JR−100

マカオさんの協力感謝します。

サウンド、テープへ

JR-100のスペック

MPU	MB8861N 89488Hz
SOUND	VIAを使用した方形波　１声
ROM	8K(拡張可)
RAM	16KB(拡張可)

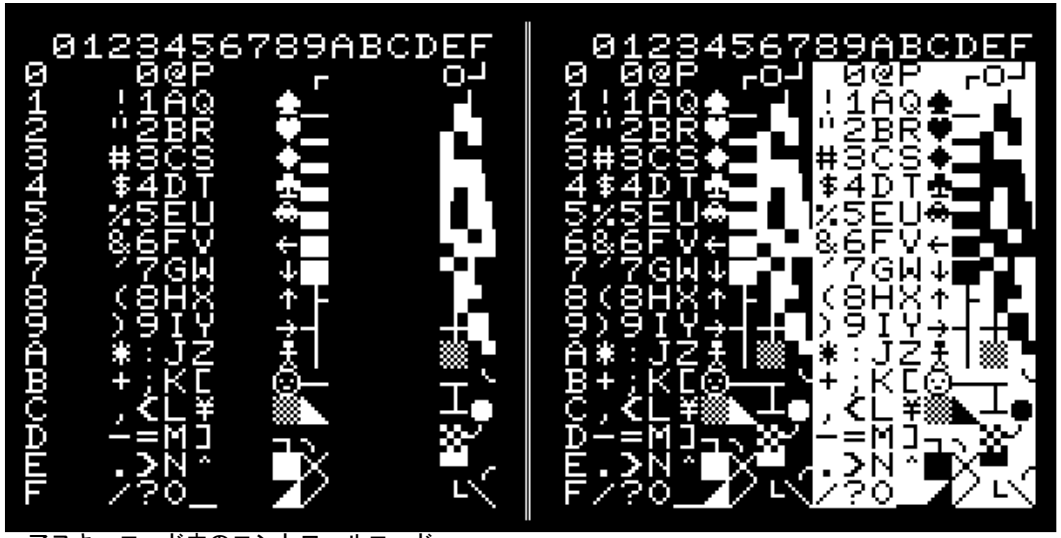
ＭＰＵはＭＢ８８６１Ｎを使用
周辺デバイスはＶＩＡ　６５２２が画面以外を受け持つようになり非常にコストパフォーマンスが高い
低価格機種にありがちなすぐにメモリ不足に悩まされる事もない
不満点としてリアルタイマ割り込みが欲しかったのとリセットスイッチが無い事ぐらい
ＢＡＳＩＣのコマンドにＲＥＳＥＴ命令があるがあまり意味が無い

拡張ROM
少なくともプリンタの制御ＲＯＭは外部から載せられる様に設計されており
＄Ｄ０００が＄００だったらプリンタ制御ＲＯＭがあると判断して
＄Ｄ００１はイニシャライズ
＄Ｄ００４は１文字出力
＄Ｄ００７はＨＣＯＰＹのフックとなっているようです

拡張RAM
16Ｋバイト拡張可能で外部バスを通して拡張します。
またＢＡＳＩＣで本体内のＲＡＭと連続してチェックしている事から
アドレスは＄４０００ー＄７ＦＦＦになります（拡張端子からＤ－ＲＡＭを接続可）

割り込み
本体内では使用していません外部端子にでています
またＲＡＭ上に飛ぶのでフックで拡張ハード用の設定が出来るようになっています
本体のみのＢＡＳＩＣの設定では全てＢＡＳＩＣのコマンド待ちへ飛ぶようになっています
ＭＢ８８６１Ｎのロットによっては３８ピンの動作が違い、割り込み関係が変わるのですが
３８ピンはオープン、本体では割り込みを使用しないので気にする事はありません
　＄００Ｆ７：ＩＲＱ（＄Ｅ５Ａ３）
　＄００ＦＡ：ＳＷＩ（＄Ｅ５Ａ３）
　＄００Ｄ７：ＮＭＩ（＄Ｅ５Ａ３）

PCG、VRAM
アスキーコードとＶＲＡＭコード



アスキーコード内のコントロールコード

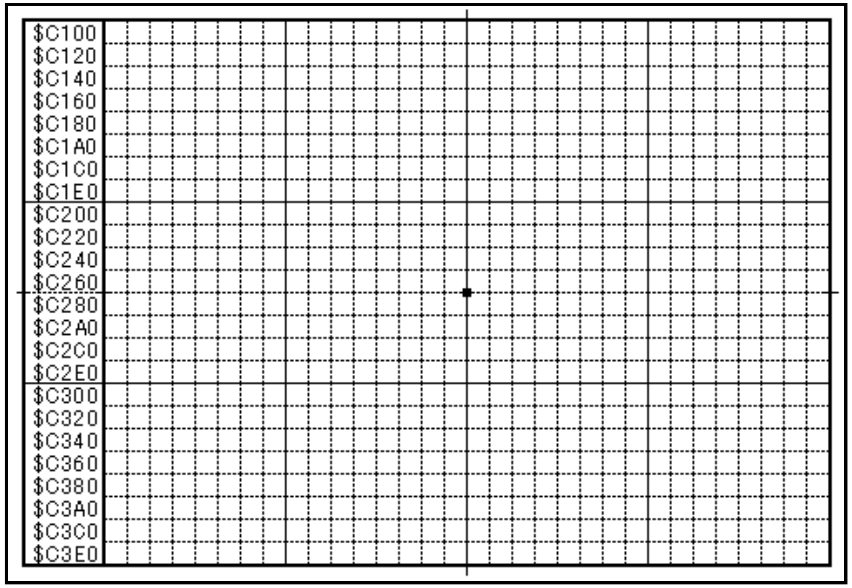
＄０３	ＢＲＥＡＫ	＄１０	ＤＥＬ
＄０８	ＲＵＢＯＵＴ	＄１１	カーソル下
＄０９	ＩＮＳ	＄１２	カーソル上
＄０Ｂ	ＧＲＡＰＨ	＄１３	カーソル右
＄０Ｃ	ＨＯＭＥ	＄１４	カーソル左
＄０Ｄ	ＲＥＴＵＲＮ	＄１８	ＣＡＮＳＥＬ
		＄１９	Ｌ．ＩＮＳ
		＄１Ａ	ＨＣＯＰＹ

アスキーコード＄８０（ＶＲＡＭコード＄４０）は画面クリアの文字
アスキーコード＄ＥＡ（ＶＲＡＭコード＄６Ａ）はカーソルの文字

＄Ｃ０００－＄Ｃ３ＦＦまで＄Ｃ０００－＄Ｃ０ＦＦがＰＣＧ定義領域
サイクルスチールによりＭＰＵを止める事なく定義、表示が可能のようです

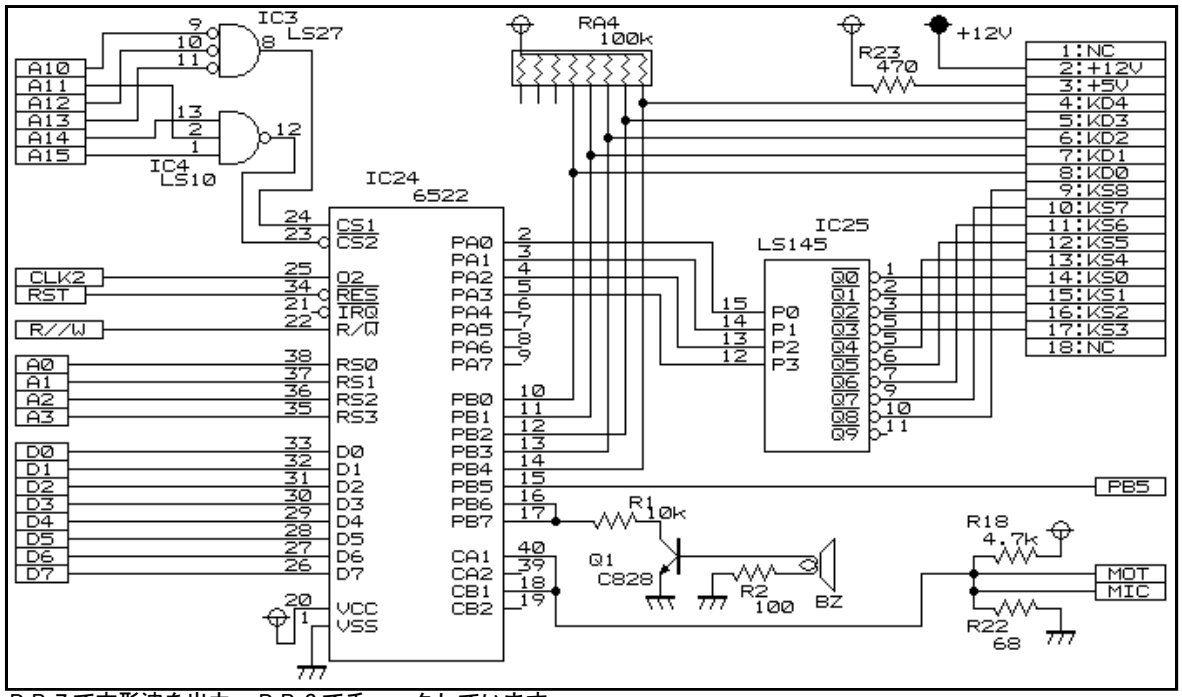
＄Ｃ０００－＄Ｃ０ＦＦまでが実質定義エリアで＄Ｃ１００－＄Ｃ３ＦＦが表示エリア（ＶＲＡＭ）
１２８キャラクタ中＄００－＄１Ｆの３２キャラクタを定義出来るようになっています
３３キャラクタデータ以降は表示エリア（ＶＲＡＭ）と重なる形になっています
つまり＄２０以降のＰＣＧ定義データが画面にキャラクタコードとして表示される事になります

ＶＲＡＭデータＢｉｔ７でハードで反転文字やＰＣＧにするようになっています（ＣＭＯＤＥで切り換わる）



テキストは３２×２４

6522 VIA
パラレルポートではキーボード、サウンド、カセット、ＣＧＲＯＭの切り換えを行います
タイマはサウンド、カセットで使用します
Ｉ／Ｏは＄Ｃ８００－＄Ｃ８０Ｆまでになります



ＰＢ７で方形波を出力、ＰＢ６でチェックしています

＄Ｃ８００:ＰＢ出力　ＯＲＢ
　ｄ７：　－
　ｄ６：　－
　ｄ５：ＣＭＯＤＥ
　ｄ４：　－
　ｄ３：　－
　ｄ２：　－
　ｄ１：　－
　ｄ０：　－

＄Ｃ８００:ＰＢ入力　ＩＲＢ
　ｄ７：ＢＥＥＰ
　ｄ６：ＢＥＥＰ
　ｄ５：　－
　ｄ４：キーデータ　Ｄ４
　ｄ３：キーデータ　Ｄ３
　ｄ２：キーデータ　Ｄ２
　ｄ１：キーデータ　Ｄ１
　ｄ０：キーデータ　Ｄ０

＄Ｃ８０１:ＰＡ出力　ＯＲＡ
　ｄ７：　－
　ｄ６：　－
　ｄ５：　－
　ｄ４：　－
　ｄ３：キーセレクト　Ｄ３
　ｄ２：キーセレクト　Ｄ２
　ｄ１：キーセレクト　Ｄ１
　ｄ０：キーセレクト　Ｄ０

＄Ｃ８０１：ＰＡ入力　ＩＲＡ
ｄ７：未使用
ｄ６：未使用
ｄ５：未使用
ｄ４：未使用
ｄ３：　－
ｄ２：　－
ｄ１：　－
ｄ０：　－

＄Ｃ８０２：ＰＢデータ方向　ＤＤＲＢ　１＝出力、０＝入力
ｄ７：０　ＰＢ７　入力
ｄ６：０　ＰＢ６　入力
ｄ５：１　ＰＢ５　出力
ｄ４：０　ＰＢ４　入力
ｄ３：０　ＰＢ３　入力
ｄ２：０　ＰＢ２　入力
ｄ１：０　ＰＢ１　入力
ｄ０：０　ＰＢ０　入力

＄Ｃ８０３：ＰＡデータ方向　ＤＤＲＡ　１＝出力、０＝入力
ｄ７：０　ＰＡ７　入力
ｄ６：０　ＰＡ６　入力
ｄ５：０　ＰＡ５　入力
ｄ４：１　ＰＡ４　出力
ｄ３：１　ＰＡ３　出力
ｄ２：１　ＰＡ２　出力
ｄ１：１　ＰＡ１　出力
ｄ０：１　ＰＡ０　出力

＄Ｃ８０４：Ｔ１Ｌ－Ｌ（出力）　ラッチ下位書き込み
　　Ｔ１Ｃ－Ｌ（入力）　カウンタ下位読み込み、Ｔ１割り込みフラグ　リセット

＄Ｃ８０５：Ｔ１Ｃ－Ｈ（出力）　ラッチ上位、カウンタ上位書き込み、ラッチカウンタ下位へ転送
　　Ｔ１割り込みフラグ　リセット
　　（入力）　上位カウンタ読み込み

＄Ｃ８０６：Ｔ１Ｌ－Ｌ（出力）　ラッチ下位書き込み
　　（入力）　ラッチ下位読み込み

＄Ｃ８０７：Ｔ１Ｌ－Ｈ（出力）　カウンタ上位書き込み、Ｔ１割り込みフラグ　リセット
　　（入力）　ラッチ上位読み込み

＄Ｃ８０８：Ｔ２Ｌ－Ｌ（出力）
　　Ｔ２Ｃ－Ｌ（入力）

＄Ｃ８０９：Ｔ２Ｃ－Ｈ（出力）
　　（入力）

＄Ｃ８０Ａ：ＳＲ（シフト・レジスタ）
　　ＣＭＴ　’０’を書き込む場合＄６６
　　’１’を書き込む場合＄ＡＡ

＄Ｃ８０Ｂ：ＡＣＲ（補助コントロールレジスタ）
ｄ７：Ｔ１コントロール
ｄ６：Ｔ１コントロール
ｄ５：Ｔ２コントロール
ｄ４：シフトレジスタコントロール
ｄ３：シフトレジスタコントロール
ｄ２：シフトレジスタコントロール
ｄ１：ＰＢラッチイネーブル
ｄ０：ＰＡラッチイネーブル
　　Ｔ１コントロール

```
　　１１＝フリーランニング連続割り込み発生、ＰＢ７＝パルス出力
　　１０＝ワンショットモード　　、ＰＢ７＝パルス出力
　　０１＝連続割り込み発生　　、ＰＢ７＝ディスエーブル
＊　００＝ワンショットモード　　、ＰＢ７＝ディスエーブル

Ｔ２コントロール
　　１＝ＰＢ６の入力パルスを設定した数だけカウント
＊　　０＝ワンショットモードのインターバルタイマ

シフトレジスタコントロール
　１１１＝外部クロックによる　　シフト出力
　１１０＝システムクロックによる　シフト出力
　１０１＝タイマ２による　　シフト出力
　１００＝タイマ２によるフリーランニング出力
　０１１＝外部クロックによる　　シフト入力
　０１０＝システムクロックによる　シフト入力
　００１＝タイマ２による　　シフト入力
＊０００＝シフトレジスタ　ディスエーブル入力

ＰＢラッチイネーブル
　　１＝ＣＢ１により入力または出力をラッチ
　　０＝出力データがラッチ
ＰＡラッチイネーブル
　　１＝ＣＡ１により入力または出力をラッチ
　　０＝出力データがラッチ

＄Ｃ８０Ｃ:ＰＣＲ（ペリフェラルコントロールレジスタ）
　ｄ７：ＣＢ２コントロール
　ｄ６：ＣＢ２コントロール
　ｄ５：ＣＢ２コントロール
　ｄ４：ＣＢ１コントロール
　ｄ３：ＣＡ２コントロール
　ｄ２：ＣＡ２コントロール
　ｄ１：ＣＡ２コントロール
　ｄ０：ＣＡ１コントロール
　　ＣＢ２コントロール
　　＊１１１＝マニュアル出力モード　　ＣＢ２をＨにする
　　　１１０＝マニュアル出力モード　　ＣＢ２をＬにする
　　　１０１＝パルス出力モード　　ＯＲＢ書き込み後の１サイクルだけＣＢ２がＬになる
　　　１００＝ハンドシェイク出力モード　ＣＢ２はＯＲＢ書き込みでＬ
　　　　　　　　　　ＣＢ１入力パルスのアクティブ・トランジシマンでＨ
　　　０１１＝単独割込み入力モード　　ＩＦＲ３はＣＢ２入力パルスの立ち上がりでセット
　　　　　　　　　　ＯＲＢの読み書きでクリアされない
　　　０１０＝割込み入力モード　　ＩＦＲ３はＣＢ２入力パルスの立ち上がりでセット
　　　　　　　　　　ＯＲＢの読み書きでクリア
　　　００１＝単独割込み入力モード　　ＩＦＲ３はＣＢ２入力パルスの立ち下がりでセット
　　　　　　　　　　ＯＲＢの読み書きでクリアされない
　　　０００＝割込み入力モード　　ＩＦＲ３はＣＢ２入力パルスの立ち下がりでセット
　　　　　　　　　　ＯＲＢの読み書きでクリア

　　ＣＢ１コントロール
　　＊　　１＝ＣＢ１の割込み入力のアクティブトランジョン　立ち上がりセット
　　　　０＝ＣＢ１の割込み入力のアクティブトランジョン　立ち下りでセット

　　ＣＡ２コントロール
　　＊１１１＝マニュアル出力モード　　ＣＡ２をＨにする
　　　１１０＝マニュアル出力モード　　ＣＡ２をＬにする
　　　１０１＝パルス出力モード　　ＯＲＡ書き込み後の１サイクルだけＣＡ２がＬになる
　　　１００＝ハンドシェイク出力モード　ＣＡ２はＯＲＡ書き込みでＬ
　　　　　　　　　　ＣＡ１入力パルスのアクティブ・トランジシマンでＨ
　　　０１１＝独立割込み入力モード　　ＩＦＲ０はＣＡ２入力パルスの立ち上がりでセット
　　　　　　　　　　ＯＲＡの読み書きでクリアされない
　　　０１０＝入カモード　　ＩＦＲ０はＣＡ２入力パルスの立ち上がりでセット
```

ＯＲＡの読み書きでクリア
００１＝独立割込み入力モード　ＩＦＲ０はＣＡ２入力パルスの立ち下がりでセット
ＯＲＡの読み書きでクリアされない
０００＝入力モード　ＩＦＲ０はＣＡ２入力パルスの立ち下がりでセット
ＯＲＡの読み書きでクリア

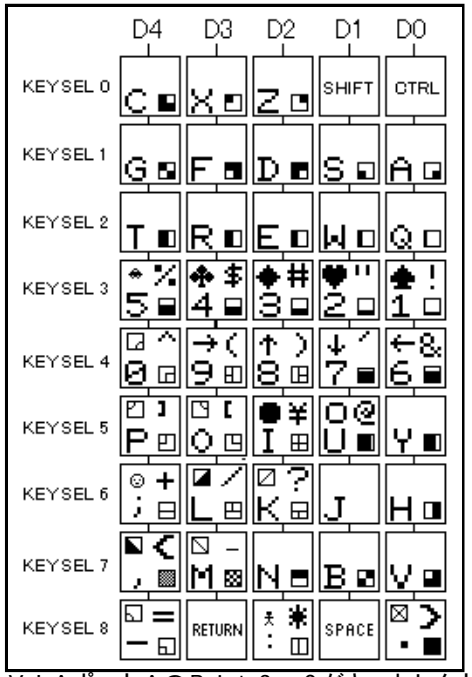
ＣＡ１コントロール
１＝ＣＡ１の割込み入力のアクティブトランジョン　立ち上がりセット
＊　０＝ＣＡ１の割込み入力のアクティブトランジョン　立ち下りでセット

＄Ｃ８０Ｄ:ＩＦＲ
ｄ７：１＝割込み有り　、割込み無し
ｄ６：１＝Ｔ１のタイムアウト　、０＝Ｔ１下位カウンタまたは上位カウンタの書き込みを行った
ｄ５：１＝Ｔ２のタイムアウト　、０＝Ｔ２下位カウンタまたは上位カウンタの書き込みを行った
ｄ４：１＝ＣＢ１のアクティブトランジョン、０＝ＯＲＢの読み書きを行った
ｄ３：１＝ＣＢ２のアクティブトランジョン、０＝ＯＲＢの読み書きを行った
ｄ２：１＝シフト８回終了　、０＝ＳＲ　の読み書きを行った
ｄ１：１＝ＣＡ１のアクティブトランジョン、０＝ＯＲＡの読み書きを行った
ｄ０：１＝ＣＡ２のアクティブトランジョン、０＝ＯＲＡの読み書きを行った

＄Ｃ８０Ｅ:ＩＥＲ
ｄ７：セットモード　１＝セット　、０＝リセット　（入力では常に０）
ｄ６：Ｔ１割込み　１＝指定する、０＝指定しない
ｄ５：Ｔ２割込み　１＝指定する、０＝指定しない
ｄ４：ＣＢ１割込み　１＝指定する、０＝指定しない
ｄ３：ＣＢ２割込み　１＝指定する、０＝指定しない
ｄ２：ＳＲ割込み　１＝指定する、０＝指定しない
ｄ１：ＣＡ１割込み　１＝指定する、０＝指定しない
ｄ０：ＣＡ２割込み　１＝指定する、０＝指定しない

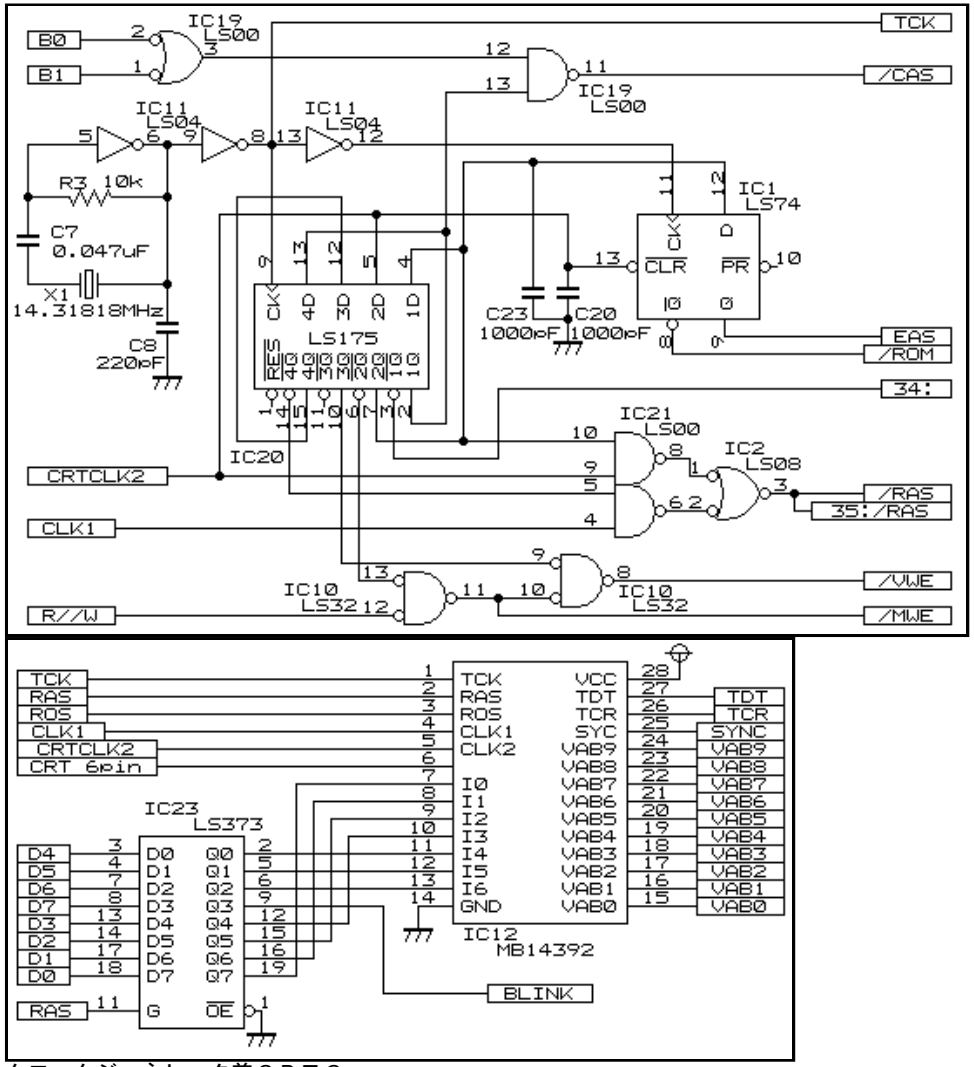
＄Ｃ８０Ｆ:ＯＲＡ　ハンドシェイクに影響されない

キーボード

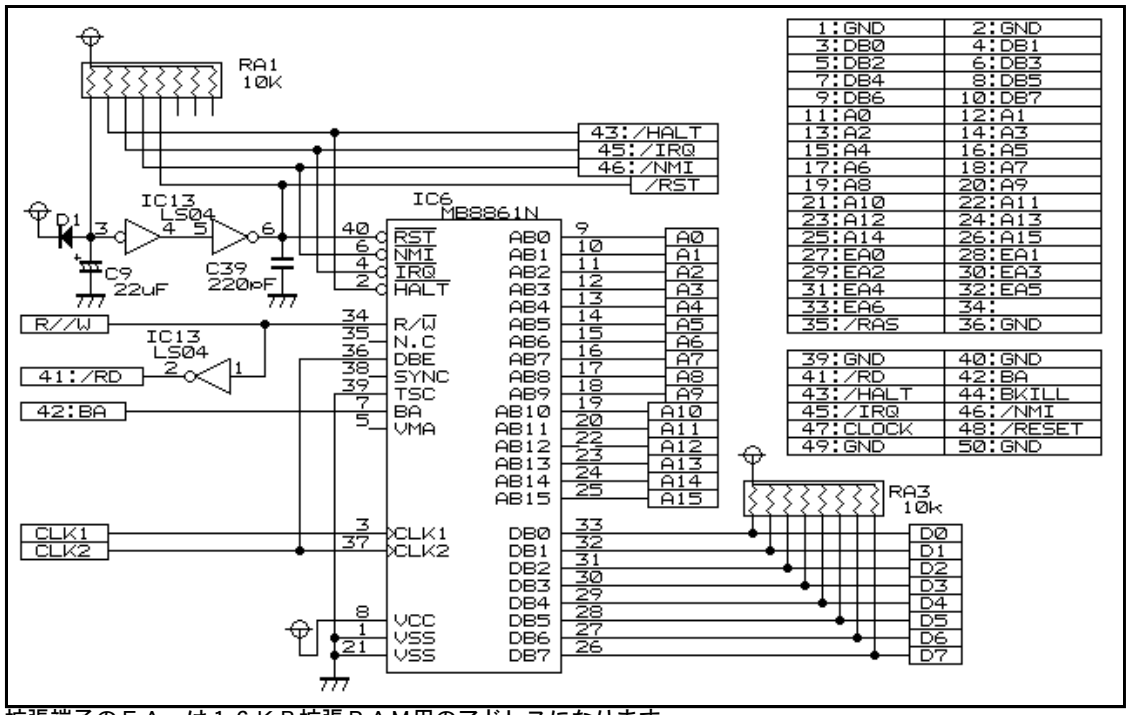


ＶＩＡポートＡのＢｉｔ３－０がキーセレクト出力

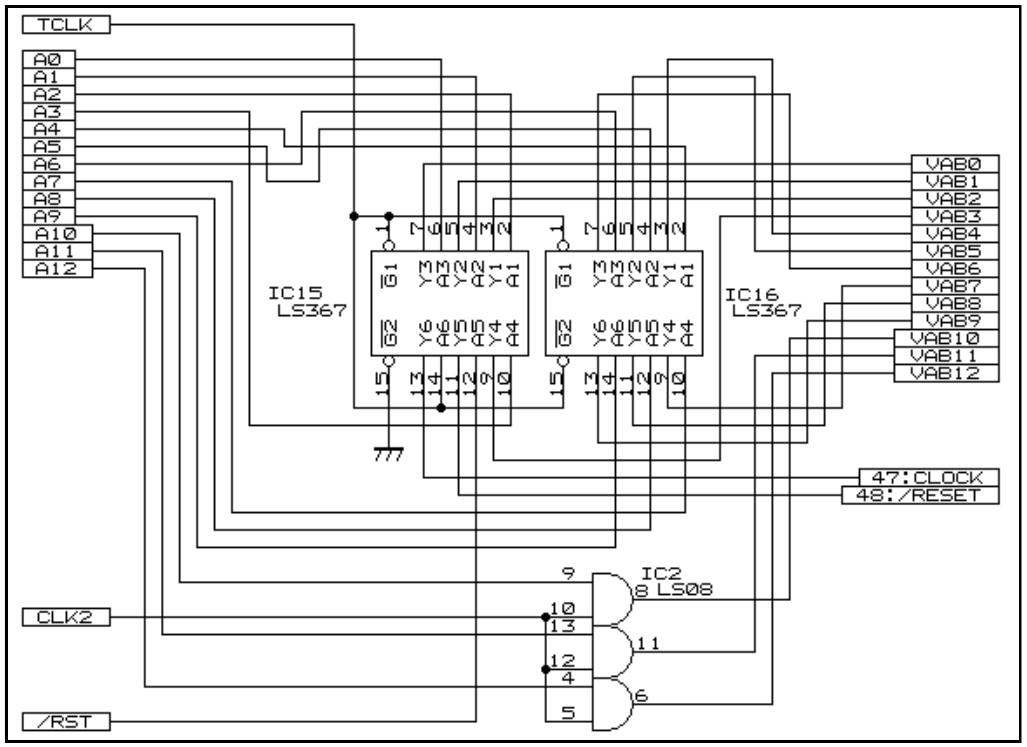
ＶＩＡポートＢのＢｉｔ４－０がキーデータ入力となります
＄Ｃ８０１にキーセレクト０－８を出力して
＄Ｃ８００のＢｉｔ４－０を読めばキーデータが読めます
　１＝押されていない
　０＝押されている

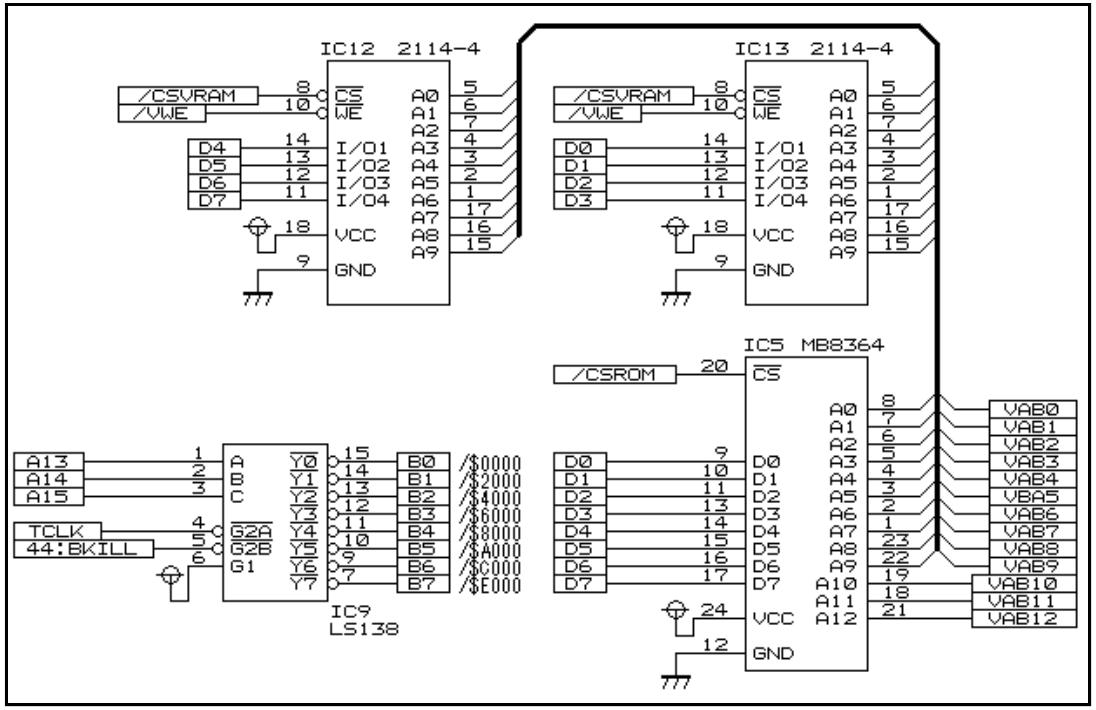


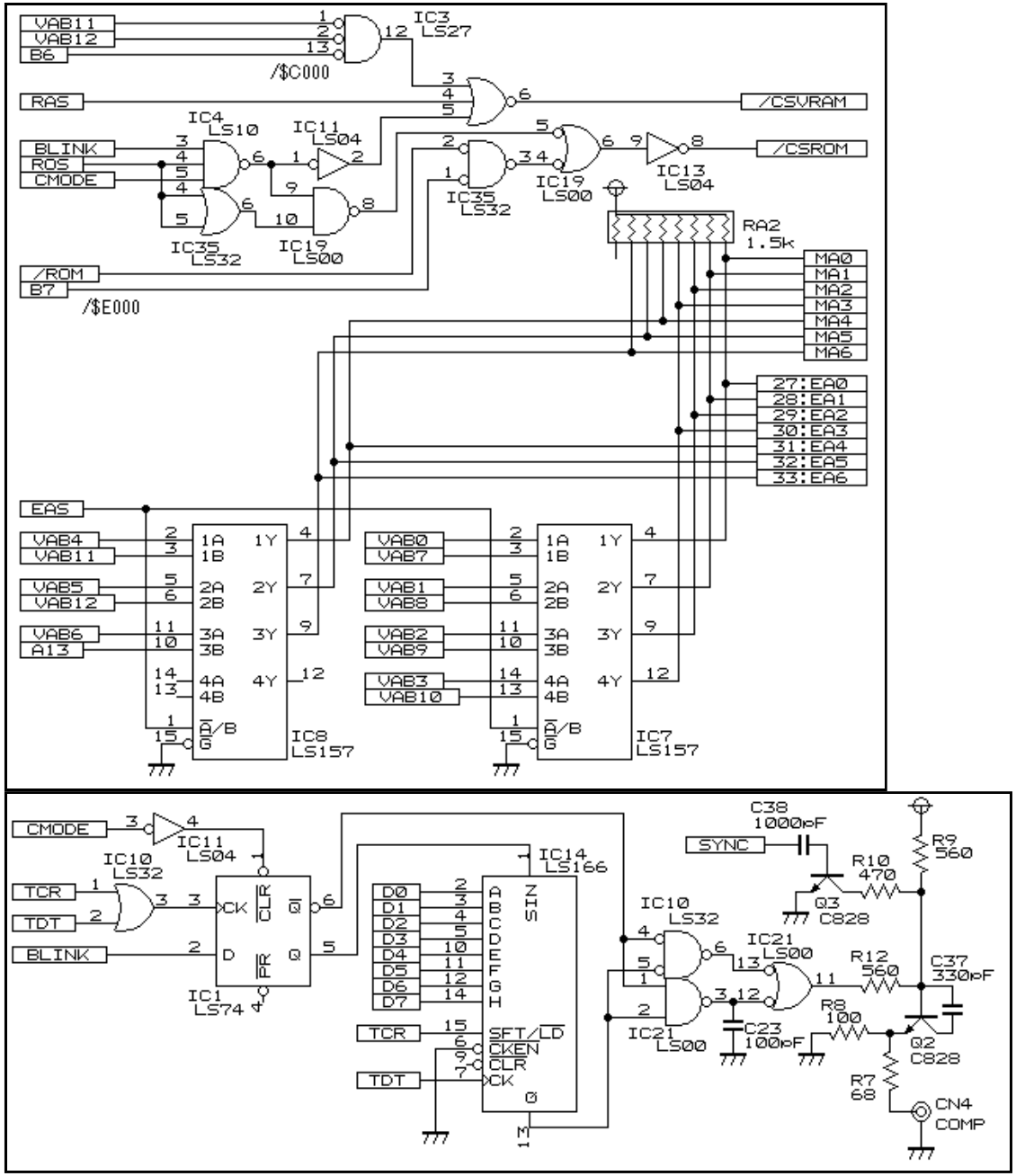
クロックジェネレータ兼ＣＲＴＣ



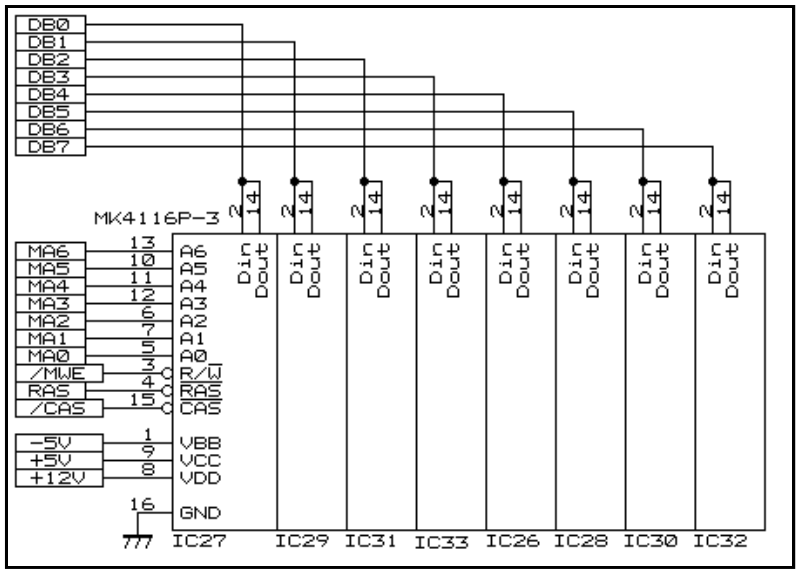
拡張端子のＥＡｎは１６ＫＢ拡張ＲＡＭ用のアドレスになります


TCLK
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
IC15
LS367
IC16
LS367
VAB0
VAB1
VAB2
VAB3
VAB4
VAB5
VAB6
VAB7
VAB8
VAB9
VAB10
VAB11
VAB12
47:CLOCK
48:/RESET
IC2
8 LS08
CLK2
/RST


IC12 2114-4
IC13 2114-4
IC5 MB8364
IC9
LS138
/CSVRAM
/VWE
/CSROM
D4
D5
D6
D7
D0
D1
D2
D3
A13
A14
A15
TCLK
44:BKILL
VCC
GND
CS
WE
I/O1
I/O2
I/O3
I/O4
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A
B
C
G2A
G2B
G1
Y0
Y1
Y2
Y3
Y4
Y5
Y6
Y7
B0
B1
B2
B3
B4
B5
B6
B7
/$0000
/$2000
/$4000
/$6000
/$8000
/$A000
/$C000
/$E000
VAB0
VAB1
VAB2
VAB3
VAB4
VBA5
VAB6
VAB7
VAB8
VAB9
VAB10
VAB11
VAB12


VAB11
VAB12
B6
IC3
LS27
/$C000
RAS
/CSVRAM
IC4
LS10
IC11
LS04
BLINK
RDS
CMODE
IC35
LS32
IC19
LS00
IC13
LS04
/CSROM
/ROM
B7
/$E000
RA2
1.5k
MA0
MA1
MA2
MA3
MA4
MA5
MA6
27:EA0
28:EA1
29:EA2
30:EA3
31:EA4
32:EA5
33:EA6
EAS
VAB4
VAB11
VAB5
VAB12
VAB6
A13
VAB0
VAB7
VAB1
VAB8
VAB2
VAB9
VAB3
VAB10
IC8
LS157
IC7
LS157

CMODE
IC11
LS04
IC10
LS32
TCR
TDT
BLINK
IC1
LS74
IC14
LS166
D0
D1
D2
D3
D4
D5
D6
D7
SFT/LD
CKEN
CLR
CK
SIN
TCR
TDT
C38
1000pF
SYNC
R9
560
R10
470
Q3
C828
IC21
LS00
R12
560
C37
330pF
R8
100
C23
100pF
Q2
C828
R7
68
CN4
COMP



メモリマップ

解析ＲＯＭはＶ１．０　ＣＲＣ：９５１ｄ０８ａ１
＄０００００－＄３ＦＦＦ　本体内ＲＡＭ　１６Ｋバイト
＄４０００－＄７ＦＦＦ　拡張ＲＡＭ　　１６Ｋバイト
＄８０００－＄ＢＦＦＦ　拡張ＲＯＭ　　１６Ｋバイト
＄Ｃ０００－＄ＣＦＦＦ　Ｉ／Ｏ領域
　　＄Ｃ０００－＄Ｃ０ＦＦ　ＰＣＧ定義用エリア
　　＄Ｃ１００－＄Ｃ３ＦＦ　ＰＣＧ定義用エリア　兼ＶＲＡＭ
　　＄Ｃ４００－＄Ｃ７ＦＦ　不明　（空き？）
　　＄Ｃ８００－＄ＣＢＦＦ　本体内　Ｉ／Ｏポート
　　＄ＣＣ００－＄ＣＦＦＦ　拡張用　Ｉ／Ｏポート
＄Ｄ０００－＄Ｄ７ＦＦ　拡張ＲＯＭ　　２Ｋバイト（プリンタ用）
＄Ｄ８００－＄ＤＦＦＦ　拡張ＲＯＭ　　２Ｋバイト

＄Ｅ０００－＄ＦＦＦＦ　本体内ＲＯＭ　　８Ｋバイト

＄００００　　：キークリック　＄００＝オフ、＄０１＝オン
＄００００１　　：ＢＥＥＰ　Ｔ１Ｌの値（Ｔ１Ｈは＄００で固定）　デフォルト値＄ＡＡ
＄０００２－＄０００３：乱数
＄０００４－＄０００５：プログラム　スタートアドレス　＄０２４６
＄０００６－＄０００７：プログラム　エンドアドレス
＄０００８－＄０００９：配列エンドアドレス
＄０００Ａ－＄０００Ｂ：２文字変数アドレス
＄０００Ｃ－＄０００Ｄ：文字列変数アドレス
＄０００Ｅ－＄０００Ｆ；プログラム　サブ用スタック　ポインタ
＄００１０－＄００１１：ＦＯＲ、ＮＥＸＴ用スタック　ポインタ
＄００１２－＄００１３：ＲＡＭ領域エンドアドレス
＄００１４　　：ＢＡＳＩＣ　フラグ　（１でオン）
　　ｄ７：ＰＩＣＫ
　　ｄ６：ＬＰＲＩＮＴ
　　ｄ５：文字出力
　　ｄ４：ＧＲＡＰＨ
　　ｄ３：ＡＵＴＯ
　　ｄ２：ＦＬＤ

```
                    ｄ１：　－
                    ｄ０：ＩＮＰＵＴ文
＄００１５　　　：空き
＄００１６－＄００１７：カーソル　ポインタ（ＶＲＡＭアドレス）
＄００１８－＄００１９：画面ポインタ（スクリーンエディット）
＄００１Ａ－＄００１Ｂ：ＢＡＳＩＣコマンド　ポインタ　　＄Ｆ８９８
＄００１Ｃ－＄００１Ｄ：ＢＡＳＩＣステートメント　ポインタ　＄Ｆ８Ｃ９
＄００１Ｅ－＄００１Ｆ：16進数変換、Ｘレジスタ退避用
＄００２０－＄００２１：行番号
＄００２２－＄００２３：プログラム　ポインタ、キーバッファ　ポインタ
＄００２４－＄００２５：スタック　ポインタ退避
＄００２６－＄００２７：行番号ポインタ
＄００２８－＄００２９：キーバッファ　エンドアドレス
＄００２Ａ　　　：カーソルに重なる文字
＄００２Ｂ　　　：ＯＶＦフラグ
＄００２Ｃ　　　：文字数カウンタ
＄００２Ｄ　　　：符号フラグ
＄００２Ｅ－＄００２Ｆ：ＤＡＴＡ文　ポインタ
＄００３０－＄００７５：キーバッファ　（ＣＭＴルーチンと共用）
＄００７６－＄００７Ｆ：演算用　スタック
＄００７７　　　：ＣＭＴ用　フラグ　＄００＝ＬＯＡＤ、＄０１＝ＶＥＲＩＦＹ
＄００８０－＄００８１：演算用　スタックポインタ
＄００８２－＄００８３：１文字変数Ａ
＄００８４－＄００８５：１文字変数Ｂ
＄００８６－＄００８７：１文字変数Ｃ
＄００８８－＄００８９：１文字変数Ｄ
＄００８Ａ－＄００８Ｂ：１文字変数Ｅ
＄００８Ｃ－＄００８Ｄ：１文字変数Ｆ
＄００８Ｅ－＄００８Ｆ：１文字変数Ｇ
＄００９０－＄００９１：１文字変数Ｈ
＄００９２－＄００９３：１文字変数Ｉ
＄００９４－＄００９５：１文字変数Ｊ
＄００９６－＄００９７：１文字変数Ｋ
＄００９８－＄００９９：１文字変数Ｌ
＄００９Ａ－＄００９Ｂ：１文字変数Ｍ
＄００９Ｃ－＄００９Ｄ：１文字変数Ｎ
＄００９Ｅ－＄００９Ｆ：１文字変数Ｏ
＄００Ａ０－＄００Ａ１：１文字変数Ｐ
＄００Ａ２－＄００Ａ３：１文字変数Ｑ
＄００Ａ４－＄００Ａ５：１文字変数Ｒ
＄００Ａ６－＄００Ａ７：１文字変数Ｓ
＄００Ａ８－＄００Ａ９：１文字変数Ｔ
＄００ＡＡ－＄００ＡＢ：１文字変数Ｕ
＄００ＡＣ－＄００ＡＤ：１文字変数Ｖ
＄００ＡＥ－＄００ＡＦ：１文字変数Ｗ
＄００Ｂ０－＄００Ｂ１：１文字変数Ｘ
＄００Ｂ２－＄００Ｂ３：１文字変数Ｙ
＄００Ｂ４－＄００Ｂ５：１文字変数Ｚ
＄００Ｂ６－＄００Ｄ３：サブルーチン用　スタック
＄００Ｄ４－＄００Ｅ２：演算用ワークエリア
＄００Ｅ３－＄００Ｆ６：空き
＄００Ｆ７－＄００Ｆ９：ＩＲＱフック　＄Ｅ５Ａ３（ＢＡＳＩＣコマンド待ち）
＄００ＦＡ－＄００ＦＣ：ＳＷＩフック　＄Ｅ５Ａ３
＄００ＦＤ－＄００ＦＥ：ＮＭＩフック　＄Ｅ５Ａ３
＄０１００－＄０１０７：空き
＄０１０８－　　：ＦＯＲ、ＮＥＸＴ用　スタック
　　－＄０２４４：ＭＰＵスタック
＄０２４５　　　：＄００
＄０２４６－　　：ＢＡＳＩＣプログラム


＄Ｅ０００－＄Ｅ３ＦＦ：キャラクタデータ
```

```
$Ｅ４００          ：ＲＥＳＥＴ
$Ｅ４１Ｅ          ：ＮＥＷ
$Ｅ４２６          ：ＣＬＥＡＲ
$Ｅ４３５          ：データ　クリア
                入力：Ｘ＝クリア（＄００を書き込む）するデータアドレス
                      Ｂ＝長さ
$Ｅ５０４：ＨＣＯＰＹ
$Ｅ５２６：ＡＵＴＯ
$Ｅ５４５：ＳＴＯＰ
$Ｅ５Ａ３：ＥＮＤ、ＢＡＳＩＣ　コマンド待ち
$Ｅ６５２：ＲＥＭ
$Ｅ６９８：フリーエリアの計算
     出力：Ｂ＝フリーエリアバイト数　上位
           Ａ＝フリーエリアバイト数　下位

$Ｅ６７Ｂ：ＩＦ
$Ｅ６ＡＢ：ＲＵＮ
$Ｅ７５６：ＬＰＲＩＮＴ
$Ｅ７５８：ＰＲＩＮＴ
$Ｅ７８Ｆ：ＴＡＢ（
$Ｅ９０１：ＩＮＩＴＰ

$Ｅ９１６：’：’のチェック
     入力：Ｘ＝検索する　アドレス
     出力：Ｘ＝見つかったアドレス
           Ａ＝アスキーデータ
           Ｃｆ＝１　アスキーデータ＄００以外（＄２０を除く）
                 ０　　　　＄００
           Ｚｆ＝１　アスキーデータ＄３Ａ
                 ０　　　　＄３Ａ以外
           ＄ＥＥ１Ｂをコールして取得したアスキーデータが’：’（＄３Ａ）のチェックを行います

$Ｅ９５Ｃ：ＡＢＳ（
$ＥＢ０Ｃ：ＭＯＤ（
$ＥＢ２Ｃ：ＲＮＤ（
$ＥＢ４１：ＵＳＲ（
$ＥＣ９Ａ：ＤＩＭ

$ＥＥ１Ｂ：スキップ　スペース、アスキーデータの取得
     入力：Ｘ＝検索する　アドレス
     出力：Ｘ＝見つかったアドレス
           Ａ＝アスキーコード
           Ｚｆ＝１　アスキーデータ＄００
                 ０　　　　＄００、＄２０以外
           指定したアドレスのデータを取得して
           アスキーデータ＄２０以外が取得されるまでデータを取得します

$ＥＥ８０：ＬＥＦＴ＄（
$ＥＥ８８：ＲＩＧＨＴ＄（
$ＥＥＡ２：ＭＩＤ＄（
$ＥＦ３８：ＰＯＫＥ
$ＥＦ４Ｄ：ＰＥＥＫ（
$ＥＦ６３：ＰＩＣＫ
$ＥＦ６Ｆ：ＦＲＥ（
$ＥＦ７４：ＳＧＮ（
$ＥＦ９５：ＳＰＣ（
$ＥＦＡＢ：ＬＥＮ（
$ＥＦＢ７：ＡＳＣ（
$ＥＦＢＦ：ＶＡＬ（
$ＥＦＤ３：ＦＬＤ（
$ＥＦＥＡ：ＯＰＴＩＯＮ
$ＥＦＦ０：ＯＶＦ０
$ＥＦＦ３：ＯＶＦ１
```

＄ＥＦＦ７：ＣＭＯＤＥ０
＄ＥＦＦＡ：ＣＭＯＤＥ１
＄Ｆ０００：ＢＥＥＰ１
＄Ｆ０１８：ＢＥＥＰ
＄Ｆ０２１：ＢＥＥＰ０
＄Ｆ０３３：ＤＡＴＡ
＄Ｆ０８Ｂ：ＲＥＡＤ
＄Ｆ０Ｆ３：ＲＥＳＴＯＲＥ
＄Ｆ１２０：ＦＩＮＤ”
＄Ｆ１４７：ＨＥＸ＄（
＄Ｆ１７５：ＨＰＯＳ（
＄Ｆ１８５：ＶＰＯＳ（
＄Ｆ１９Ｆ：ＬＯＣＡＴＥ
＄Ｆ１ＥＣ：ＣＨＲ＄（
＄Ｆ１ＦＤ：ＴＨＥＮ
＄Ｆ２１Ｅ：ＣＯＮＴ
＄Ｆ２２Ａ：ＲＥＴ
＄Ｆ２５５：ＧＯＴＯ
＄Ｆ２５６：ＧＯＳＵＢ
＄Ｆ２６Ｅ：ＬＥＴ
＄Ｆ２９４：ＦＯＲ
＄Ｆ２Ｆ８：ＮＥＸＴ

＄Ｆ３３０：Ｘ＝Ｘ＋８
　　入力：Ｘ
　　出力：Ｘ＋８

＄Ｆ３３２：Ｘ＝Ｘ＋４
　　入力：Ｘ
　　出力：Ｘ＋４

＄Ｆ３３３：Ｘ＝Ｘ＋３
　　入力：Ｘ
　　出力：Ｘ＋３

＄Ｆ３３Ａ：ＩＮＰＵＴ
＄Ｆ３ＢＡ：空白表示

＄Ｆ３ＢＣ：１文字表示
　　入力：A=アスキーコード

＄Ｆ３ＢＦ：ＬＬＩＳＴ
＄Ｆ３Ｃ２：ＬＩＳＴ
＄Ｆ４３２：文字列表示
　　入力：Ｘ＝文字列データ　アドレス（エンドマークは最後の文字のＢｉｔ７を１にする）
　　文字列はアスキーコード

＄Ｆ４３Ａ：エラー処理
　　入力：Ｂ＝エラーＮｏ.＄００－＄１３
　　出力：＄００１４＝＄００　ＢＡＳＩＣ　フラグ
　　エラー表示、ＢＥＥＰ、表示する行番号があれば表示して
　　スタックを＄０２４４
　　ＢＡＳＩＣのコマンド待ちへジャンプします

＄Ｆ４７Ｃ：ＣＬＳエントリ
＄Ｆ４７Ｆ：ＶＲＡＭクリア
　　出力：＄００１６＝＄Ｃ１　カーソルポジション上位
　　＄００１７＝＄００　カーソルポジション下位
　　使用：Ｂ
　　＄００Ｄ９＝Ｘレジスタ退避上位
　　＄００ＤＡ＝Ｘレジスタ退避上位

　　＄Ｃ１００－＄Ｃ３ＦＦを＄４０で埋め

カーソルポジションをホームポジション（＄Ｃ１００）にします

＄Ｆ４Ｅ３：ＭＬＯＡＤ
＄Ｆ４ＥＦ：ＬＯＡＤ
＄Ｆ４ＦＣ：ＶＥＲＩＦＹ
＄Ｆ５０８：ＳＡＶＥ
＄Ｆ５１７：ＭＳＡＶＥ

＄Ｆ７７７：スクロール
　出力：Ｘ＝＄Ｃ３Ｅ０
　使用：Ａ、Ｂ、Ｘ
　　＄００１６：カーソル　ポインタ（ＶＲＡＭアドレス）上位
　　＄００１７：カーソル　ポインタ（ＶＲＡＭアドレス）下位
　　＄００１８：画面ポインタ（スクリーンエディット）上位
　　＄００１９：画面ポインタ（スクリーンエディット）下位
　　画面を上へ一行スクロールします
　　最下位行はアスキー＄８０（ディスプレイ＄４０）が埋まります

＄Ｆ７９Ｅ：ＶＩＡ初期化
　出力：Ｘ＝＄Ｃ８００（ＶＩＡのポート）
　使用：Ｂ、Ｘ
　　ＰＡ、ＰＢの入出力方向の設定を行ないます
　　ＤＤＲＡ　＄Ｃ８０３＝＄１Ｆ
　　ＤＤＲＢ　＄Ｃ８０２＝＄２０

＄Ｆ７Ａ９：ＢＡＳＩＣキー入力
　出力：Ａ＝キーデータ（何も押されていない場合は＄００）
　使用：Ａ、Ｂ、Ｘ
　　＄００１４：ＢＡＳＩＣ　フラグ
　　＄００Ｄ９＝スタック退避上位
　　＄００ＤＡ＝スタック退避上位
　　＄００ＤＢ＝キーインデックス
　　＄００ＤＤ＝Ｘレジスタ退避上位
　　＄００ＤＥ＝Ｘレジスタ退避上位
　　＄００ＤＦ＝キーデータ
　　＄００Ｅ０＝キーセレクト
　　　｜

＄ＦＡ６Ａ－＄ＦＡＦ０　キーマトリクス　データ
＄ＦＡＦ１－＄ＦＢＢＦ　エラーメッセージ

＄ＦＣ９１：ＣＭＴ１バイトライト、チェックサムの加算
　入力：ＣＭＴに書き込むアドレス
　出力：＄００７３　加算されたチェックサム

＄ＦＣＡＢ：ＣＭＴ　’１’×ｎｎ　ライト（スタートビットの書き込み）
　入力：Ｂ＝ｎｎ

＄ＦＣＢ１：ＣＭＴ　ビット単位のライト
　入力：Ａ＝ライトデータ
　　Ｂ＝書き込むビット数

　　１バイト書き込む場合　　Ａ＝データ、Ｂ＝＄０８
　　’１’を２５５個書き込む場合　Ａ＝＄ＦＦ、Ｂ＝＄ＦＦ

＄ＦＣＤ４：ＣＭＴ　’１’×２５５　ライト

＄ＦＥ４７：ＣＭＴ　スタートビットを読む
　入力：Ｂ＝＄０４

＄ＦＥ０Ｅ：ＣＭＴ　１バイトリード、チェックサムの加算
　出力：Ａ＝リードデータ

　　＄００７３　加算されたチェックサム

＄ＦＥＤ７：ＣＭＴオープン
　　キーセレクト０（＾Ｃ　ＢＲＥＡＫのスキャンの為）
　　ポート＄Ｃ８０４－＄Ｃ８０Ｆの内容をフタックに保存
　　ＶＩＡをＣＭＴ用の設定（ＩＥＲ　＄７Ｆ）を行います

＄ＦＥＤ７：ＣＭＴクローズ
　　ＢＥＥＰ、オープン時に保存したポート＄Ｃ８０４－＄Ｃ８０Ｆをスタックから復活
　　画面、右上の１文字をクリアを行います

BASICプログラム

ＢＡＳＩＣプログラムの格納は他機種のような中間言語を使用せず、アスキーで格納されています。
１行は行番号（２バイト）、プログラム（アスキー）、行の終わりは＄００
最後の行は、行番号（２バイト）、プログラム（アスキー）、行の終わりは＄００、＄ＤＦ，＄ＤＦ
プログラムの後に配列になります。
配列は変数名（１バイト）、次の配列のアドレス、パラメータ、パラメータ、配列の内容となり
配列の変数名が＄ＤＦで終わりを示します。
配列の後は２文字の変数で、
変数名（２バイト）、内容（２バイト）
変数名が＄ＤＦで終わり
次が未使用で＄００＊３４
文字列で変数名（１バイト）、長さ、３２文字分の文字列
変数名が＄ＤＦで終わりになります。

Home へ戻る