 からキーボードへ通知します。

MSX0 → Keyboard

	7	6	5	4	3	2	1	0
+0	1	1	1	1	0	0	0	1
+1	更新通知	0	右LED	左LED	0	0	0	0

左LED, 右LEDは、0で消灯、1で点灯

FaceII キーボードは、Fn や Sym が物理的に異なるキーになっているため、それらの押下状態の変化のタイミングと、通常のキーの押下状態の変化のタイミングは、同時に発生しないことを前提とした実装になっているようです。

1 回のマトリクス送信で、Fn/Sym が前回の送信時から変化しているようなマトリクスで、それらによって装飾される通常キーの押下状態を同時に送っても、正しい文字を入力できません。

FaceII キーボードをエミュレーションして、別の物理キーによって入力されるキーボード（例えば、従来 MSX ど同等の配列のキーを備えるキーボード）では、FaceII キーボードとは装飾キーのかかり方が異なるため、入力するキーに応じて、Fn/Sym の状態を作り出して付与するわけですが、同時に送れないので 2 回に分けて送信する必要が発生し、複雑な制御が必要になってしまう問題があります。

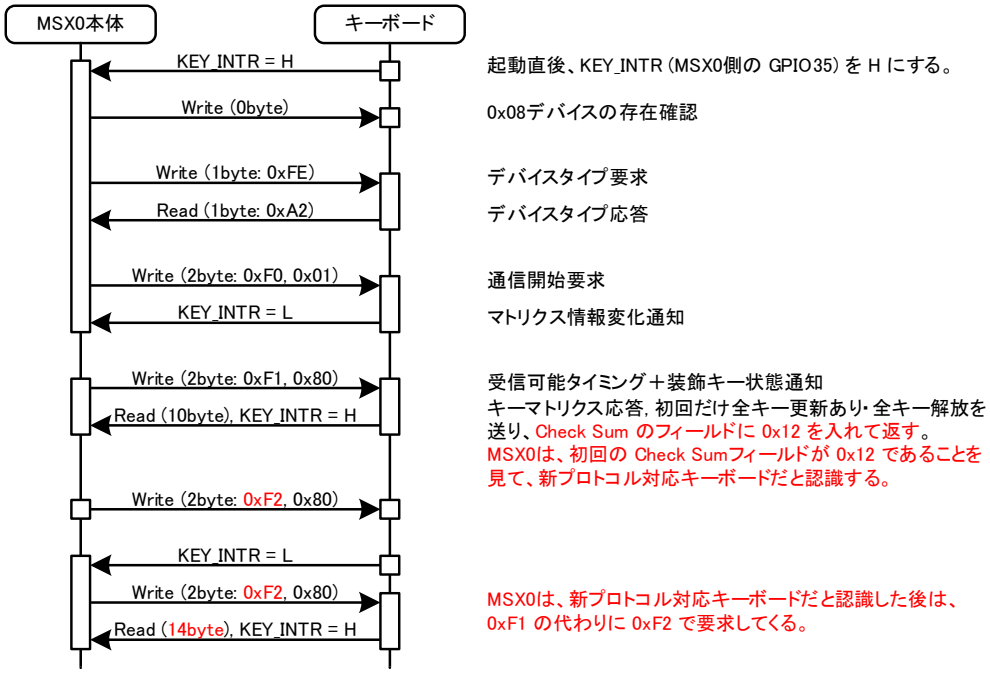
4. 新プロトコル

微妙な拡張を行って、また何か不足して微妙な拡張を繰り返していくよりは、しっかりと**「MSX0にとって都合の良い通信」になるようなプロトコルを規定して今後はよほどのことがない限り拡張しなくて済むようにする**方が得策だと考えます。

ということで、新しいデバイスとして新プロトコルを載せることを提案させていただきます。

4.1. 制御フロー

制御フローは、既存の FaceII キーボードとほぼ同じです。



異なるのは、キーマトリクスの内容と、装飾キーの扱いです。

キーボード側は、周期的に物理的なキー状態を調べ、MSX0 へ送信するキーマトリクス情報を生成します。前回 MSX0 へ送信したキーマトリクス情報を覚えておき、前回のキーマトリクスから変化があった場合に、KEY_INTR 信号（MSX0 の GPIO35）を L に落とします。MSX0 側の負荷が高くなりすぎないように、KEY_INTR を L に下げる間隔は、最低 4msec は空ける。

4.2. 受信可能タイミング＋装飾キー状態通知

MSX0 からキーボードへ送られる 2byte のコードであり、1byte 目は 0xF1、2byte 目は下記のビットマップになっている。

0xF1 コマンド 2byte 目のビットマップ

各ビットの意味は表 1 ビットの意味に纏めておきます。

表1 ビットの意味

ビットの名前	意味
Upd	更新フラグ。1 で更新あり。 更新が無い場合に 1 を指定してもよし。 0 を指定した場合、bit0-6 の内容は無視され、現状が維持される。
Kana	かな LED 表示制御。1 で点灯。
Caps	CapsLED 表示制御。1 で点灯。
Rsv	将来予約ビット。0 にする。

この 2 バイトのコマンドは、キーマトリクスを受信する準備が整ったタイミング通知も兼ねる。タイミング通知のみの場合は、Upd=0 にしておけば良い。

また、Kana/Caps は、あくまで LED 制御です。この値がキーマトリクスに影響を及ぼすことはない点が FaceII キーボードと大きな違いです。

Kana/Caps の LED は、MSX のプログラムから制御できる信号なので、MSX0 でも同様のことが出来るようになります。

4.3. キーマトリクス応答

14 バイトのキーマトリクス情報をキーボードから MSX0 へ送信する。

FaceII キーボードと異なるのは下記の点である。

1. キーマトリクスのビットマップとサイズ

2. 初回に送る内容を特別な値にする必要がない
3. キーマトリクスにはテンキーやゲームパッドのキーも含まれる
4. チェックサムは付けない

14 バイトのキーマトリクスの内容を以下に示す。

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
+1	7	6	5	4	3	2	1	0
+2	+	[	@	¥	^	-	9	8
+3	B	A	_	/	.	,	]	:
+4	J	I	H	G	F	E	D	C
+5	R	Q	P	O	N	M	L	K
+6	Z	Y	X	W	V	U	T	S
+7	F3	F2	F1	KANA	CAPS	GRPH	CTRL	SFT
+8	RET	SEL	BS	STOP	TAB	ESC	F5	F4
+9	→	↓	↑	←	DEL	INS	CLS	SPC
+10	4	3	2	1	0	Opt2	Opt1	Opt0
+11	,	.	-	9	8	7	6	5
+12	Rsv	Rsv	A7	A6	A4	A3	A2	A1
+13	Rsv	Rsv	B7	B6	B4	B3	B2	B1

先頭 +0 に関しては、14byte を示す 0x0E 固定。

+1～+11 までは MSX のキーマトリクスと同じです。押下で 0、解放で 1。

+12, +13 は、ゲームパッドの情報で +12 が PortA、+13 が PortB になります。bit0～5 が PSG#14 レジスタリード時の bit0～5 に対応する形です。こちらも押下で 0、解放で 1。

MSX0 の場合、物理的に離れた場所でのキーボード操作は、WiFi を使う仕組みを使った方が便利です。MSX0 に直接繋げたキーボードを使うのは、やはり MSX0 の近くで操作したい場合に限定して良いと思いますので、エラー検出やエラー訂正の仕組みに演算負荷を割くメリットは薄いという考えの基に、チェックサムは外しています。実際、FaceII キーボードのプロトコルに付いているチェックサムは使われていないようなので問題ないでしょう。1byte でも縮まる方がメリットが高いです。

Rsv のビットは、未使用であれば 1 を設定してください。0 で押下・1 で解放をしめしています。

Panasonic の MSX2+/turboR では、+12 のキーマトリクスを持っており、そこに実行・取消キーが充てられていますが、そのために 1byte 増えるのも勿体ないので、そこはカットしています。

マトリクスの 1 回の通信バイト数は増えていますが、装飾キー情報を別で送る必要はありません。8～16msec 程度に 1 回程度更新すれば充分です。

5. MSX0 への実装

(1) 08h デバイスの存在確認

サイズ 0 の書き込みを行う。これは現状の MSX0 でも行っているはず。

キーボードは電源投入直後は、キーボード割り込み信号(MSX0 側の GPIO35)を H にする。

(2) 08h デバイスに対して FEh を書き込む

デバイスタイプの要求コマンドを発行。

(3) 08h デバイスから 1byte の応答を受け取る

キーボードは、FaceII キーボード・新キーボード、いずれの場合も A2h を返す。

(4) 08h デバイスに対して F0h, 01h を書き込む

起動直後だという通知。キーボード側はこれを受けて初期状態にリセットし、

準備が出来たらキーボード割り込みを L にする。

(5) 08h デバイスに対して F1h, XXh を書き込む

二バイト目は Sym キー, Fn キー状態を示すフラグ指定。

キーボードはこれを受けて、FaceII キーボードフォーマットの 10byte を応答として返し、

キーボード割り込みを H へ戻す。

(6) 08h デバイスから 10byte の応答を受け取る

初期状態なので、全キー更新・全キー押していないを示す固定値。

FaceII キーボードの場合は、チェックサム 0x71

新キーボードの場合は、チェックサムフィールドに 0x12

が入っている。これを見てどちらのタイプか識別する。

(7) キーボードは、キー入力状態の変化を検知するとキーボード割り込みを L にする

(8) キーボード割り込みの状態を見て覚えておき、08h デバイスに対して FaceII キーボードと認識していれば F1h, XXh を、新キーボードと認識していれば F2h, XXh を書き込む

F1h, XXh の XXh は Sym, Fn キー情報。

F2h, XXh の XXh は かな, CAPS-LED 点灯情報。

(9) 覚えていたキーボード割り込みの状態が L であれば、続けてキーボードからのマトリクス情報を受け取る

F1h を投げた場合は 10byte。これは FaceII キーボードマトリクスなので、

現状の MSX0 内で行われている処理をそのまま実施して MSX のキーマトリクスに変換する。

F2h を投げた場合は 14byte。これは新キーボードマトリクスなので、

送られてきたマトリクスの +1～+11 を MSX のキーマトリクスとして利用する。

+12, +13 はゲームパッドのマトリクスとして利用する。

このようなドライバーを実装していただければ、既にユーザーが持っている FaceII キーボードはこれまで通り動作し、かつ新キーボードは MSX のキーを従来 MSX と同様に好きな組み合わせで同時に押したり話したりする情報を扱えるようになります。

必要なキーだけに絞ったキーボードを作って IoT の入力手段として使うこともできますし、フルキー搭載することもできます。そういった入り口を作るための規格決めです。

採用いただければ幸いです。