お客様各位

カタログ等資料中の旧社名の扱いについて

2010年4月1日を以ってNECエレクトロニクス株式会社及び株式会社ルネサステクノロジが合併し、両社の全ての事業が当社に承継されております。従いまして、本資料中には旧社名での表記が残っておりますが、当社の資料として有効ですので、ご理解の程宜しくお願い申し上げます。

ルネサスエレクトロニクス ホームページ（http://www.renesas.com）

2010年4月1日
ルネサスエレクトロニクス株式会社

【発行】ルネサスエレクトロニクス株式会社（http://www.renesas.com）

【問い合わせ先】http://japan.renesas.com/inquiry

ご注意書き

1. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
3. 当社製品を改造、改変、複製等しないでください。
4. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器の設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因しお客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
5. 輸出に際しては、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。本資料に記載されている当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途の目的で使用しないでください。また、当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器に使用することができません。
6. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されていない用途に当社製品を使用したことによりお客様または第三者に生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。

   標準水準： コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット

   高品質水準： 輸送機器（自動車、電車、船舶等）、交通用信号機器、防災・防犯装置、各種安全装置、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）

   特定水準： 航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めておりますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害などを生じさせないようお客様の責任において冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、機器またはシステムとしての出荷保証をお願いいたします。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、詳細につきましては製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。お客様がかかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

注 1. 本資料において使用されている「当社」とは、ルネサスエレクトロニクス株式会社およびルネサスエレクトロニクス株式会社がその総株主の議決権の過半数を直接または間接に保有する会社をいいます。

注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

RENESAS

アプリケーション・ノート

78K0/Kx2

サンプル・プログラム

漢字フォント編

対象デバイス

78K0/KB2マイクロコントローラ
78K0/KC2マイクロコントローラ
78K0/KD2マイクロコントローラ
78K0/KE2マイクロコントローラ
78K0/KF2マイクロコントローラ

資料番号　U19211JJ1V0AN00（第1版）
発行年月　August 2008 N

〔メ　モ〕

目次要約

CMOSデバイスの一般的注意事項

① 入力端子の印加波形

入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。

CMOSデバイスの入力がノイズなどに起因して，V_{IL}（MAX.）からV_{IH}（MIN.）までの領域にとどまるような場合は，誤動作を引き起こす恐れがあります。入力レベルが固定な場合はもちろん，V_{IL}（MAX.）からV_{IH}（MIN.）までの領域を通過する遷移期間中にチャタリングノイズ等が入らないようご使用ください。

② 未使用入力の処理

CMOSデバイスの未使用端子の入力レベルは固定してください。

未使用端子入力については，CMOSデバイスの入力に何も接続しない状態で動作させるのではなく，プルアップかプルダウンによって入力レベルを固定してください。また，未使用の入出力端子が出力となる可能性（タイミングは規定しません）を考慮すると，個別に抵抗を介してV_{DD}またはGNDに接続することが有効です。

資料中に「未使用端子の処理」について記載のある製品については，その内容を守ってください。

③ 静電気対策

MOSデバイス取り扱いの際は静電気防止を心がけてください。

MOSデバイスは強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には，当社が出荷梱包に使用している導電性のトレーやマガジン・ケース，または導電性の緩衝材，金属ケースなどを利用し，組み立て工程にはアースを施してください。プラスチック板上に放置したり，端子を触ったりしないでください。

また，MOSデバイスを実装したボードについても同様の扱いをしてください。

④ 初期化以前の状態

電源投入時，MOSデバイスの初期状態は不定です。

電源投入時の端子の出力状態や入出力設定，レジスタ内容などは保証しておりません。ただし，リセット動作やモード設定で定義している項目については，これらの動作ののちに保証の対象となります。

リセット機能を持つデバイスの電源投入後は，まずリセット動作を実行してください。

⑤ 電源投入切断順序

内部動作および外部インタフェースで異なる電源を使用するデバイスの場合，原則として内部電源を投入した後に外部電源を投入してください。切断の際には，原則として外部電源を切断した後に内部電源を切断してください。逆の電源投入切断順により，内部素子に過電圧が印加され，誤動作を引き起こしたり，異常電流が流れ内部素子を劣化させたりする場合があります。

資料中に「電源投入切断シーケンス」についての記載のある製品については，その内容を守ってください。

⑥ 電源OFF時における入力信号

当該デバイスの電源がOFF状態の時に，入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流注入により，誤動作を引き起こしたり，異常電流が流れ内部素子を劣化させたりする場合があります。

資料中に「電源OFF時における入力信号」についての記載のある製品については，その内容を守ってください。

- 本資料に記載されている内容は2008年8月現在のもので，今後，予告なく変更することがあります。量産設計の際には最新の個別データ・シート等をご参照ください。
- 文書による当社の事前の承諾なしに本資料の転載複製を禁じます。当社は，本資料の誤りに関し，一切その責を負いません。
- 当社は，本資料に記載された当社製品の使用に関連し発生した第三者の特許権，著作権その他の知的財産権の侵害等に関し，一切その責を負いません。当社は，本資料に基づき当社または第三者の特許権，著作権その他の知的財産権を何ら許諾するものではありません。
- 本資料に記載された回路，ソフトウエアおよびこれらに関連する情報は，半導体製品の動作例，応用例を説明するものです。お客様の機器の設計において，回路，ソフトウエアおよびこれらに関連する情報を使用する場合には，お客様の責任において行ってください。これらの使用に起因しお客様または第三者に生じた損害に関し，当社は，一切その責を負いません。
- 当社は，当社製品の品質，信頼性の向上に努めておりますが，当社製品の不具合が完全に発生しないことを保証するものではありません。また，当社製品は耐放射線設計については行っておりません。当社製品をお客様の機器にご使用の際には，当社製品の不具合の結果として，生命，身体および財産に対する損害や社会的損害を生じさせないよう，お客様の責任において冗長設計，延焼対策設計，誤動作防止設計等の安全設計を行ってください。
- 当社は，当社製品の品質水準を「標準水準」，「特別水準」およびお客様に品質保証プログラムを指定していただく「特定水準」に分類しております。また，各品質水準は，以下に示す用途に製品が使われることを意図しておりますので，当社製品の品質水準をご確認ください。

  標準水準：コンピュータ，OA機器，通信機器，計測機器，AV機器，家電，工作機械，パーソナル機器，産業用ロボット

  特別水準：輸送機器（自動車，電車，船舶等），交通用信号機器，防災・防犯装置，各種安全装置，生命維持を目的として設計されていない医療機器

  特定水準：航空機器，航空宇宙機器，海底中継機器，原子力制御システム，生命維持のための医療機器，生命維持のための装置またはシステム等

当社製品のデータ・シート，データ・ブック等の資料で特に品質水準の表示がない場合は，標準水準製品であることを表します。意図されていない用途で当社製品の使用をお客様が希望する場合には，事前に当社販売窓口までお問い合わせください。

（注）

（1）本事項において使用されている「当社」とは，NECエレクトロニクス株式会社およびNECエレクトロニクス株式会社がその総株主の議決権の過半数を直接または間接に保有する会社をいう。

（2）本事項において使用されている「当社製品」とは，（1）において定義された当社の開発，製造製品をいう。

M8E0710J

はじめに

対象者　このマニュアルは,78K0マイクロコントローラの応用システムを設計,開発するユーザを対象とします。

目　的　78K0マイクロコントローラ向け漢字フォント・サンプル・ライブラリを用いた漢字テキスト表示方法についてユーザに理解していただくことを目的とします。

構　成　このマニュアルは，大きく分けて次の内容で構成しています。

- 概　説
- ライブラリ仕様
- アプリケーション実装例
- サンプル・アプリケーションのビルド方法
- サンプル・アプリケーションの実行方法

読み方　このマニュアルの読者には，電気，論理回路，マイクロコンピュータおよびC言語に関する一般知識を必要とします。

本アプリケーションの実行評価環境を理解しようとするとき
→漢字表示デモンストレーション用ボードの**ユーザーズ・マニュアル**を参照してください。

マイクロコントローラのハードウエア機能の詳細を理解しようとするとき
→各製品の**ユーザーズ・マニュアル　ハードウエア編**を参照してください。

凡　例　データ表記の重み：左が上位桁，右が下位桁
アクティブ・ローの表記：$\overline{xxx}$（端子，信号名称に上線）
メモリ・マップのアドレス：上部－上位，下部－下位
注　：本文中に付けた注の説明
注意：気を付けて読んでいただきたい内容
備考：本文の補足説明
数の表記：2進数　… xxxxまたはxxxxB
10進数 … xxxx
16進数 … xxxxH
2のべき数を示す接頭語（アドレス空間，メモリ容量）：
K（キロ）… $2^{10} = 1024$
M（メガ）… $2^{20} = 1024^2$
G（ギガ）… $2^{30} = 1024^3$

関連資料　関連資料は暫定版の場合がありますが，この資料では「暫定」の表示をしておりません。あらかじめご了承ください。

漢字表示デモンストレーション・ボードの関連資料

資　料　名	資料番号	
	和　　文	英　　文
漢字表示デモンストレーション用ベース・ボード　ユーザーズ・マニュアル	U19207J	未定
漢字表示デモンストレーション用78K0/KF2ボード　ユーザーズ・マニュアル	U19208J	未定
漢字表示デモンストレーション用78K0R/KG3ボード　ユーザーズ・マニュアル	U19209J	未定
漢字表示デモンストレーション用V850ES/JG3ボード　ユーザーズ・マニュアル	U19210J	未定
78K0/Kx2サンプル・プログラム（漢字フォント編）アプリケーション・ノート	このノート	未定
78K0R/Kx3サンプル・プログラム（漢字フォント編）アプリケーション・ノート	U19212J	未定
V850ES/Jx3サンプル・プログラム（漢字フォント編）アプリケーション・ノート	U19213J	未定
78K0/Kx2サンプル・プログラム（簡易OS編）アプリケーション・ノート	U19214J	未定
78K0R/Kx3サンプル・プログラム（簡易OS編）アプリケーション・ノート	U19215J	未定
V850ES/Jx3サンプル・プログラム（簡易OS編）アプリケーション・ノート	U19216J	未定

78K0**マイクロコントローラ・デバイスの関連資料**

資　料　名	資料番号	
	和　　文	英　　文
78K0/Kx2　ユーザーズ・マニュアル	U18598J	U18598E
78K0マイクロコントローラ ユーザーズ・マニュアル 命令編	U12326J	U12326E

注意　上記関連資料は予告なしに内容を変更することがあります。設計などには，必ず最新の資料をご使用ください。

目　　次

第1章　概　　説

この資料では，78K0向け漢字フォント・サンプル・ライブラリFK114LVH.LIB（有償ツール用）およびFK114LVH_F.LIB（無償ツール用）を用いた漢字表示方法についてサンプル・プログラムを例に説明します。

1. 1　ライブラリ概要

本ライブラリには，JIS X0208の内，第1水準漢字と一部の非漢字の14×14ドットのフォントが実装されています。フォントは商用フリー・フォントを元データとして使用しています。

漢字表示を行うためのAPI関数には，漢字コード取得用関数とフォント・データ位置取得用関数の2つが用意されています。

1. 2　開発環境

サンプル・プログラムからオブジェクト・コードを生成するソフトウエア・ツールと，生成したコードを実行する評価ボードについて説明します。

1. 2. 1　ソフトウエア・ツール

本ライブラリを使用するアプリケーション・プログラムの開発に推奨するソフトウエア・ツールは，NECエレクトロニクス製の有償ソフトウエア・パッケージ（SP78K0）です。

(1) 推奨リビジョン

ソフトウエア・ツールの推奨リビジョンです。

(a) CC78K0　　：4.00以上
(b) RA78K0　　：4.01以上

(2) 無償ダウンロード版の制約

無償ダウンロード版のコンパイラおよびアセンブラ（CC78K0, RA78K0）でもかまいませんが，次の制約があります。

(a) ビルドしただけではフォント・データが実装されません。
(b) フォント・データを実装するには，エディタによるHEXファイルの編集が必要になります。
詳細は，「4. 3. 3　HEXファイルの編集」を参照してください。

1.2.2 評価ボード

評価用ボードには，用途に応じて次のものがあります。

（1）表示を伴わないライブラリ単体の評価を行う場合

QB-78K0KF2-TBまたはTK-78K0/KF2を使用できます。ただし，これらのボードでは後述のサンプル・アプリケーションを動作させることはできません。

（2）LCD表示を伴う評価を行う場合

次の3つが必要となります。

（a）漢字表示デモンストレーション用ベース・ボード（SM05A2）

（b）漢字表示デモンストレーション用78K0**ボード**（SM05F2）

（c）プログラム書き込みツール

推奨するプログラム書き込みツールは，オンチップ・デバッグ・エミュレータ（QB-MINI2）です。PG-FP4などのフラッシュ・メモリ・プログラマ（以降，プログラマ）でも可能です。

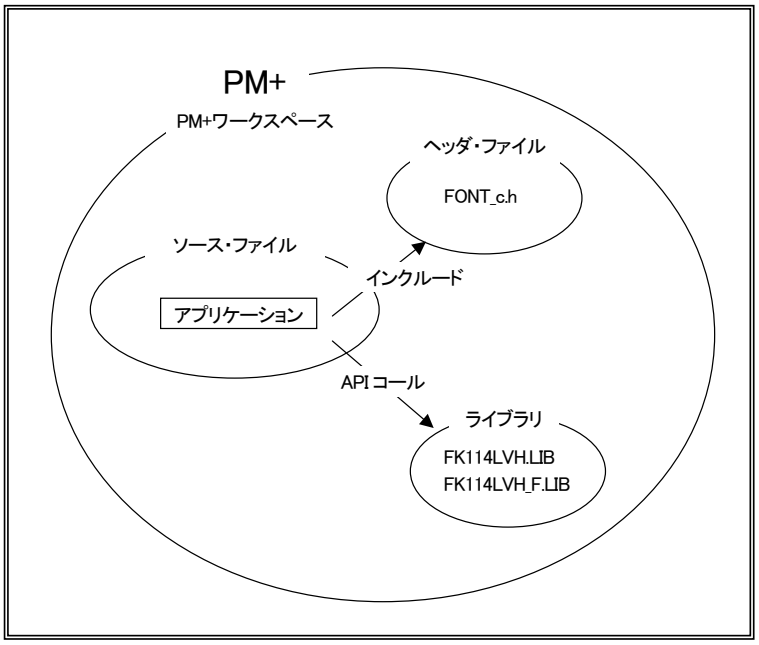
漢字表示デモンストレーション用ボードの入手に関する情報については，下記のボード設計情報をご参照ください。

http://www.necel.com/micro/ja/designsupports/index.html

1. 3　アプリケーションの依存情報

プロジェクト・マネージャ（以降，PM+）環境下での依存情報を次の図に示します。

図1－1　PM+環境下での依存情報の関係



第2章　ライブラリ仕様

この章では，漢字フォント・サンプル・ライブラリの仕様について説明します。

2. 1　フォント・データ仕様

本ライブラリに実装されているフォント・データの詳細について説明します。

2. 1. 1　実装文字数

サンプル・ライブラリFK114LVH.LIBおよびFK114LVH_F.LIBは，JIS X0208の定義文字の中から第1水準漢字2965文字，非漢字387文字の合計3352文字を抜き出してフォントを実装しています。実装文字一覧は「2. 1. 5 フォント実装文字一覧」を参照してください。

2. 1. 2　フォント・データ

フォントの元データは14×14ドットの商用フリー・フォント（Toshiyuki Imamura氏のK14-2004（version 1.00））を使用しています。詳しくは次のURLを参照してください。

http://www12.ocn.ne.jp/~imamura/jisx0213.html

なお，本ライブラリは学習および機能評価用です。実際の製品に組み込むフォントとしては，“ 高い視認性と美しいデザインで定評 ” のNEC製FontAvenueを推奨します。

2. 1. 3　フォント・データの並び

フォント・データは次の順番で並んでいます。（「図2－1　フォント・データの並びの例」を参照してください。）

相対位置　内容

＋0 ：有効ドット開始位置

14×14ドットのボディに対し，実際に文字が始まるドット位置を示します。

ライブラリFK114LVH.LIBおよびFK114LVH_F.LIBの場合は水平開始位置を示します。

プロポーショナル制御のために使用します。

＋1 ：有効ドット幅

14×14ドットのボディに対し，実際に文字が存在する幅を示します。

ライブラリFK114LVH.LIBおよびFK114LVH_F.LIBの場合は水平幅を示します。

プロポーショナル制御のために使用します。

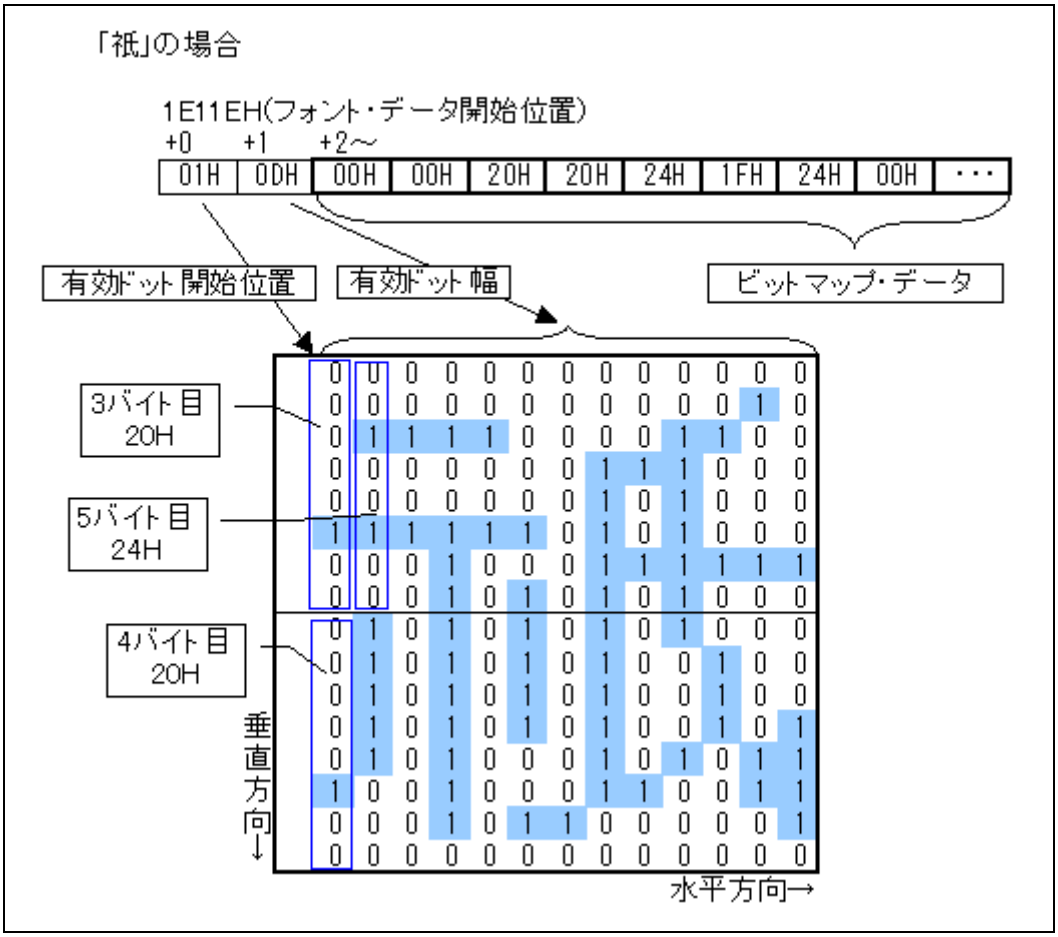
＋2～：ビットマップ・データ

ライブラリFK114LVH.LIBおよびFK114LVH_F.LIBの場合は垂直，水平のドット順に並びます。

LSBファーストです。

注　本データの並びはuFont_get_font_addr関数の返り値が示す位置からの並びになります。フラッシュ・メモリ上は，圧縮データ形式で格納するため上記とは異なる並びになります。

図2－1　フォント・データの並びの例



ビットマップ・データは，垂直方向の1列が上下8ドット単位で，上側バイト→下側バイトの順に設定されます。さらに，この2バイトで構成される列データは，水平方向の左側の列から右側の列に向かって設定されます。

2. 1. 4　フォント・データの配置

ライブラリFK114LVH.LIBおよびFK114LVH_F.LIBのフォント・データ本体は、バンク0の2B7CH番地～バンク5のBFFBH番地に配置されます。

2. 1. 5　フォント実装文字一覧

フォントが実装されている文字の一覧です。

(1) 非漢字一覧

387文字のフォントが実装されています。

、。，．・：；？！゛゜´｀¨＾￣＿ヽヾゝゞ〃仝々〆〇ー―‐／＼～∥｜…‥‘’“
”（）〔〕［］｛｝〈〉《》「」『』【】＋－±×÷＝≠＜＞≦≧∞∴♂♀°′″℃¥＄
￠￡％＃＆＊＠§☆★○●◎◇◆□■△▲▽▼※〒→←↑↓〓０１２３４５６７８９ＡＢ
ＣＤＥＦＧＨＩＪＫＬＭＮＯＰＱＲＳＴＵＶＷＸＹＺａｂｃｄｅｆｇｈｉｊｋｌｍｎｏｐ
ｑｒｓｔｕｖｗｘｙｚぁあぃいぅうぇえぉおかがきぎくぐけげこござさざしじすずせぜそぞ
ただちぢっつづてでとどなにぬねのはばぱひびぴふぶぷへべぺほぼぽまみむめもゃやゅゆ

ょよらりるれろゃわゐゑをんァアィイゥウェエォオカガキギクグケゲコゴサザシジスズセゼソゾタダチヂッツヅテデトドナニヌネノハバパヒビピフブプヘベペホボポマミムメモャヤュユョヨラリルレロヮワヰヱヲンヴヵヶΑΒΓΔΕΖΗΘΙΚΛΜΝΞΟΠΡΣΤΥΦΧΨΩαβγδεζηθικλμνξοπρστυφχψω

注 [ ] は全角の空白文字です。

(2) 漢字一覧

2965文字のフォントが実装されています。

亜唖娃阿哀愛挨姶逢葵茜穐悪握渥旭葦芦鯵梓圧斡扱宛姐虻飴絢綾鮎或粟袷安庵按暗案闇鞍杏以伊位依偉囲夷委威尉惟意慰易椅為畏異移維緯胃萎衣謂違遺医井亥域育郁磯一壱溢逸稲茨芋鰯允印咽員因姻引飲淫胤蔭院陰隠韻吋右宇烏羽迂雨卯鵜窺丑碓臼渦嘘唄欝蔚鰻姥厩浦瓜閏噂云運雲荏餌叡営嬰影映曳栄永泳洩瑛盈穎頴英衛詠鋭液疫益駅悦謁越閲榎厭円園堰奄宴延怒掩援沿演炎焔煙燕猿縁艶苑薗遠鉛鴛塩於汚甥凹央奥往応押旺横欧殴王翁襖鴬鴎黄岡

沖荻億屋憶臆桶牡乙俺卸恩温穏音下化仮何伽価佳加可嘉夏嫁家寡科暇果架歌河火珂禍禾稼箇花苛茄荷華菓蝦課嘩貨迦過霞蚊俄峨我牙画臥芽蛾賀雅餓駕介会解回塊壊廻快怪悔恢懐戒拐改魁晦械海灰界皆絵芥蟹開階貝凱劾外咳害崖慨概涯碍蓋街該鎧骸浬馨蛙垣柿蛎鈎劃嚇各廓拡撹格核殻獲確穫覚角赫較郭閣隔革学岳楽額顎掛笠樫橿梶鰍潟割喝恰括活渇滑葛褐轄且鰹叶椛樺鞄株兜竃蒲釜鎌噛鴨栢茅萱粥刈苅瓦乾侃冠寒刊勘勧巻喚堪姦完官寛干幹患感慣憾

換敢柑桓棺款歓汗漢澗潅環甘監看竿管簡緩缶翰肝艦莞観諌貫還鑑間閑関陥韓館舘丸含岸巌玩癌眼岩翫贋雁頑顔願企伎危喜器基奇嬉寄岐希幾忌揮机旗既期棋棄機帰毅気汽畿祈季稀紀徽規記貴起軌輝飢騎鬼亀偽儀妓宜戯技擬欺犠疑祇義蟻誼議掬菊鞠吉吃喫桔橘詰砧杵黍却客脚虐逆丘久仇休及吸宮弓急救朽求汲泣灸球究窮笈級糾給旧牛去居巨拒拠挙渠虚許距鋸漁凖魚亨享京供侠僑兇競共凶協匡卿叫喬境峡強彊怯恐恭挟教橋況狂狭矯胸脅興蕎郷鏡響饗驚仰

凝尭暁業局曲極玉桐粁僅勤均巾錦斤欣欽琴禁禽筋緊芹菌衿襟謹近金吟銀九倶句区狗玖矩苦躯駆駈駒具愚虞喰空偶寓遇隅串櫛釧屑屈掘窟沓靴轡窪熊隈粂栗繰桑鍬勲君薫訓群軍郡卦袈祁係傾刑兄啓圭珪型契形径恵慶慧憩掲携敬景桂渓畦稽系経継繋罫茎荊蛍計詣警軽頚鶏芸迎鯨劇戟撃激隙桁傑欠決潔穴結血訣月件倹倦健兼券剣喧圏堅嫌建憲懸拳捲検権牽犬献研硯絹県肩見謙賢軒遣鍵険顕験鹸元原厳幻弦減源玄現絃舷言諺限乎個古呼固姑孤己庫弧戸故枯湖

狐糊袴股胡菰虎誇跨鈷雇顧鼓五互伍午呉吾娯後御悟梧檎瑚碁語誤護醐乞鯉交佼侯候倖光公功効勾厚口向后喉坑垢好孔孝宏工巧巷幸広庚康弘恒慌抗拘控攻昂晃更杭校梗構江洪浩港溝甲皇硬稿糠紅紘絞綱耕考肯肱腔膏航荒行衡講貢購郊酵鉱砿鋼閤降項香高鴻剛劫号合壕拷濠豪轟麹克刻告国穀酷鵠黒獄漉腰甑忽惚骨狛込此頃今困坤墾婚恨懇昏昆根梱混痕紺艮魂些佐叉唆嵯左差査沙瑳砂詐鎖裟坐座挫債催再最哉塞妻宰彩才採栽歳済災采犀砕砦祭斎細菜裁載

際剤在材罪財冴坂阪堺榊肴咲崎埼碕鷺作削咋搾昨朔柵窄策索錯桜鮭笹匙冊刷察拶撮擦札殺薩雑皐鯖捌錆鮫皿晒三傘参山惨撒散桟燦珊産算纂蚕讃賛酸餐斬暫残仕仔伺使刺司史嗣四士

始姉姿子屍市師志思指支孜斯施旨枝止死氏獅祉私糸紙紫肢脂至視詞詩試誌諮資賜雌飼歯事似侍児字寺慈持時次滋治爾璽痔磁示而耳自蒔辞汐鹿式識鴫竺軸宍雫七叱執失嫉室悉湿漆疾質実蔀篠偲柴芝屡蕊縞舎写射捨赦斜煮社紗者謝車遮蛇邪借勺尺杓灼爵酌釈錫若寂弱惹主取

守手朱殊狩珠種腫趣酒首儒受呪寿授樹綬需囚収周宗就州修愁拾洲秀秋終繍習臭舟蒐衆襲讐蹴輯週酋酬集醜什住充十従戎柔汁渋獣縦重銃叔夙宿淑祝縮粛塾熟出術述俊峻春瞬竣舜駿准循旬楯殉淳準潤盾純巡遵醇順処初所暑曙渚庶緒署書薯藷諸助叙女序徐恕鋤除傷償勝匠升召哨商唱嘗奨妾娼宵将小少尚庄床廠彰承抄招掌捷昇昌昭晶松梢樟樵沼消渉湘焼焦照症省硝礁祥称章笑粧紹肖菖蒋蕉衝裳訟証詔詳象賞醤鉦鍾鐘障鞘上丈丞乗冗剰城場壌嬢常情擾条杖浄

状畳穣蒸譲醸錠嘱埴飾拭植殖燭織職色触食蝕辱尻伸信侵唇娠寝審心慎振新晋森榛浸深申疹真神秦紳臣芯薪親診身辛進針震人仁刃塵壬尋甚尽腎訊迅陣靭笥諏須酢図厨逗吹垂帥推水炊睡粋翠衰遂酔錐錘随瑞髄崇嵩数枢趨雛据杉椙菅頗雀裾澄摺寸世瀬畝是凄制勢姓征性成政整星晴棲栖正清牲生盛精聖声製西誠誓請逝醒青静斉税脆隻席惜戚斥昔析石積籍績脊責赤跡蹟碩切拙接摂折設窃節説雪絶舌蝉仙先千占宣専尖川戦扇撰栓栴泉浅洗染潜煎煽旋穿箭線繊羨

腺舛船薦詮賎践選遷銭銑閃鮮前善漸然全禅繕膳糎噌塑岨措曾曽楚狙疏疎礎祖租粗素組蘇訴阻遡鼠僧創双叢倉喪壮奏爽宋層匝惣想捜掃挿掻操早曹巣槍槽漕燥争痩相窓糟総綜聡草荘葬蒼藻装走送遭鎗霜騒像増憎臓蔵贈造促側則即息捉束測足速俗属賊族続卒袖其揃存孫尊損村遜他多太汰詑唾堕妥惰打柁舵楕陀駄騨体堆対耐岱帯待怠態戴替泰滞胎腿苔袋貸退逮隊黛鯛代台大第醍題鷹滝瀧卓啄宅托択拓沢濯琢託鐸濁諾茸凧蛸只叩但達辰奪脱巽竪辿棚谷狸鱈樽

誰丹単嘆坦担探旦歎淡湛炭短端箪綻耽胆蛋誕鍛団壇弾断暖檀段男談値知地弛恥智池痴稚置致蜘遅馳築畜竹筑蓄逐秩窒茶嫡着中仲宙忠抽昼柱注虫衷註酎鋳駐椁溜猪苧著貯丁兆凋喋寵帖帳庁弔張彫徴懲挑暢朝潮牒町眺聴脹腸蝶調諜超跳銚長頂鳥勅捗直朕沈珍賃鎮陳津墜椎槌追鎚痛通塚栂掴槻佃漬柘辻蔦綴鍔椿潰坪壷嬬紬爪吊釣鶴亭低停偵剃貞呈堤定帝底庭廷弟悌抵挺提梯汀碇禎程締艇訂諦蹄逓邸鄭釘鼎泥摘擢敵滴的笛適鏑溺哲徹撤轍迭鉄典填天展店添

纏甜貼転顛点伝殿澱田電兎吐堵塗妬屠徒斗杜渡登菟賭途都鍍砥砺努度土奴怒倒党冬凍刀唐塔塘套宕島嶋悼投搭東桃梼棟盗淘湯涛灯燈当痘祷等答筒糖統到董蕩藤討謄豆踏逃透鐙陶頭騰闘働動同堂導憧撞洞瞳童胴萄道銅峠鴇匿得徳涜特督禿篤毒独読栃橡凸突椴届鳶苫寅酉瀞噸屯惇敦沌豚遁頓呑曇鈍奈那内乍凪薙謎灘捺鍋楢馴縄畷南楠軟難汝二尼弐迩匂賑肉虹廿日乳入如尿韮任妊忍認濡禰祢寧葱猫熱年念捻撚燃粘乃廼之埜嚢悩濃納能脳膿農覗蚤巴把播覇

杷波派琶破婆罵芭馬俳廃拝排敗杯盃牌背肺輩配倍培媒梅楳煤狽買売賠陪這蝿秤矧萩伯剥博拍柏泊白箔粕舶薄迫曝漠爆縛莫駁麦函箱硲箸肇筈櫨幡肌畑畠八鉢溌発醗髪伐罰抜筏閥鳩噺塙蛤隼伴判半反叛帆搬斑板氾汎版犯班畔繁般藩販範采煩頒飯挽晩番盤磐蕃蛮匪卑否妃庇彼悲扉批披斐比泌疲皮碑秘緋罷肥被誹費避非飛樋簸備尾微枇毘琵眉美鼻柊稗匹疋髭彦膝菱肘弼必畢筆逼桧姫媛紐百謬俵彪標氷漂瓢票表評豹廟描病秒苗錨鋲蒜蛭鰭品彬斌浜瀕貧賓頻敏

瓶不付埠夫婦富冨布府怖扶敷斧普浮父符腐膚芙譜負賦赴阜附侮撫武舞葡蕪部封楓風葺蕗伏副復幅服福腹複覆淵弗払沸仏物鮒分吻噴墳憤扮焚奮粉糞紛雰文聞丙併兵塀幣平弊柄並蔽閉陛米頁僻壁癖碧別瞥蔑箆偏変片篇編辺返遍便勉娩弁鞭保舗鋪圃捕歩甫補輔穂募墓慕戊暮母

簿菩倣俸包呆報奉宝峰峯崩庖抱捧放方朋法泡烹砲縫胞芳萌蓬蜂褒訪豊邦鋒飽鳳鵬乏亡傍剖坊妨帽忘忙房暴望某棒冒紡肪膨謀貌貿鉾防吠頬北僕卜墨撲朴牧睦穆釦勃没殆堀幌奔本翻凡

盆摩磨魔麻埋妹昧枚毎哩槙幕膜枕鮪柾鱒桝亦俣又抹末沫迄侭繭麿万慢満漫蔓味未魅巳箕岬密蜜湊蓑稔脈妙粍民眠務夢無牟矛霧鵡椋婿娘冥名命明盟迷銘鳴姪牝滅免棉綿緬面麺摸模茂妄孟毛猛盲網耗蒙儲木黙目杢勿餅尤戻籾貰問悶紋門匁也冶夜爺耶野弥矢厄役約薬訳躍靖柳薮鑓愉愈油癒諭輸唯佑優勇友宥幽悠憂揖有柚湧涌猶猷由祐裕誘遊邑郵雄融夕予余与誉輿預傭幼妖容庸揚揺擁曜楊様洋溶熔用窯羊耀葉蓉要謡踊遥陽養慾抑欲沃浴翌翼淀羅螺裸来莱頼

雷洛絡落酪乱卵嵐欄濫藍蘭覧利吏履李梨理璃痢裏裡里離陸律率立葎掠略劉流溜琉留硫粒隆竜龍侶慮旅虜了亮僚両凌寮料梁涼猟療瞭稜糧良諒遼量陵領力緑倫厘林淋燐琳臨輪隣鱗麟瑠塁涙累類令伶例冷励嶺怜玲礼苓鈴隷零霊麗齢暦歴列劣烈裂廉恋憐漣煉簾練聯蓮連錬呂魯櫓炉賂路露労婁廊弄朗楼榔浪漏牢狼篭老聾蝋郎六麓禄肋録論倭和話歪賄脇惑枠鷲亙亘鰐詫藁蕨椀湾碗腕

2. 2　関数仕様（API仕様）

本ライブラリに用意されている関数について説明します。以下，関数の詳細説明の読み方について解説します。

【機能】

各関数の機能の詳細を示します。

【ヘッダ・ファイル】

各関数を使用する場合に必要となるヘッダ・ファイル名を示します。

【関数プロトタイプ】

各関数の指定形式を示します。

【説明】

各関数の処理の詳細を示します。

2. 2. 1　関数一覧（リソース一覧）

本ライブラリで利用できる関数の一覧を表2－1に示します。

表2－1　関数一覧（リソース一覧）

名称	コード・サイズ	RAM消費量	スタック消費量	最大実行クロック数
uFont_get_next_code	45 バイト	0	0	116 クロック
uFont_get_font_addr	259 バイト	30 バイト	4 バイト	1396 クロック

注　数値は暫定値

2. 2. 2　uFont_get_next_code（文字列から文字コードを得る）

【機能】

全角／半角の混在した日本語文字列から1文字を取り出します。

【ヘッダ・ファイル】

FONT_c.h

【関数プロトタイプ】

unsigned short uFont_get_next_code (unsigned char *text) ;

関数名	引数	返り値
uFont_get_next_code	text： 文字列上の現在位置 （文字コードの取り出しアドレス）	半角の場合： 1バイト・コード（255以下） 全角の場合： 2バイト・コード[注]

注　**上位8ビット：**textが指す位置のバイト。ただし，値がE0H以上の場合は-64した値となります。
下位8ビット：text＋1が指す位置のバイト。ただし，値が80H以上の場合は-1した値となります。

【説明】

textが指す全角／半角の混在した日本語文字列から1文字を取り出します。
全角はシフトJISコードを使用してください。
全角と判定されるのは，次の場合です。

- 第1バイト：81H～9FH, E0H～FCHの範囲
- 第2バイト：40H～7EH, 80H～FCHの範囲

2. 2. 3　uFont_get_font_addr（文字コードからフォント・データの開始位置を得る）

【機能】

文字コードから全角フォント・データの開始位置を取得します。

【ヘッダ・ファイル】

FONT_c.h

【関数プロトタイプ】

unsigned char* uFont_get_font_addr (unsigned short i) ;

関数名	引数	返り値
uFont_get_font_addr	i： フォント・データを取り出す文字コード （uFont_get_next_codeで得た文字コード）	実装されている文字の場合： フォント・データの開始位置 未実装文字の場合： 0

【説明】

iで指定される全角文字コードのフォント・データの先頭アドレスを返します。
フォント・データの並びについては，「2. 1. 3　フォント・データの並び」を参照してください。

第3章　アプリケーション実装例

この章では，漢字フォント・サンプル・ライブラリを使った漢字表示方法について，サンプル・プログラムUser_c.cを例に説明します。User_c.cはダウンロード・ファイルのSAMPLEフォルダに格納されています。

3. 1　アプリケーションの概要

サンプル・プログラムUser_c.cは，全角／半角混在の表示文字列をビットマップ・フォントに展開してビットマップ表示メモリに格納するプログラムです。以下，User_c.cの中の主要な４つの関数について説明します。

① uUser_disp_JH（詳細3. 2項）
文字列から1文字取り出して，文字に対応するフォント・データの先頭位置を計算します。

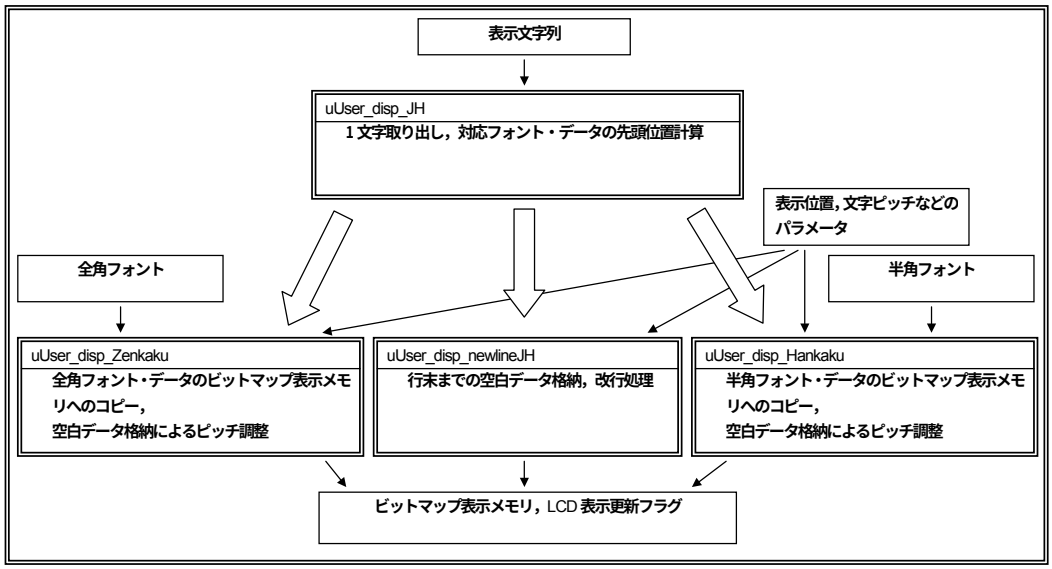
② uUser_disp_Zenkaku（詳細3. 3項）
全角1文字のフォント・データをビットマップ表示メモリの指定位置にコピーします。
また文字ピッチ調整のための空白を付加します。

③ uUser_disp_Hankaku（詳細3. 4項）
半角1文字のフォント・データをビットマップ表示メモリの指定位置にコピーします。
また文字ピッチ調整のための空白を付加します。

④ uUser_disp_newlineJH（詳細3. 5項）
行末まで空白で埋めたうえで改行処理をします。

なお，User_c.cの中の上記以外の部分は，ホスト・マシンからのコマンド処理や1/4角処理，半角専用処理であり，本資料では説明を省略します。
また，ビットマップ表示メモリのデータをLCDへ転送する部分は別プログラム（LCD_c.c）になっています。本資料ではLCD_c.cの説明は省略します。

図3－1　漢字表示アプリケーション処理のブロック図



3. 1. 1　ビットマップ表示メモリの構造

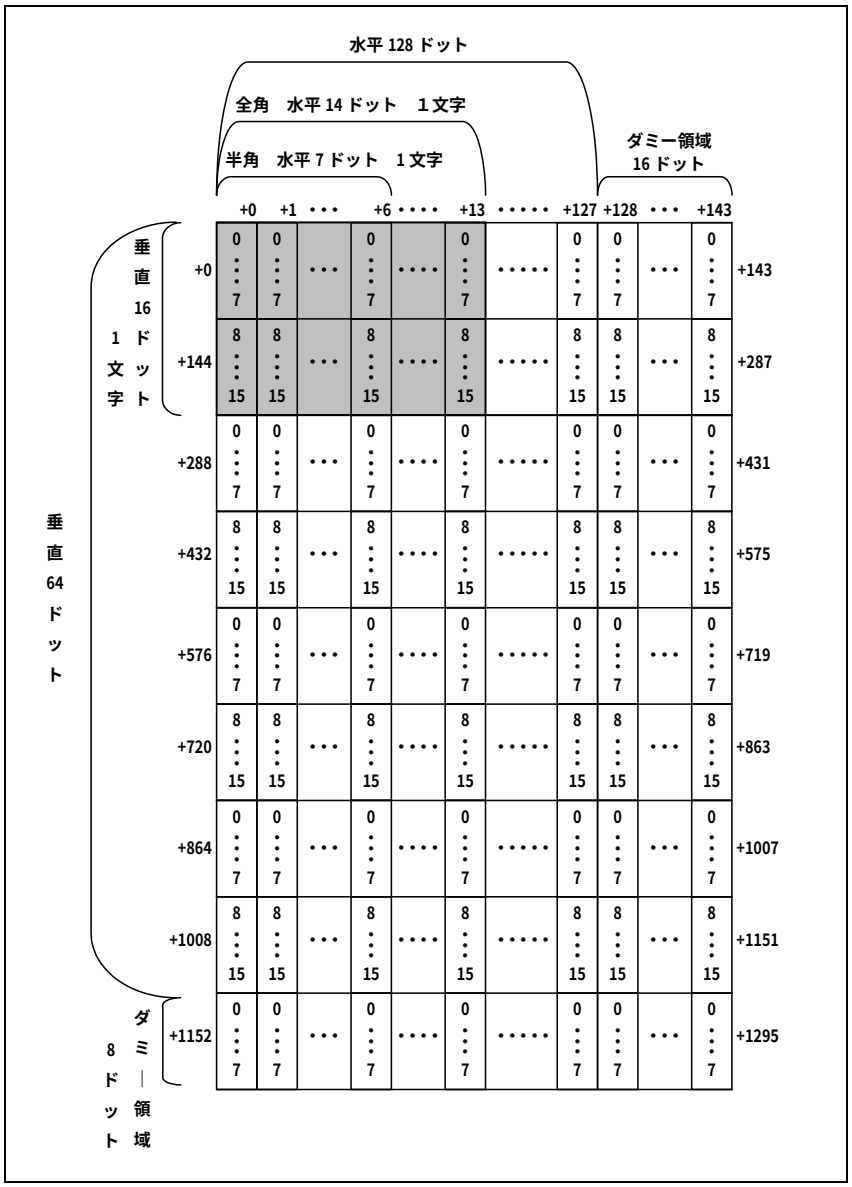
ビットマップ表示メモリは，LCD表示面全体の表示ドット情報を保持します。LCD表示面ドット数は水平×垂直が128×64ドットですが，サンプル・プログラムでは処理の都合上ビットマップ表示メモリは（128＋16）×（64＋8）ビット分のバイト領域となっており，（（128＋16）／8）×（64＋8）＝1296バイトが用意されています。領域はバイト型配列で定義されており，水平（X）から垂直（Y）方向にデータが格納されます。

次の図では，ビットマップ表示メモリの構造をLCD表示面に合わせ，構成バイトの並びで示しています。枠外＋0～＋1295の数字はビットマップ表示メモリでの相対バイト位置を示します。

図では全角14×14フォントでの1文字占有範囲の目安を灰色で示しています。実際には，文字種・ピッチ指定などにより占有範囲の位置は変化します。

サンプル・プログラムではビットマップ表示メモリは配列名gLCD.textとしてLCD_c.hで定義されています。

図3－2　ビットマップ表示メモリの構造



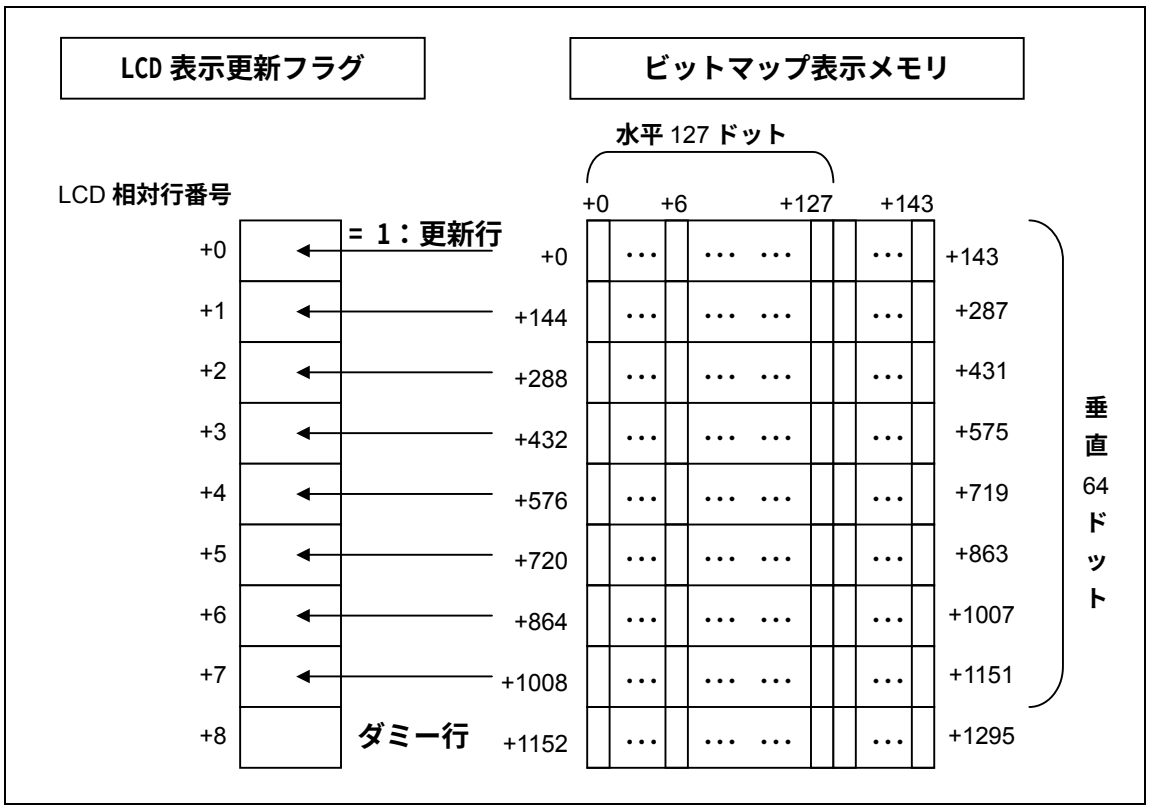
3. 1. 2　LCD表示更新フラグ

LCD表示更新フラグはフォントの展開には直接関係しませんが，変更したビットマップ表示メモリ行をLCD表示プログラム（LCD_c.c）に通知することにより，LCDへの転送負荷を低減させる役割があります。LCD表示プログラム（LCD_c.c）は，ビットマップ表示メモリの中で表示更新すべき行（8ドット・ライン単位）だけを選択してLCDへの転送を行います。

LCD相対行番号は縦8ドットを1行とした行番号です。

サンプル・プログラムではLCD表示更新フラグは配列名gLCD.dreqとしてLCD_c.hで定義されています。

図3－3　LCD表示更新フラグ



3. 1. 3　表示パラメータ

フォントを表示するために使用されているパラメータには次のようなものがあります。

定義場所	定義方法	変数名	意味
User_c.h	構造体	gUser.x	水平表示位置X コマンドによるライト／リード可能なパラメータ領域アドレス0です。
		gUser.y	垂直表示位置Y コマンドによるライト／リード可能なパラメータ領域アドレス1です。
		gUser.space	フォント表示　空白ドット数（0～3） コマンドによるライト／リード可能なパラメータ領域アドレス2のbit3-0です。
		gUser.pitch	フォント表示　ピッチ指定（0：固定ピッチ，1：プロポーショナル） コマンドによるライト／リード可能なパラメータ領域アドレス2のbit4です。
		gUser.font	フォント表示　フォント指定（0：1/4角，1：半角，2：全角／半角混在） コマンドによるライト／リード可能なパラメータ領域アドレス2のbit7-5です。
		gUser.position	ビットマップ表示メモリへの格納開始位置
		gUser.before_sp	全角　文字前空白ドット数
		gUser.after_sp	全角　文字後空白ドット数
		gUser.width	全角　実文字幅ドット数
		gUser.before_spH	半角　文字前空白ドット数
		gUser.after_spH	半角　文字後空白ドット数
		gUser.widthH	半角　実文字幅ドット数
LCD_c.h	#define	nLCD_xs	128：LCD表示横ドット数
		nLCD_ys	64：LCD表示縦ドット数
		nLCD_xm	nLCD_xs＋16：ビットマップ表示メモリの横ドット数
		nLCD_ym	nLCD_ys＋8：ビットマップ表示メモリの縦ドット数
		nLCD_text	nLCD_xm*nLCD_ym/8：ビットマップ表示メモリのバイト数
		nLCD_dreq	nLCD_ym/8：LCD表示更新フラグ領域の行数（8ドット行）
	構造体	gLCD.dreq	LCD表示更新フラグ配列（1：更新あり）
		gLCD.text	ビットマップ表示メモリ

3. 2　全角／半角文字列表示（uUser_disp_JH）

全角／半角　文字列表示処理を示します。uUser_disp_JH関数では，全角／半角混在文字列をLCDに表示します。本関数は，大きく分けると初期処理部分と文字列表示処理部分からなります。

(1) 初期処理

文字ピッチの計算とビットマップ表示メモリ格納開始バイト位置の計算を行います。

```
/*
 *       関数名：uUser_disp_JH()
 *       機能　：全角/半角 文字列表示
 *       引数　：表示文字列
 *       戻り値：なし
 */
void uUser_disp_JH(unsigned char *text) {
        unsigned char *k;
        unsigned short i;

        /*-------------------------*/
        /*   全角文字ピッチ計算        */
        /*-------------------------*/
        gUser.before_sp = 0;
        if (gUser.space >= 1) gUser.before_sp = 1;
        gUser.after_sp = 0;
        if (gUser.space >= 2) gUser.after_sp = 1;
        gUser.width = 14;
        /*-------------------------*/
        /*   半角文字ピッチ計算        */
        /*-------------------------*/
        gUser.before_spH = 0;
        if (gUser.space >= 1) gUser.before_spH = 1;
        gUser.after_spH = 0;
        gUser.widthH = 7;
        /*---------------------------------------*/
        /*   ビットマップ表示メモリ上の表示開始位置計算  */
        /*---------------------------------------*/
        if (gUser.x >= nLCD_xs) gUser.x = 0;
        if (gUser.y >= nLCD_ys) gUser.y = 0;
        gUser.position = (unsigned short)(gUser.y >> 3) * nLCD_xm + gUser.x;
```

(a) 文字ピッチの計算

(b) 表示位置チェック

(c) ビットマップ表示メモリ格納開始バイト位置の計算

（つづく）

(a) 文字ピッチの計算

文字前・文字後の空白ドット数の配分計算と文字幅を設定します。

①　全角の場合

空白ドットがなるべく均等になるように配分を行います。

・文字間空白ドット数が2ドットの場合：前後空白にそれぞれ1ドットを配分します。

・文字間空白ドット数が1ドットの場合：文字前空白ドット数に1ドットを割り当てます。

文字幅を14ドットとします。

②　半角の場合

空白ドットの配分を行います。

・文字間空白ドット数が1ドットの場合：文字前空白ドット数に1ドットを割り当てます。

・文字後空白ドットは0ドット固定です。

文字幅を7ドットとします。

(b) 表示位置チェック

表示位置がLCD表示範囲外の場合には先頭位置に戻しておきます。

(c) ビットマップ表示メモリ格納開始バイト位置の計算

文字列群のビットマップ表示メモリへの格納開始位置を計算します。

格納開始位置 = LCD相対行番号×ビットマップ表示メモリの横ドット数＋水平表示位置X

(2) 文字列表示処理

表示文字列から1文字づつ取り出し，全角／半角／改行コードに応じてフォント・データをビットマップ表示メモリに格納します。

（つづき）

```
        /*-------------------------*/
        /* 文字表示                */
        /*-------------------------*/
        while (*text) {
                i = uFont_get_next_code(text);              /* 次の文字コードを得るライブラリ関数 */
                if (i > 0x0ff) {
                        /*-------------------------*/
                        /* 全角処理                */
                        /*-------------------------*/
                        text += 2;                                      /* 次の文字位置へ移動 */
                        k = uFont_get_font_addr(i);       /* 漢字フォントの位置を得るライブラリ関数 */
                        if (!k) k=uFont_get_font_addr(0x81a0);
                                                /* エラー時は■を表示（引数には 0x81a1-1 を格納） */
                        uUser_disp_Zenkaku(k);                          /* 全角フォントを展開 */
                }
                else if (i == '¥n') {
                        /*-------------------------*/
                        /* 全半角改行処理          */
                        /*-------------------------*/
                        text ++;                                        /* 次の文字位置へ移動 */
                        uUser_disp_newlineJH();                         /* 改行 */
                }
                else {
                        /*-------------------------*/
                        /* 半角処理                */
                        /*-------------------------*/
                        text ++;                                        /* 次の文字位置へ移動 */
                        uUser_disp_Hankaku(&tcUser_hank[i * 16]);       /* 半角フォントを展開 */
                }
        }
        return;
}
```

(a) 文字コードの取り出しと各処理関数呼び出し

uFont_get_next_code関数に表示文字列領域のアドレスを引き渡して次の文字コードを取り出します。uFont_get_next_code関数では，指定された位置の文字コードが全角か半角かを判断し，16ビット文字コードを返します。取り出した文字コード種別に応じて，フォント・データをビットマップ表示メモリに格納するための処理を行います。取り出される文字コードの詳細は，「2. 2. 2 uFont_get_next_code」の関数仕様を参照してください。取り出し処理はNULLコード（0H）で終了します。

①　全角フォント処理

表示文字列の取り出し位置を2バイト分進めて，次の文字の取り出し位置へ更新します。

uFont_get_font_addr関数を呼び出してフォント・データのアドレスを取得します。フォント・データが取得できなかった場合（実装していないフォントの場合）は，本サンプル・プログラムでは代わりに表示する「■」のフォント・データのアドレスを取得します。

uUser_disp_Zenkaku関数を呼び出して，全角フォント・データをビットマップ表示メモリに格

納します。uUser_disp_Zenkaku関数の詳細は，「3. 3　全角1文字展開（uUser_disp_Zenkaku）」を参照してください。

② 全角／半角改行処理

表示文字列の取り出し位置を1バイト分進めて，次の文字の取り出し位置へ更新します。

uUser_disp_newlineJH関数を呼び出して，ビットマップ表示メモリ上の改行処理を行います。

uUser_disp_newlineJH関数の詳細は，「3. 5　全角／半角改行処理（uUser_disp_newlineJH）」を参照してください。

③ 半角フォント処理

表示文字列の取り出し位置を1バイト分進めて，次の文字の取り出し位置へ更新します。

uUser_disp_Hankaku関数を呼び出して，半角フォント・データをビットマップ表示メモリに格納します。半角フォント・データはFH114LVH.hで定数定義されています。この定数定義をバイト配列の初期値定義として組み込むことにより，文字コードから計算してフォント・データを取り出しています。（「図3－4　半角フォント・データ組み込み例」を参照してください。）

フォント・データ・アドレス = 半角フォント・データ領域先頭アドレス＋（半角コード値×16）

16：フォント・データのバイト・サイズ

uUser_disp_Hankaku関数の詳細は，「3. 4　半角1文字展開（uUser_disp_Hankaku）」を参照してください。

図3－4　半角フォント・データ組み込み例

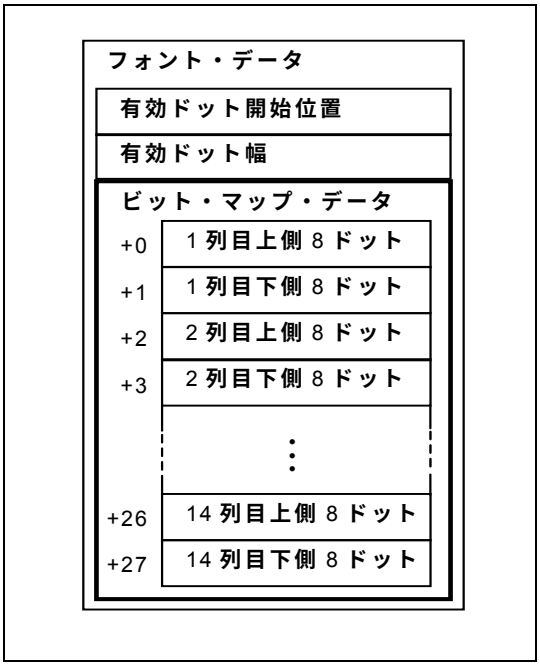
```
const unsigned char tcUser_hank[4096]={
        #include "FH114LVH.h"          /* 半角フォント [256]x[16] */
};
```

FH114LVH.hは，Toshiyuki Imamura氏が実験的に公開している7x14ドット・フォント（A14）に基づいて作成したものです。逆さ疑問符（コードBFH）の形状もそのままで変更していません。

3. 3　全角1文字展開（uUser_disp_Zenkaku）

全角フォント・データをビットマップ表示メモリに展開します。本関数は，大きく分けるとビットマップ・データ取り出し位置計算部分とビットマップ・データの格納部分からなります。引数は次の図のフォント・データの先頭位置となります。

図3－5　フォント・データ構造



(1) フォント・データ中のビットマップ・データ取り出し位置計算

ビットマップ表示メモリに格納するビットマップ・データの取り出し位置を計算します。

```
/*
 *      関数名：uUser_disp_Zenkaku()
 *      機能　：全角 1 文字を展開
 *      引数　：全角文字コード
 *      戻り値：なし
 */
void uUser_disp_Zenkaku(unsigned char *k) {
        unsigned char i;

        if (gUser.x >= nLCD_xs) return;                  /* 水平方向表示範囲を超えている場合、処理を行わない */

        /*-------------------------*/
        /* 表示幅計算              */
        /*-------------------------*/
        if (gUser.pitch) {
                /*-------------------------*/
                /* プロポーショナル        */
                /*-------------------------*/
                i = *k ++;                                                          /* 有効ドット開始位置取得 */
                gUser.width = *k ++;                                                /* 有効ドット幅取得 */
                /*-------------------------------------------------------------------------------------*/
                if (i > 13) i = 0;                              /* 無償ツール版でデバッグする場合のチェックです */
                if (gUser.width > 14) gUser.width = 14 - i;  /* 無償ツール版でデバッグする場合のチェックです */
                /*-------------------------------------------------------------------------------------*/
                k += i * 2;                                                 /* フォント・データ開始位置へ移動 */
        }
        else {
                /*-------------------------*/
                /* 固定ピッチ              */
                /*-------------------------*/
                k += 2;
        }
```

（つづく）

(a) プロポーショナル指定の場合

フォント・データから有効ドット開始位置と有効ドット幅を取り出します。

取り出したパラメータが不正な値の場合には，正常範囲に訂正を行います。無償ツールを使用したときに，不正なフォント・データが引き渡される可能性があるためにチェックを行っています。

有効ドット開始位置からビットマップ・データ取り出し位置を計算します。

取り出しバイト位置 = ビットマップ・データ先頭＋（有効ドット開始位置×2）

- 有効ドット開始位置は，フォント・データに設定されている値を使用します。
- 有効ドット開始位置を2倍しているのは，1列ドットが上側ドットと下側ドットの2バイトで構成されていることによります。

(b) 固定ピッチ指定の場合

ビットマップ・データ取り出し位置をフォント・データ中のビットマップ・データ先頭とします。

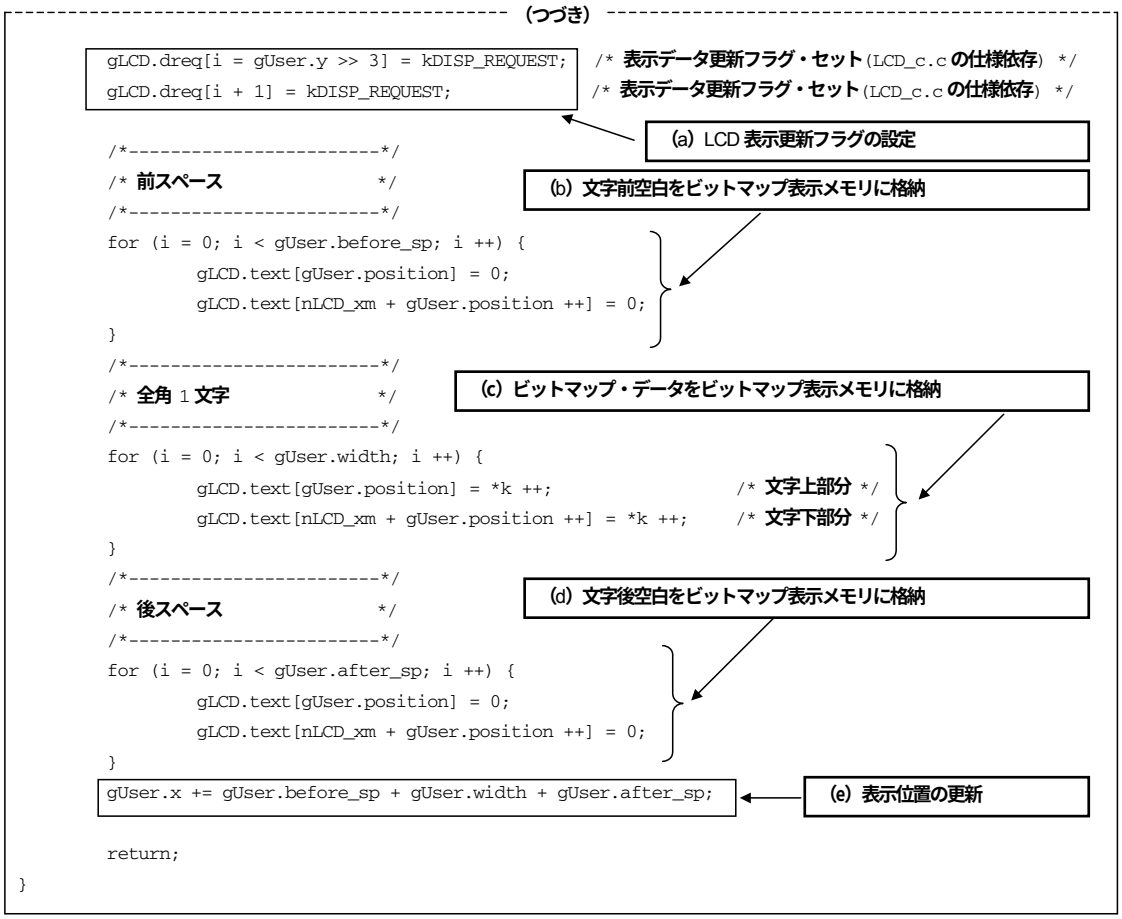
取り出しバイト位置 = ビットマップ・データ先頭

= フォント・データ先頭位置＋2

(2) ビットマップ・データの格納

LCDに表示させるために，LCD表示更新フラグをセットします。

文字前空白，フォント・データのビットマップ・データと文字後空白をビットマップ表示メモリにコピーし，表示位置を次表示位置に更新します。



(a) LCD表示更新フラグの設定

表示領域を更新するLCD相対行番号に対応したLCD表示更新フラグを設定します。

フォント上側8ドットLCD相対行番号 = （垂直表示位置Y／8）

フォント下側8ドットLCD相対行番号 = （垂直表示位置Y／8）＋1

(b) 文字前空白をビットマップ表示メモリに格納

上側8ドットと下側8ドットの2バイトの0Hを，空白ドット数分格納します。

(c) ビットマップ・データをビットマップ表示メモリに格納

上側8ドットと下側8ドットの2バイトのビットマップ・データを，有効文字幅分格納します。格納処理はフォント・データの縦列左側から上→下側ドットの順に右側に向かって行われます。

(d) 文字後空白をビットマップ表示メモリに格納

上側8ドットと下側8ドットの2バイトの0Hを，空白ドット数分格納します。

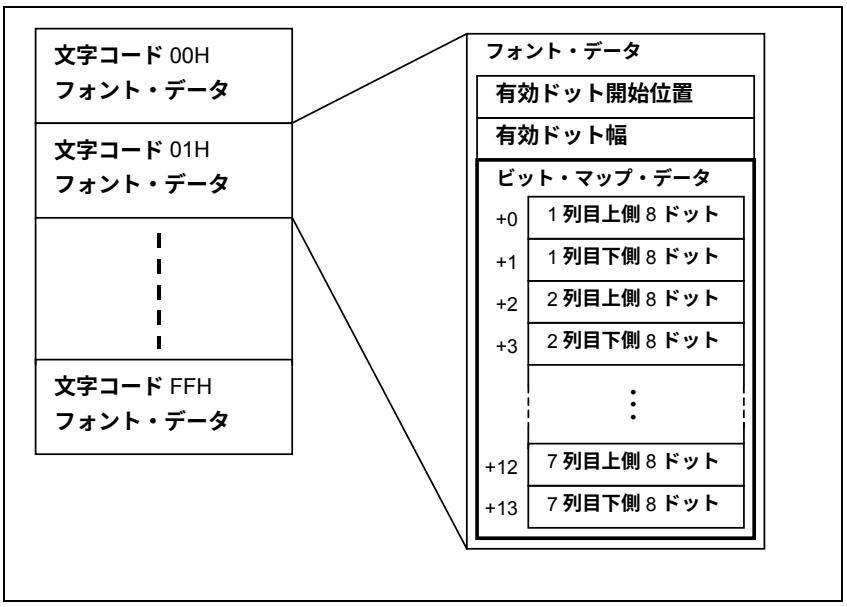
(e) 表示位置の更新

水平表示位置Xを有効文字幅ドット数と文字前後空白ドット数分加算して，次文字の開始位置に更新します。

3. 4　半角1文字展開（uUser_disp_Hankaku）

半角フォント・データをビットマップ表示メモリに展開します。本関数は，大きく分けるとビットマップ・データ取り出し位置計算部分とビットマップ・データの格納部分からなります。引数は次の図の右側コード別フォント・データの先頭位置となります。

図3－6　半角フォント・データの構造



(1) フォント・データ中のビットマップ・データ取り出し位置計算

ビットマップ表示メモリに格納するビットマップ・データの取り出し位置を計算します。

```
/*
 *       関数名：uUser_disp_Hankaku()
 *       機能　：半角 1 文字を展開
 *       引数　：半角文字コード
 *       戻り値：なし
 */
void uUser_disp_Hankaku(unsigned char *k) {
        unsigned char i;

        if (gUser.x >= nLCD_xs) return;     /* 水平方向表示範囲を超えている場合、処理を行わない */

        /*-------------------------*/
        /* 表示幅計算              */
        /*-------------------------*/
        if (gUser.pitch) {
                /*-------------------------*/
                /* プロポーショナル        */
                /*-------------------------*/
                i = *k ++;                                  /* 有効ドット開始位置取得 */
                gUser.widthH = *k ++;                       /* 有効ドット幅取得 */
                k += i * 2;                                 /* フォント・データ開始位置へ移動 */
        }
        else {
                /*-------------------------*/
                /* 固定ピッチ              */
                /*-------------------------*/
                k += 2;
        }
```

（つづく）

(a) プロポーショナル指定の場合

フォント・データから有効ドット開始位置と有効ドット幅を取り出します。

有効ドット開始位置からビットマップ・データ取り出し位置を計算します。

取り出しバイト位置 = ビットマップ・データ先頭＋（有効ドット開始位置×2）

- 有効ドット開始位置は，フォント・データに設定されている値を使用します。
- 有効ドット開始位置を2倍しているのは，1列ドットが上側8ドットと下側8ドットの2バイトで構成されていることによります。

(b) 固定ピッチ指定の場合

ビットマップ・データ取り出し位置をフォント・データ中のビットマップ・データ先頭とします。

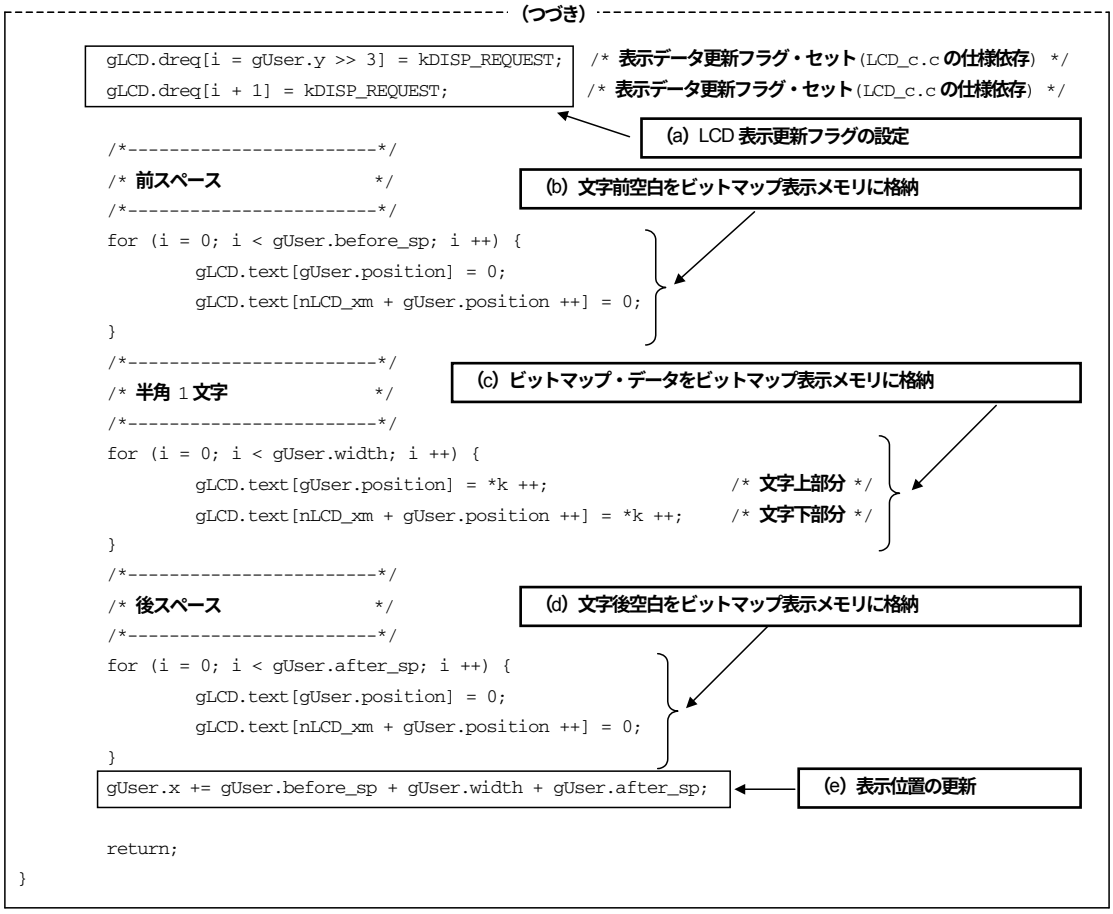
取り出しバイト位置 = ビットマップ・データ先頭

= フォント・データ先頭位置＋2

(2) ビットマップ・データの格納

LCDに表示させるために，LCD表示更新フラグをセットします。

文字前空白，フォント・データのビットマップ・データと文字後空白をビットマップ表示メモリにコピーし，表示位置を次表示位置に更新します。



(a) LCD表示更新フラグの設定

表示領域を更新するLCD相対行番号に対応したLCD表示更新フラグを設定します。

フォント上側8ドットLCD相対行番号 = （垂直表示位置Y／8）

フォント下側8ドットLCD相対行番号 = （垂直表示位置Y／8）＋1

(b) 文字前空白をビットマップ表示メモリに格納

上側8ドットと下側8ドットの2バイトの0Hを，空白ドット数分格納します。

(c) ビットマップ・データをビットマップ表示メモリに格納

上側8ドットと下側8ドットの2バイトのビットマップ・データを，有効文字幅分格納します。

(d) 文字後空白をビットマップ表示メモリに格納

上側8ドットと下側8ドットの2バイトの0Hを，空白ドット数分格納します。

(e) 表示位置の更新

水平表示位置Xを有効文字幅ドット数と文字前後空白ドット数分加算して，次文字の開始位置に更新します。

3. 5　全角／半角改行処理（uUser_disp_newlineJH）

LCD表示での改行処理を行います。

```
/*
 *       関数名：uUser_disp_newlineJH()
 *       機能　：全角/半角 改行
 *       引数　：なし
 *       戻り値：なし
 */
void uUser_disp_newlineJH(void) {
        unsigned char i;

        if (gUser.x < nLCD_xs) {
                gLCD.dreq[i = gUser.y >> 3] = kDISP_REQUEST;
                                        /* 表示データ更新フラグ・セット(LCD_c.c の仕様依存) */
                gLCD.dreq[i + 1] = kDISP_REQUEST;
                                        /* 表示データ更新フラグ・セット(LCD_c.c の仕様依存) */
        }

        /*---------------------------------*/
        /* 改行位置以降の水平方向表示処理        */
        /*---------------------------------*/
        while (gUser.x < nLCD_xs) {
                gLCD.text[gUser.position] = 0;
                gLCD.text[nLCD_xm + gUser.position ++] = 0;
                gUser.x ++;
        }

        gUser.x = 0;
        gUser.y += 16;
        if (gUser.y >= nLCD_ys) gUser.y = 0;
        gUser.position = (unsigned short)(gUser.y >> 3) * nLCD_xm;

        return;
}
```

(1) LCD 表示更新フラグの設定

(2) 現在 X 表示位置から行末までの空白設定

(3) 表示位置情報とビットマップ表示メモリの格納位置更新

（1）LCD表示更新フラグの設定

現在の水平表示位置XがLCDの水平方向表示範囲内であれば，現在行のLCD表示更新フラグを設定します。

フォント上側8ドットLCD相対行番号 ＝ （垂直表示位置Y／8）

フォント下側8ドットLCD相対行番号 ＝ （垂直表示位置Y／8）＋1

（2）現在の水平表示位置XからLCDの水平方向表示範囲までの空白設定

現在の水平表示位置XからLCDの水平方向表示範囲までに空白を格納します。

上側8ドットと下側8ドットの2バイトの0Hを，LCDの水平方向表示範囲までのドット数分格納します。

(3) 表示位置情報とビットマップ表示メモリの格納位置更新

- **水平表示位置Xを0にします。**
- **垂直表示位置Yに16を加算して，垂直方向位置を進めます。LCDの垂直方向表示範囲を超えた場合は0に初期化します。**
- **ビットマップ表示メモリの格納位置を更新します。**

  格納開始位置 = LCD相対行番号×ビットマップ表示メモリの横ドット数

第4章　サンプル・アプリケーションのビルド方法

この章では，本サンプル・アプリケーションのビルド方法の詳細について説明します。

4. 1　有償版／無償版ツールでのビルド方法の違い

開発ツールが有償版と無償版とではビルド方法が異なります。

有償版の場合：漢字フォント・サンプル・ライブラリとしてFK114LVH.LIBを指定してビルドします。そのままデバッガを立ち上げるか，生成されたHEXファイルをプログラマでデバイスに書き込めば実行できます。

無償版の場合：漢字フォント・サンプル・ライブラリとしてFK114LVH_F.LIBを指定してビルドします。そのままデバッガを立ち上げて実行すると，漢字が■で表示されます。
漢字表示を行いたい場合は，生成されたHEXファイルの8000H以上を，エディタでFK114LVH_F.HEXの内容に置き換えます。置き換えたファイルをデバッガまたはプログラマでデバイスに書き込むことにより漢字を表示することができます。

表4－1　有償版／無償版開発ツール使用上の違い

	有償版	無償版
漢字フォント・サンプル・ライブラリ	FK114LVH.LIB	FK114LVH_F.LIB
フォント・データの実装	不要です。 （自動的に実装されますので，特別な操作は不要です。）	フォント・データ実装の編集が必要になります。編集を行わずに実行すると，漢字が■で表示されます。 詳細は，「4. 3. 3　HEXファイルの編集」を参照してください。

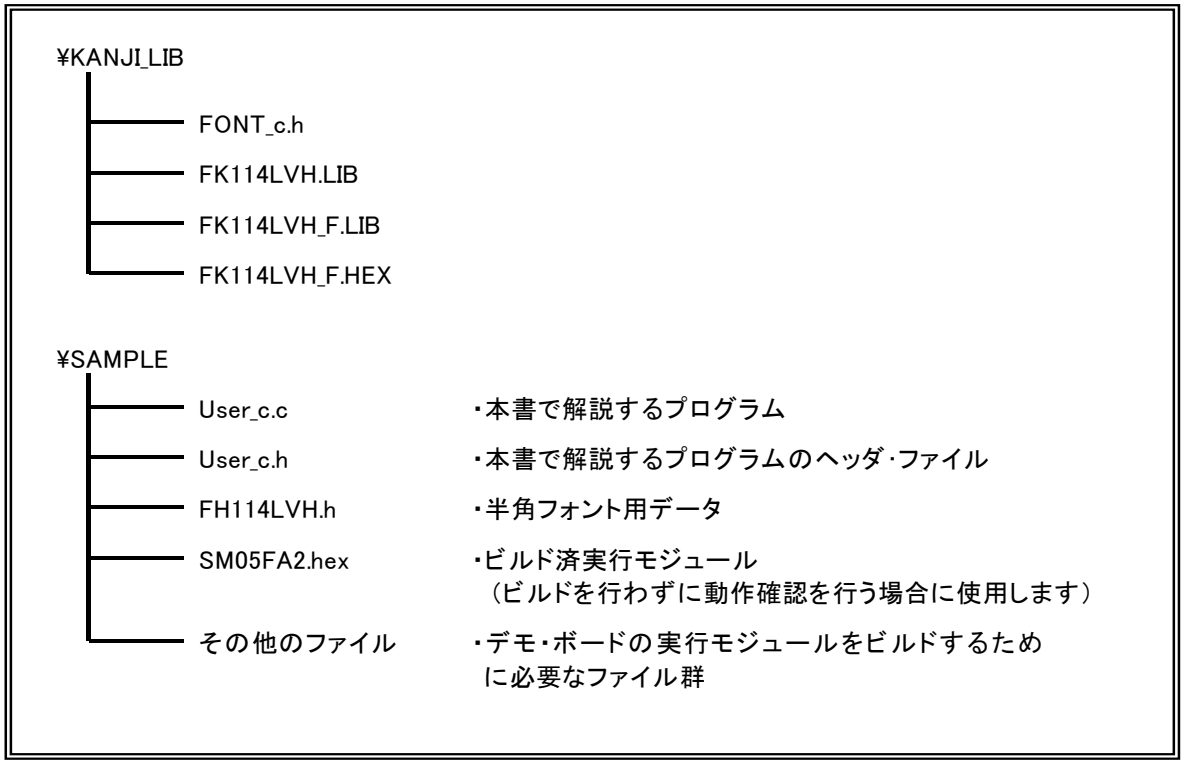
4.2　フォルダ構成

サンプル・プログラムは下記からダウンロードできます。

http://www.necel.com/micro/ja/designsupports/sampleprogram/78k0/index.html

ダウンロードしたファイルのフォルダの構成は次のようになっています。

図4－1　サンプル・アプリケーションのフォルダ構成



4. 3　実行モジュールの作成

本サンプル・プログラムの実行モジュール作成について説明します。

4. 3. 1　ファイルの選択

ワークスペースの作成とファイルの選択について説明します。

(1) ワークスペースの生成を行います。

①　ワークスペース作成の準備を行います。

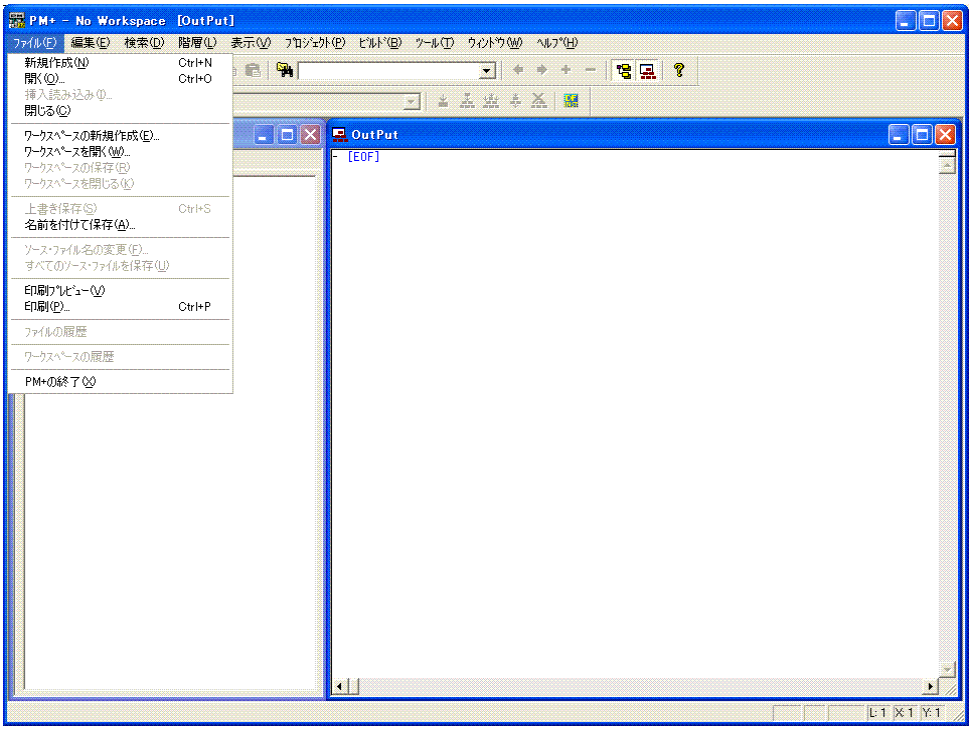
サンプル・プログラムのワークスペースを作成するフォルダを用意し，ダウンロードしたSAMPLEフォルダとKANJI_LIBフォルダにあるファイルを全てコピーしておきます。

ここではワークスペースの名称をSAMPLEAPPとして以下のフォルダにあるとします。

C:¥KANJI_SAMPLE¥78K0¥SAMPLEAPP

②　PM+を起動します。

PM+を起動後，「ファイル」メニューの「ワークスペースの新規作成」を実行します。

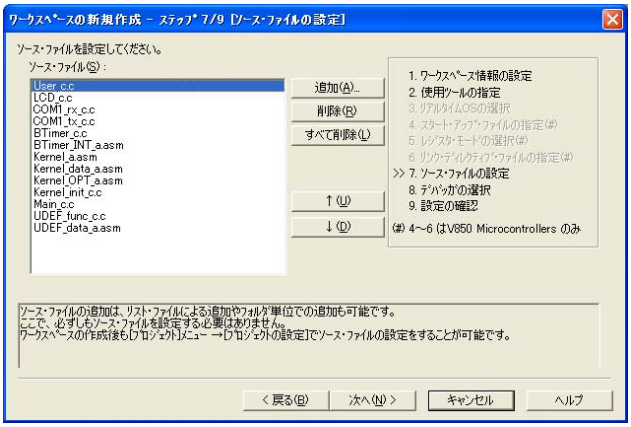


③　ワークスペースを新規作成します。

ワークスペース・ファイル名とフォルダ位置を指定し，マイクロコントローラ名とデバイス名を選択します。

③　以下のファイルを選択します。

User_c.c	：漢字表示サンプル・プログラム（サンプル・タスク）。
LCD_c.c	：表示データをLCDへ転送するタスク・プログラム。
COM1_rx_c.c	：ホスト・マシンからデータを受信するタスク・プログラム。
COM1_tx_c.c	：ホスト・マシンへ応答データを送信するタスク・プログラム。
BTimer_c.c	：OS予約のタイマ処理等。
BTimer_INT_a.asm	：OS予約のタイマ処理等（割り込みハンドラ）。
Kernel_a.asm	：OSコード部。
Kernel_data_a.asm	：OSデータ部。
Kernel_OPT_a.asm	：オプションバイト（OS依存性はありません）。
Kernel_init_c.c	：初期化プログラム（OS依存性はありません）。
Main_c.c	：メイン（システム制御）タスク・プログラム。
UDEF_func_c.c	：システム共通処理プログラム。
UDEF_data_a.asm	：システム共通処理用データ。



4. 3. 2　オプションの設定

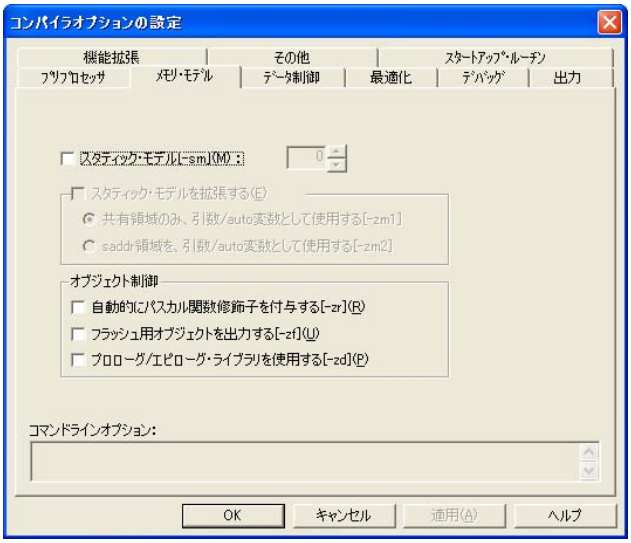
本サンプル・プログラムの実行モジュールを作成するために必要なコンパイラとリンカのオプション設定を説明します。

(1) コンパイラのオプションを設定します。

「ツール」メニューの「コンパイラオプションの設定」を選択し，設定画面を表示します。

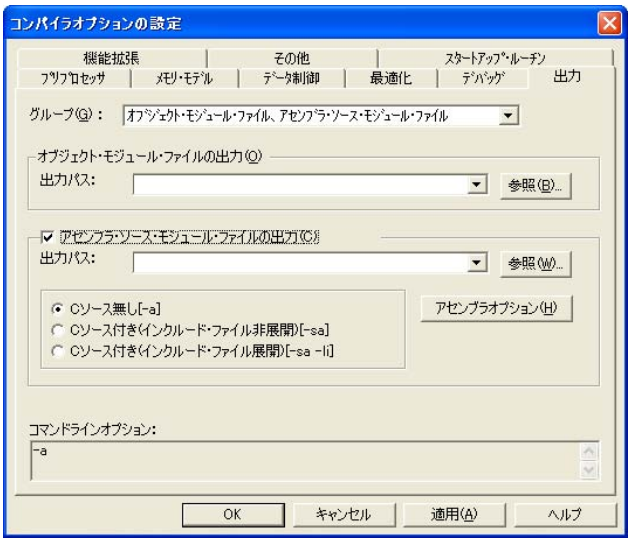
①　メモリ・モデルを選択します。

メモリ･モデルは「ノーマル」にのみ対応します。「メモリ・モデル」タブを選択し，「スタティック・モデル」オプションのチェックが外れていることを確認してください。



②　アセンブラ・ソース・モジュール・ファイルの出力をチェックします。

「出力」タブを選択し，「アセンブラ・モジュール・ファイルの出力」チェックをオンにします。



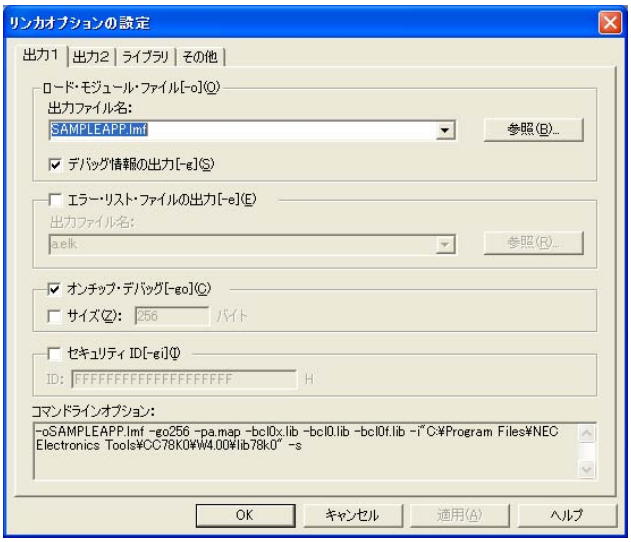
(2) リンカのオプションを設定します。

「ツール」メニューの「リンカオプションの設定」を選択し，設定画面を表示します。

① 「出力1」タブのオプションを設定します。

「ロード・モジュール・ファイル」に「SAMPLEAPP.lmf」を指定します。「セキュリティID」を必要に応じて設定します。

デバッグ時は，「オンチップ・デバッグ」を設定します。

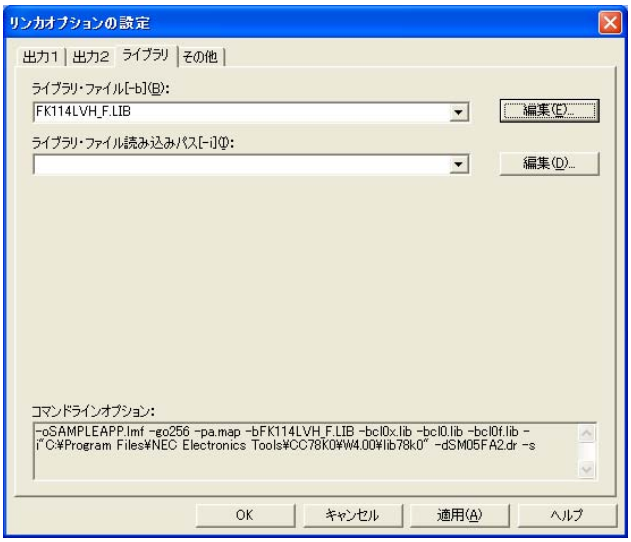


② ライブラリを指定します。

「ライブラリ」タブでFK114LVH.LIBまたはFK114LVH_F.LIBを指定します。

有償版開発ツール使用時：FK114LVH.LIB

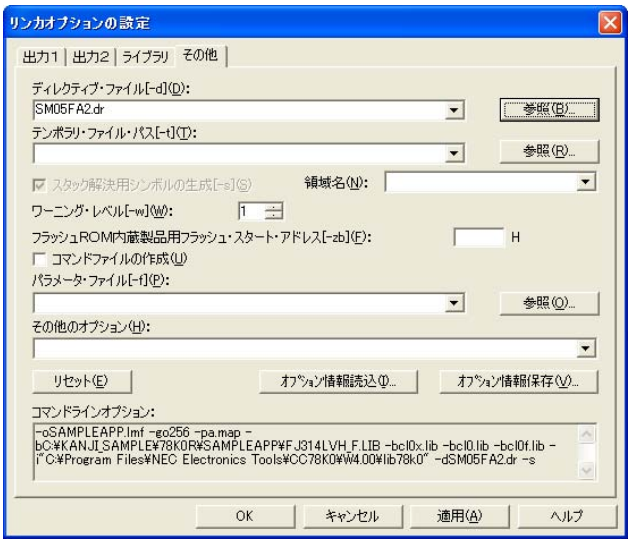
無償版開発ツール使用時：FK114LVH_F.LIB



ライブラリの指定は，ProjectWindowのプロジェクト関連ファイルで追加を行うこともできます。

③　ディレクティブ・ファイルを指定します。

「その他」タブの「ディレクティブ・ファイル」にSM05FA2.drを指定します。



4. 3. 3　HEXファイルの編集

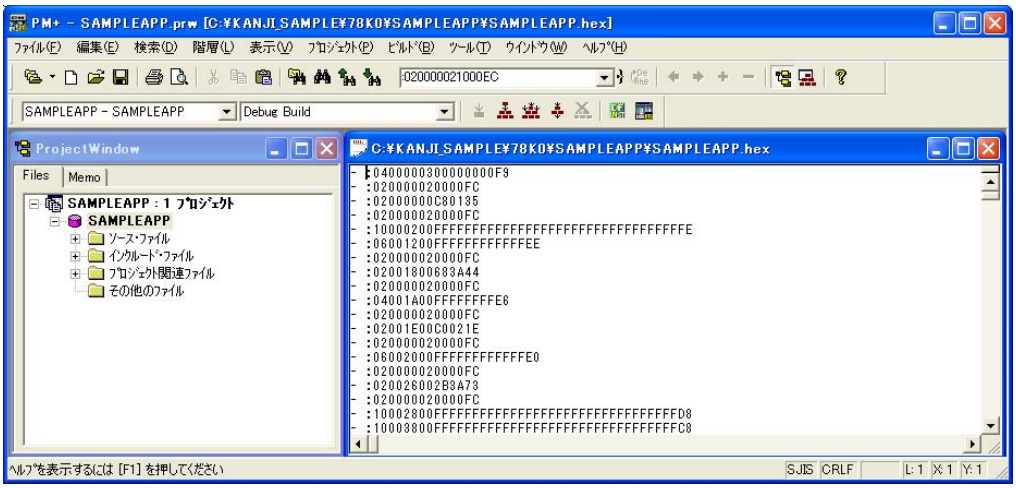
無償版ツールを使用する場合に，フォント・データを実装するためのHEXファイル編集操作について説明します。

生成されたHEXファイルの8000H以上を，FK114LVH_F.HEXの内容で置き換えて保存します。なお，80000H～AB7BHはデモ・プログラム用データのため，本書例ではAB7CH以上の置き換えだけでも動作します。

(1) ファイルの編集

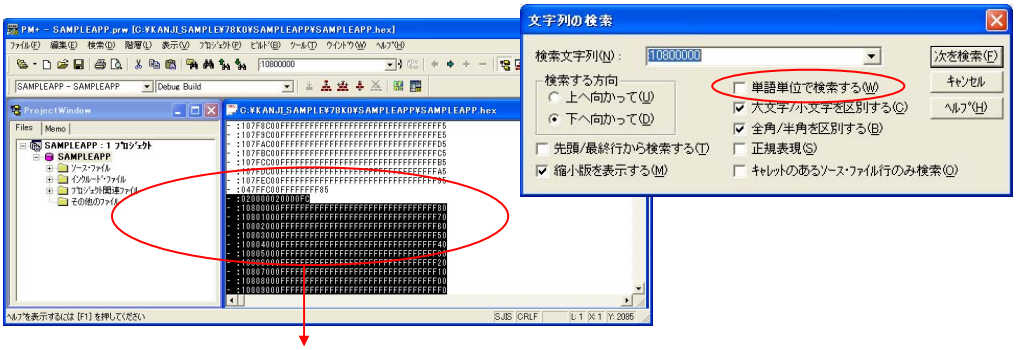
ビルドで作成されたHEXファイル（ここではSAMPLEAPP.hexファイルとします）と，SAMPPLEAPPフォルダにあるFK114LVH_F.HEXファイルをそれぞれエディタで開きます。ここでは，PM+を使用した場合の操作を説明します。

① ビルドで作成されたSAMPLEAPP.hexファイルを開きます。

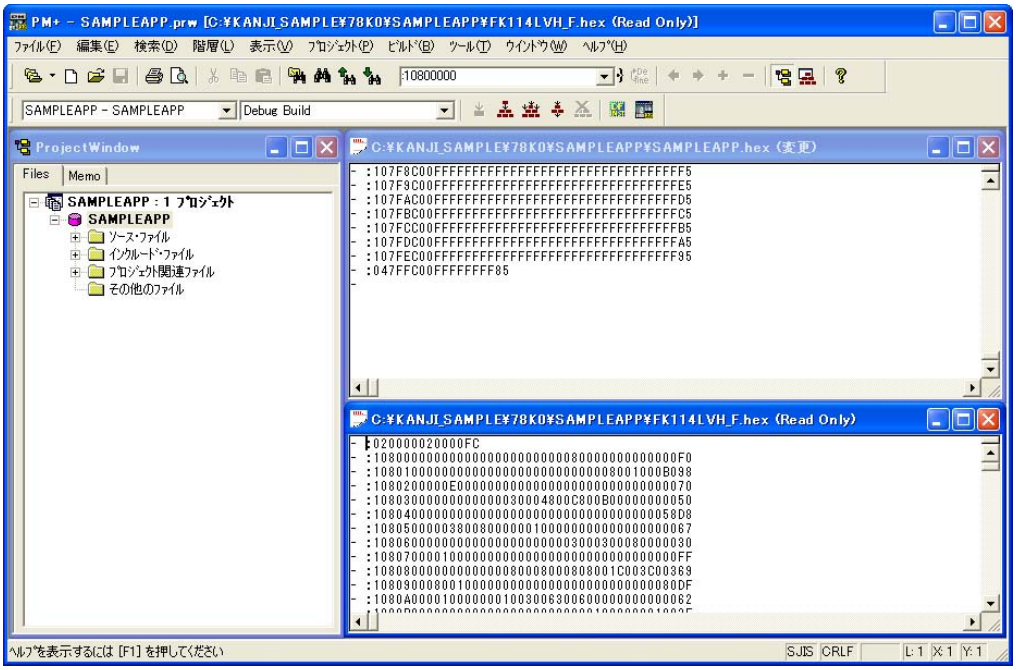


② ビルドで作成されたSAMPLEAPP.hexファイルの8000H以降を削除します。

ファイルの先頭から文字列の検索機能で文字列「:10800000」を検索します。検索は「単語単位で検索する」のチェックを外して検索してください。検索した「:10800000」から始まる行の前行「:020000020000FC」の行から，ファイルの最後まで削除します。

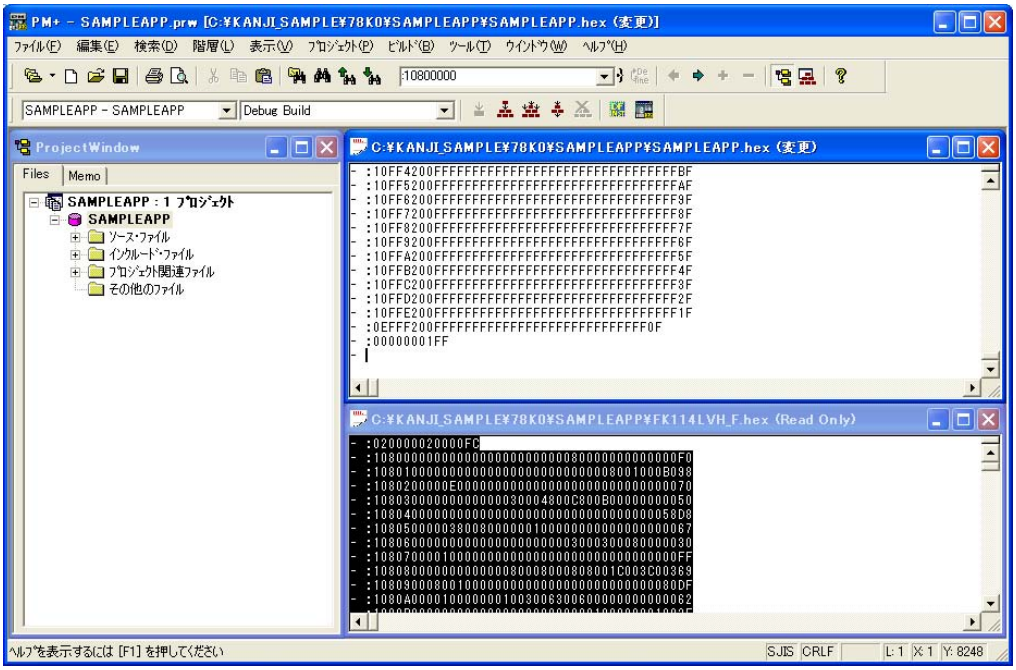


③　SAMPLEAPPフォルダにあるFK114LVH_F.HEXファイルを読み取り専用で開きます。



④　FK114LVH_F.HEXファイルの全行を②で編集したSAMPLEAPP.hexファイルに追加します。

FK114LVH_F.HEXファイルの全行を選択してコピーします。②で編集したSAMPLEAPP.hexファイルの末尾（最終行の次の行頭）にカーソルを移します。ファイル末尾が行頭でない場合は，改行文字を追加してファイル末尾の行頭にカーソルを移しておきます。その位置で，張り付け（ペースト）を行います。編集後，SAMPLEAPP.hexファイルを上書き保存します。



第5章　サンプル・アプリケーションの実行方法

この章では，プログラムの実行方法とホスト・マシンからサンプル・プログラム（User_c.c）を動作させるためのコマンド例について説明します。

デモ用ボードの設定とホスト・マシンのターミナル・ソフトウエアの設定方法については，漢字表示デモンストレーション用ベース・ボード　ユーザーズ・マニュアル（U19207J）および漢字表示デモンストレーション用78K0/KF2ボード　ユーザーズ・マニュアル（U19208J）を参照してください。

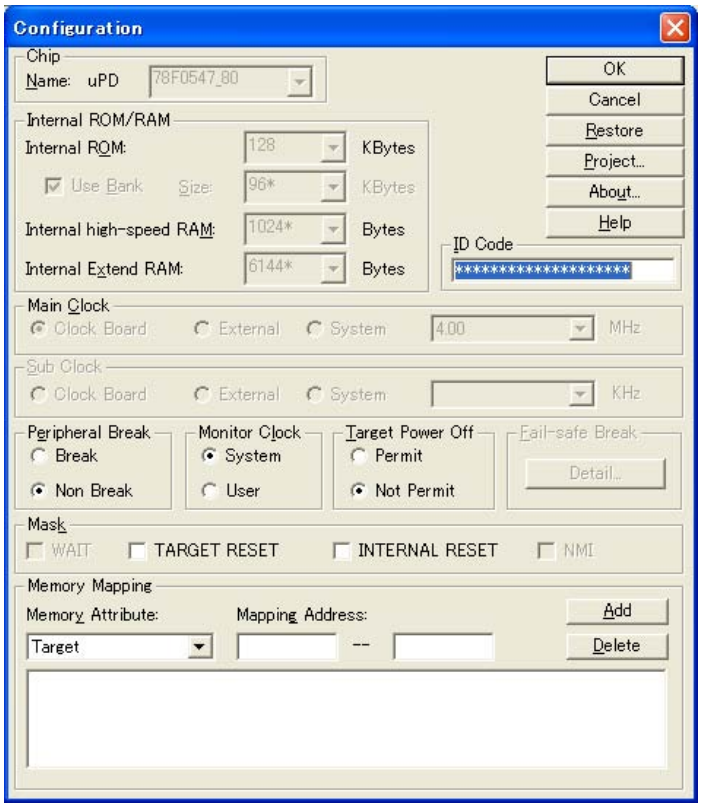
5. 1　プログラムの書き込みと起動方法

デバッガによる起動とデバッガを使わずにボード単体で起動する方法について説明します。

5. 1. 1　デバッガで起動する場合

ビルド後，デバッガ（ここではID78K0-QBを使用します）を起動します。

(1) 設定画面が表示されます。設定画面でOKボタン押下後、デバッガ画面が表示されます。

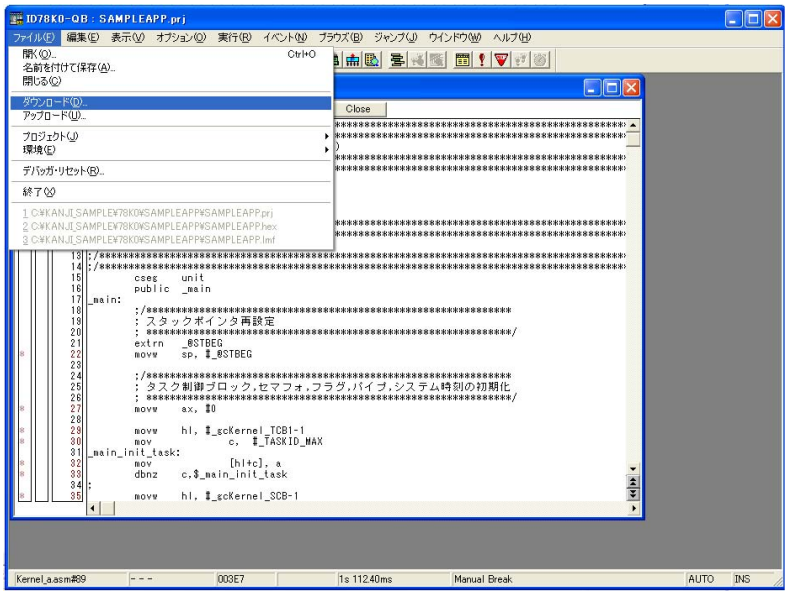


(2) 無償版開発ツールを使用した場合

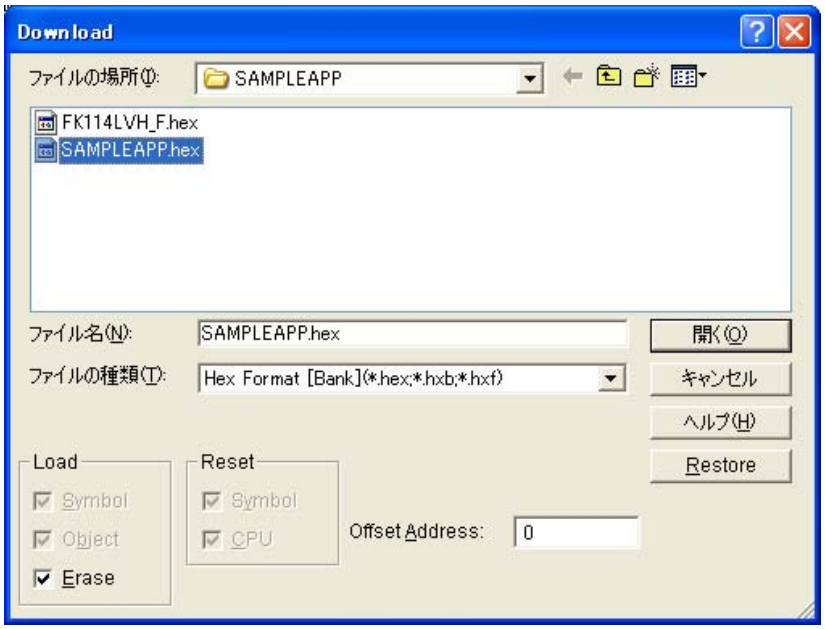
ビルド後，デバッガを起動した場合，漢字は■が表示されます。フォント・データを実装するための編集を行ったHEXファイルをダウンロードして実行することにより，漢字を正しく表示することができます。

（HEXファイル編集の詳細は，「4. 3. 3　HEXファイルの編集」を参照してください。）

(a) 「ファイル」メニューの「ダウンロード」を実行します。



(b) 編集済みHEXファイルを選択して，「開く」を実行するとダウンロードが開始されます。



(c) ダウンロード完了後に，プログラムを起動すると漢字が正しく表示されます。

5. 1. 2　プログラマで書き込んで起動する場合

ビルドで作成されたHEXファイルまたはSAMPLEフォルダダにあるSM05FA2.hexファイルをプログラマで書き込み後，ベース・ボードからプログラマを外すとデバイス・ボードが起動します。

無償版開発ツールを使用した場合には，ビルド後にHEXファイルの編集が必要です。編集を行わずに書き込み後実行しますと，漢字は■が表示されます。HEXファイル編集の詳細は「4. 3. 3　HEXファイルの編集」を参照してください。

5. 2　ターミナル・ソフトウエアの操作について

コマンド入力時の操作について説明します。

(1) キーボードからのコマンド入力

コマンド入力時に注意が必要な点を説明します。

①　コマンドの編集

コマンドはリターン・キーを押すと確定します。リターン・キーを押すまでは，バックスペースによる修正はできますが，そのほかの編集キーによる操作は無効です。

②　デモボードからの表示

入力中にデモボードからのメッセージが表示されても，表示がなかったものとして，続けて入力してください。

(2) コピー＆ペーストによるコマンド入力

コマンドは，後述のコマンド例をコピー＆ペーストしても動作します。コメントも含めて複数行を一括で張り付けしてもかまいません。

5. 3　文字の表示方法

文字を表示するには，次の順でコマンドを入力します。

(1) パラメータ・ライト・コマンドで表示位置やピッチなどを指定します。

パラメータ・ライト・コマンドを使用して表示位置やピッチなどの表示条件を指定します。文字列を続けて表示させる場合で，表示条件の変更が不要の場合は省略することができます。

(2) 文字表示コマンドで表示する日本語テキストを指定します。

文字表示コマンドで表示する日本語テキストを指定します。続けて入力することにより，前回表示位置のあとに同じ表示条件で表示させることができます。

5. 4　コマンドの書式

(1) パラメータ・ライト・コマンド

書式：$XW' {パラメータ・アドレス}　{書き込みデータの並び}

指定したパラメータを設定します。

アドレスと設定データの内容は［表5－1　パラメーター覧表］を参照してください。

16進数表記（A～Fは大文字のみ）で指定します。

表5－1　パラメーター覧表

アドレス	内容	
0	表示を開始するLCDの水平ドット位置を設定します。 有効範囲は0H～7FHです。 $XFコマンドによる文字表示実行後に次の文字表示開始位置に更新されます。	
1	表示を開始するLCDの垂直ドット位置を設定します。 有効値は0H, 8H, 10H, 18H, 20H, 28H, 30H, 38Hです。 全角／半角は垂直16ドットを使用します。 $XFコマンドによる文字表示実行後に次の文字表示開始位置に更新されます。	
2	フォント，ピッチ，字間を指定します。	
	bit7－5	フォントを指定します。 0：1/4角フォント 1：半角フォント 2：全角／半角混在フォント
	bit4	ピッチを指定します。 0：固定ピッチ 1：プロポーショナル
	bit3－0	文字の前後に付加する空白ドット数を指定します。 ・プロポーショナルの場合，フォントによらず指定数が字間になります。 ・全角固定ピッチの場合，14＋指定数の文字ピッチになります。 ・半角固定ピッチの場合，7＋指定数／2（端数切り上げ）の文字ピッチになります。 ・1/4角固定ピッチの場合，5＋指定数の文字ピッチになります。 なお設定有効値は，次のとおりです。 ・全角／半角は0～2 ・1/4角は0～3

(2) パラメータ・リード・コマンド

書式：$XR'｛パラメータ・アドレス｝　｛リード・バイト数｝

指定したパラメータ・アドレスから指定したリード・バイト数分の設定内容を読み出せます。

［パラメータ・アドレス＋リード・バイト数］は3以下（パラメータ領域サイズ以下）で指定してください。

(3) 文字表示コマンド

書式：$XF｛表示文字列｝

指定した文字列を表示します。

文字列中では，改行は'A'と書きます。'（アポストロフィ）は'27'と書きます。

全角はシフトJISコードを使います。

表示後は，パラメータ領域のアドレス0, 1の表示位置が更新されます。

注　本版では1コマンドあたりに指定できる表示文字列は25バイトまでです。

5.5　コマンド例

コマンドの使用例について説明します。表示例はそれぞれLCDに表示されるイメージを表現したものですので，個々の文字フォントや文字間スペースなどは実際の表示と異なるものがあります。

5.5.1　全角／半角文字表示コマンド例

(1) 全半角，プロポーショナル，字間１ドット

```
$XW'0 0 0 51
$XFUserタスク(ユニット'27'X'27')'A
$XFは，日本語テキストを表'A
$XF示するSampleプログラ'A
$XFムです。'A
```

```
Userタスク(ユニット'X')
は,日本語テキストを表
示するSampleプログラ
ムです。
```

(2) 全半角，16ドット固定ピッチ

以降の表示例では，平家物語の冒頭部分を引用しています。固定ピッチ表示の場合はプロポーショナル表示との差異を示すために，同じ行数で一部文字を省いて表示しています。

```
$XW'0 0 0 42
$XF祇園精舎の鐘の声'A
$XF諸行無常の響有り'A
$XF沙羅双樹の花の色'A
$XF盛者必衰の理表す'A
```

```
祇園精舎の鐘の声
諸行無常の響有り
沙羅双樹の花の色
盛者必衰の理表す
```

```
$XF驕者も久しからず'A
$XF唯春の夜の夢の如'A
$XF猛者も遂に亡びぬ'A
$XF偏に風前の塵に同'A
```

```
■者も久しからず
唯春の夜の夢の如
猛者も遂に亡びぬ
偏に風前の塵に同
```

注　'驕'は第2水準の漢字のため，本サンプル・プログラムでは■表示になります。

（3）全半角，プロポーショナル，字間1ドット

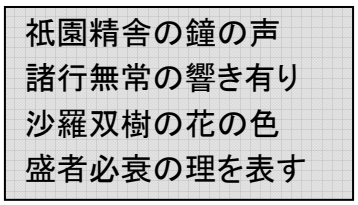
$XW'0 0 0 51

$XF**祇園精舎の鐘の声**'A

$XF**諸行無常の響き有り**'A

$XF**沙羅双樹の花の色**'A

$XF**盛者必衰の理を表す**'A



字間0

$XW'2 50

$XF**おごれる人も久しからず**'A

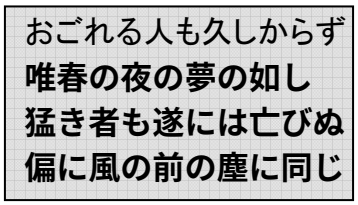
字間1

$XW'2 51

$XF**唯春の夜の夢の如し**'A

$XF**猛き者も遂には亡びぬ**'A

$XF**偏に風の前の塵に同じ**'A



5. 5. 2　1/4角文字表示コマンド例

3章で説明していませんが，User_c.cに組み込まれている1/4角表示のコマンド例を参考として載せます。

（1）1/4角，6ドット固定ピッチとプロポーショナル字間1ドットの交互表示

$XW'0 0 0 1

$XF''ABCDEFGHIJKLMNOPQRSTU'A

$XW'2 11

$XF''ABCDEFGHIJKLMNOPQRSTU'A

$XW'2 1

$XF''VWXYZabcdefghijklmnop'A

$XW'2 11

$XF''VWXYZabcdefghijklmnop'A

$XW'2 1

$XF''qrstuvwxyz0123456789.'A

$XW'2 11

$XF''qrstuvwxyz0123456789.'A

$XW'2 1

$XF"!"#$%&'27'()=~¥-^|@`[{<>'A

$XW'2 11

$XF"!"#$%&'27'()=~¥-^|@`[{<>'A

5. 5. 3　半角限定文字表示コマンド例

（1）半角限定，8ドット固定ピッチ

$XW'0 0 0 22

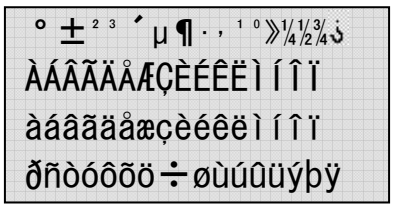
$XFｰｱｲｳｴｵｶｷｸｹｺｻｼｽｾｿ'A

$XFﾀﾁﾂﾃﾄﾅﾆﾇﾈﾉﾊﾋﾌﾍﾎﾏ'A

$XF'E0E1E2E3E4E5E6E7E8E9EAEBECEDEEEF0A

$XF'F0F1F2F3F4F5F6F7F8F9FAFBFCFDFEFF0A

（半角フォントは英語用を組み込んでいるため半角カタカナの代わりにラテン文字が表示されます。）



〔メ　モ〕

【発　行】

NEC**エレクトロニクス株式会社**

〒211-8668　神奈川県川崎市中原区下沼部1753

電話（代表）：(044)435−5111

お問い合わせ先

【ホームページ】

NEC**エレクトロニクスの情報がインターネットでご覧になれます。**

URL(**アドレス**)　http://www.necel.co.jp/

【営業関係，デバイスの技術関係お問い合わせ先】

半導体ホットライン

（電話：午前 9:00〜12:00，**午後** 1:00〜5:00）

電　話　：(044)435−9494

E-mail　：info@necel.com

【マイコン開発ツールの技術関係お問い合わせ先】

開発ツールサポートセンター

E-mail　：toolsupport-micom@ml.necel.com

【漢字表示プログラム／ボードの技術関係お問い合わせ先】　E-mail　：kanji_demo@ml.necel.com

【資料請求先】

NEC**エレクトロニクスのホームページよりダウンロードいただくか，**NEC**エレクトロニクスの販売特約店へお申し付けください。**