

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のですが、誤りが無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、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。

アプリケーション・ノート

78K0/Kx2-L, 78K0/Ix2

8ビット・シングルチップ・マイクロコントローラ

フラッシュ・メモリ・プログラミング (プログラマ編)

78K0/KY2-L: μPD78F0550, 78F0551, 78F0552, 78F0555, 78F0556, 78F0557

78K0/KA2-L: μPD78F0560, 78F0561, 78F0562, 78F0565, 78F0566, 78F0567

78K0/KB2-L: μPD78F0571, 78F0572, 78F0573, 78F0576, 78F0577, 78F0578

78K0/KC2-L: μPD78F0581, 78F0582, 78F0583, 78F0586, 78F0587, 78F0588

78K0/IY2: μPD78F0740, 78F0741, 78F0742, 78F0750, 78F0751, 78F0752

78K0/IA2: μPD78F0743, 78F0744, 78F0753, 78F0754

78K0/IB2: μPD78F0745, 78F0746, 78F0755, 78F0756

〔メ モ〕

目次要約

第1章	フラッシュ・メモリ・プログラミング	...	12
第2章	コマンド/データ・フレーム・フォーマット	...	28
第3章	コマンド処理説明	...	31
第4章	UART通信方式	...	54
第5章	フラッシュ・メモリ・プログラミング・パラメータ特性	...	105
付 録 A	参考回路図	...	122

CMOSデバイスの一般的注意事項

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。

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。

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イやマガジン・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。

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。

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。

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2009年5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に、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。

はじめに

- 注 意** このアプリケーション・ノートでは、特に断りがないかぎり、78K0/Kx2-Lを代表製品として記述しています。78K0/Kx2-L以外の製品については、78K0/Kx2-Lをそれぞれの製品名に読み替えてください。
- 対 象 者** このアプリケーション・ノートは、78K0/Kx2-L, 78K0/lx2の機能を理解し、それを用いたアプリケーション・システムを設計するユーザを対象としています。
- 目 的** このアプリケーション・ノートは、78K0/Kx2-L, 78K0/lx2内蔵のフラッシュ・メモリの書き換えを行うのに、ユーザ専用のフラッシュ・メモリ・プログラムを開発するための方法をユーザに理解していただくことを目的としています。
- なお、掲載のプログラムおよび回路図は例示したものであり、量産設計を対象とするものではありません。
- したがって、お客様の機器に使用される場合には、設計後、お客様の責任において十分な評価を行ってください。
- 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。
- ・フラッシュ・メモリ・プログラミング
 - ・コマンド/データ・フレーム・フォーマット
 - ・コマンド処理説明
 - ・UART通信方式
 - ・フラッシュ・メモリ・プログラミング・パラメータ特性
- 読 み 方** このマニュアルを読むにあたっては、電気、論理回路、マイクロコントローラの一般知識を必要とします。
- 一通りの機能を理解しようとするとき
- 目次に従って読んでください。
- 78K0/Kx2-L, 78K0/lx2のハードウェア機能を知りたいとき
- 78K0/Kx2-L, 78K0/lx2各製品のユーザーズ・マニュアル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・ロウの表記 | : $\overline{\text{xxx}}$ (端子, 信号名称に上線) |
| 注 | : 本文中につけた注の説明 |
| 注意 | : 気をつ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... xxx または xxx B |
| | 10進数... xxx |
| | 16進数... xxx H |

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。

デバイスの関連資料

資 料 名	資料番号	
	和 文	英 文
78K0/Kx2-L ユーザーズ・マニュアル	U19111J	U19111E
78K0/lx2 ユーザーズ・マニュアル	U19353J	U19353E
78K0マイクロコントローラ 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 フラッシュ・メモリ・プログラミング ... 12

- 1.1 概 要 ... 12
- 1.2 システム構成 ... 13
- 1.3 フラッシュ・メモリ構成 ... 14
- 1.4 コマンド/ステータス一覧 ... 16
 - 1.4.1 コマンド一覧 ... 16
 - 1.4.2 ステータス一覧 ... 17
- 1.5 電源投入とプログラミング・モードへの遷移 ... 18
 - 1.5.1 単線UART通信方式 ... 19
 - 1.5.2 モード引き込みのフロー・チャート ... 20
 - 1.5.3 サンプル・プログラム ... 23
- 1.6 ターゲットの電源遮断処理 ... 25
- 1.7 フラッシュ・メモリ書き換えコマンド・フロー ... 25

第2章 コマンド/データ・フレーム・フォーマット ... 28

- 2.1 コマンド・フレーム送信処理 ... 30
- 2.2 データ・フレーム送信処理 ... 30
- 2.3 データ・フレーム受信処理 ... 30

第3章 コマンド処理説明 ... 31

- 3.1 Resetコマンド ... 31
 - 3.1.1 説 明 ... 31
 - 3.1.2 コマンド・フレームとステータス・フレーム ... 31
- 3.2 Baud Rate Setコマンド ... 32
 - 3.2.1 説 明 ... 32
 - 3.2.2 コマンド・フレームとステータス・フレーム ... 32
- 3.3 Chip Eraseコマンド ... 33
 - 3.3.1 説 明 ... 33
 - 3.3.2 コマンド・フレームとステータス・フレーム ... 33
- 3.4 Block Eraseコマンド ... 34
 - 3.4.1 説 明 ... 34
 - 3.4.2 コマンド・フレームとステータス・フレーム ... 34
- 3.5 Programmingコマンド ... 35
 - 3.5.1 説 明 ... 35
 - 3.5.2 コマンド・フレームとステータス・フレーム ... 35
 - 3.5.3 データ・フレームとステータス・フレーム ... 35
 - 3.5.4 全データ転送完了とステータス・フレーム ... 36
- 3.6 Verifyコマンド ... 37
 - 3.6.1 説 明 ... 37
 - 3.6.2 コマンド・フレームとステータス・フレーム ... 37
 - 3.6.3 データ・フレームとステータス・フレーム ... 37
- 3.7 Block Blank Checkコマンド ... 39

3.7.1	説 明 ...	39
3.7.2	コマンド・フレームとステータス・フレーム ...	39
3.8	Silicon Signature コマンド ...	40
3.8.1	説 明 ...	40
3.8.2	コマンド・フレームとステータス・フレーム ...	40
3.8.3	シリコン・シグネチャ・データ・フレーム ...	41
3.8.4	シリコン・シグネチャー覧 ...	43
3.9	Version Get コマンド ...	48
3.9.1	説 明 ...	48
3.9.2	コマンド・フレームとステータス・フレーム ...	48
3.9.3	バージョン・データ・フレーム ...	49
3.10	Checksum コマンド ...	50
3.10.1	説 明 ...	50
3.10.2	コマンド・フレームとステータス・フレーム ...	50
3.10.3	チェックサム・データ・フレーム ...	50
3.11	Security Set コマンド ...	51
3.11.1	説 明 ...	51
3.11.2	コマンド・フレームとステータス・フレーム ...	51
3.11.3	データ・フレームとステータス・フレーム ...	52
3.11.4	内部ペリファイ確認とステータス・フレーム ...	52

第4章 UART通信方式 ... 54

4.1	コマンド・フレーム送信処理のフロー・チャート ...	54
4.2	データ・フレーム送信処理のフロー・チャート ...	55
4.3	データ・フレーム受信処理のフロー・チャート ...	56
4.4	Reset コマンド ...	57
4.4.1	処理手順チャート ...	57
4.4.2	処理手順説明 ...	58
4.4.3	終了時の内容 ...	58
4.4.4	フロー・チャート ...	59
4.4.5	サンプル・プログラム ...	60
4.5	Baud Rate Set コマンド ...	61
4.5.1	処理手順チャート ...	61
4.5.2	処理手順説明 ...	62
4.5.3	終了時の内容 ...	62
4.5.4	フロー・チャート ...	63
4.5.5	サンプル・プログラム ...	64
4.6	Chip Erase コマンド ...	65
4.6.1	処理手順チャート ...	65
4.6.2	処理手順説明 ...	66
4.6.3	終了時の内容 ...	66
4.6.4	フロー・チャート ...	67
4.6.5	サンプル・プログラム ...	68
4.7	Block Erase コマンド ...	69
4.7.1	処理手順チャート ...	69
4.7.2	処理手順説明 ...	70
4.7.3	終了時の内容 ...	70
4.7.4	フロー・チャート ...	71
4.7.5	サンプル・プログラム ...	72

4. 8	Programming コマンド	...	73
4. 8. 1	処理手順チャート	...	73
4. 8. 2	処理手順説明	...	74
4. 8. 3	終了時の内容	...	75
4. 8. 4	フロー・チャート	...	76
4. 8. 5	サンプル・プログラム	...	77
4. 9	Verify コマンド	...	79
4. 9. 1	処理手順チャート	...	79
4. 9. 2	処理手順説明	...	80
4. 9. 3	終了時の内容	...	80
4. 9. 4	フロー・チャート	...	81
4. 9. 5	サンプル・プログラム	...	82
4. 10	Block Blank Check コマンド	...	84
4. 10. 1	処理手順チャート	...	84
4. 10. 2	処理手順説明	...	85
4. 10. 3	終了時の内容	...	85
4. 10. 4	フロー・チャート	...	86
4. 10. 5	サンプル・プログラム	...	87
4. 11	Silicon Signature コマンド	...	88
4. 11. 1	処理手順チャート	...	88
4. 11. 2	処理手順説明	...	89
4. 11. 3	終了時の内容	...	89
4. 11. 4	フロー・チャート	...	90
4. 11. 5	サンプル・プログラム	...	91
4. 12	Version Get コマンド	...	92
4. 12. 1	処理手順チャート	...	92
4. 12. 2	処理手順説明	...	93
4. 12. 3	終了時の内容	...	93
4. 12. 4	フロー・チャート	...	94
4. 12. 5	サンプル・プログラム	...	95
4. 13	Checksum コマンド	...	96
4. 13. 1	処理手順チャート	...	96
4. 13. 2	処理手順説明	...	97
4. 13. 3	終了時の内容	...	97
4. 13. 4	フロー・チャート	...	98
4. 13. 5	サンプル・プログラム	...	99
4. 14	Security Set コマンド	...	100
4. 14. 1	処理手順チャート	...	100
4. 14. 2	処理手順説明	...	101
4. 14. 3	終了時の内容	...	101
4. 14. 4	フロー・チャート	...	102
4. 14. 5	サンプル・プログラム	...	103

第5章 フラッシュ・メモリ・プログラミング・パラメータ特性 ... 105

5. 1	78K0/Kx2-Lのフラッシュ・メモリ・パラメータ特性	...	105
5. 1. 1	フラッシュ・メモリ・プログラミング・モード・セット時間	...	105
5. 1. 2	プログラミング特性	...	106
5. 1. 3	コマンド特性	...	107

5.2	78K0/1x2のフラッシュ・メモリ・パラメータ特性	...	108
5.2.1	フラッシュ・メモリ・プログラミング・モード・セット時間	...	108
5.2.2	プログラミング特性	...	109
5.2.3	コマンド特性	...	110
5.3	Block Eraseコマンドにおける同時選択消去について	...	111
5.3.1	同時選択消去ブロック数の求め方	...	111
5.3.2	同時選択消去の実行回数 (M) の求め方	...	112
5.4	UART通信方式	...	119
付録A	参考回路図	...	122

第1章 フラッシュ・メモリ・プログラミング

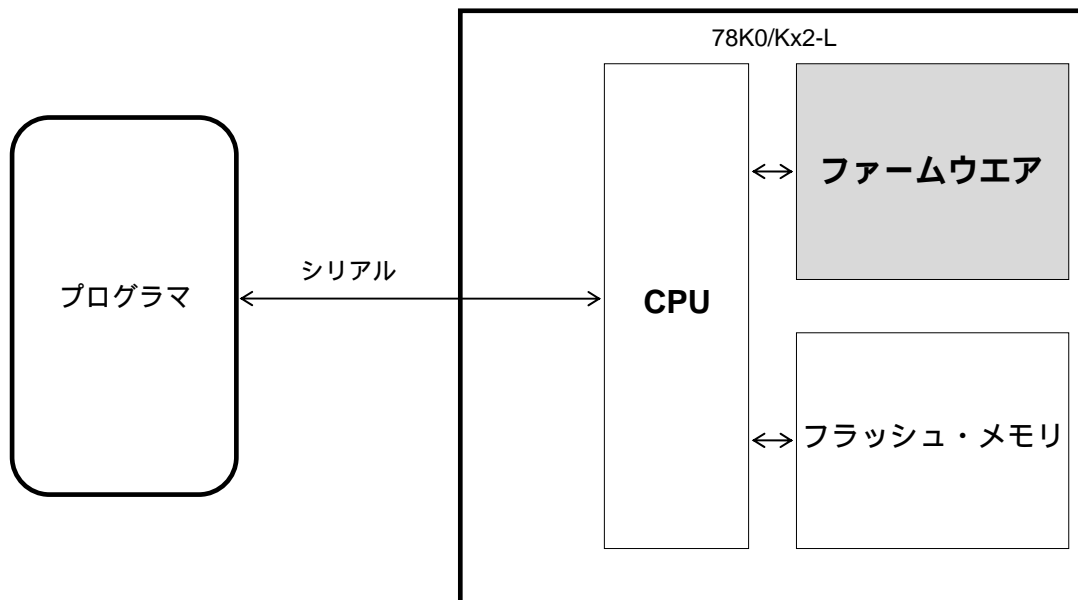
78K0/Kx2-Lに内蔵されるフラッシュ・メモリの書き換えを行うには、通常は専用のフラッシュ・メモリ・プログラマ（以降プログラマ）を使う必要があります。

このアプリケーション・ノートでは、ユーザが専用のプログラマを開発するための方法を説明します。

1.1 概 要

78K0/Kx2-Lは、フラッシュ・メモリ書き換え制御を行うファームウェアを内蔵しています。シリアル通信により、プログラマと78K0/Kx2-L間でコマンドを送受信し、内蔵フラッシュ・メモリの書き換えを行います。

図1 - 1 78K0/Kx2-Lのフラッシュ・メモリ・プログラミングのシステム概略



1.2 システム構成

フラッシュ・メモリ・プログラミング時のシステム構成例を次に示します。

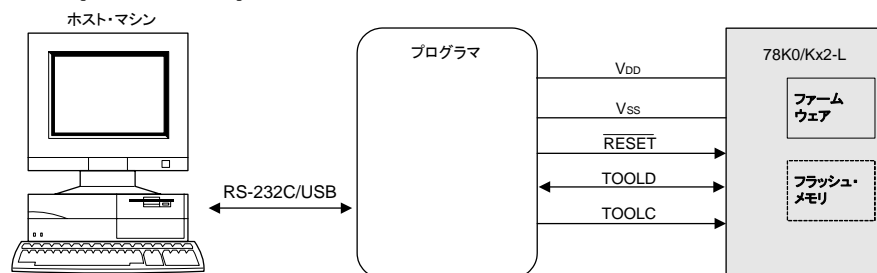
図1 - 2は、ホスト・マシンからの制御によりプログラマを使用するプログラミング方法を示しています。

プログラマの実装方法によって、あらかじめユーザ・プログラムがプログラマにダウンロードされている場合には、ホスト・マシンを使用せずにスタンド・アローンでもプログラマを動作させることができます。

たとえば、NECエレクトロニクス製フラッシュ・メモリ・プログラマ PG-FP5は、ホスト・マシンを接続してGUIソフトウェアにより実行する方法と、スタンド・アローンで実行する方法のどちらでも動作可能です。

図1 - 2 システム構成例

単線UART通信方式（LSB先頭転送）



備考1. 78K0/Kx2-Lの通信方式は、単線UART通信方式のみ使用可です。

2. プログラミング時に使用する端子名および未使用端子の処理に関しては各製品のユーザーズ・マニュアルを参照してください。

1.3 フラッシュ・メモリ構成

各製品は、プログラマ側で製品固有情報（デバイス名、メモリ情報）を管理しておく必要があります。

表1 - 1に各製品のフラッシュ・メモリ・サイズ、図1 - 3にフラッシュ・メモリ構成を示します。

表1 - 1 各製品のフラッシュ・メモリ・サイズ

(a) 78K0/Kx2-Lのフラッシュ・メモリ・サイズ

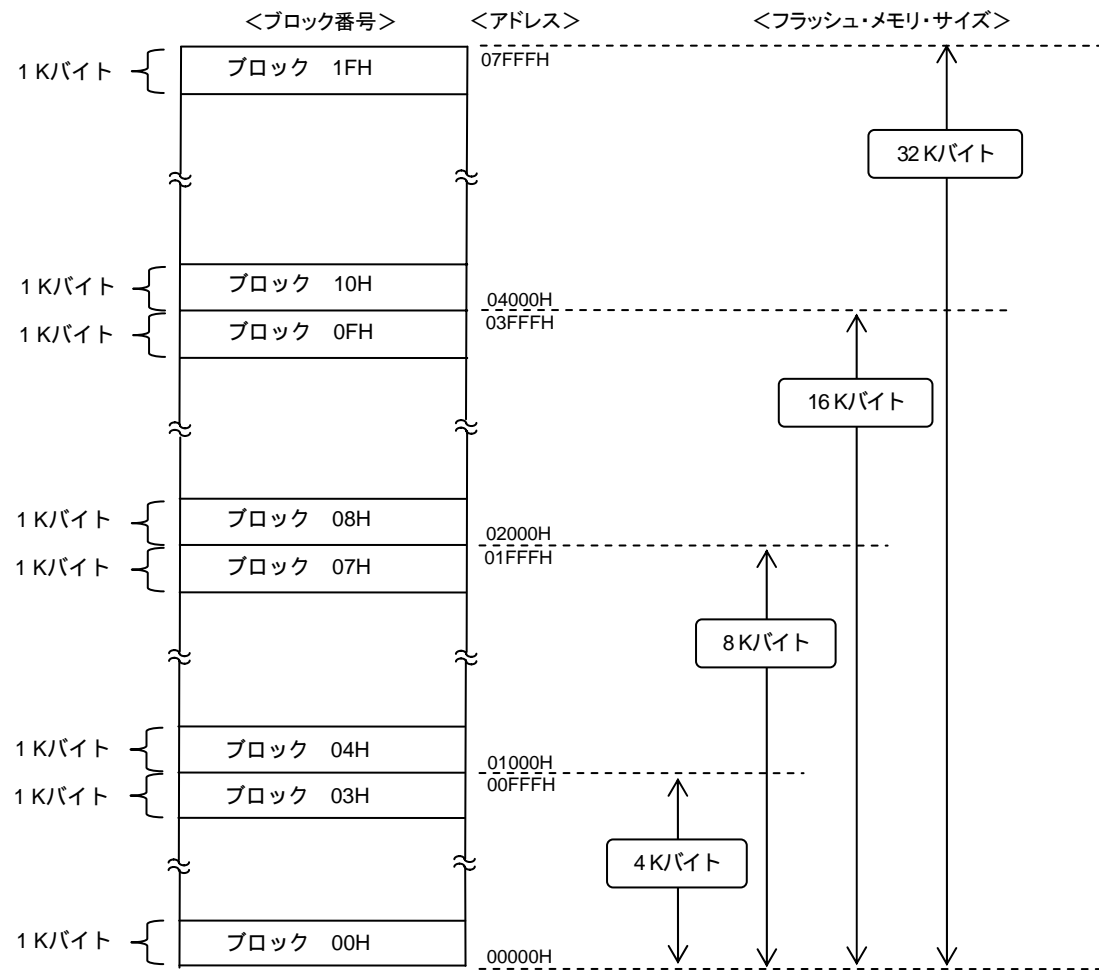
デバイス名	フラッシュ・メモリ・サイズ
μ PD78F0550, 78F0555, 78F0560, 78F0565	4 Kバイト
μ PD78F0551, 78F0556, 78F0561, 78F0566, 78F0571, 78F0576, 78F0581, 78F0586	8 Kバイト
μ PD78F0552, 78F0557, 78F0562, 78F0567, 78F0572, 78F0577, 78F0582, 78F0587	16 Kバイト
μ PD78F0573, 78F0578, 78F0583, 78F0588	32 Kバイト

(b) 78K0/lx3のフラッシュ・メモリ・サイズ

デバイス名	フラッシュ・メモリ・サイズ
μ PD78F0740, 78F0750	4 Kバイト
μ PD78F0741, 78F0743, 78F0745, 78F0751, 78F0753, 78F0755	8 Kバイト
μ PD78F0742, 78F0744, 78F0746, 78F0752, 78F0754, 78F0756	16 Kバイト

備考 上記は、開発中の製品も含みます。

図1 - 3 フラッシュ・メモリ構成



備考 1ブロックは、すべて1 Kバイトです（この図では、ブロックは一部分しか表記しておりません）。

1.4 コマンド/ステータス一覧

78K0/Kx2-Lが内蔵するフラッシュ・メモリには、フラッシュ・メモリ書き換えのための機能が内蔵されており、表1-2に示すようなフラッシュ・メモリ操作機能があります。プログラマは、これらの機能を制御するコマンドを78K0/Kx2-Lに送信し、78K0/Kx2-Lからの応答ステータスを確認しながらフラッシュ・メモリを操作します。

1.4.1 コマンド一覧

プログラマで使用されるコマンドの一覧と機能を次に示します。

表1-2 プログラマから78K0/Kx2-Lへの送信コマンド一覧

コマンド番号	コマンド名	機能名	機 能
20H	Chip Erase	消去	全フラッシュ・メモリを消去します。
22H	Block Erase		指定された領域のフラッシュ・メモリを消去します。
40H	Programming	書き込み	フラッシュ・メモリの指定された領域にデータを書き込みます。
13H	Verify	ベリファイ	フラッシュ・メモリの指定された領域の内容とプログラマから送信されたデータを比較します。
32H	Block Blank Check	ブランク・チェック	指定されたブロックのフラッシュ・メモリの消去状態をチェックします。
C0H	Silicon Signature	情報取得	78K0/Kx2-L情報（品名、フラッシュ・メモリ構成など）を取得します。
C5H	Version Get		78K0/Kx2-Lのバージョン、ファームウェア・バージョンを取得します。
B0H	Checksum		指定された領域のチェックサム・データを取得します。
A0H	Security Set	セキュリティ	セキュリティ情報を設定します。
00H	Reset	その他	通信同期検出に使用します。
9AH	Baud Rate Set		UART選択時のボー・レートを設定します。

1.4.2 ステータス一覧

プログラマが78K0/Kx2-Lから受信するステータス・コードの一覧を次に示します。

表1 - 3 ステータス・コード一覧

ステータス・コード	ステータス	内 容
04H	Command number error	サポートされていないコマンドを受信した場合のエラー
05H	Parameter error	コマンド情報（パラメータ）が適切でない場合のエラー
06H	正常応答（ACK）	正常応答
07H	Checksum error	プログラマから送信されたフレームのデータが異常の場合のエラー
0FH	Verify error	プログラマから送信されたデータとのベリファイでエラー発生
10H	Protect error	Security Setコマンドで禁止した処理を実行しようとした場合のエラー
15H	否定応答（NACK）	否定応答
1AH	MRG10 error	消去エラー
1BH	MRG11 error	データ書き込み時の内部ベリファイ・エラー，またはブランク・チェック・エラー
20H	Read error	セキュリティ情報読み出しに失敗した場合のエラー
1CH	Write error	書き込みエラー

なお，このマニュアルではChecksum errorやNACKを受信した際は即時異常終了として扱っていますが，実際にプログラマを作る際は，Checksum errorやNACKが発生したコマンド送信直前のウェイトからリトライしても構いません。ただし，無限にリトライを繰り返さないようにリトライの回数制限を設けることを推奨します。

また，上記ステータス・コード一覧には出てきませんが，各種タイムアウト・エラー（BUSYのタイムアウト，UART通信時のデータ・フレーム受信のタイムアウトなど）が発生した場合は，一度78K0/Kx2-Lに対して電源遮断処理（1.6 ターゲットの電源遮断処理参照）を行ってから改めて接続することを推奨します。

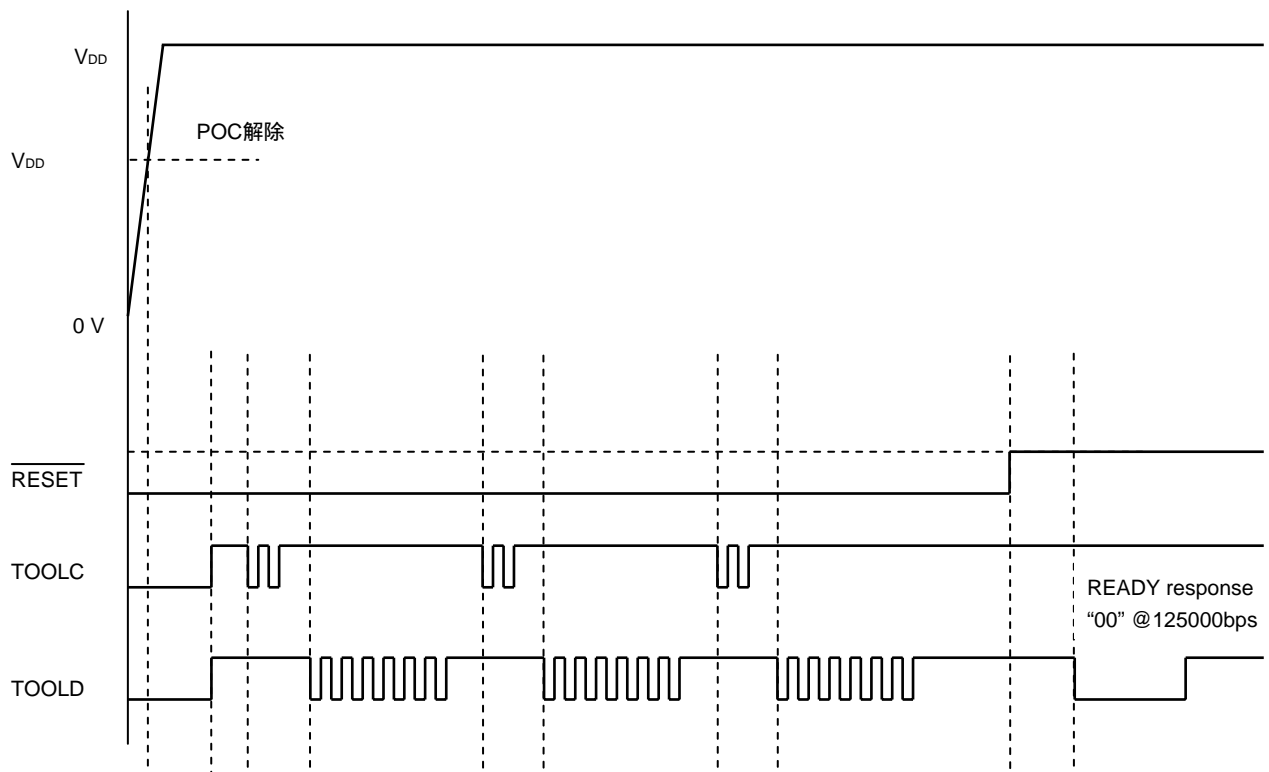
1.5 電源投入とプログラミング・モードへの遷移

プログラマにてフラッシュ・メモリの書き換えを行うには、まず78K0/Kx2-Lの動作モードをフラッシュ・メモリ・プログラミング・モードに遷移させる必要があります。

最初に $\overline{\text{RESET}}$, TOOLD, TOOLC端子をロウ・レベルにして、 $\overline{\text{RESET}}$ 端子のロウ・レベルを保ったまま、TOOLD, TOOLC端子をハイ・レベルに変更します。その後、TOOLC端子から2パルスとTOOLD端子から7パルスを交互に3回ずつ入力し、リセット解除することで、フラッシュ・メモリ・プログラミング・モードの動作モードに遷移可能となります。

フラッシュ・メモリ・プログラミング・モードへの遷移と通信方式の選択のタイミング図を次に示します。

図1-4 フラッシュ・メモリ・プログラミング・モードへの遷移および通信方式の選択



電源 (V_{DD}) 投入

TOOLC/TOOLD = ロウ・レベル出力 ハイ・レベル出力

TOOLCパルス出力 (2パルス)

TOOLDパルス出力 (7パルス)

TOOLCパルス出力 (2パルス)

TOOLDパルス出力 (7パルス)

TOOLCパルス出力 (2パルス)

TOOLDパルス出力 (7パルス)

リセット解除 (シリアル・プログラミング・モード確定)

READYパルス ("00" @125000bps入力開始 (マイコン プログラマ))

78K0/Kx2-Lで選択できる通信方式と使用するポートを次に示します。

表1 - 4 78K0/Kx2-Lの通信方式と使用ポート

通信方式	使用通信ポート
単線UART	TOOLD0, TOOLD1

1.5.1 単線UART通信方式

単線UART通信は、TOOLD端子を使用します。通信条件は次のようになります。

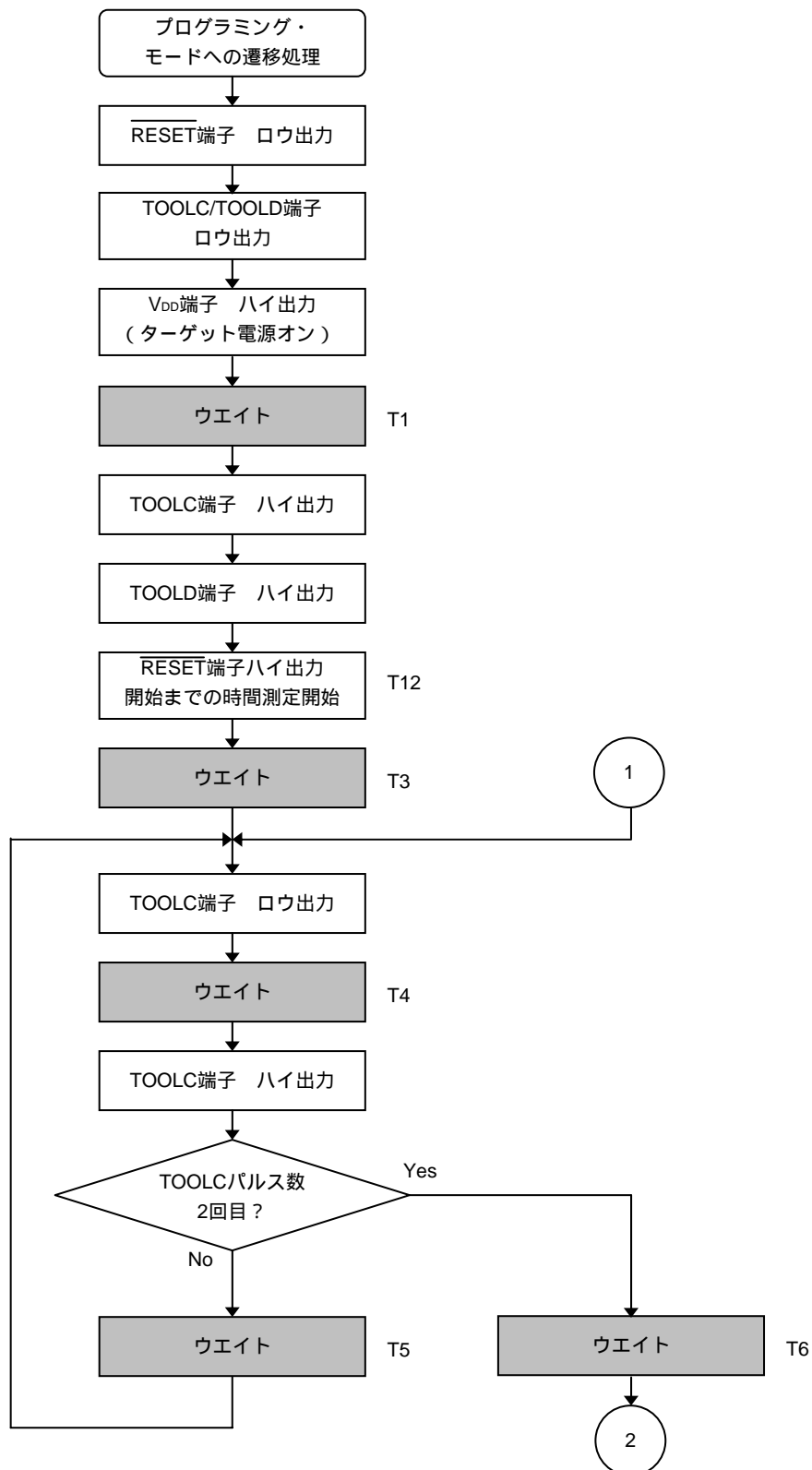
表1 - 5 単線UART通信の通信条件

項 目	内 容
ボー・レート	ボー・レート設定コマンド処理のBaud Rate Setコマンドの送信までは125000 bpsで通信を行います。そして、ボー・レート・コマンド処理のResetコマンドの送信からBaud Rate Setコマンドで設定したボー・レートに通信レートが変更になります。設定可能なボー・レートに関しては3.2 Baud Rate Setコマンドを参照してください。
パリティ・ビット	なし
データ長	8ビット (LSB先頭)
ストップ・ビット	2ビット (プログラマ 78K0/Kx2-L) / 1ビット (78K0/Kx2-L プログラマ)

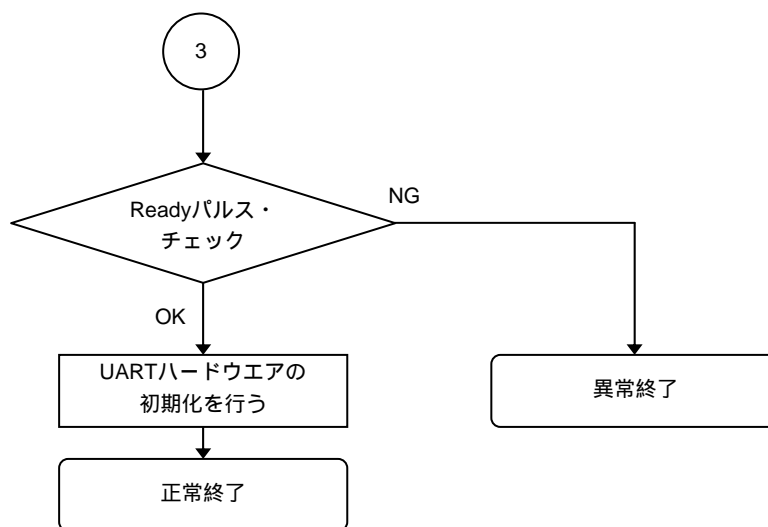
UART通信では、マスタとスレーブの関係を入れ換えながら通信を行うため、最適なタイミングでの通信が可能です。

注意 UART通信を行う場合は、マスタとスレーブともに同一のボー・レートにしてください。

1.5.2 モード引き込みのフロー・チャート







1.5.3 サンプル・プログラム

引き込み処理のサンプル・プログラムです。

```

/*****
/*
/* connect to Flash device
/*
/*
*****/
u16      fl_con_dev(void)
{
extern    void    init_fl_uart(void);
extern    void    init_fl_csi(void);
extern    void    stop_UART0(void);

    u16      rc = NO_ERROR;
    u8      cnt1,cnt2;

    SRMK0 = true;          // disable UART Rx INT.
    UARTE0 = false;        // disable UART H.W.
    stop_UART0();          // TxD/RxD = Hi-Z
    OE_ENABLE()            // TxD = output

    pFL_RES      = low;          // RESET = low
    pmFL_FLMD0    = PM_OUT;      // FLMD0 = Low output
    pFL_FLMD0     = low;
    pmFL_TxD      = PM_OUT;      // TxD = Low output
    pFL_TxD       = low;

    //VDD low => high
    FL_VDD_HI();              // VDD = high
    fl_wait(t1);             // wait

    //TOOLC(FLMD0),TOOLD low => high
    pFL_FLMD0     = hi;          // FLMD0 = high
    pFL_TxD       = hi;          // TOOLD = high
    start_flto(t12);          // start "t12" wait timer
    fl_wait(t3);             // wait

    cnt1 = 3;
    while(cnt1)
    {
        cnt1--;
        //TOOLC(FLMD0) output pulse (2 times)
        cnt2 = 2;
        while(cnt2)
        {
            cnt2--;
            pFL_FLMD0     = low;          // FLMD0 = low
            fl_wait(t4);          // wait
            pFL_FLMD0     = hi;          // FLMD0 = high
            if(cnt2){          fl_wait(t5);} // wait
            else{          fl_wait(t6);} // wait
        }

        //TOOLD output pulse (7 times)
        cnt2 = 7;
        while(cnt2)

```

```

        {
            cnt2--;
            pFL_TxD      = low;          // TOOLD = low
            fl_wait(t7);    // wait
            pFL_TxD      = hi;           // TOOLD = high
            if(cnt2){      fl_wait(t8);}  // wait
        }

        if(cnt1){      fl_wait(t9);}      // wait
        else{          fl_wait(t10);}      // wait
    }

    if(check_flto()){      // timeout "t12" ?
        return FLC_COM_ERR;    // yes
    }
    else{
        stop_fl_timer();      // no
    }

    //RESET low => high
    pFL_RES      = hi;          // RESET = high
    OE_DISABLE()    // TxD = Hi-Z
    pmFL_TxD      = PM_IN;      // TxD = Hi-Z

    fl_wait(t11_MIN);    // wait

    rc = check_ready_pulse();    // check "READY PULSE" from target device
    if (rc){
        return rc;            // pulse width/timing error
    }

    init_fl_uart();      // Initialize UART h.w.(for Flash device control)
    UARTE0 = true;      // enable UART h.w.
    SRIF0 = false;      // clear UART Rx IRQ flag
    SRMK0 = false;      // enable UART Rx INT.

    return rc;
    // start RESET command proc.
}

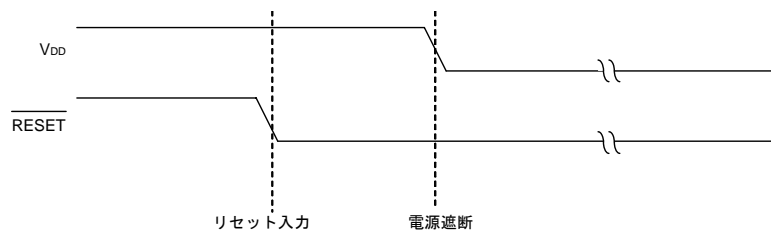
```

1.6 ターゲットの電源遮断処理

各コマンド実行の終了後に、下記のようにRESET端子をロウ・レベルにしてから電源を遮断してください。
また他の端子は、電源遮断時はHi-Zにしてください。

注意 コマンド処理中の電源遮断およびリセット入力は禁止です。

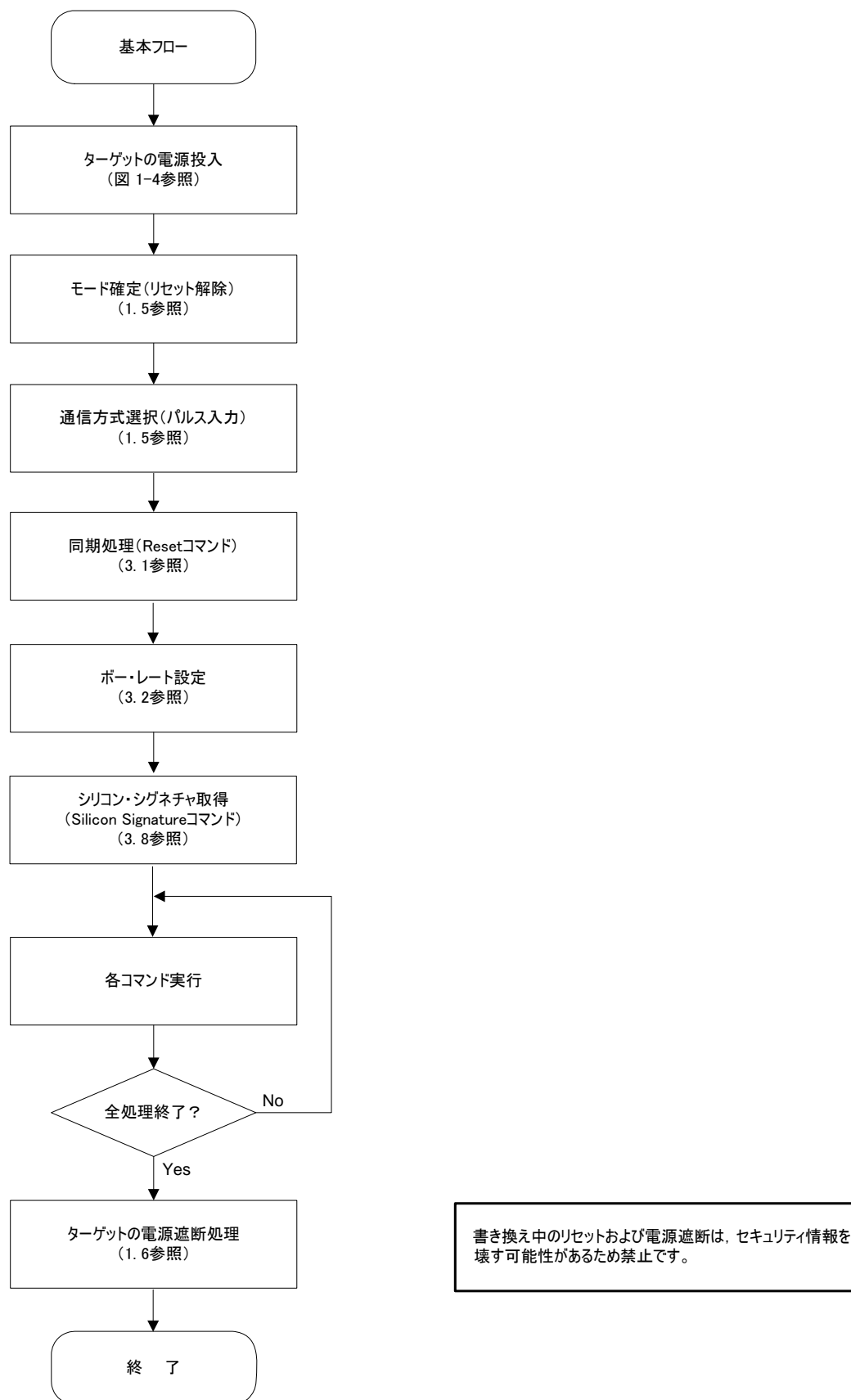
図1 - 5 フラッシュ・メモリ・プログラミングモードの終了手順



1.7 フラッシュ・メモリ書き換えコマンド・フロー

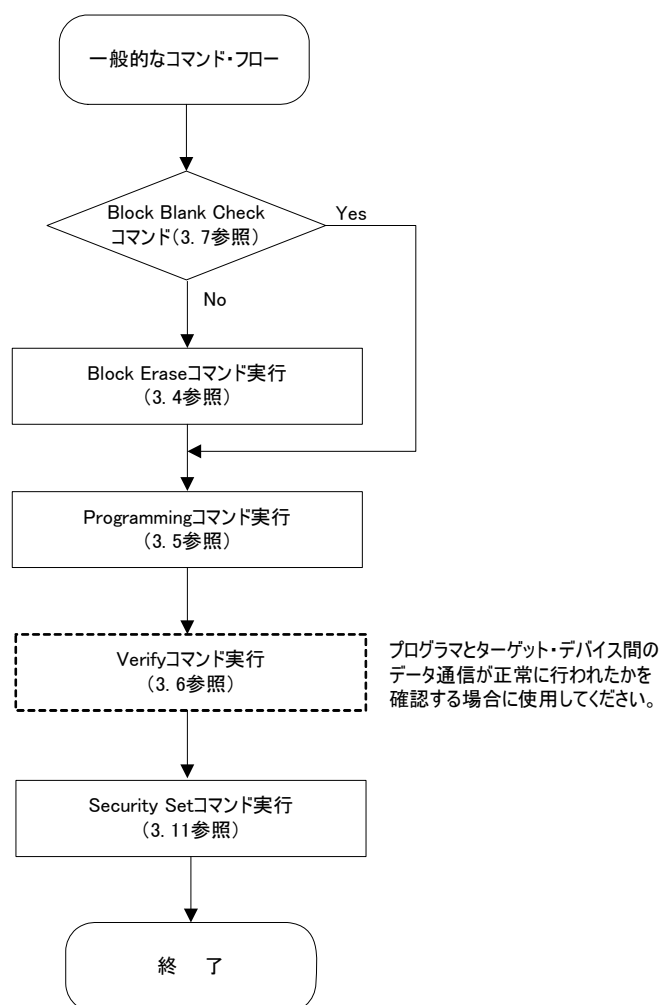
プログラマにてフラッシュ・メモリの書き換えを行う際の基本フロー・チャートを次に示します。
下記の基本フローで示したコマンドのほかにVerifyコマンドやChecksumコマンドをサポートしています。

図1 - 6 フラッシュ処理の基本フロー・チャート



備考 各コマンド実行の例は図1 - 7に示します。

図1 - 7 フラッシュ・メモリ書き換え時の一般的なコマンド・フロー



第2章 コマンド/データ・フレーム・フォーマット

プログラマと78K0/Kx2-L間でデータを送受信する際、プログラマがコマンドを送信する場合は、コマンド・フレームを使用します。78K0/Kx2-Lからプログラマに書き込みデータやペリファイ・データなどを送信する場合は、データ・フレームを使用します。これらのフレームには、転送データの信頼性を向上させるために、フレーム単位でヘッダ、フッタ、データ長情報、チェックサムを付けて送受信します。

次に両フレーム・フォーマットを示します。

図2 - 1 コマンド・フレームのフォーマット

SOH (1 バイト)	LEN (1 バイト)	COM (1 バイト)	コマンド情報 (可変長) (最大 255 バイト)	SUM (1 バイト)	ETX (1 バイト)
----------------	----------------	----------------	------------------------------	----------------	----------------

図2 - 2 データ・フレームのフォーマット

STX (1 バイト)	LEN (1 バイト)	データ (可変長) (最大 256 バイト)	SUM (1 バイト)	ETX or ETB (1 バイト)
----------------	----------------	---------------------------	----------------	-----------------------

表2 - 1 各フレームの記号説明

記号	値	内 容
SOH	01H	コマンド・フレームのヘッダ
STX	02H	データ・フレームのヘッダ
LEN	-	データ長情報 (00H = 256を示します)。 コマンド・フレームの場合 : COM + コマンド情報の長さ データ・フレームの場合 : データ・フィールドの長さ
COM	-	コマンド番号
SUM	-	フレーム内のチェックサム・データ。 初期値00Hから計算対象すべてのデータを1バイトごとに減算した値 (ボローは無視)。計算対象を次に示します。 コマンド・フレームの場合 : LEN + COM + コマンド情報すべて データ・フレームの場合 : LEN + データすべて
ETB	17H	データ・フレームの最終フレーム以外のフッタ
ETX	03H	コマンド・フレームのフッタ, またはデータ・フレームの最終フレームのフッタ

フレーム内のチェックサム (SUM) の計算例を次に示します。

【コマンド・フレームの場合】

Chip Eraseコマンド・フレームは次のようになります。この場合、コマンド情報がないので、チェックサム計算の対象になるのはLENとCOMです。

SOH	LEN	COM	SUM	ETX
01H	01H	20H	Checksum	03H
チェックサム計算対象				

この場合、チェックサム・データは次のように計算します。

$$00\text{H (初期値)} - 01\text{H (LEN)} - 20\text{H (COM)} = \text{DFH (ボロー無視。下位8ビットのみ)}$$

よって、Chip Eraseコマンド・フレームは最終的に次のようになります。

SOH	LEN	COM	SUM	ETX
01H	01H	20H	DFH	03H

【データ・フレームの場合】

たとえば、次のようなデータ・フレームを送信する場合、チェックサム計算の対象はLENからD4までです。

STX	LEN	D1	D2	D3	D4	SUM	ETX
02H	04H	FFH	80H	40H	22H	Checksum	03H
チェックサム計算対象							

この場合、チェックサム・データは次のように計算します。

$$\begin{aligned} &00\text{H (初期値)} - 04\text{H (LEN)} - \text{FFH (D1)} - 80\text{H (D2)} - 40\text{H (D3)} - 22\text{H (D4)} \\ &= 1\text{BH (ボロー無視。下位8ビットのみ)} \end{aligned}$$

よって、このデータ・フレームは最終的に次のようになります。

STX	LEN	D1	D2	D3	D4	SUM	ETX
02H	04H	FFH	80H	40H	22H	1BH	03H

データ・フレームを受信した場合も同様にチェックサム・データを計算して、その値が受信したSUMフィールドの値と同じであるか否かでチェックサム・エラーを検出できます。たとえば、次のようなデータ・フレームを受信した場合は、チェックサム・エラーとして扱います。

STX	LEN	D1	D2	D3	D4	SUM	ETX
02H	04H	FFH	80H	40H	22H	1AH	03H

本来なら 1BH

2.1 コマンド・フレーム送信処理

コマンド・フレームを送信する処理のフロー・チャートについては、4.1 コマンド・フレーム送信処理のフロー・チャートをお読みください。

2.2 データ・フレーム送信処理

データ・フレームとして送信するものは、書き込みデータ・フレーム（ユーザ・プログラム）、ベリファイ・データ・フレーム（ユーザ・プログラム）、セキュリティ・データ・フレーム（セキュリティ・フラグ）があります。

データ・フレームを送信する処理のフロー・チャートについては、4.2 データ・フレーム送信処理のフロー・チャートをお読みください。

2.3 データ・フレーム受信処理

データ・フレームとして受信するものは、ステータス・フレーム、シリコン・シグネチャ・データ・フレーム、バージョン・データ・フレーム、チェックサム・データ・フレームがあります。

データ・フレームを受信する処理のフロー・チャートについては、4.3 データ・フレーム受信処理のフロー・チャートをお読みください。

第3章 コマンド処理説明

3.1 Resetコマンド

3.1.1 説 明

通信方式設定後に、プログラマと78K0/Kx2-L間の通信が確立されたことを確認します。

プログラマと78K0/Kx2-Lは、同じボー・レートである必要があります。そのため、シリアル・プログラミング・モードの動作モード遷移後の最初にResetコマンド処理を実行して同期確認を行います。なお、78K0/Kx2-Lのボー・レート初期値は125000 bpsです。

3.1.2 コマンド・フレームとステータス・フレーム

Resetコマンドのコマンド・フレームは図3 - 1、そのコマンドに対するステータス・フレームは図3 - 2のようになります。

図3 - 1 Resetコマンド・フレーム（プログラマから78K0/Kx2-Lへ）

SOH	LEN	COM	SUM	ETX
01H	01H	00H(Reset)	Checksum	03H

図3 - 2 Resetコマンドに対するステータス・フレーム（78K0/Kx2-Lからプログラマへ）

STX	LEN	Data	SUM	ETX
02H	1	ST1	Checksum	03H

備考 ST1 : 同期検出結果

プログラマと78K0/Kx2-L間の処理手順チャート、コマンド処理のフロー・チャート、サンプル・プログラムについては、4.4 Resetコマンドをお読みください。

3.2 Baud Rate Setコマンド

3.2.1 説明

UART通信でのボー・レートの変更を行います（初期値125000 bps）。

ボー・レートの変更を行わない場合も、同期処理後、必ず、本コマンドを実行してください。実行しなかった場合、以降のコマンドが正常に実行できません。

Baud Rate Setコマンドのあとには、変更したボー・レートでの同期確認のためにResetコマンドを実行する必要があります。

3.2.2 コマンド・フレームとステータス・フレーム

Baud Rate Setコマンドのコマンド・フレームは図3 - 3，そのコマンドに対するステータス・フレームは図3 - 4のようになります。

図3 - 3 Baud Rate Setコマンド・フレーム（プログラマから78K0/Kx2-Lへ）

SOH	LEN	COM	コマンド情報 ^注					SUM	ETX
01H	06H	9AH	D01	D02H	D02L	D03	D04	Checksum	03H

注 コマンド情報の設定の詳細は表3 - 1を参照してください。表3 - 1以外のデータを設定した場合、タイムアウト・エラーとなります。

タイムアウト・エラーが発生した場合は、ハードウェア・リセットを実行し、再度フラッシュ・メモリ・プログラミング・モードに設定してください。

備考 D01 : 無効データ
D02H, D02L : ボー・レート設定
D03 : 無効データ
D04 : 無効データ

表3 - 1 コマンド情報の設定

ボー・レート	D01	D02H	D02L	D03	D04
125000 bps	FFH固定	00H	00H	FFH固定	FFH固定
250000 bps		00H	01H		
500000 bps		00H	02H		

図3 - 4 Baud Rate Setコマンドに対するステータス・フレーム（78K0/Kx2-Lからプログラマへ）

STX	LEN	Data	SUM	ETX
02H	01H	ST1	checksum	03H

備考 ST1 : 同期検出結果

プログラマと78K0/Kx2-L間の処理手順チャート，コマンド処理のフロー・チャート，サンプル・プログラムについては，4.5 Baud Rate Setコマンドをお読みください。

3.3 Chip Eraseコマンド

3.3.1 説 明

全フラッシュ・メモリの内容を消去します。また、チップ消去処理によりセキュリティ設定処理で設定されたすべての情報を初期化できます。ただし、セキュリティ設定により消去禁止となっている場合は消去できません（3.11 Security Setコマンド参照）。

3.3.2 コマンド・フレームとステータス・フレーム

Chip Eraseコマンドのコマンド・フレームは図3 - 5、そのコマンドに対するステータス・フレームは、図3 - 6のようになります。

図3 - 5 Chip Eraseコマンド・フレーム（プログラマから78K0/Kx2-Lへ）

SOH	LEN	COM	SUM	ETX
01H	01H	20H (Chip Erase)	Checksum	03H

図3 - 6 Chip Eraseコマンドに対するステータス・フレーム（78K0/Kx2-Lからプログラマへ）

STX	LEN	Data	SUM	ETX
02H	01H	ST1	Checksum	03H

備考 ST1 : チップ消去結果

プログラマと78K0/Kx2-L間の処理手順チャート、コマンド処理のフロー・チャート、サンプル・プログラムについては、4.6 Chip Eraseコマンドをお読みください。

3. 4 Block Eraseコマンド

3. 4. 1 説 明

指定したブロック番号のフラッシュ・メモリの内容を消去します。

ブロックの指定は、消去開始ブロックの先頭アドレスから、消去終了ブロックの最終アドレスの指定で行い、連続した複数のブロックの設定が可能です。

ただし、セキュリティ設定により消去禁止となっている場合は消去できません(3. 11 Security Setコマンド参照)。

3. 4. 2 コマンド・フレームとステータス・フレーム

Block Eraseコマンドのコマンド・フレームは図3 - 7、そのコマンドに対するステータス・フレームは図3 - 8のようになります。

図3 - 7 Block Eraseコマンド・フレーム (プログラマから78K0/Kx2-Lへ)

SOH	LEN	COM	コマンド情報						SUM	ETX
01H	07H	22H (Block Erase)	SAH	SAM	SAL	EAH	EAM	EAL	Checksum	03H

備考 SAH - SAL : ブロック消去開始アドレス (任意のブロックの先頭アドレス)

SAH : 開始アドレスHigh (ビット23 - ビット16)

SAM : 開始アドレスMiddle (ビット15 - ビット8)

SAL : 開始アドレスLow (ビット7 - ビット0)

EAH - EAL : ブロック消去終了アドレス (任意のブロックの最終アドレス)

EAH : 最終アドレスHigh (ビット23 - ビット16)

EAM : 最終アドレスMiddle (ビット15 - ビット8)

EAL : 最終アドレスLow (ビット7 - ビット0)

図3 - 8 Block Eraseコマンドに対するステータス・フレーム (78K0/Kx2-Lからプログラマへ)

STX	LEN	Data	SUM	ETX
02H	01H	ST1	Checksum	03H

備考 ST1 : ブロック消去結果

プログラマと78K0/Kx2-L間の処理手順チャート、コマンド処理のフロー・チャート、サンプル・プログラムについては、4. 7 Block Eraseコマンドをお読みください。

3.5 Programmingコマンド

3.5.1 説 明

書き込み開始アドレス、書き込み終了アドレスを送信したあとに、書き込みデータを送信しユーザ・プログラムをフラッシュ・メモリに書き込みます。最終データ送信後、書き込みが終了すると内部ペリファイを実行します。

書き込み開始/終了アドレスは、ブロックの開始/終了アドレス単位でのみ設定できます。

最終データ送信後のステータス・フレーム（ST1，ST2）が両方ともACKであれば、78K0/Kx2-Lのファームウェアは自動的に内部ペリファイを実行するので、さらにこの内部ペリファイに対するステータスの確認が必要となります。

3.5.2 コマンド・フレームとステータス・フレーム

Programmingコマンドのコマンド・フレームは図3 - 9，そのコマンドに対するステータス・フレームは図3 - 10のようになります。

図3 - 9 Programmingコマンド・フレーム（プログラマから78K0/Kx2-Lへ）

SOH	LEN	COM	コマンド情報						SUM	ETX
01H	07H	40H (Programming)	SAH	SAM	SAL	EAH	EAM	EAL	Checksum	03H

備考 SAH - SAL : 書き込み開始アドレス

EAH - EAL : 書き込み終了アドレス

図3 - 10 Programmingコマンドに対するステータス・フレーム（78K0/Kx2-Lからプログラマへ）

STX	LEN	Data	SUM	ETX
02H	01H	ST1(a)	Checksum	03H

備考 ST1(a) : コマンド受信結果

3.5.3 データ・フレームとステータス・フレーム

書き込みを行うデータのデータ・フレームは図3 - 11，そのデータに対するステータス・フレームは図3 - 12のようになります。

図3 - 11 書き込みを行うデータ・フレーム（プログラマから78K0/Kx2-Lへ）

STX	LEN	Data	SUM	ETX/ETB
02H	00H-FFH (00H=256)	Write Data	Checksum	03H/17H

備考 Write Data : 書き込むユーザ・プログラム

図3 - 12 データ・フレームに対するステータス・フレーム（78K0/Kx2-Lからプログラマへ）

STX	LEN	Data		SUM	ETX
02H	02H	ST1(b)	ST2(b)	Checksum	03H

備考 ST1(b) : データ受信確認結果

ST2(b) : 書き込み結果

3.5.4 全データ転送完了とステータス・フレーム

全データ転送完了後のステータス・フレームは図3 - 13のようになります。

図3 - 13 全データ転送完了後のステータス・フレーム（78K0/Kx2-Lからプログラマへ）

STX	LEN	Data		SUM	ETX
02H	01H	ST1(c)		Checksum	03H

備考 ST1(c) : 内部ベリファイ結果

プログラマと78K0/Kx2-L間の処理手順チャート，コマンド処理のフロー・チャート，サンプル・プログラムについては，4.8 Programmingコマンドをお読みください。

3.6 Verifyコマンド

3.6.1 説明

指定したアドレス範囲のデータに対して、プログラマから送信したデータと78K0/Kx2-Lから読み出したデータ（リード・レベル）を比較し、一致しているかを確認します。

ベリファイ開始アドレス/ベリファイ終了アドレスは、ブロックの開始アドレス/終了アドレス単位でのみ設定できます。

3.6.2 コマンド・フレームとステータス・フレーム

Verifyコマンドのコマンド・フレームは図3 - 14、そのコマンドに対するステータス・フレームは図3 - 15のようになります。

図3 - 14 Verifyコマンド・フレーム（プログラマから78K0/Kx2-Lへ）

SOH	LEN	COM	コマンド情報						SUM	ETX
01H	07H	13H (Verify)	SAH	SAM	SAL	EAH	EAM	EAL	Checksum	03H

備考 SAH - SAL : ベリファイ開始アドレス
EAH - EAL : ベリファイ終了アドレス

図3 - 15 Verifyコマンドに対するステータス・フレーム（78K0/Kx2-Lからプログラマへ）

STX	LEN	Data	SUM	ETX
02H	01H	ST1(a)	Checksum	03H

備考 ST1(a) : コマンド受信結果

3.6.3 データ・フレームとステータス・フレーム

ベリファイを行うデータのデータ・フレームは図3 - 16、そのデータに対するステータス・フレームは図3 - 17のようになります。

図3 - 16 ベリファイを行うデータのデータ・フレーム（プログラマから78K0/Kx2-Lへ）

STX	LEN	Data	SUM	ETX/ETB
02H	00H-FFH (00H=256)	Verify Data	Checksum	03H/17H

備考 Verify Data : ベリファイを行うユーザ・プログラム

図3 - 17 データ・フレームに対するステータス・フレーム (78K0/Kx2-Lからプログラマへ)

STX	LEN	Data		SUM	ETX
02H	02H	ST1 (b)	ST2 (b)	Checksum	03H

備考 ST1(b) : データ受信確認結果
ST2(b) : ベリファイ結果[※]

注 ベリファイ結果は指定したアドレス範囲の途中でベリファイ・エラーが発生しても、ステータスとしては必ずACKを返し、最終データのベリファイ結果にすべてのエラーが反映されます。したがって、指定したアドレス範囲すべてのベリファイが終了した時点でのみ、ベリファイ・エラーが発生したかどうかを確認できます。

プログラマと78K0/Kx2-L間の処理手順チャート、コマンド処理のフロー・チャート、サンプル・プログラムについては、4.9 Verifyコマンドをお読みください。

3.7 Block Blank Checkコマンド

3.7.1 説明

指定したブロック番号のフラッシュ・メモリのデータがブランク（消去状態）であるかを確認します。

ブロックの指定は、ブランク・チェック開始ブロックの先頭アドレスから、ブランク・チェック終了ブロックの最終アドレスの指定で行い、連続した複数のブロックの設定が可能です。

3.7.2 コマンド・フレームとステータス・フレーム

Block Blank Checkコマンドのコマンド・フレームは図3 - 18、そのコマンドに対するステータス・フレームは図3 - 19のようになります。

図3 - 18 Block Blank Checkコマンド・フレーム（プログラマから78K0/Kx2-Lへ）

SOH	LEN	COM	コマンド情報							SUM	ETX
01H	08H	32H (Block Blank Check)	SAH	SAM	SAL	EAH	EAM	EAL	D01	Checksum	03H

備考 SAH - SAL : ブロック・ブランク・チェック開始アドレス（任意のブロックの先頭アドレス）
 SAH : 開始アドレスHigh（ビット23 - ビット16）
 SAM : 開始アドレスMiddle（ビット15 - ビット8）
 SAL : 開始アドレスLow（ビット7 - ビット0）
 EAH - EAL : ブロック・ブランク・チェック終了アドレス（任意のブロックの最終アドレス）
 EAH : 最終アドレスHigh（ビット23 - ビット16）
 EAM : 最終アドレスMiddle（ビット15 - ビット8）
 EAL : 最終アドレスLow（ビット7 - ビット0）
 D01 : 00H : 単独でブロック・ブランク・チェックを行う場合。
 01H : チップ消去前に全領域のブランク・チェックをする場合

図3 - 19 Block Blank Checkコマンドに対するステータス・フレーム（78K0/Kx2-Lからプログラマへ）

STX	LEN	Data	SUM	ETX
02H	01H	ST1	Checksum	03H

備考 ST1 : ブロック・ブランク・チェック結果

プログラマと78K0/Kx2-L間の処理手順チャート、コマンド処理のフロー・チャート、サンプル・プログラムについては、4. 10 Block Blank Checkコマンドをお読みください。

3. 8 Silicon Signatureコマンド

3. 8. 1 説 明

デバイスの書き込みプロトコル情報(シリコン・シグネチャ)やセキュリティ・フラグ情報を読み出します。

3. 8. 2 コマンド・フレームとステータス・フレーム

Silicon Signatureコマンドのコマンド・フレームは図3 - 20 , そのコマンドに対するステータス・フレームは図3 - 21のようになります。

図3 - 20 Silicon Signatureコマンド・フレームのフォーマット (プログラマから78K0/Kx2-Lへ)

SOH	LEN	COM	SUM	ETX
01H	01H	C0H (Silicon Signature)	Checksum	03H

図3 - 21 Silicon Signatureコマンドに対するステータス・フレーム (78K0/Kx2-Lからプログラマへ)

STX	LEN	Data	SUM	ETX
02H	01H	ST1	Checksum	03H

備考 ST1 : コマンド受信結果

3.8.3 シリコン・シグネチャ・データ・フレーム

シリコン・シグネチャ・データのデータ・フレームは図3 - 22のようになります。

図3 - 22 シリコン・シグネチャ・データ・フレーム (78K0/Kx2-Lからプログラマへ)

STX	LEN	Data							
02H	1BH	VEN	MET	MSC	DEC1	DEC2	DEC3	UAE(3)	DEV(10)

Data (続き)			SUM	ETX
SCF	BOT	RES(6)	checksum	03H

- 備考1. LEN : データ長
 VEN : ベンダー・コード (NEC : 10H)
 MET : マクロ拡張コード
 MSC : マクロ機能コード
 DEC1 : デバイス拡張コード1
 DEC2 : デバイス拡張コード2
 DEC3 : デバイス拡張コード3
 UAE : ユーザ・フラッシュROM最終アドレス (3バイト)
 DEV : デバイス名 (10バイト)
 SCF : セキュリティ・フラグ情報
 BOT : ブート・ブロック番号
 RES : 予約 (6バイト)
2. 上記ベンダー・コード (VEN) , 拡張コード (MET) , 機能コード (MSC) , デバイス拡張コード1 (DEC1) , デバイス拡張コード2 (DEC2) , デバイス拡張コード3 (DEC3) は , 下位7ビットをデータ本体 , 上位1ビットを奇数パリティとして使用します。次ページに例を示します。

表3 - 2 シリコン・シグネチャ・データの例 (μ PD78F0588 (78K0/KC2-L) の場合)

フィールド名	内 容	長さ (バイト)	シグネチャ・データの例	実際の値	パリティ 付加
VEN	ベンダー・コード (NEC)	1	10H (00010000B)	10H	あり
MET	マクロ拡張コード	1	DFH (11011111B)	DFH	あり
MSC	マクロ機能コード	1	04H (01000000B)	04H	あり
DEC1	デバイス拡張コード1	1	FEH (11111110B)	FEH	あり
DEC2	デバイス拡張コード2	1	7FH (01111111B)	7FH	あり
DEC3	デバイス拡張コード3	1	7FH (01111111B)	7FH	あり
UAE	ユーザ・フラッシュROM最終 アドレス	3	FFH (11111111B)	007FFFH	なし
			7FH (01111111B)		
			00H (00000000B)		
DEV	デバイス名	10	44H (01000100B) = 'D'	'D'	なし
			37H (00110111B) = '7'	'7'	
			38H (00111000B) = '8'	'8'	
			46H (01000110B) = 'F'	'F'	
			30H (00110001B) = '0'	'0'	
			35H (00110000B) = '5'	'5'	
			38H (00111000B) = '8'	'8'	
			38H (00111000B) = '8'	'8'	
			20H (00100000B) = ''	''	
			20H (00100000B) = ''	''	
SCF	セキュリティ・フラグ情報	1	任意	左欄に同じ	なし
BOT	ブート・ブロック番号 (固定値)	1	03H (00000011B)	03H	なし
RES	予約	6	FFFFFFFFFFFFH	FFFFFFFFFFFFH	なし

プログラマと78K0/Kx2-L間の処理手順チャート，コマンド処理のフロー・チャート，サンプル・プログラムについては，4. 11 Silicon Signatureコマンドをお読みください。

3.8.4 シリコン・シグネチャー一覧

(1) 78K0/Kx2-Lのシリコン・シグネチャ・データ一覧

表3 - 3 78K0/Kx2-Lのシリコン・シグネチャ・データ一覧

項 目	内 容	長さ (バイト)	データ (Hex)
ベンダー・コード	NEC	1	10
拡張コード	拡張コード	1	DF
機能情報	機能情報	1	04
デバイス情報	デバイス情報	3	FE
			7F
			7F
内蔵フラッシュROMの 最終アドレス	アドレスの下位バイトから送信	3	注1
デバイス名 (μPD)	78F0550, 78F0551, 78F0552, 78F0555, 78F0556, 78F0557 78F0560, 78F0561, 78F0562, 78F0565, 78F0566, 78F0567 78F0571, 78F0572, 78F0573, 78F0576, 78F0577, 78F0578 78F0581, 78F0582, 78F0583, 78F0586, 78F0587, 78F0588	10	注2
セキュリティ情報	セキュリティ情報	1	任意
ブート・ブロック番号	現在, 選択されているブート・クラスタの最終ブロック番号	1	03
予約	予約	6	FFFFFFFFFFFF

注1. 内蔵フラッシュROMの最終アドレス・リスト

項 目	内 容	長さ (バイト)	データ (Hex)
内蔵フラッシュROM の最終アドレス	4 Kバイト (0FFFH)	3	FF0F00
	8 Kバイト (1FFFH)		FF1F00
	16 Kバイト (3FFFH)		FF3F00
	32 Kバイト (7FFFH)		FF7F00

(注2は次ページにあります。)

注2. デバイス名リストを次に示します。

デバイス名リスト

(1/2)

愛 称	デバイス名	長さ (バイト)	実際の値									
			〔 上段：シグネチャ・データ 〕									
			下段：文字コード									
78K0/KY2-L	D78F0550	10	44	37	38	46	30	35	35	30	20	20
			D	7	8	F	0	5	5	0	-	-
	D78F0551		44	37	38	46	30	35	35	31	20	20
			D	7	8	F	0	5	5	1	-	-
	D78F0552		44	37	38	46	30	35	35	32	20	20
			D	7	8	F	0	5	5	2	-	-
	D78F0555		44	37	38	46	30	35	35	35	20	20
			D	7	8	F	0	5	5	5	-	-
	D78F0556		44	37	38	46	30	35	35	36	20	20
			D	7	8	F	0	5	5	6	-	-
D78F0557	44		37	38	46	30	35	35	37	20	20	
	D		7	8	F	0	5	5	7	-	-	
78K0/KA2-L	D78F0560		44	37	38	46	30	35	36	30	20	20
			D	7	8	F	0	5	6	0	-	-
	D78F0561		44	37	38	46	30	35	36	31	20	20
			D	7	8	F	0	5	6	1	-	-
	D78F0562		44	37	38	46	30	35	36	32	20	20
			D	7	8	F	0	5	6	2	-	-
	D78F0565		44	37	38	46	30	35	36	35	20	20
			D	7	8	F	0	5	6	5	-	-
	D78F0566	44	37	38	46	30	35	36	36	20	20	
		D	7	8	F	0	5	6	6	-	-	
D78F0567	44	37	38	46	30	35	36	37	20	20		
	D	7	8	F	0	5	6	7	-	-		
78K0/KB2-L	D78F0571	44	37	38	46	30	35	37	31	20	20	
		D	7	8	F	0	5	7	1	-	-	
	D78F0572	44	37	38	46	30	35	37	32	20	20	
		D	7	8	F	0	5	7	2	-	-	
	D78F0573	44	37	38	46	30	35	37	33	20	20	
		D	7	8	F	0	5	7	3	-	-	
	D78F0576	44	37	38	46	30	35	37	36	20	20	
		D	7	8	F	0	5	7	6	-	-	
	D78F0577	44	37	38	46	30	35	37	37	20	20	
		D	7	8	F	0	5	7	7	-	-	
D78F0578	44	37	38	46	30	35	37	38	20	20		
	D	7	8	F	0	5	7	8	-	-		

(2/2)

愛 称	デバイス名	長さ (バイト)	実際の値									
			〔 上段：シグネチャ・データ 下段：文字コード 〕									
78K0/KC2-L	D78F0581	10	44	37	38	46	30	35	38	31	20	20
			D	7	8	F	0	5	8	1	-	-
	D78F0582		44	37	38	46	30	35	38	32	20	20
			D	7	8	F	0	5	8	2	-	-
	D78F0583		44	37	38	46	30	35	38	33	20	20
			D	7	8	F	0	5	8	3	-	-
	D78F0586		44	37	38	46	30	35	38	36	20	20
			D	7	8	F	0	5	8	6	-	-
	D78F0587		44	37	38	46	30	35	38	37	20	20
			D	7	8	F	0	5	8	7	-	-
	D78F0588		44	37	38	46	30	35	38	38	20	20
			D	7	8	F	0	5	8	8	-	-

(2) 78K0/Ix2のシリコン・シグネチャ・データ一覧

表3 - 4 78K0/Ix2のシリコン・シグネチャ・データ一覧

項 目	内 容	長さ (バイト)	データ (Hex)
ベンダー・コード	NEC	1	10
拡張コード	拡張コード	1	DF
機能情報	機能情報	1	04
デバイス情報	デバイス情報	3	FE
			7F
			7F
内蔵フラッシュROMの 最終アドレス	アドレスの下位バイトから送信	3	注1
デバイス名 (μPD)	78F0740, 78F0741, 78F0742, 78F0750, 78F0751, 78F0752 78F0743, 78F0744, 78F0753, 78F0754 78F0745, 78F0746, 78F0755, 78F0756	10	注2
セキュリティ情報	セキュリティ情報	1	任意
ブート・ブロック番号	現在, 選択されているブート・クラスタの最終ブロック番号	1	03
予約	予約	6	FFFFFFFFFFFF

注1. 内蔵フラッシュROMの最終アドレス・リスト

項 目	内 容	長さ (バイト)	データ (Hex)
内蔵フラッシュROM の最終アドレス	4 Kバイト (0FFFFH)	3	FF0F00
	8 Kバイト (1FFFFH)		FF1F00
	16 Kバイト (3FFFFH)		FF3F00

(注2は次ページにあります。)

注2. デバイス名リストを次に示します。

デバイス名リスト

愛 称	デバイス名	長さ (バイト)	実際の値									
			〔 上段：シグネチャ・データ 〕									
			下段：文字コード									
78K0/IY2	D78F0740	10	44	37	38	46	30	37	34	30	20	20
			D	7	8	F	0	7	4	0	-	-
	D78F0741		44	37	38	46	30	37	34	31	20	20
			D	7	8	F	0	7	4	1	-	-
	D78F0742		44	37	38	46	30	37	34	32	20	20
			D	7	8	F	0	7	4	2	-	-
	D78F0750		44	37	38	46	30	37	35	30	20	20
			D	7	8	F	0	7	5	0	-	-
	D78F0751		44	37	38	46	30	37	35	31	20	20
			D	7	8	F	0	7	5	1	-	-
D78F0752	44	37	38	46	30	37	35	32	20	20		
	D	7	8	F	0	7	5	2	-	-		
78K0/IA2	D78F0743	44	37	38	46	30	37	34	33	20	20	
		D	7	8	F	0	7	4	3	-	-	
	D78F0744	44	37	38	46	30	37	34	34	20	20	
		D	7	8	F	0	7	4	4	-	-	
	D78F0753	44	37	38	46	30	37	35	33	20	20	
		D	7	8	F	0	7	5	3	-	-	
	D78F0754	44	37	38	46	30	37	35	34	20	20	
		D	7	8	F	0	7	5	4	-	-	
78K0/IB2	D78F0745	44	37	38	46	30	37	34	35	20	20	
		D	7	8	F	0	7	4	5	-	-	
	D78F0746	44	37	38	46	30	37	34	36	20	20	
		D	7	8	F	0	7	4	6	-	-	
	D78F0755	44	37	38	46	30	37	35	35	20	20	
		D	7	8	F	0	7	5	5	-	-	
	D78F0756	44	37	38	46	30	37	35	36	20	20	
		D	7	8	F	0	7	5	6	-	-	

3.9 Version Getコマンド

3.9.1 説明

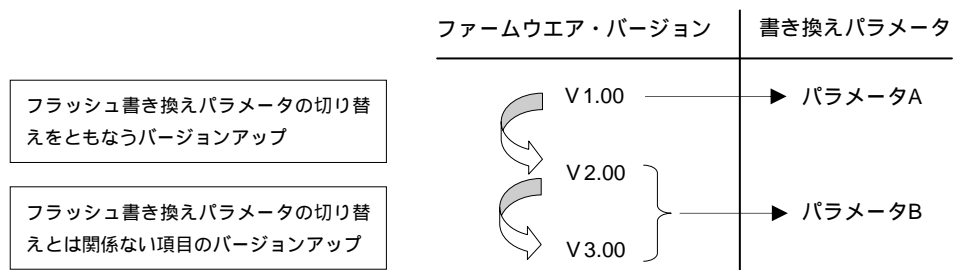
78K0/Kx2-Lのデバイス・バージョン、ファームウェア・バージョン情報を取得します。

デバイス・バージョンは、00H固定です。

書き換え用パラメータを78K0/Kx2-Lのファームウェア・バージョンに従い、切り替える必要がある場合に、このコマンドを使用します。

注意 フラッシュ書き換え用パラメータの変更とは関係ないファームウェア改版時も、ファームウェア・バージョンが更新される場合があります（このとき、ファームウェア・バージョン更新の通知は行いません）。

例 ファームウェア・バージョンと書き換えパラメータ



3.9.2 コマンド・フレームとステータス・フレーム

Version Getコマンドのコマンド・フレームは図3 - 23，そのコマンドに対するステータス・フレームは図3 - 24のようになります。

図3 - 23 Version Getコマンド・フレーム（プログラマから78K0/Kx2-Lへ）

SOH	LEN	COM	SUM	ETX
01H	01H	C5H (Version Get)	Checksum	03H

図3 - 24 Version Getコマンドに対するステータス・フレーム（78K0/Kx2-Lからプログラマへ）

STX	LEN	Data	SUM	ETX
02H	01H	ST1	Checksum	03H

備考 ST1 : コマンド受信結果

3.9.3 バージョン・データ・フレーム

バージョン・データのデータ・フレームは図3 - 25のようになります。

図3 - 25 バージョン・データ・フレーム (78K0/Kx2-Lからプログラマへ)

STX	LEN	Data						SUM	ETX
02H	06H	DV1	DV2	DV3	FV1	FV2	FV3	Checksum	03H

備考 DV1 : デバイス・バージョン整数値 (00H固定)
DV2 : デバイス・バージョン小数点第一位 (00H固定)
DV3 : デバイス・バージョン小数点第二位 (00H固定)
FV1 : ファームウェア・バージョン整数値
FV2 : ファームウェア・バージョン小数点第一位
FV3 : ファームウェア・バージョン小数点第二位

プログラマと78K0/Kx2-L間の処理手順チャート，コマンド処理のフロー・チャート，サンプル・プログラムについては，4. 12 Version Getコマンドをお読みください。

3. 10 Checksumコマンド

3. 10. 1 説 明

指定された領域のデータのチェックサム・データを取得します。

チェックサム計算の開始 / 終了アドレスは、フラッシュ・メモリの先頭からブロック単位（2 Kバイト）ごとの固定アドレスを指定してください。

チェックサム・データは、指定されたアドレス範囲のデータを1バイト単位で順次初期値0000Hから引き算したものです。

3. 10. 2 コマンド・フレームとステータス・フレーム

Checksumコマンドのコマンド・フレームは図3 - 26，そのコマンドに対するステータス・フレームは図3 - 27のようになります。

図3 - 26 Checksumコマンド・フレーム（プログラマから78K0/Kx2-Lへ）

SOH	LEN	COM	コマンド情報						SUM	ETX
01H	07H	B0H (Checksum)	SAH	SAM	SAL	EAH	EAM	EAL	Checksum	03H

備考 SAH-SAL : チェックサム計算開始アドレス

EAH-EAL : チェックサム計算終了アドレス

図3 - 27 Checksumコマンドに対するステータス・フレーム（78K0/Kx2-Lからプログラマへ）

STX	LEN	Data	SUM	ETX
02H	01H	ST1	Checksum	03H

備考 ST1 : コマンド受信結果

3. 10. 3 チェックサム・データ・フレーム

チェックサム・データのデータ・フレームは図3 - 28のようになります。

図3 - 28 チェックサム・データ・フレーム（78K0/Kx2-Lからプログラマへ）

STX	LEN	Data		SUM	ETX
02H	02H	CK1	CK2	Checksum	03H

備考 CK1 : チェックサム・データの上位8ビット

CK2 : チェックサム・データの低位8ビット

プログラマと78K0/Kx2-L間の処理手順チャート，コマンド処理のフロー・チャート，サンプル・プログラムについては，4. 13 Checksumコマンドをお読みください。

3. 11 Security Setコマンド

3. 11. 1 説 明

セキュリティに関する設定（書き込み，ブロック消去，チップ消去，ブート・ブロック・クラスタ書き換えの許可／禁止）を行います。Security Setコマンドでこれらの設定を行うことで，第三者からのフラッシュの書き換えを制限します。

注意 セキュリティ設定後も許可から禁止への追加設定が可能です。ただし禁止から許可への変更は行えず，実行しようとした場合，Protect error（10H）が発生します。禁止から許可への設定変更が必要な場合は，Chip Eraseコマンドの実行によって全セキュリティ・フラグの初期化をする必要があります（Block Eraseコマンドでは，セキュリティ・フラグの初期化はできません）。そのあとに設定変更を行ってください。

ただし，チップ消去禁止，またはブート・ブロック書き換え禁止の設定をした場合，チップ消去自体が不可能になり，プログラマからは消去ができなくなります。プログラマの仕様としては，チップ消去禁止の設定を行う前に，設定実行の再確認をすることを推奨します。

3. 11. 2 コマンド・フレームとステータス・フレーム

Security Setコマンドのコマンド・フレームは図3 - 29，そのコマンドに対するステータス・フレームは図3 - 30のようになります。

図3 - 29 Security Setコマンド・フレーム（プログラマから78K0/Kx2-Lへ）

SOH	LEN	COM	コマンド情報		SUM	ETX
01H	03H	A0H (Security Set)	00H (固定)	00H (固定)	Checksum	03H

図3 - 30 Security Setコマンドに対するステータス・フレーム（78K0/Kx2-Lからプログラマへ）

STX	LEN	Data	SUM	ETX
02H	01H	ST1(a)	Checksum	03H

備考 ST1(a) : コマンド受信結果

3. 11. 3 データ・フレームとステータス・フレーム

セキュリティ・データのデータ・フレームは図3 - 31，そのデータに対するステータス・フレームは図3 - 32のようになります。

図3 - 31 セキュリティ・データ・フレーム（プログラマから78K0/Kx2-Lへ）

STX	LEN	Data								SUM	ETX
02H	08H	FLG	BOT	FFH (固定)	FFH (固定)	FFH (固定)	FFH (固定)	FFH (固定)	FFH (固定)	Checksum	03H

備考 FLG : セキュリティ・フラグ

BOT : ブート・ブロック・クラスタの最終ブロック番号（03H固定）

図3 - 32 セキュリティ・データ書き込みに対するステータス・フレーム（78K0/Kx2-Lからプログラマへ）

STX	LEN	Data	SUM	ETX
02H	01H	ST1(b)	Checksum	03H

備考 ST1(b) : セキュリティ・データ書き込み結果

3. 11. 4 内部ベリファイ確認とステータス・フレーム

内部ベリファイ確認に対するステータス・フレームは図3 - 33のようになります。

図3 - 33 内部ベリファイ確認に対するステータス・フレーム（78K0/Kx2-Lからプログラマへ）

STX	LEN	Data	SUM	ETX
02H	01H	ST1(c)	Checksum	03H

備考 ST1(c) : 内部ベリファイ結果

セキュリティ・フラグ・フィールドの内容を次に示します。

表3 - 5 セキュリティ・フラグ・フィールドの内容

項 目	内 容
ビット7	1 固定
ビット6	
ビット5	
ビット4	ブート・ブロック・クラスタ書き換え禁止フラグ（1：許可，0：禁止）
ビット3	1 固定
ビット2	書き込み禁止フラグ（1：書き込み許可，0：書き込み禁止）
ビット1	ブロック消去禁止フラグ（1：ブロック消去許可，0：ブロック消去禁止）
ビット0	チップ消去禁止フラグ（1：チップ消去許可，0：チップ消去禁止）

セキュリティ・フラグ・フィールドの設定と、各動作の禁止 / 許可の関係を次に示します。

表3 - 6 セキュリティ・フラグ・フィールドと各動作の禁止 / 許可

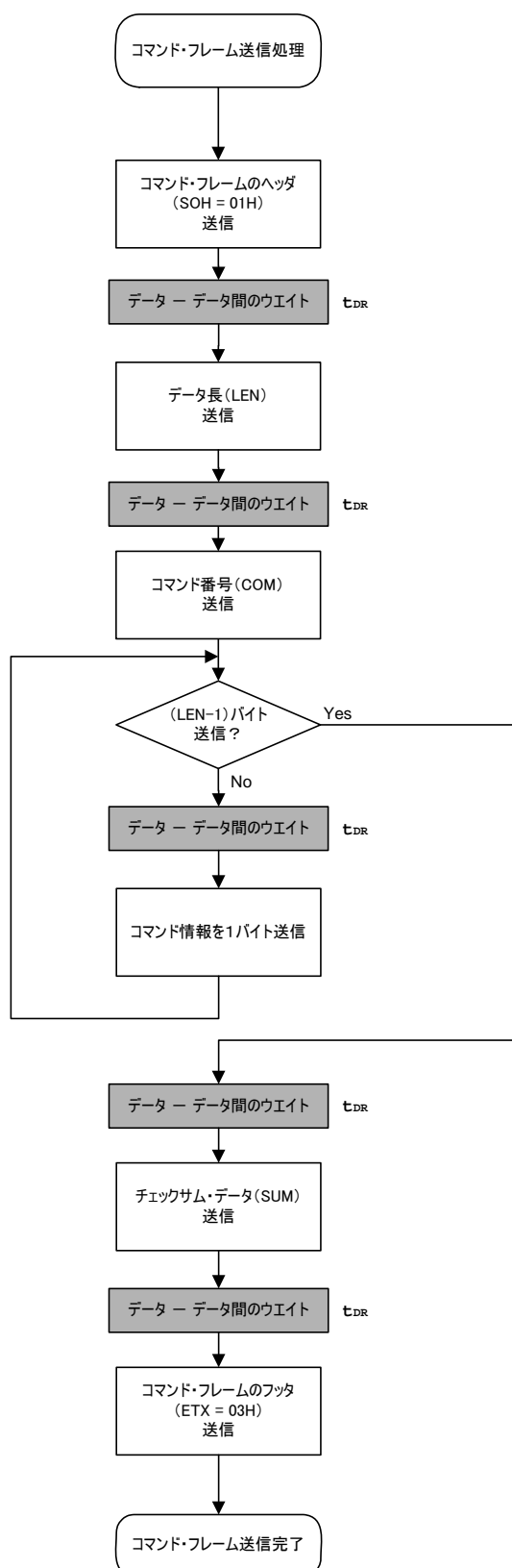
動作モード	フラッシュ・メモリ・プログラミング・モード		
<div> <div>コマンド</div> <div>セキュリティ 設定項目</div> </div>	セキュリティ設定後のコマンド動作 : 実行可能 x : 実行不可 : ブート・ブロック・クラスタへの書き込み, またはブロック消去は不可 : ブート・ブロック・クラスタ以外への書き込み, またはブロック消去は可能		
	Programming	Chip Erase	Block Erase
書き込み禁止	x		x
チップ消去禁止		x	x
ブロック消去禁止			x
ブート・ブロック・クラスタ書き換え禁止フラグ		x	

備考 セルフ・プログラミング・モードの場合については、各デバイスのユーザズ・マニュアルを参照してください。

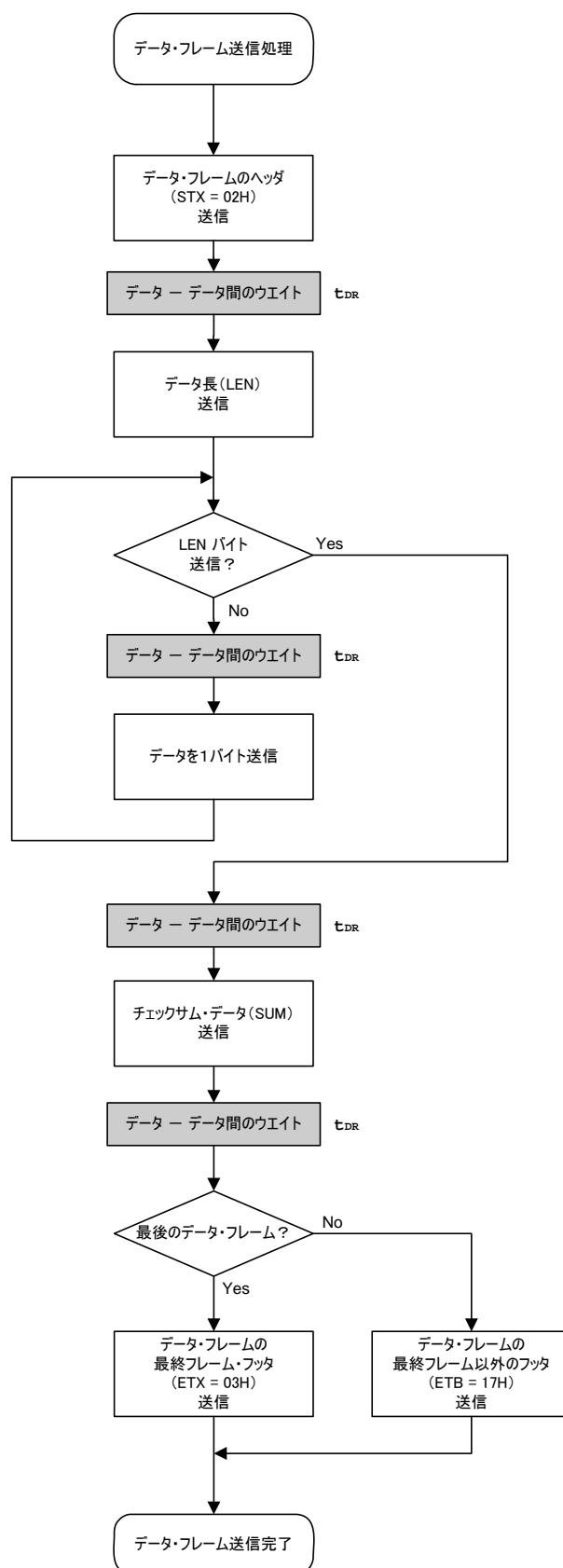
プログラマと78K0/Kx2-L間の処理手順チャート, コマンド処理のフロー・チャート, サンプル・プログラムについては、4. 14 Security Setコマンドをお読みください。

第4章 UART通信方式

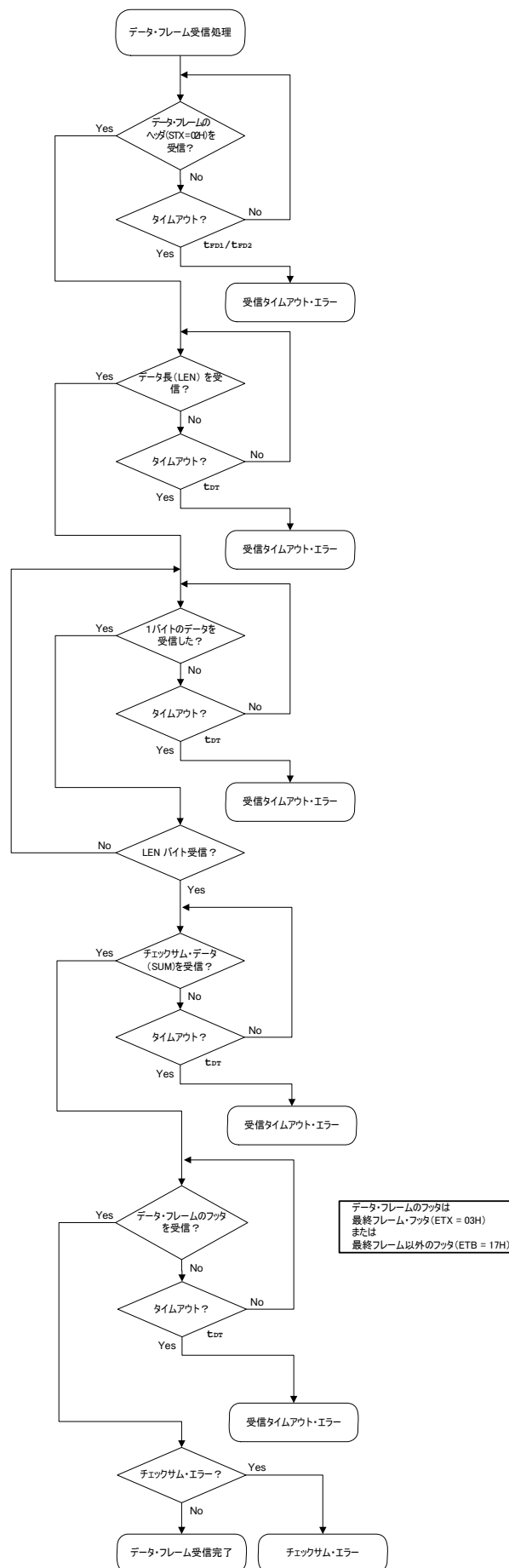
4.1 コマンド・フレーム送信処理のフロー・チャート



4.2 データ・フレーム送信処理のフロー・チャート



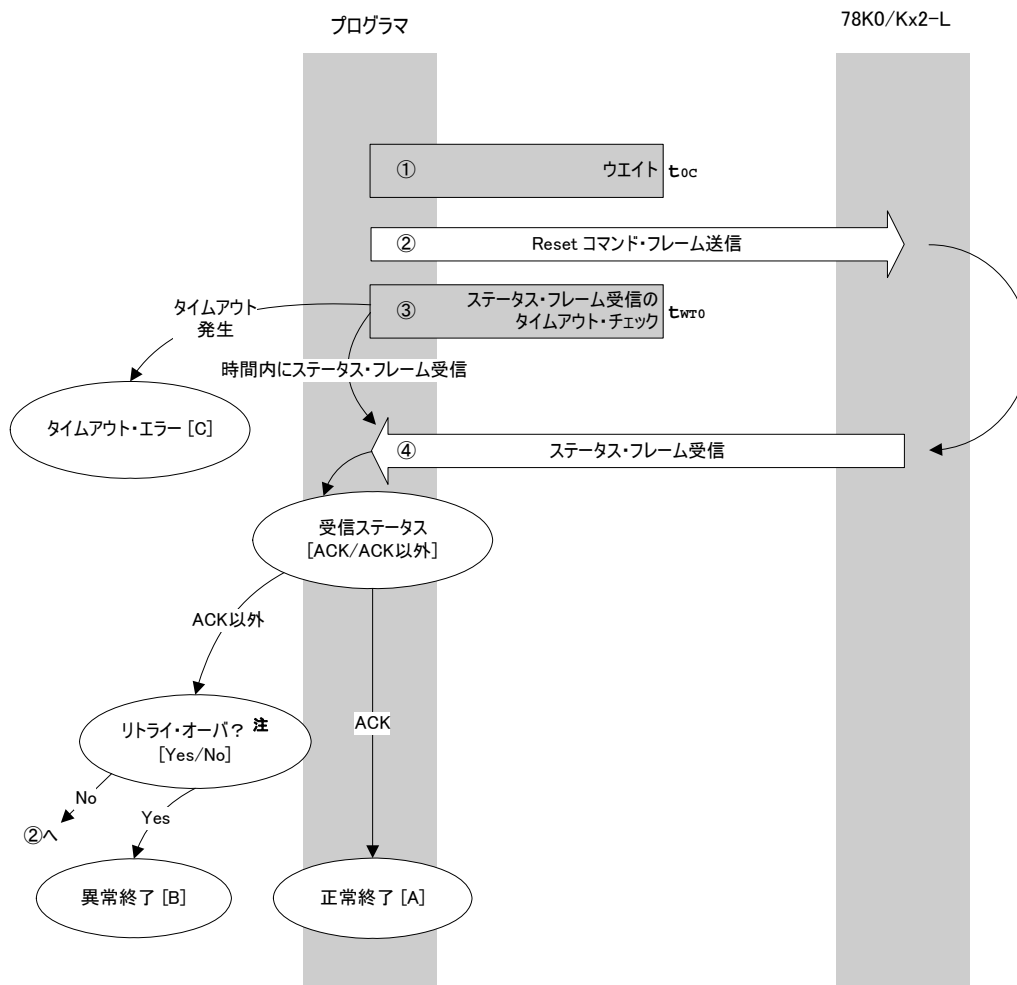
4.3 データ・フレーム受信処理のフロー・チャート



4.4 Resetコマンド

4.4.1 処理手順チャート

Resetコマンド処理手順



注 リセット・コマンドの送信は16回 (MAX.) としてください。

4.4.2 処理手順説明

ウェイトします（ウェイト時間 t_{oc} ）。

コマンド・フレーム送信処理にて **Resetコマンド** を送信します。

コマンド送信からステータス・フレーム受信までのタイムアウト・チェックを行います。

タイムアウトが発生した場合は **タイムアウト・エラー [C]** となります

（タイムアウト時間 t_{wto} ）。

ステータス・コードをチェックします。

ST1 = ACKの場合 : **正常終了 [A]** です。

ST1 = ACK以外の場合 : リトライ回数（ t_{rs} ）をチェックします。

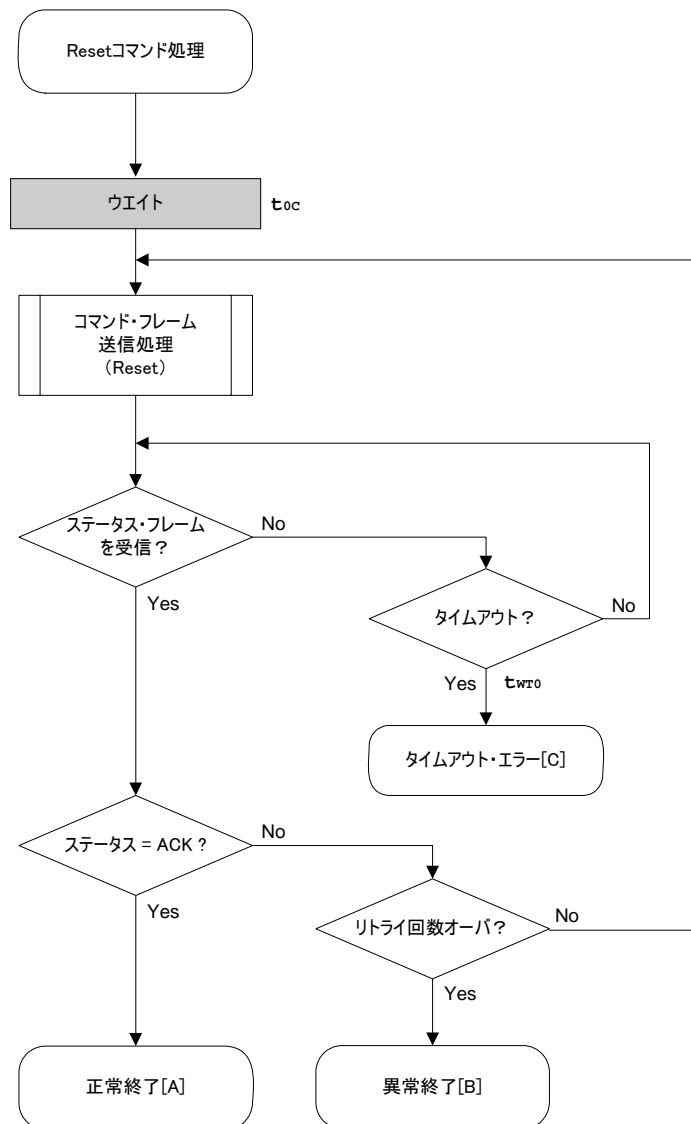
リトライ・オーバでなければ からやり直します。

リトライ・オーバであれば **異常終了 [B]** です。

4.4.3 終了時の内容

終了内容		ステータス・コード	内 容
正常終了 [A]	正常応答（ACK）	06H	コマンドが正常に実行され、プログラマと78K0/Kx2-L間で同期が取れたことを示します。
異常終了 [B]	チェックサム・エラー	07H	送信したコマンド・フレームのチェックサムが異常です。
	否定応答（NACK）	15H	コマンド・フレーム・データが異常です（データ長（LEN）不正，ETXなしなど）。
タイムアウト・エラー [C]		-	規定の時間内にステータス・フレームが受信できませんでした。

4.4.4 フロー・チャート



4.4.5 サンプル・プログラム

Resetコマンド処理のサンプル・プログラムです。

```

/*****
/*
/* Reset command
/*
/*****
/* [r] u16          ... error code
/*****
u16      fl_ua_reset(void)
{
    u16      rc;
    u32      retry;

    set_uart0_br(BR_125000);          // change to 125000bps

    // fl_wait(tCOM);                  // wait

    set_ua_dir_rx();                  // Change Mono-wire UART receive mode

    for (retry = 0; retry < tRS; retry++){

        fl_wait(t0C_MIN);             // wait

        put_cmd_ua(FL_COM_RESET, 1, fl_cmd_prm);          // send RESET command

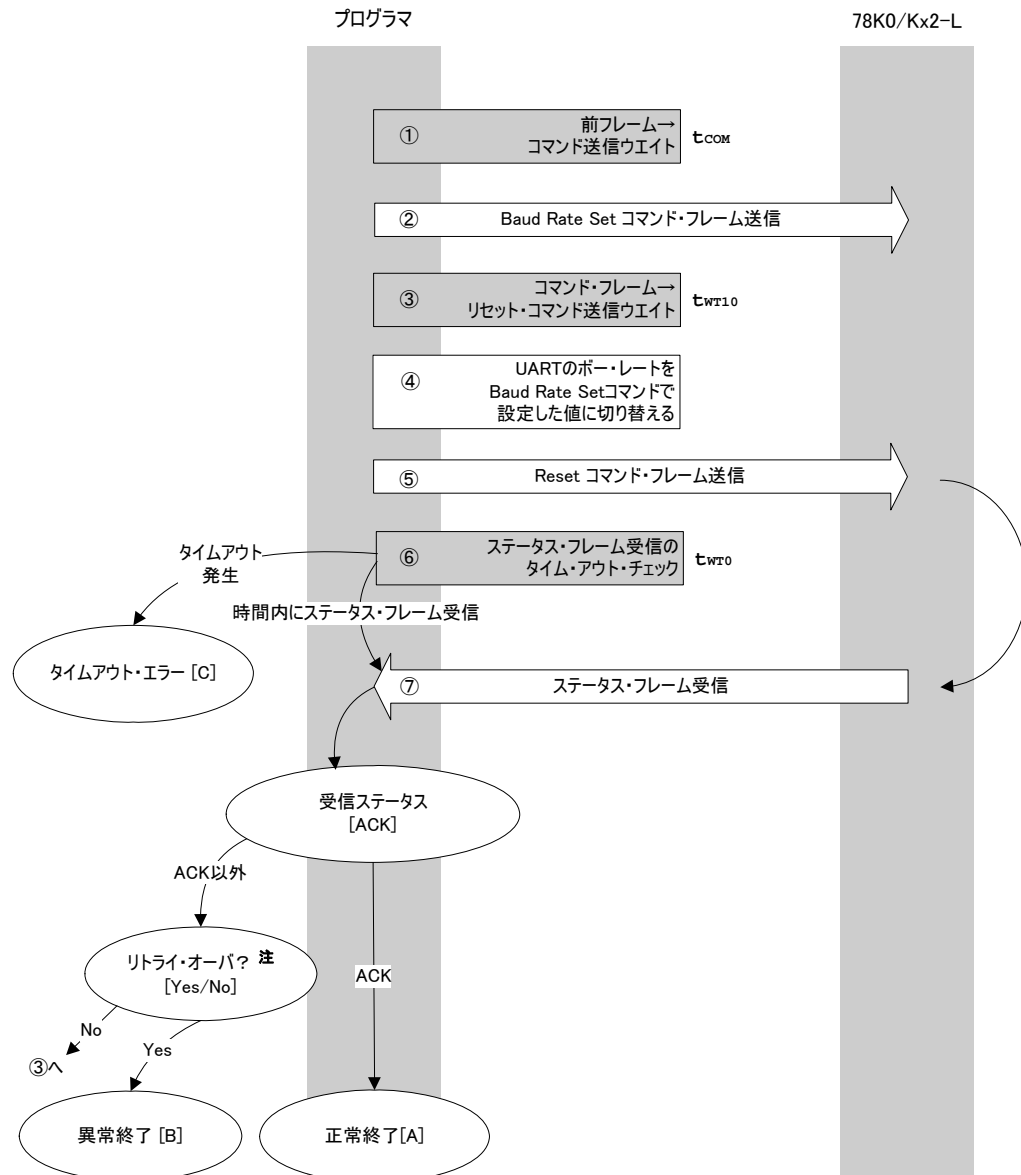
        rc = get_sfrm_ua(fl_ua_sfrm, tWT0_TO);
        if (rc == FLC_DFTO_ERR)          // t.o. ?
            break;                      // yes // case [C]
        if (rc == FLC_ACK){              // ACK ?
            break;                      // yes // case [A]
        }
        else{
            NOP();
        }
        //continue;                    // case [B] (if exit from loop)
    }
    // switch(rc) {
    //
    //      case    FLC_NO_ERR:    return rc;    break; // case [A]
    //      case    FLC_DFTO_ERR:  return rc;    break; // case [C]
    //      default:               return rc;    break; // case [B]
    // }
    return rc;
}

```

4.5 Baud Rate Setコマンド

4.5.1 処理手順チャート

Baud Rate Setコマンド処理手順



注 リセット・コマンドの送信は16回 (MAX.) としてください。

4.5.2 処理手順説明

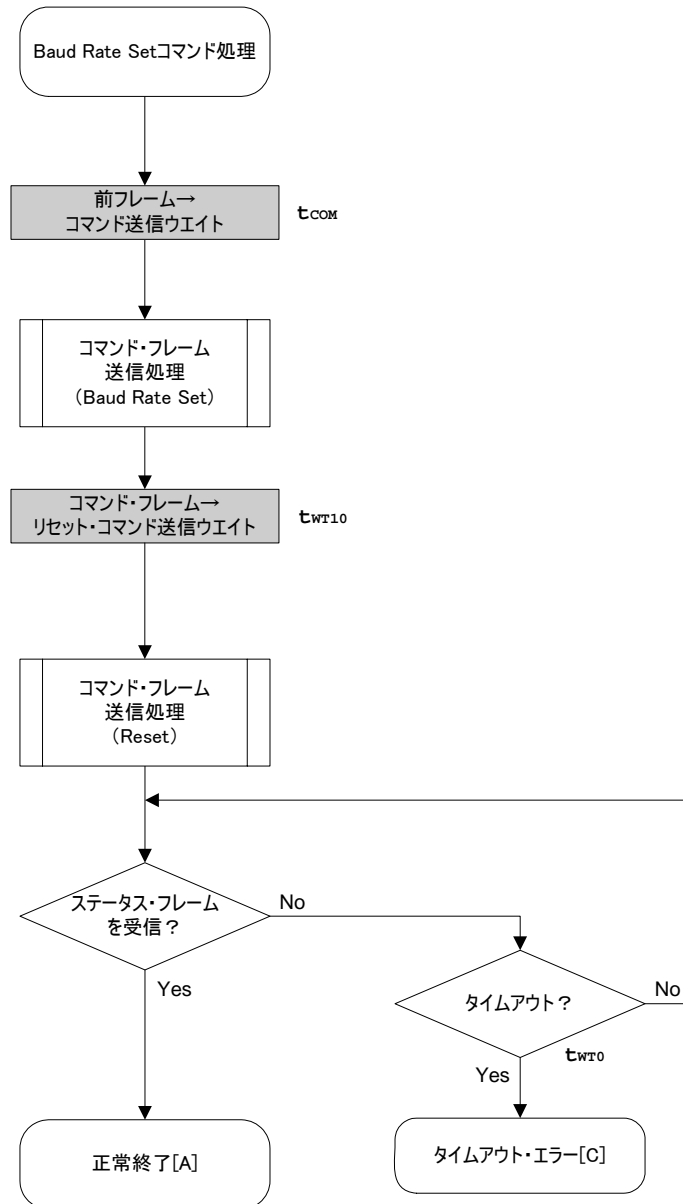
直前のフレームからコマンド送信までのウェイトをします（ウェイト時間 t_{COM} ）。
 コマンド・フレーム送信処理にて **Baud Rate Setコマンド** を送信します。
 コマンド送信からリセット・コマンド送信までのウェイトをします（ウェイト時間 t_{WT10} ）。
 UART通信のボー・レートはBaud Rate Setコマンドで設定した値に切り替えます。
 コマンド・フレーム送信処理にて **Resetコマンド** を送信します。
 コマンド送信からステータス・フレーム受信までのタイムアウト・チェックを行います。
 タイムアウト発生であれば **タイムアウト・エラー[C]** となります
 （タイムアウト時間 t_{WT0} ）。
 ステータス・コードはACKのはずですので、 **正常終了[A]** となります。

4.5.3 終了時の内容

終了内容		ステータス・コード	内 容
正常終了 [A]	正常応答 (ACK)	06H	コマンドが正常に実行され、プログラマと 78K0/Kx2-L 間で UART 通信速度の同期が取れたことを示します。
異常終了 [B]	チェックサム・エラー	07H	送信したコマンド・フレームのチェックサムが異常です。
	否定応答 (NACK)	15H	コマンド・フレーム・データが異常です（データ長 (LEN) 不正, ETX なしなど）。
タイムアウト・エラー [C] ^注		-	<p>データ・フレームの受信でタイムアウトが発生しました。</p> <p>ただし、このコマンドにおいて 78K0/Kx2-L では次の場合もこのエラーになります。</p> <ul style="list-style-type: none"> ・コマンド情報 (D01, D02H, D02L, D03, D04) が不正な場合 ・ボー・レート設定後、16 回分のコマンド・フレーム・データを受信しても Reset コマンドが検出できなかった場合

注 タイムアウト・エラーが発生した場合は、ハードウェア・リセットを実行し、再度フラッシュ・メモリ・プログラミング・モードに設定してください。

4.5.4 フロー・チャート



4.5.5 サンプル・プログラム

Baud Rate Setコマンド処理のサンプル・プログラムです。

```

/*****
/*
/*Set baudrate command
/*
/*****
/* [i] u8 brid ... baudrate ID
/* [r] ul6 ... error code
/*****
ul6      fl_ua_setbaud(u8 brid)
{
    ul6      rc;
    u8       br;
    u32      retry;

    fl_cmd_prm[0] = 0xff; // "D01" : invalid data
    fl_cmd_prm[1] = 0x00; // "D02H" : adjust by target device
    fl_cmd_prm[2] = brid; // "D02L" : (fixed value)
    fl_cmd_prm[3] = 0xff; // "D03" : invalid data
    fl_cmd_prm[4] = 0xff; // "D04" : invalid data

    switch(brid){
        default:
        case 0x00: br = BR_125000; break;
        case 0x01: br = BR_250000; break;
        case 0x02: br = BR_500000; break;
    }

    fl_wait(tCOM); // wait before sending command
    put_cmd_ua(FL_COM_SET_BAUDRATE, 1+5, fl_cmd_prm); // send "Baudrate Set"
command
    set_flbaud(br); // change baud-rate
    set_uart0_br(br); // change baud-rate (h.w.)

    retry = tRS;
    while(1){
        fl_wait(tWT10);

        put_cmd_ua(FL_COM_RESET, 1, fl_cmd_prm); // send RESET command
        rc = get_sfrm_ua(fl_ua_sfrm, tWT0_TO); // get status frame
        if (rc){
            if (retry--){
                continue;
            }
            else {
                return rc;
            }
        }
        break; // got ACK !!
    }

    // switch(rc) {
    //     case FLC_NO_ERR: return rc; break; // case [A]
    //     case FLC_DFTO_ERR: return rc; break; // case [C]
    //     default: return rc; break; // case [B]
    // }

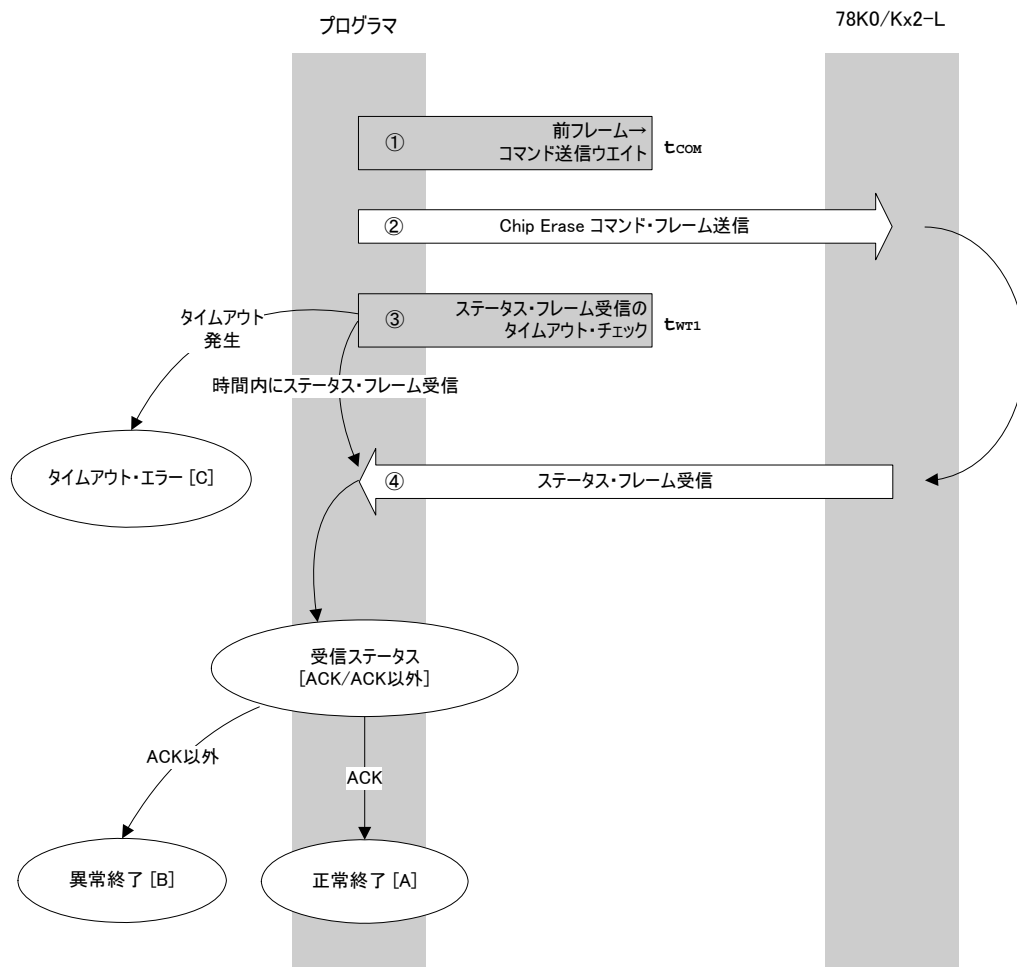
    return rc;
}

```

4.6 Chip Eraseコマンド

4.6.1 処理手順チャート

Chip Eraseコマンド処理手順



4.6.2 処理手順説明

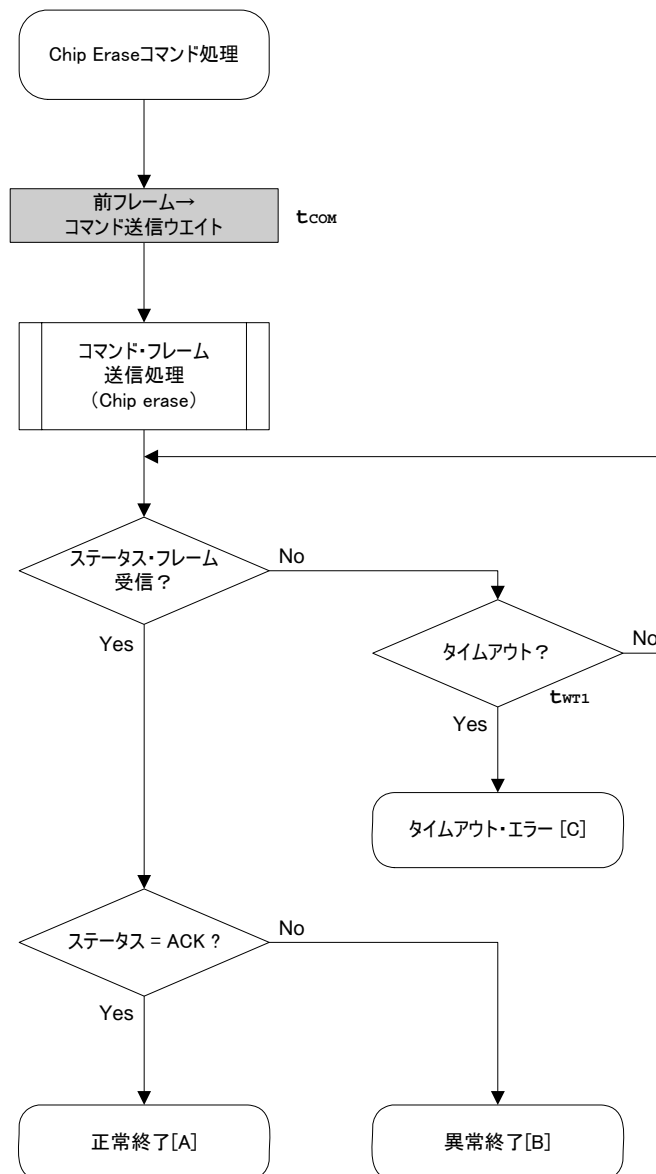
直前のフレームからコマンド送信までのウェイトをします（ウェイト時間 t_{COM} ）。
 コマンド・フレーム送信処理にて **Chip Eraseコマンド** を送信します。
 コマンド送信からステータス・フレーム受信までのタイムアウト・チェックを行います。タイムアウトが発生した場合は、**タイムアウト・エラー[C]** となります（タイムアウト時間 t_{WT1} ）。
 ステータス・コードをチェックします。

ST1 = ACKの場合 : **正常終了[A]** です。
 ST1 = ACK以外の場合 : **異常終了[B]** です。

4.6.3 終了時の内容

終了内容		ステータス・コード	内 容
正常終了 [A]	正常応答（ACK）	06H	コマンドが正常に実行され、チップ消去が正常に実行されたことを示します。
異常終了 [B]	チェックサム・エラー	07H	送信したコマンド・フレームのチェックサムが異常です。
	プロテクト・エラー	10H	セキュリティ設定で「チップ消去禁止」、「ブート・ブロック書き換え禁止」のいずれかの状態になっています。
	否定応答（NACK）	15H	コマンド・フレーム・データが異常です（データ長（LEN）不正，ETX なしなど）。
	MRG10 エラー	1AH	消去エラーが発生しました。
	MRG11 エラー	1BH	
	Write エラー	1CH	
タイムアウト・エラー [C]		-	規定の時間内にステータス・フレームの受信ができませんでした。

4.6.4 フロー・チャート



4.6.5 サンプル・プログラム

Chip Eraseコマンド処理のサンプル・プログラムです。

```

/*****
/*
/* Erase all(chip) command
/*
/*****
/* [r] u16          ... error code
/*****
u16          fl_ua_erase_all(void)
{
    u16      rc;

    fl_wait(tCOM);          // wait before sending command

    put_cmd_ua(FL_COM_ERASE_CHIP, 1, fl_cmd_prm); // send ERASE CHIP command

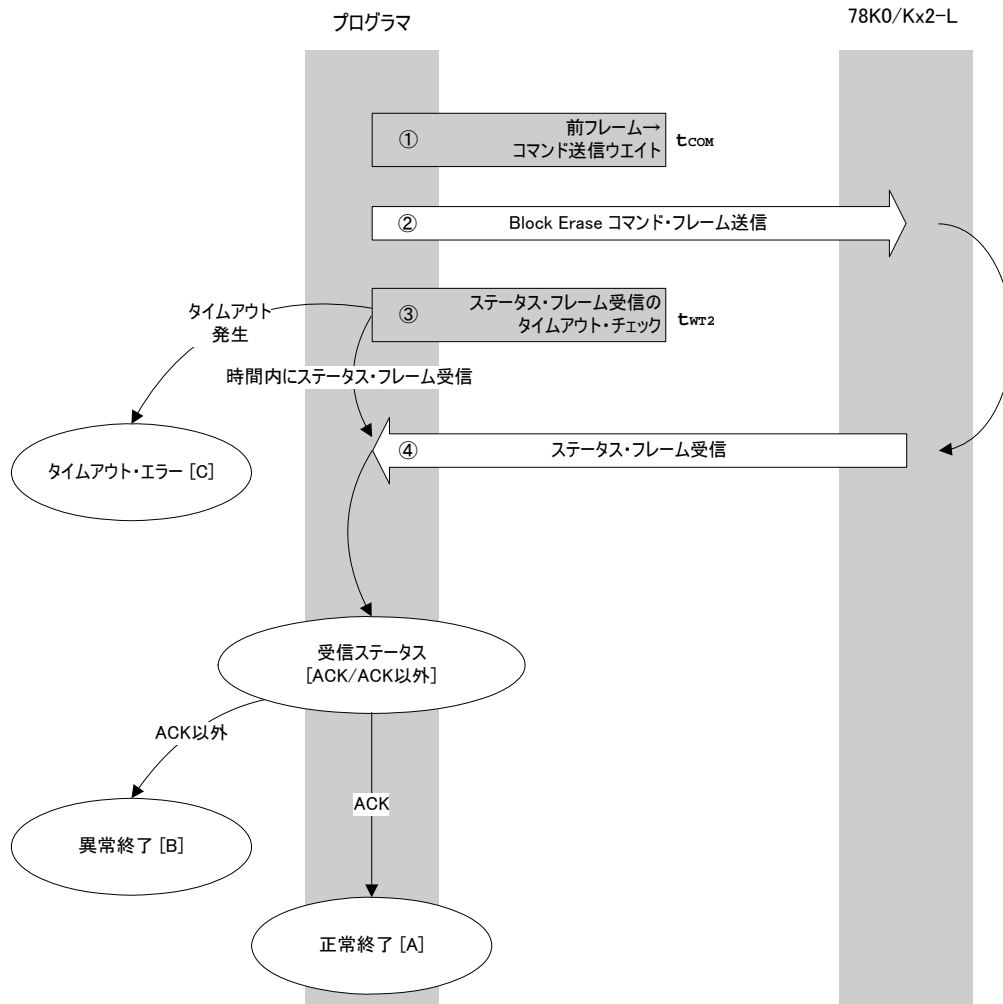
    rc = get_sfrm_ua(fl_ua_sfrm, tWT1_MAX); // get status frame
    // switch(rc) {
    //
    //     case    FLC_NO_ERR:      return rc;      break; // case [A]
    //     case    FLC_DFTO_ERR:   return rc;      break; // case [C]
    //     default:                return rc;      break; // case [B]
    // }
    return rc;
}

```

4.7 Block Eraseコマンド

4.7.1 処理手順チャート

Block Eraseコマンド処理手順



4.7.2 処理手順説明

直前のフレームからコマンド送信までのウェイトをします（ウェイト時間 t_{COM} ）。
 コマンド・フレーム送信処理にて **Block Eraseコマンド** を送信します。
 コマンド送信からステータス・フレーム受信までのタイムアウト・チェックを行います。タイムアウトが発生した場合は、**タイムアウト・エラー[C]** となります
 （タイムアウト時間 t_{WT2} ）。
 ステータス・コードをチェックします。

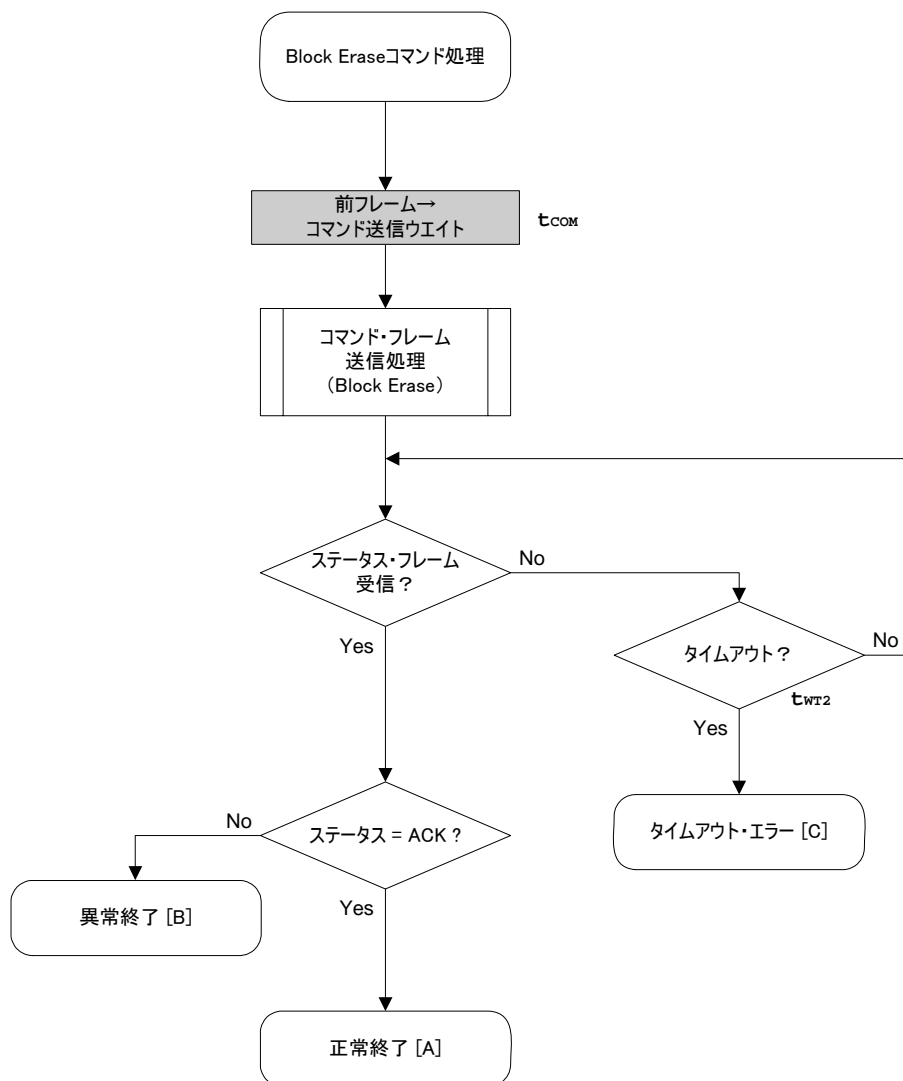
ST1 = ACKの場合 : **正常終了[A]** です。

ST1 = ACK以外の場合 : **異常終了[B]** です。

4.7.3 終了時の内容

終了内容		ステータス・コード	内 容
正常終了 [A]	正常応答 (ACK)	06H	コマンドが正常に実行され、ブロック消去が正常に実行されたことを示します。
異常終了 [B]	パラメータ・エラー	05H	指定した終了アドレスがフラッシュ・メモリの範囲外です。または指定した開始・終了アドレスが、ブロックの先頭・終了アドレスではありません。
	チェックサム・エラー	07H	送信したコマンド・フレームのチェックサムが異常です。
	プロテクト・エラー	10H	セキュリティ設定で「書き込み禁止」, 「ブロック消去禁止」, 「チップ消去禁止」のいずれかの状態になっています。指定範囲にブート・ブロックが含まれており「ブート・ブロック書き換え禁止」が設定されています。
	否定応答 (NACK)	15H	コマンド・フレーム・データが異常です（データ長 (LEN) 不正, ETX なしなど）。
	MRG10 エラー	1AH	消去エラーが発生しました。
タイムアウト・エラー [C]		-	規定の時間内にステータス・フレームの受信ができませんでした。

4.7.4 フロー・チャート



4.7.5 サンプル・プログラム

Block Eraseコマンド処理のサンプル・プログラムです。

```

/*****
/*
/* Erase block command
/*
/*****
/* [i] u8 block    ... block number
/* [r] u16         ... error code
/*****
u16      fl_ua_erase_blk(u16 sblk, u16 eblk)
{

    u16      rc;
    u32      wt2_max;
    u32      top, bottom;

    top = get_top_addr(sblk);          // get start address of start block
    bottom = get_bottom_addr(eblk);    // get end address of end block

    set_range_prm(fl_cmd_prm, top, bottom); // set SAH/SAM/SAL, EAH/EAM/EAL

    wt2_max = make_wt2_max(sblk, eblk);

    fl_wait(tCOM);                    // wait before sending command

    put_cmd_ua(FL_COM_ERASE_BLOCK, 1+6, fl_cmd_prm); // send ERASE CHIP command

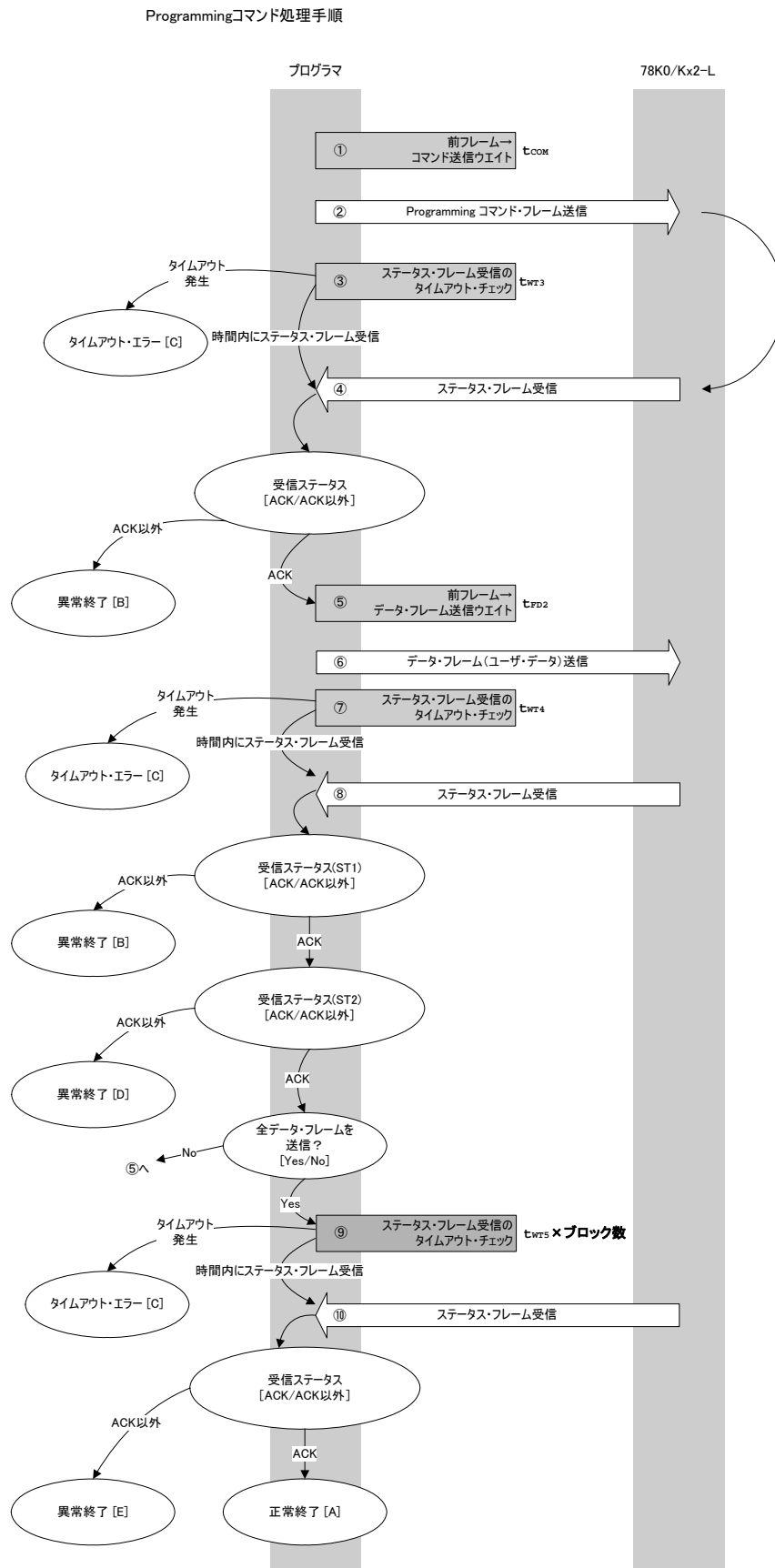
    rc = get_sfrm_ua(fl_ua_sfrm, wt2_max); // get status frame
    // switch(rc) {
    //
    //      case   FLC_NO_ERR:      return rc;      break; // case [A]
    //      case   FLC_DFTO_ERR:   return rc;      break; // case [C]
    //      default:                return rc;      break; // case [B]
    // }

    return rc;
}

```

4.8 Programmingコマンド

4.8.1 処理手順チャート



4.8.2 処理手順説明

直前のフレームからコマンド送信までのウェイトをします（ウェイト時間 t_{COM} ）。
 コマンド・フレーム送信処理により、**Programmingコマンド**を送信します。
 コマンド送信からステータス・フレーム受信までのタイムアウト・チェックを行います。
 タイムアウトが発生した場合は**タイムアウト・エラー[C]**となります
 （タイムアウト時間 t_{WT3} ）。
 ステータス・コードをチェックします。

ST1 = ACKの場合 : に進みます。
ST1 = ACK以外の場合 : **異常終了[B]**です。

直前のフレームからデータ・フレーム送信までのウェイトをします（ウェイト時間 t_{FD2} ）。
 データ・フレーム送信処理により、ユーザ・データを送信します。
 ユーザ・データ送信からデータ・フレーム受信までのタイムアウト・チェックを行います。
 タイムアウトが発生した場合は**タイムアウト・エラー[C]**となります
 （タイムアウト時間 t_{WT4} ）。
 ステータス・コード（ST1/ST2）をチェックします（処理手順チャートやフロー・チャートも参照してください）。

ST1 = ACK以外の場合 : **異常終了[B]**です。
ST1 = ACKの場合 : ST2の値に応じて以下の処理を行います。
 ・ ST2 = ACKの場合 : 全データ・フレームの送信が完了した場合は、 に進みます。
 まだ送信するデータ・フレームが残っている場合は、 より再実行します。
 ・ ST2 = ACK以外の場合 : **異常終了[D]**です。

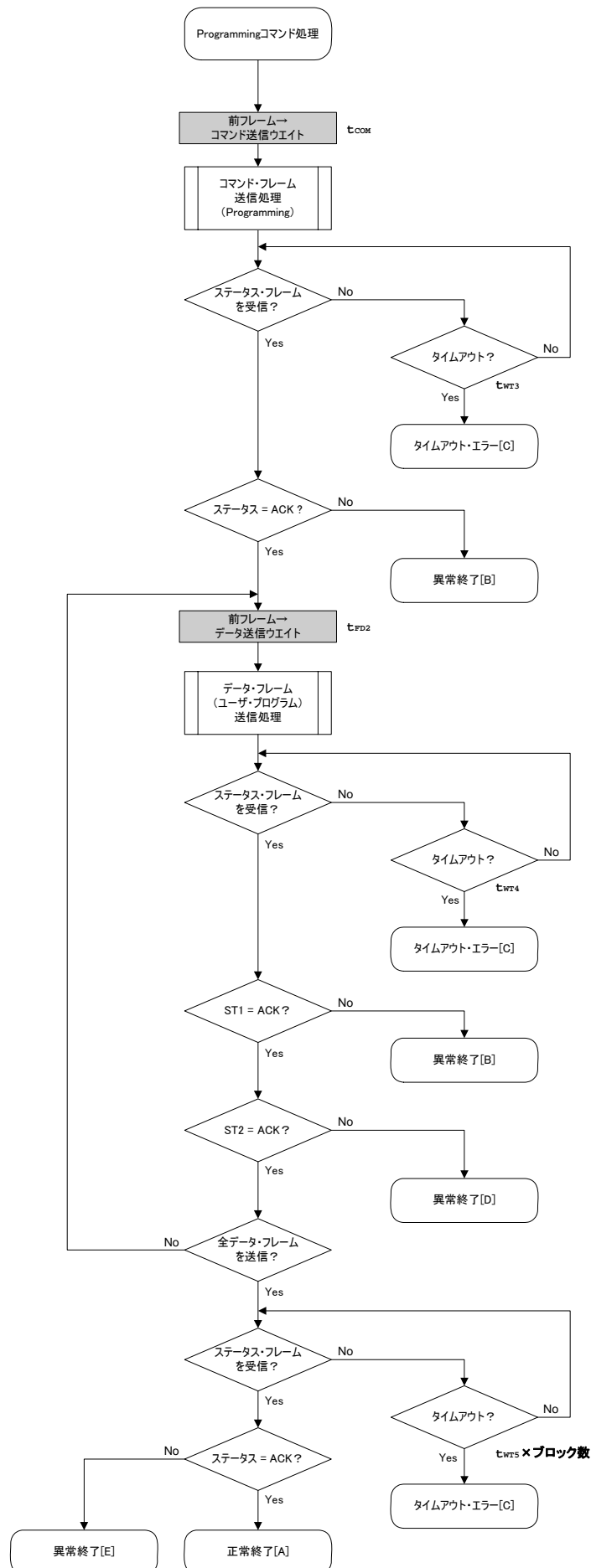
ステータス・フレーム受信までのタイムアウト・チェックを行います。
 タイムアウトが発生した場合は**タイムアウト・エラー[C]**となります
 （タイムアウト時間 $t_{WT5} \times \text{ブロック数}$ ）。
 ステータス・コードをチェックします。

ST1 = ACKの場合 : **正常終了[A]**です。
ST1 = ACK以外の場合 : **異常終了[E]**です。

4.8.3 終了時の内容

終了内容		ステータス・コード	内 容
正常終了 [A]	正常応答 (ACK)	06H	コマンドが正常に実行され、ユーザ・データの書き込みが正常に終了したことを示します。
異常終了 [B]	パラメータ・エラー	05H	開始 / 終了アドレスがフラッシュ・メモリの範囲外です。または、指定した開始 / 終了アドレスがブロックの先頭 / 終了アドレスではありません。
	チェックサム・エラー	07H	送信したコマンド・フレーム、またはデータ・フレームのチェックサムが異常です。
	プロテクト・エラー	10H	セキュリティ設定で、「書き込み禁止」になっています。また書き込み指定範囲にブート・ブロックが含まれ、「ブート・ブロック書き換え禁止」になっています。
	否定応答 (NACK)	15H	コマンド・フレーム・データまたはデータ・フレーム・データが異常です (データ長 (LEN) 不正, ETX なしなど)。
タイムアウト・エラー [C]		-	規定の時間内にステータス・フレームの受信ができませんでした。
異常終了 [D], [E]	MRG10 エラー	1AH	書き込みエラーが発生しました。
	MRG11 エラー	1BH	
	Write エラー	1CH	

4.8.4 フロー・チャート



4. 8. 5 サンプル・プログラム

Programmingコマンド処理のサンプル・プログラムです。

```

/*****
/*
/* Write command
/*
/*****
/* [i] u32 top      ... start address
/* [i] u32 bottom  ... end address
/* [r] u16          ... error code
/*****

#define          fl_st2_ua          (fl_ua_sfrm[OFS_STA_PLD+1])

u16          fl_ua_write(u32 top, u32 bottom)
{
    u16      rc;
    u32      send_head, send_size;
    bool     is_end;
    u16      block_num;

    block_num = get_block_num(top, bottom); // get block num

    /*****
    /*      set params
    /*****
    set_range_prm(fl_cmd_prm, top, bottom); // set SAH/SAM/SAL, EAH/EAM/EAL

    /*****
    /*      send command & check status
    /*****
    fl_wait(tCOM);          // wait before sending command

    put_cmd_ua(FL_COM_WRITE, 7, fl_cmd_prm); // send "Programming" command

    rc = get_sfrm_ua(fl_ua_sfrm, tWT3_TO); // get status frame
    switch(rc) {
        case    FLC_NO_ERR:                break; // continue
    //      case    FLC_DFTO_ERR:    return rc;    break; // case [C]
        default:                return rc;    break; // case [B]
    }

    /*****
    /*      send user data
    /*****
    send_head = top;

    while(1){

        // make send data frame
        if ((bottom - send_head) > 256){          // rest size > 256 ?
            is_end = false;                        // yes, not is_end frame
            send_size = 256;                        // transmit size = 256 byte
        }
        else{

```

```

        is_end = true;
        send_size = bottom - send_head + 1;    // transmit size = (bottom
- send_head)+1 byte

    }
    memcpy(fl_txdata_frm, rom_buf+send_head, send_size);    // set data frame
payload
    send_head += send_size;

    fl_wait(tFD2);    // wait before sending data frame

    put_dfrm_ua(send_size, fl_txdata_frm, is_end);    // send user data

    rc = get_sfrm_ua(fl_ua_sfrm, tWT4_MAX);    // get status frame
    switch(rc) {
        case FLC_NO_ERR:    break;    // continue
        case FLC_DFTO_ERR:    return rc;    break;    // case [C]
        default:    return rc;    break;    // case [B]
    }
    if (fl_st2_ua != FLST_ACK){    // ST2 = ACK ?
        rc = decode_status(fl_st2_ua);    // No
        return rc;    // case [D]
    }
    if (is_end)
        break;

}
/*****
/*      Check internally verify      */
/*****/
// get status frame again

rc = get_sfrm_ua(fl_ua_sfrm, tWT5_MAX*block_num);

switch(rc) {
//      case FLC_NO_ERR:    return rc;    break;    // case [A]
      case FLC_DFTO_ERR:    return rc;    break;    // case [C]
      default:    return rc;    break;    // case [E]
}

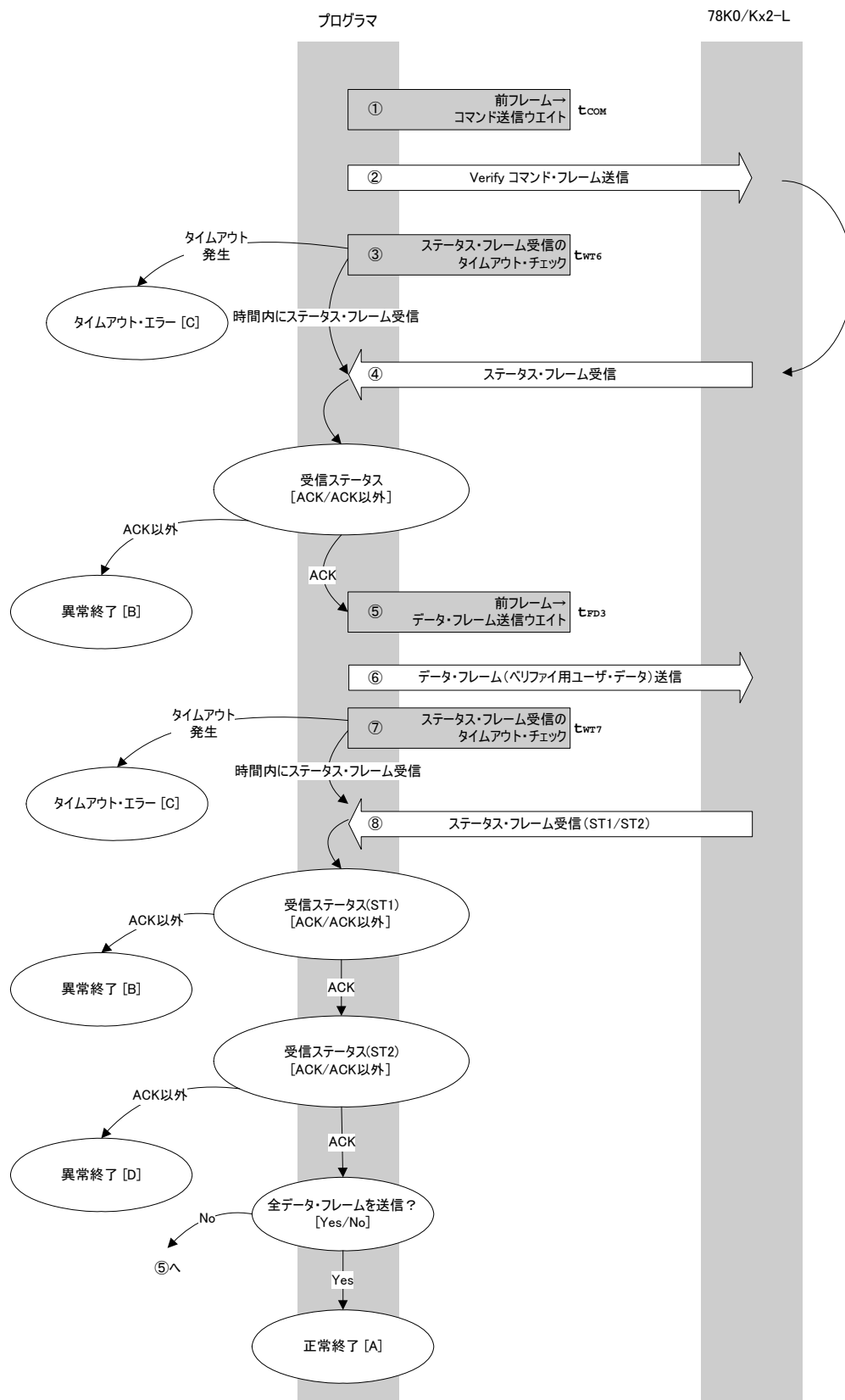
return rc;
}

```


4.9 Verifyコマンド

4.9.1 処理手順チャート

Verifyコマンド処理手順



4.9.2 処理手順説明

直前のフレームからコマンド送信までのウェイトをします（ウェイト時間 t_{COM} ）。
 コマンド・フレーム送信処理により、**Verifyコマンド**を送信します。
 コマンド送信からステータス・フレーム受信までのタイムアウト・チェックを行います。
 タイムアウトが発生した場合は**タイムアウト・エラー[C]**となります
 （タイムアウト時間 t_{WT6} ）。
 ステータス・コードをチェックします。

ST1 = ACKの場合 : に進みます。
 ST1 = ACK以外の場合 : **異常終了[B]**です。

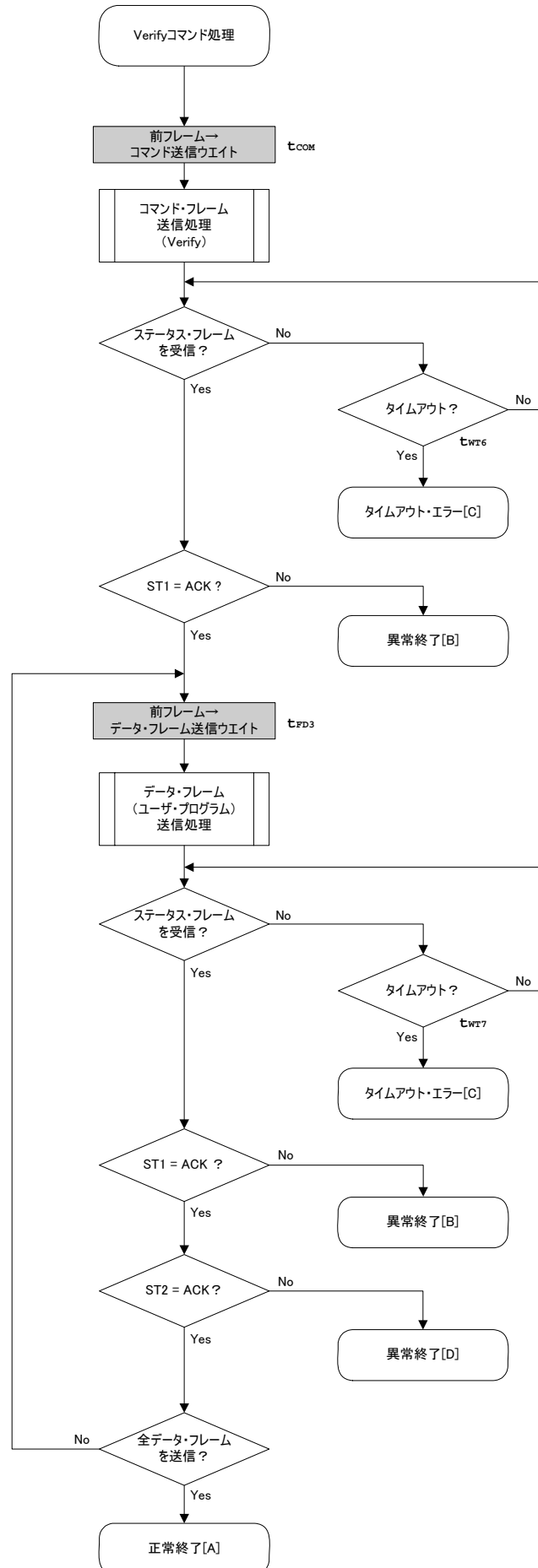
直前のフレームからデータ・フレーム送信までのウェイトをします（ウェイト時間 t_{FD3} ）。
 データ・フレーム送信処理により、ペリファイ用ユーザ・データを送信します。
 ユーザ・データ送信からステータス・フレーム受信までのタイムアウト・チェックを行います。
 タイムアウトが発生した場合は**タイムアウト・エラー[C]**となります
 （タイムアウト時間 t_{WT7} ）。
 ステータス・コード（ST1/ST2）をチェックします（処理手順チャートやフロー・チャートも参照してください）。

ST1 = ACK以外の場合 : **異常終了[B]**です。
 ST1 = ACKの場合 : 受信ステータス（ST2）の値に応じて次の処理を行います。
 ・ ST2 = ACKの場合 : 全データ・フレームを送信済みの場合は**正常終了[A]**です。
 まだ送信すべきデータ・フレームがある場合は から再実行します。
 ・ ST2 = ACK以外の場合 : **異常終了[D]**です。

4.9.3 終了時の内容

終了内容		ステータス・コード	内 容
正常終了 [A]	正常応答（ACK）	06H	コマンドが正常に実行され、ユーザ・データのペリファイが正常に実行されたことを示します。
異常終了 [B]	パラメータ・エラー	05H	開始／終了アドレスがフラッシュ・メモリの範囲外です。または、開始／終了アドレスがブロックの開始／終了アドレスではありません。
	チェックサム・エラー	07H	送信したコマンド・フレーム、またはデータ・フレームのチェックサムが異常です。
	否定応答（NACK）	15H	コマンド・フレーム・データが異常です（データ長（LEN）不正、ETX なしなど）。
タイムアウト・エラー [C]		-	規定の時間内にステータス・フレームの受信ができませんでした。
異常終了 [D]	ペリファイ・エラー	0FH（ST2）	ペリファイ・エラーが発生しました。

4.9.4 フロー・チャート



4.9.5 サンプル・プログラム

Verifyコマンド処理のサンプル・プログラムです。

```

/*****
/*
/* Verify command
/*
/*****
/* [i] u32 top      ... start address
/* [i] u32 bottom   ... end address
/* [r] u16          ... error code
/*****
u16      fl_ua_verify(u32 top, u32 bottom, u8 *buf)
{
    u16      rc;
    u32      send_head, send_size;
    bool     is_end;

    /*****
    /*      set params
    /*****
    set_range_prm(fl_cmd_prm, top, bottom); // set SAH/SAM/SAL, EAH/EAM/EAL

    /*****
    /*      send command & check status
    /*****

    fl_wait(tCOM); // wait before sending command

    put_cmd_ua(FL_COM_VERIFY, 7, fl_cmd_prm); // send VERIFY command

    rc = get_sfrm_ua(fl_ua_sfrm, tWT6_MAX); // get status frame
    switch(rc) {
        case FLC_NO_ERR: break; // continue
        // case FLC_DFTO_ERR: return rc; break; // case [C]
        default: return rc; break; // case [B]
    }

    /*****
    /*      send user data
    /*****
    send_head = top;

    while(1){
        // make send data frame
        if ((bottom - send_head) > 256){ // rest size > 256 ?
            is_end = false; // yes, not is_end frame
            send_size = 256; // transmit size = 256 byte
        }
        else{
            is_end = true;
            send_size = bottom - send_head + 1; // transmit size = (bottom -
send_head)+1 byte
        }
        memcpy(fl_txdata_frm, buf+send_head, send_size); // set data frame payload

```

```
send_head += send_size;

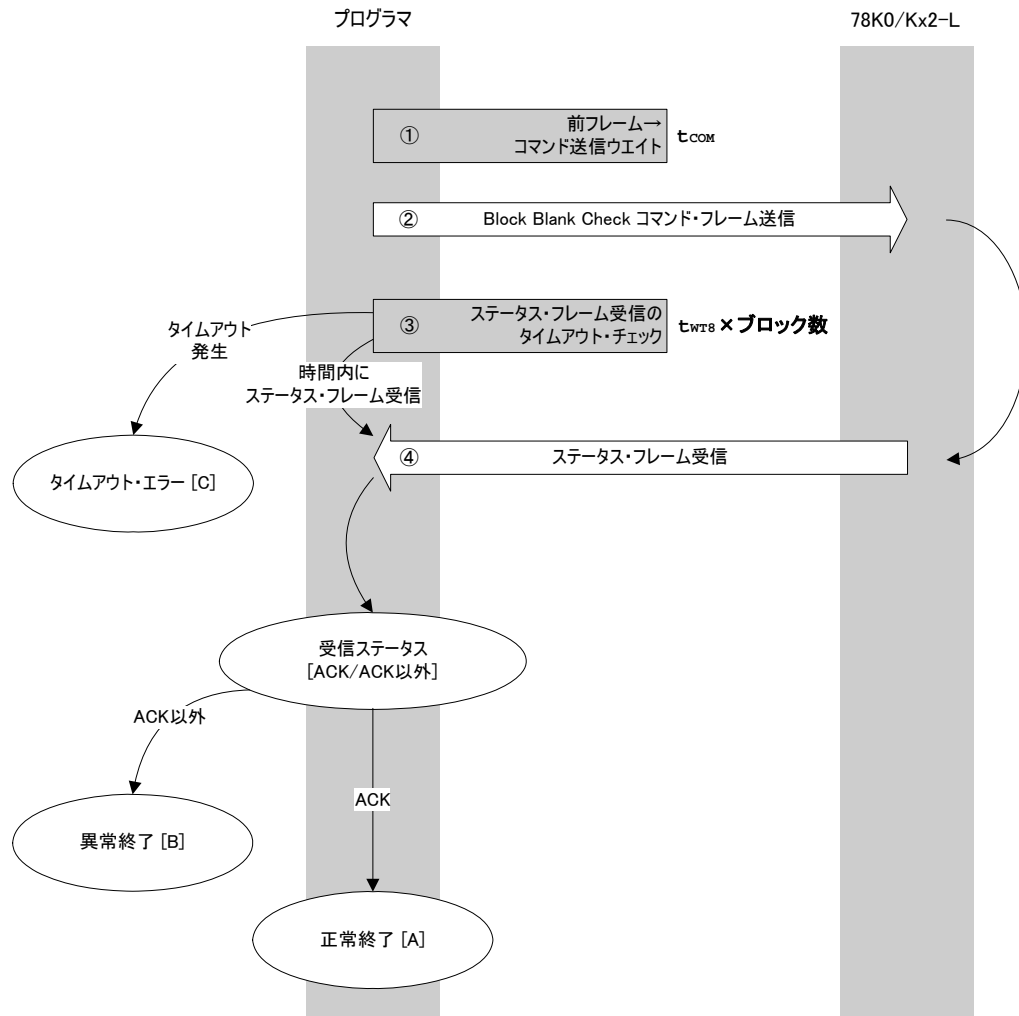
fl_wait(tFD3);
put_dfrm_ua(send_size, fl_txdata_frm, is_end); // send user data

rc = get_sfrm_ua(fl_ua_sfrm, tWT7_MAX);          // get status frame
switch(rc) {
    case FLC_NO_ERR:                             break; // continue
    // case FLC_DFTO_ERR: return rc; break; // case [C]
    default: return rc; break; // case [B]
}
if (fl_st2_ua != FLST_ACK){                      // ST2 = ACK ?
    rc = decode_status(fl_st2_ua); // No
    return rc;                                // case [D]
}
if (is_end)                                     // send all user data ?
    break;                                     // yes
//continue;
}
return FLC_NO_ERR; // case [A]
}
```

4. 10 Block Blank Checkコマンド

4. 10. 1 処理手順チャート

Block Blank Checkコマンド処理手順



4. 10. 2 処理手順説明

直前のフレームからコマンド送信までのウェイトをします（ウェイト時間 t_{COM} ）。
 コマンド・フレーム送信処理にて **Block Blank Checkコマンド** を送信します。
 コマンド送信からステータス・フレーム受信までのタイムアウト・チェックを行います。タイムアウトが発生した場合は **タイムアウト・エラー[C]** となります
 （タイムアウト時間 $t_{WT8} \times$ ブロック数）。
 ステータス・コードをチェックします。

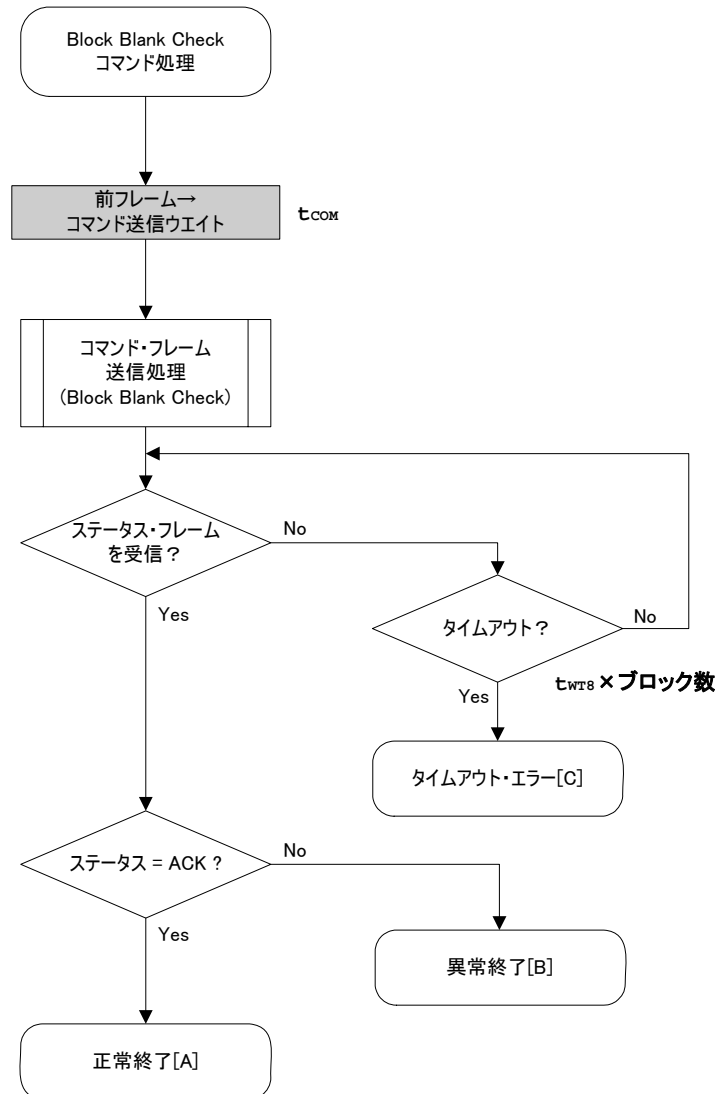
ST1 = ACKの場合 : **正常終了[A]** です。

ST1 = ACK以外の場合 : **異常終了[B]** です。

4. 10. 3 終了時の内容

終了内容		ステータス・コード	内 容
正常終了 [A]	正常応答 (ACK)	06H	コマンドが正常に実行され、ブロック・ブランク・チェックが正常に実行されたことを示します。
異常終了 [B]	パラメータ・エラー	05H	終了アドレスがフラッシュ・メモリの範囲外です。 または、開始 / 終了アドレスがブロックの先頭・終了アドレスではありません。またはパラメータ D01が、00H, 01H 以外の値です。
	チェックサム・エラー	07H	送信したコマンド・フレームのチェックサムが異常です。
	否定応答 (NACK)	15H	コマンド・フレーム・データが異常です（データ長 (LEN) 不正, ETX なしなど）。
	MRG11 エラー	1BH	指定したブロックのフラッシュ・メモリがブランクではありません。
タイムアウト・エラー [C]		-	規定の時間内にステータス・フレームの受信ができませんでした。

4.10.4 フロー・チャート



4.10.5 サンプル・プログラム

Block Blank Checkコマンド処理のサンプル・プログラムです。

```

/*****
/*
/* Block blank check command
/*
/*
/*****
/* [i] u32 top      ... top address of blank check
/* [i] u32 bottom   ... bottom address of blank check
/* [i] u8 whole     ... <1>check w/NON user flash
/*
/*                  <0>chek only user flash
/* [r] u16          ... error code
/*****
u16      fl_ua_blk_blank_chk(u32 top, u32 bottom, u8 whole)
{
    u16      rc;
    u16      block_num;

    set_range_prm(fl_cmd_prm, top, bottom); // set SAH/SAM/SAL, EAH/EAM/EAL
    block_num = get_block_num(top, bottom); // get block num
    fl_cmd_prm[6] = whole;                  // check only user area or not

    fl_wait(tCOM);                          // wait before sending command

    put_cmd_ua(FL_COM_BLOCK_BLANK_CHK, 7+1, fl_cmd_prm);

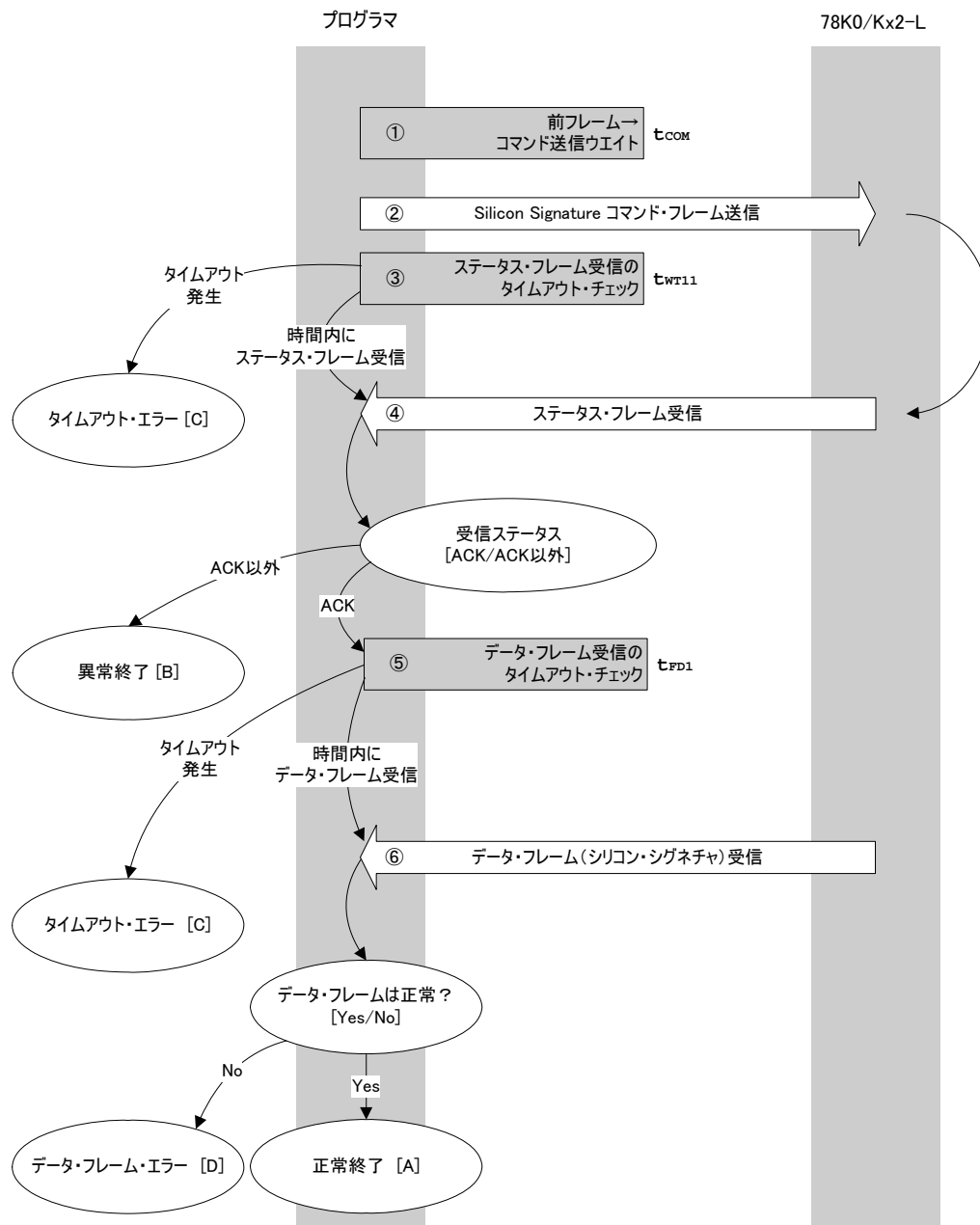
    rc = get_sfrm_ua(fl_ua_sfrm, tWT8_MAX * block_num); // get status frame
    // switch(rc) {
    //
    //      case   FLC_NO_ERR:      return rc;      break; // case [A]
    //      case   FLC_DFTO_ERR:    return rc;      break; // case [C]
    //      default:                return rc;      break; // case [B]
    // }
    return rc;
}

```

4. 11 Silicon Signatureコマンド

4. 11. 1 処理手順チャート

Silicon Signatureコマンド処理手順



4. 11. 2 処理手順説明

直前のフレームからコマンド送信までのウェイトをします（ウェイト時間 t_{COM} ）。
 コマンド・フレーム送信処理により，Silicon Signatureコマンドを送信します。
 コマンド送信からステータス・フレーム受信までのタイムアウト・チェックを行います。タイムアウトが発生した場合はタイムアウト・エラー[C]です（タイムアウト時間 t_{WT11} ）。
 ステータス・コードをチェックします。

ST1 = ACKの場合 : に進みます。
 ST1 = ACK以外の場合 : 異常終了[B]です。

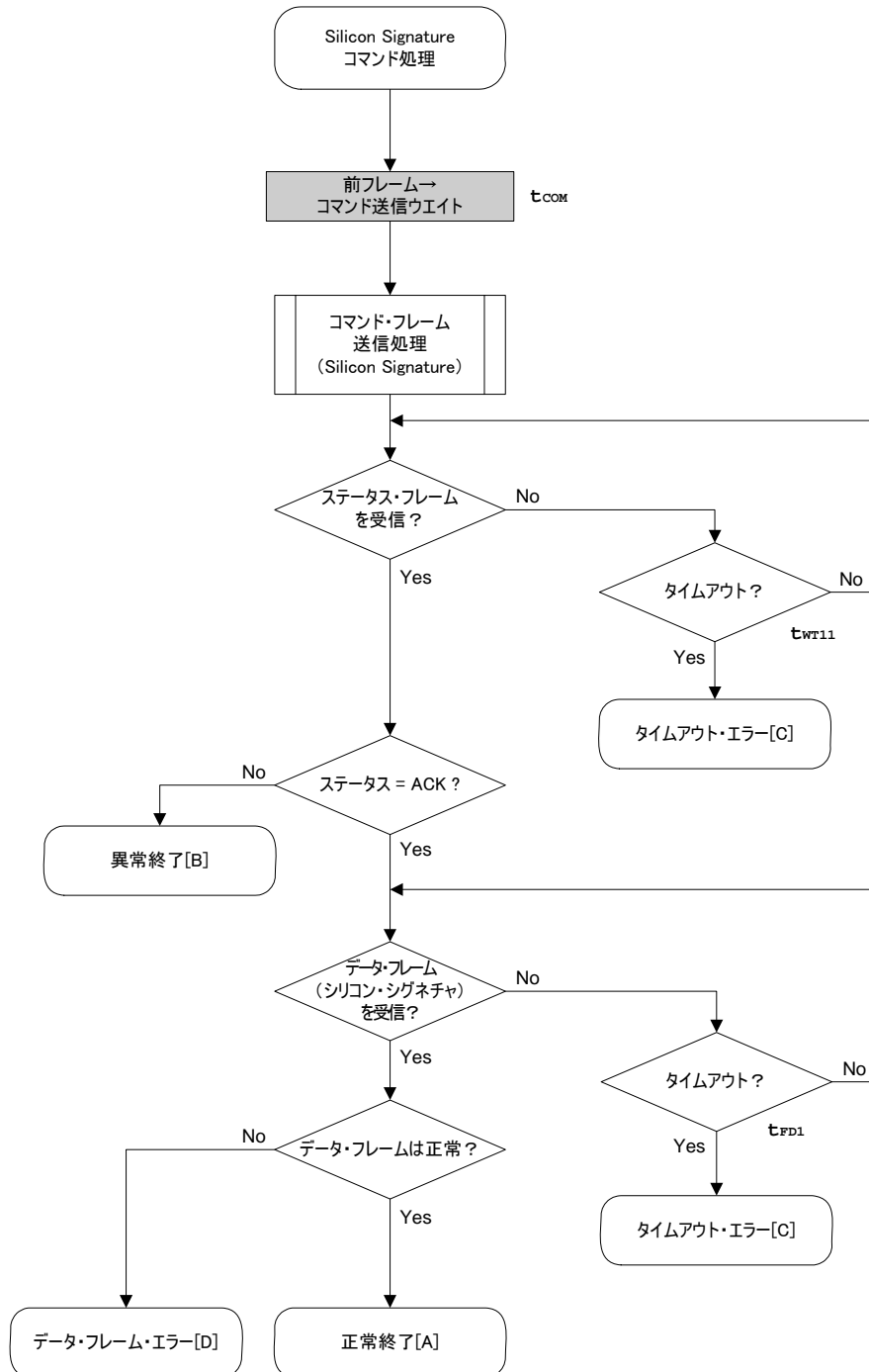
データ・フレーム（シリコン・シグネチャ・データ）受信までのタイムアウト・チェックを行います。
 タイムアウトが発生した場合はタイムアウト・エラー[C]です（タイムアウト時間 t_{FD1} ）。
 受信したデータ・フレーム（シリコン・シグネチャ・データ）をチェックします。

データ・フレームが正常の場合 : 正常終了[A]です。
 データ・フレームが異常の場合 : データ・フレーム・エラー[D]です。

4. 11. 3 終了時の内容

終了内容		ステータス・コード	内 容
正常終了 [A]	正常応答（ACK）	06H	コマンドが正常に実行され，シリコン・シグネチャ・データの取得が正常に実行されたことを示します。
異常終了 [B]	チェックサム・エラー	07H	送信したコマンド・フレームのチェックサムが異常です。
	否定応答（NACK）	15H	コマンド・フレーム・データが異常です（データ長（LEN）不正，ETXなしなど）。
	リード・エラー	20H	セキュリティ情報の読み出しに失敗しました。
タイムアウト・エラー [C]		-	ステータス・フレーム，またはデータ・フレームの受信でタイムアウトが発生しました。
データ・フレーム・エラー[D]		-	シリコン・シグネチャ・データとして受信したデータ・フレームのチェックサムが異常です。

4.11.4 フロー・チャート



4. 11. 5 サンプル・プログラム

Silicon Signatureコマンド処理のサンプル・プログラムです。

```

/*****
/*
/* Get silicon signature command
/*
/*****
/* [i] u8 *sig      ... pointer to signature save area
/* [r] u16          ... error code
/*****
u16      fl_ua_getsig(u8 *sig)
{
    u16      rc;

    fl_wait(tCOM);                // wait before sending command

    put_cmd_ua(FL_COM_GET_SIGNATURE, 1, fl_cmd_prm); // send GET SIGNATURE command

    rc = get_sfrm_ua(fl_ua_sfrm, tWT11_MAX);        // get status frame
    switch(rc) {
        case FLC_NO_ERR:                break; // continue
        // case FLC_DFTO_ERR:    return rc;    break; // case [C]
        default:                    return rc;    break; // case [B]
    }

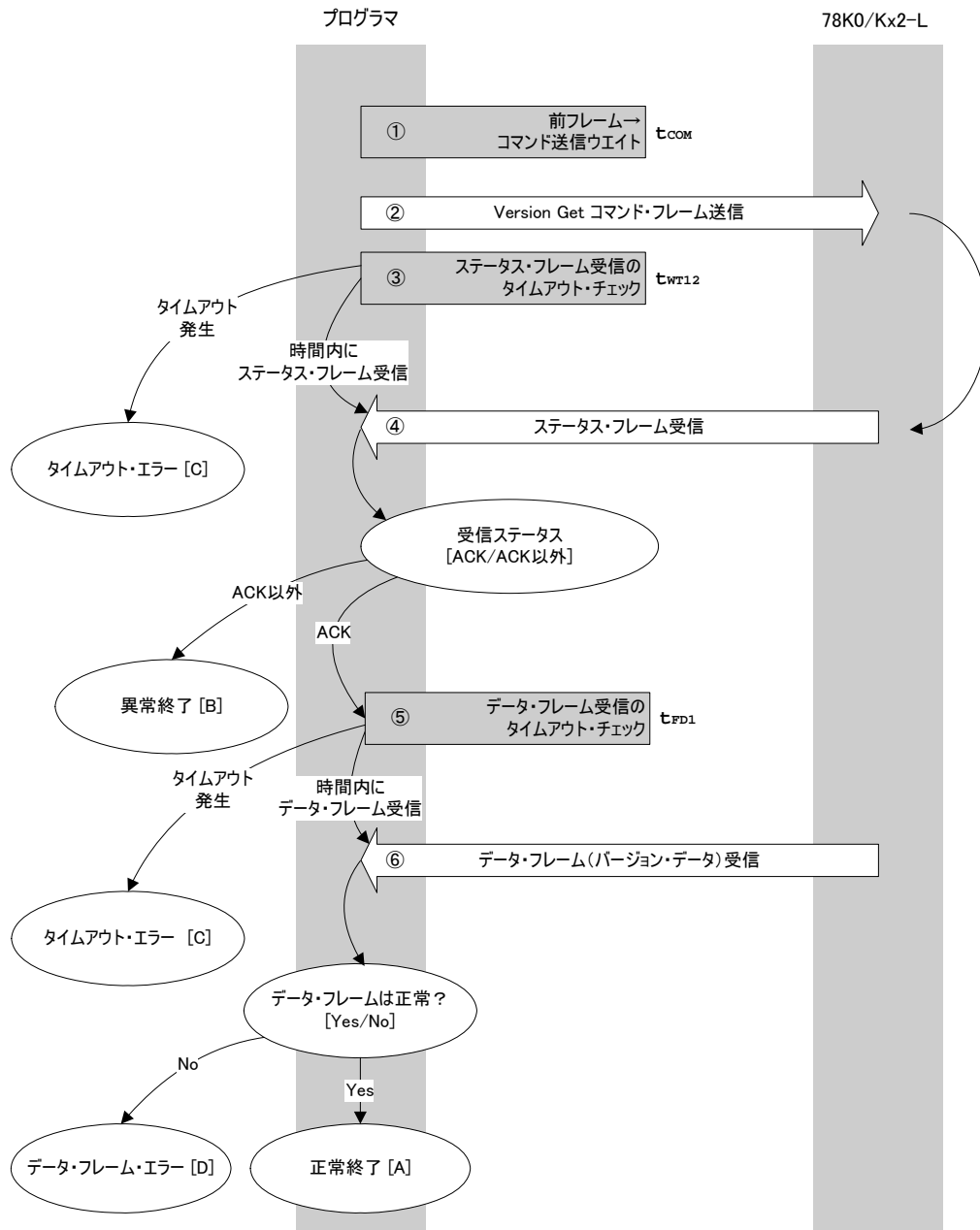
    rc = get_dfrm_ua(fl_rxdata_frm, tFD2_MAX);        // get status frame
    if (rc){
        return rc;                // if error
        // case [D]
    }
    memcpy(sig, fl_rxdata_frm+OFS_STA_PLD, fl_rxdata_frm[OFS_LEN]); // copy Signature
data
    return rc;                // case [A]
}

```

4. 12 Version Getコマンド

4. 12. 1 処理手順チャート

Version Getコマンド処理手順



4. 12. 2 処理手順説明

直前のフレームからコマンド送信までのウェイトをします（ウェイト時間 t_{COM} ）。
 コマンド・フレーム送信処理により、**Version Getコマンド**を送信します。
 コマンド送信からステータス・フレーム受信までのタイムアウト・チェックを行います。
 タイムアウトが発生した場合は**タイムアウト・エラー[C]**です
 （タイムアウト時間 t_{WT12} ）。
 ステータス・コードをチェックします。

ST1 = ACKの場合 : に進みます。
ST1 = ACK以外の場合 : **異常終了[B]**です。

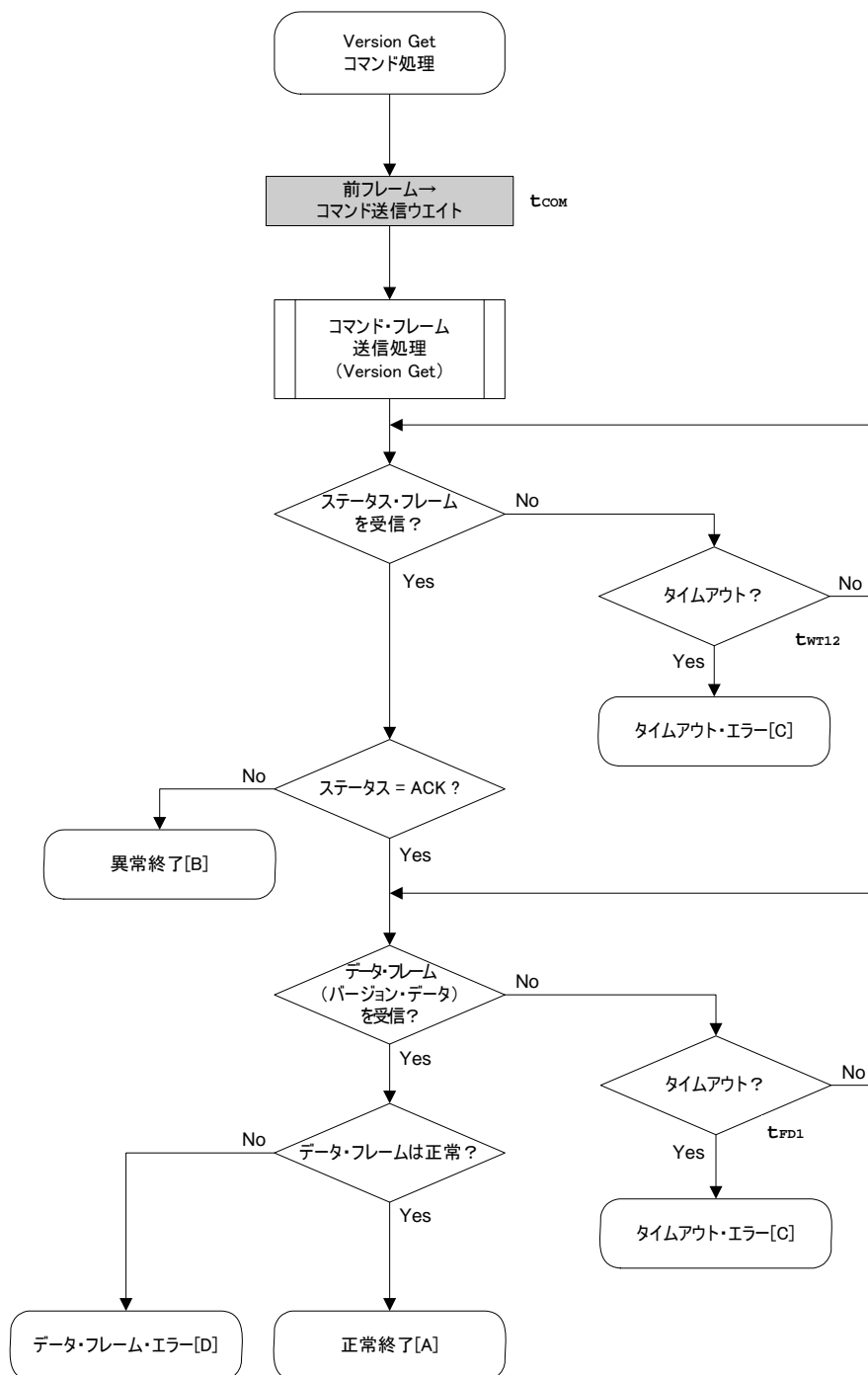
データ・フレーム（バージョン・データ）受信までのタイムアウト・チェックを行います。タイムアウトが発生した場合は**タイムアウト・エラー[C]**です
 （タイムアウト時間 t_{FD1} ）。
 受信したデータ・フレーム（バージョン・データ）をチェックします。

データ・フレームが正常の場合 : **正常終了[A]**です。
データ・フレームが異常の場合 : **データ・フレーム・エラー[D]**です。

4. 12. 3 終了時の内容

終了内容		ステータス・コード	内 容
正常終了 [A]	正常応答（ACK）	06H	コマンドが正常に実行され、バージョン・データを取得できたことを示します。
異常終了 [B]	チェックサム・エラー	07H	送信したコマンド・フレームのチェックサムが異常です。
	否定応答（NACK）	15H	コマンド・フレーム・データが異常です（データ長（LEN）不正，ETX なしなど）。
タイムアウト・エラー [C]		-	ステータス・フレーム，またはデータ・フレームの受信でタイムアウトが発生しました。
データ・フレーム・エラー [D]		-	バージョン・データとして受信したデータ・フレームのチェックサムが異常です。

4. 12. 4 フロー・チャート



4. 12. 5 サンプル・プログラム

Version Getコマンド処理のサンプル・プログラムです。

```

/*****
/*
/* Get device/firmware version command
/*
/*
/*****
/* [i] u8 *buf      ... pointer to version data save area
/* [r] u16          ... error code
/*****
u16      fl_ua_getver(u8 *buf)
{
    u16      rc;

    fl_wait(tCOM);                                // wait before sending command

    put_cmd_ua(FL_COM_GET_VERSION, 1, fl_cmd_prm); // send GET VERSION command

    rc = get_sfrm_ua(fl_ua_sfrm, tWT12_TO);        // get status frame
    switch(rc) {
        case FLC_NO_ERR:                          break; // continue
        // case FLC_DFTO_ERR:      return rc;      break; // case [C]
        default:                    return rc;      break; // case [B]
    }

    rc = get_dfrm_ua(fl_rxdata_frm, tFD1_TO);      // get data frame
    if (rc){
        return rc;                                // case [D]
    }

    memcpy(buf, fl_rxdata_frm+OFS_STA_PLD, DFV_LEN); // copy version data
    return rc;                                     // case [A]
}
u16      fl_ua_getver(u8 *buf)
{
    u16      rc;

    fl_wait(tCOM);                                // wait before sending command

    put_cmd_ua(FL_COM_GET_VERSION, 1, fl_cmd_prm); // send GET VERSION command

    rc = get_sfrm_ua(fl_ua_sfrm, tWT12_MAX);        // get status frame
    switch(rc) {
        case FLC_NO_ERR:                          break; // continue
        // case FLC_DFTO_ERR:      return rc;      break; // case [C]
        default:                    return rc;      break; // case [B]
    }

    rc = get_dfrm_ua(fl_rxdata_frm, tFD2_MAX);      // get data frame
    if (rc){
        return rc;                                // case [D]
    }

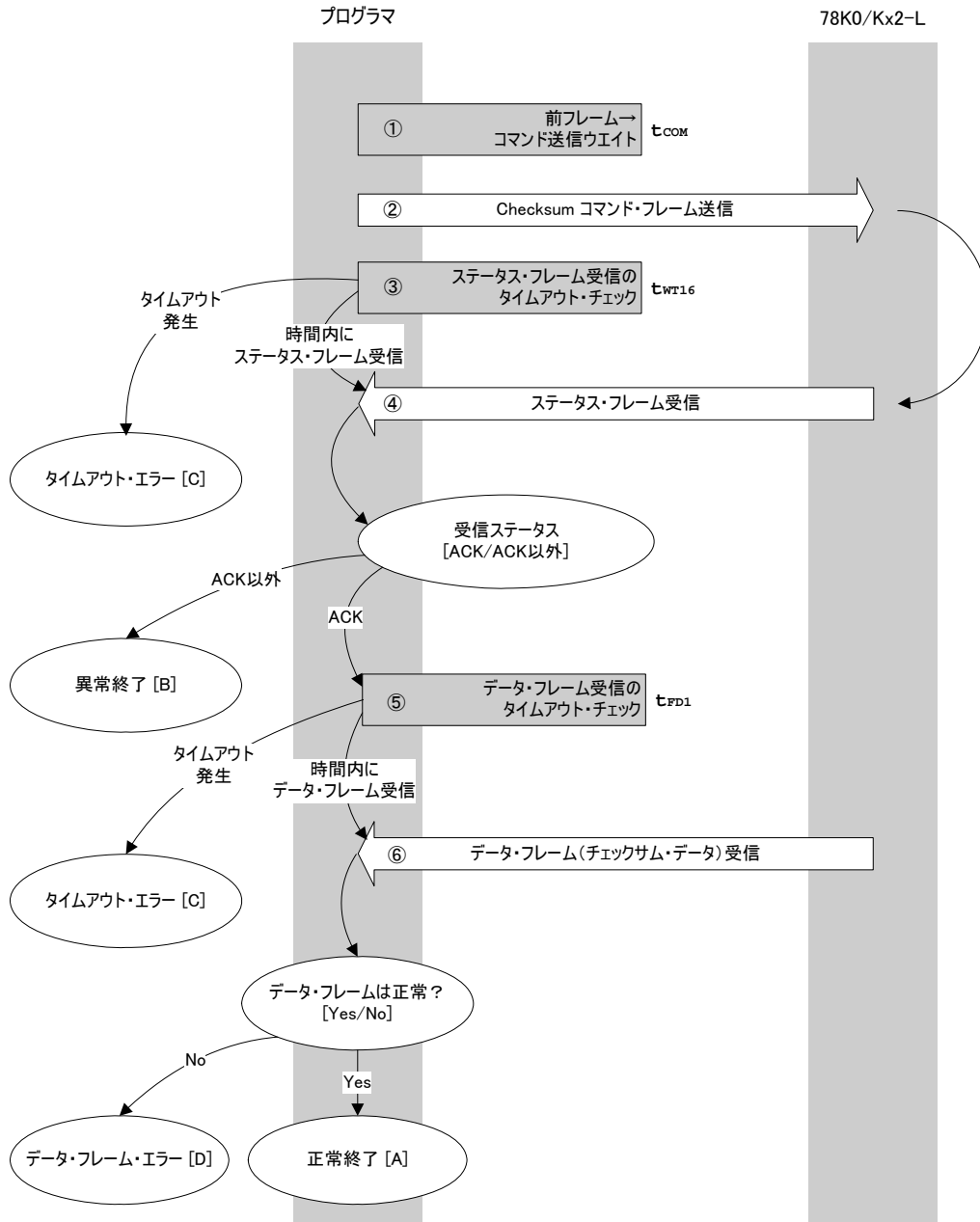
    memcpy(buf, fl_rxdata_frm+OFS_STA_PLD, DFV_LEN); // copy version data
    return rc;                                     // case [A]
}

```

4. 13 Checksumコマンド

4. 13. 1 処理手順チャート

Checksumコマンド処理手順



4. 13. 2 処理手順説明

直前のフレームからコマンド送信までのウェイトをします（ウェイト時間 t_{COM} ）。
 コマンド・フレーム送信処理により、Checksumコマンドを送信します。
 コマンドの送信からステータス・フレーム受信までのタイムアウト・チェックを行います。タイムアウトが発生した場合はタイムアウト・エラー[C]です（タイムアウト時間 t_{WT16} ）。
 ステータス・コードをチェックします。

ST1 = ACKの場合 : に進みます。
 ST1 = ACK以外の場合 : 異常終了[B]です。

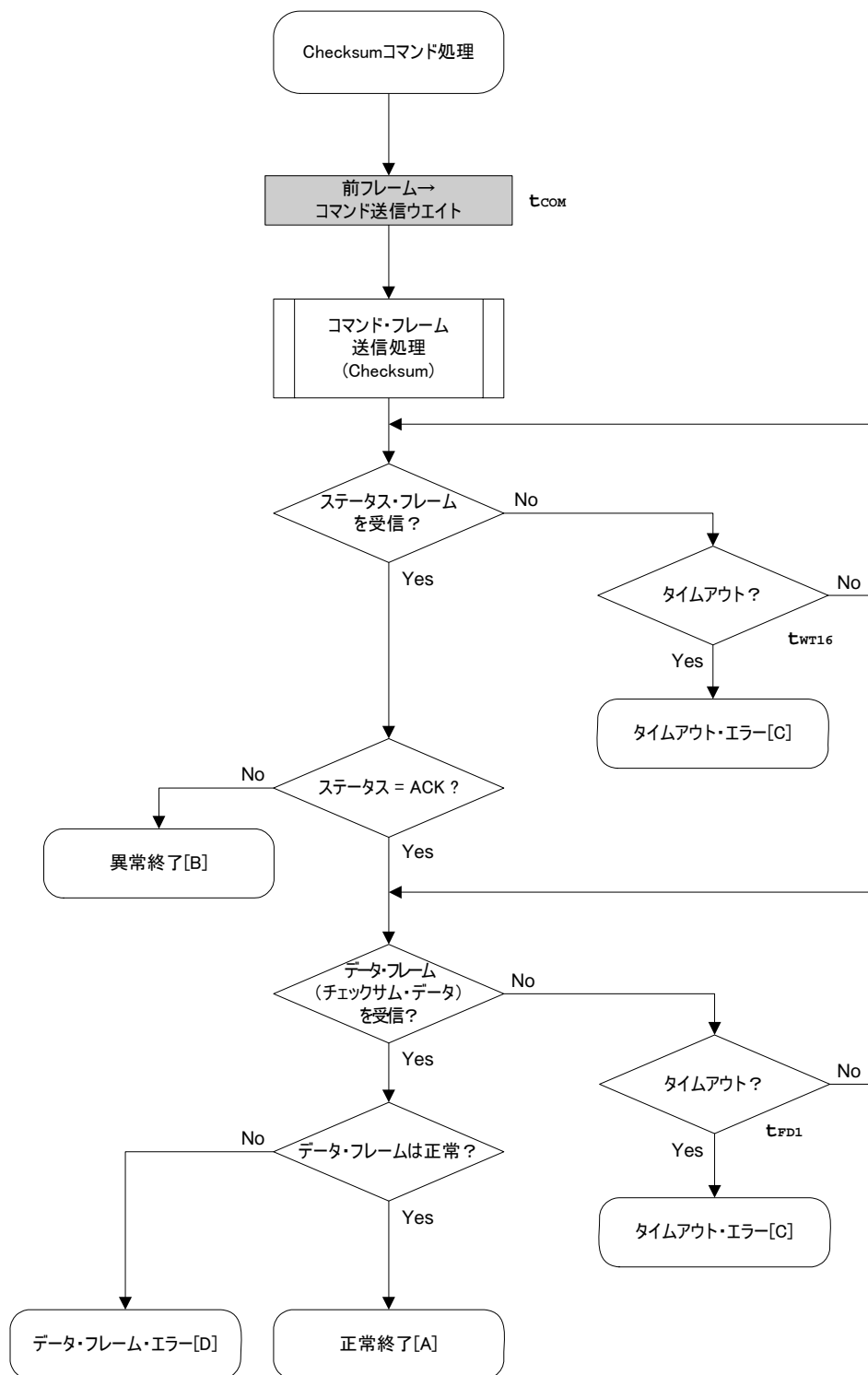
データ・フレーム（チェックサム・データ）受信までのタイムアウト・チェックを行います。タイムアウトが発生した場合はタイムアウト・エラー[C]です（タイムアウト時間 t_{FD1} ）。
 受信したデータ・フレーム（チェックサム・データ）をチェックします。

データ・フレームが正常の場合 : 正常終了[A]です。
 データ・フレームが異常の場合 : データ・フレーム・エラー[D]です。

4. 13. 3 終了時の内容

終了内容		ステータス・コード	内 容
正常終了 [A]	正常応答（ACK）	06H	コマンドが正常に実行され、チェックサム・データを取得できたことを示します。
異常終了 [B]	パラメータ・エラー	05H	開始 / 終了アドレスがフラッシュ・メモリの範囲外です。または、開始 / 終了アドレスがブロックの開始 / 終了アドレスではありません。
	チェックサム・エラー	07H	送信したコマンド・フレームのチェックサムが異常です。
	否定応答（NACK）	15H	コマンド・フレーム・データが異常です（データ長（LEN）不正、ETX なしなど）。
タイムアウト・エラー [C]		-	ステータス・フレーム、またはデータ・フレームの受信でタイムアウトが発生しました。
データ・フレーム・エラー [D]		-	チェックサム・データとして受信したデータ・フレームのチェックサムが異常です。

4. 13. 4 フロー・チャート



4.13.5 サンプル・プログラム

Checksumコマンド処理のサンプル・プログラムです。

```

/*****
/*
/* Get checksum command
/*
/*****
/* [i] u16 *sum    ... pointer to checksum save area
/* [i] u32 top     ... start address
/* [i] u32 bottom  ... end address
/* [r] u16         ... error code
/*****
u16      fl_ua_getsum(u16 *sum, u32 top, u32 bottom)
{
    u16      rc;

    /*****
    /*      set params
    /*****
    // set params
    set_range_prm(fl_cmd_prm, top, bottom); // set SAH/SAM/SAL, EAH/EAM/EAL

    /*****
    /*      send command
    /*****

    fl_wait(tCOM); // wait before sending command

    put_cmd_ua(FL_COM_GET_CHECK_SUM, 7, fl_cmd_prm); // send GET VERSION command

    rc = get_sfrm_ua(fl_ua_sfrm, tWT16_MAX); // get status frame
    switch(rc) {
        case FLC_NO_ERR: break; // continue
        // case FLC_DFTO_ERR: return rc; break; // case [C]
        default: return rc; break; // case [B]
    }

    /*****
    /*      get data frame (Checksum data)
    /*****
    rc = get_dfrm_ua(fl_rxdata_frm, tFD1_MAX); // get status frame
    if (rc){ // if no error,
        return rc; // case [D]
    }

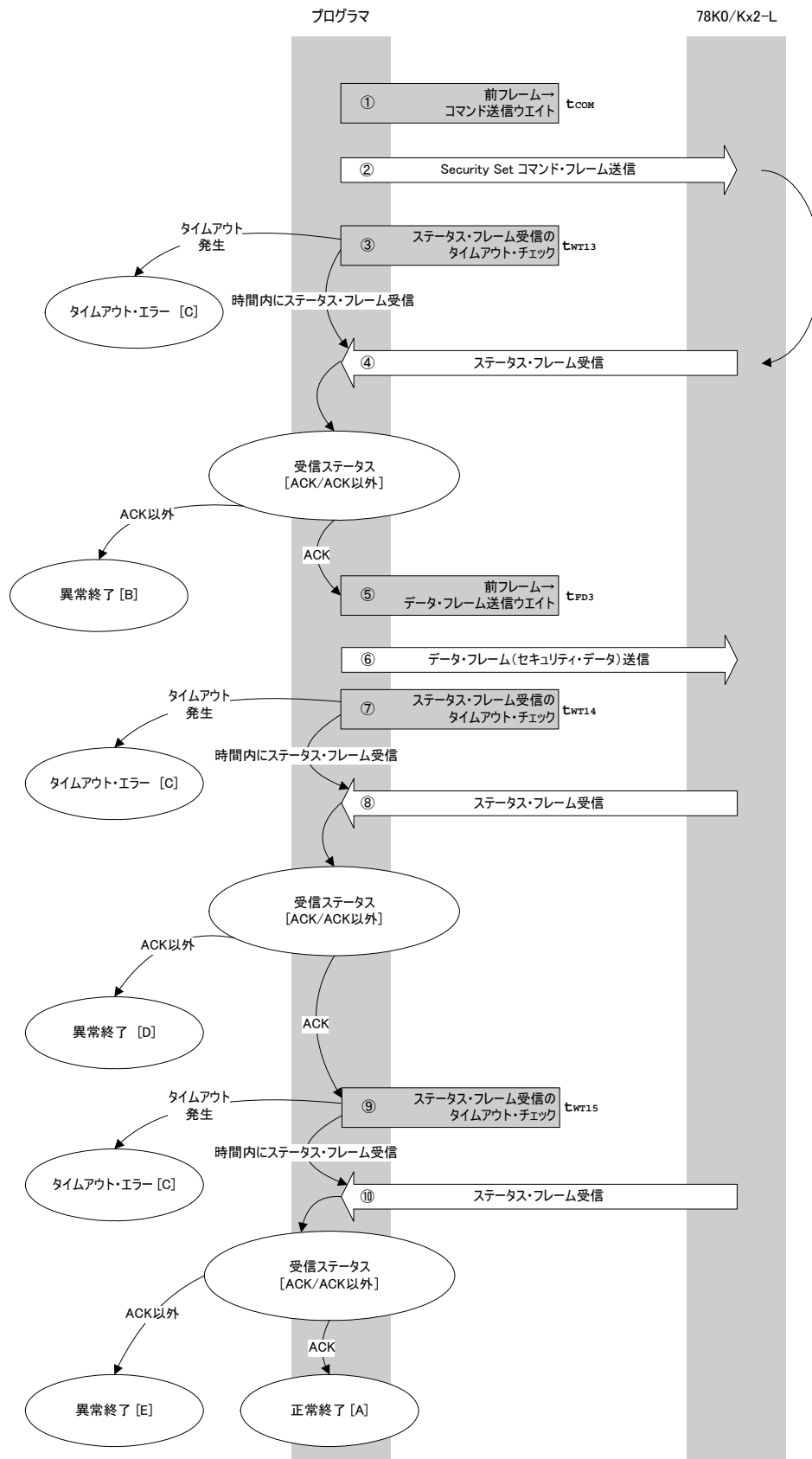
    *sum = (fl_rxdata_frm[OFS_STA_PLD] << 8) + fl_rxdata_frm[OFS_STA_PLD+1]; // set SUM
data
    return rc; // case [A]
}

```

4. 14 Security Setコマンド

4. 14. 1 処理手順チャート

Security Setコマンド処理手順



4. 14. 2 処理手順説明

直前のフレームからコマンド送信までのウェイトをします（ウェイト時間 t_{COM} ）。
 コマンド・フレーム送信処理により、**Security Setコマンド**を送信します。
 コマンド送信からステータス・フレーム受信までのタイムアウト・チェックを行います。
 タイムアウトが発生した場合は**タイムアウト・エラー[C]**となります
 （タイムアウト時間 t_{WT13} ）。
 ステータス・コードをチェックします。

ST1 = ACKの場合 : に進みます。
ST1 = ACK以外の場合 : **異常終了[B]**です。

直前のフレームからデータ・フレーム送信までのウェイトをします（ウェイト時間 t_{FD3} ）。
 データ・フレーム送信処理によりデータ・フレーム(セキュリティ設定データ)を送信します。
 ステータス・フレーム受信までのタイムアウト・チェックを行います。
 タイムアウトが発生した場合は**タイムアウト・エラー[C]**となります
 （タイムアウト時間 t_{WT14} ）。
 ステータス・コードをチェックします。

ST1 = ACKの場合 : に進みます。
ST1 = ACK以外の場合 : **異常終了[D]**です。

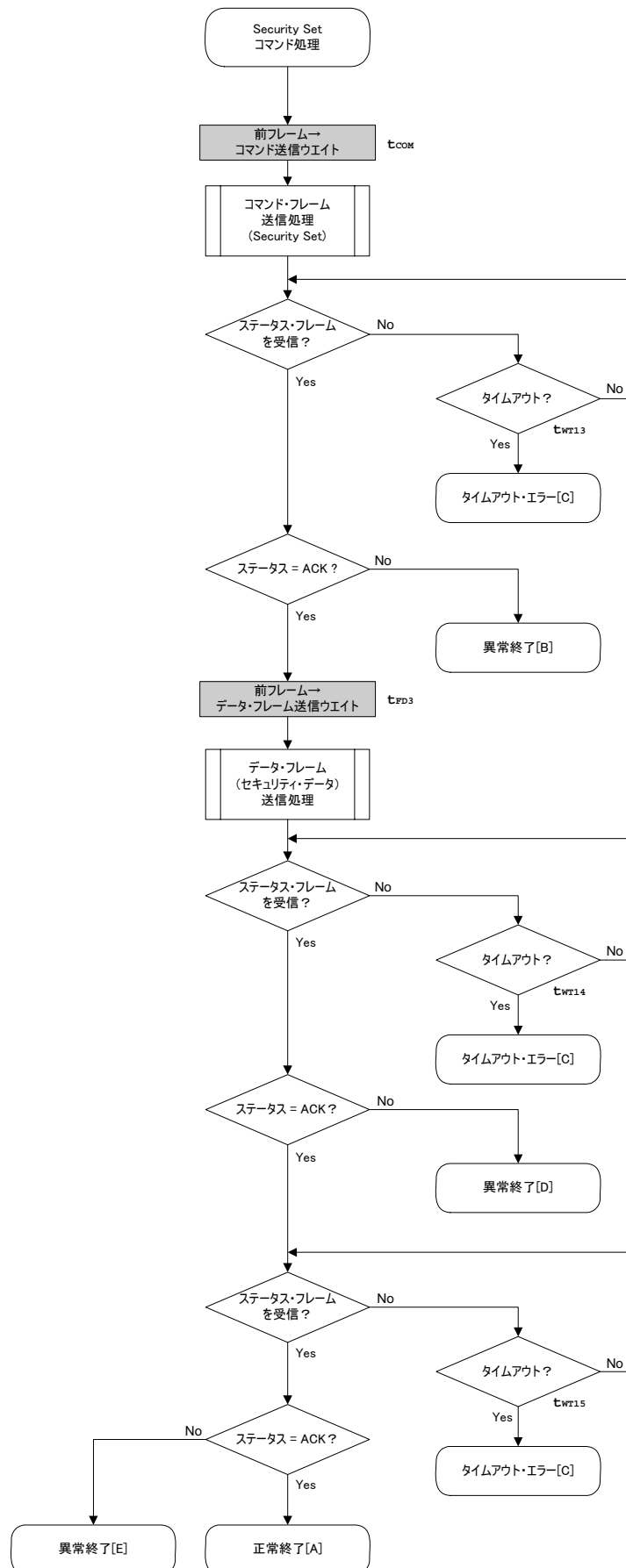
ステータス・フレーム受信までのタイムアウト・チェックを行います。
 タイムアウトが発生した場合は**タイムアウト・エラー[C]**となります
 （タイムアウト時間 t_{WT15} ）。
 ステータス・コードをチェックします。

ST1 = ACKの場合 : **正常終了[A]**です。
ST1 = ACK以外の場合 : **異常終了[E]**です。

4. 14. 3 終了時の内容

終了内容		ステータス・コード	内 容
正常終了 [A]	正常応答 (ACK)	06H	コマンドが正常に実行され、セキュリティ設定データが正しく設定されたことを示します。
異常終了 [B]	パラメータ・エラー	05H	コマンド情報 (パラメータ) が00Hではありません。 または、パラメータBOTが03Hではありません。
	チェックサム・エラー	07H	送信したコマンド・フレーム、またはデータ・フレームのチェックサムが異常です。
	プロテクト・エラー	10H	すでに禁止されているフラグを許可にしようとしています。
	否定応答 (NACK)	15H	コマンド・フレーム・データが異常です (データ長 (LEN) 不正, ETXなしなど)。
タイムアウト・エラー [C]		-	ステータス・フレーム、またはデータ・フレームの受信でタイムアウトが発生しました。
異常終了 [D], [E]	MRG10エラー	1AH	セキュリティ・データの書き込みに失敗しました。
	MRG11エラー	1BH	
	Writeエラー	1CH	

4.14.4 フロー・チャート



4. 14. 5 サンプル・プログラム

Security Setコマンド処理のサンプル・プログラムです。

```

/*****
/*
/*      Set security flag command
/*
/*
/*****
/* [i] u8 scf      ... Security flag data
/* [r] ul6         ... error code
/*****
ul6      fl_ua_setscf(u8 scf, u8 bot)
{
    ul6      rc;

    /*****
    /*      set params
    /*
    /*****
    fl_cmd_prm[0] = 0x00;          // "BLK" (must be 0x00)
    fl_cmd_prm[1] = 0x00;          // "PAG" (must be 0x00)

    fl_txdata_frm[0] = scf | 0b11101000; // "FLG" (bit 7,6,5,3 must be '1')
    fl_txdata_frm[1] = bot;           // "BOT"
    fl_txdata_frm[2] = 0xff;          // (must be 0xff)
    fl_txdata_frm[3] = 0xff;          // (must be 0xff)
    fl_txdata_frm[4] = 0xff;          // (must be 0xff)
    fl_txdata_frm[5] = 0xff;          // (must be 0xff)
    fl_txdata_frm[6] = 0xff;          // (must be 0xff)
    fl_txdata_frm[7] = 0xff;          // (must be 0xff)

    /*****
    /*      send command
    /*
    /*****
    fl_wait(tCOM);                  // wait before sending command

    put_cmd_ua(FL_COM_SET_SECURITY, 3, fl_cmd_prm);

    rc = get_sfrm_ua(fl_ua_sfrm, tWT13_TO);          // get status frame
    switch(rc) {
        case FLC_NO_ERR:                break; // continue
    // case FLC_DFTO_ERR:    return rc;    break; // case [C]
        default:                return rc;    break; // case [B]
    }

    /*****
    /*      send data frame (security setting data)
    /*
    /*****

    fl_wait(tFD3);
    put_dfrm_ua(6, fl_txdata_frm, true);            // send securithi setting data

    rc = get_sfrm_ua(fl_ua_sfrm, tWT14_MAX);          // get status frame
    switch(rc) {
        case FLC_NO_ERR:                break; // continue
    // case FLC_DFTO_ERR:    return rc;    break; // case [C]
        default:                return rc;    break; // case [B]
    }

```

```

/*****
/*      Check internally verify      */
/*****/
rc = get_sfrm_ua(fl_ua_sfrm, tWT15_MAX);           // get status frame
// switch(rc) {
//
//      case   FLC_NO_ERR:      return rc;      break; // case [A]
//      case   FLC_DFTO_ERR:    return rc;      break; // case [C]
//      default:                return rc;      break; // case [B]
// }
return rc;
}
```

第5章 フラッシュ・メモリ・プログラミング・パラメータ特性

この章では、フラッシュ・メモリ・プログラミング・モード時のプログラマとデバイス（78K0/Kx2-L, 78K0/lx2）の間のパラメータ特性を記載しています。その他の電気的特性は、デバイス（78K0/Kx2-L, 78K0/lx2）各製品のユーザーズ・マニュアルを参照のうえ、設計してください。

5.1 78K0/Kx2-Lのフラッシュ・メモリ・パラメータ特性

5.1.1 フラッシュ・メモリ・プログラミング・モード・セット時間

項 目	略 号	MIN.	TYP.	MAX.
Ready start time from $\overline{\text{RESET}}$	t_{R0}	1.5 ms		3 s
Wait for Reset command	t_{0C}	14.0 μ s		3 s

（１）モード設定時間の詳細

項 目	略 号	MIN.	TYP.	MAX.
Release of POC to TOOLC	T1	1.0 μ s		
TOOLD to TOOLC pulse start	T3	110.0 μ s		
Minimum Low width of TOOLC	T4	110.0 μ s		
Minimum High width of TOOLC	T5	110.0 μ s		
Setup time of TOOLC to TOOLD for Mode Setting	T6	55.0 μ s		
Minimum Low width of TOOLD	T7	110.0 μ s		
Minimum High width of TOOLD	T8	110.0 μ s		
Setup time of TOOLD to TOOLC for Mode Setting	T9	55.0 μ s		
Setup time of TOOLD to $\overline{\text{RESET}}$ for Mode Setting	T10	55.0 μ s		
Ready start time from $\overline{\text{RESET}}$	T11 (= t_{R0})	1.5 ms		3 s
Maximum Setup time	T12			130.0 ms

（２）モード遷移時間の詳細（プログラミング・モードから通常モード）

項 目	略 号	MIN.	TYP.	MAX.
$\overline{\text{RESET}}$ ~ TOOLC, TOOLD	T20	500.0 μ s		
Minimum Low width TOOLC and TOOLD	T21	110.0 μ s		
Minimum High width TOOLC and TOOLD	T22	55.0 μ s		

5.1.2 プログラミング特性

ウエイト	条 件	略 号	MIN.	MAX.
データ・フレーム～データ・フレーム	データ・フレーム受信	tDR	21.0 μ s	
	データ・フレーム送信	tDT	0 ^{注1}	
ステータス・フレーム送信～データ・フレーム送信(1)		tFD1 ^{注2}	0 ^{注1}	
ステータス・フレーム送信～データ・フレーム送信(2)		tFD2	0 ^{注1}	
ステータス・フレーム送信～データ・フレーム受信(3)		tFD3	20.0 μ s	
ステータス・フレーム送信～コマンド・フレーム受信		tCOM	30.8 μ s ^{注1}	

注1. プログラマは連続受信許可にしておいてください。

2. 1ブロックの場合の時間

備考 ウエイトには、次のような定義があります。

< tDR, tFD3, tCOM >

78K0/Kx2-Lは、直前の通信完了後、MIN.後から次の通信が可能となります。

プログラマは、直前の通信完了後、MIN.～MAX.時間内に次データの送信を行ってください。

MAX.時間の規定はありませんが、3 s程度を目安としてください。

< tDT, tFD1, tFD2 >

78K0/Kx2-Lは、直前の通信完了後、MIN.後から次の通信が可能となります。

プログラマは、直前の通信完了後、MIN.～MAX.時間内に次データの受信を行ってください。

MAX.時間の規定はありませんが、3 s程度を目安としタイムアウトを設定してください。

5.1.3 コマンド特性

コマンド	略号	条件	MIN.	MAX.
Reset	t _{WT0}	-	注1	
Chip Erase	t _{WT1}	-	(54.4 + 6.2 × 全ブロック数) ms	(883.6 + 136.1 × 全ブロック数) ms
Block Erase	t _{WT2} ^{注2}	-	15.2 ms	(0.3 + 190.3 × 同時選択消去の実行回数 + 139.9 × 消去するブロック数) ms
Programming	t _{WT3}	-	注1	
	t _{WT4} ^{注3}	-	7.4 ms	132.9 ms
	t _{WT5} ^{注4}	-	注1	723.6 ms
Verify	t _{WT6}	-	注1	
	t _{WT7} ^{注3}	-	注1	
Block Blank Check	t _{WT8} ^{注4}	-	注1	9.9 ms
Baud Rate Set	t _{WT10}	-	62.7 μs	
Silicon Signature	t _{WT11}	-	注1	
Version Get	t _{WT12}	-	注1	
Security Set	t _{WT13}	-	注1	
	t _{WT14}	-	16.9 ms	351.3 ms
	t _{WT15}	-	注1	352.5 ms
Checksum	t _{WT16}	-	注1	

注1. プログラマは、コマンド・フレーム送信前に受信許可にしておいてください。

2. 同時選択消去の実行回数の求め方に関しては、5.3 Block Eraseコマンドにおける同時選択消去についてを参照してください。
3. 256バイトの場合の時間
4. 1ブロックの場合の時間

備考 ウェイトには、次のような定義があります。

< t_{WT0} ~ t_{WT16} >

78K0/Kx2-Lは、MIN. ~ MAX.時間内に各コマンド処理を終了します。

プログラマは、受信フレームのスタート・ピットをMAX.時間までステータス・チェックを行う必要があります。

MAX.時間の規定がないコマンドについては、3 s程度を目安としてください。

5.2 78K0/Ix2のフラッシュ・メモリ・パラメータ特性

5.2.1 フラッシュ・メモリ・プログラミング・モード・セット時間

項 目	略 号	MIN.	TYP.	MAX.
Ready start time from $\overline{\text{RESET}}$ 注	t _{R0}	1.5 ms		3 s
Wait for Reset command	t _{0c}	14.0 μ s		3 s

注 下表を参照してください。下表の値は参考値です。下表の値は発振器などに依存するため、最大値を上回ることや、最小値を下回ることがあります。

(1) モード設定時間の詳細

項 目	略 号	MIN.	TYP.	MAX.
Release of POC to TOOLC	T1	1.0 μ s		
TOOLD to TOOLC pulse start	T3	110.0 μ s		
Minimum Low width of TOOLC	T4	110.0 μ s		
Minimum High width of TOOLC	T5	110.0 μ s		
Setup time of TOOLC to TOOLD for Mode Setting	T6	55.0 μ s		
Minimum Low width of TOOLD	T7	110.0 μ s		
Minimum High width of TOOLD	T8	110.0 μ s		
Setup time of TOOLD to TOOLC for Mode Setting	T9	55.0 μ s		
Setup time of TOOLD to $\overline{\text{RESET}}$ for Mode Setting	T10	55.0 μ s		
Ready start time from $\overline{\text{RESET}}$	T11 (= t _{R0})	1.5 ms		3 s
Maximum Setup time (13-bit counter)	T12			130.0 ms ^注

注 @30 kHz \pm 50%

(2) モード遷移時間の詳細 (プログラミング・モードから通常モード)

項 目	略 号	MIN.	TYP.	MAX.
$\overline{\text{RESET}}$ ~ TOOLC, TOOLD	T20	500.0 μ s		
Minimum Low width TOOLC and TOOLD	T21	110.0 μ s		
Minimum High width TOOLC and TOOLD	T22	55.0 μ s		

5.2.2 プログラミング特性

ウエイト	条 件	略 号	MIN.	MAX.
データ・フレーム～データ・フレーム	データ・フレーム受信	tDR	21.0 μ s	
	データ・フレーム送信	tDT	0 ^{注1}	
ステータス・フレーム送信～データ・フレーム送信(1)		tFD1 ^{注2}	0 ^{注1}	
ステータス・フレーム送信～データ・フレーム送信(2)		tFD2	0 ^{注1}	
ステータス・フレーム送信～データ・フレーム受信(3)		tFD3	20.0 μ s	
ステータス・フレーム送信～コマンド・フレーム受信		tCOM	30.8 μ s ^{注1}	

注1. プログラマは連続受信許可にしておいてください。

- 1ブロックの場合の時間

備考 ウエイトには、次のような定義があります。

< tDR, tFD3, tCOM >

78K0/Ix2は、直前の通信完了後、MIN.後から次の通信が可能となります。

プログラマは、直前の通信完了後、MIN.～MAX.時間内に次データの送信を行ってください。

MAX.時間の規定はありませんが、3 s程度を目安としてください。

< tDT, tFD1, tFD2 >

78K0/Ix2は、直前の通信完了後、MIN.後から次の通信が可能となります。

プログラマは、直前の通信完了後、MIN.～MAX.時間内に次データの受信を行ってください。

MAX.時間の規定はありませんが、3 s程度を目安としタイムアウトを設定してください。

5.2.3 コマンド特性

コマンド	略号	条件	MIN.	MAX.
Reset	t _{WT0}	-	注1	
Chip Erase	t _{WT1}	-	(54.4 + 6.2 × 全ブロック数) ms	(883.6 + 136.1 × 全ブロック数) ms
Block Erase	t _{WT2} ^{注2}	-	15.2 ms	(0.3 + 190.3 × 同時選択消去の実行回数 + 139.9 × 消去するブロック数) ms
Programming	t _{WT3}	-	注1	
	t _{WT4} ^{注3}	-	7.4 ms	132.9 ms
	t _{WT5} ^{注4}	-	注1	723.6 ms
Verify	t _{WT6}	-	注1	
	t _{WT7} ^{注3}	-	注1	
Block Blank Check	t _{WT8} ^{注4}	-	注1	9.9 ms
Baud Rate Set	t _{WT10}	-	62.7 μs	
Silicon Signature	t _{WT11}	-	注1	
Version Get	t _{WT12}	-	注1	
Security Set	t _{WT13}	-	注1	
	t _{WT14}	-	16.9 ms	351.3 ms
	t _{WT15}	-	注1	352.5 ms
Checksum	t _{WT16}	-	注1	

注1. プログラマは、コマンド・フレーム送信前に受信許可にしておいてください。

2. 同時選択消去の実行回数の求め方に関しては、5.3 Block Eraseコマンドにおける同時選択消去についてを参照してください。
3. 256バイトの場合の時間
4. 1ブロックの場合の時間

備考 ウェイトには、次のような定義があります。

< t_{WT0} ~ t_{WT16} >

78K0/Ix2は、MIN. ~ MAX.時間内に各コマンド処理を終了します。

プログラマは、受信フレームのスタート・ピットをMAX.時間までステータス・チェックを行う必要があります。

MAX.時間の規定がないコマンドについては、3 s程度を目安としてください。

5.3 Block Eraseコマンドにおける同時選択消去について

78K0/Kx2-L, 78K0/lx2のBlock Eraseコマンド実行は、複数ブロックを同時に消去する“同時選択消去”を繰り返し実行することにより実現しています。

したがって、ブロック消去コマンド実行時のウェイト時間は、“同時選択消去”の実行時間の総和となります。
“同時選択消去の実行時間の総和”を算出するためには、同時選択消去の実行回数（M）を算出する必要があります。

Mは、同時に消去するブロック数（同時選択消去ブロック数）を求めながら、算出します。

次に同時選択消去ブロック数とMの求め方を記載します。

5.3.1 同時選択消去ブロック数の求め方

選択消去ブロック数は次の条件をすべて満たす，“1, 2, 4, 8, 16, 32, 64, 128”のいずれかの数値になります。

【条件1】

消去ブロック数 同時選択消去ブロック数

【条件2】

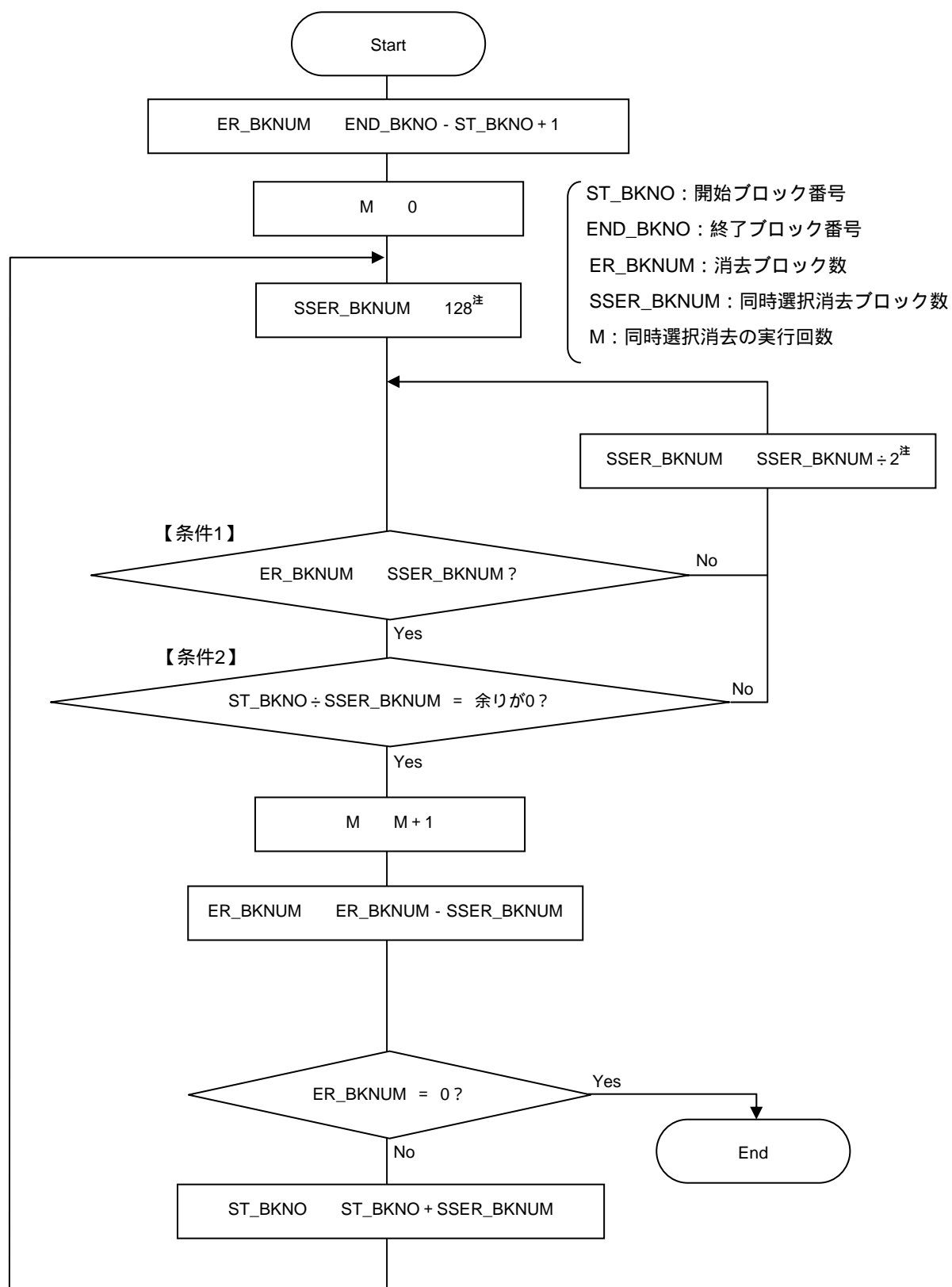
開始ブロック番号 ÷ 同時選択消去ブロック数 = 余りが0

【条件3】

【条件1】と【条件2】を満たす最も大きな数値

5.3.2 同時選択消去の実行回数 (M) の求め方

Mの算出方法をフローで表現すると次のようになります。



注 SSER_BKNUMの最大値(128)から,【条件1】と【条件2】に当てはまる数値を, SSER_BKNUM ÷ 2しながら算出することで,【条件3】は満たされます。

例1 ブロック1～127を消去する場合（N（消去するブロック数）= 127）

最初の開始ブロック番号は1で、消去ブロック数が127であることから、【条件1】を満たす数値は、1, 2, 4, 8, 16, 32, 64となります。

さらに【条件2】を満たす数値は、1となり、【条件3】を満たす数値は“1”であるため、同時選択消去ブロック数は“1”となることから、ブロック1のみ消去します。

ブロック1を消去すると、次の開始ブロック番号は2で、消去ブロック数が126であることから、【条件1】を満たす数値は、1, 2, 4, 8, 16, 32, 64となります。

さらに【条件2】を満たす数値は、1, 2となり、【条件3】を満たす数値は“2”であるため、同時選択消去ブロック数は“2”となることから、ブロック2～3を消去します。

ブロック2～3を消去すると、次の開始ブロック番号は4で、消去ブロック数が124であることから、【条件1】を満たす数値は、1, 2, 4, 8, 16, 32, 64となります。

さらに【条件2】を満たす数値は、1, 2, 4となり、【条件3】を満たす数値は“4”であるため、同時選択消去ブロック数は“4”となることから、ブロック4～7を消去します。

ブロック4～7を消去すると、次の開始ブロック番号は8で、消去ブロック数が120であることから、【条件1】を満たす数値は、1, 2, 4, 8, 16, 32, 64となります。

さらに【条件2】を満たす数値は、1, 2, 4, 8となり、【条件3】を満たす数値は“8”であるため、同時選択消去ブロック数は“8”となることから、ブロック8～15を消去します。

ブロック8～15を消去すると、次の開始ブロック番号は16で、消去ブロック数が112であることから、【条件1】を満たす数値は、1, 2, 4, 8, 16, 32, 64となります。

さらに【条件2】を満たす数値は、1, 2, 4, 8, 16となり、【条件3】を満たす数値は“16”であるため、同時選択消去ブロック数は“16”となることから、ブロック16～31を消去します。

ブロック16～31を消去すると、次の開始ブロック番号は32で、消去ブロック数が96であることから、【条件1】を満たす数値は、1, 2, 4, 8, 16, 32, 64となります。

さらに【条件2】を満たす数値は、1, 2, 4, 8, 16, 32となり、【条件3】を満たす数値は“32”であるため、同時選択消去ブロック数は“32”となることから、ブロック32～63を消去します。

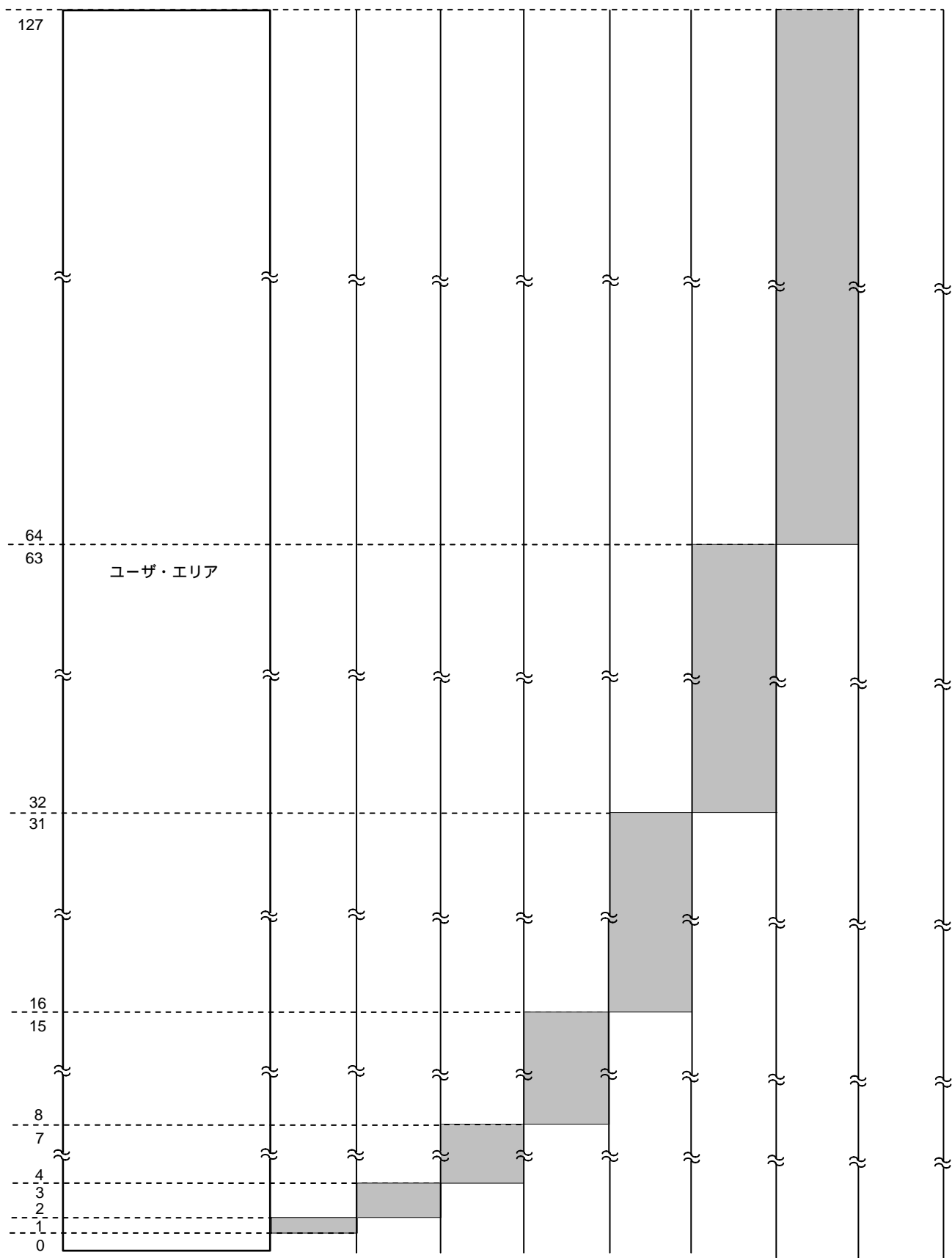
ブロック32～63を消去すると、次の開始ブロック番号は64で、消去ブロック数が64であることから、【条件1】を満たす数値は、1, 2, 4, 8, 16, 32, 64となります。

さらに【条件2】を満たす数値は、1, 2, 4, 8, 16, 32, 64となり、【条件3】を満たす数値は“64”であるため、同時選択消去ブロック数は“64”となることから、ブロック64～127を消去します。

以上より、ブロック1～127を消去する場合、1, 2～3, 4～7, 8～15, 16～31, 32～63, 64～127の7回、同時選択消去を実行するため、M = 7となります。

同時選択消去実行時のブロック構成（ブロック1～127を消去する場合）

<ブロック番号>



<同時選択消去可能なブロックの範囲>

例2 ブロック5～10を消去する場合（N（消去するブロック数）= 6）

最初の開始ブロック番号は5で、消去ブロック数が6であることから、【条件1】を満たす数値は、1, 2, 4となります。

さらに【条件2】を満たす数値は、1となり、【条件3】を満たす数値は“1”であるため、同時選択消去ブロック数は“1”となることから、ブロック5のみ消去します。

ブロック5を消去すると、次の開始ブロック番号は6で、消去ブロック数が5であることから、【条件1】を満たす数値は、1, 2, 4となります。

さらに【条件2】を満たす数値は、1, 2となり、【条件3】を満たす数値は“2”であるため、同時選択消去ブロック数は“2”となることから、ブロック6～7を消去します。

ブロック6～7を消去すると、次の開始ブロック番号は8で、消去ブロック数が3であることから、【条件1】を満たす数値は、1, 2となります。

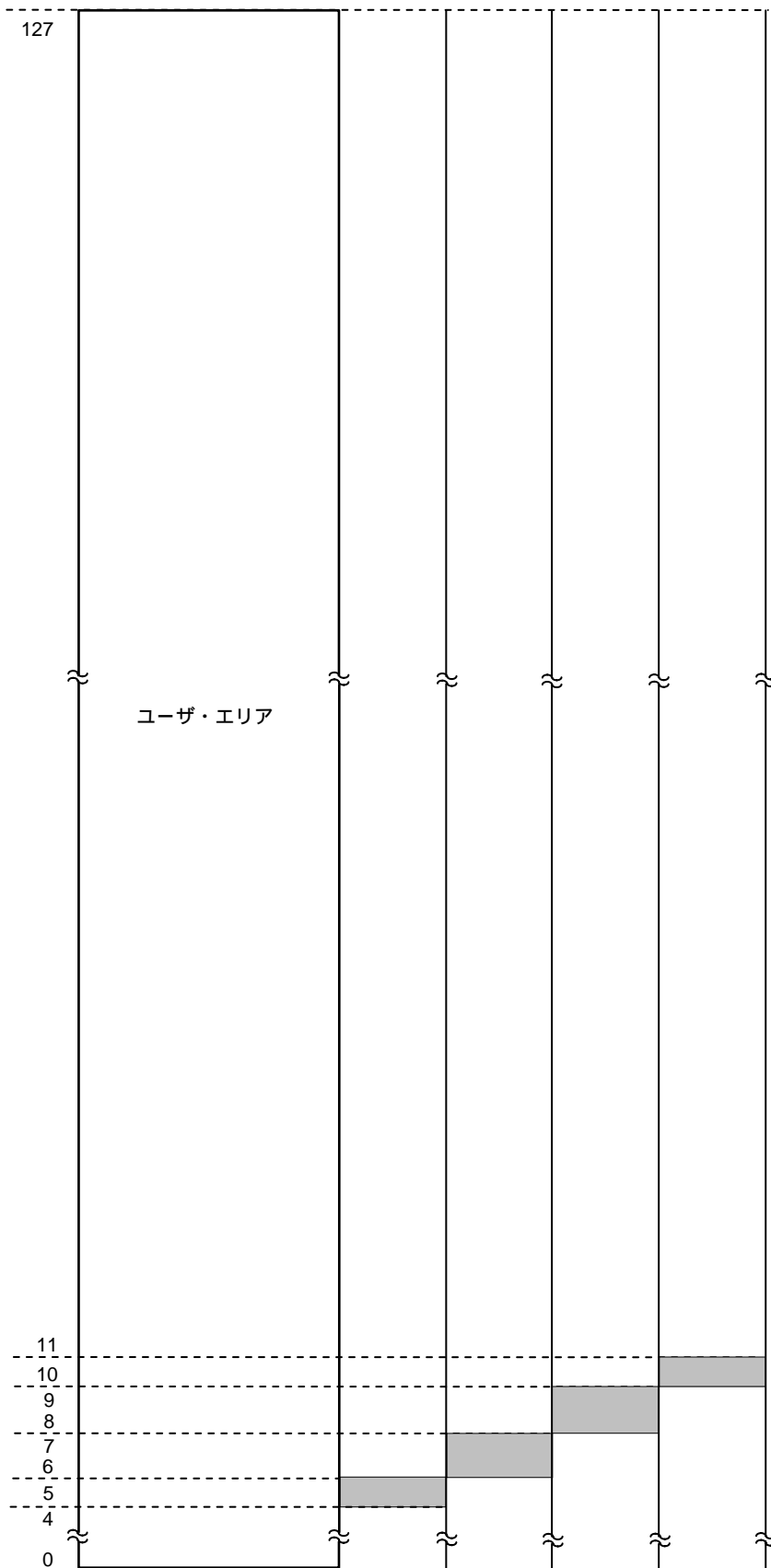
さらに【条件2】を満たす数値は、1, 2となり、【条件3】を満たす数値は“2”であるため、同時選択消去ブロック数は“2”となることから、ブロック8～9を消去する。

ブロック8～9を消去すると、次の開始ブロック番号は10で、消去ブロック数が1であることから、【条件1】を満たす数値は、1となり、これは、【条件2】と【条件3】も満たしているため、同時選択消去ブロック数は“1”となることから、ブロック10を消去する。

以上より、ブロック5～10を消去する場合、5, 6～7, 8～9, 10の4回、同時選択消去を実行するため、M = 4となります。

同時選択消去実行時のブロック構成（ブロック5～10を消去する場合）

< ブロック番号 >



< 同時選択消去可能なブロックの範囲 >

例3 ブロック25～73を消去する場合（N（消去するブロック数）= 49）

最初の開始ブロック番号は25で、消去ブロック数が49であることから、【条件1】を満たす数値は、1, 2, 4, 8, 16, 32となります。

さらに【条件2】を満たす数値は、1となり、【条件3】を満たす数値は“1”であるため、同時選択消去ブロック数は“1”となることから、ブロック25のみ消去します。

ブロック25を消去すると、次の開始ブロック番号は26で、消去ブロック数が48であることから、【条件1】を満たす数値は、1, 2, 4, 8, 16, 32となります。

さらに【条件2】を満たす数値は、1, 2となり、【条件3】を満たす数値は“2”であるため、同時選択消去ブロック数は“2”となることから、ブロック26～27を消去します。

ブロック26～27を消去すると、次の開始ブロック番号は28で、消去ブロック数が46であることから、【条件1】を満たす数値は、1, 2, 4, 8, 16, 32となります。

さらに【条件2】を満たす数値は、1, 2, 4となり、【条件3】を満たす数値は“4”であるため、同時選択消去ブロック数は“4”となることから、ブロック28～31を消去します。

ブロック28～31を消去すると、次の開始ブロック番号は32で、消去ブロック数が42であることから、【条件1】を満たす数値は、1, 2, 4, 8, 16, 32となります。

さらに【条件2】を満たす数値は、1, 2, 4, 8, 32となり、【条件3】を満たす数値は“32”であるため、同時選択消去ブロック数は“32”となることから、ブロック32～63を消去します。

ブロック32～63を消去すると、次の開始ブロック番号は64で、消去ブロック数が10であることから、【条件1】を満たす数値は、1, 2, 4, 8となります。

さらに【条件2】を満たす数値は、1, 2, 4, 8となり、【条件3】を満たす数値は“8”であるため、同時選択消去ブロック数は“8”となることから、ブロック64～71を消去します。

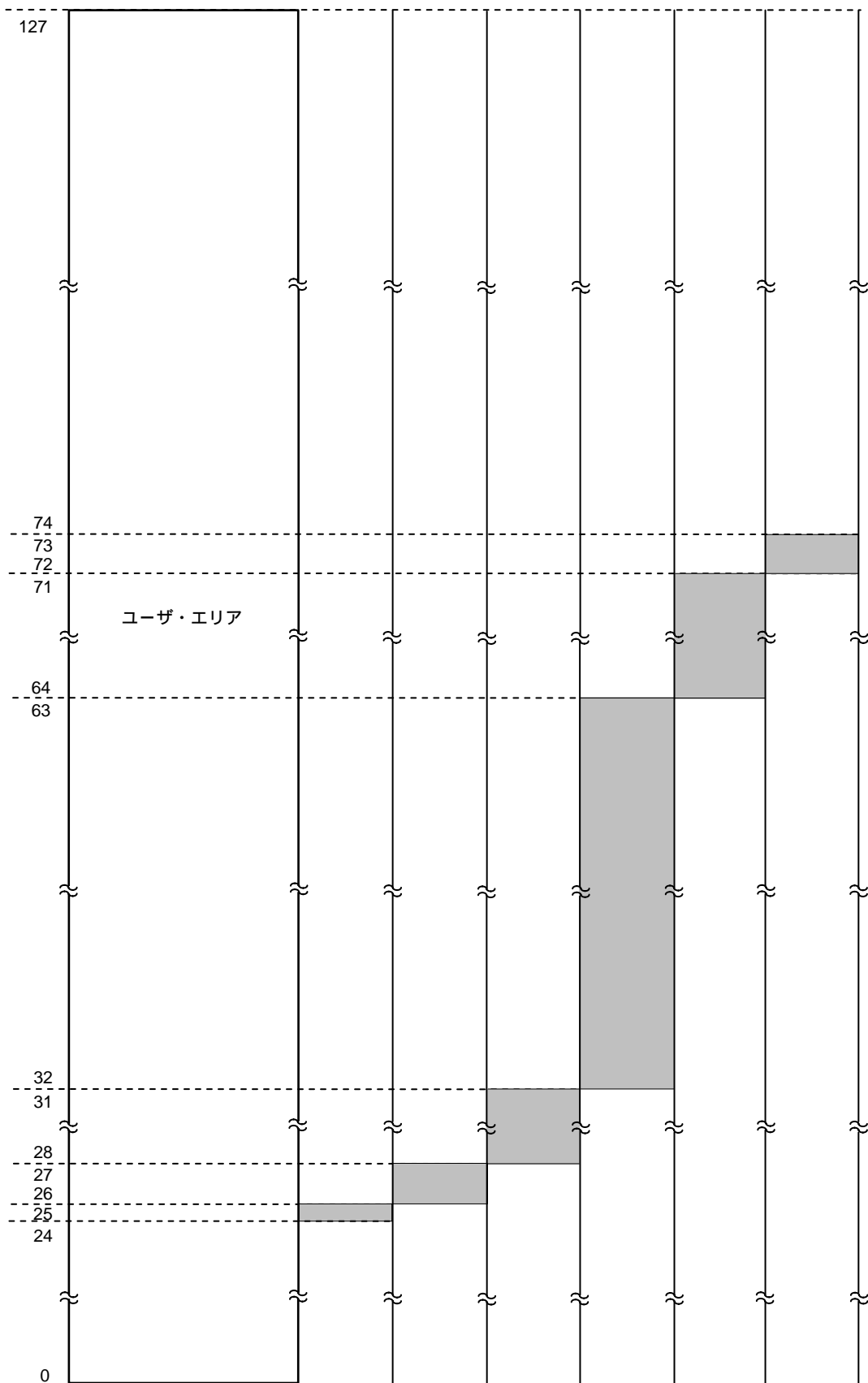
ブロック64～71を消去すると、次の開始ブロック番号は72で、消去ブロック数が2であることから、【条件1】を満たす数値は、1, 2となります。

さらに【条件2】を満たす数値は、1, 2となり、【条件3】を満たす数値は“2”であるため、同時選択消去ブロック数は“2”となることから、ブロック72～73を消去します。

以上より、ブロック25～73を消去する場合、25, 26～27, 28～31, 32～63, 64～71, 72～73の6回消去されるため、M = 6となります。

同時選択消去実行時のブロック構成（ブロック25～73を消去する場合）

<ブロック番号>

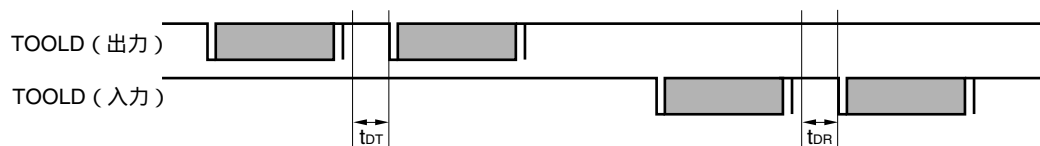


<同時選択消去可能なブロックの範囲>

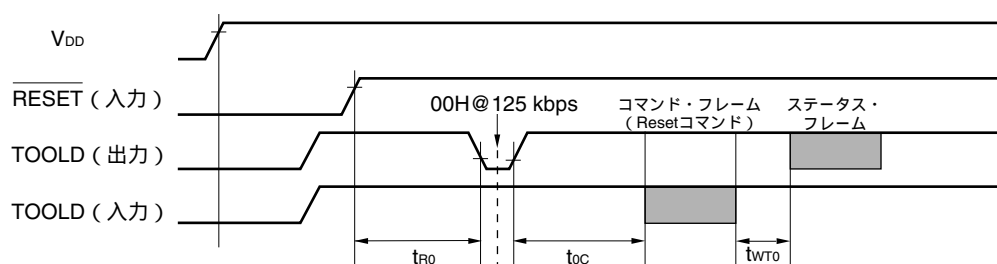
5.4 UART通信方式

次に示す図では説明のため、TOOLDを2本に分けて記載していますが、実際は単線です。

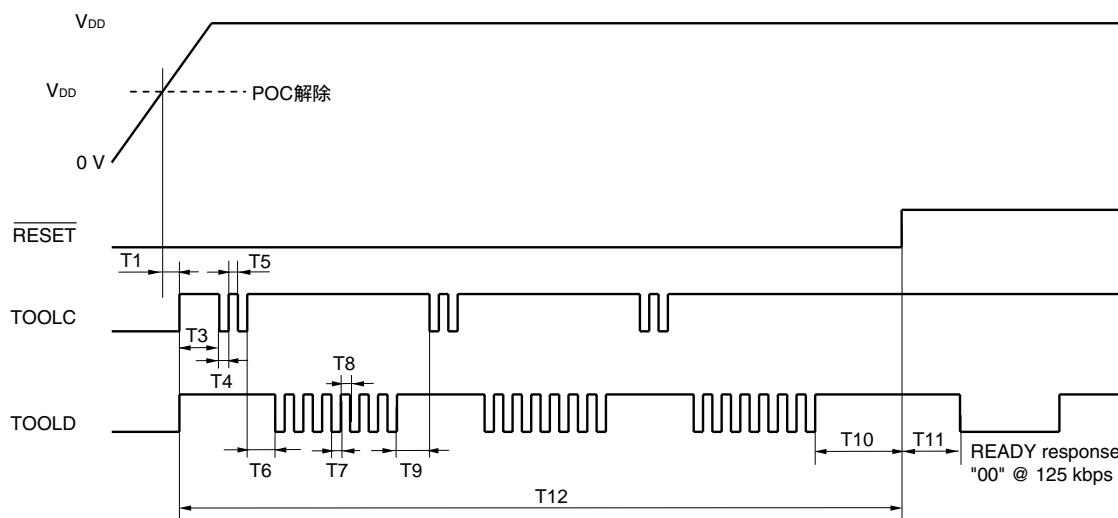
(1) データ・フレーム



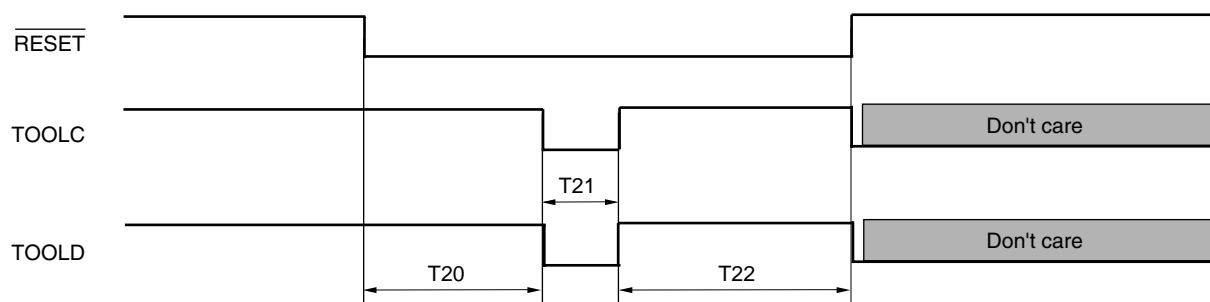
(2) プログラミング・モード設定 / Resetコマンド



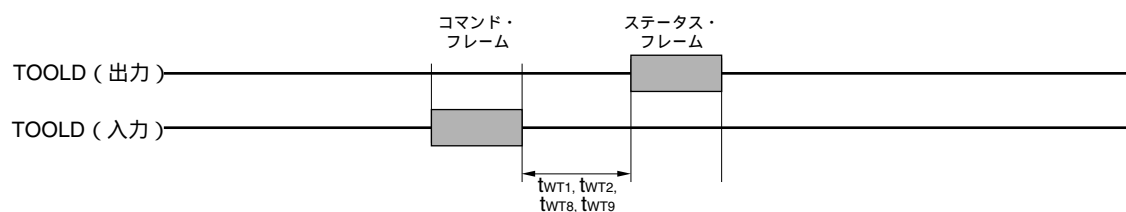
(a) プログラミング・モード設定の詳細



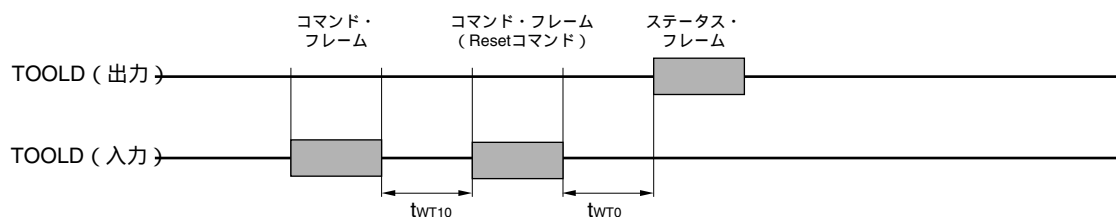
(3) プログラミング・モードから通常動作モードへの遷移



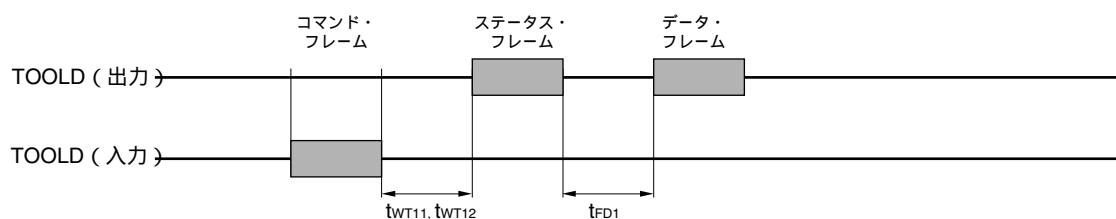
(4) Chip Eraseコマンド / Block Eraseコマンド / Block Blank Checkコマンド / Oscillating Frequency Setコマンド



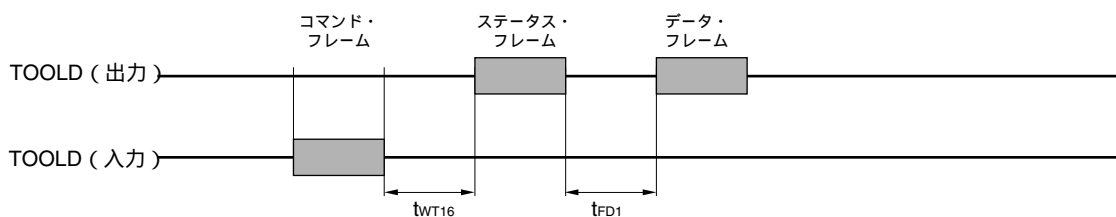
(5) Baud Rate Setコマンド



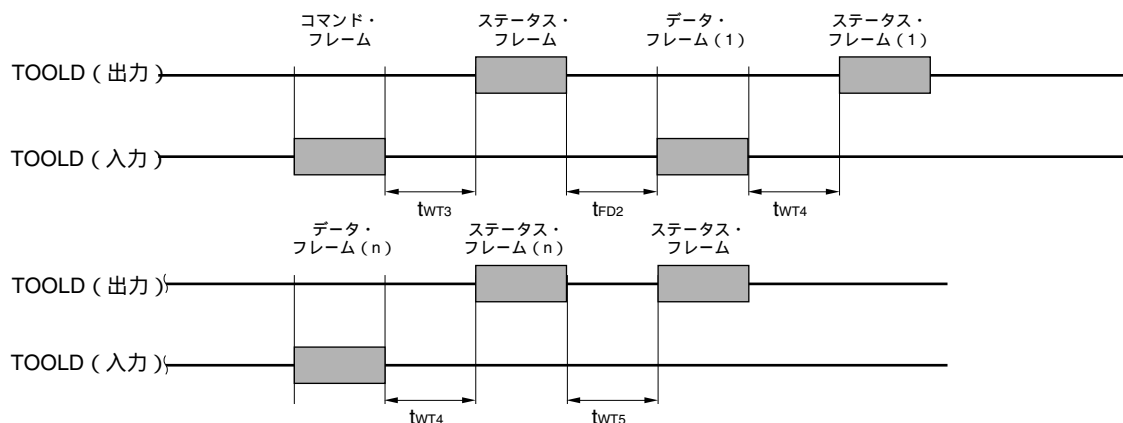
(6) Silicon Signatureコマンド / Version Getコマンド



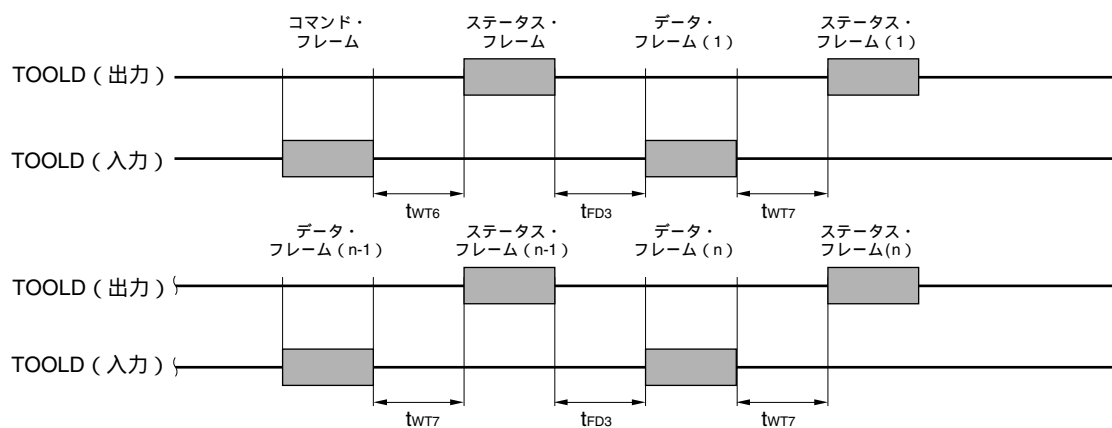
(7) Checksumコマンド



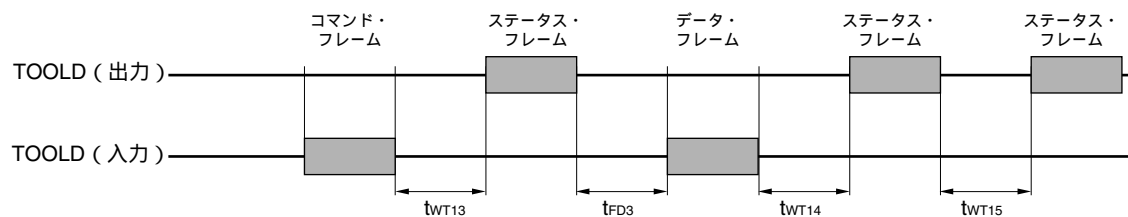
(8) Programmingコマンド



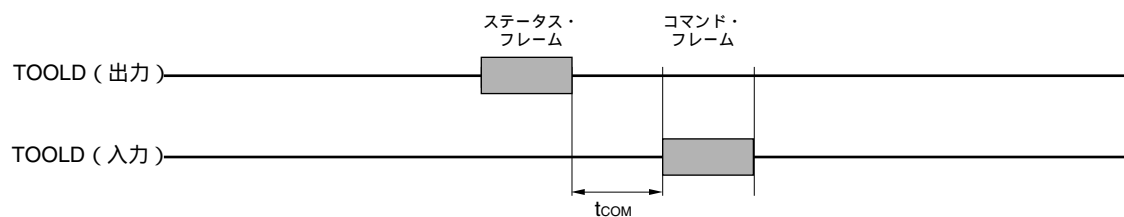
(9) Verifyコマンド



(10) Security Setコマンド



(11) コマンド・フレーム送信前のウェイト



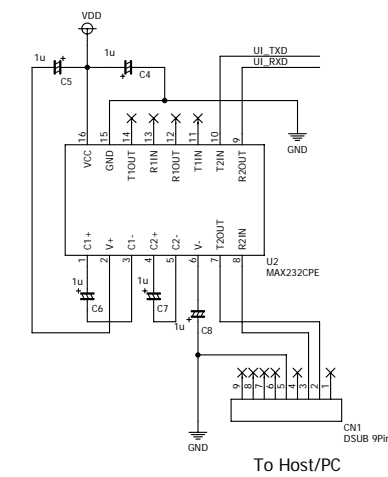
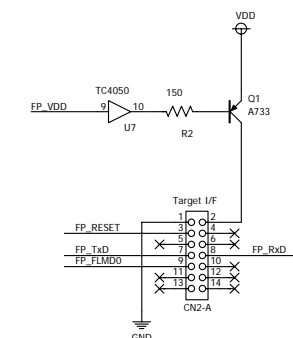
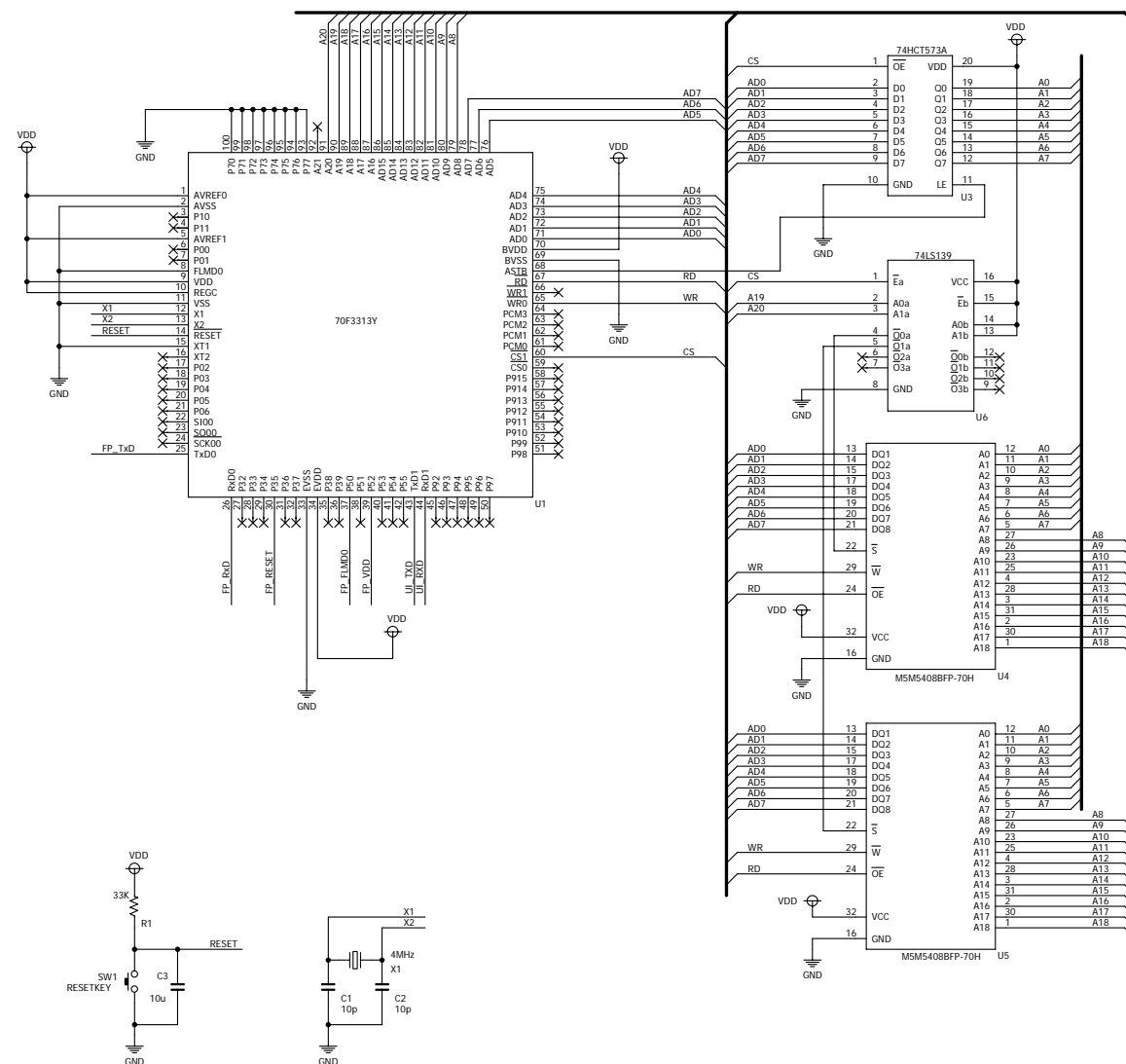
付録A 参考回路図

プログラマと78K0/Kx2-Lの参考回路図を図A - 1 , 図A - 2に示します。

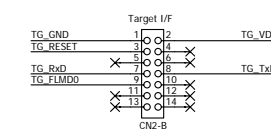
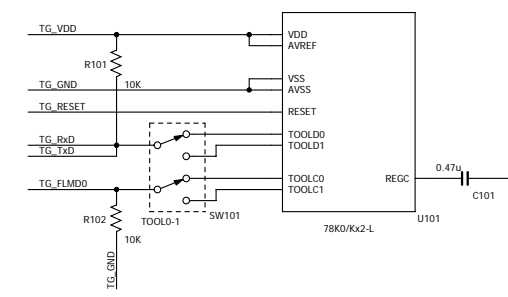
図A - 1 プログラマと78K0/Kx2-Lの参考回路図（メイン・ボード）

78K0/Kx2-L Flash Programmer Sample Application Main Board for Monowire UART I/F

(VDD = 5.0V)



78K0/Kx2-L Target Board for Monowire UART I/F

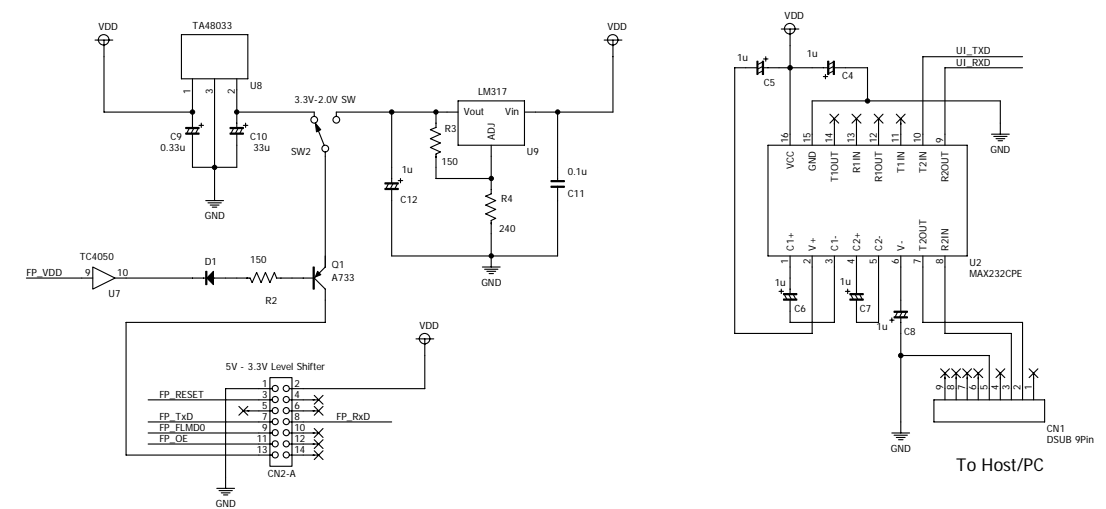
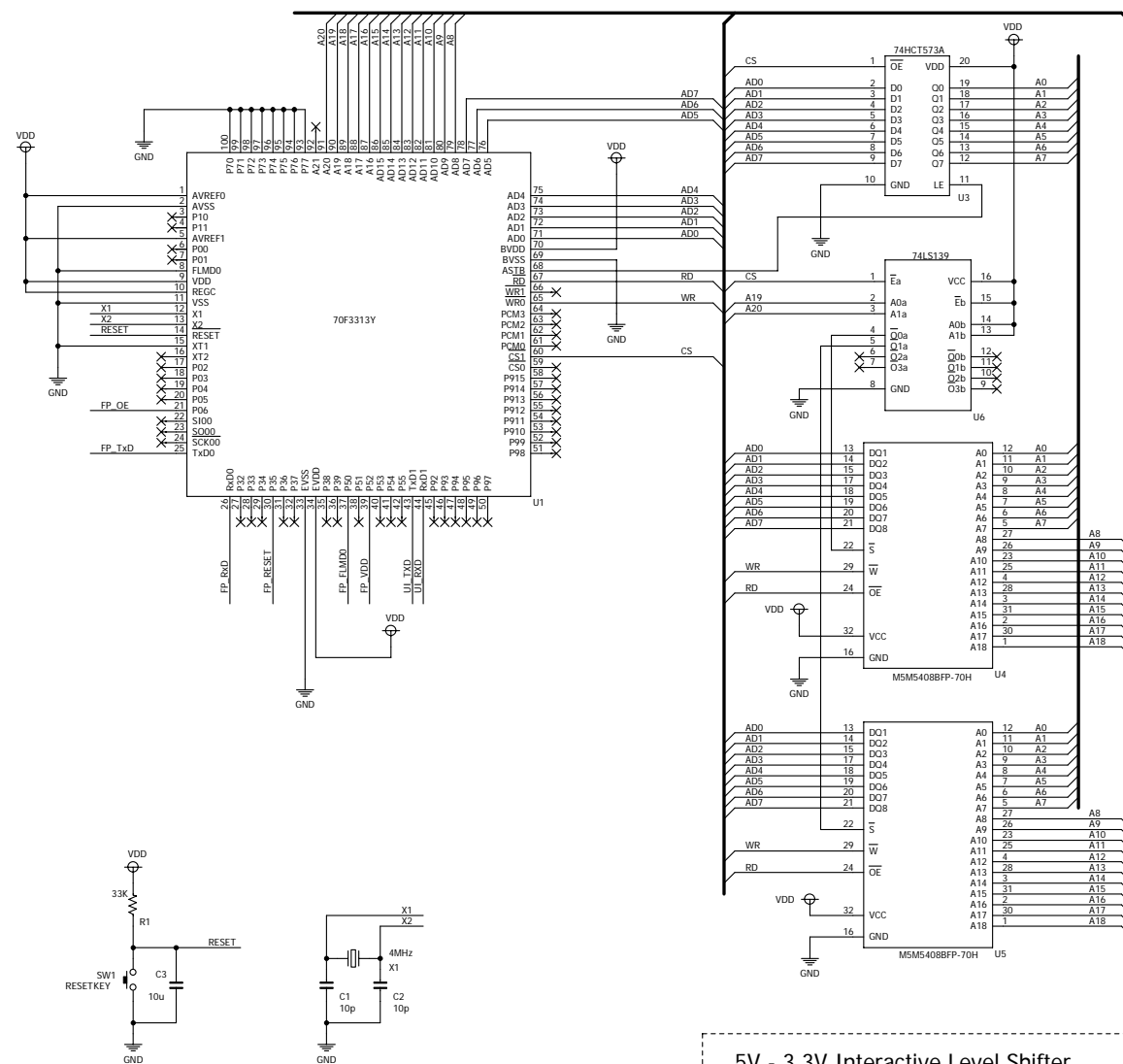


備考 回路図中の未使用端子に関しては、各製品のユーザーズ・マニュアルに従い、端子処理を行ってください。

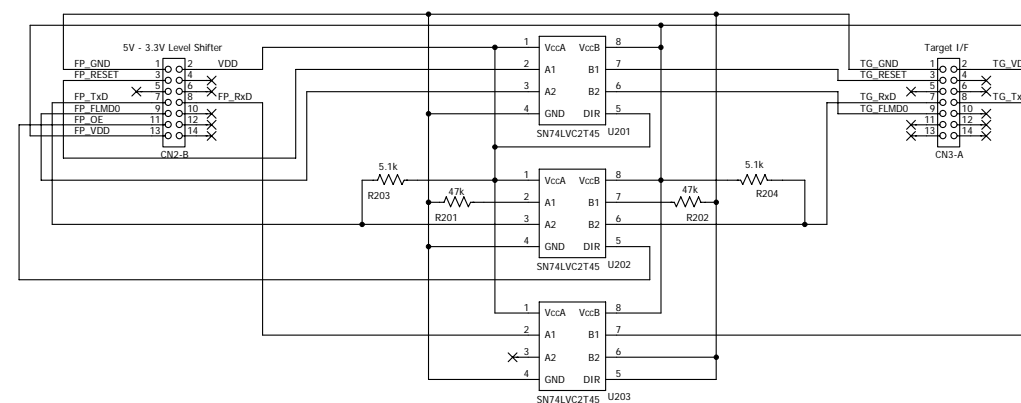
図A - 2 プログラマと78K0/Kx2-Lの参考回路図（メイン・ボード）（3.3 V双方向レベル・シフタを使用する場合）

78K0/Kx2-L Flash Programmer Sample Application Main Board for Monowire UART I/F

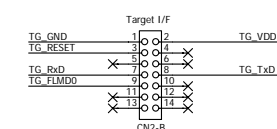
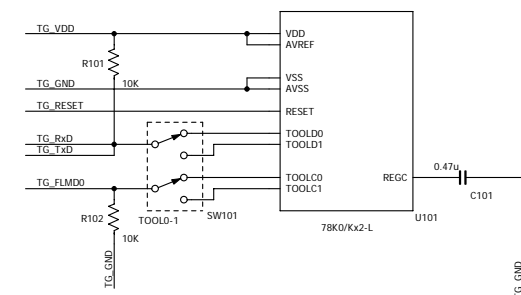
(VDD = 5.0V)



5V - 3.3V Interactive Level Shifter



78K0/Kx2-L Target Board for Monowire UART I/F



備考 回路図中の未使用端子に関しては、各製品のユーザーズ・マニュアルに従い、端子処理を行ってください。

〔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

【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/>

【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。

—— お問い合わせ先 ——

【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