Universal Probe

チュートリアル

Memory Command Builder CPU 内蔵フラッシュメモリ編

Copyright © 2014 Sohwa & Sophia Technologies Inc.

No. J090970-01



目 次

注意事項3
使用上の注意4
略語・用語・記載ルール
1. CPU 内蔵フラッシュメモリ(MK70FN1M0VMJ12)6
1.1. 準備
1.2. 参考資料
1.3. 70–9
1.4. 構成10
1.5. Memory Command Builder 起動11
1.6. デバイス情報設定12
1.7. コマンド組み立て13
1.7.1. IJ-ド13
1.7.2. JTL
1.7.3. ブロックイレース15
1.7.4. チップイレース
1.8. フラッシュメモリ設定ファイル生成17
1.9. ターゲットとプローブの接続18
1.10. ARM Writer 起動19
1.11. フラッシュメモリ設定ファイル読み込み27
1.12. CPU 内蔵フラッシュメモリヘアクセス28
1.12.1. チップイレースの確認28
1.12.2. ライトの確認31
1.12.3. ブロックイレースの確認
1.12.4. リードの確認
み 訂 履 歴 / 1
製造者情報



注意事項

このたびは株式会社 Sohwa & Sophia Technologies 製「Universal Probe」をお買い上げいただき、誠にありがとう ございます。本書に記載されている注意事項などを正しくご理解のうえ、お使いいただきますようお願い申し上げます。

- 1. 本書に記載の製品及び技術で、『外国為替及び外国貿易法』に該当するものを輸出する時、又は、国外に持ち出す時は、 日本政府の許可が必要です。
- 2. 本書に記載されている製品は、一般電子機器(事務機器、通信機器、計測機器、家電製品など)に使用されることを 意図しております。特別な品質、信頼性が要求され、その故障や誤動作が直接人命を脅かしたり、人体に危害を及ぼした りする恐れのある特定用途機器(自動車・鉄道・船舶・航空・宇宙用機器、交通機器、燃焼機器、安全装置、医療機 器、インフラ機器、原子力など)には使用しないでください。もしこれらの機器でご使用になる場合は、お客様の責任のもとで ご使用ください。
- 3. 本書の内容の一部または全部を当社の文書による承諾なしに、無断で転載することは固くお断りいたします。
- 4. 本書に記載の内容は、将来予告なしに変更される場合があります。
- 5. 本書に記載の仕様は、お客様の環境、測定条件によって異なる結果が得られる場合があります。
- 6. 運用した結果の影響について、一切の責任を負いかねますのでご了承ください。
- 7. 本書に記載の「使用上のご注意」は、使用者や他者への危害と財産の損害を未然に防ぎ、安全に正しくお使いいただくための重要な注意事項です。ご使用になる前に必ずお読みください。
- 8. 本書に記載されている製品名および商品名は、各社の商標または登録商標です。



連絡先は 株式会社 Sohwa & Sophia Technologies のホームページでご確認ください。 URL > <u>http://www.ss-technologies.co.jp</u>



使用上の注意

	下記の注意を守らないと人が死亡する、または重傷を負う可能性があります。
1 強制	本製品に仕様で規定した範囲外の電源電圧を加えないでください。 範囲外の電源電圧を加えると、破損・火災の恐れがあります。
1 強制	アース端子が付いているターゲットに使用する場合は、ターゲットや周辺機器のアースを確実に接続してください。機器の故障や感電の恐れがあります。 また、ガス管にアース端子をつながないでください。火災や爆発の原因になります。
公 禁止	本製品に接続した機器を取り付けたまま持ち運ばないでください。 特にケーブルはプラグを持って抜き差ししてください。ケーブルが破損し、火災・感電の恐れがあります。
秋 山	ケーブルを取り扱う場合は次の点を守ってください。「傷つけない」「加工しない」「無理に曲げない」「ねじらない」「引っ張らない」「物を載せない」「加熱しない」「熱器具に近づけない」「濡れた手で触らない」。 これらを守らないと火災・感電の恐れがあります。 もしケーブルが破損した場合、そのケーブルの使用を中止してください。
公 禁止	雷が鳴りだしたら、電源プラグに触れないでください。感電の原因となります。 落雷により製品が破損したと思われる場合は、本製品の使用を中止してください。
公 禁止	ステープラの針、クリップなどの金属を内部に入れないでください。火災・故障の恐れがあります。
秋 山	直射日光の当たる場所、熱器具の近く、極端な高温環境、極端な低温環境、振動の激しいところ、金属 や油を含むほこりの多い場所、スパイク系のノイズが発生する場所で使用したり、放置しないでください。 また、強い衝撃を与えないでください。
父 分解禁止	分解・改造・修理しないでください。火災・感電の恐れがあります。
水濡れ禁止	風呂場やコップの近くなど、液体のある場所、湿気の多い場所では使用しないでください。 感電する恐れがあります。 液体が本製品内部に入った場合はすぐに電源を切り、使用を中止してください。
注意	通電中の本製品に長時間触れていると低温やけどになる恐れがあります。 また、本製品を布団などで覆った状態で使用しないでください。
プラグを抜く	もし、異常なにおい・異常な音・発煙・発火した場合、または落としたり、強い衝撃を与えたりして破損、破損した恐れのある場合は、すぐに電源を切ってください。そのまま使うと重大な事故を起こす可能性がありますので、使用を中止してください。



略語・用語・記載ルール

本書で使用する略語・用語や記載ルールについて説明します。

- 数値について … 特に記載がない限り、数値はすべてプラスの値とします。
- K(大文字) ···· 2¹⁰=1024 を表します。(例:16K=16384)
- k(小文字) ···· 1000 を表します。(例:1kHz=1000Hz)
- [xxxxx] … xxxxx というウィンドウタイトルを示します。
- <xxxxx> ··· xxxxxというウィンドウ内の項目名を示します。

本書で使用する注釈・注意点などについては Figure 1 の通りです。



Figure 1

略語・用語の解説は Table 1 の通りです。

	Table 1
略語・用語	説明
フラッシュメモリ	Flash メモリ、EEPROM などの総称。
プローブ	Universal Probe または EJSCATT 本体のこと。
ターゲット	Universal Probe または EJSCATT によって制御、計測する対象のこと。



1. CPU 内蔵フラッシュメモリ(MK70FN1M0VMJ12)

Memory Command Builder を使用して、TWR-K70F120M-KIT に搭載された CPU(MK70FN1M0VMJ12)の内蔵フラ ッシュメモリへアクセスするためのコマンドを作成する方法を説明します。また、最終的に CPU 内蔵フラッシュメモリへアクセスするまで の手順も記載します。

1.1. 準備

使用する機材、ソフトウェア、ファイルを以下に示します。

●機材

- プローブ: Universal Probe ×1個

または





EJSCATT ×1個

- JTAG ケーブル: J-Link 19-pin Cortex-M Adapter ×1本



- USB ケーブル ×2本 (miniB-A コネクタケーブル)





- PC ×1 台 (ソフトウェアをあらかじめインストールしてください。)



- ターゲット:

TWR-K70F120M-KIT ×1台





例として、TWR-K70F120M-KIT (Freescale Semiconductor 社)を使用します。 TWR-K70F120M-KIT には、Kinetis K70 ファミリ CPU: MK70FN1M0VMJ12(Cortex-M4)が搭載されて います。

●ソフトウェア

- Memory Command Builder
- ARM Writer または WATCHPOINT(ARM または Cortex コア用)



ARM Writer は Universal Probe 用、WATCHPOINT は EJSCATT 用のソフトウェアとなります。

●インストール方法(Memory Command Builder, ARM Writer)については、 「<u>Universal Probe インストールマニュアル</u>」を参照してください。



Memory Command Builderの詳細については、
 「<u>Universal Probe ソフトウェアユーザーズマニュアル Memory Command Builder</u>」を参照してください。

●ARM Writer の詳細については、 「Universal Probe ソフトウェアユーザーズマニュアル ARM Writer」を参照してください。

• WATCHPOINT の詳細については、 付属 CD-ROM またはメニュー $\rightarrow \land$ ルプ を参照してください。



1.2. 参考資料

参考資料を以下に示します。必要に応じて参照してください。

●コマンド組み立てするための参考資料

- K70 Sub-Family Reference Manual (Rev.2 Dec 2011) (Freescale Semiconductor 社)

K70 Sub-Family Reference Manual はデバイスメーカーの Web からダウンロードしてください。

●本チュートリアルで作成するファイル

以下のファイルは、本チュートリアルを進めて行く事で完成する予定のファイルです。

- コマンド組み立てリストファイル (Memory Command Builder 用) K70_1M0_ProgramFlash.txt
- フラッシュメモリ設定ファイル (ARM Writer or WATCHPOINT 用) K70_1M0_ProgramFlash.fsh



 \rightarrow

本チュートリアルで作成するファイルは、<u>Universal Probe Web サイト</u>の「チュートリアル」からダウンロードできます。 サンプルファイルは、「Memory Command Builder CPU 内蔵フラッシュメモリ編」覧にある"sample.zip"ファイルに 含まれています。

●本チュートリアルで必要であるファイル

以下のファイルは、本チュートリアルを進めて行く上で必要であるファイルです。

 ウォッチドックタイマをオフするためのオブジェクトファイル (ARM Writer, WATCHPOINT 共用) Kinetis_WDOG_Disable.srec



本チュートリアルで必要なファイルは、<u>Universal Probe Webサイト</u>の「チュートリアル」からダウンロードできます。 サンプルファイルは、「Memory Command Builder CPU 内蔵フラッシュメモリ編」覧にある"sample.zip"ファイルに 含まれています。



1.3. 70-

CPU 内蔵フラッシュメモリヘアクセスするためのフローを以下に示します。



Figure 2



1.4. 構成

CPU 内蔵フラッシュメモリへのアクセスするための構成を以下に示します。 Universal Probe, EJSCATT いずれかを使用する方法があります。



Figure 3



以降の詳細ではUniversal ProbeとARM Writerの組合せを使用した説明のみを記載していますが、EJSCATT と WATCHPOINT の組合せも基本的に操作は同様です。 お手持ちの環境・機材に置き換えてお読みください。



1.5. Memory Command Builder 起動

Memory Command Builder を起動します。 以下①、②いずれかの方法で実行してください。

- ① 「デスクトップ」にある Memory Command Builder のアイコンをダブルクリックします。
- ② 「スタート」ボタンをクリックし、「すべてのプログラム」→「Memory Command Builder」→「Memory Command Builder」をクリックします。



😁 Memory Command Builder	
ファイル(F) 編集(E) ヘルプ(H)	
: 🔁 📂 🛃 📀 🚑 🞑 🚌 🗙 🔺 🔻 🗈 🛍	
デバイス情報: デバイス情報を設定してください。	設定
リード ライト ブロックイレース チップイレース	
コマンドー質	
	-

Figure 5



Memory Command Builder の操作方法、コマンドの仕様等は 「Universal Probe ソフトウェアユーザーズマニュアル Memory Command Builder」を参照してください。



1.6. デバイス情報設定

デバイス情報を設定します。以下のように設定してください。 完了したら、設定 ボタンを押してください。

項曰名	設定値
デバイス名	K70 1M0 ProgramFlash
接続タイプ	JTAG
アドレス	0x0000000
範囲長	0x100000
データアライメント	0x10
ワークメモリアドレス	0x2000000
ワークメモリ範囲長	0x2000

Figure 6

Table 2

デバイス名	メモリデバイスの名前を設定します。任意の名前で結構です。
接続タイプ	プローブとターゲット間の接続インターフェースを設定します。ここでは JTAG を選択してください。
アドレス	対象デバイスの開始アドレスを設定します。ここでは 0x0000000 を入力してください。
範囲長	対象デバイスのアドレス範囲長を設定します。ここでは 0x100000 を入力してください。
データアライメント	リード/ライト用プログラムの処理するデータのアライメントを設定します。
	ここでは1を入力してください。
ワークメモリアドレス	ワークメモリの先頭アドレスを設定します。ここでは 0x2000000 を入力してください。
ワークメモリ範囲長	ワークメモリの範囲長を設定します。ここでは 0x2000 を入力してください。



上記の設定は TWR-K70F120M-KIT の仕様に合わせて設定しています。



1.7. コマンド組み立て

本構成における、CPU内蔵フラッシュメモリへアクセスするためのコマンド組み立て方法(リード、ライト、ブロックイレース、チップイレース)を以下に示します。

1.7.1. リード

CPU 内蔵フラッシュメモリ リード処理のコマンド組み立てフローと詳細を以下に示します。



Figure 7

K70_1M0_ProgramFlash.txtを Memory Command Builder で開くと Figure 7のフローの詳細が確認できます。

* 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2			
		aXAL	
コマンド一覧 ADD AND CALL DEC DEFINE END GOTO IF INC LABEL LSHIFT NOT OR RBIT READ REIT READ RET RSHIFT SBIT SET SUB WAIT	1.CALL 呼び出し先: A:0x20000081 2.READ 転送先: REG1 転送元: REG0 データ長: REG2 3.END	^A	
	·		
	Figure 8		

Universal Probe チュートリアル - Memory Command Builder CPU 内蔵フラッシュメモリ編



1.7.2. ライト

CPU 内蔵フラッシュメモリ ライト処理のコマンド組み立てフローと詳細を以下に示します。



Figure 9

K70_1M0_ProgramFlash.txtを Memory Command Builder で開くと Figure 9 のフローの詳細が確認できます。

デバイス情報: K70	1M0_ProgramFlash	
コマンド一覧	5	
ADD		
AND		
CALL		
DEC	1.CALL	
END	呼び出し先: A:0x20000081	
GOTO	2.CALL	
IF		
INC	呼び出し先: Check_Flash_Memory	
LABEL	3.CALL	
NOT		
OR	呼び出し先: Program_Phrase	
RBIT	4.CALL	
READ	呼び出した: Cache Invalidate	
RET	5 END	
SBIT	5.610	
SET		
SUB	6.LABEL	
WAIT	Program Phrase command	-
WRITE	< III	→
	Figure 10	
	5	
マの加田は 女士	*次約 + 1/70 Cub Family Deference Marriel た公共に	レテいナオ
この処理は、参考	5貝科:K/U SUD-Family Reference Manual を参考に	しています。



1.7.3. ブロックイレース

CPU 内蔵フラッシュメモリブロックイレース処理のコマンド組み立てフローと詳細を以下に示します。



Figure 11

K70_1M0_ProgramFlash.txtを Memory Command Builder で開くと Figure 11のフローの詳細が確認できます。

ファイル(E) 編集(E) ヘルブ(H) デバイス情報: K70_1M0_ProgramFlash 設定 リード ライト ブロックイレース チップイレース コマンドー覧 ADD AND CALL DEC DEFINE END GOTO IF INC LABEL LSHIFT NOT OR RET READ RET
アバイス情報: K70_1M0_ProgramFlash 設定 アバイス情報: K70_1M0_ProgramFlash 設定 リード ライト プロックイレース チップイレース コマンドー覧 ADD ADD ADD AND CALL 「 「 DEC 1.CALL 「 「 DEFINE FULL先: A:0x20000081 「 「 GOTO I.CALL 「 町び出し先: A:0x20000081 「 INC 呼び出し先: Check_Flash_Memory I.CALL 「 ISHIFT NOT 「 呼び出し先: Check_Flash_Memory I.CALL RBIT RBIT 4.CALL RET 「 「 び出し先: Cache_Invalidate
デバイス情報: K70_1M0_ProgramFlash 設定 リード ライト ブロックイレース チップイレース ADD AND CALL DEC LALL DEFINE END GOTO IF INC LABEL LSHIFT NOT OR REIT READ RET
リード ライト プロックイレース チップイレース コマンド一覧 ADD AND CALL DEC 1.CALL 「 DEFINE FUTULLE: A:0x20000081 E GOTO 2.CALL 「 IF NC 呼び出し先: A:0x20000081 E LABEL 3.CALL 「 ISHIFT NOT 呼び出し先: Check_Flash_Memory NOT 呼び出し先: Erase_Flash_Sector OR 4.CALL RET 呼び出し先: Cache_Invalidate
コマンドー覧 ADD AND CALL DEC DEC DEFINE END GOTO I.CALL IF END GOTO I.CALL IF INC IF INC LABEL LSHIFT NOT OR RBIT READ 展T T Pび出し先: A:0x20000081 I CALL IF IVULし先: A:0x20000081 I I I I I I I I I I I I I
コマンドー覧 ADD AND CALL DEC DEC DEFINE END GOTO I.CALL IF END GOTO I.CALL IF INC IVC IF INC INC IABEL LSHIFT NOT OR RBIT READ RET MULD: Cache_Invalidate
ADD AND CALL DEC 1.CALL 単び出し先: A:0x20000081 END 呼び出し先: A:0x20000081 GOTO 2.CALL IF IC ICALL IF 3.CALL IF IC ICALL IF IC ICALL ISHIFT ICALL ISHIFT ICALL ISHIFT ICALL ISHIFT ICALL ISHIFT ICALL ISHIFT ICALL ISHIFT ICALL ISHIC ICALL ISH
AND CALL DEC 1.CALL 単び出し先: A:0x20000081 END 呼び出し先: A:0x20000081 GOTO 2.CALL IF IC 単び出し先: Check_Flash_Memory LABEL 3.CALL IC
CALL DEC 1.CALL DEFINE END GOTO 2.CALL IF INC LABEL 3.CALL SHIFT NOT OR REIT READ 呼び出し先: Cache_Invalidate Flash_Memory ABEL 4.CALL 呼び出し先: Erase_Flash_Sector Flash_Sector Flash_Memory Flash_Sector Fla
DEC 1.CALL ACL ACL ACL ACL ACL ACL ACL ACL ACL
DEFINE END 呼び出し先: A:0x20000081 GOTO 2.CALL IF INC 呼び出し先: Check_Flash_Memory LABEL 3.CALL SHIFT NOT 呼び出し先: Erase_Flash_Sector OR 4.CALL READ 呼び出し先: Cache_Invalidate
END 呼び出し先: A:0x20000081 GOTO 2.CALL IF IIC 呼び出し先: Check_Flash_Memory LABEL 3.CALL NOT 呼び出し先: Erase_Flash_Sector OR 4.CALL READ 呼び出し先: Cache_Invalidate
GOTO 2.CALL IF INC 呼び出し先: Check_Flash_Memory LABEL 3.CALL SHIFT NOT 呼び出し先: Erase_Flash_Sector OR 4.CALL READ FT Fび出し先: Cache_Invalidate
IF INC 呼び出し先: Check_Flash_Memory LABEL 3.CALL 3.CALL USHIFT 4.CALL Flash_Sector OR 4.CALL 4.CALL READ 呼び出し先: Cache_Invalidate
INC 呼び出し先: Check_Flash_Memory LABEL 3.CALL 3.CALL NOT 呼び出し先: Erase_Flash_Sector OR 4.CALL 4.CALL READ 呼び出し先: Cache_Invalidate
LABEL LSHIFT 3.CALL ISHIFT 9 OR 呼び出し先: Erase_Flash_Sector OR 4.CALL READ 呼び出し先: Cache_Invalidate
LSHIFT NOT 呼び出し先: Erase_Flash_Sector OR 4.CALL READ 呼び出し先: Cache_Invalidate
NOT 呼び出し先: Erase_Flash_Sector OR RBIT 4.CALL READ 呼び出し先: Cache_Invalidate
OR
RBIT +.CALL READ 呼び出し先: Cache_Invalidate
READ RET 呼び出し先: Cache_Invalidate
RET Promote, Cache_invalidate
RSHIFT 5.END
SBIT
SET
SUB 6.LABEL
TIAN
WRITE // // // // // // // // // // // // //
Figure 12





この処理は、参考資料: K70 Sub-Family Reference Manual を参考にしています。



1.7.4. チップイレース

CPU 内蔵フラッシュメモリチップイレース処理のコマンド組み立てフローと詳細を以下に示します。



Figure 13

K70_1M0_ProgramFlash.txtを Memory Command Builder で開くと Figure 13のフローの詳細が確認できます。

😁 Memory Command Builder	- K70_1M0_ProgramFlash.TXT	
ファイル(E) 編集(E) ヘルプ((<u>H</u>)	
i 🔁 💕 🖬 🛞 🎒 🖾 📖 🕽	X 🐟 🔹 📴 🛍	5
デバイス情報: K70_1M0_Pr	rogramFlash	設定
リード ライト ブロックイレ		
コマンド一覧		
ADD		
AND		
DEC	1.CALL	A
DEFINE		E
END	呼び出し先: A:0x20000081	
GOTO	2.CALL	
IF		
LABEL	呼び出し先: Check_Flash_Memory	
LSHIFT	3.CALL	
NOT	呼び出し先: Erase Flash Block	
OR	4.CALL	
RBIT		
READ	呼び出し先: Cache_Invalidate	
RSHIFT	5.END	
SBIT		
SET		
SUB	6.LABEL	
WRITE		×
	Figure 14	



この処理は、参考資料: K70 Sub-Family Reference Manual を参考にしています。



1.8. フラッシュメモリ設定ファイル生成

ARM Writer / WATCHPOINT で使用するためのフラッシュメモリ設定ファイル(.fsh)を生成します。

メニューバーから「ファイル」→「フラッシュメモリ設定ファイル出力」を選択し、[フラッシュメモリ設定ファイル作成情報]ウィンドウを表示 してください。以降、①~④の手順に沿って設定してください。

○ フラッシュメモリ設定ファイル作成情報	X
命令セット Thumb-2 ・	エンディアン Little
出力ファイル D:XK70_1M0_ProgramElach_fch	
	作成開始 キャンセル

Figure 15

① **命令セット**

命令セットを選択します。ここでは Thumb-2 を選択してください。

② エンディアン

エンディアンを選択します。ここでは Little を選択してください。

③ 出力ファイル

出力するフラッシュメモリ設定ファイルを設定してください(拡張子: *.fsh)。 ファイル名、保存場所は任意です。

④ 作成開始

作成開始ボタンを押してください。フラッシュメモリ設定ファイルが生成されます。

	以下のダイアログが表示される場合は、コマンドの記述に誤りがあります。再度確認してください。 例)ラベル名、CALL するラベル名などが間違っている場合に表示されます。
	警告メッセージ
(I) INFO	フラッシュメモリファイルが作成できませんでした。
	ОК



1.9. ターゲットとプローブの接続

ターゲット: TWR-K70F120M-KITと プローブ: Universal Probeの接続例を以下に示します。 以下の接続手順に沿って、プローブとターゲットを接続してください。

- 1) プローブとターゲットを JTAG ケーブルで接続します。
- 2) プローブを USB ケーブルで接続し、プローブの電源を入れます。
- 3) ターゲットを USB ケーブルで接続し、ターゲットの電源を入れます。



Figure 16



1.10. ARM Writer 起動

ARM Writer を起動します。

以下①、②いずれかの方法で実行してください。

- ① デスクトップにある Universal Probe ARM Writer のアイコンをダブルクリックします。
- ② 「スタート」ボタンをクリックし、「すべてのプログラム」→「Universal Probe」→「Universal Probe ARM Writer」をクリックします。



Figure 17



ARM Writer で説明を行いますが、WATCHPOINT での操作も基本的に同様です。 お手持ちの環境・機材に置き換えてお読みください。

以下のウィンドウが起動します。

7711/(F)	表示(V) 実行(X) ^//	л° (Н)		
	P 💕 🖳			
[<i>ν</i> ¯,*1			11.	
		Figure 18		



以下の手順で[プロジェクトの新規作成]ダイアログを開いてください。



プロジェクトの新規作成ダイアログが起動します。<プロジェクト名>と<位置>(保存先)を設定してください。 設定後、プローブ ボタンを押してください。

5	『ロジェクトの新規作成	×
	7 [°] ፲››ェ小名 (Ŋ): K70	OK キャンセル
	ゲフローフ アローフ 運択されていません ターケット: 選択されていません ライセンス: 選択されていません	<u>ז״ב−ז״נפ</u>)
	位置 (L): D:¥K70.armwpj	参照(<u>B</u>)

Figure 19

[プローブ選択]ダイアログが起動します。シリアル番号を選択して、 接続 ボタンを押してください。

		プローブ選択		×	
		·川アル番号	ステータス 技術主要的	接続①	
		HM54000075		更新(U) ライセンス登録(L)	
			Figure 20		
)	ライセン	ス登録を行っていない場合	は、ARM Writer のマニュ	アルを参照して登録を行ってください。	



[プロジェクトの新規作成]ダイアログが以下のようになります。 OK ボタンを押してください。

プロジェクトの新規作成	×
7 [°] ロジェクト名(<u>N</u>): K70 - 7 [°] ローフ [°] ワ [°] ローフ [°] 種類: UniPro [S/N HM540000075] ターケット: ARM ライセンス: Available	ок ++>tu
位置(L): D:¥K70.armwpj	参照(<u>R</u>)

Figure 21

[CPU の選択]ダイアログが起動します。Cortex-M を選択してください。 選択後、 OK ボタンを押してください。

CPUの選択		×
CPU選択(S)	Cortex-M	_
	ОК	

Figure 22



[デバイスの初期設定]ダイアログが起動します。以下のように設定してください。 設定後、 OK ボタンを押してください。

デバイスの初其	朋設定
メーカー(<u>M</u>)	Freescale
シリース*(<u>S</u>)	Kinetis_K70 💌
デバイス(<u>D</u>)	Freescale_Kinetis_K70_MK70FN1M0VMJ12
ファイル	MK70FN1M0VMJ12_Setupfile.ini
	参照(目)
─接続方法 ● JTA(の選択 G(J) C SWD(W)
⊂%=9 %reg ⊙ VTre	源電圧取得力法の運択 1端子からターグットの電源電圧を取得する(V)
⊖ VTre	f端子を無視する(<u>N</u>)
JTA 3.3V 故郎	AG通信/SWD通信を3.3Vの信号レベルで行います。 Vより電圧が低いターケットでは設定しないで下さい。ターケットシステムが 章する可能性があります。
	ОК

Figure 23



[JTAG の設定]ダイアログが起動します。以下のデフォルト設定のままにして OK ボタンを押してください。

JTAGの設定	×
「JTAGの加ック周波数の指定	
自動設定	•
低速加ックの周波数(1~ 1000KH	z) 500 × KHz
自動設定: ターゲットとの正常な通信 低い周波数を設定します	が確認できた最高周波数の一段階
◎ JTAG がデイシーチェーン接続されて	(แก่สมา
│ ○ JTAG がデイシーチェーン接続されて	115
「Shift IR の設定	
総ビット数 4 🔹	ビット位置 0
「ハイハペスレシジスタの設定―――	
総七字外数 1	ビット位置 0
ОК	キャンセル

Figure 24

[プローブ起動時の設定]ダイアログが起動します。以下のデフォルト設定のままにして OK ボタンを押してください。

RICK212H19 O(B)			
リセット信号をアクティフキこする時 (リセットのハ°ルス報	镅(P) 畐)(0 ~ 65535)	ms	
リセット信号をネケート後待つ時	間(N) (0 ~ 65535)	ms	
ARM Writer起動時の動作語	没定(₩)		
ターグットをリセットして起動しま	हे जे	•	
▼ コプロセッサに対してアクセス> (MMU、キャッシュ制御?	をする(<u>C</u>) を行います。)		
メモリアクセスエリア設定(<u>A</u>)	メモリアクセス経路の設定は、メモリアク から行うことができます。	セスエリア設定	
	オプション設定は、弊社からの指示	など特別な	
オプジョン設定(<u>0</u>)	場合を除き、操作しないようにし	(1/280%	

Figure 25



[CPU 動作モードの設定]ダイアログが起動します。以下のデフォルト設定のままにして OK ボタンを押してください。

CPU動作モードの設定	
「エンディアンの選択 ――	
○ ビックエンディアン(B)	• <u>IHUID7*17X()</u>
┌─メモリアクセス時にアボート検出を行う	
⊙ O <u>F</u> F	C 0 <u>N</u>
	ОК

Figure 26

設定が完了すると、以下のように、GO、RESET ボタン等が有効になった状態になります。

🚺 Universal Probe - ARM Writer (Freescale_Kinetis_K70_MK70FN1M 💷 💷 💌
ファイル(F) 表示(V) リソース(R) 実行(G) ウィンドウ(W) ヘルプ(H)
BREAK

Figure 27



次に、あらかじめ CPU のウォッチドックタイマをオフにするためのプログラムを内蔵 RAM にダウンロードします。 以下の設定を行ってください。

以下のメニューを選択し、[ダウンロード設定]ダイアログを表示します。

ツールボタン	メニューバーの操作	ショートカットキー
	リソース → ダウンロード	

ダイアログが表示されたら、 追加 ボタンを押して、Kinetis_WDOG_Disable.srec ファイルを開いてください。

**ウンロート*設定			×
モシ [*] ュールリスト (M):	フォーマット(F):	 ▼ アトシス変換設定 ▼ デバックファイル設定 アトシス変換設定 ハースパス置換設定 ハースパス置換設定 (E) 示する(J) 示する(G) 	追加(A) 削除(D) タウンロートで(L) 開じる タウンロート前に チップイレースを 自動実行する (U)

Figure 28



以下のようにファイルが<モジュールリスト>に登録されたら、 ダウンロード ボタンを押してください。 ダウンロードは1秒未満で終了します。 ダウンロードが終了したら、 閉じる ボタンを押してください。



Figure 29



1.11. フラッシュメモリ設定ファイル読み込み

作成したフラッシュメモリ設定ファイル(.fsh)を読み込みます。 以下のメニューを選択し、[フラッシュメモリ設定]ダイアログを表示してください。

ツールボタン	メニューバーの操作	ショートカットキー
	リソース → フラッシュメモリ → 設定ファイル	

[フラッシュメモリ設定]ダイアログが起動します。設定ファイルの保存/読み込みタブを選択し、ファイルを開く」ボタンを押して、作成したフラッシュメモリ設定ファイル(.fsh)を読み込んでください。 <設定ファイル名>に指定したファイルが表示されます。

フラッシュメモリ設定	x
フラッシュメモリのワリア フラッシュメモリのフィルアップ その他の設定 デハドイス設定 ワークメモリ フロック情報 設定ファイルの(保存/読み込み) フラッシュメモリ設定ファイル ファルを開く(0) 上書き(保存(ビ)) ション・ジョン・ジョン・ジョン・ジョン・ジョン・ジョン・ジョン・ジョン・ジョン・ジ	<u> 道加(A)</u> 削除(D) 全削除(L) 閉じる(C) タ*ウンロート*(Y)
R;0x0000000 R;0x000fffff WM K70_1M0_ProgramFlash R0D0 16Byte	e R;0x20002000
□ ダウンロード時設定を無視する(<u>I</u>) ワークメモリ R;0×20000000 -	R;0x20001fff

Figure 30



1.12. CPU 内蔵フラッシュメモリへアクセス

Memory Command Builder で組み立てた リード、ライト、ブロックイレース、チップイレース コマンドの動作確認を行います。

1.12.1. チップイレースの確認

フラッシュメモリをクリアし、クリアされたかを確認することによって、Memory Command Builder で組み立てた"チップイレース"の動作を確認します。

フラッシュメモリのクリア タブを選択し、以下のように<登録デバイスを全てクリア>にチェックを入れた状態で、開始 ボタンを押してください。

フラッシュメモリ設定	X
デバイス設定 ワーりメモリ フ`ロック'情報 設定ファイルの保存/読み込み フラッシュメモリのフィルアッフ° その他の設定 'リア範囲の指定 ()) * 澄蒜デバイスを全てクリフ(()) ()) 'リアドレス(B) 0×00000000 'アト*レス領域の設定 ()) ? フト*レスを含むデバイス全体をクリア(N) ? ドレスを含むブロックをクリア(B) 'デバイス全体をクリフ	<u>追加(A)</u> 削除(<u>D</u>) 全削除(L) 閉じる(<u>C</u>)
	ጵ°ዕንロ∽ኑ°(₩) アッフ°ロ∽ኑ°(Ⴞ)
▼ R;0×00000000 R;0×000fffff WM K70_1M0_ProgramFlash R0D0 16Byte -	R;0×20002000
□ ダウンロード時設定を無視する(<u>I</u>) ワークメモリ R;0x20000000 - I	R;0x20001fff

Figure 31



クリア処理中は以下のダイアログが表示されます。

チップ消去中です		
ፖኑን	0×0000000	
経過時間	2秒	
	キャンセル	
L		

Figure 32



本構成の場合、1秒以内に完了してしまいますので、ダイアログはほとんど表示されません。

クリア処理が終了すると、[フラッシュメモリ設定]ダイアログに操作結果が表示されます。

// フラ┉シュメモリクリアを実行します // All Clear : FMCLEAR ALL CLEAR : BLANK CHECK NOT SUPPORT // コマンド実行をします // 終了しました

Figure 33

指定範囲が正常クリアされたかは、ダンプウィンドウで確認します。 [フラッシュメモリ設定]ダイアログを 閉じる ボタンで閉じてください。 次に、以下のボタンもしくはメニューでダンプウィンドウを起動してください。

ツールボタン	メニューバーの操作	ショートカットキー
MEM	表示 → ダンプウィンドウ	



ダンプウィンドウ起動後、 <アドレス>に 0x00000000, <範囲長>に 0x100 を入力して Enter キーを押してください。 以下のように全て 0xFF がリードでき、クリア(チップイレース)されていることを確認します。

💷 ቃ"ንፓ° ዕብንኮ	°ウ1 0x0	0000	000	- 0x	0000	000f	f (0x	100))								(1 🔀	3
7ドレス(<u>S</u>): [0	×000000	00			•	O f	冬了ア	ドレス	(<u>E</u>)	0×01	0000	Dff			-	アクセスサイス (<u>A</u>)	 ロングワード 	-]	
データ(<u>D</u>): [0	lxff			X	✓	• i	範囲	€(<u>L</u>)		0×10	00				•	Q ♀ □ 彫	新しない(山)			
Address	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F	ASCII			4 7
0×00000000) ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff				
0x000000020) ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	i			
0×00000030) ff) ff	ff ff	ff ff	ff ff	ff ff	ff ff	ff ff	ff ff	ff ff	ff ff										
0×00000050) ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff			iii	
0×00000060 0×00000070) ff) ff	ff ff	ff ff	ff ff	ff ff	ff ff	ff ff	ff ff	ff ff	†† ff										
0×00000080) ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff			i	
0×00000090 0×0000000a0	J ††) ff	†† ff	tt ff	tt ff	tt ff	tt ff	tt ff	†† ff	tt ff	†† ff	†† f f	†† f f	†† f f	†† f f	†† ff	†† ff				
0×000000b0) ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff				
0×000000d0) ††) ff	tt ff	tt ff	tt ff	tt ff	tt ff	tt ff	tt ff	tt ff	†† ff										
0x000000e0) ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff				
		†† 	††	††	††	††	††	††	††	††	††	††	††	††	††	†† }			•••• ••	-

Figure 34



1.12.2. ライトの確認

フラッシュメモリヘデータを書込み、正しいデータか確認することによって、Memory Command Builder で組み立てた"ライト"の動作を確認します。

[フラッシュメモリ設定]ダイアログを起動してください。

フラッシュメモリのフィルアップ タブを選択し、以下を参照して<開始アドレス>、<範囲長>、<フィルアップデータ>を設定してください。設定が完了したら、開始 ボタンを押してください。

	יעכי	シュメモリ設定						X
	Ŧ	バイス設定 フラッシュメモリのクリア	ワークメモリ フラ・	フロック情報 ッシュメモリの	報 設定ファ フィルアッフ°	ァイルの保存/ その他	(読み込み) の設定)	追加(<u>A</u>) 削除(D)
	7	ドレス範囲の指定 ≈開始アドレス(<u>T</u>)	0×00000000	•		開始(<u>S</u>)		全削除(L)
,)終了外しみ(<u>E</u>) 「範囲長(<u>R</u>)	0×000000ff 0×2000	_	1			<u>閉じる(C)</u>
	-1 - 74	ルアッフ [®] テ [®] ータ(1ハ [®] イ 0×99	N(<u>B)</u>				_	
								<u>\$`</u> ウンロード(₩) ア₀プロード(U)
		R;0×00000000	R;0×000fffff	WM	K70_1M0_Prog	ramFlash	RODO 16Byte	e R;0x20002000
	ا ا	ウンロード時設定を	無視する(<u>I</u>)		ŋ	ታኦቺሃ <mark>R;0</mark> ×	20000000 -	R;0×20001fff



フィルアップ中は以下のダイアログが表示されます。







本構成の場合、数秒で完了します。

フィルアップ処理が終了すると、[フラッシュメモリ設定]ダイアログに操作結果が表示されます。

// フラッシュメモリフィルを実行します // (0x00000000 - 0x99) コアンド実行をします	
// 終了しました	

Figure 37

[フラッシュメモリ設定]ダイアログを 閉じる ボタンで閉じてください。 正常にフィルアップされたか、ダンプウィンドウを追加起動して、以下の3パターンを確認してください。

💷 ቃግንም ታብን	›ドウ1 (0x00	000	000	- 0x	0000	000fi	F (Ox	100))										×
7ኑኄレス(<u>S</u>):	0×00	0000	00			•	○ 終了アドレス(E) 0×000000ff アクセスサイスで(A):										: ロングワート*]	•		
テ°─タ(<u>D</u>):	0×99				X	V	Θi	範囲	畏(L)		0×1	00				-	Q 🤉 🗆 更	新しない(山)		
Address		+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F	ASCII		
0×000000	00	<u>99</u>	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	<mark>-</mark>)劒劒劒劒劒	劔劔	
0×000000	10	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	刻刻刻刻刻刻	劔劔	
0×000000	20	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	劔劔劔劔劔劔	劔劔	
0×000000	30	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	劔劔劔劔劔劔	劔劔	
0×0000004	40	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	劒劒劒劒劒劒劒	劔劔	
0×000000!	50	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	劔劔劔劔劔劔	劔劔	
0×000000	60	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	劔劔劔劔劔劔	劔劔	
0×000000	70	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	叙叙叙叙叙	劔劔	
0×000000	80	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	剑剑剑剑剑剑	劒劒	
0×000000	90	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	剑剑剑剑剑剑	劒劒	
0×000000;	a0	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	动动动动动动	劒劒	
0×000000	b0	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	动动动动动动	劒劒	
0×000000	cÛ	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	动动动动动动	創刻	
0×000000	dŨ	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	动动动动动动	刻刻	
0×000000	eÛ	99	99	99	<u>99</u>	99	99	99	<u>99</u>	99	99	99	<u>99</u>	99	99	<u>99</u>	99	动动动动动动	(新)	
0x000000	fÕ	99	99	99	99	99	99	99	<u>99</u>	<u>99</u>	99	<u>99</u>	<u>99</u>	<u>99</u>	<u>99</u>	<u>99</u>	99	剑剑剑剑剑剑剑	劒劒	
		Ľ.																~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~		Ľ
	P	1															•	L	<u> </u>	1



💷 ቃ [°] ንፓ° ዕብ	ንኑ*92 (0x00	000	f80 -	- 0x0	0000	107f	(0x	100])										×
アドレス(<u>S</u>):	0×000)00f8	0			•	0 8	冬了 7	ドレス	(<u>E</u>)	0×0	0001)7f			T	アクセスサイス (<u>A</u>):	ロングワード	•	
データ(<u>D</u>):	0×99				X	✓	• 1	御田	€(<u>L</u>)		0×1	00				•	Q 💡 🗆 更新	所しない(<u>U</u>)		
Address		+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F	ASCII		*
0×00000f	80	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	<mark>.</mark>)),,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	驗劔	
0x00000f	90	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	刻刻刻刻刻	別劔劔	
0x00000t	a0	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	, MMMMMM	题题	
0x00000t	b0	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99		则 劍 劒 劒	
UXUUUUUt	CU I	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99		<u>影影</u>	
UXUUUUUt	dŬ	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99			
UXUUUUUt	eU	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99		りまた。 1997年1月11日 1997年1月11日	
UXUUUUUT	100	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	, MMMMMMM Anananana	NMMM) NANAN	
0×000010	100	33	33	99	99	99	99	99	33	99	99	99	99	99	99	33	99	andndndndnd	ŊĦŊĦŊ nenen	
0.000010	10	33	99	33	99	99	99	33	99	99	99	99	33	99	99	33	99	ananananana	み553み553 カ会の会の	
0x000010	120	99	00	00 00	99	99	00 00	99	00	00 00	99	99	00 00	99	99	00 00	99		8558555 NGNGN	
0,000010	140	00 00	00	00 00	00	00	00	00 00	00	00	00	00 00	00 00	00 00	00 QQ	00 00	00 00		9.009.009 NGNGN	
0,000010	150	99	99	99	99	99	99	99	99	99	99	99	gg	99	99	99	99	මා නා නා නා නා නා න	9229229 161161	
	0.00	<u>99</u>	99	99	99	99	99	99	99	99	99	99	<u>99</u>	99	99	99	99	බ බ බ බ බ බ බ බ	ग केरा केरा रा केरा केरा	
	170	<u>99</u>	<u>99</u>	99	99	99	99	<u>99</u>	<u>99</u>	99	99	<u>99</u>	<u>99</u>	<u>99</u>	<u>99</u>	<u>99</u>	<u>ăă</u>	刷刷刷刷刷刷	日前前	
		Ě	~~	~~	~~	~~	~~	~~	~~	~~	~~	~~	~~	~~	~~	~~	~~	101000000000000000000000000000000000000	en en e	

Figure 39

💷 ቃ ኦንም ዕብኦ	۲°93 0>	×00	001	f80 -	• 0x0	0000	207f	f (0x	100))									
アドレス(<u>S</u>):	0×0000	1f80	0			•	C á	冬了 7	ドレス	(<u>E</u>)	0×01	00020)7f			-	アクセスサイス (<u>A</u>):	: ロンがワート [:] ・	
テ°-妳(<u>D</u>):	0×99		_	_	X	✓	• 1	範囲長	€(<u>L</u>)		0×11	00				•	Q 🖓 🗆 更新	新しない(世)	
Address		+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F	ASCII	A
0x00001fs 0x00001fs 0x00001fs 0x00001fs 0x00001fc		39 39 39 39 39	99 99 99 99 99	99 99 99 99 99	99 99 99 99 99	99 99 99 99 99	99 99 99 99 99	99 99 99 99 99	99 99 99 99 99	99 99 99 99 99	99 99 99 99 99	99 99 99 99 99	99 99 99 99 99	99 99 99 99 99	99 99 99 99 99	99 99 99 99 99	99 99 99 99	2.000000000000000000000000000000000000	1
0x00001fe 0x00001fe 0x00001ff 0x0000200 0x0000201	10 1 10 1 10 1 10 1	99 99 99 6 f 6 f	99 99 99 ff ff	99 99 99 ff ff	99 99 99 ff ff	99 99 99 ff ff	99 99 99 ff ff	99 99 99 ff ff	99 99 99 ff ff	99 99 99 ff ff	99 99 99 ff ff	99 99 99 ff ff	99 99 99 ff ff	99 99 99 ff ff	99 99 99 ff ff	99 99 99 ff ff	99 99 99 ff ff	财财财财财财财财财 叙叙叙叙叙叙叙叙叙叙 ●●●●●●●●●●●●●●●●●●	
0×0000202 0×0000203 0×0000204 0×0000205 0×0000206 0×0000207	20 1 30 1 30 1 50 1 30 1 70 1	f	†† ff ff ff ff ff	++ ff ff ff ff ff	++ ff ff ff ff ff ff	++ ff ff ff ff ff ff	++ ff ff ff ff ff ff	++ ff ff ff ff ff	++ ff ff ff ff ff	++ ff ff ff ff ff ff	++ ff ff ff ff ff ff	++ ff ff ff ff ff	++ ff ff ff ff ff	++ ff ff ff ff ff	++ ff ff ff ff ff	++ ff ff ff ff ff ff	tt ff ff ff ff ff		-
4																	►	< ▶	



1.12.3. ブロックイレースの確認

フラッシュメモリのブロックをクリアし、ブロックがクリアされているか確認ことによって、Memory Command Builder で組み立てた "ブロックイレース"の動作を確認します。

[フラッシュメモリ設定]ダイアログを起動してください。

フラッシュメモリのクリア タブを選択し、以下のように<アドレスを含むブロックをクリア>にチェックを入れた状態で、開始 ボタンを押してください。

フラッシュメモリ設定	×
デバイス設定 ワークメモリ フ`ロック'情報 設定ファイルの保存/読み込み フラッシュメモリのフィルアッフ° その他の設定 'ワノ*範囲の指定 () ⑦ 登録デバイスを全てクリア(G) () 'リンアドレス(B) 0x0000000 アト*レス領域の設定 () ? アドレスを含むデバイス全体をクリア(N) () ? アドレスを含むブロックをクリア(B); ()	追加(<u>A</u>) 肖ᆙ除(<u>D</u>) 全削除(<u>L</u>) 閉じる(<u>C</u>)
	タ [°] ウンロート [°] (型) アッフ [°] ロート [°] (<u>U</u>)
R;0x00000000 R;0x000fffff WM K70_1M0_ProgramFlash R0D0 16Byte	e R;0×20002000
□ ダウンロード時設定を無視する(<u>I</u>) ワークメモリ R;0×20000000 -	R;0x20001fff

Figure 41



本構成の場合、0x1000 単位でブロッククリアされます。 (ブロックイレースはフラッシュメモリの Erase Flash Sector を使用していますが、この Erase Flash Sector の仕様が 0x1000 単位でクリアするようになっているためです。)



クリア処理中は以下のダイアログが表示されます。

チップ消去中です	
アドレス 経過時間	0×00000000 3 秒
	キャンセル





本構成の場合、1秒以内に完了してしまいますので、ダイアログはほとんど表示されません。

クリア処理が終了すると、フラッシュメモリ設定ダイアログに操作結果が表示されます。

// フラッシュメモリクリアを実行します // Block Clear : 0x00000000 FMCLEAR BLOCK CLEAR : 0x00000000 // コアンド実行をします // 終了しました

Figure 43

[フラッシュメモリ設定]ダイアログを 閉じる ボタンで閉じてください。 指定範囲が正常クリア(ブロックイレース)されたかどうか、ダンプウィンドウで確認してください。



0x0000000 番地を含むブロックが消去されるので、0x0000000~0x00000FFF 番地のみクリア(0xFF)され、 0x00001000~0x00001FFF 番地はライトで書いた 0x99 がそのまま残るのが正しい結果です。



💷 ቃ*ንፓ° ኃብ	ント*ウ1 (0x00	000	000	- 0x	0000	000fl	f (0x	100])									- 0	×
アドレス(S):	0×00	00000	00			•	0 8	終了)	アドレス	z(<u>E</u>)	0×0	0000	Off			Ŧ	アクセスサイスで <u>A</u>): ロングワート	-	
テ°─タ(<u>D</u>):	0×ff	_	_	_	X	V	⊙ i	範囲	辰(L)		0×1	00				•	Q 🛛 🗆 更	新しない(山)		
Address		+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F	ASCII		- A-
0x000000 0x000000 0x000000 0x000000 0x000000	00 10 20 30 40 50 60 70 80 90 80 90 b0 c0 c0 c0	ff ff ff ff ff ff ff ff ff ff ff	ff ff ff ff ff ff ff ff ff ff ff ff ff	ff ff ff ff ff ff ff ff ff ff ff ff	ff ff ff ff ff ff ff ff ff ff ff ff	ff ff ff ff ff ff ff ff ff ff ff ff ff	ff ff ff ff ff ff ff ff ff ff ff ff	ff ff ff ff ff ff ff ff ff ff ff ff	ff ff ff ff ff ff ff ff ff ff ff ff	ff ff ff ff ff ff ff ff ff ff ff	ff ff ff ff ff ff ff ff ff ff ff	ff ff ff ff ff ff ff ff ff ff ff ff ff	ff ff ff ff ff ff ff ff ff ff ff ff ff	ff ff ff ff ff ff ff ff ff ff ff ff ff	ff ff ff ff ff ff ff ff ff ff ff ff ff	ff ff ff ff ff ff ff ff ff ff ff ff ff	ff ff			
0×000000	f0 ▶	ff ∢	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	ff	4		▼ }

Figure 44

💷 ቃ ኦንፖ ሳብኦኑ ሳ2 0x00	0000f80 - 0x000	0107f (0x100)		_ • •
アドレス(<u>S</u>): 0×00000f	80 💌	○ 終了アドレス(<u>E</u>)	0×0000107f	アクセスサイス(A): ロングワード
データ(<u>D</u>): 0×ff	XV	⊙ 範囲長(<u>L</u>)	0×100	💌 🔍 🗆 更新しない(U)
Address +0	+1 +2 +3 +4	+5 +6 +7 +8	+9 +A +B +C +D +	FE +F ASCII 🚔
0×00000f80 ff	ff ff ff ff	ff ff ff ff	ff ff ff ff ff ff f	f ff
0x00000190 11	++ ++ ++ ++	11 11 11 11 11	++++++++++++++++++++++++++++++++++++	
0x000001a0 ff		ff ff ff ff	ff ff ff ff ff ff ff	
0x00000fc0 ff	ff ff ff ff	ff ff ff ff	ff ff ff ff ff ff f	fff IIIIIIIIIII
0x00000fd0 ff	ff ff ff ff	ff ff ff ff	ff ff ff ff ff ff f	f ff
	11 11 11 11 ff ff ff ff ff	11 11 11 11 ff ff ff ff ff	TT TT TT TT TT TT T ff ff ff ff ff ff ff	
0×00001000 99	99 99 99 99	99 99 99 99 99	99 99 99 99 99 9	999
0×00001010 99	99 99 99 99	99 99 99 99 99	99 99 99 99 99 9	9 99 劒劒劒劒劒劒劒劒
UXUUUU1U2U 99	99 99 99 99 95 99 99 99 99 95	99 99 99 99 99 aa aa aa aa	99 99 99 99 99 99 9 99 99 99 99 99 99 9	19 99
0×00001040 99	99 99 99 99 99	99 99 99 99 99	99 99 99 99 99 99 9	999 俞敏敏敏敏敏敏
0×00001050 99	99 99 99 99	99 99 99 99	99 99 99 99 99 9	99 99 🛛 劒 劒劍 劒劍 劒劍 劒 🗐
0×00001060 99	99 99 99 99 99	99 99 99 99 99 00 00 00 00	99 99 99 99 99 99 9	
UXUUUU1070 99	୍ରର ସହ ସହ ସହ ।	22 22 23 23	33 33 33 33 33 33 3	19 99



💷 ቃ*ንプウィンドウ3 0x00001f80 - 0x000	0207f (0x100)	- • •
7ドレス(<u>S</u>): 0x00001f80 ・	○ 終了アドレス(E) 0x0000207f 🛛 🖳	アクセスサイス (<u>A</u>): ロングワード 💌
テ ^ĸ -ḥ(<u>D</u>): 0×99 X√	● 範囲長(L)	🔍 💡 🗆 更新しない(山)
Address +0 +1 +2 +3 +4	4 +5 +6 +7 +8 +9 +A +B +C +D +E	+F ASCII 🛨
0×00001f80 99 99 99 99 99 0×00001f90 99 99 99 99 99 0×00001f90 99 99 99 99 99) 99 99 99 99 99 99 99 99 99 99) 99 99 99 99 99 99 99 99 99 99) 99 99 99 99 99 99 99 99 99 99	99
0×00001fb0 99 99 99 99 99 0×00001fc0 99 99 99 99 99	9 99 </td <td>99 劒劒劒劒劒劒劒劒劒 99 劒劒劒劒劒劒劒劒劒 99 劒劒劒劒劒劒劒劒劒</td>	99 劒劒劒劒劒劒劒劒劒 99 劒劒劒劒劒劒劒劒劒 99 劒劒劒劒劒劒劒劒劒
0×00001fd0 99 99 99 99 99 0×00001fe0 99 99 99 99 99 0×00001ff0 99 99 99 99 99) 99 99 99 99 99 99 99 99 99 99 99) 99 99 99 99 99 99 99 99 99 99 99) 99 99 99 99 99 99 99 99 99 99 99	99
0×00002000 ff ff ff ff f 0×00002010 ff ff ff ff ff	F FF FF <t< td=""><td>ff ff ff</td></t<>	ff ff ff
0×00002020 ff ff ff ff ff 0×00002030 ff ff ff ff ff 0×00002040 ff ff ff ff ff	F FF	ff ff
0×00002050 ff ff ff ff f 0×00002060 ff ff ff ff ff	f ff ff f ff f	ff ff tt
	- 11 11 11 11 11 11 TT TT TT TT TT	



1.12.4. リードの確認

フラッシュメモリからデータを読み出し、正しいデータであるか確認することによって、Memory Command Builder で組み立てた "リード"の動作を確認します。

フラッシュメモリ設定ダイアログを表示してください。

デバイス設定 タブを選択し、<メモリリードもプログラムを使用>にチェックを入れて、 追加 ボタンを押してください。 次に アップロード ボタンを押してください。

フラッシュメモリ設定					
フラッシュメモリのフィルアッフ。 その他の設定 道加(A) ア・カメモリ フロック情報風 設定ファイルの保存/読み込み 川時(D) ア・カ~(M) CUSTOM サイス*(Z) その他 ビー 解告 (L) (L) 全削除(L) 全削除(L) 日 0×100000 マー ア・ヴァック特報級を使用(B) 第じこる(C) リート*・ケット* うっパト フ*ロか*うは使用タイフ* ア・ウック情報級を使用(B) ア・ウックト・ケット 第じこる(C) リート*・ケット* うっパト フ*ロか*うは使用タイフ* ア・ウックをしまり ア・ウックをしまり ア・ウックト・・ケット ア・ド・ケット* うっか* うん処理開始タト* しス(P) ア・ウックをしまり ア・ウックロート*(M) ア・ウッロート*(M) ア・ウッロート*(M) ア・ウンウ・うんを実行する 直前で で・レーン 16 Byte Al ignment マ ア・フ*ロート*(U) ア・フ*ロート*(U) ア・フ*ロート*(U)					
✓ III ト					

Figure 47



<メモリリードもプログラムを使用>にチェックを入れると、Memory Command Builder で組み立てた"リード"を使用 するようになります。アップロード、ベリファイ機能に利用されます。 WATCHPOINT の場合は、逆アセンブラウィンドウにも使用されます(ワークメモリが書き換わるので注意ください)。



アップロードダイアログが起動しますので、以下を参照して、アップロード範囲を決めてください。 設定したら アップロード ボタンを押してください。

747° 0-1×*		x
フォーマット (<u>M</u>) :	/\`イナリデータ	アッフ⁰ロートヾ <u>(U</u>)
ファイル名(E):	D:¥K70_ProgramFlash	閉じる
- バッフロート、範囲の設定 開始アトシス (<u>A</u>):	0x00000000	
○ 終了アドレス(<u>E</u>):		
 レンゲス (L): 	0×2000	
PCアドレスの設定 □ PCアドレスを指定 (P):		

Figure 48

アップロードが完了すると、以下のダイアログが表示されます。 OK ボタンを押してください。

Universal Probe
D:¥K70_ProgramFlash アップロードは正常に終了しました
ОК

Figure 49



保存先に アップロードしたファイルがあるか確認して、バイナリファイルが読み込めるエディッタでアップロードしたデータの内容が正しい か確認してください。

ADDRESS	00	01	02	03	04	05	06	07	08	09	0A	0B	00	0D	0E	0F
000F80	FF															
000F90	FF															
000FA0	FF															
000FB0	FF															
000FC0	FF															
000FD0	FF															
000FE0	FF															
000FF0	FF															
001000	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99
001010	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99
001020	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99
001030	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99
001040	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99
001050	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99
001060	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99
001070	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99

Figure 50

以上で終了です。



改訂履歴

版数	改訂日	改訂内容
01	2014/12/12	初版



製造者情報



株式会社 Sohwa & Sophia Technologies

	〒215-8588
[本社]	神奈川県川崎市麻生区南黒川 6-2
	ホームページ: <u>http://www.ss-technologies.co.jp</u>

子会社



Unit 5-2, Level 5, Tower 6, Avenue 5, The Horizon, Bangsar South No.8, Jalan Kerinchi 59200, Kuala Lumpur, Malaysia

HomePage : http://www.sohwa-m.com.my/