KEYENCE

084659

プログラマブルコントローラ
KV-8000/7000/5000/3000/1000シリーズ
KV Nanoシリーズ

スクリプトプログラミングマニュアル

お使いになる前に、このマニュアルをお読みください。
いつでも使用できるように大切に保管してください。

1章 概 要
2章 スクリプトを作成する
3章 データの取り扱い
4章 スクリプトの文法と注意
5章 基本関数
6章 応用関数
7章 演算関数
8章 拡張関数
・コントロールメモリ、コントロールリレー一覧
・索引

対応 CPU ユニット
・KV-8000
・KV-7500
・KV-7300
・KV-5500
・KV-5000
・KV-3000
・KV-1000
・KV Nanoシリーズ

本社技術スタッフ直営！
FAQ よくあるご質問
www.keyence.co.jp/seigyofaq

はじめに

本書は、《KV-8000/7500/7300/5500/5000/3000/1000》、《KV Nanoシリーズ》のスクリプトプログラムを作成する上で必要となる機能の仕様や設定方法について説明しています。このマニュアルをよくお読みいただき、十分に理解した上でプログラムを作成してください。
プログラムされる際には、『KV-8000シリーズ ユーザーズマニュアル』、『KV-7000シリーズ ユーザーズマニュアル』、『KV-5500/5000/3000シリーズ ユーザーズマニュアル』、『KV Nanoシリーズ ユーザーズマニュアル』、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』『KV STUDIOユーザーズマニュアル』を、CPU内蔵機能に関する命令語や拡張ユニット専用命令語については、各ユニットのユーザーズマニュアルをご覧ください。
本書はいつでもご利用いただけますよう、大切に保管してください。
本書が最終的にご利用いただく方のお手元に届くよう、ご配慮ください。

■KV-8000/7000/5000/3000/1000シリーズ、KV Nanoシリーズ関連マニュアル

以下の全てのPDFマニュアルは、KV STUDIOのヘルプから開けます。また、最新のPDFマニュアルは弊社のホームページよりダウンロード可能です。
印刷版マニュアルのご請求は、本書裏表紙のお問い合わせ電話番号にてご用命ください。

種類	内容	製本
KV-8000シリーズ ユーザーズマニュアル	《KV-8000 シリーズ》の接続・保守方法、CPU 内蔵機能、ラダープログラミングを作成する方法を説明しています。	あり（無償）
KV-7000シリーズ ユーザーズマニュアル	《KV-7000 シリーズ》の接続・保守方法、CPU 内蔵機能、ラダープログラミングを作成する方法を説明しています。	あり（無償）
KV-5500/5000/3000シリーズ ユーザーズマニュアル	《KV-5500/5000/3000 シリーズ》のシステム構成・仕様、CPU内蔵機能、ラダープログラムを作成する方法を説明しています。	あり（無償）
KV Nanoシリーズ 基本ユニット（端子台タイプ） ユーザーズマニュアル	端子台タイプの《KV Nanoシリーズ》のシステム構成・仕様、基本ユニット内蔵機能、ラダープログラムを作成する方法を説明しています。	あり（無償）
KV Nano シリーズ 基本ユニット（コネクタタイプ） ユーザーズマニュアル	コネクタタイプの《KV Nano シリーズ》のシステム構成・仕様、CPU 内蔵機能、ラダープログラムを作成する方法を説明しています。	あり（無償）
KV-8000/7000/5000/3000/1000シリーズ、 KV Nanoシリーズ 命令語リファレンスマニュアル	ラダープログラミングで使用可能な各命令語を説明しています。	なし（PDFのみ）
KV-8000/7000/5000/3000/1000シリーズ、 KV Nanoシリーズ スクリプトプログラミングマニュアル	本マニュアルです。 スクリプトプログラムを作成する方法と使用可能な演算子・制御文・関数を説明しています。	なし（PDFのみ）
KV STUDIO ユーザーズマニュアル	《KV STUDIO》の操作方法を説明しています。	なし（PDFのみ）
KV-5500/5000/3000/1000シリーズ システムマクロマニュアル	《KV-5500/5000/3000/1000シリーズ》で使用可能なシステムマクロを説明しています。	なし（PDFのみ）
KV-1000シリーズ 導入マニュアル	《KV-1000》をはじめてお使いになる方のために、基本的な機能や操作方法をやさしく解説したマニュアルです。	なし（PDFのみ）
KV-1000シリーズ ユーザーズマニュアル	《KV-1000》の仕様、内蔵機能、操作方法について説明しています。	あり（無償）
KV-1000シリーズ プログラミングマニュアル	《KV-1000》を使用してラダープログラムを作成する方法やモジュール・マクロとローカルデバイスについて説明しています。	なし（PDFのみ）

安全にお使いいただくために

■記号の見方

本書では、人への危害や機器への損害を未然に防ぐため、守っていただきたい事項を下記のように表示区分しています。

⚠ 危 険	ここに記載されている記載事項を遵守しない場合、結果的に死亡又は重傷を引き起こす危害が発生します。
⚠ 警 告	ここに記載されている記載事項を遵守しない場合、結果的に死亡又は重傷を引き起こす危害が発生する可能性があります。
⚠ 注 意	ここに記載されている記載事項を遵守しない場合、中程度の傷害又は軽傷を引き起こす危害が発生する可能性があります。
注 記	ここに記載されている記載事項を遵守しない場合、商品自体の損害 (自損) のみならず、他の財物に対する損傷を引き起こす可能性があります。

▶ 重 要　かならずおこなう操作などについての注意を示しています。

ポイント　誤りやすい操作などについての注意を示しています。

参 考　本文の理解を深める事項や、知っておくと役に立つ情報を示しています。

📖　参照すべきページや別のマニュアルの参照ページを示しています。

■一般的な注意事項

⚠ 危 険	• 人体および、人体の一部を保護する目的でこの製品を使用してはいけません。 • この製品は、防爆エリアで使用することを想定していないので、防爆エリアでは決して使用しないでください。 • この商品の動作状態によって人および財物に大きな危害が及ぶ可能性のある用途（発電所、航空、鉄道、船舶、車両、医療機器、娯楽遊具等）では使用しないでください。
⚠ 警 告	• 《KV STUDIO》の使用時には、《KV-8000/7500/7300/5500/5000/3000/1000》、《KV Nano シリーズ》の機能および性能が正常に動作していることを確認してからご使用ください。 • 始業または操作時には、当社製品の機能および性能が正常に動作していることを確認してからご使用ください。 • プログラマブルコントローラ本体に異常が発生した場合でも、フェールセーフの点から、システム全体が安全側に働くように、プログラマブルコントローラを介さない方法で安全回路を設けてください。 • 出力回路や内部回路の故障により、正常に制御動作できなくなる場合が考えられます。火災など、重大な事故の要因となる制御については、安全回路を必ず設けてください。 • この商品を設置する際は、装置全体のリスクアセスメントを実施し、適切なリスク低減がなされていることを確認してください。異常発生時でも装置全体が安全に動作するよう、この商品を介さずに適切な保護手段を設けてください。
⚠ 注 意	この取扱説明書に規定された方法以外でこの商品を使用すると、商品が備えている保護が損なわれる可能性があります。
注 記	• 仕様に示された規格以外での使用、または改造された製品については、機能および性能の保証はできかねますのでご留意ください。 • 当社製品を他の機器と組み合わせてご使用になる場合、使用条件、環境などにより、機能および性能が満足できない場合がありますので、十分ご検討のうえご使用ください。

KV-8000　CPU機能バージョンについて

KV-8000 シリーズは、CPU 機能バージョンにより使用できる機能が異なります。
CPU 機能バージョンは、KV STUDIO のメニューで【ツール】→【CPU 機能バージョン確認】→【PLC】から確認できます。（※ KV STUDIO Ver.10.0 以降が必要です）
また、CPU 機能バージョンは、KV STUDIO のメニューで【ツール】→【CPU 機能バージョン確認】→【PLC】→〔システムプログラム更新〕から更新できます。（※KV STUDIO Ver.10.0 以降が必要です。）

■CPU機能バージョンによるCPU機能差分

公開		2021/09	2021/07	2021/04	2020/04	2019/04
CPU機能バージョン		2.3	2.2	2.1	2.0	1.0
KV STUDIOの対応バージョン		Ver.11.3以降	Ver.11.2以降	Ver.11.1以降	Ver.11.0以降	Ver.10.0以降
CPU機能	デバイスコメント	最大128文字	最大128文字	最大128文字	最大32文字	最大32文字
	変数	○	○	○	○※1	―
	配列	ファンクションブロック型に対応	STRING型に対応	―	―	―
	構造体	○	○	○	○	―
	ファンクションブロック	データ型の自動変換に対応	○	○	○（EN/ENOに対応、インスタンスに対応）	○
	ST言語	○	○	○	―	―
関数・スクリプト	ファンクションブロック関数	○	○	○	2関数追加（FBCALL, FBSTRT）	追加なし
	データサイズ取得関数	○	○	○	2関数追加（WSIZE, ACNT）	―

※1　CPU機能バージョン1.0で対応していたラベルに代わり、CPU機能バージョン2.0以降ではVT5シリーズとの通信でも使用できる変数を使用できます。

KV-7500/7300　CPU機能バージョンについて

KV-7000シリーズは、CPU機能バージョンにより使用できる機能が異なります。
CPU機能バージョンは、KV STUDIOのメニューで【ツール】→【CPU 機能バージョン確認】→【PLC】から確認できます（※KV STUDIO Ver.8 以降が必要です）。
また、CPU 機能バージョンは、KV STUDIO のメニューで【ツール】→【CPU 機能バージョン確認】→【PLC】→〔システムプログラム更新〕から更新できます。（※KV STUDIO Ver.8.1 以降が必要です。）

■CPU機能バージョンによるCPU機能差分

<table>
<tr><th colspan="2">公開</th><th>2018/12</th><th>2017/09</th><th>2017/06</th><th>2016/06</th><th>2015/12</th><th>2015/06</th><th>2015/03</th></tr>
<tr><td colspan="2">CPU機能バージョン</td><td>2.3</td><td>2.3</td><td>2.2</td><td>2.1</td><td>2.0</td><td>1.1</td><td>1.0</td></tr>
<tr><td colspan="2">KV STUDIOの対応バージョン</td><td>9.45以降</td><td>9.3以降</td><td>9.2以降</td><td>9.1以降</td><td>9.0以降</td><td>8.1以降</td><td>8.0以降</td></tr>
<tr><td rowspan="2">対応CPUユニット</td><td>KV-7500</td><td>○</td><td>○</td><td>○</td><td>○</td><td>○</td><td>○</td><td>○</td></tr>
<tr><td>KV-7300</td><td>○</td><td>○</td><td>○</td><td>○</td><td>○</td><td>○</td><td>○</td></tr>
<tr><td rowspan="2">対応拡張ユニット</td><td>KV-XLE02
KV-XL402
KV-XL202</td><td colspan="3">○</td><td colspan="4">×</td></tr>
<tr><td>KV-XH16ML
KV-XH04ML</td><td colspan="5">○</td><td colspan="2">×</td></tr>
<tr><td rowspan="10">CPU機能</td><td>プロジェクト名</td><td colspan="5">最大64文字</td><td colspan="2">最大32文字</td></tr>
<tr><td>プログラム名</td><td colspan="5">最大64文字</td><td colspan="2">最大24文字</td></tr>
<tr><td>プログラムコメント</td><td colspan="5">最大256文字</td><td colspan="2">最大32文字</td></tr>
<tr><td>プログラム実行停止時OFF処理設定</td><td colspan="5">○</td><td colspan="2">×</td></tr>
<tr><td>スクリプト文字列用ワーク領域の割り込みプログラム用割り当て</td><td colspan="3">○</td><td colspan="4">×</td></tr>
<tr><td>ユニット間同期</td><td colspan="6">○</td><td>×</td></tr>
<tr><td>ユニット間同期CPUデバイス読み書き</td><td colspan="5">○</td><td colspan="2">×</td></tr>
<tr><td>KVソケット通信</td><td colspan="6">○※1</td><td>×</td></tr>
<tr><td>ファンクションブロック機能</td><td colspan="4">○（ユニット引数の種別指定なしに対応）</td><td>○</td><td colspan="2">×</td></tr>
<tr><td>アプリケーションパッケージ</td><td colspan="4">○</td><td>×</td><td colspan="2">×</td></tr>
<tr><td rowspan="6">関数・スクリプト</td><td>ファンクションブロック関数</td><td colspan="5">3関数追加（FB,FEND,FUN）</td><td colspan="2">追加なし</td></tr>
<tr><td>ユニットプログラム関数</td><td colspan="5">5関数追加
（UMALLOC,UPSTRT,UPSTOP,UFSUS,UFRSM）</td><td colspan="2">追加なし</td></tr>
<tr><td>浮動小数関数</td><td colspan="5">1関数追加（ATAN2）</td><td colspan="2">追加なし</td></tr>
<tr><td>データ制御関数</td><td colspan="5">1関数仕様拡張（APR）※2</td><td colspan="2">追加なし</td></tr>
<tr><td>データ処理関数</td><td colspan="5">4関数仕様拡張（MAX,MIN,AVG,WSUM）※3</td><td colspan="2">追加なし</td></tr>
<tr><td>ユニット種別判定関数</td><td colspan="4">○</td><td colspan="3">×</td></tr>
</table>

※1　KVソケット通信は、KV-7500でのみ使用できます。
※2　折れ線近似関数（APR）に倍精度浮動小数点型実数を指定できるようになりました。（スケーリング）
※3　最大値検索関数（MAX）、最小値検索関数（MIN）、アベレージ関数（AVG）、ワードサム関数（WSUM）に単精度浮動小数点型実数・倍精度浮動小数点型実数を指定できるようになりました。

KV Nanoシリーズ　CPU機能バージョンについて

KV Nanoシリーズは、CPU機能バージョンにより使用できる機能が異なります。
CPU機能バージョンは、KV STUDIOのメニューで【ツール】→【CPU機能バージョン確認】→【PLC】から確認できます（※KV STUDIO Ver.7以降が必要です）。
また、CPUバージョンはKV STUDIOのメニューで【ツール】→【CPU機能バージョン確認】→【システムプログラム更新】から更新できます。（※KV STUDIO Ver.7.1以降が必要です。）

■CPU機能バージョンによるCPU機能差分

KV-N14□/N24□/N40□/N60□

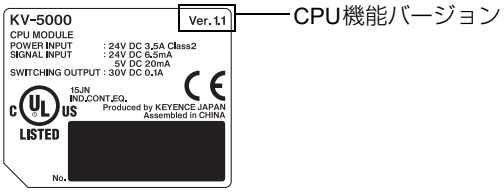
CPU機能バージョン	公開日	対応機能
Ver.1.0	2012年10月17日	初期バージョン
Ver.2.0	2013年5月15日	EtherNet/IPユニット　KV-NC1EPに対応

KV-NC32T

CPU機能バージョン	公開日	対応機能
Ver.2.0	2013年5月15日	初期バージョン

KV-5500/5000/3000　CPU機能バージョンについて

2009年9月10日以降に出荷したKV-5500/5000/3000シリーズには、CPU機能バージョンが追加されています。
CPU機能バージョンは、側面のシリアルシールを確認してください。



KV-5500/5000/3000 CPU機能バージョン差分

CPU機能バージョンによる差分は以下になります。

■CPU機能バージョンによるCPU機能差分

出荷時期		2010/06/14以降	2009/09/10以降	2007/06/14以降
CPU機能バージョン		Ver.2.0	Ver.1.1	Ver.1.0
KV STUDIOの対応バージョン		Ver.6.0以降	Ver.5.6以降	Ver.4.0以降
対応CPUユニット	KV-5500	○	−	−
	KV-5000	○	○	○
	KV-3000	○	○	○
CPU機能	Bluetooth通信	○	○※1	×
	バッテリーレス運転	○	×	×
	初期画面表示(アクセスウィンドウ)	○	×	×
	読み出しフォルダ指定(オートローディング機能)	○	×	×
	SDHCカード(メモリカード)	○	×	×
	ワードデバイスのビットコメント(KV STUDIO)	○	×	×
	パフォーマンスモニタ(KV STUDIO)	○	×	×
	微分モニタ(KV STUDIO)	○	×	×
関数・スクリプト	倍精度浮動小数点型実数	○	×	×
	タイマ･カウンタ関数	1関数追加(TMU)	追加なし	−
	PID制御関数	1関数追加(PIDAT)	追加なし	−
	データ制御関数	3関数追加(RAMP,TPOUT,LLFLT)	追加なし	−
	転送関数	2関数追加(BYLMOV,BYBMOV) 1関数仕様拡張 ※2	追加なし	−
	算術/比較関数	1関数追加(POW)	追加なし	−
	データシフト関数	2関数追加(ASRA,ASLA)	追加なし	−
	データ変換関数	4関数追加(ABS,BSWAP,CPMSET,CPMGET) 2関数仕様拡張 ※3	追加なし	−
	浮動小数関数	5関数追加(LOG10,DFLOAT,DINTG,FTODF,DFTOF)	追加なし	−
	シンプル演算関数	2関数仕様拡張 ※4	追加なし	−
	文字列処理関数	5関数追加(SPLIT,STRIM,SFINDN,CPSASC,RCPSASC) ※5 1関数仕様拡張 ※6	追加なし	−
	データ処理関数	2関数追加(SORT,SORTN)	追加なし	−
	メモリカード関数	7関数追加(MPRINT,MREADL,MCOPY,MMOV,MREN,MFREEK,MSTAT)	追加なし	−
	センサ設定関数	3関数追加(SPRD,SPWR,SSVC)	追加なし	−
	リフレッシュ関数	2関数追加(RFSCI,RFSCO)	追加なし	−

※1　KV-3000 Ver.1.1は、Bluetooth通信に対応していません。
※2　PMOVのオペランドSを定数指定可能としました。
※3　DECO/ENCOのオペランドnをデバイス指定可能としました。
※4　シフト関数(CAL>>, CAL<<) に符号付き整数を指定可能としました。(算術シフト)
※5　SPLITはスクリプト関数のみの追加。
※6　RCOMをワードデバイスのビットコメントに対応しました。

■CPU機能バージョンによるイーサネット機能差分

出荷時期		2010/06/14以降	2009/09/10以降	2007/06/14以降
対応ユニット	KV-5500	Ver.2.0	–	–
	KV-5000	Ver.2.0	Ver.1.1	Ver.1.0
	EtherNet/IPユニット	KV-EP21V	–	–
	イーサネットユニット	KV-LE21V	KV-LE21V	KV-LE20V
イーサネット機能	簡易PLCリンク	○	○	×
	FTPクライアント	○	○	×
	BOOTPクライアント	○	○	×
	ユニット設定レス通信	○	×	×
	APPEコマンド対応 (FTPサーバ機能)	○	×	×

●拡張入出力ユニット (KV-B8XTD 、KV-C16XTD、KV-C32XTD) 使用時の制限事項

CPU機能バージョン1.0のCPUユニットに、拡張入出力ユニットを接続して使用する場合、以下の制限事項があります。(CPU機能バージョン1.1以降のCPUユニットには制限事項はありません。)

- I/O テストにおいて、KV-C16XTD は 32DO、KV-C32XTD は 64DO と表示されます。
- I/O テストにおいて、KV-B8XTD はCPUユニットに格納されているユニット設定情報により表示内容が異なります。
  CPU ユニットに格納されているユニット情報が実際のユニット構成に等しい場合。
  32DO と表示されます。
  CPU ユニットに格納されているユニット情報が実際のユニット構成と異なる場合。
  16DO と表示されます。
- CPU ユニットに格納されているユニット情報が実際のユニット構成と異なる場合、KV-B8XTD は右隣のユニットの種類により I/O テストの動作が異なります。
  右隣が特殊ユニット、または右隣にユニットがない場合
  KV-B8XTD の出力リレーを制御できません。
  右隣が出力ユニットの場合
  右隣のユニットの出力リレーを ON すると、KV-B8XTD の出力リレーも一緒に ON します。
  右隣が入力ユニットの場合
  右隣のユニットの入力リレーが ON すると、KV-B8XTD の出力リレーも一緒に ON します。
- KV-B8XTD、KV-C16XTD、KV-C32XTD は、入力リフレッシュ禁止設定が無効です。
  また、出力禁止を設定した場合は、入力リフレッシュも禁止された状態になります。
- KV-B8XTD、KV-C16XTD、KV-C32XTD に割り当てられた入力リレーの強制セットを解除したとき、対象の入力リレーが 1 スキャン OFF した後、入力状態が反映されます。

マニュアルの構成

1章	概要	スクリプトの利用例(記述例)やKVスクリプトの種別、演算子や制御文、関数などKVスクリプトの機能概要について説明します。
2章	スクリプトを作成する	スクリプトを作成する手順や、変換時に発生するエラーメッセージ･警告メッセージについて説明します。
3章	データの取り扱い	スクリプトで取り扱うデバイスや変数、定数の範囲(型)と記述方法、インデックス修飾と間接指定などについて説明します。
4章	スクリプトの文法と注意	スクリプトで使用できる代入文や演算子の使い方、制御文の文法などについて説明します。
5章	基本関数	接点関数、ビット関数、ビット出力関数、出力関数、タイマ･カウンタ関数などについて説明します。
6章	応用関数	フロー関数、マクロ関数、インデックスレジスタ関数、間接指定関数などについて説明します。
7章	演算関数	データ転送関数、算術関数、論理演算関数、データシフト関数、データ変換関数、浮動小数点関数、文字処理関数などについて説明します。
8章	拡張関数	データ処理関数、テーブル処理関数、時計処理関数、高速処理関数、ログ関数、メモリカード関数、アクセスウィンドウ関数などについて説明します。
	付　録	文字コード表、コントロールリレー･コントロールメモリー覧、索引などを説明します。

目　次

はじめに
安全にお使いいただくために 1
マニュアルの構成 7
目　次 6

1章　概要

KV スクリプトとは 1-2
KVスクリプトの特長 1-2
KVスクリプトの利用方法 1-3
KV スクリプトの種類 1-8
ボックススクリプト 1-8
エリアスクリプト 1-9
KV スクリプトの機能 1-10
代入文について 1-10
演算子について 1-11
制御文について 1-12
サフィックスについて 1-14
スクリプト関数について 1-17

2章　スクリプトを作成する

スクリプトを作成する 2-2
スクリプト記述 2-2
ボックススクリプト 2-4
ボックススクリプトの特長 2-4
ボックススクリプトを作成する 2-6
ボックススクリプトを使用するときの注意 2-13
エリアスクリプト 2-16
エリアスクリプトの特長 2-16
エリアスクリプトを作成する 2-18
変数プログラミングと配列・構造体 2-25
変数プログラミング 2-25
グローバル変数とローカル変数 2-25
変数とデータ型一覧 2-26
配列 2-27
構造体 2-29
変数の設定 2-32
変数使用時の注意事項 2-35
ラベルプログラミングとラベル配列 2-37
ラベルプログラミング 2-37
ラベル配列 2-46
スクリプトを記述するときの注意事項 2-51
ボックススクリプトの並列接続について 2-51
スクリプトで演算式を記述するときの注意事項 2-51

スクリプト変換時のメッセージ.....2-52
警告メッセージ.....2-52
エラーメッセージ.....2-54
ファンクションブロック.....2-58
ファンクションブロックの呼び出し.....2-58
ファンクションブロックをスクリプトで記述する.....2-59
マクロ.....2-60
マクロの呼び出し.....2-60
マクロをスクリプトで記述する.....2-61
スクリプトのデバッグ機能.....2-62
モニタ機能.....2-62
入力補助機能・その他.....2-64
入力補助機能について.....2-64
コメント機能について.....2-69
見やすいスクリプトを書く.....2-70
インラインニモニック.....2-72
編集画面の注意事項とカスタマイズ.....2-73
スクリプト用ワークメモリ(ワークデバイス)について.....2-74
XYM表記.....2-77

3章　データの取り扱い

デバイス.....3-2
デバイス一覧(KV-8000シリーズの場合).....3-2
デバイス一覧(KV-7000シリーズの場合).....3-6
デバイス一覧(KV-5500/5000/3000/1000の場合).....3-10
デバイス一覧(KV Nanoシリーズの場合).....3-17
デバイスの表記方法.....3-21
ビットデバイス.....3-24
ワードデバイス.....3-35
ビットデバイスのワード単位扱い.....3-45
ワードデバイスのビットデバイス扱い.....3-46
ダイレクト入出力リレー.....3-47
変数.....3-49
定数.....3-51
インデックス修飾と間接指定.....3-54
インデックス修飾.....3-55
間接指定.....3-57
インデックス修飾と間接指定を併用する.....3-61
データの型（サフィックス）.....3-62
サフィックス.....3-62
型宣言 (TYPE) について.....3-69
演算の型について.....3-70
PRAGMA 指示命令.....3-72

4 章　スクリプトの文法と注意

代入文 4-2
代入文の使用方法 4-2
演算子 4-6
演算子一覧 4-6
制御文 4-10
制御文一覧 4-10
IF ～ 文 (条件分岐 ①) 4-12
IF ～ ELSE 文 (条件分岐 ②) 4-14
IF ～ ELSEIF ～ 文 (条件分岐 ③) 4-16
SELECT CASE ～ 文 (多分岐制御) 4-18
MC ～ MCR 文 (マスターコントロール) 4-22
FOR ～ NEXT 文 (指定回数の繰り返し制御) 4-24
WHILE 文 (前判定繰返し制御) 4-27
DO ～ UNTIL 文 (後判定繰返し制御) 4-29
BREAK 処理 4-31
制御文を使う時の注意 4-32
スクリプトを記述するときの注意 4-38
制御したビットデータの状態保持 4-38
タイマ / カウンタを使う時の注意 4-40

5 章　基本関数

接点関数 **5-2**
LDP　指定ビットの立ち上がり微分 5-2
LDF　指定ビットの立ち下がり微分 5-4
LDPB　指定ビットの立ち上がり微分の反転 5-6
LDFB　指定ビットの立ち下がり微分の反転 5-7
ビット接点関数 **5-8**
BLD　ワードデータの指定ビット情報 5-8
BLDB　ワードデータの指定ビット反転情報 5-10
出力関数 **5-12**
SET　指定ビットの出力 ON 5-12
RES　指定ビットを出力 OFF 5-13
ビット出力関数 **5-14**
BOUT　ワードデータの指定ビット出力 5-14
BOUB　ワードデータの指定ビット反転出力 5-16
BSET　ワードデータの指定ビット出力 ON 5-18
BRES　ワードデータの指定ビット出力 OFF 5-20
タイマ・カウンタ関数 **5-22**
TMR　100ms32 ビット減算式タイマ 5-22
TMH　10ms32 ビット減算式タイマ 5-26
TMS　1ms32 ビット減算式タイマ 5-28
TMU　10μs32 ビット減算式タイマ 5-30
CNT　32 ビット加算式カウンタ 5-32
ダイレクト入出力関数 **5-36**
RFSX　入力リフレッシュ 5-36
RFSY　出力リフレッシュ 5-37

6 章　応用関数

フロー関数 6-2
CALL　指定サブルーチンの起動 6-2
ECALL　モジュール間サブルーチンコール 6-4
マクロ関数 6-6
MCALL　サブルーチン型マクロの起動 6-6
MSTRT　自己保持型マクロの起動 6-7
MEND　自己保持型マクロの終了 6-8
ファンクションブロック関数 6-10
FB　ファンクションブロックの起動 6-10
FBCALL　ファンクションブロックコール 6-11
FBSTRT　ファンクションブロックスタート 6-13
FEND　ファンクションブロックの終了 6-15
FUN　ファンクションの起動 6-16
モジュール関数 6-18
MDSTRT　モジュール実行開始 6-18
MDSTOP　モジュール実行停止 6-19
インデックスレジスタ関数 6-20
ZPUSH　インデックスレジスタの一括退避 6-20
ZPOP　インデックスレジスタの一括復帰 6-24
間接指定関数 6-26
ADRSET　間接指定対象のアドレス格納 6-26
ADRINC　格納アドレスに 1 を加算 6-28
ADRDEC　格納アドレスから 1 を減算 6-30
ADRADD　格納アドレスを加算 6-32
ADRSUB　格納アドレスを減算 6-34
ファイルレジスタ関数 6-36
FRSET　ファイルレジスタのバンク切り換え 6-36
FRSTM　ファイルレジスタの一括保存 6-38
FRLDM　ファイルレジスタの一括読み出し 6-40
ユニットプログラム関数 6-42
UPSTRT　ユニットプログラム開始 6-42
UPSTOP　ユニットプログラム強制終了 6-44
UFSUS　フロー一時停止 6-46
UFRSM　フロー再開 6-48
UMALLOC　バッファメモリ確保 6-50
バッファメモリ関数 6-52
UREAD　バッファメモリ読み出し 6-52
UWRIT　バッファメモリ書き込み 6-54
UFILL　バッファメモリのデータ列に同じ値を挿入 6-56
ユニット種別判定関数 6-58
IS_ □□□　ユニット種別判定 6-58
配列要素数取得関数 6-60
ACNT　配列要素数取得 6-60
ワードサイズ取得関数 6-62
WSIZE　ワードサイズ取得 6-62

7 章　演算関数

データ転送関数 7-2

BMOV　ワードブロックの一括データ転送 7-2
FMOV　データ列に同じ値を代入 7-4
PMOV　連続ビット情報を指定位置に転送 7-6
BYBMOV　連続バイト列情報を指定位置に転送 (H → L) 7-10
BYLMOV　連続バイト列情報を指定位置に転送 (L → H) 7-12

算術関数 7-14

INC　1 を加算 7-14
DEC　1 を減算 7-16
ROOT　平方根 7-18
SQRT　平方根 7-20

論理演算関数 7-22

ANDA　論理積演算 7-22
ORA　論理和演算 7-24
EORA　排他的論理和演算 7-26
ENRA　排他的論理和否定演算 7-28
COM　ビット反転 7-30
NEG　符号反転 7-32
ABS　絶対値 7-36

データシフト関数 7-38

SRA　右シフト (ビット単位) 7-38
SLA　左シフト (ビット単位) 7-40
ASRA　算術右シフト (ビット単位) 7-42
ASLA　算術左シフト (ビット単位) 7-44
RRA　右ローテート (ビット単位キャリ付) 7-46
RLA　左ローテート (ビット単位キャリ付) 7-48
RRNCA　右ローテート (ビット単位キャリ無) 7-50
RLNCA　左ローテート (ビット単位キャリ無) 7-52
WSR　ワードデバイス右シフト 7-54
WSL　ワードデバイス左シフト 7-56
BSR　ビットデバイス右シフト 7-58
BSL　ビットデバイス左シフト 7-60

データ制御関数 7-62

LIMIT　上下限リミット制御 7-62
BANDC　不感帯制御 7-64
ZONE　ゾーン制御 7-66
APR　折れ線近似 (スケーリング) 7-68
RAMP　傾斜信号 7-74
TPOUT　時分割比例出力 7-76
LLFLT　進み遅れフィルタ 7-78

データ変換関数 7-80
TBCD　BIN データ→ BCD データ変換 7-80
TBIN　BCD データ→ BIN データ変換 7-82
MPX　4 ビットデータデコード 7-84
DMX　4 ビットデータエンコード 7-86
GRY　BIN データ→グレイコード変換 7-88
RGRY　グレイコード→ BIN データ変換 7-90
DISN　ニブル分離 7-92
UNIN　ニブル結合 7-94
DISB　バイト分離 7-96
UNIB　バイト結合 7-98
SWAP　上位・下位バイト入れ替え 7-100
BSWAP　ブロックスワップ 7-102
DECO　8 ビットデコード 7-104
ENCO　8 ビットエンコード 7-106
TOU　データの型を .U に変換 7-108
TOS　データの型を .S に変換 7-108
TOD　データの型を .D に変換 7-108
TOL　データの型を .L に変換 7-108
CPMSET　CIP メッセージ作成 7-110
CPMGET　CIP メッセージ取得 7-112
浮動小数点関数 7-114
FLOAT　BIN データ→単精度浮動小数点型実数データ変換 7-114
TOF　BIN データ→単精度浮動小数点型実数データ変換 7-116
DFLOAT　BIN データ→倍精度浮動小数点実数データ変換 7-118
TODF　BIN データ→倍精度浮動小数点実数データ変換 7-120
INTG　単精度 / 倍精度浮動小数点型実数データ→ BIN データ変換 7-122
INT　単精度浮動小数点型実数データ→ BIN データ変換 7-124
DISF　単精度浮動小数点型実数の仮数・指数分離 7-126
UNIF　仮数・指数の単精度浮動小数点型実数への結合 7-128
EXP　指数関数演算 7-130
LOG　自然対数演算 7-132
LOG10　常用対数演算 7-134
RAD　度 (°) →ラジアン単位変換 7-136
DEG　ラジアン→度 (°) 単位変換 7-138
SIN　角度 (ラジアン) →正弦値 7-140
COS　角度 (ラジアン) →余弦値 7-142
TAN　角度 (ラジアン) →正接値 7-144
ASIN　正弦値→角度 (ラジアン) 7-146
ACOS　余弦値→角度 (ラジアン) 7-148
ATAN　正接値→角度 (ラジアン) 7-150
ATAN2　座標→角度 (ラジアン) 7-152

文字処理関数 7-154
ASC　BIN データ→ 16 進アスキー文字コード変換 7-154
RASC　16 進アスキー文字コード→ BIN データ変換 7-156
DASC　BIN データ→ 10 進アスキー文字列変換.....7-158
STR　数値データ→アスキー文字列変換.....7-162
RDASC　10 進アスキー文字列→ BIN データ変換.....7-166
HASC　BIN データ→ 16 進アスキー文字列変換.....7-168
RHASC　16 進アスキー文字列→ BIN データ変換.....7-172
FASC　単精度浮動小数点型実数→文字列変換.....7-174
RFASC　文字列→単精度浮動小数点型実数変換.....7-178
VAL　文字列→単精度浮動小数点型実数変換.....7-181
LEN　文字列長検出 7-182
SRGHT　右文字列切り出し 7-184
RIGHT　右文字列切り出し 7-186
SLEFT　左文字列切り出し 7-188
LEFT　左文字列切り出し 7-190
SMID　文字列切り出し.....7-192
MID　文字列切り出し.....7-194
SRPLC　文字列部分置換.....7-196
REPLACE　文字列部分置換.....7-199
SINS　文字列挿入.....7-200
INSERT　文字列挿入.....7-202
SDEL　文字列の部分削除 7-204
DELETE　文字列の部分削除 7-206
TRIM　文字列両端の半角スペース・タブ削除.....7-208
LTRIM　文字列左端の半角スペース・タブ削除.....7-210
RTRIM　文字列右端の半角スペース・タブ削除.....7-212
STRIM　文字列指定位置の半角スペース・タブ削除 7-214
SPLIT　文字列分離.....7-216
SFIND　文字列検索.....7-218
SFINDN　文字列検索.....7-220
INSTR　文字列検索.....7-222
FIND　文字列検索.....7-226
DISS　文字列バイト分離 7-228
UNIS　バイト文字列結合 7-230
CHR　16 進アスキー文字コード→文字変換.....7-232
CPSASC　CIP 文字列変換.....7-234
RCPSASC　CIP 文字列逆変換 7-236

8 章　拡張関数

データ処理関数 8-2

SEG　4 ビットデータの 7 セグメントデコード 8-2
BCNT　対象の ON ビットカウント 8-6
DCNT　指定ブロック内の検出データカウント 8-8
SER　指定ブロック内のデータ検出 8-10
DSER　指定範囲内のデータ検索 8-12
MAX　指定ブロック内の最大値検索 8-14
MIN　指定ブロック内の最小値検索 8-18
AVG　指定ブロックの平均値 8-22
WSUM　指定ブロックの合計値 8-26
BSUM　指定ブロックのバイト単位の合計値 8-30
CRC　指定ブロック内の CRC 値算出 8-32
ZRES　指定ブロックのリセット (0 を格納) 8-34
BCMP　指定ブロック間データ比較 8-36
BCMPI　ブロックと値のデータ比較 8-38
RND　乱数発生 8-40
SORT　バイナリデータソート 8-42
SORTN　バイナリデータ分割ソート 8-44

テーブル処理関数 8-48

FIFOW　FIFO テーブルへ書き込み 8-48
FIFOR　FIFO テーブルから読み出し 8-52
LIFOW　LIFO テーブルへ書き込み 8-56
LIFOR　LIFO テーブルから読み出し 8-60
FWRIT　データテーブル指定位置の上書き 8-64
FINS　データテーブル指定位置への挿入 8-66
FDEL　データテーブル指定位置の削除 8-68

時計処理関数 8-70

WTIME　カレンダタイマに時刻データ書き込み 8-70
SEC　日時データを秒形式に変換 8-72
RSEC　秒形式を日時データに変換 8-74
AJST　カレンダタイマ補正 8-76

高速処理関数 8-78

DI　割り込み禁止 8-78
EI　割り込み許可 8-78
RFSCTH　高速カウンタリフレッシュ 8-79
DIC　割り込み禁止範囲の設定 8-80
IEDGE　割り込み条件設定 8-82

PID 関数 8-84

PIDAT　オートチューニング付き PID 制御 8-84

ログ関数 8-86

LOGE　指定 ID のロギング許可 8-86
LOGD　指定 ID のロギング禁止 8-86
TRGD　ロギングデータ取得 8-87
TRGR　デバイス / 変数データ取得 8-88

ストレージ関数.....8-90
MWRIT　ストレージ（メモリカード /CPU メモリ）書き込み.....8-90
MREAD　ストレージ（メモリカード /CPU メモリ）読み出し.....8-92
MFREE　ストレージ（メモリカード /CPU メモリ）の空き容量をバイト単位で取得.....8-94
MFREEK　ストレージ（メモリカード /CPU メモリ）の空き容量をキロバイト単位で取得.....8-96
MMKDIR　ストレージ（メモリカード /CPU メモリ）フォルダ作成.....8-98
MRMDIR　ストレージ（メモリカード /CPU メモリ）フォルダ削除.....8-102
MDEL　ストレージ（メモリカード /CPU メモリ）ファイル削除.....8-104
MPRINT　ストレージ（メモリカード /CPU メモリ）のファイルにテキスト書き込み....8-106
MREADL　ストレージ（メモリカード /CPU メモリ）のファイルから 1 行読み出し....8-108
MCOPY　ストレージ（メモリカード /CPU メモリ）のファイルをコピー.....8-110
MMOV　ストレージ（メモリカード /CPU メモリ）のファイルを移動.....8-112
MREN　ストレージ（メモリカード /CPU メモリ）のファイル名を変更.....8-114
MSTAT　ストレージ（メモリカード /CPU メモリ）のファイル状態取得.....8-116
アクセスウィンドウ関数.....8-118
AWNUM　アクセスウィンドウメッセージ 1 表示.....8-118
AWMSG　アクセスウィンドウメッセージ 2 表示.....8-120
AWSHOW　ユーザメッセージ表示.....8-122
AWHIDE　ユーザメッセージ表示解除.....8-123
フリーランカウンタ関数.....8-124
RFSFRC　フリーランカウンタリフレッシュ.....8-124
センサ設定関数.....8-126
SPRD　センサパラメータ読み出し.....8-126
SPWR　センサパラメータ書き込み.....8-128
SSVC　センササービス実行.....8-130
サイクリック通信リフレッシュ関数.....8-132
RFSCI　サイクリック通信入力リフレッシュ.....8-132
RFSCO　サイクリック通信出力リフレッシュ.....8-134

付　録

CR/CM 一覧.....付 -2
コントロールリレー CR KV-8000の場合.....付-2
コントロールメモリ CM KV-8000の場合.....付-5
コントロールリレー CR KV-7500/7300の場合.....付-10
コントロールメモリ CM KV-7500/7300の場合.....付-13
コントロールリレー CR KV-5500/5000/3000の場合.....付-18
コントロールメモリ CM KV-5500/5000/3000の場合.....付-24
コントロールリレー CR KV Nanoシリーズの場合.....付-31
コントロールメモリ CM KV Nanoシリーズの場合.....付-41
ASCII コード表.....付 -53
使用できない文字一覧.....付 -54
使用できない関数一覧.....付 -58
索　引.....付 -59
関数索引.....付 -64

MEMO

用語について

このマニュアルの説明では一部をのぞき、次のような用語を使っています。

用　語	説　明
プログラマブルコントローラ	プログラムを変更することで自由に機器を制御できる電子装置で、PLC（Programmable Logic Controller）とも呼ばれています。
《KV-8000/7500/7300/5500/5000/3000/1000》	当社製プログラマブルコントローラです。
KV-8000/7000/5000/3000/1000 シリーズ	CPU ユニット《KV-8000/7000/5000/3000/1000》と接続可能な拡張ユニットの総称です。
CPU ユニット	《KV-8000/7500/7300/5500/5000/3000/1000》です。
《KV-N14/N24/N40/N60/NC32T》	当社製プログラマブルコントローラです。
KV Nano シリーズ	基本ユニット《KV-N14/N24/N40/N60/NC32T》と接続可能な拡張ユニットの総称です。
基本ユニット	《KV-N14/N24/N40/N60/NC32T》です。
KV-8000 シリーズ 拡張ユニット	KV-8000 シリーズで使用できる CPU ユニット以外の拡張特殊ユニットです。
KV-8000/7000 シリーズ 拡張ユニット	KV-8000/7000 シリーズで使用できる CPU ユニット以外の拡張入出力ユニット、拡張特殊ユニットです。
KV-5000/3000/1000 シリーズ 拡張ユニット	KV-5500/5000/3000/1000 シリーズで使用できる CPU ユニット以外の拡張入出力ユニット、拡張特殊ユニットです。
KV Nano シリーズ 拡張ユニット	KV Nano シリーズで使用できる基本ユニット以外の拡張 I/O ユニット、拡張特殊ユニットです。
《KV STUDIO》	KV シリーズのプログラム作成をサポートするソフトウェアです。
ラダープログラム	《KV STUDIO》で作成されたプログラムです。
《KV-5000 Ver.1.1》 《KV-3000 Ver.1.1》	2009 年 9 月 10 日以降に発売された KV-5000/3000 で、側面のシリアルシールに、Ver.1.1 と記載されている CPU 機能バージョン 1.1 の KV-5000/3000 です。 CPU 機能バージョン 1.1 より前の KV-5000/3000 と区別する場合に使用します。

記号について

このマニュアルの説明では、メニューやボタンを次のような記号であらわしています。

用　語	説　明
#	「#」を先頭に表記している値は、10 進数を表わしています。また、数値のみの場合（先頭に#がついていない値）も基本的に 10 進数を表わしています。
$	「$」を先頭に表記している値は、16 進数を表わしています。
[DM0・DM1]	32 ビットの数値を扱うときは、デバイスを 2 点使用します。プログラム上では、下位 16 ビットを格納しているデバイス番号（左の例では DM0）だけを記述します。特に気を付けていただきたいときには、左の例のように記述して、32 ビットのデータを扱っていることを表わしています。
↑	「信号の立ち上がり（ON になったとき）」を表わしています。
⎍	「信号が ON 状態になっている間」を表わしています。
↓	「信号の立ち上がり（OFF になったとき）」を表わしています。

1

概要

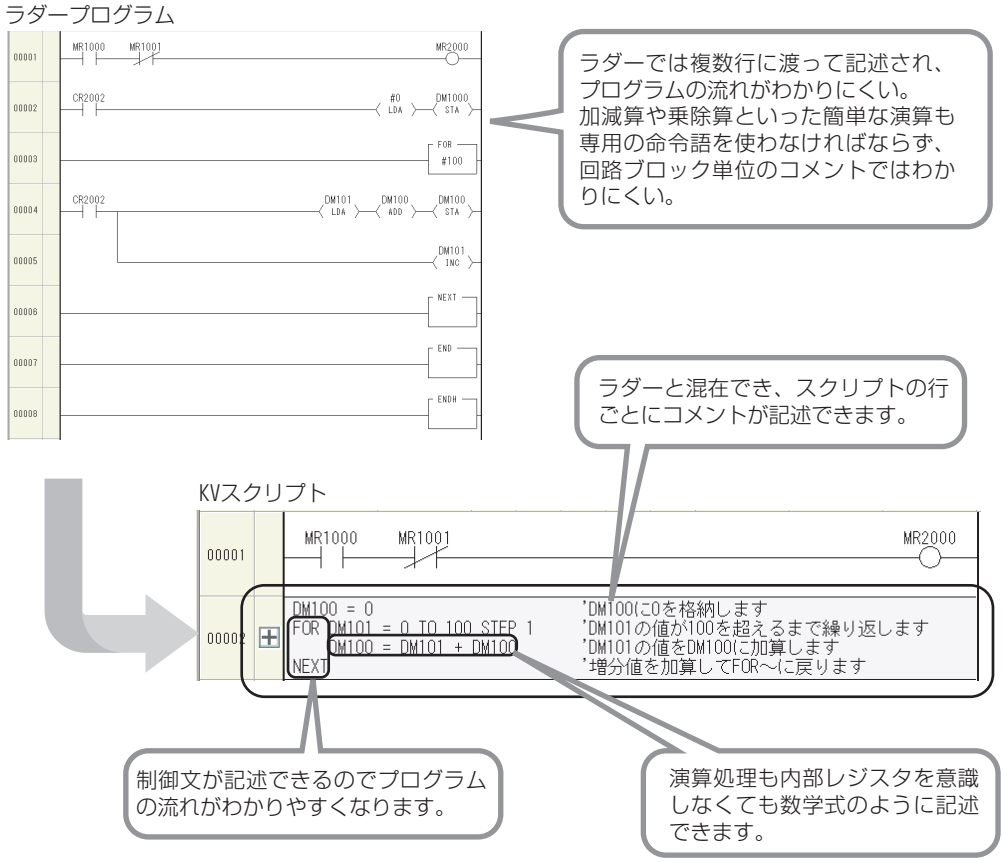
KVスクリプトとは 1-2
KVスクリプトの種類..... 1-8
KVスクリプトの機能..... 1-10

KVスクリプトとは

KVスクリプトとは、従来ラダー言語では記述しにくかった、演算処理や文字列処理などを簡単に記述することを目的に開発された言語です。ラダー言語に混在して記述することが可能ですので、ラダープログラムの作成を中断することなく利用できます。

KVスクリプトの特長

ラダープログラムは、駆動機器の制御やセンサなどを使ったフィードバック制御の記述に適しています。一方、数値を扱う複雑な演算処理や文字列を扱う処理などのプログラムについては、記述が複雑になりデバッグやメンテナンスも困難になりがちでした。
KVスクリプトは、これらラダープログラムの問題点を補うために開発されたプログラミング言語です。ラダープログラムでは記述が複雑になる部分をスクリプトで代替記述して、効率よくプログラムを作成できます。



参考 スクリプトプログラムはラダープログラムに変換されてから実行されます。
スクリプトプログラムの動作確認などは、変換されたラダープログラムでモニタする事ができます。

KVスクリプトの利用方法

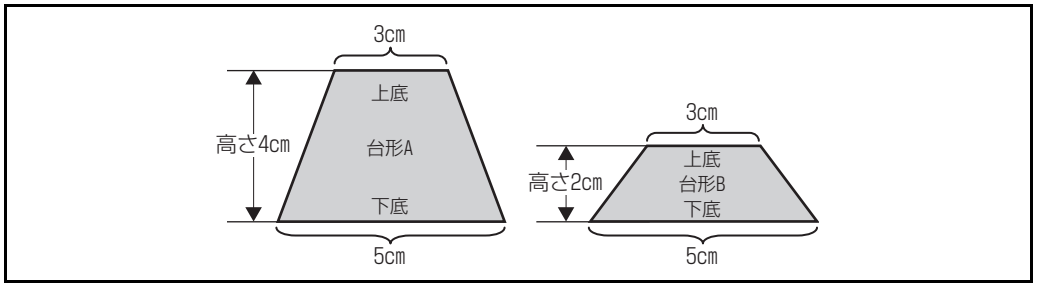
ラダープログラムでは記述が複雑になる演算処理と文字列処理が、スクリプトでどのように記述されるか例をあげて説明します。

■演算処理をKVスクリプトで記述する

ラダープログラムで演算処理を記述するためには、ラダープログラム独特のコツのようなものが必要です。演算式の中に複数の演算子や括弧が含まれる場合、一つ一つ分解して、演算の優先順序にしたがって記述しなければ、正しい結果が得られません。

●演算子を使用する場合

下図のようにそれぞれ高さが違う台形AとBがあります。
台形AとBの面積の和を求め、解をDM50に格納するスクリプトプログラムを記述します。

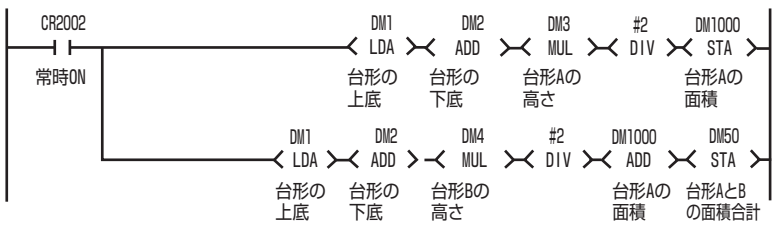


スクリプト記述

```
'------<   ここからコメント   >----------------------------------------------
' 台形の面積を (上底＋下底)×高さ÷2   で求めます。
' DM01 = 3 ：上底の値が格納されています
' DM02 = 5 ：下底の値が格納されています
' DM03 = 4 ：台形Aの高さの値が格納されています
' DM04 = 2 ：台形Bの高さの値が格納されています
' 台形AとBの面積の和を求めます
'------<   ここまでコメント   >----------------------------------------------
DM50 = ((DM01 + DM02) * DM03 / 2) +((DM01 + DM02) * DM04 / 2)
```

スクリプトプログラムでは台形の面積を求める公式を、上記のように一行で記述できました。
次に、同様の処理をラダーで記述してみます。

ラダー記述



ラダープログラムでは、「まず台形Aの面積を一時記憶デバイス(DM1000)に保存し、次に台形Bの面積を求めて台形Aの面積と加算する」という記述をする必要があります。

●演算関数を使用する場合

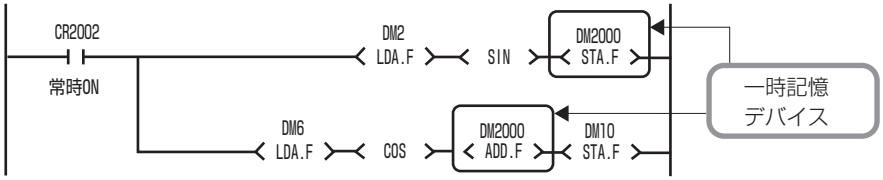
角度の正弦(sin)値と、余弦(cos)値を加算します。
演算の解(加算結果)は　DM10.F　に求めます。

スクリプト記述

```
'------<   ここからコメント   >-----------------------------------------------
' DM02.F  : 正弦値を求める角度の値が格納されています
' DM06.F  : 余弦値を求める角度の値が格納されています
'------<   ここまでコメント   >-----------------------------------------------
DM10.F = SIN(DM02.F) + COS(DM06.F)    '正弦値と余弦値を加算します
```

正弦値や余弦値を求めるSIN関数やCOS関数といったスクリプト関数を使用すると、ラダーで同様の処理をおこなうより、簡単に記述できます。
次に、サンプルプログラムをラダーで記述してみます。

ラダー記述



ラダーで記述すると、SIN関数、COS関数の演算結果を、いったん一時記憶デバイスに格納してから、加算する必要があります。

参考　演算関数は、算術/比較処理や論理演算処理といった演算処理をおこなう関数です。
ビットデータの演算をおこなうANDA関数、ORA関数や、データ転送をおこなうBMOV関数、FMOV関数、また、浮動小数点関数のSIN関数、COS関数などもあります。

■文字列処理をKVスクリプトで記述する

KVスクリプトでは、文字処理(文字列処理)についてもわかりやすく記述できます。

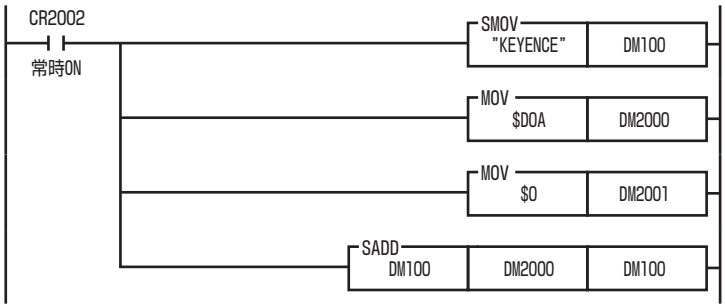
●文字列演算子を使用する場合

DM100を先頭にして、文字列"KEYENCE"にCR(0DH)とLF(0AH)を加えたものを格納します。

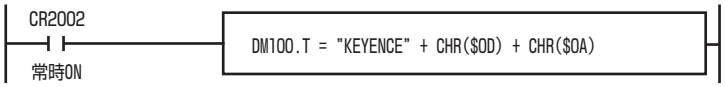
データ格納状態

K(4BH)	E(45H)	: DM100
Y(59H)	E(45H)	: DM101
N(4EH)	C(43H)	: DM102
E(45H)	CR(0DH)	: DM103
LF(0AH)	NUL(00H)	: DM104

ラダー記述



スクリプト記述



ラダー記述で使用するSADD命令は、制御コードを直接扱えないため、一旦デバイスに格納してから任意の文字列と連結させる必要があります。KVスクリプトでは、専用関数を使用することで簡単に文字と制御コードを連結させることができます。

参考

- 文字列とは、その名の通り"文字"の"列"という意味です。文字(1文字)が列をなして並んでいるものを"文字列"と呼びます。文字列の表記は「" "」で囲んでいます。
  詳細は、「固定文字列(文字列型定数)」(3-53ページ)を参照してください。
- DM100に付加するサフィックス.TはKVスクリプトで文字列を格納するデバイスをあらわします。
  詳細は、「データの型(サフィックス)」(3-62ページ)を参照してください。

●文字列処理関数を使用する場合

"Keyence@KV-5000"から"@"を検索し、以降に記述された"KV-5000"を抜き出し DM50.Tに格納するプログラムを、スクリプトで記述すると以下のようになります。

スクリプト記述
```
'------<   ここからコメント   >--------------------------------------------
' DM40.T: 文字列 "Keyence@KV-5000" を格納します
' DM10  : "@" の位置が格納されます (SFINDの検索結果)
' DM20  : DM40.T の文字列長が格納されます (LENの戻り値)
' DM30  : "KV-5000" の文字列長を格納します
'------<   ここまでコメント   >--------------------------------------------
DM40.T = "Keyence@KV-5000"

DM10 = SFIND(DM40.T, "@", 0)      'DM40.T から"@"を検索し位置をDM10に
                                  '取得します
DM20 = LEN(DM40.T)                'DM40.T の文字列長を取得します

DM30 = DM20 – DM10 – 1            ' "KV-5000" の文字列長を取得します

DM50.T = SRGHT(DM40.T, DM30)      ' "KV-5000" を取得します
```

文字列を扱う煩雑な処理も、上記サンプルプログラムのようにスクリプト関数を使用すると、簡潔に記述できます。
また、上記サンプルプログラムは次のように、記述することもできます。

スクリプト記述
```
DM40.T = "Keyence@KV-5000"
DM50.T = SRGHT(DM40.T, LEN(DM40.T) – SFIND(DM40.T, "@", 0) – 1)
```

参考
文字列処理関数は、文字列の加工や変換といった文字列操作や、検索といった文字列データ処理をおこなう関数です。
文字列長を検出するLEN関数や、文字列取り出しをおこなうSMID関数、SLEFT関数などがあります。

文字列関数の詳細については、「7章 演算関数」(7-1ページ)を参照してください。

■制御文をKVスクリプトで記述する

KVスクリプトでは、条件分岐や繰り返し処理を制御文を使うことで簡単に実現できます。

●複数の分岐がある制御をする場合

ワードデータの値により該当するバージョン情報を格納する、といった制御の場合、SELECT文を使って記述すると、以下のように記述できます。

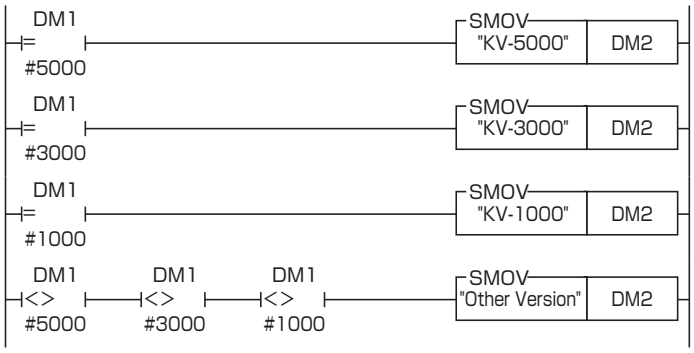
スクリプト記述

```
'------<   ここからコメント   >----------------------------------------------
' DM01 に格納されているバージョン番号により該当する
' バージョン情報を DM02 に格納します
'------<   ここまでコメント   >----------------------------------------------
SELECT CASE DM01                    'バージョン番号で判定をおこないます
CASE 5000                           'バージョン番号が5000の場合
    DM02.T = "KV-5000"
CASE 3000                           'バージョン番号が3000の場合
    DM02.T = "KV-3000"
CASE 1000                           'バージョン番号が1000の場合
    DM02.T = "KV-1000"
CASE ELSE                           '該当するバージョン番号がない場合
    DM02.T = "Other Version"
END SELECT
```

SELECT文の詳細については、「SELECT」CASE ～ 文(多分岐制御)」(4-18ページ)を参照してください。

上記スクリプトをラダープログラムで記述すると以下のようになります。

ラダー記述　DM01に格納されているバージョン番号に該当するバージョン情報をDM02に格納します。



KVスクリプトの種類

KVスクリプトには、記述スタイルの異なる2種類のスクリプトがあります。

ボックススクリプト

ボックススクリプトは、実行条件をラダーで記述できるスクリプトです。
編集画面の記述範囲は、1行の第5セルから第10セルまでの6セル内になります。記述に関しての文字コード、文字数限定等の制約事項がありますので、その範囲で記述してください。

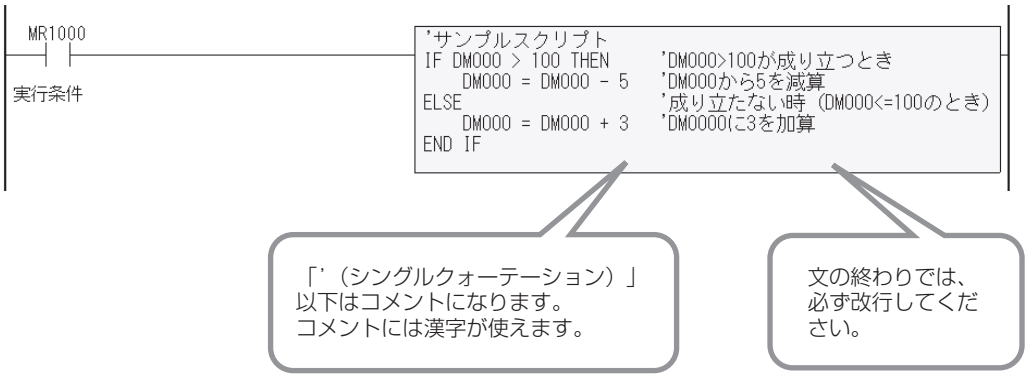
「スクリプトを作成する」(2-2ページ)

スクリプト記述

```
'サンプルスクリプト
IF DM000 > 100 THEN          'DM000>100が成り立つとき
   DM000 = DM000 – 5         'DM000から5を減算
ELSE                         '成り立たない時(DM000<=100のとき)
   DM000 = DM000 + 3         'DM0000に3を加算
END IF
```

そのまま記述できます。

ボックススクリプト記述



「'（シングルクォーテーション）」以下はコメントになります。コメントには漢字が使えます。

文の終わりでは、必ず改行してください。

ポイント **実行条件が満たされないとき、ボックススクリプト内は実行されず、スキップされます(微分型命令など、一部の関数において注意が必要です)。**

エリアスクリプト

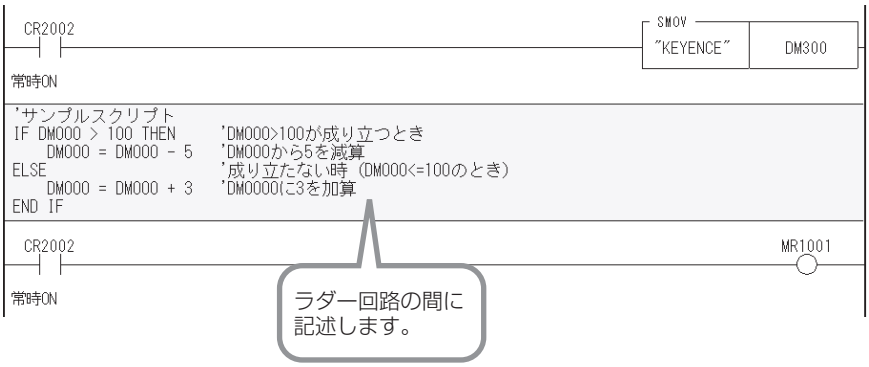
エリアスクリプトは、ラダーの実行条件を持たない、毎スキャン実行型スクリプトです。
編集画面上の記述範囲は、1行の全てを占有します。

スクリプト記述

```
'サンプルスクリプト
IF DM000 > 100 THEN          'DM000>100が成り立つとき
   DM000 = DM000 – 5         'DM000から5を減算
ELSE                         '成り立たない時(DM000<=100のとき)
   DM000 = DM000 + 3         'DM0000に3を加算
END IF
```

そのまま記述できます。

エリアスクリプト記述



KVスクリプトの機能

KVスクリプトは、ラダープログラムで記述される一連の制御を、BASIC言語に近いプログラミング形式で記述できます。

代入文について

KVスクリプトでは、デバイスや変数に値や演算結果を格納する処理を、代入文を使って記述します。

■代入文

DM1000 = DM1001 + 10

"="の左辺に、右辺の結果（値）を格納します。

●数値代入

定数1000をDM1000に格納します。

スクリプト記述　DM1000 = 1000

ラダー記述

●文字列代入

文字列"KEYENCE"をDM1000から格納します（文字列の終わりを示すNUL（00H）は自動的に加えられます）。

スクリプト記述　DM1000.T = "KEYENCE"

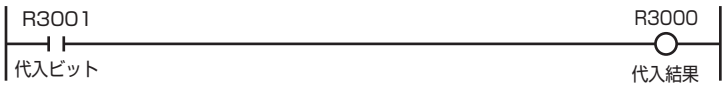
ラダー記述

●ビット代入

ビットデバイスR3001の状態を、R3000に反映します。

スクリプト記述　R3000 = R3001

ラダー記述



代入文の詳細については、「代入文」（4-2ページ）を参照してください。

演算子について

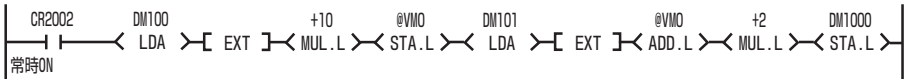
KVスクリプトでは、算術演算や論理演算などをラダープログラムのような専用の命令語に置きかえる必要がなく、演算子をそのまま記述できるので、より直感的なプログラミングが可能です。

●演算例

DM100の値を10倍し、DM101を加えた結果を2倍します。演算結果をDM1000に代入（格納）します。

スクリプト記述　DM1000.L = 2 ∗ (DM100 ∗ 10 + DM101)

ラダー記述



●演算子一覧

記　号		処理内容	記述例
算術演算子	^	累乗を求めます	DM500 = DM100 ^ 2
	∗	2つの値の積を求めます（乗算）	DM502 = DM100 ∗ 3
	/	2つの値の商を求めます（除算）	DM504 = DM100 / 4
	MOD	2つの値の除算を行い、その剰余を返します	DM506 = DM100 MOD 5
	+	2つの値の和を求めます（加算）	DM508 = DM100 + 6
	−	2つの値の差を求めます（減算）	DM510 = DM100 - 7
比較演算子	<	より小さい	MR500 = DM100 < 10
	<=	小さいか等しい（以下）	MR501 = DM100 <= 20
	>	より大きい	MR502 = DM100 > 30
	>=	大きいか等しい（以上）	MR503 = DM100 >= 40
	=	等しい（等価） ※代入の「＝」と同じ	MR504 = DM100 = 50
	<>	等しくない（不等価）	MR505 = DM100 <> 60
文字列演算子	+, &	2つの文字列を連結します	DM600.T = "KEY" + "ENCE"
論理演算子	NOT	値の論理否定を求めます	MR600 = NOT R000
	AND	2つの値（ビットデータ同士、ワードデータ同士）の論理積を求めます（かつ）	MR601 = R000 AND R001
	OR	2つの値（ビットデータ同士、ワードデータ同士）の論理和を求めます（または）	MR602 = R000 OR R001
	XOR	2つの値（ビットデータ同士、ワードデータ同士）の排他的論理和を求めます	MR603 = R000 XOR R001

演算子の詳細については、「演算子」（4-6ページ）を参照してください。

制御文について

KVスクリプトでは、さまざまな制御文を記述できます。制御文を利用することにより、簡単にプログラムの流れを変える(制御)ことができます。
制御文には、条件により処理を分岐させる条件分岐文と、繰り返し処理をおこなわせる繰り返し制御文があります。

■条件分岐制御文

条件によって実行する処理を選択します。

●条件分岐文の種類

制御文	基本書式	説明
IF文	IF␣<条件式>␣THEN ↵ 　　<処理文> ↵ END␣IF ↵	<条件式>の結果により、処理を実行する/しないを決める分岐制御文
	IF␣<条件式>␣THEN ↵ 　　<処理文1> ↵ ELSE␣ ↵ 　　<処理文2> ↵ END␣IF ↵	<条件式>の結果に従って別々の処理を実行する2分岐制御文
	IF␣<条件式1>␣THEN ↵ 　　<処理文1> ↵ ELSE␣IF␣<条件式2>␣THEN ↵ 　　<処理文2> ↵ 　　. 　　. 　　. ELSE ↵ 　　<処理文n> ↵ END␣IF ↵	複数の条件式(<条件式1>、<条件式2>‥<条件式n>)の結果に従って別々の処理を実行する多分岐制御文
SELECT文	SELECT␣CASE␣<ワードデータ> ↵ CASE␣<比較値1> ↵ 　　<処理文1> ↵ CASE␣<比較値2> ↵ 　　<処理文2> ↵ 　　. 　　. 　　. CASE␣ELSE ↵ 　　<処理文n> ↵ END␣SELECT ↵	<ワードデータ>の値に従い、対応する処理を選択して実行する多分岐制御文
MC文	MC␣<MC実行条件>␣THEN ↵ 　　<処理文> ↵ MCR ↵	マスターコントロールの制御文

※ ␣は半角スペースの挿入を表します。
※ ↵マークがあるところは、必ず改行を入れてください。

■繰り返し制御文

指定された条件で処理を繰り返します。

●繰り返し制御文の種類

制御文	基本書式	説明
FOR文	FOR␣<ワードデータ>=<初期値> ␣TO␣<最終値>␣STEP␣<増分値> ↵ <処理文> ↵ NEXT ↵	<ワードデータ>の値が <最終値>に到達するまで処理を繰り返す制御文
WHILE文	WHILE␣<条件式> ↵ <処理文> ↵ END␣WHILE ↵	条件式が成立している(真である)間、(不定回数)処理を繰り返す制御文
DO文	DO ↵ <処理文> ↵ UNTIL␣<条件式> ↵	条件式が成立しない(偽である)間、(不定回数)処理を繰り返す制御文
BREAK文	<繰り返し条件文始まり> ↵ BREAK ↵ <繰り返し条件文終わり> ↵	繰り返し制御文の中で、条件とは関係なく繰り返し制御から抜ける(FOR文、WHILE文、DO文の処理文の中でのみ使用)

※ ␣は半角スペースの挿入を表します。

※ ↵マークがあるところは、必ず改行を入れてください。

制御文の詳細については、「制御文」(4-10ページ)を参照してください。

サフィックスについて

ラダー言語では各命令語にサフィックスを付けることで、命令の動作の型が決まるのに対して、KVスクリプトではデバイスや定数にサフィックスを付けることで、関数の動作の型や、デバイスに格納されている値の意味が決まります。
データの型（サフィックス）の詳細については、「データの型（サフィックス）」（3-62ページ）を参照してください。

ポイント

変数を使用する場合は、以下のとおり動作します。

- **変数にサフィックスを付ける場合　・・・サフィックスで指定したデータ型で動作します**
- **変数にサフィックスを付けない場合 ・・・変数編集ウィンドウで設定したデータ型で動作します**

「変数」（3-49ページ）

●サフィックスの種類

サフィックス	型	扱う範囲
.U	16ビット符号無しデータ型	0～65535
.S	16ビット符号付きデータ型	－32768～＋32767
.D	32ビット符号無しデータ型	0～4294967295
.L	32ビット符号付きデータ型	－2147483648～+2147483647
.F	単精度浮動小数点型	－3.4E+38≦N≦－1.4E－45 N＝0 1.4E－45≦N≦3.4E+38 （有効桁数：約7桁）
.DF※	倍精度浮動小数点型	－1.79E+308 ≦N≦－2.23E－308 N＝0 2.23E－308 ≦N≦1.79E+308 （有効桁数：約16桁）
.B	ビット型（ブール値）	1（ON, TRUE）、0（OFF, FALSE）
.T	文字列型	文字列

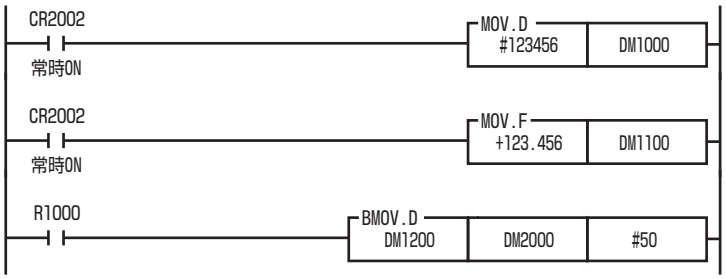
※ ・「.DF」サフィックスは、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。
・「.DF」サフィックスは、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。
「KV-5500/5000/3000　CPU機能バージョンについて」（4ページ）

●演算例

スクリプト記述

```
DM1000.D = 123456
DM1100.F = 123.456
BMOV(R1000, DM1200.D, DM2000, 50)
```

ラダー記述



●サフィックスの省略

デバイスのサフィックスを省略して記述した場合、デバイスの種別により扱われる型が決まります。

デバイスの種類	サフィックスを省略した場合の型(デフォルトの型)
R(DR),B,MR,LR,T,C,CTC※3,CR,UR	.B(ビット型)
DM,W,EM※2,ZF※2,FM※2,TM,CM,V,P,UM,UV, 間接指定(*)するデバイス	.U(16ビット符号無しデータ型)
Z※1	KV-8000 :.L(32ビット符号付きデータ型) KV-7500/7300 :.L(32ビット符号付きデータ型) KV-5500/5000/3000 :.L(32ビット符号付きデータ型) KV-1000 :.S(16ビット符号付きデータ型) KV Nanoシリーズ :.L(32ビット符号付きデータ型)
CTH※3	.D(32ビット符号無しデータ型)

※1　使用するCPUユニットにより、デフォルトの型が異なります。

※2　KV NanoシリーズではEM,FM,ZFは使用できません。

※3　KV-8000/7000シリーズではCTC、CTHは使用できません。

●変数のサフィックス

グローバル変数、ローカル変数のサフィックスを省略して記述した場合、変数登録時に選択する「データ型」により扱われる型が決まります。

変数のデータ型		サフィックスを省略した場合の型(デフォルトの型)
BOOL	ビット	.B
INT	1ワード符号あり整数	.S
DINT	2ワード符号あり整数	.L
UINT	1ワード符号なし整数	.U
UDINT	2ワード符号なし整数	.D
REAL	単精度浮動小数点型実数	.F
LREAL	倍精度浮動小数点型実数	.DF
STRING	文字列	(.T)※1
TIMER	タイマ	.B
COUNTER	カウンタ	.B
ARRAY		変数登録時に指定するデータ型のデフォルトの型
構造体型		構造体定義で設定した各メンバのデータ型のデフォルトの型
ファンクションブロック型		引数設定で設定した各引数のデータ型のデフォルトの型

※1　データ型に「STRING」を設定した変数に、サフィックスは記述できません(常に省略した状態で使用してください)。

●ラベルのサフィックス

グローバルラベル･ローカルラベルのサフィックスを省略して記述した場合、ラベル登録時選択する「データ形式」により扱われる型が決まります。

ラベルのデータ型		サフィックスを省略した場合の型(デフォルトの型)
ビット	(BOOL)	.B
1ワード符号なし整数	(UINT)	.U
2ワード符号なし整数	(UDINT)	.D
1ワード符号付き整数	(INT)	.S
2ワード符号付き整数	(DINT)	.L
単精度浮動小数点型実数	(REAL)	.F
倍精度浮動小数点型実数	(LREAL)	.DF※1
文字列	(STRING)	(.T)※2
タイマ		.B
カウンタ		.B
配列		配列サイズ指定時のデータ形式による

※1 ・「.DF」サフィックスは、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。
・「.DF」サフィックスは、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。
「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

※2 データ型に「文字列」を選択したラベルに、サフィックスは記述できません(常に省略した状態で使用してください)。

●デバイスの型宣言について

スクリプトプログラムの先頭でデバイスの型を宣言することにより、各デバイスのデフォルトの型を指定できます。

宣言文	書　式	説　明
TYPE	TYPE DM1000.D	DM1000はサフィックスを省略しても.D(32ビット符号無し)として扱われます。
	TYPE EM1000 – EM2000.F (TYPE EM1000.F – EM2000)	EM1000～EM2000はサフィックスを省略しても.F(32ビット単精度浮動小数点型)として扱われます。

型宣言の詳細については 「データの型(サフィックス)」(3-62ページ)を参照してください。

ポイント
- **型宣言は、TYPEを記述したスクリプトのみ有効です。**
- **ローカルラベルや配列を型宣言する場合、ラベル登録時に選択したデータ形式より大きな型や文字列型は、宣言できません。**

  例　**データ形式で「1ワード符号なし整数」を選択した場合、32ビット型の宣言(TYPE〔ラベル名〕.Dなど)をするとエラーになります。**
- **グローバルラベルをTYPEで型宣言した場合は、型宣言したサフィックスが有効になります。**

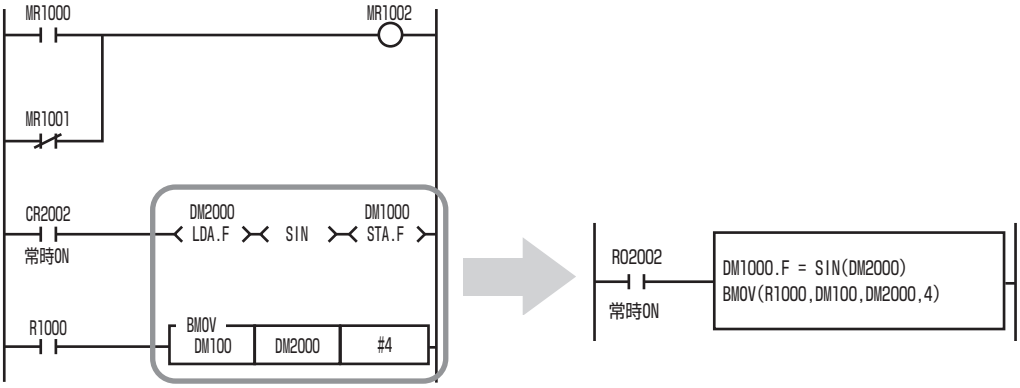
参考　型宣言したデバイスを間接指定で使用する場合、デフォルトの型は.Uとして扱われます。

スクリプト関数について

スクリプト関数は、KV-8000/7500/7300/5500/5000/3000/1000、KV Nanoシリーズのラダー命令語をKVスクリプトから利用できるようにしたものです。
ラダー命令語と区別するため、KVスクリプトで使用する命令語は「スクリプト関数」と呼びます。

■関数

ラダー言語では命令語をシンボル線でつないでプログラムを作成しますが、KVスクリプトは代入文・演算子・制御文やスクリプト関数を使ってプログラムを作成します。
スクリプト関数はラダー命令語のほとんどをカバーしていますが、演算子や代入文が使用できるようになったことで、演算系命令（ADD、MUL等）やデータ転送命令（MOV、DW等）の一部に対応する関数は存在しません。



スクリプトの関数のほとんどは、ラダー命令語と同じ関数名で記述します。
動作はラダー命令語と同じです。

参考　スクリプトでは、内部レジスタを操作しなくてもよいので、LDAやSTA命令は存在しません。
ラダー命令語にはない、KVスクリプトで追加になっている関数もあります。

●スクリプト関数の種類

スクリプト関数は、大きく分けて以下の種類に分けられます。

種　類	説　明
演算処理関数	ワードデータやビットデータの演算をおこなう関数群です。論理演算をおこなうANDA・ORAや、データ転送をおこなう BMOV・FMOV、浮動小数点型演算をおこなう SIN・COSなどがあります。
文字列処理関数	文字列処理をおこなうための関数群です。文字列長を検出するLENや、文字列取り出しをおこなうSMID・SLEFTなどがあります。
データ処理関数	データ処理をおこなうための関数群です。データの最大値を求めるMAXや、最小値を求めるMIN、指定ワードデータ内のONしているビット数を数えるBCNTなどがあります。
その他の関数	演算、文字列、データ処理以外の処理をおこなう関数群です。タイマを操作するTMRや、間接指定をおこなうADRSET、メモリカード制御をおこなうMWRITやMREADなどの関数があります。

■演算フラグについて（CR2009～CR2012）

各スクリプトはラダープログラムに変換され、関数を実行したとき演算フラグも変化します。そのため演算フラグの状態は、スクリプトプログラムの実行の前後で変化しています。
また、スクリプトを変換して生成されるラダープログラムは、関数の組み合わせなどにより色々な記述ができるため、スクリプトプログラムによって変化した演算フラグを利用することは基本的にできません。

■間接指定やインデックス修飾時の注意事項について

間接指定やインデックス修飾などを使用する場合、範囲外のデバイスやワーク領域を指定しないように注意してください。
もしも各関数の引数に対して、間接参照やインデックス修飾を使用して範囲外のデバイスやワーク領域にアクセスしようとした場合、それぞれ次のような動きになります。

●不正なアドレスを指定したときの各関数の動作一覧

関数の動作	関数名
戻り値に0を格納する	LDP, LDF, BLD, BLDB, LDPB, LDFB, UNIF, RDASC, RHASC, RFASC, VAL, LEN, SFIND, INSTR, FIND, SEG, MAX, MIN, AVG, WSUM, BSUM, BCMP, BCMPI, SEC
戻り値にNUL（00H）を格納する	DASC, STR, HASC, FASC, SRGHT, RIGHT, SLEFT, LEFT, SMID, MID, SRPLC, REPLACE, SINS, INSERT, SDEL, DELETE
戻り値に第1引数の値を格納する※1	ROOT, ANDA, ORA, EORA, ENRA, COM, NEG, SRA, SLA, RRA, RLA, RRNCA, RLNCA, TBCD, TBIN, MPX, DMX, GRY, RGRY, SWAP, DECO, ENCO, TO,USDLF, FLOAT, INTG, INT, EXP, LOG, RAD, DEG, SIN, COS, TAN, ASIN, ACOS, ATAN, ASC, RASC, DASC, CHR, BCNT, DCNT, SER, CRC, WTIME, AWNUM, AWMSG, WTIME, SQRT, ABS, RND
関数を実行しない	SET, RES, BOUT, BOUB, BSET, BRES, MCALL, MSTRT, ZPUSH, ZPOP, ADRSET, ADRINC, ADRDEC, ADRADD, ADRSUB, BMOV, FMOV, PMOV, INC, DEC, WSR, WSL, BSR, BSL, DISF, DISS, UNIS, ZRES, FIFOW, FIFOR, LIFOW, LIFOR, FWRIT, FINS, FDEL, RSEC, DIC, LOGE, LOGD, MWRIT, MREAD, MFREE, MMKDIR, RFSX, RFSY, ECALL, MDSTRT, MDSTOP, FRSET, FRSTM, FRLDM, UREAD, UWRIT, UFILL, LIMIT, BANDC, ZONE, APR, DISN, DISB, UNIN, UNIB, DSER, RFSCTH, TRGD, MRMDIR, MDEL, RFSFRC、ACNT

※1　ただし、第1引数に不正な間接参照をした場合、戻り値の値は保証されません。

2

スクリプトを作成する

スクリプトを作成する 2-2
ボックススクリプト 2-4
エリアスクリプト 2-16
変数プログラミングと配列･構造体 2-25
ラベルプログラミングとラベル配列 2-37
スクリプトを記述するときの注意事項 2-51
スクリプト変換時のメッセージ 2-52
ファンクションブロック 2-58
マクロ 2-60
スクリプトのデバッグ機能 2-62
入力補助機能･その他 2-64

スクリプトを作成する

ボックススクリプト/エリアスクリプトの作成方法、メニューや編集画面について説明します。

スクリプト記述

KVスクリプトは、ラダー編集画面に直接テキスト形式で記述できます。
また、Windowsの標準機能であるカット&ペーストや、コピー、UNDO/REDOにも対応します。

スクリプトの記述方法

ボックススクリプト/エリアスクリプトはラダー編集ウィンドウ上でラダープログラムに混在させた形で記述できます。
KVスクリプトは、ラダー編集ウィンドウ上でスクリプト記述エリアを確保してから、記述します。

①メニューから呼び出す

【スクリプト(S)】→【エリアスクリプト挿入(A)】または【ボックススクリプト挿入(B)】を選択する。

②右クリックメニューから呼び出す

カーソルがラダー編集ウィンドウの上にあるとき、右クリックメニューから【スクリプト編集(X)】を選択し、【エリアスクリプト挿入(A)】または【ボックススクリプト挿入(B)】を選択する。

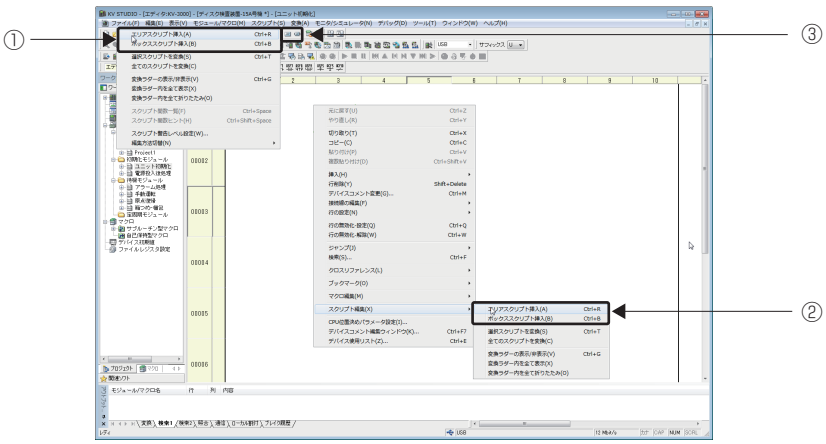
③アイコンから呼び出す

ツールバーのアイコンをクリックします。

- エリアスクリプト：
- ボックススクリプト：

④ショートカットキーで呼び出す

- エリアスクリプト：Ctrl キー＋R キー
- ボックススクリプト：Ctrl キー＋B キー

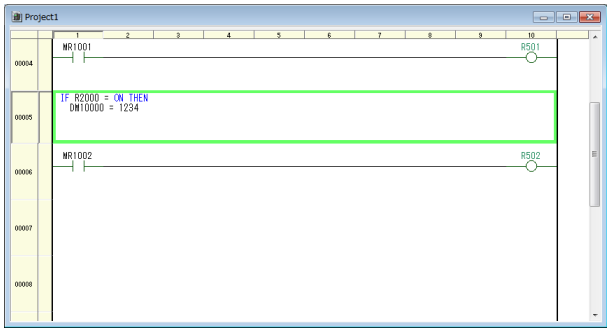


スクリプトの入力方法

スクリプトには2種類の入力方法があります。入力方法の違いによる機能の違いはありませんので、お好みにより選択してください。

■直接入力(デフォルト)

ラダープログラムの命令語と同じように、ラダーエディタ上に直接記述します。

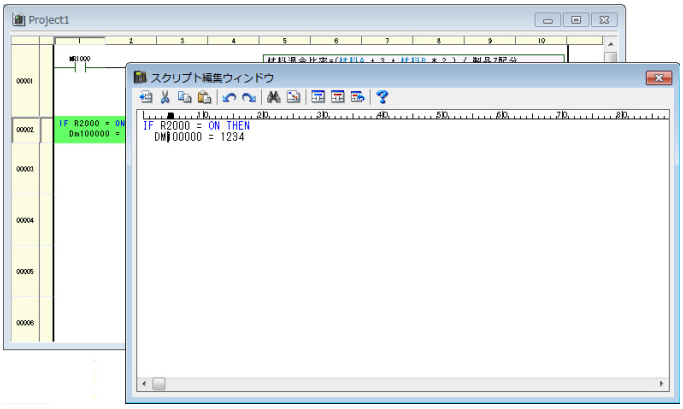


1 **メニューから【スクリプト(S)】→【編集方法切替(N)】→【直接入力(D)】を選択します。**

別手順　メニューから【ツール(T)】→【オプション(O)】、「スクリプト設定」タブの[入力形式]から「直接入力」を選択。

■ウィンドウ入力

スクリプト記述時に表示されるテキストエディタウィンドウに入力します。



1 **メニューから【スクリプト(S)】→【編集方法切替(N)】→【ウィンドウ(W)】を選択します。**

別手順　メニューから【ツール (T)】→【オプション (O)】、「スクリプト設定」タブの [入力形式] から「ウィンドウ」を選択。

ボックススクリプト

ここではボックススクリプトについての作成/編集方法、使用時の注意点について説明します。

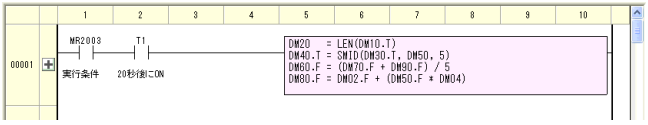
ボックススクリプトの特長

ボックススクリプトは、いつ、どのタイミングで、KVスクリプトで記述されたプログラムを実行するかをラダーで指定できる、ラダー実行条件付きスクリプトです。
ラダーで指定した実行条件が成立している間だけボックススクリプトが実行されるので、演算式の記述などに適しています。

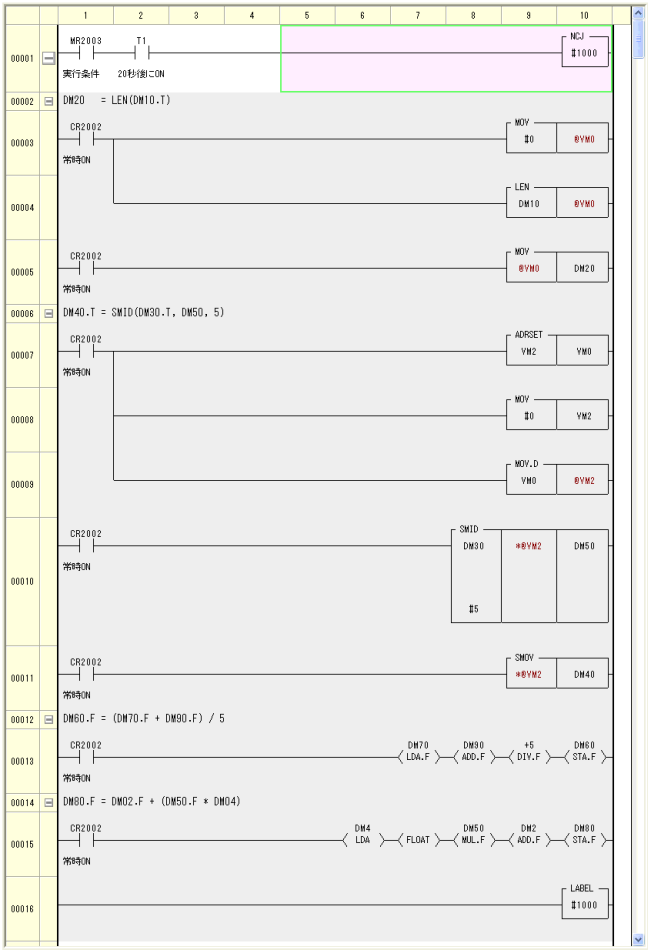
ボックススクリプトはこんなときに使う

ボックススクリプトは数行程度の演算式や複雑な制御文を含まないプログラムの記述に適しています。
また、実行条件をラダーで記述することで、ボックススクリプト内のプログラムは実行処理文だけの記述が可能ですので、制御文を使う必要がなく、簡単に利用できます。

ボックススクリプト



ラダー展開

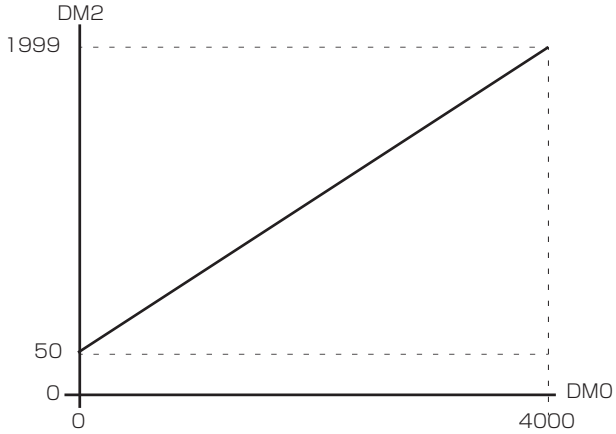


●ボックススクリプトを使用したサンプルプログラム

[スケーリング処理]

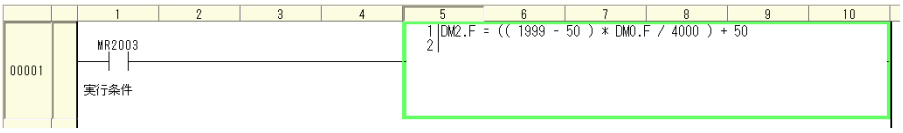
DM0.F([DM1･DM0])に格納した0～4000の値を50～1999の範囲にスケーリングし、DM2.F([DM3･DM2])に格納します。演算は単精度浮動小数点型でおこないます。

- ･DM0＝0 → DM2＝50
- ･DM0＝50 → DM2＝74
- ･DM0＝100 → DM2＝98
- ･DM0＝500 → DM2＝293
- ･DM0＝1000 → DM2＝537
- ･DM0＝2000 → DM2＝1024
- ･DM0＝4000 → DM2＝1999

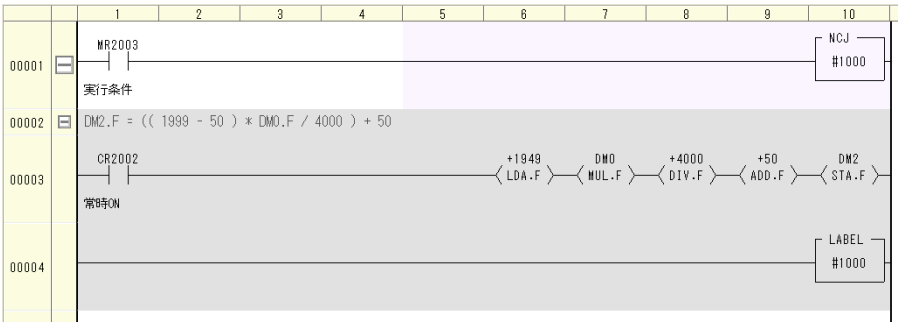


上記の動作をボックススクリプトを使用して記述します。

ボックススクリプト



ラダー展開

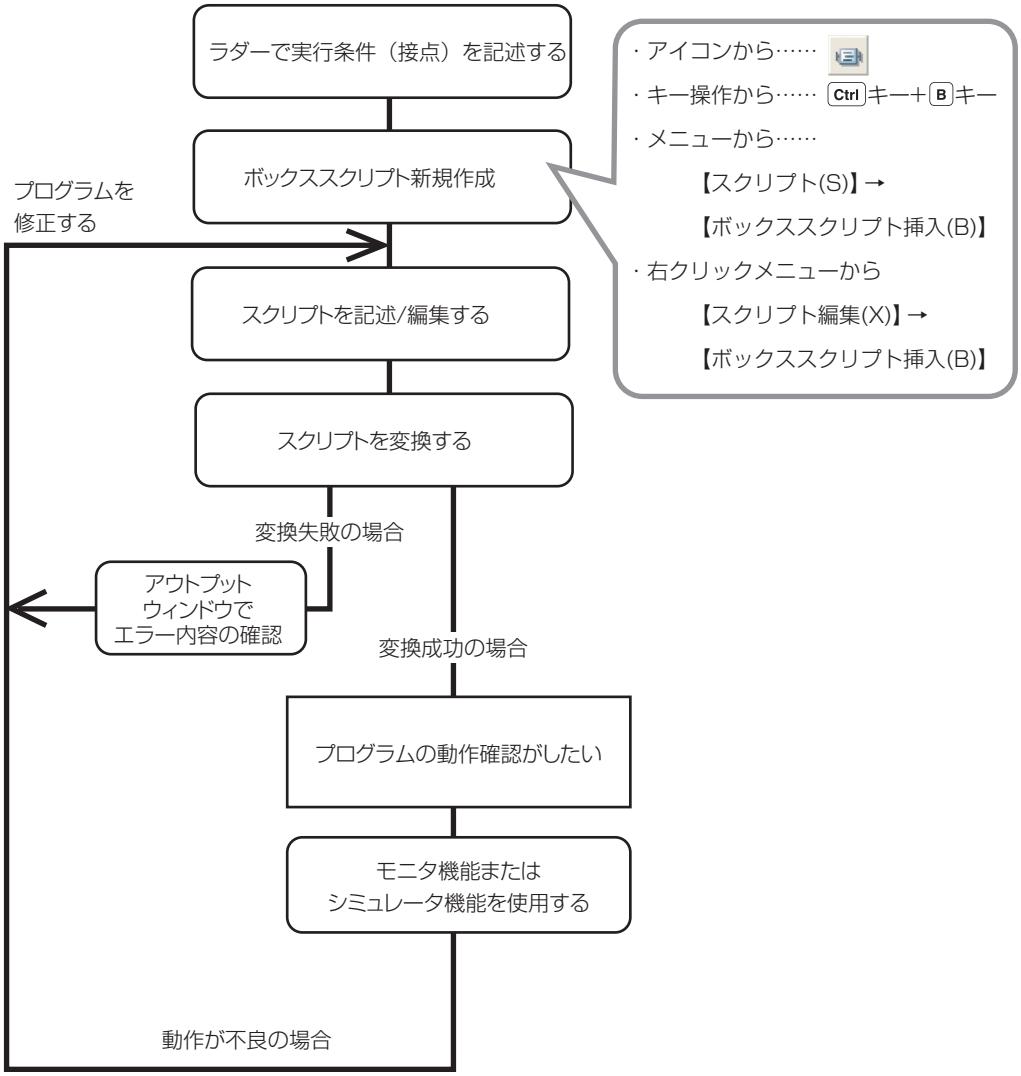


ボックススクリプトを作成する

ボックススクリプトの作成からデバッグまでの流れを説明します。

操作の流れ

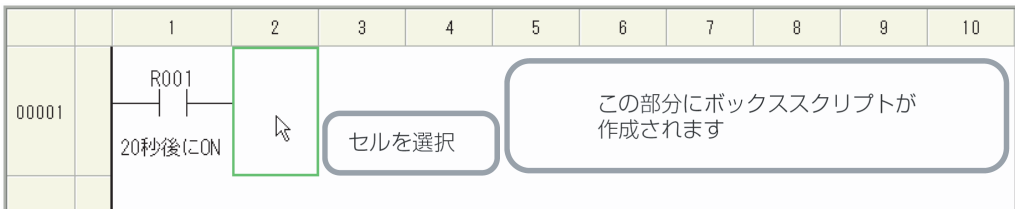
ボックススクリプトを作成/編集するときの、操作の流れについて説明します。



ボックススクリプトを新規作成する

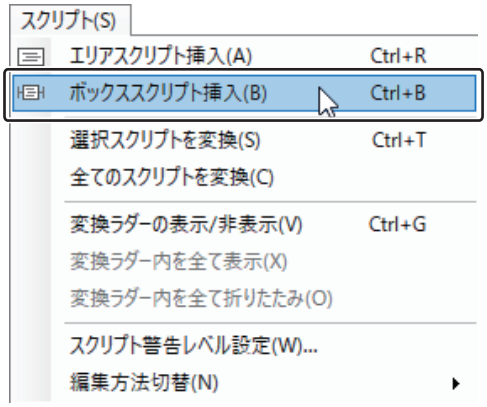
ボックススクリプトを新規に作成する時の手順を説明します。

1 **ラダーエディタで実行条件を記述し、ボックススクリプトを作成したいセルをクリックします。**



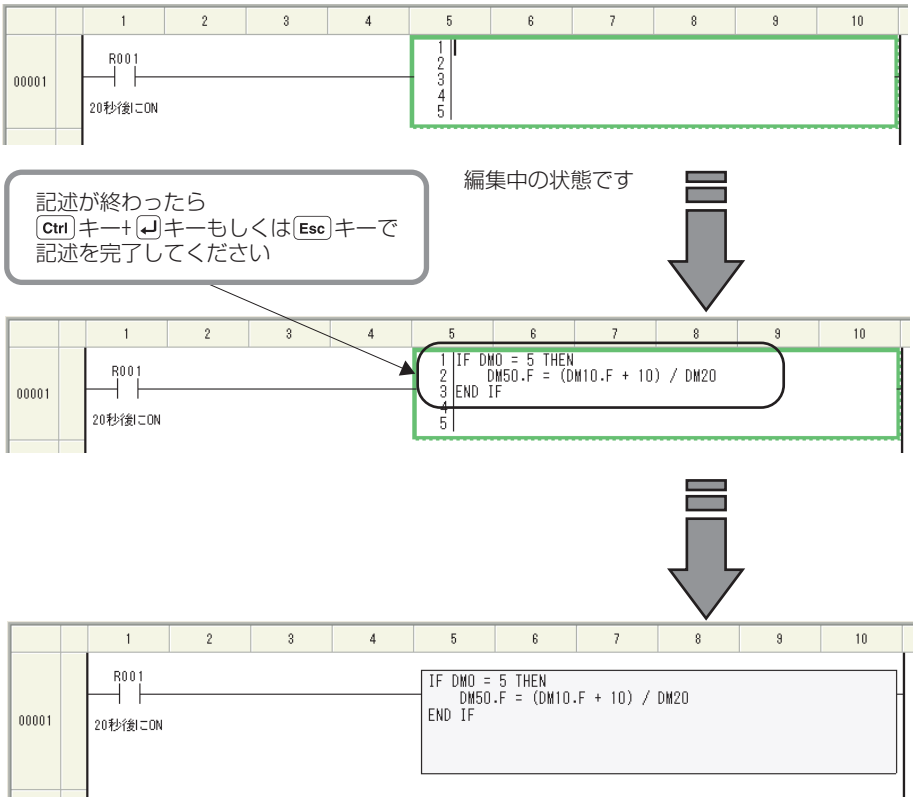
参 考
- ボックススクリプトは6セル(5～10)を占有します。実行条件は1～4セル目までに記述してください。
- 選択したセルがどの位置でも5～10セル部分にボックススクリプトを作成します。

2 **メニューから【スクリプト(S)】→【ボックススクリプト挿入(B)】を選択します。**



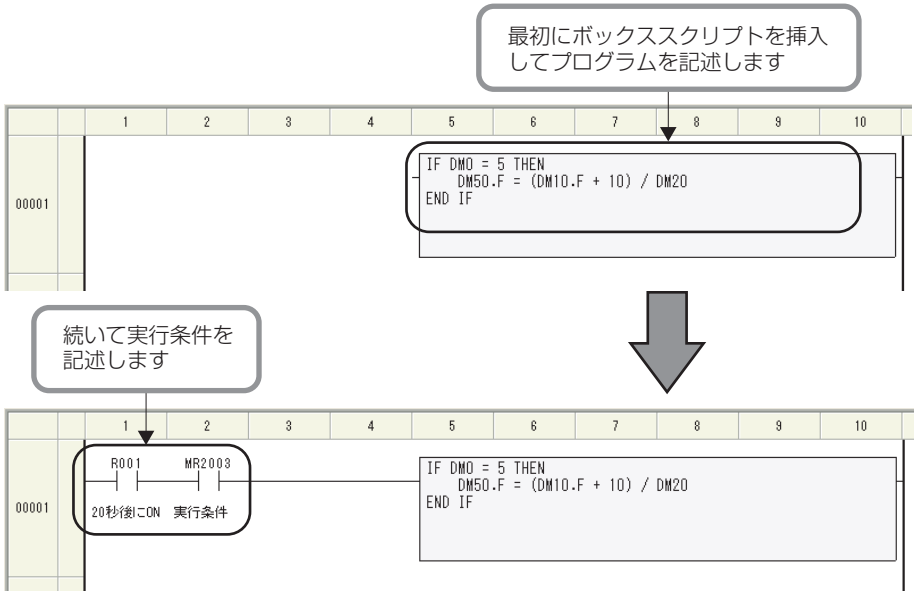
別手順
- Ctrl キー+ B キー
- 右クリックメニューから【スクリプト編集(X)】→【ボックススクリプト挿入(B)】

3 ボックススクリプトが挿入されますので、プログラムを記述してください。
記述が終わったら、Ctrl キー +↵キーもしくは Esc キーを押すか、編集中のボックススクリプト以外のラダー編集ウィンドウをクリックし、記述を完了してください。



記述が完了した状態です

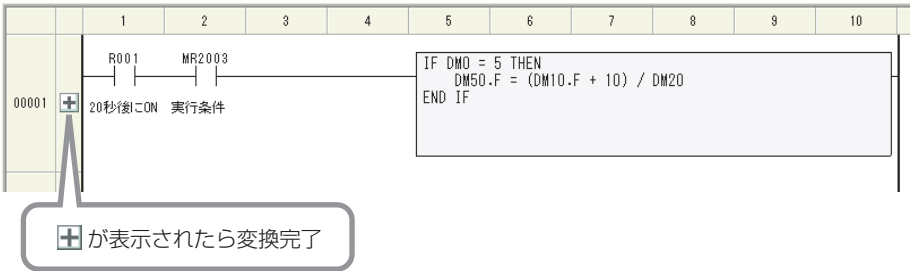
参 考 ボックススクリプトを作成後に、実行条件を記述することもできます。



4 記述が完了したらボックススクリプトを変換します。

別手順　右クリックメニューから【スクリプト編集(X)】→【全てのスクリプトを変換(C)】

●正常に変換された場合



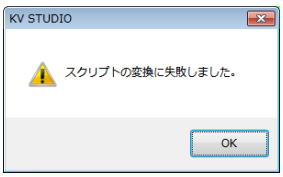
アウトプットウィンドウに「警告」メッセージが表示される場合もあります。

「警告メッセージ」(2-52ページ)

●変換できなかった場合

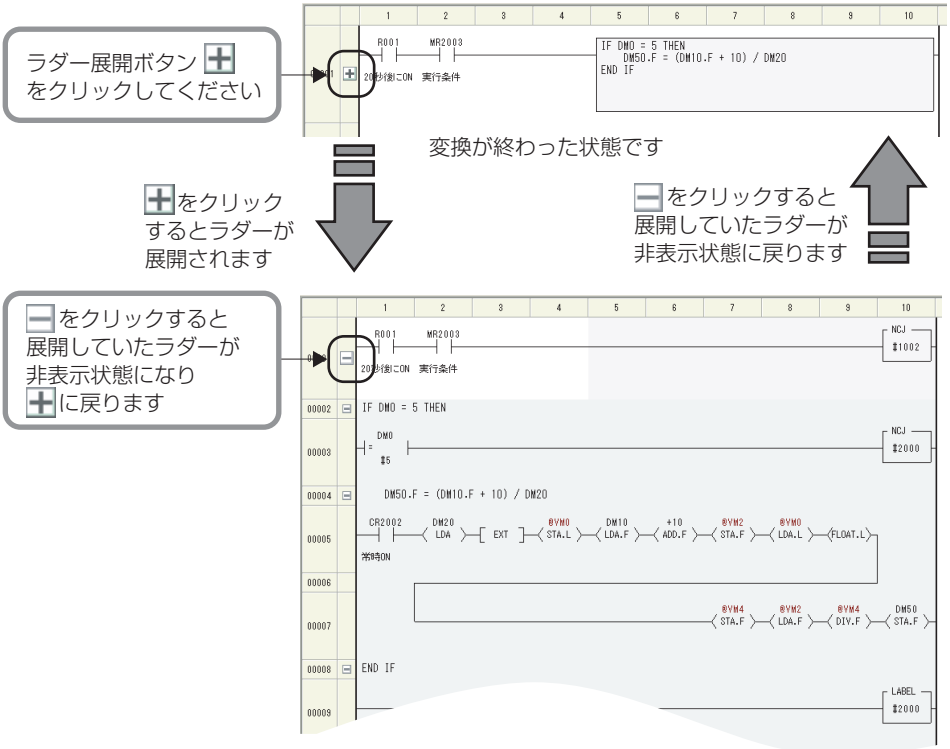
「スクリプト変換エラー」が表示されます。
また、アウトプットウィンドウにエラーメッセージが表示されます。

「変換エラーが発生したときは」(2-55ページ)



5 **スクリプトの変換が成功したらプログラムをラダーで確認できます。**
ボックススクリプトを作成した行の左に表示されている⊞をクリックし、変換されたラダーを表示してください。(スクリプト変換に失敗すると⊞は表示されません)

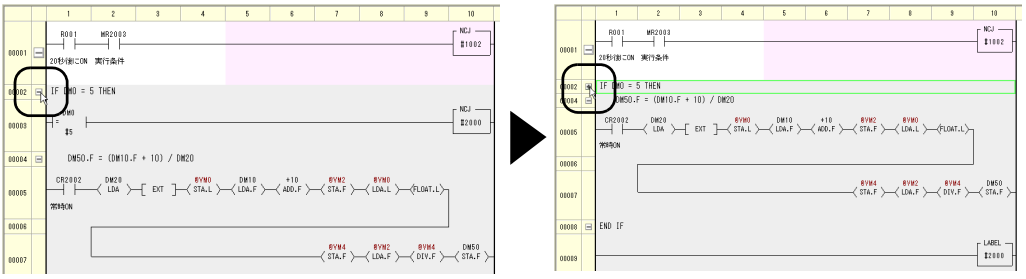
別手順
- 展開するボックススクリプトを選択し、右クリックメニューから【スクリプト編集 (X)】→【変換ラダーの表示/非表示(V)】
- Ctrl キー+ G キー



参考 スクリプト変換後にスクリプトを編集すると変換したラダーは表示されなくなります。(⊞がなくなります)
ラダーを表示するには、再度、スクリプト変換をおこなってください。

6 **展開されたラダーは部分的に非表示(折りたたむ)にできます。**
展開されたラダーの横にある⊟をクリックしてください。

別手順 部分的に折りたたむラダーの行コメント部 (⊞マークのある所) を選択し、右クリックメニューから【変換ラダーの表示/非表示(V)】



ボックススクリプトを再編集する

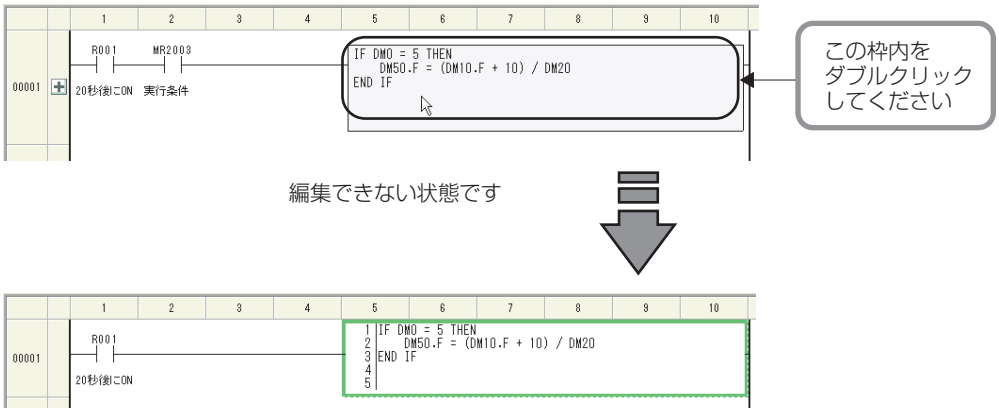
ボックススクリプトを再編集する場合の手順について説明します。

■ボックススクリプトを編集するには

1 **すでに記述しているボックススクリプトを再編集する場合は、編集したいボックススクリプトをダブルクリックしてください。**

別手順　編集したいボックススクリプトを選択してから

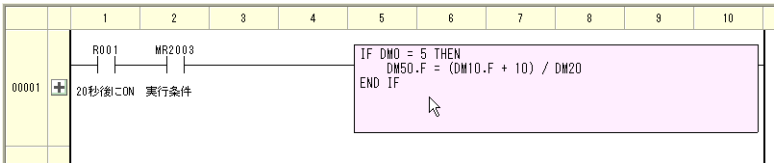
- Esc キー
- F2 キー
- ↵ キー



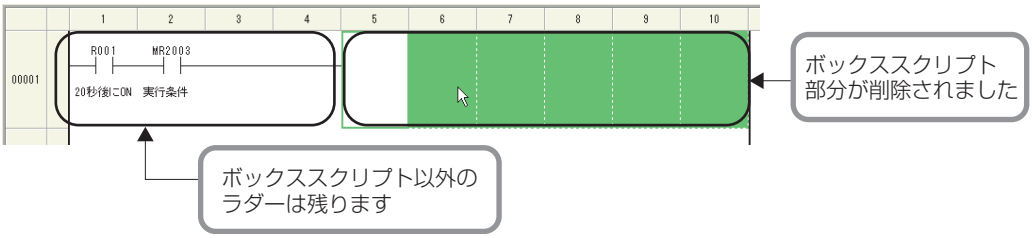
編集可能になりました

■ボックススクリプトを削除するには

1 **削除したいボックススクリプトを選択します。**

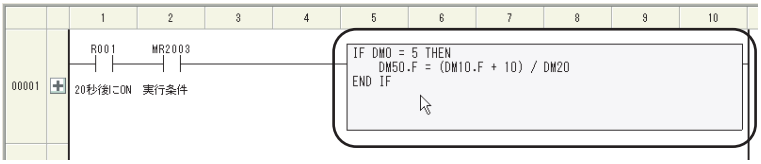


2 **キーボードの Delete キーを入力します。**



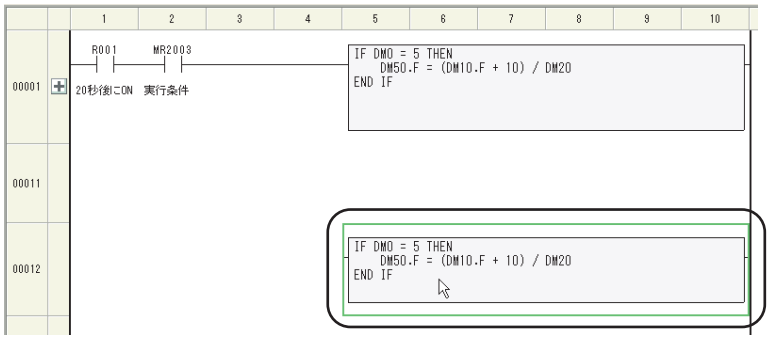
■ボックススクリプトをコピーするには

1 コピーしたいボックススクリプトを選択して、メニューから【編集(E)】→【コピー (C)】を選択します。



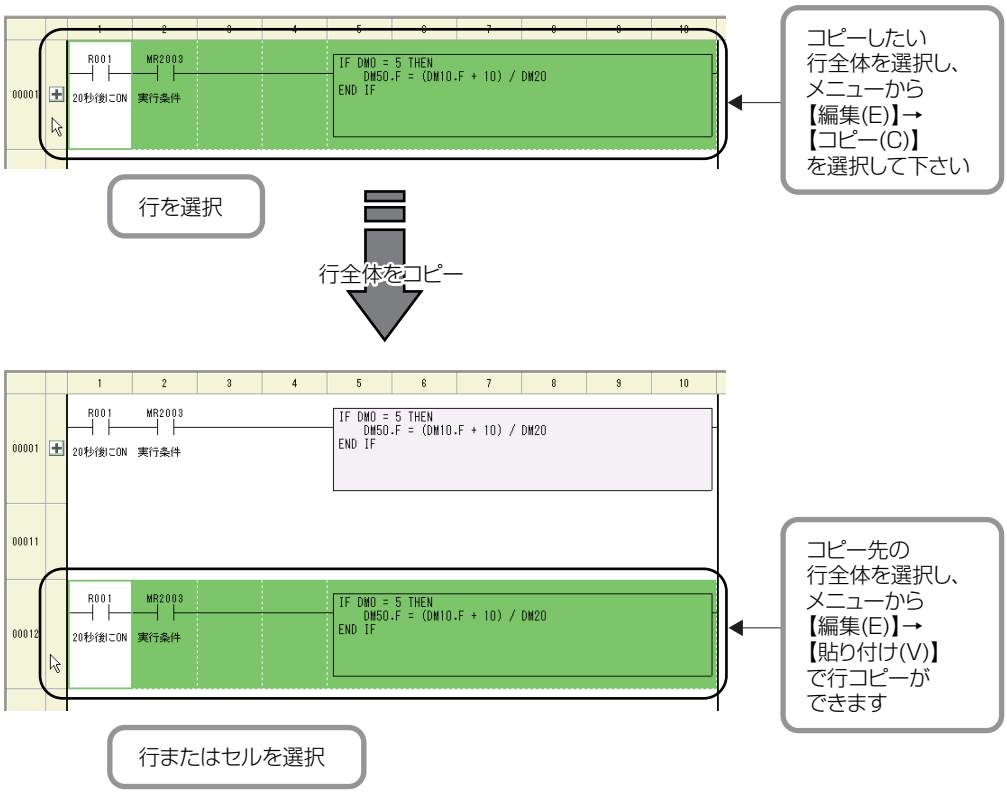
別手順
- Ctrl キー+ C キー
- 右クリックメニューから【コピー(C)】

2 手順1でコピーしたボックススクリプトを貼り付けるセル(行)を選択します。
メニューから【編集(E)】→【貼り付け(P)】を選択します。



別手順
- Ctrl キー+ V キー
- 右クリックメニューから【貼り付け(P)】

参 考 ボックススクリプトを含む行全体をコピーすることもできます。



ボックススクリプトを使用するときの注意

ボックススクリプトは実行条件がOFFのとき、スクリプトプログラムを実行しません。このため、以下のような場合に注意が必要です。

●微分実行型関数を使用する場合

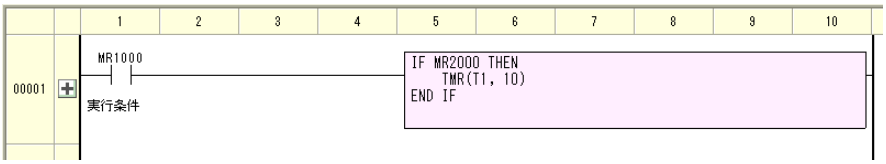
微分実行型関数の実行条件の前状態は前回ボックススクリプト(サブルーチンプログラム)が実行されたときの状態になります。
n回目のボックススクリプト実行と、n+1回目のボックススクリプト実行が連続したスキャンで実行されない場合や、1スキャン内に2回実行されるとき注意が必要です。

詳細は、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「微分実行型命令を使用する時の注意」を参照してください。

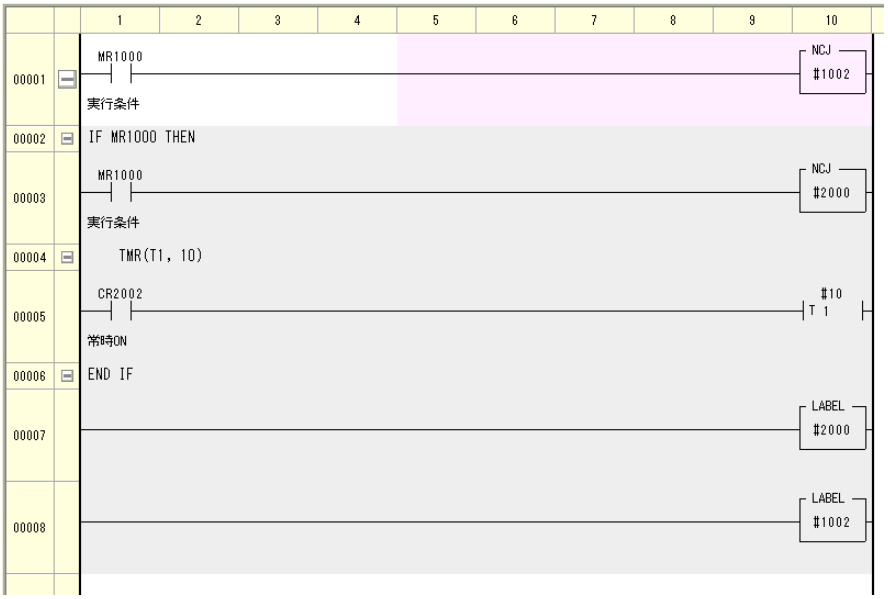
●タイマ関数を使用する場合

ボックススクリプトでタイマ関数を使用した場合、展開したラダープログラムでは、NCJ～LABEL間にTMR命令が記述されるため、タイマ接点が正しくON/OFFしないことがあるので注意してください。また、実行条件を省略したTMR関数も注意が必要です。
例えば、次のようなプログラムを記述した場合、

ボックススクリプト



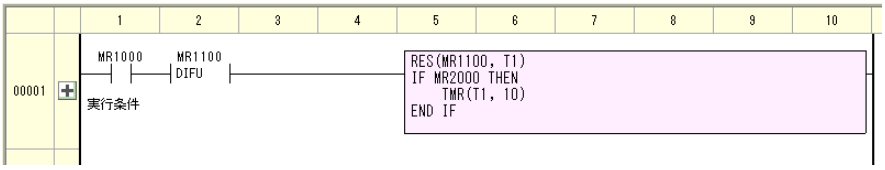
ラダー展開



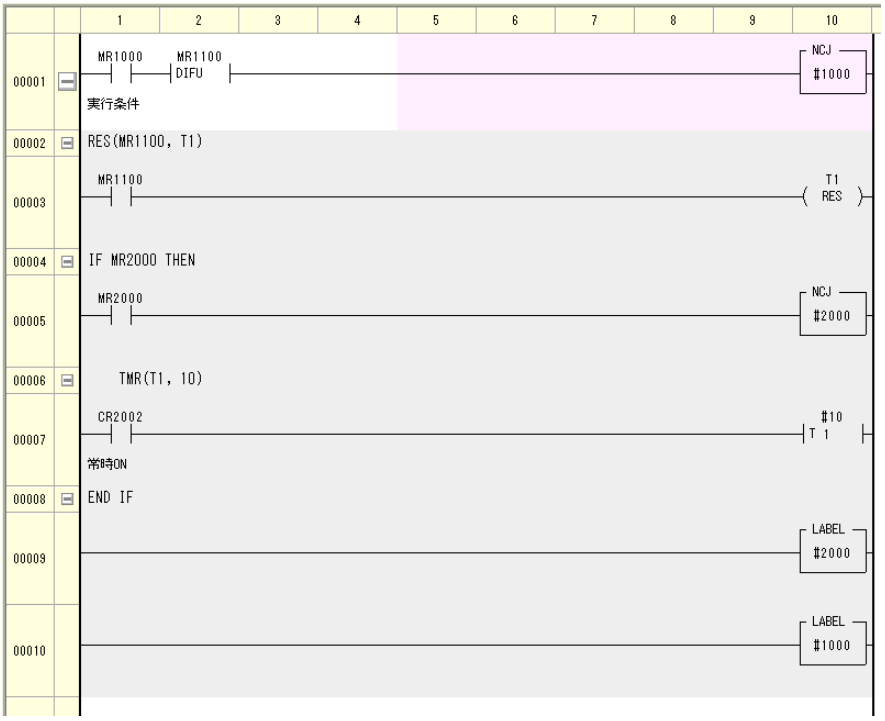
展開したラダープログラムを見ると、タイマ命令のリセット条件がCR2002(常時ON)になっているため、一度実行するとタイマをリセットできません。
この問題を解決するためには、「実行条件としてDIFU命令をAND接続する方法」があります。

実行条件としてDIFU命令をAND接続する方法

ボックススクリプト



ラダー展開



ボックススクリプトの実行条件(MR1000)の立ち上がり時、DIFU命令によりMR1100が1スキャンだけONします。このとき、ボックススクリプトの1行目に記述されたRES関数がタイマをリセットするので、タイマの前状態にかかわらず、常にボックススクリプトの実行開始直後はタイマがクリアされた状態になるので、タイマ関数(接点)は正しく動作します。

エリアスクリプト

ここではエリアスクリプトについての作成/編集方法、使用時の注意点について説明します。

エリアスクリプトの特長

エリアスクリプトは、行のすべてをスクリプトで記述します。

エリアスクリプトはこんなときに使う

エリアスクリプトは毎スキャン実行型なので記述内容は、条件分岐や繰り返し制御を含んだ記述に適しています。
複雑な条件分岐や多数分岐を記述する場合に、プログラムをコンパクトにまとめることができます。

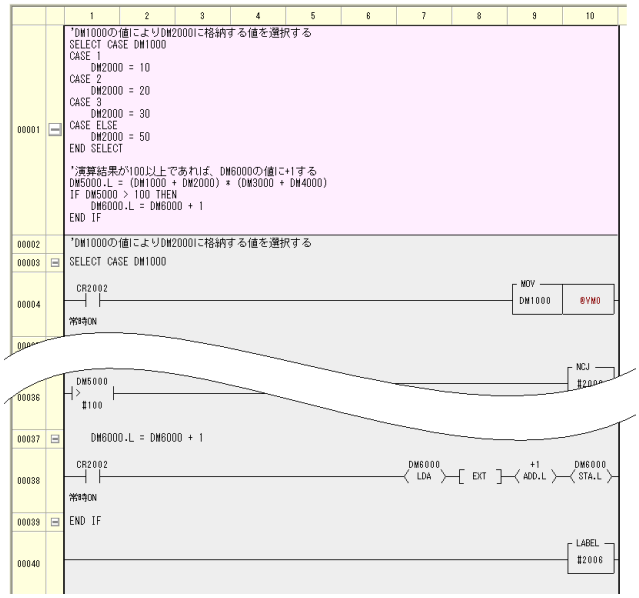
エリアスクリプト

```
00001
'DM1000の値によりDM2000に格納する値を選択する
SELECT CASE DM1000
CASE 1
    DM2000 = 10
CASE 2
    DM2000 = 20
CASE 3
    DM2000 = 30
CASE ELSE
    DM2000 = 50
END SELECT

'演算結果が100以上であれば、DM6000の値に+1する
DM5000.L = (DM1000 + DM2000) * (DM3000 + DM4000)
IF DM5000 > 100 THEN
    DM6000.L = DM6000 + 1
END IF
```

※'(シングルクォーテーション)より右側の記述はコメントとして処理されます。

ラダー記述



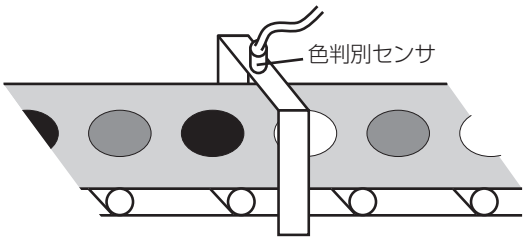
条件分岐が複数ある場合や、ラダーで記述すると長くなる処理などは、エリアスクリプトを使用すると作業効率が上がります。

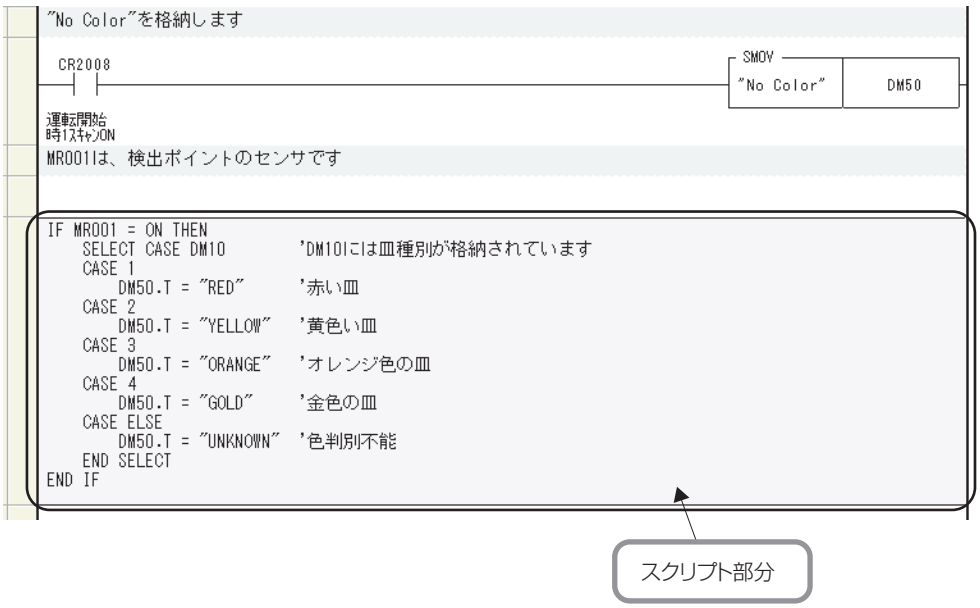
●エリアスクリプトを使用したサンプルプログラム

ベルトコンベア上の、あるポイントを通過した皿の種類を判定して、対応する色の名前を格納します。ポイントに皿が通過していない場合、ビットデバイス MR001 はOFFです。皿が通過するとONになります。MR001 がONの時、DM10に格納された皿種別を判定してDM50.T に対応する皿の色を格納します。皿種別の判定から格納までの部分を、エリアスクリプトを使用して記述します。

検出ポイントに皿が来たら、色判別センサで皿の色をしらべます。

- •赤の時　　DM10に1を格納します
- •黄の時　　DM10に2を格納します
- •オレンジの時　DM10に3を格納します
- •金色の時　　DM10に4を格納します



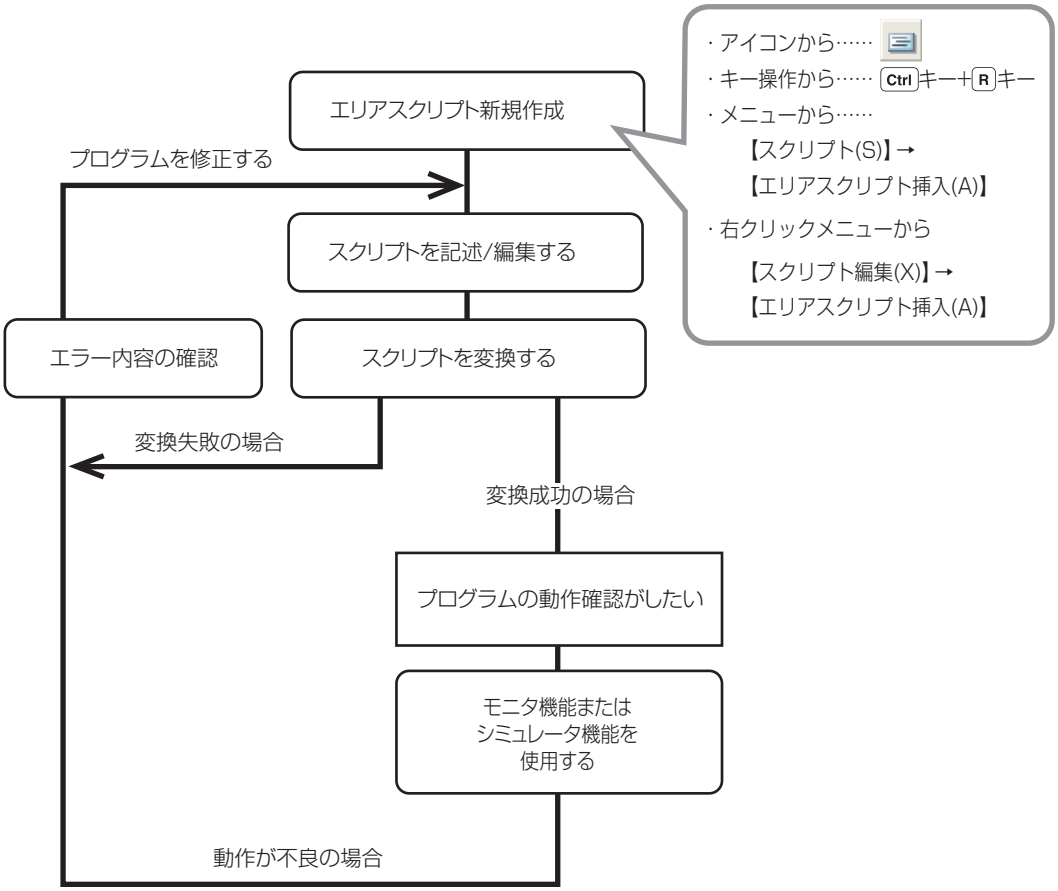


エリアスクリプトを作成する

エリアスクリプトの作成からデバッグまでの流れを説明します。

エリアスクリプトはこんなときに使う

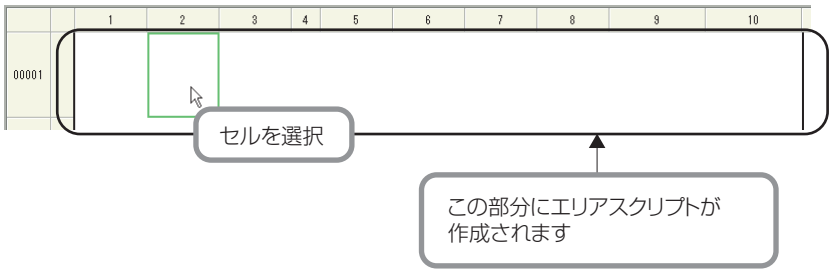
エリアスクリプトを作成/編集するときの、操作の流れについて説明します。



エリアスクリプトを新規作成する

エリアスクリプトを新規に作成するときの手順を説明します。

1 エリアスクリプトを作成したいセル(行)をクリックします。



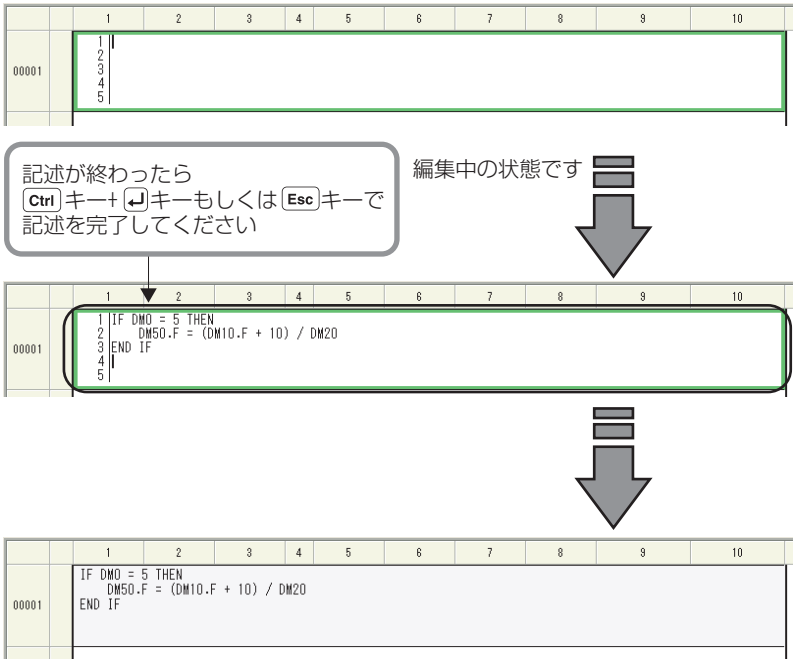
参 考　エリアスクリプトは1行(10セル)全てを占有します。

2 メニューから【スクリプト(S)】→【エリアスクリプト挿入(A)】を選択します。

※ エリアスクリプトはメニューバーとツールバーとキーボード上から挿入できます。

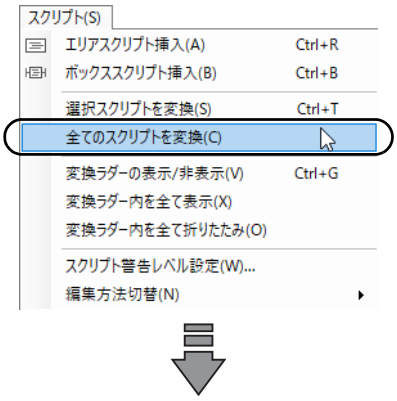
別手順
- Ctrl キー+ R キーを押します。
- 右クリックメニューから【スクリプト編集(X)】→【エリアスクリプト挿入(A)】

3 エリアスクリプトが挿入されますので、プログラムを記述してください。
記述が終わったら、Ctrl キー + ↵ キーもしくは Esc キーを押すか、編集中のエリアスクリプト以外のラダー編集ウィンドウをクリックしてください。



記述が完了した状態です

4 記述を確定したらエリアスクリプトを変換します。



●正常に変換された場合

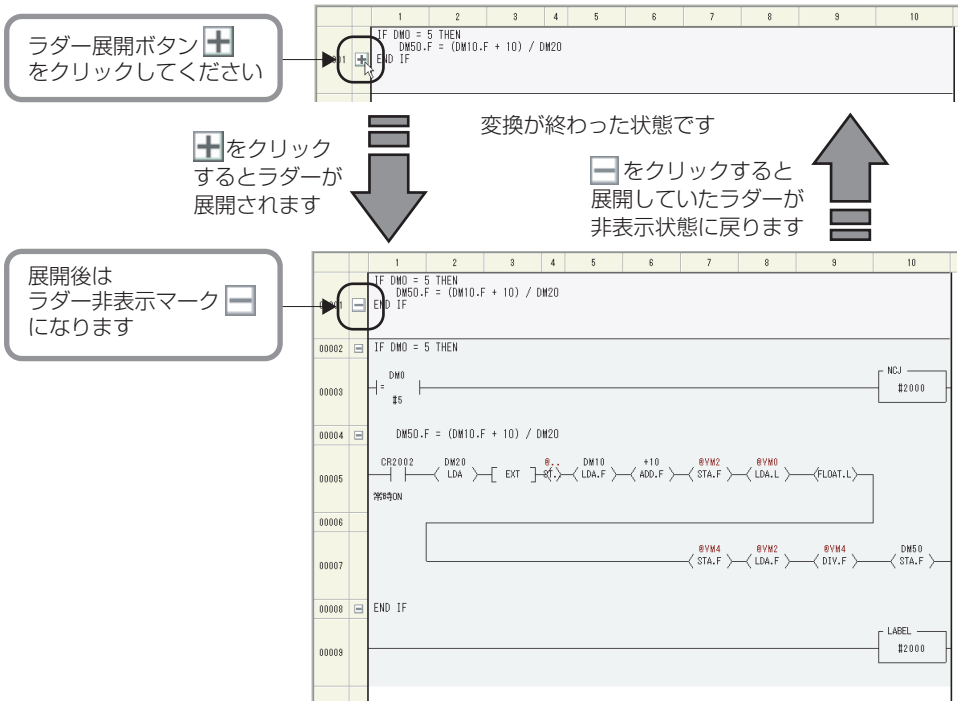
アウトプットウィンドウに「警告」メッセージが表示される場合もあります。
「警告メッセージ」(2-52ページ)

●変換できなかった場合

「スクリプト変換エラー」が表示されます。
また、アウトプットウィンドウにエラーメッセージが表示されます。
「変換エラーが発生したときは」(2-55ページ)

5 スクリプト変換が成功するとエリアスクリプトの左横にラダー展開ボタン⊞が表示されます。この⊞をクリックすると、エリアスクリプトの直下にラダーが展開されます。
(スクリプト変換に失敗すると⊞は表示されません。)

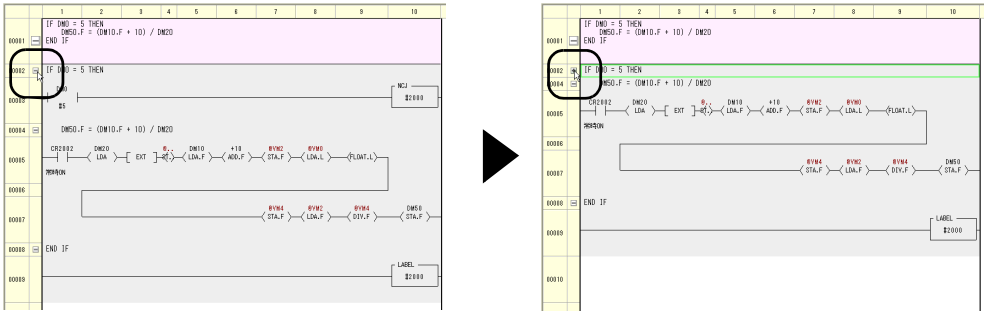
別手順
- 展開するボックススクリプトを選択し、右クリックメニューから【スクリプト編集 (X)】→【変換ラダーの表示/非表示(V)】
- Ctrl キー+ G キー



ラダーが展開された状態です

参考 スクリプト変換後にスクリプトを編集すると変換したラダーは表示されなくなります。(⊞がなくなります)
ラダーを表示するには、再度、スクリプト変換をおこなってください。

6 展開されたラダーは部分的に非表示(折りたたむ)にできます。
展開されたラダーの横にある ⊟ をクリックしてください。



エリアスクリプトを再編集する

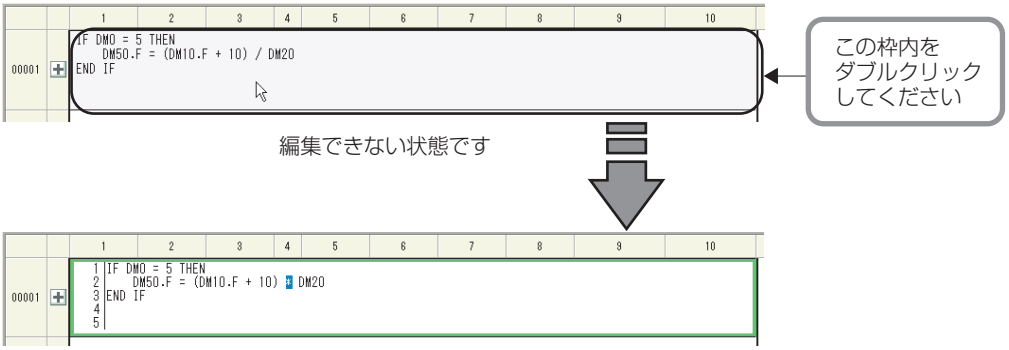
エリアスクリプトを再編集する場合の手順について説明します。

■エリアスクリプトを編集するには

1 すでに記述しているエリアスクリプトを再編集する場合は、編集をおこないたいエリアスクリプトをダブルクリックしてください。

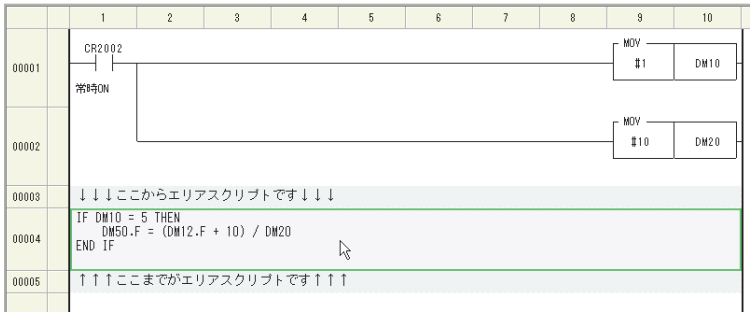
別手順　編集したいエリアスクリプトを選択してから

- Esc キー
- F2 キー



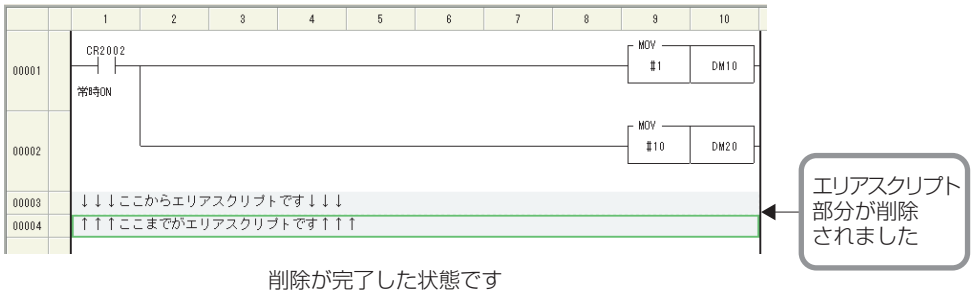
■エリアスクリプトを削除するには

1 削除したいエリアスクリプトを選択します。



エリアスクリプトが選択されている状態です

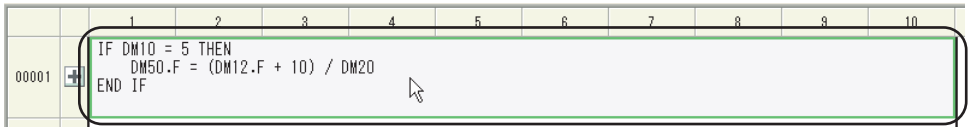
2 キーボードの Delete キーを入力します



削除が完了した状態です

■エリアスクリプトをコピーするには

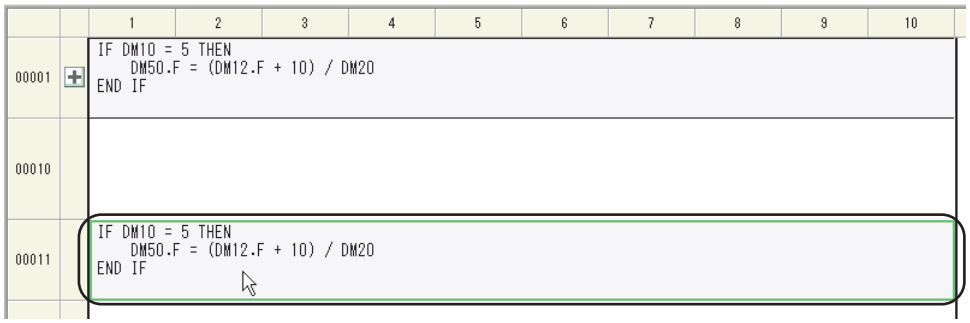
1 **コピーしたいエリアスクリプトを選択して、メニューから【編集(E)】→【コピー (C)】を選択します。**



エリアスクリプトが選択されている状態です

別手順
- Ctrl キー+ C キー
- 右クリックメニューから【コピー(C)】

2 **手順1でコピーしたエリアスクリプトを選択したセル(行)に貼り付けます。**
メニューから【編集(E)】→【貼り付け(P)】を選択します。



コピーが完了した状態です

別手順
- Ctrl キー+ V キー
- 右クリックメニューから【貼り付け(P)】

変数プログラミングと配列 ・ 構造体

変数を使用したプログラミングの方法や注意事項と、配列･構造体の使用方法を説明します。

ポイント **変数を使用できるのは、KV STUDIO Ver11.0以降で、対応機種がKV-8000のVer.2.0以降の場合のみです。**

変数プログラミング

変数とは、データ型と名前が定義された、プログラミングで使用できる要素のことを言います。KV STUDIOでは、デバイスの他に変数を用いてプログラミングできます。変数を用いたプログラミングには以下のような特長があります。

- あらかじめ定義した名前を用いてプログラミングできるため、プログラムの可読性が向上します。
- 変数は、定義した変数に合わせて必要なサイズが CPU ユニットのワーク領域に自動的に割り付けられます。そのため、割り付けを意識する必要がありません。
- あらかじめデバイスマップを作成しておく必要がなくなる、デバッグ効率が向上するなど、プログラムの作成工数が短くなります。
- 異なるプロジェクト間でプログラムをコピーした場合も、変数の割り付けは自動で行われます。デバイスのように、デバイスマップを変更してコピーする必要がなく、プログラムの流用性が向上します。
- タッチパネルディスプレイ VT5 シリーズも、KV-8000 シリーズの変数に対応しているため、デバイスのようにPLCのデバイス番号を変更した際にタッチパネル側のデバイス番号もあわせて変更する、といった作業が不要になります。

グローバル変数とローカル変数

変数にはグローバル変数とローカル変数の2種類があります。それぞれの特徴を以下に示します。

内　容	グローバル変数	ローカル変数
実体	プロジェクト全体で同じ実体を共通して扱います。異なるプログラムで同じグローバル変数名を記述した場合は同じ実体を参照します。	プログラム内だけで使用できる変数です。異なるプログラムで同じローカル変数名を記述した場合でも異なる実体を参照します。
割付	デバイスに割り付ける手動割付と、ワークメモリにCPUユニットが自動で割り付ける自動割付があります。	ワークメモリにCPUユニットが自動で割り付けます。デバイスには割り付けられません。
用途	タッチパネルなど外部機器との通信領域に使用できます。また、拡張ユニットのデバイスを割り付けておくと、動作許可などの拡張ユニットのデバイスを変数でプログラミングできます。	同じ変数名でもプログラムごとに競合しないため、演算のための領域として使用できます。

変数とデータ型一覧

変数を使用する場合、あらかじめデータ型を定義する必要があります。CPUユニットは定義されたデータ型に応じて必要なデータサイズを自動的にワーク領域に割り付けます※1。
KV-8000で使用できるワーク領域のサイズと、変数の仕様は以下のとおりです。

項　目		内　容
ワーク領域のサイズ	ビット	64000点
	ワード	589824ワード
	タイマ	4000点
	カウンタ	4000点
変数の仕様	定義可能数	100000点※2
	変数名の最大長	128文字※3

※1　グローバル変数として定義した場合は、手動でCPUユニットのデバイスに割り付けることも可能です。
※2　定義可能数とは変数編集ウィンドウで登録できる変数名を表します。例えば10000個の要素をもつ配列型変数を1つ定義した場合は1点としてカウントします。
※3　変数名の文字コードはUTF-16です。

ポイント

BOOL型変数は自動割付時にCH先頭に割り付かない場合が有ります。CHの先頭を指定する必要があるオペランドにBOOL型変数を使用する場合は以下のいずれかの方法でCH先頭に割り付けた状態で指定してください。

- **ビットオフセットを0にしたBOOL型メンバを定義した構造体型のローカル変数/グローバル変数を作成する。**
- **割り付けデバイスをグローバルデバイスのCH先頭に指定して、グローバル変数を作成する。**

KV-8000で使用できる変数のデータ型と、データサイズは以下のとおりです。

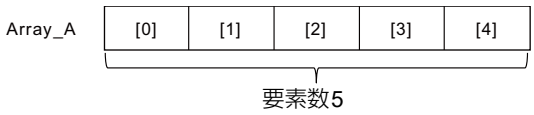
データ型		データサイズ (ビット単位)
型　名	説　明	
BOOL	ビット	1
UINT	1ワード符号なし整数	16
INT	1ワード符号あり整数	16
UDINT	2ワード符号なし整数	32
DINT	2ワード符号あり整数	32
REAL	単精度浮動小数点型	32
LREAL	倍精度浮動小数点型	64
STRING※1	文字列	(文字数※2+1)×8を16の整数倍に切り上げ
TIMER	タイマ	—
COUNTER	カウンタ	—
ARRAY	配列	データ型のデータサイズ × 配列サイズ
構造体	—	定義による※3
ファンクションブロック	ファンクションブロック	定義による※3

※1　STRING 型で扱う文字コードはプロジェクト言語設定で設定した言語に従います。プロジェクト言語設定の詳細は『KV STUDIO ユーザーズマニュアル』を参照してください。
※2　文字数とはNULLまでのバイト数のことです。Shift-JISなどの2バイト文字の場合は1文字で文字数を2として計算します。
※3　構造体およびファンクションブロックは複数のデータ型を設定して使用するデータ型です。そのため、設定内容によりデータサイズが決まります。

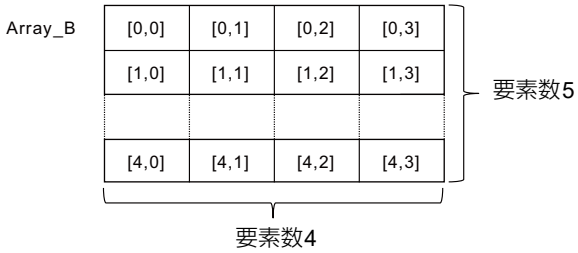
配列

配列とは、同じデータ型のデータの集合体を1つの名前で定義するデータ型です。配列を構成する1つ1つのデータを要素といいます。配列では、変数名の他に、データ型と要素数と次元数を設定します。次元数がnの配列を、n次元配列と呼びます。以下に、1次元配列～3次元配列の例とイメージ図を示します。

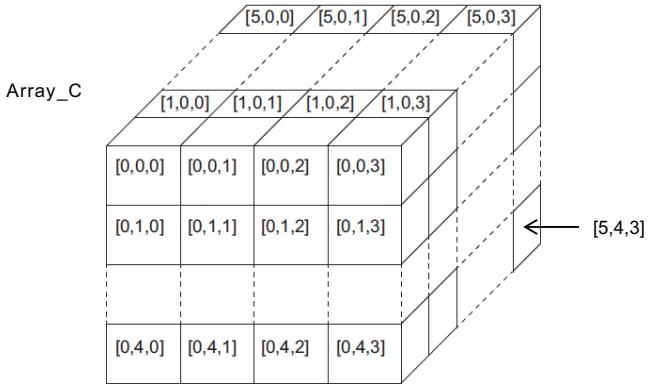
- 1次元配列　変数名：Array_A　要素数：5の場合



- 2次元配列　変数名：Array_B　要素数：5×4の場合



- 3次元配列　変数名：Array_C　要素数：6×5×4の場合



■配列型変数の記載

●配列型変数の要素の指定

プログラム上で配列の要素を指定して使用する場合は、配列名と[　]の添え字を組み合わせて使用します。2次元以上の配列の場合は、要素番号を「カンマ(,)」で区切って指定します。例えば、3次元配列Array_Cの[5,0,3]の要素を使用する場合、プログラム上では以下のとおり「Array_C[#5,#0,#3]」と記載します。(#は省略して入力できます)

```
Array_C[#5, #0, #3] = #0
```

●配列型変数の全体の指定

配列を構成する全要素のことを「配列全体」といいます。スクリプト上で配列全体を指定する場合は、変数名だけを記載します。例えば、Array_Aという変数の配列全体を指定する場合は「Array_A」と記載します。

```
ACNT(Array_A, #2)
```

配列の仕様

項　目	内　容
1次元あたりの最大要素数	589823
最大次元数	8
1配列あたりで占有可能な最大データサイズ	589823ワード
指定可能なデータ型	BOOL / UINT / INT / UDINT / DINT / REAL / LREAL / STRING / ファンクションブロック / 構造体
指定可能な添え字※1	デバイス/定数/変数/演算式※2
添え字に使用できるデータ型	UINT / INT / UDINT / DINT

※1　添え字には0または正の整数を指定してください。その他の数を指定した場合、変換エラーまたは演算エラーが発生します。

※2　添え字に演算式を使用できるのは KV スクリプトのみです。ラダープログラム上では添え字に演算式を使用できません。

ポイント

- **要素数「10」で設定したArray_Aという配列に対してArray_A[10](11番目の要素)と指定するなど、指定可能な範囲を超えて要素を指定した場合、意図しない領域の値を参照する可能性があります。特に、添え字に変数や演算式を使用する場合は、配列を参照する前にACNT関数やWSIZE関数を実行して配列の要素数や配列全体のデータサイズを取得し、指定可能な範囲を超えた要素を参照しないようにしてください。**
  「ACNT関数」(6-60ページ)
  「WSIZE関数」(6-62ページ)
- **配列の添え字にデバイスを指定した場合は16ビット符号無し整数として扱われます。**
- **添え字にビットデバイス(RやMRなど)、ワードデバイスのビット指定、BOOL型の変数は指定できません。**
- **KV STUDIOのバージョンによって、配列のデータ型の表記が以下のとおり異なります。**
  例：要素数が3のUINT型配列の場合
  Ver.11.20以降　ARRAY[0..2] OF UINT
  Ver.11.10以前　UINT[3]

構造体

構造体とは、複数のデータ型のデータの集合体を1つの名前で定義するデータ型です。構造体を構成する1つ1つのデータをメンバといいます。構造体では、定義名の他に、構造体を構成するメンバ名とメンバのデータ型を設定します。構造体には以下のような特長があります。

- 事前に構造体のメンバとデータ型を登録しておくことで、同じ構造を持った変数を作りやすくなる
- 登録モニタウィンドウなどでモニタする際に構造体変数を登録すると、すべてのメンバを一括でモニタできる

モータ軸の以下の情報をプログラムで使用する場合を例に説明します。

現在座標(CurCoordinate)
現在速度(CurSpeed)
軸制御中(Control)
軸エラー(Error)
軸エラーコード(ErrorCode)

構造体を使用しない場合、軸を複数使用しようとすると以下のように軸番号が異なるだけの同一内容の変数を登録する必要があります。
構造体を使用する場合は、軸情報のデータ構造を構造体として事前に定義しておくことで、複数の軸を使用する場合でもデータ型に同じ構造体を設定するだけで済みます。

<構造体を使用しない場合>

変数名	データ型
Ax1_CurCoordinate	DINT
Ax1_CurSpeed	DINT
Ax1_Control	BOOL
Ax1_Error	BOOL
Ax1_ErrorCode	UINT
Ax2_CurCoordinate	DINT
Ax2_CurSpeed	DINT
Ax2_Control	BOOL
Ax2_Error	BOOL
Ax2_ErrorCode	UINT

<構造体を使用する場合>

変数名	データ型
Ax1	AxisInfo
Ax2	AxisInfo

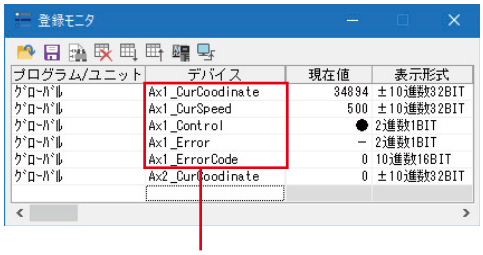
「AxisInfo」型データの定義

メンバ名	データ型
CurCoordinate	DINT
CurSpeed	DINT
Control	BOOL
Error	BOOL
ErrorCode	UINT

事前に設定した構造体を流用

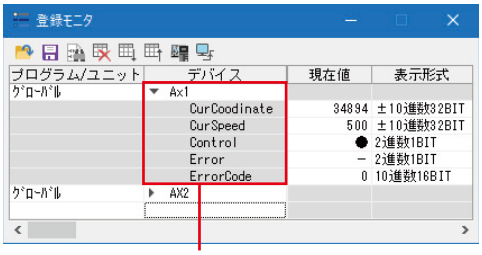
また、KV STUDIO上でモニタする際も、構造体を使用しない場合はすべての変数を手動で入力する必要がありますが、構造体を使用する場合は変数名を入力するだけで構造体内のメンバも一括でモニタできます。

<構造体を使用しない場合>



モニタしたい変数はすべて入力する必要があります。

<構造体を使用する場合>



構造体の場合、変数名を入力するとメンバを一括でモニタできます。

■構造体型変数の記載
プログラム上で構造体型変数を使用する際の記載方法について説明します。

●構造体型変数のメンバの指定
プログラムで構造体型変数のメンバを使用する場合は、「構造体型変数名+ピリオド(.)+メンバ名」と記載します。例えば、Ax1というAxisInfo型変数のCurSpeedというメンバを参照する場合は、「Ax1.CurSpeed」と記載します。

```
DMO.L = Ax1.CurSpeed
```

また、構造体型変数のメンバのデータ型に構造体を指定できます。以下の例では、「ProductInfo」という構造体型の変数Product_Aに「ProductionDate」という構造体型変数がネストされています。

変数名	データ型
Product_A	ProductInfo

「ProductInfo」型データの定義

メンバ名	データ型
Name	STRING[20]
SerialCode	STRING[20]
ProductionDate	DateInfo

「DateInfo」型データの定義

メンバ名	データ型
Year	UINT
Month	UINT
Day	UINT

Product_AのDayをプログラムで指定する場合は、「Product_A.ProductionDate.Day」と記載します。

```
Product_A.ProductionDate.Day = #31
```

ポイント **構造体のネストは最大8段までです。**

●構造体型変数の全体の指定
構造体型変数を構成する全メンバを「構造体全体」といいます。プログラムで構造体全体を指定する場合は、変数名だけを記載します。例えば、Product_Aという構造体型変数の構造体全体を指定する場合は「Product_A」と記載します。

```
WSIZE(Product_A)
```

ポイント
- **FMOV　関数などを使用して構造体全体に値を書き込む場合、転送データ数に構造体全体のデータ数より大きな値を指定すると、意図しない領域の値を参照する可能性があります。FMOV関数などで構造体全体を参照する場合は、関数実行前にWSIZE関数を実行して構造体全体のデータサイズを取得してください。**
  「WSIZE関数」(6-62ページ)

構造体の仕様

項　目	内　容
構造体の最大定義数	800
構造体1つあたりの最大メンバ数	500
構造体のネスト最大段数	8※1
メンバとして指定可能なデータ型	BOOL / UINT / INT / UDINT / DINT / REAL / LREAL / STRING/ 配列 / 構造体

※1　stAAA.memberと記載した場合、ネスト段数は1段です。stAAA.stBBB.memberと記載した場合、ネスト段数は2段です。

変数の設定

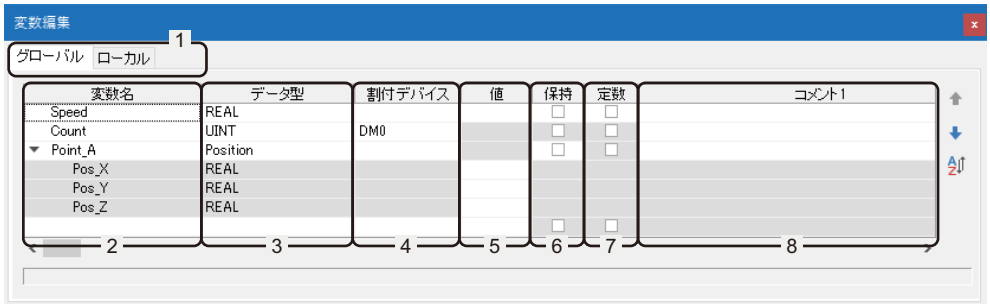
変数を定義してラダープログラムで使用するまでの手順を説明します。

■変数編集ウィンドウから設定する方法

この方法ではグローバル変数、およびローカル変数を設定できます。

1 **【表示(V)】→【変数編集ウィンドウ(L)】を選択し、【編集設定】ウィンドウを開きます。**

別手順 ワークスペースの「変数」をダブルクリック



番　号	項　目	内　容
1	グローバル/ローカルタブ	グローバル変数編集タブとローカル変数編集タブを切り換えます。 「グローバル変数とローカル変数」(2-25ページ)
2	変数名	変数名を入力します(半角128文字/全角128文字)。
3	データ型	変数のデータ型を以下から選択します。 BOOL/UINT/INT/UDINT/DINT/REAL/LREAL/STRING/ TIMER/COUNTER/ARRAY/構造体/ファンクションブロック
4	割付デバイス	変数をCPUユニットのデバイスに手動で割り付ける場合に設定します。割付は必須ではありません。ローカル変数編集タブでは存在しない項目です。
5	値	「定数」の項目にチェックを入れた場合: 変数の初期値を指定します。定数のチェックが入っている場合は値が固定され、プログラムから値を変更できなくなります。 「定数」の項目にチェックを入れない場合: 変数の初期値を設定します。プログラム内で値を変更できます。割付デバイスを設定している場合は、変数編集ウィンドウでは初期値を設定できません。デバイス初期値設定で、割り付けたデバイスの初期値を設定してください。詳細はCPUユニットのユーザーズマニュアルを参照してください。
6	保持	チェックを入れた変数は、電源断時も値が保持されます。保持できる変数はビット、ワード、カウンタの合計で最大30000個です。 割付デバイスを設定している場合は、上記の30000個に含まれないため設定したデバイスに対して停電保持設定を行ってください。 保持にチェックを入れていない変数は、PROGRAMモード→RUNモード切り換え時に「値」で設定した値に初期化されます。設定していない場合は0に初期化されます。 詳細はCPUユニットのユーザーズマニュアルを参照してください。
7	定数	チェックを入れた変数は、定数として扱われます。定数に設定した変数は、最大定義数100000個の中に含まれますが、ワーク領域を消費しません。
8	コメント	変数のコメントを設定します(半角32文字/全角32文字)。設定したコメントはKV STUDIOのラダー編集ウィンドウに表示されます。1つの変数に対してコメント1～コメント8までを設定できます。

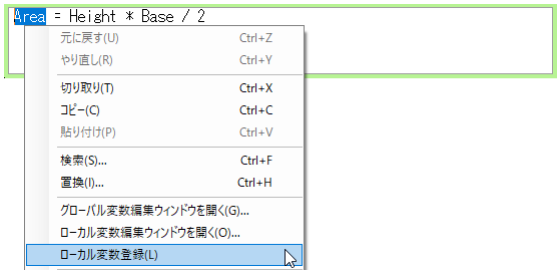
■スクリプト内の未登録変数を取り込んで設定する方法

この方法ではローカル変数のみ設定できます。

1 変数を使用して、スクリプトを記述します。

```
Area = Height * Base / 2
```

2 スクリプト内のローカル変数として登録したい文字列を選択した状態で、右クリックメニューの【ローカル変数登録(L)】を選択します。

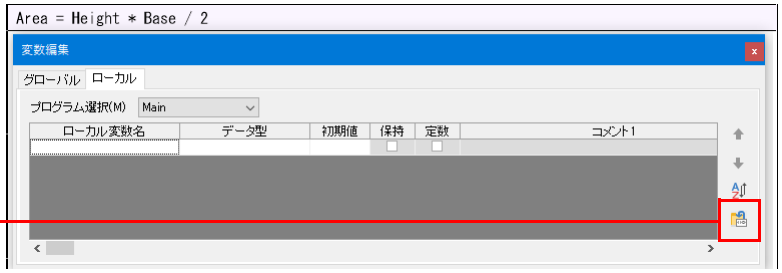


3 変数名が取り込まれた状態で変数編集ウィンドウが表示されるので、データ型を設定します。

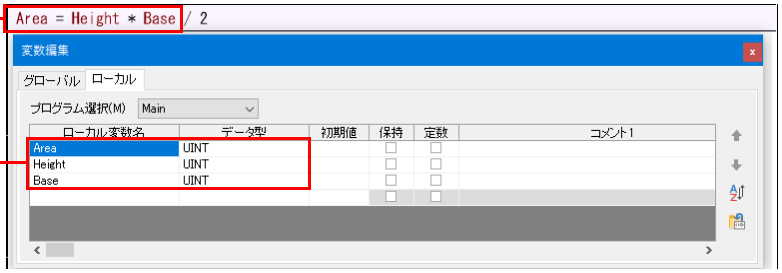
●ローカル変数の一括登録

モジュール内の未登録変数をローカル変数として一括で取り込む方法を説明します。未登録変数が存在する状態で、変数編集ウィンドウの をクリックします。対象モジュール内に存在するすべての未登録変数がローカル変数として取り込まれるので、データ型を設定します。

変数取り込みの
アイコンを
クリックします。



未登録変数が
一括で取り込まれ、
登録された変数の
表示色が変わります。



ポイント

一括登録を実行すると、モジュール内のすべての変数が登録されます。そのため、誤記部も変数として登録されるので、本来変換時に発生するエラーが、正常変換される可能性があります。

例 DN1000=10 .... 一括登録前：スクリプト変換時にエラーになる
.... 一括登録後：スクリプト変換時にエラーは発生しない。

一括登録後に、変数編集ウィンドウにて登録内容を確認することをおすすめします。

変数使用時の注意事項

変数を使用する際の制限や注意事項について説明します。

●変数のサフィックスについて

変数は、変数編集ウィンドウでデータ型を設定するため、スクリプトで記述するときはサフィックスを付加せずに記述できますが、デバイスと同様にサフィックスを付加した記述もできます。また、TYPEによる型宣言もできます。それぞれの優先順位の関係は以下のとおりです。

データ型 < 型宣言 < サフィックス

低　　優先順位　　高

変数登録時に設定したデータ型により、指定できるサフィックスが制限されます。

項　目	グローバル変数(手動割付)									グローバル変数(自動割付)/ローカル変数								
	.B	.U	.S	.D	.L	.F	.DF	.T	省略	.B	.U	.S	.D	.L	.F	.DF	.T	省略
BOOL	○	×	×	×	×	×	×	×	.B	○	×	×	×	×	×	×	×	.B
UINT	○	○	○	○	○	○	○	○	.U	○	○	○	×	×	×	×	×	.U
UDINT	○	○	○	○	○	○	○	○	.D	○	○	○	○	○	○	×	×	.D
INT	○	○	○	○	○	○	○	○	.S	○	○	○	×	×	×	×	×	.S
DINT	○	○	○	○	○	○	○	○	.L	○	○	○	○	○	○	×	×	.L
REAL	×	○	○	○	○	○	○	×	.F	×	○	○	○	○	○	×	×	.F
LREAL	×	○	○	○	○	○	○	×	.DF	×	○	○	○	○	○	○	×	.DF
STRING	×	×	×	×	×	×	×	○	.T	×	×	×	×	×	×	×	○	.T
タイマ	○	○	○	○	○	×	×	×	.B	○	○	○	○	○	×	×	×	.B
カウンタ	○	○	○	○	○	×	×	×	.B	○	○	○	○	○	×	×	×	.B

○：サフィックスに指定可能。ただし、データ型と異なるサフィックスを指定すると変換時に警告が出ます。

×：サフィックスの指定はできません。

● 変数の範囲チェック

KV-8000では関数を実行した結果、変数の範囲を超えた領域を参照しないように範囲チェックを行っています。範囲チェックの内容は変数の設定により異なります。
要素数を10に設定したArray_Aという変数を使用する場合を例に説明します。

グローバル変数（自動割付）/ローカル変数の場合

Array_A[20]のように、要素数を超える定数を指定した場合は、スクリプト変換時にエラーとなります。
Array_A[idx]のように、要素数に定数以外を指定した場合は、スクリプト変換時にはエラーにはなりません。プログラム実行時に、idxの値が0～9の範囲を超えている場合は命令実行時に演算エラーが発生します。

グローバル変数（手動割付）の場合

Array_Aの先頭をDM0に割り付けている場合を例に説明します。
Array_A[20]のように、要素数を超える定数を指定した場合は、スクリプト変換時にエラーとなります。
Array_A[idx]のように、要素数に定数以外を指定した場合は、スクリプト変換時に警告が出ますがエラーにはなりません。プログラム実行時にidxの値が0～9の範囲を超えている場合は、対象となるデバイスが扱われ、演算エラーは発生しません。例えば、idx=19の場合、Array_A[idx]はDM19として扱われます。

● その他の注意事項

- グローバル変数とローカル変数で同じ変数名を設定できますが、使用した場合はローカル変数として動作します。
- 変数名には、予約語など使用できない文字列があります。
  「使用できない文字一覧」付-54ページ
- 自動割付したグローバル変数およびローカル変数はインデックス修飾できません。
- 配列型、構造体型の変数はインデックス修飾や間接指定はできません。
- ［変数編集］ウィンドウで「定数」にチェックを入れた場合、インデックス修飾や間接指定はできません。

ラベルプログラミングとラベル配列

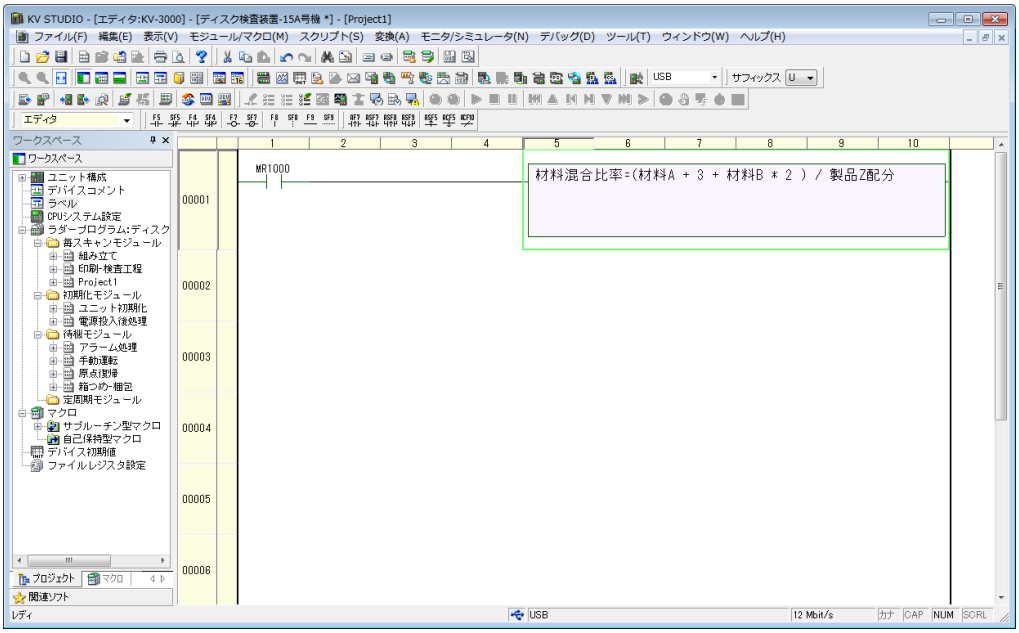
ラベルを使用したプログラミングの方法や注意事項と、ラベル配列の使用方法などについて説明します。

▶ 重 要

- **機種設定が KV-8000（CPU 機能バージョン 2.0 以降）以外の機種のプロジェクトではラベルのみ使用できます。**
  機種設定がKV-8000（CPU機能バージョン2.0以降）の場合、ラベルの代わりに変数を使用できます。
  「変数プログラミング」（2-25ページ）
- **KV-1000 はローカルラベル、ラベル配列を使用できません。グローバルラベルのみ使用できます。**

ラベルプログラミング

プログラムで使用するデバイスにラベルを設定しておくことで、ラベルプログラミングが可能になります。



ラベルには、グローバルラベルとローカルラベルの2種類があります。

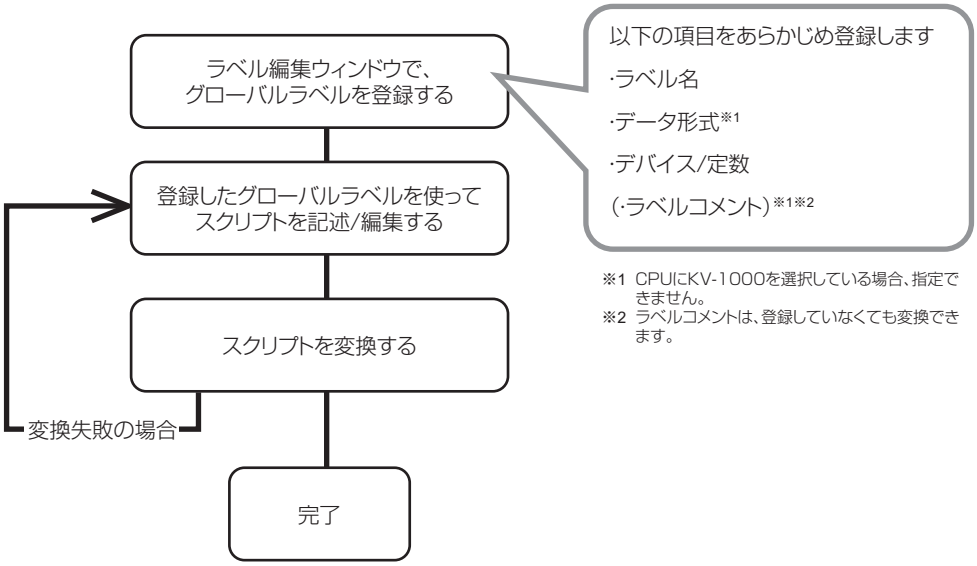
ポイント

- **機種設定が KV-8000（CPU 機能バージョン 2.0 以降）以外の機種のプロジェクトではラベルのみ使用できます。**
  機種設定がKV-8000（CPU機能バージョン2.0以降）の場合、ラベルの代わりに変数を使用できます。
  「変数プログラミング」（2-25ページ）
- **ローカルラベルは、CPUユニットがKV-1000の場合使用できません。**

■グローバルラベル

グローバルラベルとは、プロジェクト全体で共通して扱うラベルのことです。
1つのグローバルラベルには、1つのグローバルデバイスを割り付ける必要があります。

●グローバルラベルを使ったプログラミングの流れ

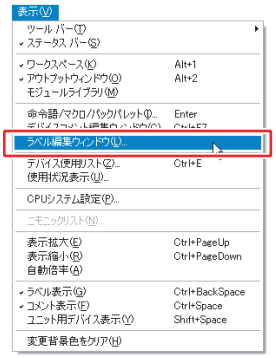


●グローバルラベルを登録する

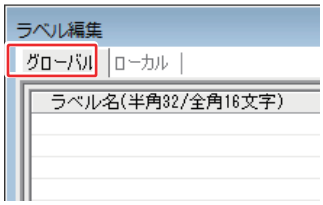
グローバルラベルを登録するときの手順について説明します。

1 **メニューから【表示(V)】→【ラベル編集ウィンドウ(L)】を選択し、ラベル編集ウィンドウを表示させます。**

別手順
- ワークスペースの[ラベル]をダブルクリックする。
- スクリプト編集ウィンドウでツールバーをクリックする。



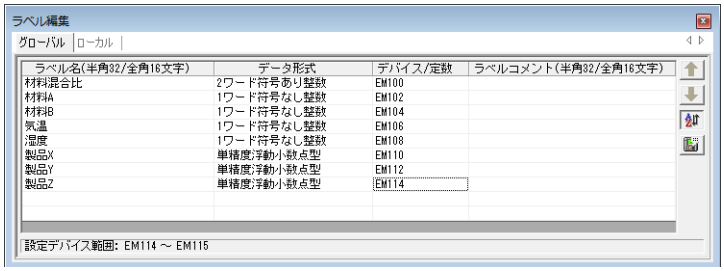
使用するCPUがKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合、[グローバル]タブをクリックします。



2 使用するグローバルラベルを登録します。

ラベル編集ウィンドウの詳細は、『KV STUDIOユーザーズマニュアル』を参照してください。

《CPUにKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズを選択している場合》



項　目	説　明
ラベル名	32文字(KV-5500/5000/3000/1000, KV Nanoシリーズの場合半角32文字/全角16文字)以内で指定します。※1
データ形式	ラベルの型を指定します。入力ボックスをクリックすると表示されるプルダウンメニューから、選択します。
デバイス/定数	グローバルラベルに割り当てるデバイスまたは、定数を指定します。 データ形式により、指定できるデバイス、定数の範囲が異なります。 「ラベルの型について」(2-44ページ)
ラベルコメント	グローバルラベルに付けるコメントを指定します。

※1　KV STUDIOの予約語など、指定できない文字や文字列もあります。
「ラベル使用時の注意事項」(2-44ページ)

《CPUにKV-1000を選択している場合》

項　目	説　明
ラベル名	半角16文字/全角8文字以内で指定します。※1
デバイス	グローバルラベルに割り当てるデバイスを指定します。※2
デバイスコメント	割り付けたデバイスのデバイスコメントを表示します。編集はできません。

※1　KV STUDIOの予約語など、指定できない文字や文字列もあります。
「ラベル使用時の注意事項」(2-44ページ)
※2　指定できないデバイスもあります。
『KV STUDIOユーザーズマニュアル』

ポイント　**グローバルデバイスにのみラベルを設定できます。ローカルデバイスには設定できません。**

3 登録したグローバルラベルを使って、スクリプトプログラムを作成します。

```
00001
1 材料混合比 = (((材料A * 10) + (材料B * 3)) * 気温 / 湿度) * 製品X
2
3
```

登録済みのラベルは、スクリプト編集中に文字色が変わります。

記述が終了したら、スクリプトを変換します。

```
00001  材料混合比 = (((材料A * 10) + (材料B * 3)) * 気温 / 湿度) * 製品X
00002  材料混合比 = (((材料A * 10) + (材料B * 3)) * 気温 / 湿度) * 製品X
00003  CR2002 ─┤├─ 湿度 LDA ─[ EXT ]─ @VM0 STA.L ─ 材料B LDA ─[ EXT ]─ +3 MUL.L ─ @VM2 STA.L ─ 材料A LDA ─
       常時ON
00004
00005  [ EXT ]─ +10 MUL.L ─ @VM2 ADD.L ─ @VM4 STA.L ─ 気温 LDA.S ─[ EXT.S ]─ @VM4 MUL.L ─ @VM0 DIV.L ─
00006
00007  (FLOAT.L)─ 製品X MUL.F ─ (INTG.L)─ 材料混合比 STA.L ─
```

●グローバルラベルを使ったプログラム例

```
00009
'---FOR-NEXT文---

'<<ラベルプログラミングで記述がよりわかりやすくなります>>
'-----------------------------------------------------------------------
'X=EM100.U  Y=EM102.U  結果値=EM104.F  中止ボタン=MR200  警告ランプ=MR201
'-----------------------------------------------------------------------

FOR X = 0 TO 1000 STEP 2        '「2」インクリメントします
    FOR Y = 0 TO 200 STEP 1     '「1」インクリメントします
        結果値 = Y * X / 3
        IF 中止ボタン = ON THEN
            警告ランプ = ON
            BREAK
        END IF
    NEXT
NEXT
```

■ローカルラベル

ローカルラベルとは、CPUがKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズのときに使用可能なラベルで、各モジュール内でのみ有効になり、モジュールごとに設定する必要があります。
ローカルラベルは、KV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズのシステム領域(ワークメモリ)のデバイス(VB、VM)を使用しているので、デバイスの使用状況を意識せずにローカルラベルを設定できます。

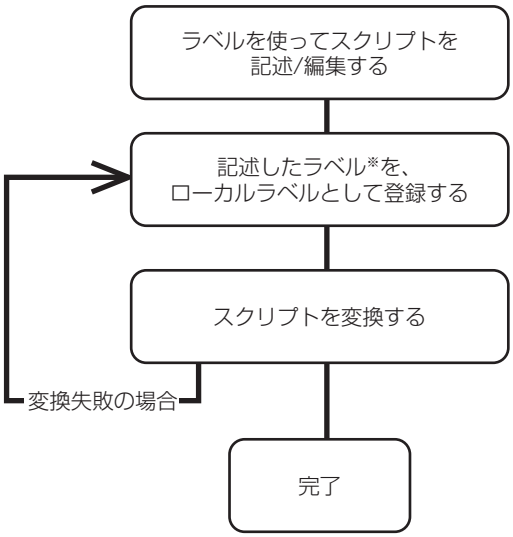
ポイント **使用するCPUがKV-1000の場合、ローカルラベルは使用できません。**

参考 ワークメモリは以下のとおり用意されています。

KV-8000	VM:589824ワード	VB:64000点(4000ch)
KV-7500/7300/5500/5000/3000	VM:50000ワード	VB:16384点(2024ch)
KV Nanoシリーズ	VM:9500ワード	VB:8192点(512ch)

割り当て状況はKV STUDIOで管理します。
「スクリプト用ワークメモリ(ワークデバイス)について」(2-74ページ)

●ローカルラベルを使ったプログラミングの流れ

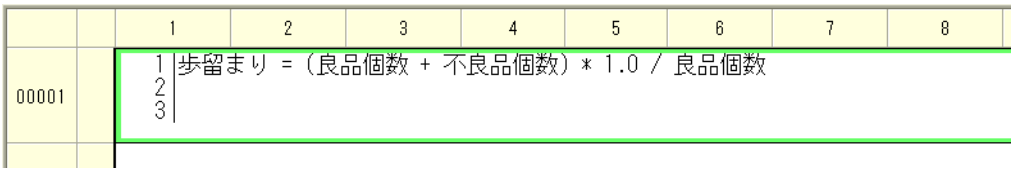


※ スクリプトプログラムを記述する前に、あらかじめローカルラベルを登録することもできます。

●ローカルラベルを登録する

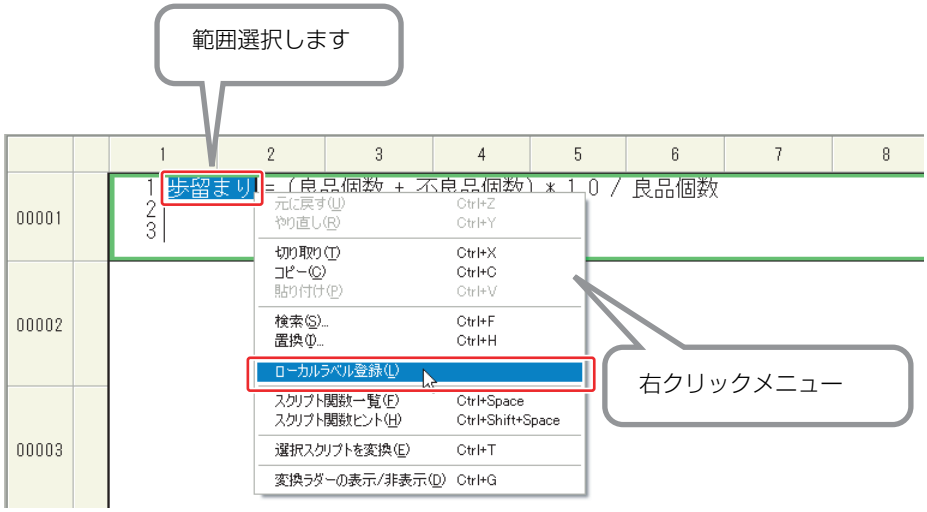
ローカルラベルを登録するときの手順について説明します。

1 ラベルを使用して、スクリプトを記述します。

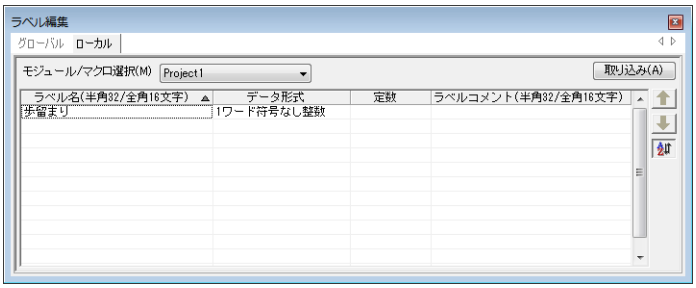


ラベルがまだ登録されていないので、文字色は変わりません。

2 スクリプトの編集状態のまま、ローカルラベルとして登録したいラベルを範囲選択し、右クリックメニューの【ローカルラベル登録(L)】を選択します。



3 ラベル編集ウィンドウで、データ形式を設定します。

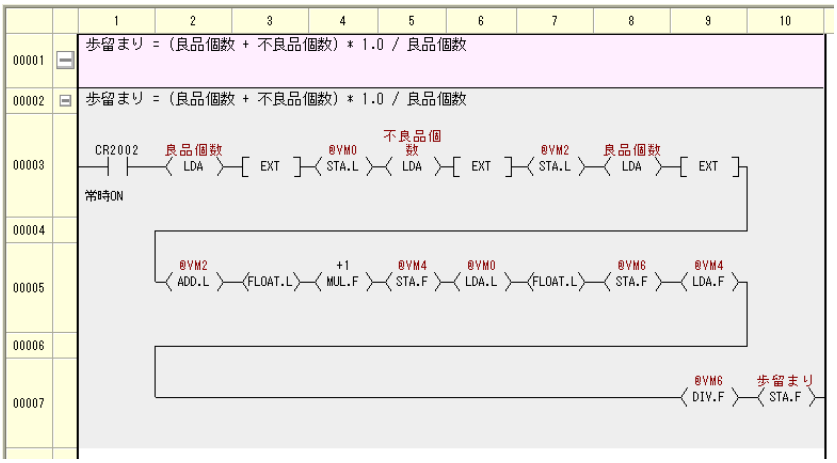


項　目	説　明
ラベル名	範囲選択したラベルが自動的に格納されます。※1
データ形式	デフォルトで「1ワード符号なし整数」が選択されます。 クリックするとプルダウンメニューが表示されるので、データ形式を変更できます。 「ラベルの型について」(2-44ページ)
定数	ローカルラベルに定数を割り付ける場合は、定数を指定します。空白の場合、定数は割り付きません。
ラベルコメント	ローカルラベルに付けるコメントを指定します。

※1　KV STUDIOの予約語など、指定できない文字や文字列もあります。
「ラベル使用時の注意事項」(2-44ページ)

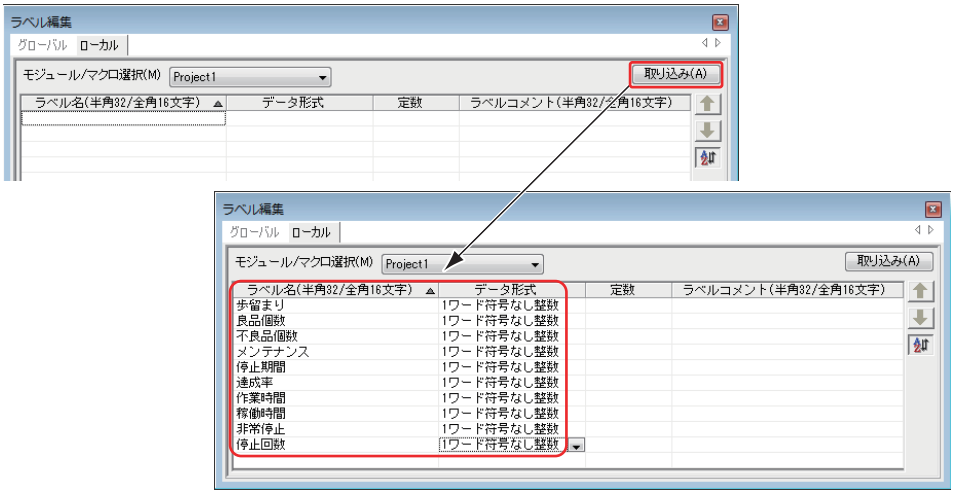
参 考　編集中モジュールのすべてのラベルをローカルラベルとして、一括登録できます。
「ローカルラベルの一括登録」(2-43ページ)

4 すべてのローカルラベルの登録が完了したら、変換します。



●ローカルラベルの一括登録

編集中のモジュールで、未登録ラベルのすべてをローカルラベルとして一括登録します。



各ローカルラベルのデータ形式は「1ワード符号なし整数」になるため、個別に変更します。

ポイント

一括登録を実行すると、モジュール内のすべてのラベルが登録されます。そのため、誤記部もラベルとして登録されるので、本来変換時に発生するエラーが、正常変換される可能性があります。

例 DN1000=10 ....一括登録前:スクリプト変換時にエラーになる

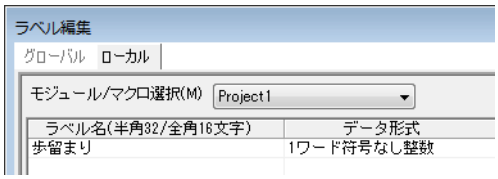
....一括登録後:スクリプト変換時にエラーは発生しない。

一括登録後に、ラベル編集ウィンドウにて登録内容を確認することを推奨します。

■ラベル使用時の注意事項

●ラベルの型について

CPUにKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズを使用する場合、ラベルにはデータ形式(型)を指定します。



データ形式は、次の項目から選択します。

項　目		扱うデータの範囲
ビット	BOOL	0(OFF,FALSE),1(ON,TRUE)
1ワード符号なし整数	UINT	0～65535
2ワード符号なし整数	UDINT	0～4294967295
1ワード符号付き整数	INT	－32768～+32767
2ワード符号付き整数	DINT	－2147483648～+2147483647
単精度浮動小数点型実数	REAL	－3.4E+38≦N≦－1.4E－45 N＝0 1.4E－45≦N≦3.4E+38 (KV Nanoシリーズの場合、絶対値が1.4E－45～1.2E－38の範囲は使用できません。)
倍精度浮動小数点型実数※1	LREAL	－1.79E+308≦N≦－2.23E－308 N＝0 2.23E－308≦N≦1.79E+308
文字列	STRING	32文字以下(半角文字)
タイマ※2	―――	0(OFF,FALSE),1(ON,TRUE)
カウンタ※2	―――	0(OFF,FALSE),1(ON,TRUE)
配列	―――	[配列サイズ]ダイアログで指定します 「ラベル配列」(2-46ページ)

※1 ・「.DF」サフィックスは、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。
・「.DF」サフィックスは、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。
「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

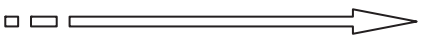
※2　ラベルに.Dサフィックスを付加した場合、現在値として扱われます。

●ラベルのサフィックスについて

ラベルは、ラベル登録時にラベル編集ウィンドウでデータの型(データ形式)を設定するため、スクリプトプログラムを記述するときはサフィックスを付加せずに記述できますが、他のデバイスと同様にサフィックスを付加した記述も可能です。

参考　TYPEによる型宣言もできます。ラベル編集ウィンドウで設定する「データ形式」と、TYPEによる「型宣言」、サフィックスの「直接指定」の関係は、以下の通りです。

データ形式　＜　型宣言　＜　直接指定



低　　優先順位　　高

ラベル登録時にラベル編集ウィンドウでデータの型(データ形式)により、指定できるサフィックスが制限されます。

項　目	グローバルラベル									ローカルラベル								
	.B	.U	.S	.D	.L	.F	.DF	.T	省略	.B	.U	.S	.D	.L	.F	.DF	.T	省略
ビット	○	×	×	×	×	×	×	×	.B	○	×	×	×	×	×	×	×	.B
1ワード符号なし整数	○	○	○	○	○	○	○	○	.U	○	○	○	×	×	×	×	×	.U
2ワード符号なし整数	○	○	○	○	○	○	○	○	.D	○	○	○	○	○	○	○	×	.D
1ワード符号付き整数	○	○	○	○	○	○	○	○	.S	○	○	○	×	×	×	×	×	.S
2ワード符号付き整数	○	○	○	○	○	○	○	○	.L	○	○	○	○	○	○	○	×	.L
単精度浮動小数点型実数	×	○	○	○	○	○	○	×	.F	×	○	○	○	○	○	○	×	.F
倍精度浮動小数点型実数	×	○	○	○	○	○	○	×	.DF	×	○	○	○	○	○	○	×	.DF
文字列	×	×	×	×	×	×	×	○	.T	×	×	×	×	×	×	×	○	.T
タイマ	○	○	○	○	○	×	×	×	.B	○	○	○	○	○	×	×	×	.B
カウンタ	○	○	○	○	○	×	×	×	.B	○	○	○	○	○	×	×	×	.B

○: サフィックスの指定可能(ただし、データ形式と異なるサフィックスを指定した場合、変換時に警告が出ます)

×: サフィックスの指定はできません

参 考　指定できないサフィックスを記述した場合、スクリプト変換時にエラーが発生します。

●グローバルラベルのビットについて

グローバルラベルのデータ形式にビットを指定した場合、割り付けるデバイスの種類により使用できるサフィックスが異なります。

割り付けるデバイス	指定できるサフィックス
R、B、MR、LR	.U/.S/.D/.L./.F/.DF/.T※/.B
ワードデバイスのビット指定 (DM1000.12など)	.B

※ .Tを指定するとき割り当てるビットデバイスは、チャンネルの先頭デバイスを指定してください。

●その他の注意事項

- グローバルラベルとローカルラベルで、同じラベル名が設定できますが、使用した場合、ローカルラベルとして動作します。
- ラベル名には、予約語など使用できない文字列があります。
  「使用できない文字一覧」(付-54ページ)
- 使用するCPUユニットがKV-8000/7500/7300/5500/5000/3000、KV　Nanoシリーズの場合、ラベルは全角文字/半角文字を区別しますが、大文字/小文字は区別しません。
- ローカルラベルは、インデックス修飾できません。
  「インデックス修飾」(3-55ページ)
- ラベル配列は、インデックス修飾や間接指定はできません。
  「ラベル配列」(2-46ページ)
- ラベルに定数を割り付けた場合、インデックス修飾や間接指定はできません。
- ラベルは、ワードデバイスのビット指定はできません。
  「ワードデバイスのビットデバイス扱い」(3-46ページ)

ラベル配列

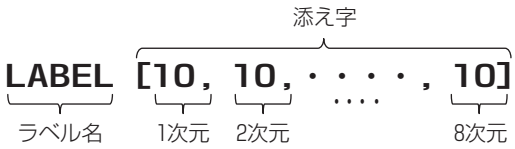
ラベル登録時、ラベル編集ウィンドウでデータ形式に[配列]を指定すると、ラベル配列が使用可能になります。

ポイント **使用するCPUがKV-1000の場合、ラベル配列(グローバルラベル配列、ローカルラベル配列)は使用できません。**

●配列

配列とは、同じ型のデータの連続的な集合体のことを指します。
KVスクリプトでは、「ラベル＋[添え字]」で配列を表現し、添え字をカンマで区切ることで多次元配列を実現します。
ラベル配列は、最大8次元まで指定可能です。

表記 ラベル配列は、次のように表記します。



参考 ラベル配列の添え字には、整数(10進数、16進数)のほか、整数型(.U/.S/.D/.L)のデバイスおよびマクロ引数(P0～P9,V0～V9)や、式、戻り値のある関数も記述できます。

例: LABEL[1, 2, DM0]
LABEL[1, DM2 * 3, Z01]

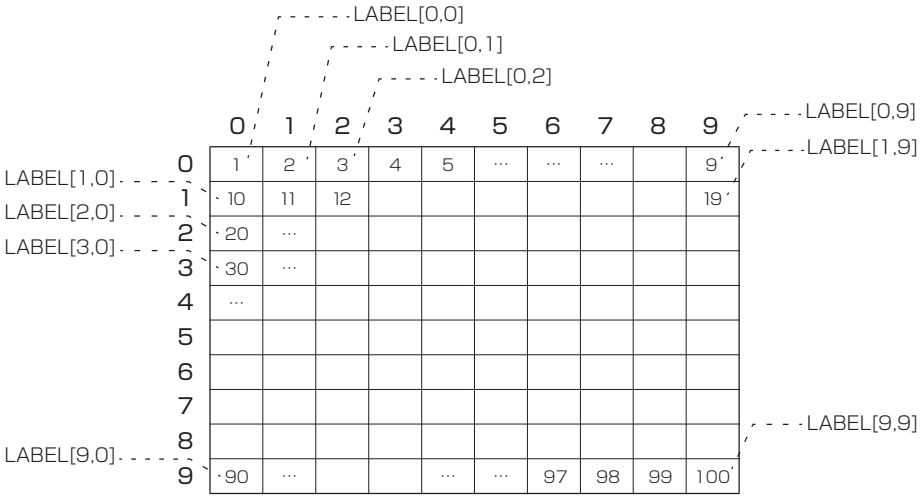
ポイント
- **ラベル配列は、インデックス修飾できません。**
- **ラベル配列に間接参照デバイスは指定できません。**
- **ラベル配列の添え字にデバイスや式を記述する場合、格納される値が添え字の範囲を超えないようにしてください。**

範囲

多次元配列	:最大8次元
添え字の定義範囲	:KV-5500/5000/3000、KV Nanoシリーズの場合:1～131071 KV-8000/7500/7300の場合:1～589823

例

ラベル編集ウィンドウで、ラベルの配列サイズを「10,10」と登録したときのデータの格納イメージを説明します。



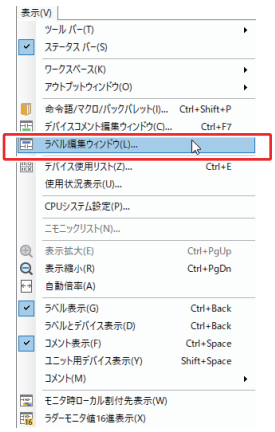
ラベル配列のデータ形式に、「1 ワード符号なし整数」または「1 ワード符号付き整数」を選択した場合、上記ラベル配列では、
10×10=100ワード
の領域をラベル配列として確保します。

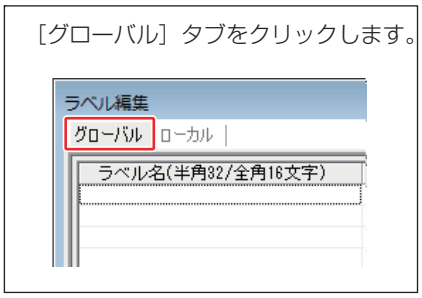
■グローバルラベル配列

グローバルラベル配列を登録するには、データ形式に［配列］を指定します。

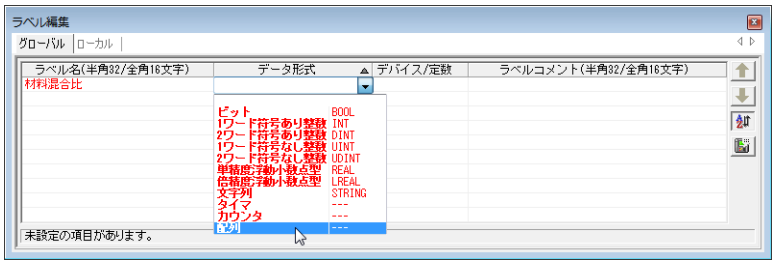
1 **メニューから【表示(V)】→【ラベル編集ウィンドウ(L)】を選択し、ラベル編集ウィンドウを表示させます。**

別手順
- ワークスペースの［ラベル］をダブルクリックする。
- スクリプト編集ウィンドウでツールバーをクリックする。



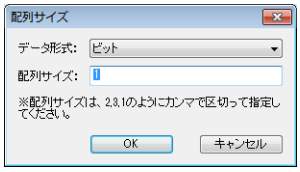


2 グローバルラベル配列名を登録し、データ形式に［配列］を選択します。



［配列］を選択すると、［配列サイズ］ダイアログが表示されます。

3 配列のデータ形式とサイズを設定します。



項　目	説　明
データ形式	ラベルの型を指定します。入力ボックスをクリックすると表示されるプルダウンメニューから選択します。
配列サイズ	• 配列のサイズを指定します。カンマ区切りで、最大8次元の配列が指定できます。 • サイズは、1～131071の範囲で指定してください。※1 • 配列の要素数が未確定の場合、「?」を代用数字として指定できます。※2

※1　割り当てるグローバルデバイスの範囲を超えないようにしてください。

※2　「?」のまま変換すると、エラーが発生します。

参 考　データ形式は、以下の項目から選択します。

項　目	扱うデータの範囲
ビット	0(OFF,FALSE),1(ON,TRUE)
1ワード符号なし整数	0～65535
2ワード符号なし整数	0～4294967295
1ワード符号付き整数	－32768～+32767
2ワード符号付き整数	－2147483648～+2147483647
単精度浮動小数点型	－3.4E+38≦N≦－1.4E－45 N＝0 1.4E－45≦N≦3.4E+38 (KV Nanoシリーズの場合、絶対値が1.4E－45～1.2E－38の範囲は使用できません。)
倍精度浮動小数点型	－1.79E+308≦N≦－2.23E－308 N＝0 2.23E－308≦N≦1.79E+308

4 割り当てるグローバルデバイスを指定します。

ポイント　グローバルラベル配列は、配列で指定したサイズのグローバルデバイスを占有します。

■ローカルラベル配列

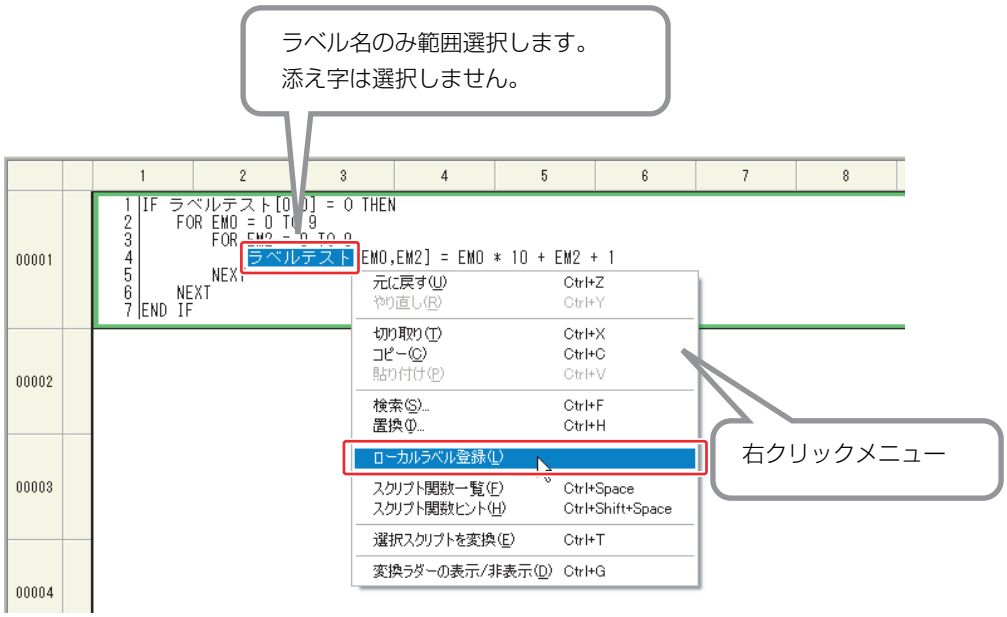
ローカルラベル配列は、グローバルラベル配列と同様にあらかじめラベルを登録する方法と、スクリプトプログラム記述後に登録する方法があります。
ここでは、スクリプトプログラム記述後にローカルラベル配列を登録する方法について説明します。

1 ローカルラベル配列を使用して、スクリプトを記述します。

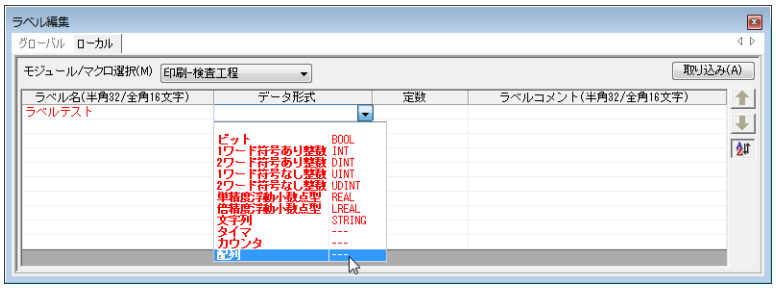
```
00001
1 IF ラベルテスト[0,0] = 0 THEN
2     FOR EM0 = 0 TO 9
3         FOR EM2 = 0 TO 9
4             ラベルテスト[EM0,EM2] = EM0 * 10 + EM2 + 1
5         NEXT
6     NEXT
7 END IF
```

ラベルは登録されていないので、文字色が変わりません。

2 スクリプトの編集状態のまま、ローカルラベル配列として登録したいラベル名を範囲選択し、右クリックメニューの【ローカルラベル登録(L)】を選択します。

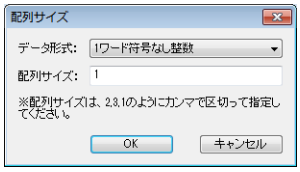


3 ラベル編集ウィンドウで、ローカルラベル配列を設定します。



[配列]を選択すると、[配列サイズ]ダイアログが表示されます。

4 配列のデータ形式とサイズを設定します。



項　目	説　明
データ形式	ローカルラベル配列の型を指定します。入力ボックスをクリックすると表示されるプルダウンメニューから選択します。
配列サイズ※	• 配列のサイズを指定します。カンマ区切りで、最大8次元の配列が指定できます。 • 配列の要素数が未確定の場合、「?」を代用数字として指定できます。

※ 割り当てるワークメモリの範囲を超えないようにしてください。

「ワークメモリの使用領域」(2-74ページ)

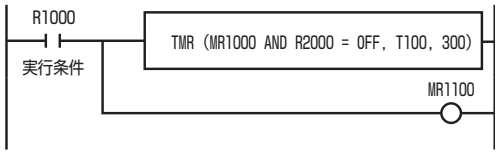
参 考　データ形式は、以下の項目から選択します。

項　目	扱うデータの範囲
ビット	0(OFF,FALSE),1(ON,TRUE)
1ワード符号なし整数	0～65535
2ワード符号なし整数	0～4294967295
1ワード符号付き整数	－32768～+32767
2ワード符号付き整数	－2147483648～+2147483647
単精度浮動小数点型	－3.4E+38≦N≦－1.4E－45 N＝0 1.4E－45≦N≦3.4E+38 (KV Nanoシリーズの場合、絶対値が1.4E－45～1.2E－38の範囲は使用できません。)
倍精度浮動小数点型	－1.79E+308 ≦N≦－2.23E－308 N＝0 2.23E－308 ≦N≦1.79E+308

スクリプトを記述するときの注意事項

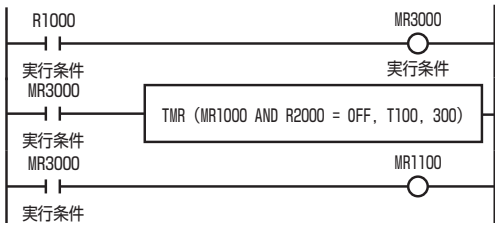
ボックススクリプトの並列接続について

ボックススクリプトは、ラダー命令をOR接続できません。
右のような記述は、変換時にエラーが発生します。



●OR接続と等価なラダープログラム

いったん別のデバイス(右の場合は、MR3000)を経由して、論理的にOR接続しているように記述してください。



スクリプトで演算式を記述するときの注意事項

スクリプトでは、演算式は全て.L(32ビット符号付き)、.F(単精度浮動小数点型)もしくは.DF(倍精度浮動小数点型)として扱います。

- 演算式の中の演算要素に.L.F.DF以外(.U .D .S)の型を指定しても、すべて.Lで扱われます。演算結果も.Lで求められます。
- 演算式内の演算要素に.Fが含まれる場合、.Fの演算要素が出現するまでは.Lとして演算し、.Fの演算が始まると以降はすべて.Fとして扱われ、演算結果も.F(単精度浮動小数点型)で求められます。
- 演算式内の演算要素に.DFが含まれる場合、.DFの演算要素が出現するまでは.Lまたは.Fとして演算し、.DFの演算が始まると以降はすべて.DFとして扱われ、演算結果も.DF(倍精度浮動小数点型)で求められます。

ポイント

- **演算式は.L(32ビット符号付き)の範囲内(−2147483648〜+2147483647)で処理されます。.Lの範囲を超える演算をおこなう場合は注意が必要です。**
- **演算式は、演算要素を全て.Lに拡張して演算するため、式に.D(32ビット符号無し)が含まれていると、精度落ちが発生する場合があります。**
  「型変換によるデータの欠落」(2-52ページ)
- **精度落ちが発生した場合「警告」を表示する設定も可能です。**
  「警告レベルの設定方法」(2-53ページ)

スクリプト変換時のメッセージ

スクリプトプログラムは、いったんラダープログラムに変換されてからCPUユニットに転送されます。スクリプトプログラムをラダープログラムに変換するときに表示される、警告メッセージとエラーメッセージについて説明します。

警告メッセージ

スクリプトプログラムからラダープログラムに変換はできますが、実行すると期待する演算結果が得られない場合などに表示されるメッセージです。できるかぎり警告の発生原因は取り除くことをお勧めします。
代入文で右辺の結果の型と左辺の型が異なっていたり、演算式に.Dが含まれたり、現在の機種では実行できない関数であったりする場合などに発生します。

■型変換によるデータの欠落

以下のスクリプトではデータが欠落し、必要なデータを得られない可能性があります。

スクリプト記述 DM1000.U = EM2000.S ＊ 10000

- 例えばEM2000の値が－1の場合、DM1000の型は16ビット符号無しなので正しい演算結果を格納できません。また、EM2000の値が10の場合、演算結果が16ビット符号無しの範囲(0～65535)を超えるので、期待する(正しい)演算結果を代入できません。

上記スクリプトを変換すると次のメッセージがアウトプットウィンドウに表示されます(スクリプト警告レベルの設定を「高」にした場合)。

[警告1103]："＝"：(符号付き32ビット整数型－>符号無し16ビット整数型)　代入によりデータが欠落する可能性があります

この場合、左辺にあるデバイスのサフィックスを.Uから.Lに変更する(左辺の型と右辺の型をあわせる)ことで警告メッセージが表示されなくなります。

「スクリプトを記述するときの注意事項」(2-51ページ)

■現在の機種では実行できない関数

以下の機種に対して、関数を実行した場合実行できないため警告が発生します。

警告表示の対象となる関数一覧	KV-N14□□	KV-N24□□ KV-N40□□ KV-N60□□	KV-NC32T
WTIME/AJST	警告	－	－
MWRIT/MREAD/MFREE/MFREEK/ MMKDIR/MRMDIR/MDEL/MPRINT/ MREADL/MCOPY/MMOV/MREN/MSTAT	警告	警告	－
AWNUM/AWMSG	－	－	警告

変換すると次のメッセージがアウトプットウィンドウに表示されます(スクリプト警告レベルの設定を「高」にした場合)。

[警告1008]："MWRIT"：現在の機種では実行できない関数を使用しています。

また、KV Nanoシリーズでは内蔵位置決めの軸数、内蔵の高速カウンタ番号がユニットにより異なります。使用できない軸番号や、カウンタ番号を指定した関数がある場合、変換時に警告が発生します。
KV Nanoシリーズで使用できる軸番号や高速カウンタ番号については『KV Nanoシリーズユーザーズマニュアル』を参照してください。

スクリプト警告レベルの設定

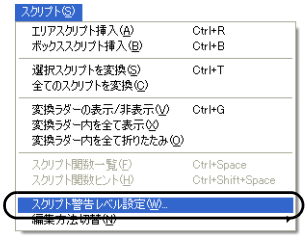
スクリプト変換時にアウトプットウィンドウに表示される「警告」について、表示レベルを変更します。

■警告レベルについて

警告にはその内容により、「強い警告」と「弱い警告」の2種類あります。

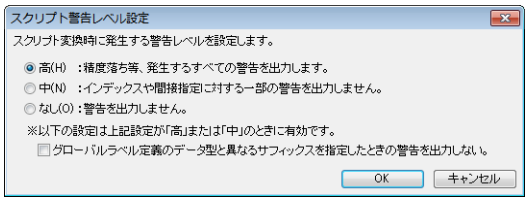
●警告レベルの設定方法

1 **メニューから【スクリプト(S)】→【スクリプト警告レベル設定(W)】を選択します。**



別手順　メニューから【ツール(T)】→【オプション(O)】、スクリプトのタブの[警告レベル]から[高]・[中]・[なし]のいずれかを選択。

2 **スクリプト警告レベルダイアログが表示されますので、警告レベルを選択してください。**



項目	内容
高	すべての警告(強い警告+弱い警告)を表示します。 例)演算式などで精度落ちなどが発生する場合 ・DM1000.U = DM2000.L(.L型のデータを.U型のデバイスや変数に代入しようとした)
中	強い警告(範囲外の定数を扱おうとした場合など)だけを表示します。 例)定数同士の演算結果が定数の範囲を超えてしまう場合 ・DM1000.D= 90000 * 50000 (定数の範囲外の値(4500000000)を代入しようとした)
なし	警告は表示しません。
グローバルラベル定義のデータ型と異なるサフィックスを指定したときの警告を出力しない。	チェックボックスにチェックを入れると、グローバルラベル登録時に定義したデータの型と異なるサフィックスを指定したときに警告を出力しなくなります。 例)グローバルラベル定義で「1ワード符号無し整数:UINT」を指定し、スクリプトで.Lサフィックスを指定した場合など

エラーメッセージ

スクリプトプログラム内に、ラダープログラムに変換できない記述があった場合、エラーが発生しエラー要因をアウトプットウィンドウに表示します。
エラー要因を取り除き変換に成功しないと、プログラムはCPUユニットに転送できません。アウトプットウィンドウに表示されているエラーメッセージを元に、スクリプトを修正してください。

■スクリプト変換時のエラー

次のような原因が考えられます。

●タイプミスをしている

- 関数名を間違えている。
  例)　正→BLD (DM1000, 10)　　誤→BLF (DM1000, 10)
- デバイス名を間違えている。
  例)　正→DM1000.D　　誤→DN1000.D
- 2バイト文字を使っている(コメントと文字定数には2バイト文字が使用できます)。
  例)　正→DM1000.D　　誤→DM1000.D
- 0(ゼロ)とO(オー)の間違い。
- 半角スペースと全角スペースの間違い。
  例)　正→END IF(間が半角スペース)　　誤→END　IF(間が全角スペース)

など。

●文法を間違えている

- 制御文の書式を間違えている。
  例)　正→IF R1000 THEN　　誤→IF R1000 THEN
  　　DM1000 += 1　　DM1000 += 1
  　　END IF　　END␣␣␣
- BREAKが繰り返し制御文の外に記述されている。
- DOに対するUNTILが無い。
- その他スクリプトの文法間違い全般

●関数引数間違い

- 引数の数を間違えている。
  例)　正→FIND (DM1000.T, DM1200.T)　　誤→FIND (DM1000.T, DM1200.T, 8)
- 呼び出すマクロ自体が記述されていない。

など。

●不正な型指定

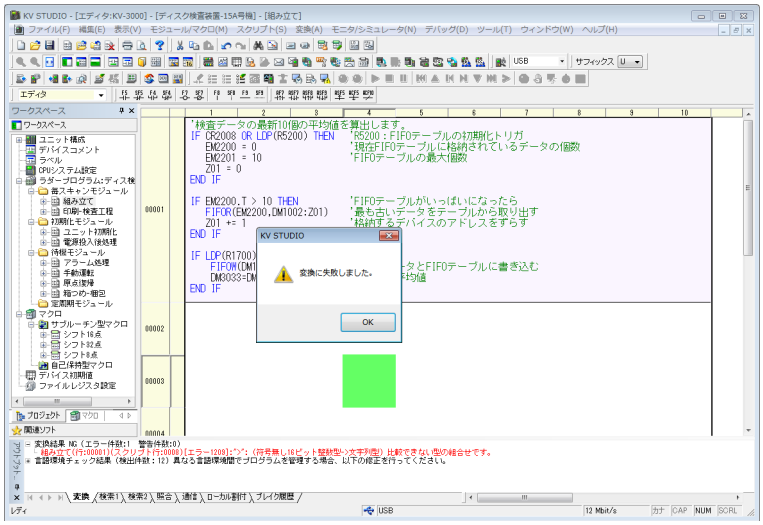
- STRING型以外の変数や.Tをつけていないデバイスに文字列を代入しようとした。
  例)　正→DM1000.T = "ABCD"　　誤→DM1000 = "ABCD"
- 文字列で指定しなければならない引数に、ビットデータやワードデータを指定した。
- ビットデータに数値定数を代入しようとした。
- その他不正な型指定全般。
- ローカルラベルに登録時に定義したデータ形式よりも大きなサイズの型を指定した。
- KV-1000のときに.DFの型を指定した。

●その他

- 存在しないデバイスを指定した。
- スクリプトの1行が(半角)1024文字を超えている。
- 配列として登録されていないラベルに、配列の添え字を指定した。

変換エラーが発生したときは

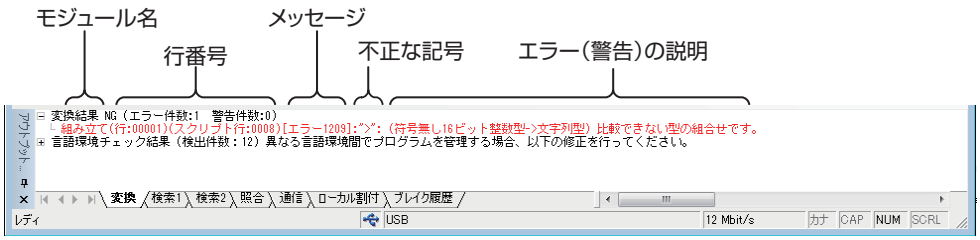
関数名のタイプミスや、不正な型(サフィックス)指定、制御文の使い方を間違えるなど、スクリプトプログラムの記述が不正な場合、スクリプトを変換すると「スクリプト変換エラー」が発生し、ダイアログボックスが表示されます。



変換エラーが発生すると、アウトプットウィンドウにエラー発生した場所とエラーの内容が表示されます。

■アウトプットウィンドウのエラー表示

スクリプト変換エラーが発生すると、次のようにエラー内容が表示されます。



メッセージが[エラー]の場合:ラダープログラムに変換不可能な状態です。
この状態ではプログラムをPLCに転送できません。

メッセージが[警告]の場合 :ラダープログラムに変換されます。プログラムを PLC に転送して実行したり、シミュレータで実行可能な状態です。
ただし、プログラムを実行すると、期待する演算結果が得られない恐れがあります。

■アウトプットウィンドウの使いかた

変換エラーが発生すると、アウトプットウィンドウにエラー(警告)内容が表示されます。
表示された行をダブルクリックすると、ラダー編集ウィンドウのエラー行にカーソルが移動します。
エラー内容を元に修正してください。

1 編集したスクリプトを変換します。

メニューから【スクリプト(S)】→【選択スクリプトを変換(S)】を選択します。

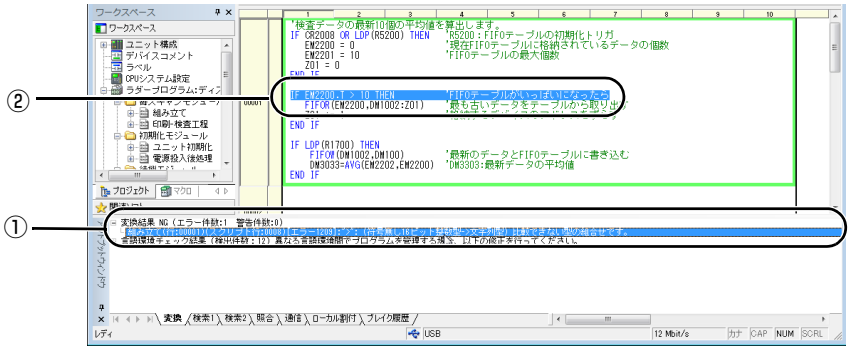
別手順
- Ctrl キー＋ T キー
- 右クリックメニューから【スクリプト編集(X)】→【選択スクリプトを変換(S)】

変換エラーが発生した場合、右のダイアログボックスが表示され、アウトプットウィンドウにエラーメッセージが表示されます。

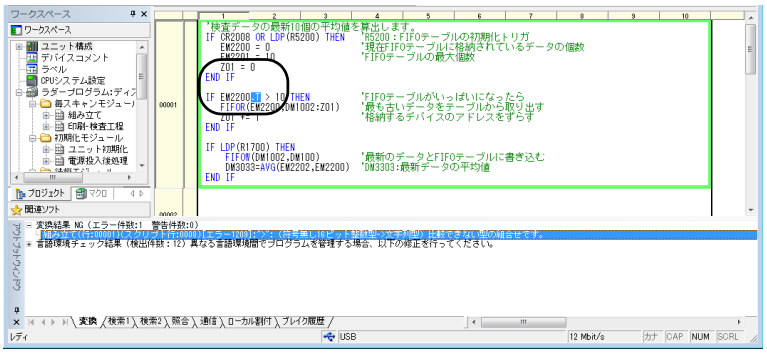
〔OK〕ボタンをクリックします。

2 アウトプットウィンドウのエラー内容(①)をダブルクリックします。

対応するエラー記述のある行(②)にカーソルが移動(フォーカス移動)し、スクリプトが編集可能状態になります。



3 エラー記述を修正します。



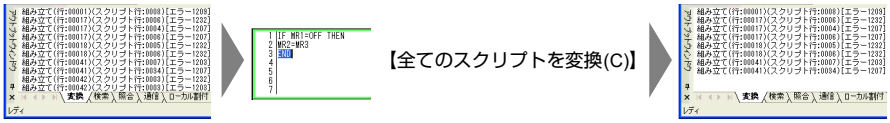
4 **修正後、修正したスクリプトだけを変換します。**
メニューから【スクリプト(S)】→【選択スクリプトを変換(S)】を選択します。

別手順
- Ctrl キー＋ T キー
- 右クリックメニューから【スクリプト編集(X)】→【選択スクリプトを変換(S)】

アウトプットウィンドウに、エラーが表示されなければ修正完了です。

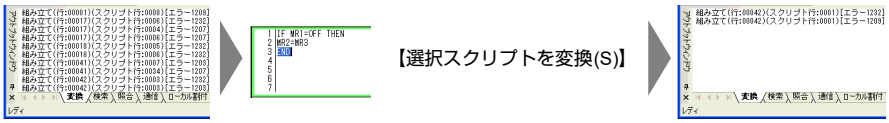
参 考
- エラーやワーニングが大量に発生している状態で、エラーを1つだけ修正してから【全てのスクリプトを変換(C)】を選択すると、他の修正しなかったエラーが再度大量に表示されます。

大量のワーニング　　1つだけ修正　　大量のワーニング



このようなときは、メニューから【選択スクリプトを変換(S)】を選択します。

大量のワーニング　　1つだけ修正



修正したスクリプトだけが変換されますので、そのスクリプトに残っているエラーだけが表示され、正しく修正できたか確認しやすくなります。
- 変換ラダーの表示 / 非表示 (Ctrl+G) でスクリプトを編集したまま、変換ラダーを表示して確認したり、展開ラダーを閉じたりできます。

ファンクションブロック

ファンクションブロックはプロジェクト内で何度も繰り返して使用するラダー回路に、任意の名前を付けて登録し、共通化したものです。
登録されたファンクションブロックはスクリプトプログラムから呼び出すこともできますし、ファンクションブロック内でスクリプトで使用することも可能です。

ファンクションブロックの呼び出し

■ファンクションブロックの呼び出し

ファンクションブロックはFB/FBCALL/FBSTRT関数を使用して呼び出すことができます。ファンクションブロックを停止するには、以下の手順を実行します。

- 呼び出し元のプログラムを停止する(FB/FBSTRT/FBCALL)
- ファンクションブロック内でFEND関数(命令)を実行する(FB/FBSTRT)
- ファンクションブロックの実行条件をOFFする(FBCALL)

「FB関数」(6-10ページ)
「FBCALL関数」(6-11ページ)
「FBSTRT関数」(6-13ページ)
「FEND関数」(6-15ページ)

●使用方法

スクリプト記述 FB(MR1000,"FunctionBlock",DM0,#0,MR2000)
スクリプト記述 FBSTRT(MR1000, "FunctionBlock", Instance, DM0, #0, MR2000)
スクリプト記述 FBCALL(MR1000, "FunctionBlock", Instance, DM0, #0, MR2000)

ポイント **FBCALL/FBSTRT関数を使用する場合はKV-8000のCPU機能バージョンを2.0以降にアップデートしてください。**

■ファンクションの呼び出し

ファンクションはFUN命令を使用して呼び出すことができます。ファンクションの実行条件をOFFすると、ファンクションは停止します。

●使用方法

スクリプト記述 FUN(MR1000,"Function",DM0,#0,MR2000)

参考 CPU機能バージョン2.0以降のKV-8000では、FUN関数を演算式に記述することで、戻り値を扱えます。
以下は、Function1の戻り値をDM0に格納する例です。
スクリプト記述 DM100 = FUN(MR1000, "Function1", DM0, DM1)

ファンクションブロックをスクリプトで記述する

新規作成するファンクションブロックは、ラダープログラムやスクリプトを混在して記述することやスクリプトだけで作成することもできます。
ファンクションブロックの作成方法については、使用しているCPUユニットのユーザーズマニュアルを参照してください。

ポイント **END、ENDH命令はスクリプトでは記載できません。**

マクロ

プロジェクト内で何度も繰り返して使用するラダー回路に、任意の名前を付けて登録したものをマクロと呼びます。
登録されたマクロはスクリプトプログラムから呼び出すこともできますし、マクロをスクリプトで作成することも可能です。

マクロの呼び出し

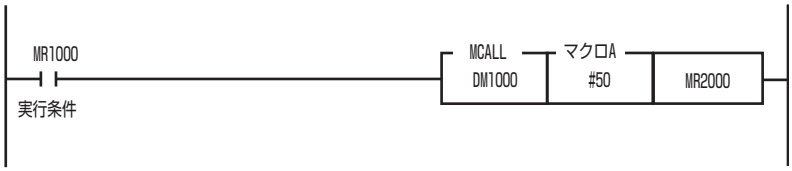
■サブルーチン型マクロの呼び出し

サブルーチン型として登録されているマクロは、MCALL関数を使用して呼び出すことができます。
「MCALL関数」(6-6ページ)

●使用方法

スクリプト記述　MCALL(MR1000,"マクロA",DM1000,#50,MR2000)

ラダー記述



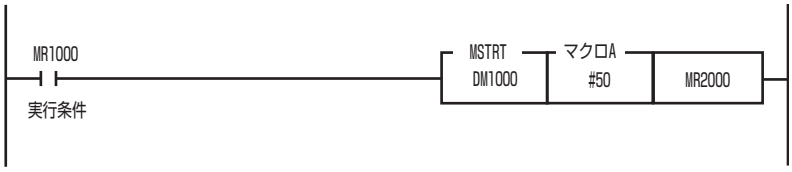
■自己保持型マクロの呼び出し

自己保持型として登録されているマクロは、MSTRT関数を使用して呼び出すことができます。このとき、呼び出すマクロの中にはMEND命令(関数)が必要です。
「MSTRT関数」(6-7ページ)
「MEND関数」(6-8ページ)

●使用方法

スクリプト記述　MSTRT(MR1000,"マクロA",DM1000,#50,MR2000)

ラダー記述



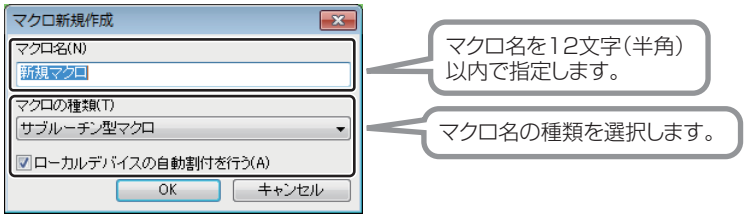
マクロをスクリプトで記述する

新規作成するマクロは、ラダープログラムにスクリプトを混在させて記述することや、スクリプトだけで作成することもできます。

1 メニューから【モジュール/マクロ(M)】→【マクロの新規作成(M)】を選択します。

別手順
- アイコンをクリック
- 右クリックメニューから【マクロ編集(M)】→【マクロの新規作成(N)】

2 マクロ名とマクロの種類を指定します。

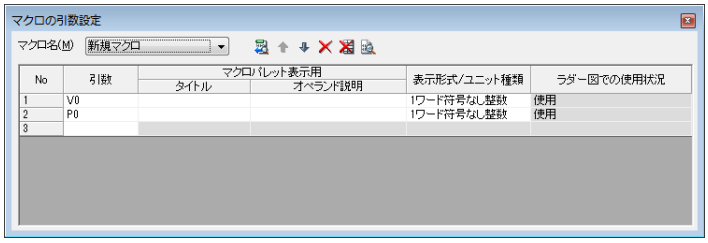


ポイント
マクロ名に「+」「-」が含まれている場合、スクリプトでは「マクロ実行中ビット」が記述できません(スクリプトでは、+-を演算子として認識してしまうため)。
「マクロ実行中ビット」の詳細は使用しているCPUユニットのユーザーズマニュアルを参照してください。

参考
- マクロ名には使用できない文字があります。
- マクロの種類はマクロ作成後でも変更できます。
- ローカルデバイスを手動で割り付ける場合は、チェックボックスをOFFにしてください。

3 マクロの引数を設定します。

[マクロの引数設定]ダイアログで、登録するマクロで使用する引数を設定します。



4 作成したマクロにプログラムを記述します。

[ワークスペース]に新しいマクロが追加されます。マクロに記述するプログラムは、ラダーのみ・スクリプトのみ・ラダーとスクリプトの混在のいずれでも可能です。

参考
スクリプトプログラム内にユニットデバイスが含まれる場合やマクロ引数デバイス設定については、使用しているCPUユニットのユーザーズマニュアルの「マクロ」を参照してください。

ポイント
スクリプトには、END命令・ENDH命令に相当する関数がありません。END・ENDH命令をラダーで記述してください。

スクリプトのデバッグ機能

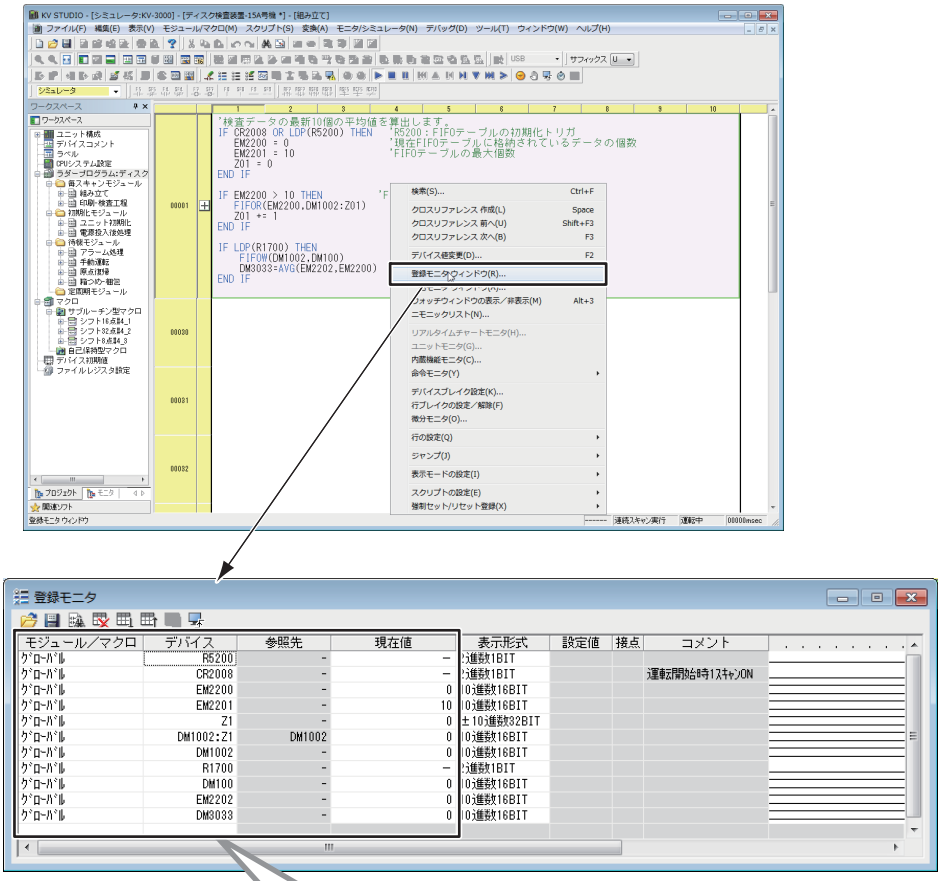
モニタ機能

オンライン中もしくは、シミュレータ起動中にモニタ機能を利用して、スクリプトプログラムのデバッグをおこないます。

■登録モニタ

スクリプトに記述されている各デバイスや変数の現在値がモニタできます。

1 **KV STUDIOがモニタ状態、もしくはシミュレータ状態で、モニタしたいスクリプトをクリックし、右クリックメニューから［登録モニタ］を選択します。**



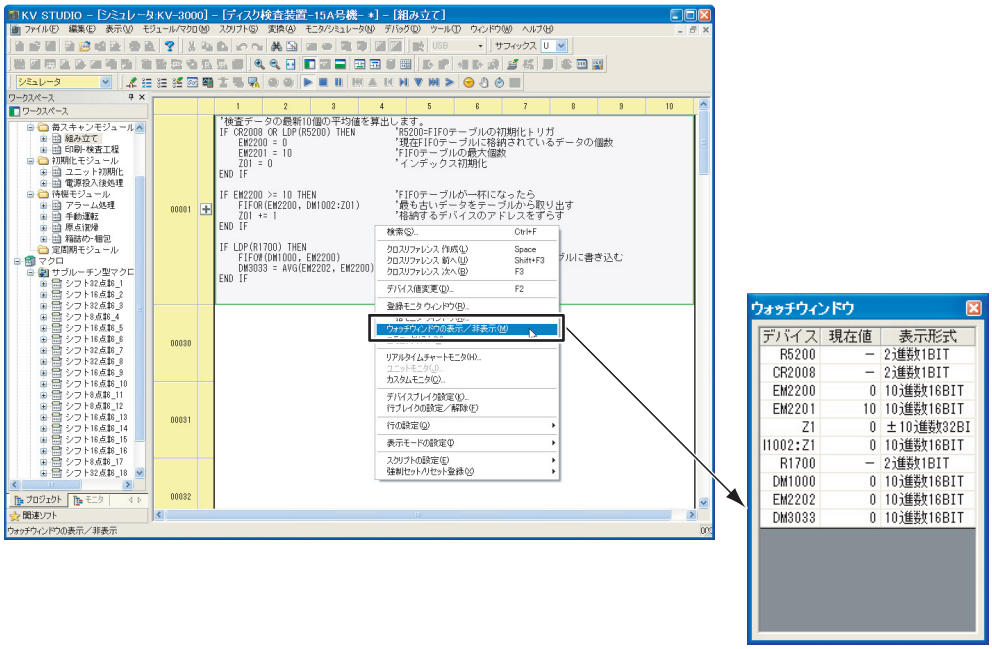
スクリプトに記述されている、すべてのデバイスや変数が登録モニタウィンドウに登録されて、現在値やビット状態がモニタできます。
設定値やコメント、タイミングチャートもモニタできます。

■ウォッチウィンドウ

ウォッチウィンドウは、スクリプトを選択するだけで、スクリプト内のすべてのデバイスや変数の現在値をモニタできます。

1 **KV STUDIOがモニタ状態、もしくはシミュレータ状態で、モニタしたいスクリプトをクリックし、右クリックメニューから［ウォッチウィンドウ］を選択します。**

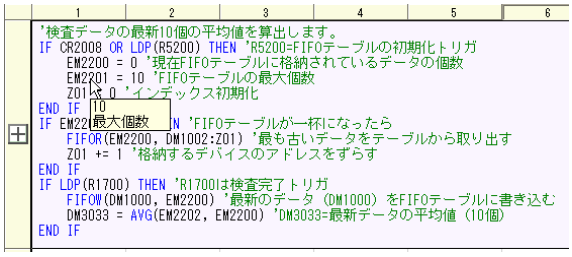
別手順



別のスクリプト記述を選択すると、ウォッチウィンドウに表示されている変数やデバイスが自動的に切り換わります。

■ツールチップによるモニタ

KV STUDIOがモニタ状態、もしくはシミュレータ状態で、モニタしたいデバイスにマウスカーソルをあわせるとツールチップで現在値とデバイスコメント（変数の場合は変数コメント）を表示します。



入力補助機能 ・ その他

入力補助機能について

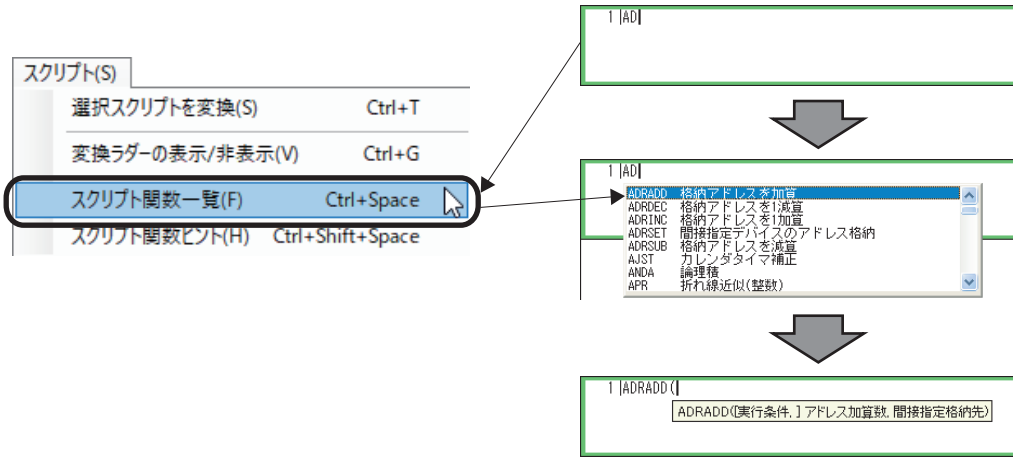
KV STUDIOには、スクリプトの記述をサポートするために、様々な便利機能や入力補助機能が存在します。

スクリプト関数一覧

関数名そのものがわからなくなった場合、関数を途中まで入力した後に、次のいずれかの方法で関数候補を呼び出すことができます。

●関数一覧の呼び出し

メニューから【スクリプト(S)】→【スクリプト関数一覧(F)】を選択します。



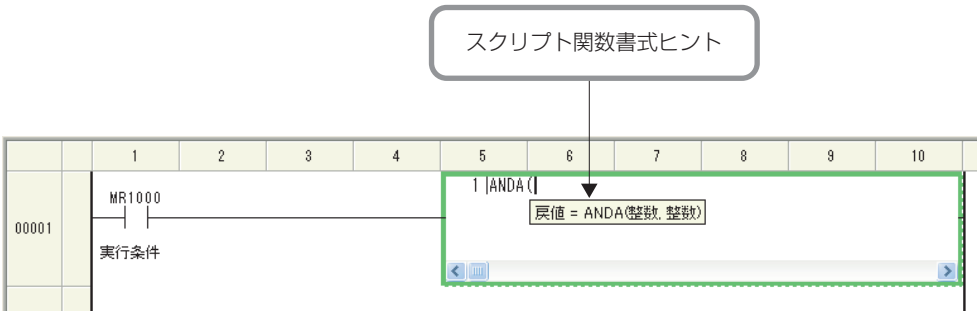
使用する関数を選択し↵キーを押すか、ダブルクリックすると、スクリプトエディタのカーソルがある場所に選択した関数が挿入され、スクリプト関数書式ヒントが表示されます。

別手順
- Ctrl キー＋ Space キー
- 右クリックメニューから【スクリプト関数一覧(F)】

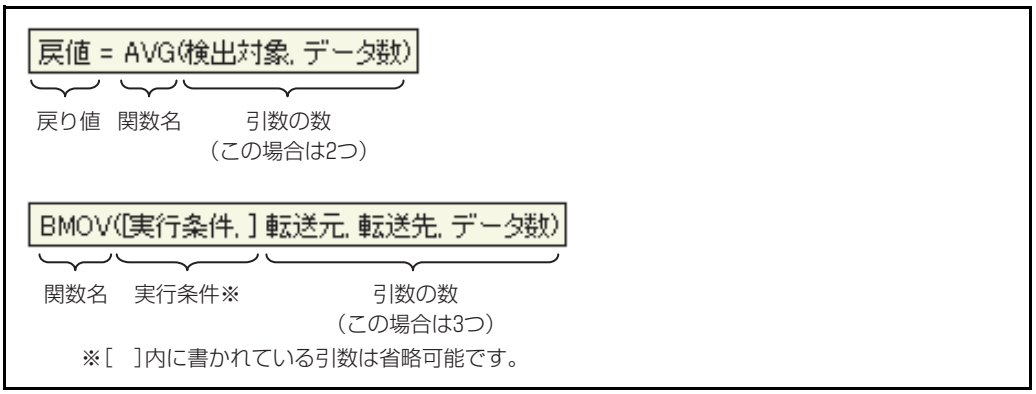
参 考
- 何も入力していない状態で、スクリプト関数候補を呼び出すこともできます。その場合アルファベット順にすべての関数候補が表示されます。
- 「スクリプト関数候補」はスクリプト記述中のみ呼び出すことができます。

スクリプト関数書式ヒント

関数を入力する時、引数や戻り値の有り無しなどの情報を、簡易的に「スクリプト関数書式ヒント」が教えてくれます。

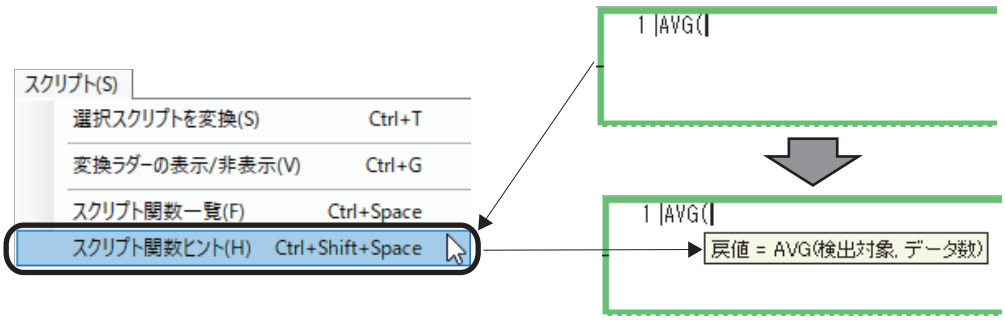


「関数名(」まで入力すると、自動的に「スクリプト関数書式ヒント」が表示されます。



●スクリプト関数ヒントの呼び出し

カーソル「Ｉ」を「(」の右側に置き、メニューから【スクリプト(S)】→【スクリプト関数ヒント(H)】を選択します。



別手順 Ctrl キー+ Shift キー+ Space キー

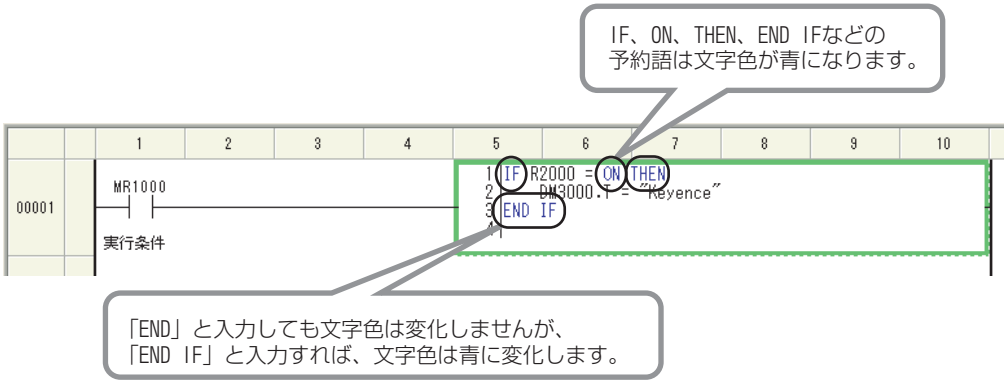
参 考 「スクリプト関数ヒント」はスクリプト記述中のみ呼び出すことができます。

ポイント **「スクリプト関数ヒント」を呼び出すときは、必ずカーソルを「(」の右側か、関数の先頭に置いてください。他の位置にカーソルがあるとスクリプト関数ヒントは呼び出されません。**

予約語

制御文や関数など、KVスクリプトに登録されている単語(予約語)については、記述時に文字色が青になります。文字色が変化しなかった場合、入力が間違っている可能性があります。

参 考　予約語の文字色は、KV STUDIOの[オプション設定]ダイアログで変更可能です。



HELP機能

関数の書式や使い方、制御文の説明や演算子の詳細は「スクリプト関数ヘルプ」で調べることができます。

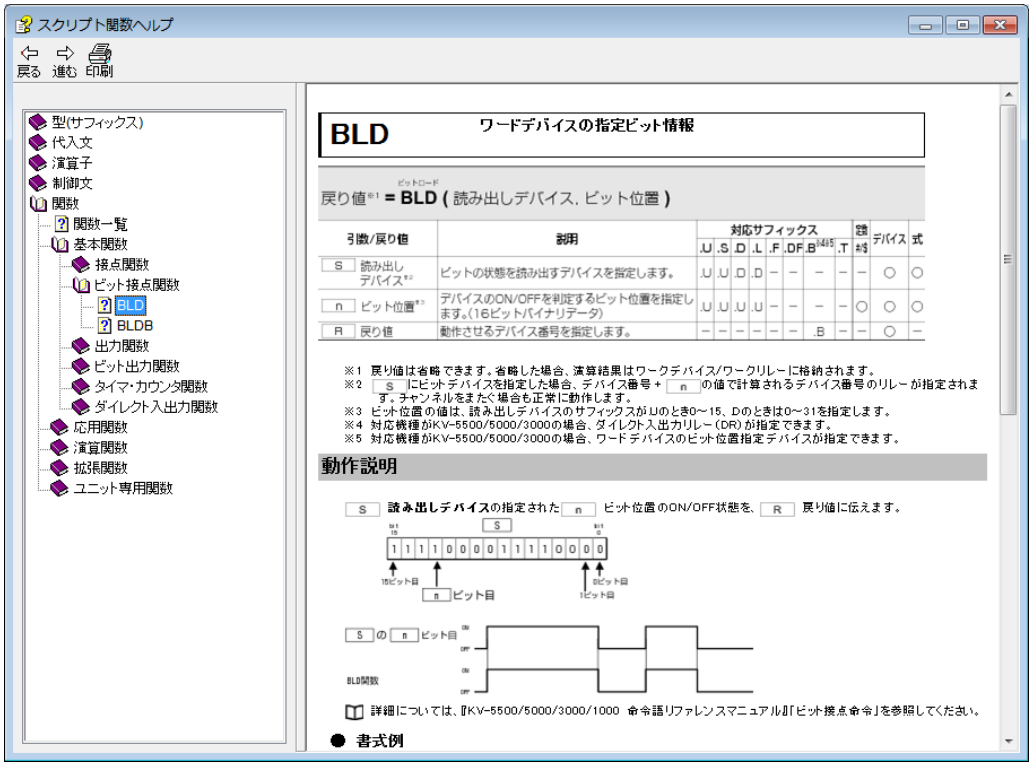
■スクリプト関数ヘルプを起動する

1 **メニューから【ヘルプ(H)】→【スクリプト関数ヘルプ(L)】を選択します。**

「スクリプト関数ヘルプ」の目次が表示されます。

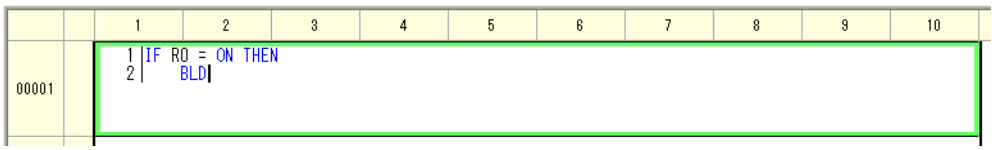
2 **調べたい関数を選択します。**

関数の詳細な説明が表示されます。

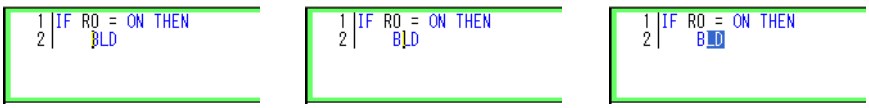


■スクリプト編集画面上で指定した関数のヘルプを表示する

1 **調べたい関数にカーソルをあわせます。**

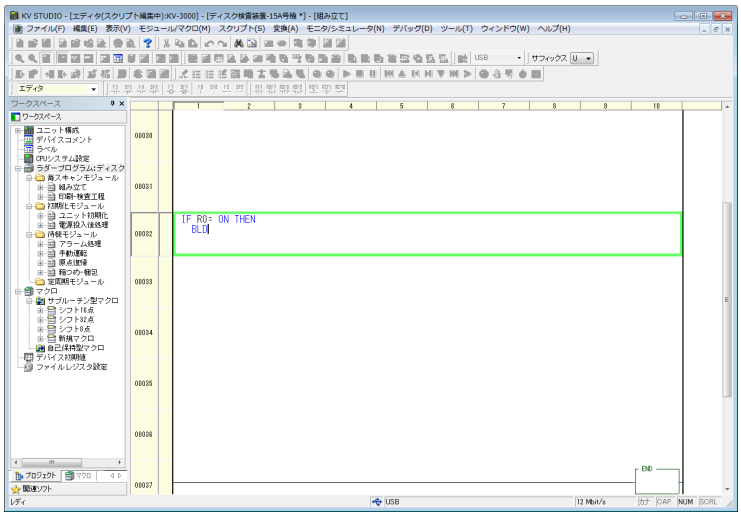


カーソルの位置は、下図のように関数の前後や中、または関数の一部でも範囲選択した状態にしてください。



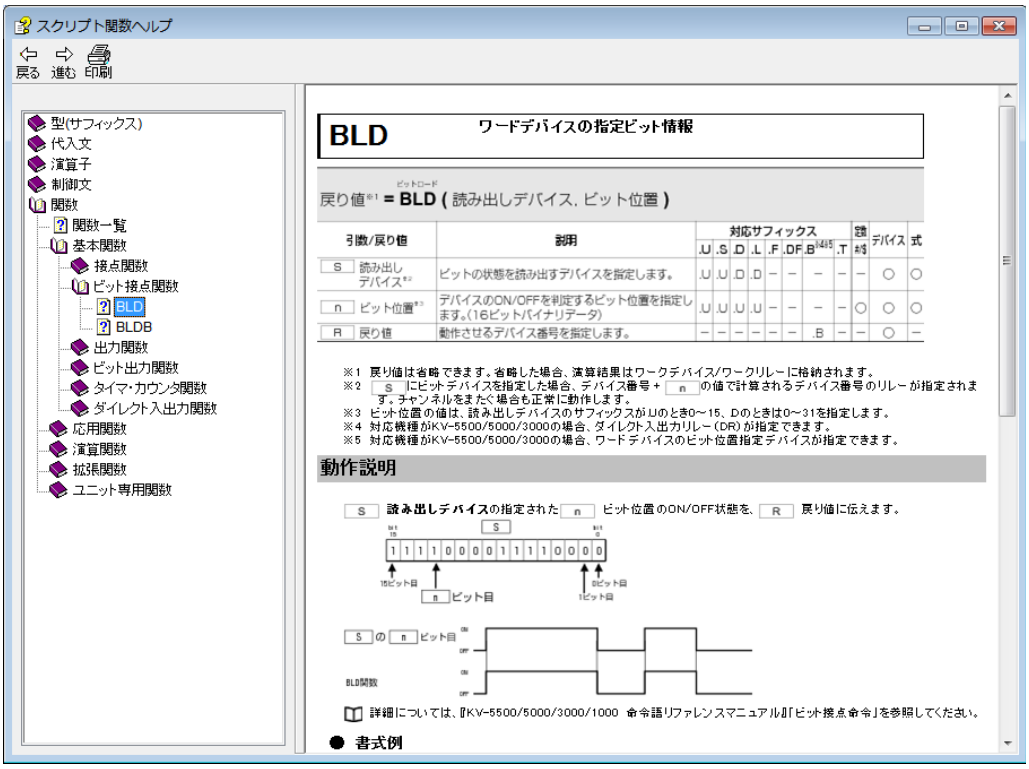
入力補助機能・その他

2 ボタンをクリックします。



別手順　関数を選択して、F1 キーを押す。

選択した関数のヘルプ画面が表示されます。

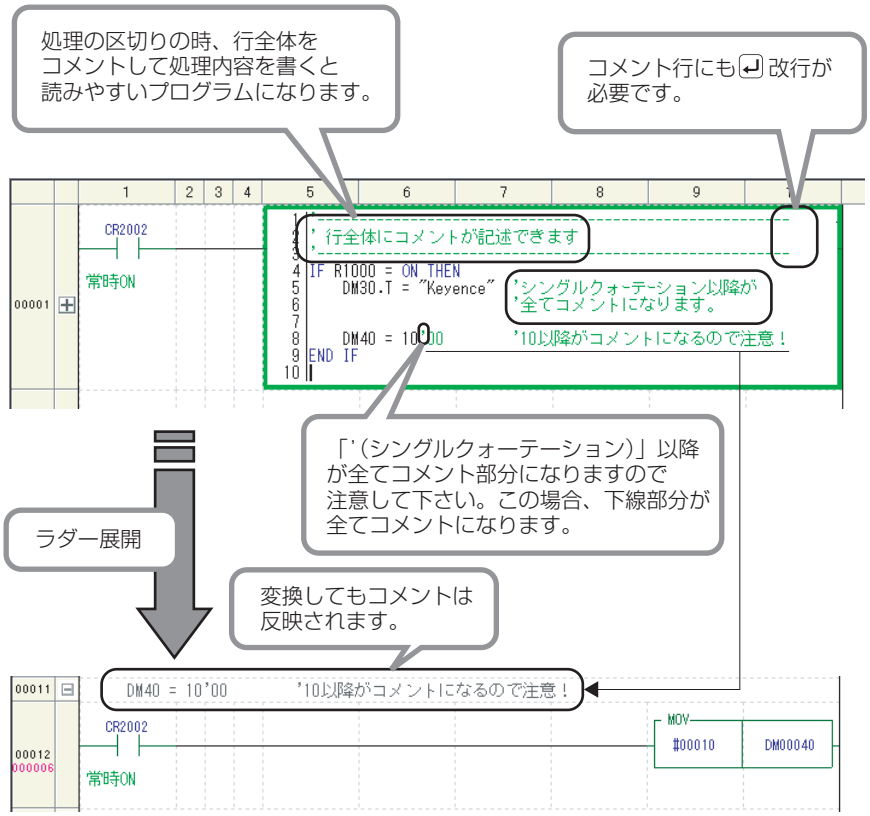


コメント機能について

KVスクリプトでは、スクリプトプログラムの任意の場所に「' (シングルコーテーション)」(もしくは「REM」)から「改行(リターン入力)」までをコメントとして記述できます。コメントの記述は、効率の良いデバッグやプログラムの改造をおこなう上で効果的です。スクリプトとあわせて積極的に記述されることをお勧めします。

●コメントの書き方

コメントは、文(式)の右側や行全体に記述できます。



参 考

- コメント行が増えてもプログラムのサイズやスキャンタイムには影響しません。
- 一行に入力できる文字数はスクリプト、コメントを含み1024文字(半角)までです。
- 以下の方法で選択している複数の行をまとめてコメント行にできます(KV STUDIO Ver.11.30以降)。
  - 右クリックメニューの【行の無効化-設定(Q)】
  - メニューの【ST/スクリプト(S)】→【行の無効化-設定(Q)】
  - Ctrl + Q
- 以下の方法で選択している複数の行のコメントを解除できます(KV STUDIO Ver.11.30以降)。
  - 右クリックメニューの【行の無効化-解除(W)】
  - メニューの【ST/スクリプト(S)】→【行の無効化-解除(W)】
  - Ctrl + W

見やすいスクリプトを書く

スクリプトプログラムを作成するとき、効率よくデバッグをおこなうためには、第三者にとって見やすく記述することを推奨します。
また、ラダープログラムと同様に、でき上がったスクリプトプログラムは、改造やメンテナンスあるいは再利用の際に他の人の目に触れます。ですから、できるだけわかりやすく且つ見やすく(美しく)書く必要があります。
そうすることで、プログラマからの手離れもよく、効率的に運用できます。

●サンプルプログラム

わかりにくい記述

```
IF CR2008 OR LDP(R5200) THEN
EM2200=0
EM2201=10
Z01=0
END IF
IF EM2200>=10 THEN
FIFOR(EM2200,DM1002:Z01)
Z01+=1
END IF
IF LDP(R1700) THEN
FIFOW(DM1000,EM2200)
DM3033=AVG(EM2202,EM2200)
END IF
```

わかりやすい記述

```
'検査データの最新10個の平均値を算出します。
IF CR2008 OR LDP(R5200) THEN        'R5200=FIFOテーブルの初期化トリガ
    EM2200 = 0                      '現在FIFOテーブルに格納されているデータの個数
    EM2201 = 10                     'FIFOテーブルの最大個数
    Z01 = 0                         'インデックス初期化
END IF

IF EM2200 >= 10 THEN                'FIFOテーブルが一杯になったら
    FIFOR(EM2200, DM1002:Z01)       '最も古いデータをテーブルから取り出す
    Z01 += 1                        '格納するデバイスのアドレスをずらす
END IF

IF LDP(R1700) THEN                  'R1700は検査完了トリガ
    FIFOW(DM1000, EM2200)           '最新のデータ(DM1000)をFIFOテーブルに書き込む
    DM3033 = AVG(EM2202, EM2200)    'DM3033=最新データの平均値(10個)
END IF
```

読みやすいスクリプトを書くための工夫

- インデント(字下げ)を使う。
- 演算子の前後にスペースを入れる。
- 改行(空白行)を入れる。
- 行コメントや文末のコメントを入れるようにする。

参 考

・インデントを自動的に付ける機能があります。
メニューより【ツール】→【オプション】のスクリプト設定タブにあるオートインデントの設定を[する]にしてください。

・スクリプトの1行が長くなる場合、末尾に_(アンダースコア)を付けて、改行できます。

インラインニモニック

ニモニックは、作成したプログラムをCPUユニットで動作させるための言語です。
KVスクリプトや拡張ラダーで記述したプログラムは、CPUユニットへの転送時にニモニックに変換されます。
動作イメージの詳細は、使用しているCPUユニットのユーザーズマニュアルを参照してください。

インラインニモニックとは、スクリプトプログラム中にニモニックを直接記述できる機能のことで、ニモニック言語をスクリプト言語に混在させて記述できます。
ニモニックの文頭に「!!」をつけて記述することで、スクリプトと区別します。

ポイント

以下のインラインニモニックは使用できません。

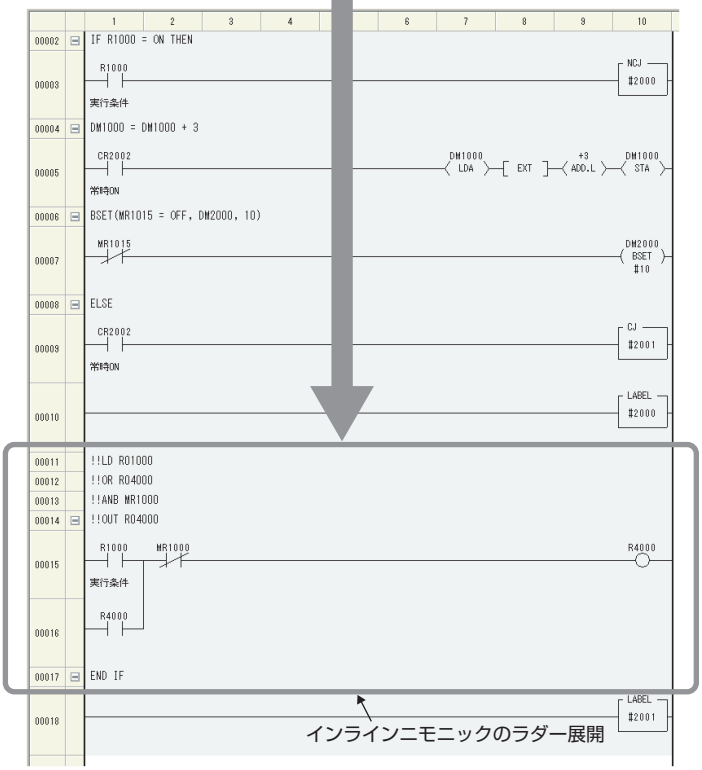
- **!!END**
- **!!ENDH**

スクリプト記述

```
IF R1000 = ON THEN
    DM1000 = DM1000 + 3
    BSET(MR1015 = OFF, DM2000, 10)
ELSE
    !!LD     R01000
    !!OR     R04000
    !!ANB    MR1000
    !!OUT    R04000
END IF
```

インラインニモニック記述部

ラダー展開



編集画面の注意事項とカスタマイズ

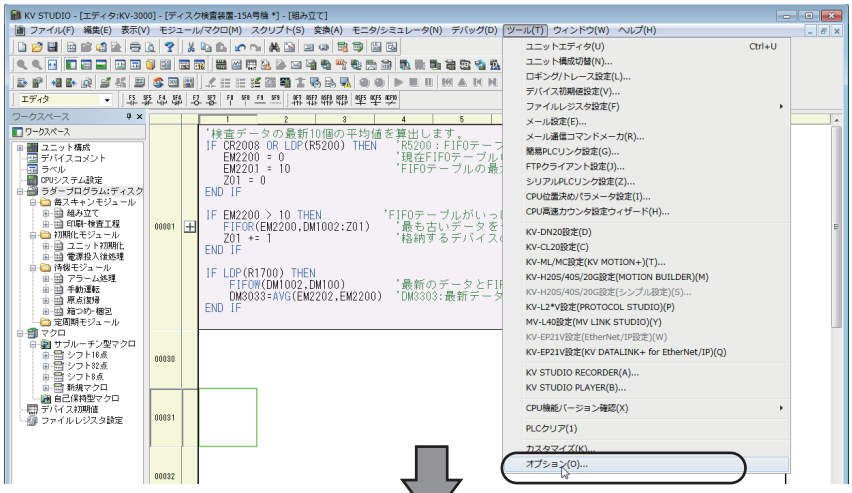
■スクリプト記述時の注意事項

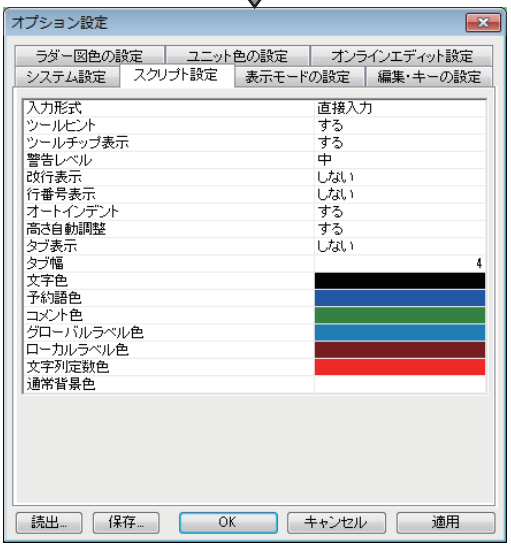
- スクリプトの記述には2バイト(漢字)コードは使用できません(ただし、コメント部や文字列定数、グローバルラベル/ローカルラベルには漢字が使用できます)。
- 1行の文字数はスクリプト、コメントを含めて1024文字(半角)までです。
- KVスクリプトを展開したラダーを直接は編集できません。

■スクリプトウィンドウのカスタマイズ

メニューから、【ツール(T)】→【オプション(O)】→【スクリプト設定】で、編集ウィンドウのカスタマイズがおこなえます。

詳細は、『KV STUDIOユーザーズマニュアル』を参照してください。





スクリプト用ワークメモリ(ワークデバイス)について

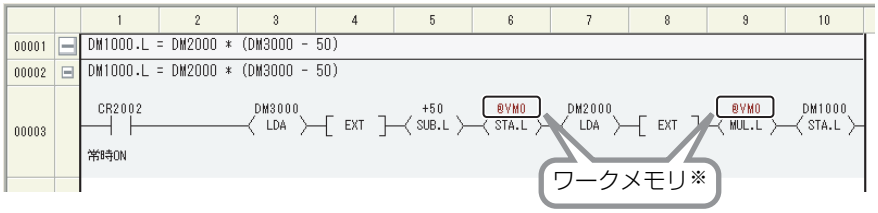
スクリプトプログラムをラダープログラムに変換したとき、CPUユニットがKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合ワークメモリを、CPUユニットがKV-1000の場合ワークデバイスを使用することがあります。

ワークメモリ(ワークデバイス)

ワークメモリ(ワークデバイス)とは、複雑な演算式などで部分的な演算結果を一時的に保存しておくシステム領域のデバイスのことです。
スクリプトプログラムでは、スクリプト変換時にKV STUDIOがあらかじめモジュール/マクロ用に確保している領域から、自動的にワークメモリを割り当ててくれるので、デバイスの重複使用などを意識せずに使用できます。

スクリプト記述　DM1000.L = DM2000 * (DM3000 – 50)

ラダー変換



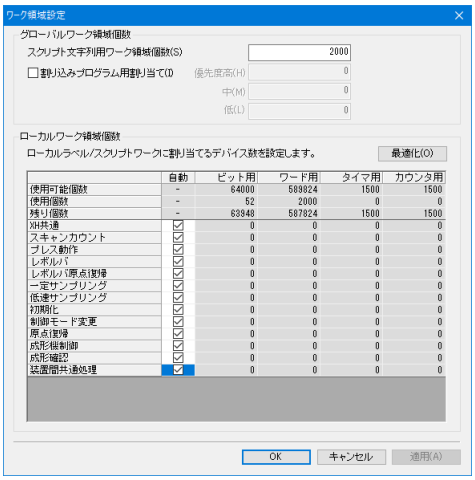
※CPUユニットがKV-1000の場合、EM64536などのワークデバイスが使用されます。

動作説明　最初に (DM3000-50) の演算をおこなってワークメモリ (@VM0) に一時的に格納し、ワークメモリ(@VM0)に格納されている値とDM2000を乗算した結果を、左辺のDM1000.Lに格納します。

■ワークメモリの使用領域

使用するCPUユニットがKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合、スクリプトはワークメモリを利用します。
スクリプト用ワークメモリは、[ワーク領域設定] ダイアログで使用状況を確認します。

『KV STUDIOユーザーズマニュアル』

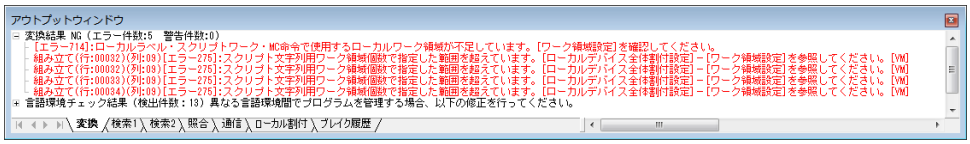


■ワークデバイスの使用領域

使用するCPUユニットがKV-1000の場合、スクリプトはワークデバイスを利用します。
ワークデバイスは、あらかじめ決められた領域を「ビット用ワーク領域」「ワード用ワーク領域」「文字列用ワーク領域」として確保していますが、プログラム内容によってはワークデバイスの数が不足することがあります。

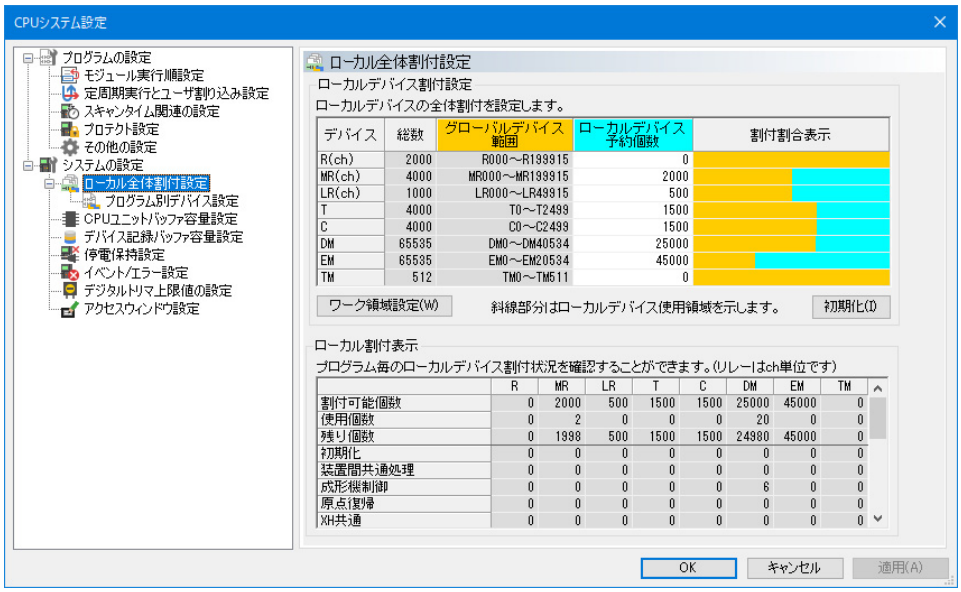
●ワークデバイスに不足が生じた場合

ワークデバイスに不足が発生すると、ラダープログラムを変換したときアウトプットウィンドウに以下のメッセージが表示されます。



この状態ではプログラムをPLCに転送できませんので、以下の手順でワークデバイスの領域を拡大してください。

1 **メニューから【変換(A)】→【ローカルデバイスの全体割付設定(A)】を選択して、[CPUシステム設定]ダイアログを表示させます。**

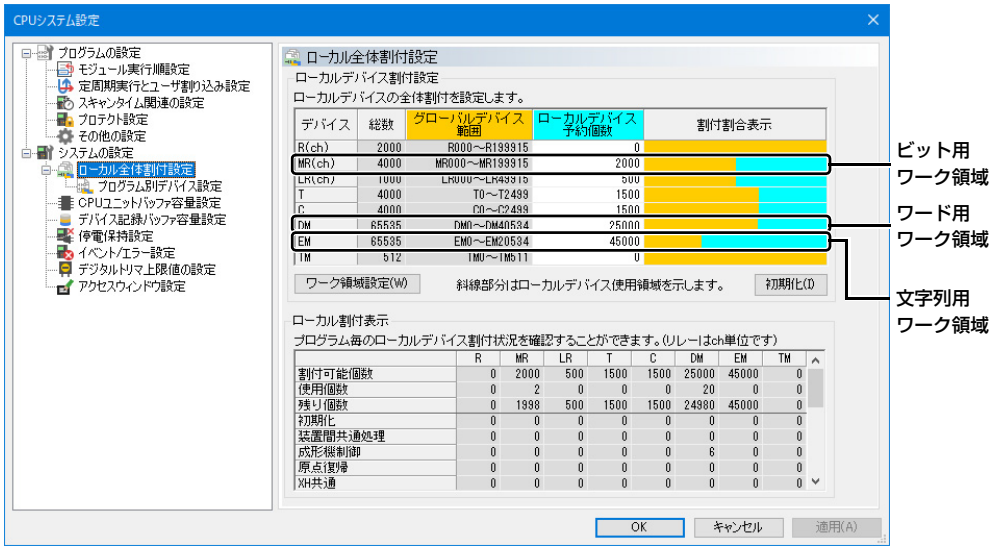


別手順
- ワークスペースの[CPUシステム設定]をダブルクリックする。
- アウトプットウィンドウのエラーメッセージをダブルクリックする
- ボタンをクリックする。

参考　スクリプト用ワーク予約個数の設定は、対応機種 (CPU ユニット) に「KV-1000」を選択しているプロジェクトのみ、表示されます。

2 スクリプト用ワーク予約個数の欄を編集します。

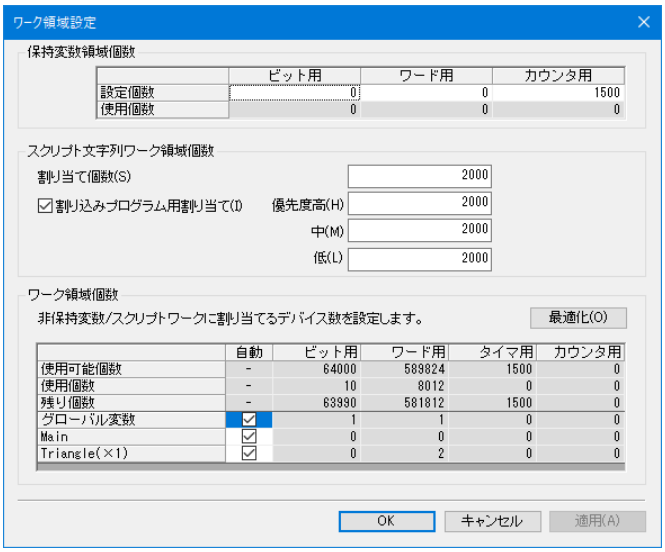
アウトプットウィンドウのエラーメッセージにしたがって、各ワーク領域の割付を変更してください。



参 考 ワークデバイスには、ビット状態を一時的に格納する「ビット用ワークデバイス」、演算結果などを一時的に格納する「ワード用ワークデバイス」、文字列を一時的に格納する「文字列用ワークデバイス」があります。

ポイント **文字列用ワークデバイスはプロジェクト内で何度も再利用されるため割り込みプログラムで文字列を扱うスクリプトを記述するときは注意が必要です。**

KV-8000および機能バージョン2.2以降のKV-7500/7300では〔ワーク領域設定(W)〕をクリックすると、割り込みプログラム用にワーク領域を割り当てられます。



XYM表記

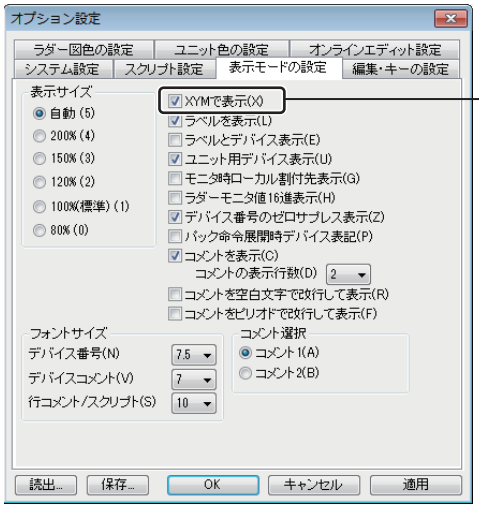
XYM表記でスクリプトプログラムが作成できます。

■XYM表記でプログラムを作成する

1 **メニューから【ツール(T)】→【オプション(O)】を選択します。**

［オプション設定］ダイアログボックスが表示されます。

2 **［表示モードの設定］タブをクリックします。**



［XYMで表示(X)］のチェックボックスをチェックして、OKをクリックします。

3 **XYM表記でプログラミングします。**

```
1 IF D1000 > K20 THEN
2     D0 = K50
3     SET(X10)
4 END IF
5
```

参 考 「XYMで表示」を選択後、通常デバイス番号での記述(混在可)もできます。

XYM表記でスクリプトを編集したあとXYM表記を解除すると、X、Yを使用しているスクリプトを再変換するとエラーが発生します。M、Lについては、デバイス番号が10進数の扱いからRのようなCH表記に変わるため、変換結果のデバイス番号が変わることがあるので注意してください。

例) M100　=　ON

このスクリプトのM100のラダーへの変換結果は、

・XYM表記時は、MR604に変換される。

・XYM表記でないときは、MR100に変換される。

MEMO

3

データの取り扱い

デバイス 3-2
インデックス修飾と間接指定 3-54
データの型（サフィックス） 3-62

デバイス

デバイスとは、CPUユニット内にあるリレーやデータメモリなど、プログラムで用いられる要素（変数）のことをいいます。演算や関数で使用するデータを与えたり、関数の実行結果を格納するために使用します。
デバイスは、対応機種（使用するCPUユニット）により異なります。

デバイス一覧 （KV-8000シリーズの場合）

■KV-8000シリーズのビットデバイス

デバイス名	範囲 ※1	点数	主な機能	運転開始時（電源 ON、PRG→RUN）	ローカルデバイス（@） ※1	インデックス修飾（:Z/:#）	間接指定（＊） ※8	
							参照	格納
入力リレー※10	R000 ～ R199915※2	合計 32000 点	外部機器から ON/OFF の情報を取り込むためのデバイス	クリア / 保持 ※3※12	○	○	○	×
出力リレー※10			外部機器へ ON/OFF 情報を送り出すためのデバイス					
内部補助リレー※10			CPU 内部と拡張ユニットで使用できるデバイス					
リンクリレー	B00 ～ B7FFF※10	32768 点	CPU 内部でのみ使用できるデバイス	クリア / 保持※3	×	○	○	×
内部補助リレー	MR000 ～ MR399915	64000 点		クリア	○	○	○	×
ラッチリレー	LR000 ～ LR99915	16000 点		保持	○	○	○	×
タイマ（接点）	T0 ～ T3999	4000 点	10μs、1ms、10ms、100ms の減算タイマ・10ms のアップダウンタイマの接点デバイス	クリア	○	○	○	×
カウンタ（接点）	C0 ～ C3999	4000 点	加算カウンタ・アップダウンカウンタの接点デバイス	クリア / 保持※7	○	○	○	×
コントロールリレー	CR000 ～ CR7915	1280 点	PLC の機能を制御したり、状態を取得するためのデバイス	保持	×	×	×	×

■KV-8000シリーズのワードデバイス

デバイス名	範囲 ※1	点数	主な機能	運転開始時（電源 ON、PRG → RUN）	ローカルデバイス（@）※1	インデックス修飾（:Z/:#）	間接指定（＊）※8	
							参照	格納
データメモリ	DM0 ～ DM65534	65535 点	数値データを格納できるデバイス（16 ビット）	クリア / 保持※3	○	○	○	○
拡張データメモリ	EM0 ～ EM65534	65535 点		クリア / 保持※3	○	○	○	○
ファイルレジスタ	連番方式：ZF0 ～ ZF524287	524288 点		クリア / 保持※3	×	○	○	○
	カレントバンク：FM0 ～ FM32767	（32768×16 バンク）点			○※11			
リンクレジスタ	W0 ～ W7FFF※10	32768 点		クリア / 保持※3	×	○	○	○
テンポラリメモリ	TM0 ～ TM511	512 点	数値データを格納できるデバイス（16 ビット）	クリア	○	○	○	○
タイマ（現在値 / 設定値）	T0 ～ T3999	4000 点	10μs、1ms、10ms、100ms の減算タイマ・10ms のアップダウンタイマの現在値 / 設定値（32 ビット）	現在値：クリア※5 現在値：保持	○	○	○	×
カウンタ（現在値 / 設定値）	C0 ～ C3999	4000 点	加算カウンタ・アップダウンカウンタの現在値 / 設定値（32 ビット）	クリア / 保持※7	○	○	○	×
インデックスレジスタ	Z1 ～ Z23※6	23 点	インデックス修飾で使用するデバイス（32 ビット）	クリア	×	×	×	×
コントロールメモリ	CM0 ～ CM7599	7600 点	PLC の機能を制御したり、状態を取得するためのデバイス（16 ビット）	保持	×	×	×	×

※1　ローカルデバイスとして使用するために、ローカルデバイスに予約した個数分、グローバルデバイスの使用可能な範囲が制限されます。
「ローカルデバイス一覧」（3-5ページ）
※2　R000～R199915は、《ユニットエディタ》で拡張入出力リレーに設定できます。
※3　CPUシステム設定の［停電保持設定］でクリア/ 保持の設定が可能です。
※4　『KV-8000 シリーズ ユーザーズマニュアル』「CR/CM 一覧」
※5　アップダウンタイマ（UDT）は、現在値を保持します。
※6　Z11～Z23 は、システム予約されているため使用できません。
※7　CPUシステム設定の［停電保持設定］でクリア/ 保持の設定が可能です。（接点、現在値のみ）
※8　「格納」は、「＊（デバイス）」と記述できるデバイスです。「参照」は、間接指定できるデバイスです。
『KV-8000シリーズ ユーザーズマニュアル』「間接指定」
※9　デバイス番号は、16進数で指定します。
※10　ダイレクト処理する場合は、「DR」で表記します。
『KV-8000シリーズ ユーザーズマニュアル』「ダイレクト処理について」
※11　ローカルデバイスは、バンク切り替えできません。
※12　入力リレーは、クリア/保持の設定はできません。

■マクロ引数デバイス

マクロ内のみで使用されるマクロ引数デバイスは以下のとおりです。

属性		記号	範囲	説明	間接指定（＊）	ローカルデバイス（@）	インデックス修飾（：）
デバイス		P	P0 ～ P9	デバイスを引数として受け取ります。	×	×	○
値		V	V0 ～ V9	デバイスの値または定数を引数として受け取ります。	×	×	×
ユニット	先頭 R	UR	UR0 ～ UR9	拡張ユニットに割り付けられた先頭リレー（R）を引数として受け取ります。	×	×	○
	先頭 DM	UM	UM0 ～ UM9	拡張ユニットに割り付けられた先頭データメモリ（DM）を引数として受け取ります。	×	×	○
	番号	UV	UV0 ～ UV9	拡張ユニットのユニット番号を引数として受け取ります。	×	×	×

■ファンクションブロック引数

ファンクションブロック内のみで使用される引数は以下のとおりです。

引数タイプ		引数名 / デバイス		説明	間接指定（＊）	ローカルデバイス（@）	インデックス修飾（：）
IN		任意のラベル		デバイスの値または定数を引数として受け取ります。	×	×	×
OUT		任意のラベル		演算結果を指定したデバイスに書き込みます。	×	×	×
IN-OUT		任意のラベル		デバイスを引数として受け取ります。	×	×	○
ユニット	先頭 R	UR	UR0 ～ UR7	ユニットに割り付けられた先頭リレー（R）を引数として受け取ります。	×	×	○
	先頭 DM	UM	UM0 ～ UM7	ユニットに割り付けられた先頭データメモリ（DM）を引数として受け取ります。	×	×	○
	番号	UV	UV0 ～ UV7	ユニットの番号を引数として受け取ります。	×	×	×

■定数

定数	範囲	主な機能
10 進数（#）	#0 ～ #65535（16 ビット符号無し） -32768 ～ +32767（16 ビット符号有り） #0 ～ #4294967295（32 ビット符号無し） -2147483648 ～ +2147483647（32 ビット符号有り）	10 進定数を指定する時に使用します。 ※　XYM 表示モードにすると「K」で表示します。
単精度浮動小数点型実数 ※	-3.4E+38 ≦ N ≦ -1.4E-45 N ＝ 0 +1.4E-45 ≦ N ≦ +3.4E+38 （有効桁数：約 7 桁）	単精度浮動小数点型定数を指定する時に使用します。
倍精度浮動小数点型実数 ※	-1.79E+308 ≦ N ≦ -2.23E-308 N ＝ 0 +2.23E-308 ≦ N ≦ +1.79E+308 （有効桁数：約 16 桁）	倍精度浮動小数点型定数を指定する時に使用します。
16 進数（$）	$0000 ～ $FFFF（16 ビット） $00000000 ～ $FFFFFFFF（32 ビット）	16 進定数を指定する時に使用します。 ※　XYM 表示モードにすると「H」で表示します。
固定文字列（" "）	（例）　" 型式 ABC"、" 月 /08:30:15"	固定文字列を指定する時に使用します。

■内部レジスタ

デバイス名	範囲	点数	主な機能	運転開始時（電源 ON、PROG → RUN）
内部レジスタ	－	1 点	データを一時的に格納するデバイス（16 ビット /32 ビット /64 ビット）	クリア

ローカルデバイス一覧

プロジェクト作成時に確保されるローカルデバイス / グローバルデバイスの個数･範囲の初期値を下記に示します。ローカルデバイスの予約個数は、［ワークスペース］上の［CPU システム設定］→［ローカル全体割付設定］で変更できます。

デバイス名 ※1	ローカルデバイス	グローバルデバイス	合計個数
	プロジェクト全体での予約個数 ※2	個数・範囲	
リレー（R）※3	なし	32000 点（2000ch） R000 ～ R199915	32000 点 （2000ch）
内部補助リレー（MR）※3	32000 点（2000ch）	32000 点（2000ch） MR000 ～ MR199915	64000 点 （4000ch）
ラッチリレー（LR）※3	8000 点（500ch）	8000 点（500ch） LR000 ～ LR49915	16000 点 （1000ch）
タイマ（T）※4	1500 点	2500 点 T0 ～ T2499	4000 点
カウンタ（C）※4	1500 点	2500 点 C0 ～ C2499	4000 点
データメモリ（DM）	25000 点	40535 点 DM0 ～ DM40534	65535 点
拡張データメモリ（EM）	45000 点	20535 点 EM0 ～ EM20534	65535 点
テンポラリデータメモリ（TM）	なし	512 点 TM000 ～ TM511	512 点

※1　リンクリレー(B)、リンクレジスタ(W)、ファイルレジスタ(ZF)、高速カウンタ(CTH)、高速カウンタコンパレータ(CTC)、コントロールリレー(CR)、コントロールメモリ(CM)、インデックスレジスタ(Z)は、ローカルデバイスとして使用できません。
@FMxxxを使用した場合、ワーク領域をローカルデバイスとして消費します。

※2　プロジェクト新規作成時に、ローカルデバイスとして使用可能なデバイスの領域を一括で確保します。(モジュールを10個使用して、マクロを50回呼び出すと想定して計算しています。)
各デバイスの予約個数は任意に変更できます。

※3　チャンネル(ch)単位で使用個数を設定します。

※4　タイマ･カウンタ命令のオペランドとしてデバイス番号を指定する場合のみ、「@＋(デバイス番号)」と記述します(ニモニック入力例:　TMR @0 #100)。

ポイント

- **リレー(R)とテンポラリデータメモリ(TM)をローカルデバイスとして使用する場合は、使用個数を設定する必要があります。**
- **以下のデバイスは、グローバルデバイスとして扱う必要があります。**
  - **拡張ユニットに割り当てられているデバイス**
  - **タッチパネルなどの外部接続機器で参照するデバイス**
  - **プロジェクト全体で共通して使用するデバイス**
  - **モジュール間でデータの受け渡しが必要なデバイス**
- **ローカルデバイスは、0番を先頭にできるだけ詰めて使用してください。**
- **R や DM は拡張ユニットに割り当てられるため、グローバルデバイスとして使用する領域が必要です。ローカルデバイスは、MRやEMを優先して使用してください。**
- **使用可能範囲外でローカルデバイス / グローバルデバイスを使用すると、変換エラーになります。**
- **KV-8000 シリーズではインデックスデバイスや、間接参照を使用して、ローカルデバイス/グローバルデバイスの使用可能範囲外を指定した場合演算エラーが発生します。**
- **オペランドに先頭デバイスを指定する各命令語、インデックス修飾、間接指定を使用する場合は、ローカルデバイス/グローバルデバイスの使用可能範囲を超えないようにしてください。**

デバイス一覧 （KV-7000シリーズの場合）

■KV-7000シリーズのビットデバイス

デバイス名	範囲 ※1	点数	主な機能	運転開始時（電源 ON、PRG→RUN）	ローカルデバイス（@）※1	インデックス修飾（:Z/:#）	間接指定（＊）※8	
							参照	格納
入力リレー※10	R000 ～ R199915 ※2※13	合計 32000 点	外部機器から ON/OFF の情報を取り込むためのデバイス	クリア / 保持 ※3※12	○	○	○	×
出力リレー※10			外部機器へ ON/OFF 情報を送り出すためのデバイス					
内部補助リレー※10			CPU 内部と拡張ユニットで使用できるデバイス					
リンクリレー	B00 ～ B7FFF※10	32768 点	CPU 内部でのみ使用できるデバイス	クリア / 保持※3	×	○	○	×
内部補助リレー	MR000 ～ MR399915	64000 点		クリア	○	○	○	×
ラッチリレー	LR000 ～ LR99915	16000 点		保持	○	○	○	×
タイマ（接点）	T0 ～ T3999	4000 点	10μs、1ms、10ms、100ms の減算タイマ・10ms のアップダウンタイマの接点デバイス	クリア	○	○	○	×
カウンタ（接点）	C0 ～ C3999	4000 点	加算カウンタ・アップダウンカウンタの接点デバイス	クリア / 保持※7	○	○	○	×
コントロールリレー	CR000 ～ CR7915	1280 点	PLC の機能を制御したり、状態を取得するためのデバイス	保持	×	×	×	×

■KV-7000シリーズのワードデバイス

デバイス名	範囲 ※1	点数	主な機能	運転開始時（電源 ON、PRG → RUN）	ローカルデバイス（@）※1	インデックス修飾（:Z/:#）	間接指定（＊）※8	
							参照	格納
データメモリ	DM0 ～ DM65534	65535 点	数値データを格納できるデバイス（16 ビット）	クリア / 保持※3	○	○	○	○
拡張データメモリ	EM0 ～ EM65534	65535 点		クリア / 保持※3	○	○	○	○
ファイルレジスタ	連番方式：ZF0 ～ ZF524287	524288 点		クリア / 保持※3	×	○	○	○
	カレントバンク：FM0 ～ FM32767	（32768×16 バンク）点			○※11			
リンクレジスタ	W0 ～ W7FFF※10	32768 点		クリア / 保持※3	×	○	○	○
テンポラリメモリ	TM0 ～ TM511	512 点	数値データを格納できるデバイス（16 ビット）	クリア	○	○	○	○
タイマ（現在値 / 設定値）	T0 ～ T3999	4000 点	10μs、1ms、10ms、100ms の減算タイマ・10ms のアップダウンタイマの現在値 / 設定値（32 ビット）	現在値：クリア※5 現在値：保持	○	○	○	×
カウンタ（現在値 / 設定値）	C0 ～ C3999	4000 点	加算カウンタ・アップダウンカウンタの現在値 / 設定値（32 ビット）	クリア / 保持※7	○	○	○	×
インデックスレジスタ	Z1 ～ Z12※6	12 点	インデックス修飾で使用するデバイス（32 ビット）	クリア	×	×	×	×
コントロールメモリ	CM0 ～ CM5999	6000 点	PLC の機能を制御したり、状態を取得するためのデバイス（16 ビット）	保持	×	×	×	×

※1　ローカルデバイスとして使用するために、ローカルデバイスに予約した個数分、グローバルデバイスの使用可能な範囲が制限されます。
「ローカルデバイス一覧」（3-9ページ）
※2　R000～R199915は、《ユニットエディタ》で拡張入出力リレーに設定できます。
※3　CPUシステム設定の［停電保持設定］でクリア/ 保持の設定が可能です。
※4　『KV-7000 シリーズ ユーザーズマニュアル』「CR/CM 一覧」
※5　アップダウンタイマ（UDT）は、現在値を保持します。
※6　Z11, Z12 は、システム予約されているため使用できません。
※7　CPUシステム設定の［停電保持設定］でクリア/ 保持の設定が可能です。（接点、現在値のみ）
※8　「格納」は、「＊（デバイス）」と記述できるデバイスです。「参照」は、間接指定できるデバイスです。
『KV-7000シリーズ ユーザーズマニュアル』「間接指定」
※9　デバイス番号は、16進数で指定します。
※10　ダイレクト処理する場合は、「DR」で表記します。
『KV-7000シリーズ ユーザーズマニュアル』「ダイレクト処理について」
※11　ローカルデバイスは、バンク切り替えできません。
※12　入力リレーは、クリア/保持の設定はできません。
※13　CPU機能バージョン2.2以前の場合は、Rの点数は16000、範囲はR00000～R99915になります。

■マクロ引数デバイス

マクロ内のみで使用されるマクロ引数デバイスは以下のとおりです。

属性		記号	範囲	説明	間接指定（＊）	ローカルデバイス（@）	インデックス修飾（：）
デバイス		P	P0 ～ P9	デバイスを引数として受け取ります。	×	×	○
値		V	V0 ～ V9	デバイスの値または定数を引数として受け取ります。	×	×	×
ユニット	先頭 R	UR	UR0 ～ UR9	拡張ユニットに割り付けられた先頭リレー（R）を引数として受け取ります。	×	×	○
	先頭 DM	UM	UM0 ～ UM9	拡張ユニットに割り付けられた先頭データメモリ（DM）を引数として受け取ります。	×	×	○
	番号	UV	UV0 ～ UV9	拡張ユニットのユニット番号を引数として受け取ります。	×	×	×

■ファンクションブロック引数

ファンクションブロック内のみで使用される引数は以下のとおりです。

引数タイプ		引数名 / デバイス		説明	間接指定（＊）	ローカルデバイス（@）	インデックス修飾（：）
IN		任意のラベル		デバイスの値または定数を引数として受け取ります。	×	×	×
OUT		任意のラベル		演算結果を指定したデバイスに書き込みます。	×	×	×
IN-OUT		任意のラベル		デバイスを引数として受け取ります。	×	×	○
ユニット	先頭 R	UR	UR0 ～ UR7	ユニットに割り付けられた先頭リレー（R）を引数として受け取ります。	×	×	○
	先頭 DM	UM	UM0 ～ UM7	ユニットに割り付けられた先頭データメモリ（DM）を引数として受け取ります。	×	×	○
	番号	UV	UV0 ～ UV7	ユニットの番号を引数として受け取ります。	×	×	×

■定数

定数	範囲	主な機能
10 進数（#）	#0 ～ #65535（16 ビット符号無し） -32768 ～ +32767（16 ビット符号有り） #0 ～ #4294967295（32 ビット符号無し） -2147483648 ～ +2147483647（32 ビット符号有り）	10 進定数を指定する時に使用します。 ※　XYM 表示モードにすると「K」で表示します。
単精度浮動小数点型実数 ※	-3.4E+38 ≦ N ≦ -1.4E-45 N ＝ 0 +1.4E-45 ≦ N ≦ +3.4E+38 （有効桁数：約 7 桁）	単精度浮動小数点型定数を指定する時に使用します。
倍精度浮動小数点型実数 ※	-1.79E+308 ≦ N ≦ -2.23E-308 N ＝ 0 +2.23E-308 ≦ N ≦ +1.79E+308 （有効桁数：約 16 桁）	倍精度浮動小数点型定数を指定する時に使用します。
16 進数（$）	$0000 ～ $FFFF（16 ビット） $00000000 ～ $FFFFFFFF（32 ビット）	16 進定数を指定する時に使用します。 ※　XYM 表示モードにすると「H」で表示します。
固定文字列（" "）	（例）　" 型式 ABC"、" 月 /08:30:15"	固定文字列を指定する時に使用します。

■内部レジスタ

デバイス名	範囲	点数	主な機能	運転開始時（電源 ON、PROG → RUN）
内部レジスタ	－	1 点	データを一時的に格納するデバイス（16 ビット /32 ビット /64 ビット）	クリア

ローカルデバイス一覧

プロジェクト作成時に確保されるローカルデバイス / グローバルデバイスの個数･範囲の初期値を下記に示します。ローカルデバイスの予約個数は、［ワークスペース］上の［CPU システム設定］→［ローカル全体割付設定］で変更できます。

デバイス名 ※1	ローカルデバイス	グローバルデバイス	合計個数
	プロジェクト全体での予約個数 ※2	個数・範囲	
リレー（R）※3	なし	32000 点（2000ch） R000 ～ R199915	32000 点 （2000ch）
内部補助リレー（MR）※3	32000 点（2000ch）	32000 点（2000ch） MR000 ～ MR199915	64000 点 （4000ch）
ラッチリレー（LR）※3	8000 点（500ch）	8000 点（500ch） LR000 ～ LR49915	16000 点 （1000ch）
タイマ（T）※4	1500 点	2500 点 T0 ～ T2499	4000 点
カウンタ（C）※4	1500 点	2500 点 C0 ～ C2499	4000 点
データメモリ（DM）	25000 点	40535 点 DM0 ～ DM40534	65535 点
拡張データメモリ（EM）	45000 点	20535 点 EM0 ～ EM20534	65535 点
テンポラリデータメモリ（TM）	なし	512 点 TM000 ～ TM511	512 点

※1　リンクリレー(B)、リンクレジスタ(W)、ファイルレジスタ(ZF)、高速カウンタ(CTH)、高速カウンタコンパレータ(CTC)、コントロールリレー(CR)、コントロールメモリ(CM)、インデックスレジスタ(Z)は、ローカルデバイスとして使用できません。
@FMxxxを使用した場合、ワーク領域をローカルデバイスとして消費します。

※2　プロジェクト新規作成時に、ローカルデバイスとして使用可能なデバイスの領域を一括で確保します。(モジュールを10個使用して、マクロを50回呼び出すと想定して計算しています。)
各デバイスの予約個数は任意に変更できます。

※3　チャンネル(ch)単位で使用個数を設定します。

※4　タイマ･カウンタ命令のオペランドとしてデバイス番号を指定する場合のみ、「@＋(デバイス番号)」と記述します(ニモニック入力例:　TMR @0 #100)。

ポイント

- **リレー(R)とテンポラリデータメモリ(TM)をローカルデバイスとして使用する場合は、使用個数を設定する必要があります。**
- **以下のデバイスは、グローバルデバイスとして扱う必要があります。**
  - **拡張ユニットに割り当てられているデバイス**
  - **タッチパネルなどの外部接続機器で参照するデバイス**
  - **プロジェクト全体で共通して使用するデバイス**
  - **モジュール間でデータの受け渡しが必要なデバイス**
- **ローカルデバイスは、0番を先頭にできるだけ詰めて使用してください。**
- **R や DM は拡張ユニットに割り当てられるため、グローバルデバイスとして使用する領域が必要です。ローカルデバイスは、MRやEMを優先して使用してください。**
- **使用可能範囲外でローカルデバイス / グローバルデバイスを使用すると、変換エラーになります。**
- **KV-7000 シリーズではインデックスデバイスや、間接参照を使用して、ローカルデバイス/グローバルデバイスの使用可能範囲外を指定した場合演算エラーが発生します。**
- **オペランドに先頭デバイスを指定する各命令語、インデックス修飾、間接指定を使用する場合は、ローカルデバイス/グローバルデバイスの使用可能範囲を超えないようにしてください。**

デバイス一覧 （KV-5500/5000/3000/1000の場合）

■KV-5500/5000/3000のビットデバイス

デバイス名	範囲※1	点数	主な機能	運転開始時（電源ON、PROG→RUN）	ローカルデバイス（@）※1	インデックス修飾（:Z/:#）	間接指定（*）※9	
							参照	格納
入力リレー	R000～R015 （R1000～R99915※3）	入力･出力･内部補助リレー合計で15936点	外部機器からON/OFFの情報を取り込むためのデバイス	—	○	○	○	×
出力リレー	R500～R507 （R1000～R99915※3）		外部機器へON/OFF情報を送り出すためのデバイス	クリア/保持※4	○	○	○	×
内部補助リレー	R508～R915※2 R1000～R99915※3		CPU内部と拡張ユニットで使用できるデバイス	クリア/保持※4	○	○	○	×
リンクリレー	B00～B1FFF※10	16384点	CPU内部でのみ使用できるデバイス	クリア/保持※4	×	○	○	×
内部補助リレー	MR000～MR99915	16000点		クリア	○	○	○	×
ラッチリレー	LR000～LR99915	16000点		保持	○	○	○	×
タイマ（接点）	T0～T3999	4000点	10μs、1ms、10ms、100msの減算タイマ･10msのアップダウンタイマの接点デバイス	クリア	○	○	○	×
カウンタ（接点）	C0～C3999	4000点	加算カウンタ･アップダウンカウンタの接点デバイス	クリア/保持※8	○	○	○	×
高速カウンタコンパレータ（接点）	CTC0～CTC3	4点	高速カウンタの接点デバイス	クリア/保持※8	×	×	×	×
コントロールリレー	CR000～CR3915	640点	PLCの機能を制御や、状態を取得するためのデバイス	保持	×	×	×	×

参 考 入力/出力/内部補助リレーの一部は、割り込みや高速カウンタに割り当てられています。
『KV-5500/5000/3000シリーズ ユーザーズマニュアル』「3-6 デバイスと定数」

■KV-5500/5000/3000のワードデバイス

デバイス名	範囲※1	点数	主な機能	運転開始時（電源ON、PROG→RUN）	ローカルデバイス（@）※1	インデックス修飾（:Z/:#）	間接指定（＊）※9	
							参照	格納
データメモリ	DM0～DM65534	65535点	数値データを格納できるデバイス（16ビット）	クリア/保持※4	○	○	○	○
拡張データメモリ	EM0～EM65534	65535点		クリア/保持※4	○	○	○	○
ファイルレジスタ	連番方式：ZF0～ZF131071	131072点		保持	×	○	○	○
	バンクメモリ：FM0～FM32767	（32768×4バンク）点			○※12	○	○	○
リンクレジスタ	W0～W3FFF※10	16384点		クリア/保持※4	×	○	○	○
テンポラリメモリ	TM0～TM511	512点	数値データの格納、簡易間接指定に使用するデバイス（16ビット）	クリア	○	○	○	○
タイマ（現在値/設定値）	T0～T3999	4000点	10μs、1ms、10ms、100msの減算タイマ・10msのアップダウンタイマの現在値/設定値（32ビット）	現在値：クリア※6 設定値：保持	○	○	○	×
カウンタ（現在値/設定値）	C0～C3999	4000点	加算カウンタ・アップダウンカウンタの現在値/設定値（32ビット）	クリア/保持※8	○	○	○	×
高速カウンタ（現在値）	CTH0～CTH1	2点	高速カウンタの現在値（32ビット）	クリア/保持※8	×	×	×	×
高速カウンタコンパレータ（設定値）	CTC0～CTC3	4点	高速カウンタの設定値（32ビット）	クリア/保持※4	×	×	×	×
インデックスレジスタ	Z1～Z12※7	12点	インデックス修飾で使用するデバイス（32ビット）※11	クリア	×	×	×	×
コントロールメモリ	CM0～CM5999	6000点	PLCの機能を制御や、状態を取得するためのデバイス（16ビット）	保持	×	×	×	×

※1　ローカルデバイスとしても使用するために、ローカルデバイスに予約した個数分、グローバルデバイスの使用可能な範囲が制限されます。
「ローカルデバイス一覧（KV-5500/5000/3000の場合）」（3-15ページ）
※2　R100～R415、R508～R915は、システム予約されているため使用できません。
※3　R1000～R99915は、《ユニットエディタ》で拡張入出力リレーに設定できます。
※4　CPUシステム設定の［停電保持設定］でクリア/保持の設定が可能です。
※5　「CR/CM一覧」（付-2ページ）
※6　アップダウンタイマ（UDT）は、現在値を保持します。
※7　Z11, Z12は、システム予約されているため使用できません。
※8　CPUシステム設定の［停電保持設定］で保持/クリアの設定が可能です。（接点、現在値のみ）
※9　「格納」は、「＊（デバイス）」と記述できるデバイスです。「参照」は、間接指定できるデバイスです。
「間接指定」（3-57ページ）
※10　デバイス番号は、16進数で指定します。
※11　インデックスレジスタが扱える値の範囲は、「-131072～+131071」です。
※12　ローカルデバイスは、バンク切り替えできません。

■KV-1000のビットデバイス

デバイス名	範囲※1	点数	主な機能	運転開始時（電源ON、PROG→RUN）	ローカルデバイス（@）※1	インデックス修飾（:Z/:#）	間接指定（*）※10	
							参照	格納
入力リレー	R000～R015（R1000～R59915※3）	入力･出力･内部補助リレー合計で9536点	外部機器からON/OFFの情報を取り込むためのデバイス	—	○	○	○	×
出力リレー	R500～R507（R1000～R59915※3）		外部機器へON/OFF情報を送り出すためのデバイス	クリア※4	○	○	○	×
内部補助リレー	R508～R915※2 R1000～R59915※3		CPU内部と拡張ユニットで使用できるデバイス	クリア※4	○	○	○	×
	MR000～MR99915※4	16000点	CPU内部でのみ使用できるデバイス	クリア	○	○	○	×
内部補助リレー（ラッチリレー）	LR000～LR99915	16000点		保持	○	○	○	×
タイマ（接点）	T0～T3999	タイマ･カウンタ合計で4000点	1ms、10ms、100msの減算タイマ･10msのアップダウンタイマの接点デバイス	クリア	○	○	○	×
カウンタ（接点）	C0～C3999		加算カウンタ･アップダウンカウンタの接点デバイス	保持※9	○	○	○	×
高速カウンタコンパレータ（接点）	CTC0～CTC3	4点	高速カウンタの接点デバイス	保持※9	×	×	×	×
コントロールリレー※6	CR000～CR3915	640点	PLCの機能を制御したり、状態を取得するためのデバイス	—	×	×	×	×

参考　入力/出力/内部補助リレーの一部は、割り込みや高速カウンタに割り当てられています。

『KV-1000 ユーザーズマニュアル』「4章 割り込み」
『KV-1000 ユーザーズマニュアル』「5章 高速カウンタ」

■KV-1000のワードデバイス

デバイス名	範囲※1	点数	主な機能	運転開始時（電源ON、PROG→RUN）	ローカルデバイス（@）※1	インデックス修飾（:Z/:#）	間接指定（*）※10	
							参照	格納
データメモリ	DM0～DM65534	65535点	数値データを格納できるデバイス（16ビット）	保持※4	○	○	○	○
拡張データメモリ	EM0～EM65534※5	65535点		保持※4	○	○	○	○
	FM0～FM32766※5	32767点		保持※4	○	○	○	○
テンポラリデータメモリ	TM0～TM511	512点	数値データの格納、簡易間接指定・四則演算時に使用するデバイス（16ビット）	クリア	○	○	○	○
タイマ（現在値/設定値）	T0～T3999	タイマ・カウンタ合計で4000点	1ms、10ms、100msの減算タイマ・10msのアップダウンタイマの現在値/設定値（32ビット）	現在値：クリア※7 設定値：保持	○	○	○	×
カウンタ（現在値/設定値）	C0～C3999		加算カウンタ・アップダウンカウンタ（32ビット）	保持※9	○	○	○	×
高速カウンタ（現在値）	CTH0～CTH1	2点	高速カウンタの現在値（32ビット）	保持※9	×	×	×	×
高速カウンタコンパレータ（設定値）	CTC0～CTC3	4点	高速カウンタの設定値（32ビット）	保持	×	×	×	×
インデックスレジスタ	Z1～Z12※8	12点	インデックス修飾で使用するデバイス（16ビット）	クリア	×	×	×	×
コントロールメモリ※6	CM0～CM11998	11999点	PLCの機能を制御したり、状態を取得するためのデバイス（16ビット）	—	×	×	×	×

※1　ローカルデバイスとして使用するために予約する個数分、グローバルデバイスの使用可能な範囲が制限されます。
「ローカルデバイス一覧（KV-1000の場合）」（3-16ページ）
※2　R100～R415は、システム予約されているため使用できません。
※3　R1000～R59915は、《ユニットエディタ》で拡張入出力リレーに設定できます。
※4　MEMSW命令でクリア/保持の設定が可能です。
※5　スクリプト用ワークデバイスの割付設定により、一部システム予約されている領域があります。
※6　「CR/CM一覧」（付-2ページ）
※7　アップダウンタイマ（UDT）は、現在値を保持します。
※8　Z11, Z12は、システム予約されているため使用できません。
※9　MEMSW命令で保持/クリアの設定が可能です。（接点、現在値のみ）
※10　「格納」は、「*（デバイス）」と記述できるデバイスです。「参照」は、間接指定できるデバイスです。
「間接指定」（3-57ページ）

■マクロ引数デバイス

マクロ内のみで使用されるマクロ引数デバイスは以下のとおりです。

属性		記号	範囲	説明	間接指定（＊）	ローカルデバイス（@）	インデックス修飾（：）
デバイス		P	P0 ～ P9	デバイスを引数として受け取ります。	×	×	○
値		V	V0 ～ V9	デバイスの値または定数を引数として受け取ります。	×	×	×
ユニット	先頭 R	UR	UR0 ～ UR9	拡張ユニットに割り付けられた先頭リレー（R）を引数として受け取ります。	×	×	○
	先頭 DM	UM	UM0 ～ UM9	拡張ユニットに割り付けられた先頭データメモリ（DM）を引数として受け取ります。	×	×	○
	番号	UV	UV0 ～ UV9	拡張ユニットのユニット番号を引数として受け取ります。	×	×	×

■定数

定数	範囲	主な機能
10進数(#)	(#)0～(#)65535（16ビット符号無し） (#)–32768～(#)+32767（16ビット符号付き） (#)0～(#)4294967295（32ビット符号無し） (#)–2147483648～(#)+2147483647（32ビット符号付き）	10進定数を指定する時に使用します。(#は省略できます。) 参考 XYM表示モードにすると「K」で表示します。（Kは省略できます。）
単精度浮動小数点型実数	(#)–3.4E+38≦N≦(#)–1.4E–45 N=(#)0 (#)+1.4E–45≦N≦(#)+3.4E+38 （有効桁数：約7桁）	単精度浮動小数点型定数を指定する時に使用します。 (#は省略できます。)
倍精度浮動小数点型実数	(#)－1.79E+308≦N≦(#)－2.23E－308 N=(#)0 (#)+2.23E－308≦N≦(#)+1.79E+308 （有効桁数：約16桁）	倍精度浮動小数点型定数を指定する時に使用します。 (#は省略できます。)
16進数($)	$0～$FFFF（16ビット） $0～$FFFFFFFF（32ビット）	16進定数を指定する時に使用します。 参考 XYM表示モードにすると「H」で表示します。
固定文字列(" ")	例）　"型式ABC"、"月/08:30:15"	固定文字列を指定する時に使用します。

■内部レジスタ

デバイス名	範囲	点数	主な機能	運転開始時（電源 ON、PROG → RUN）
内部レジスタ	－	1 点	データを一時的に格納するデバイス（16 ビット /32 ビット /64 ビット）	クリア

ローカルデバイス一覧（KV-5500/5000/3000の場合）

プロジェクト作成時に確保されるローカルデバイス/グローバルデバイスの個数･範囲の初期値を下記に示します。ローカルデバイスの予約個数は、[ワークスペース]上の[CPUシステム設定]→[ローカル全体割付設定]で変更できます。

デバイス名※1	ローカルデバイス プロジェクト全体での予約個数※2	グローバルデバイス 個数・範囲	合計個数
リレー（R）※3	なし	16000 点（1000ch） R000 ～ R99915	16000 点 （1000ch）
内部補助リレー（MR）※3	8000 点（500ch）	8000 点（500ch） MR000 ～ MR49915	16000 点 （1000ch）
ラッチリレー（LR）※3	8000 点（500ch）	8000 点（500ch） LR000 ～ LR49915	16000 点 （1000ch）
タイマ（T）※4	1500 点	2500 点 T0 ～ T2499	4000 点
カウンタ（C）※4	1500 点	2500 点 C0 ～ C2499	4000 点
データメモリ（DM）	25000 点	40535 点 DM0 ～ DM40534	65535 点
拡張データメモリ（EM）	45000 点	20535 点 DM0 ～ DM20534	65535 点
テンポラリデータメモリ（TM）	なし	512 点 TM0 ～ TM511	512 点

※1　リンクリレー(B)、リンクレジスタ(W)、ファイルレジスタ(ZF)、高速カウンタ(CTH)、高速カウンタコンパレータ(CTC)、コントロールリレー(CR)、コントロールメモリ(CM)、インデックスレジスタ(Z)は、ローカルデバイスとして使用できません。
@FMxxxを使用した場合、ワーク領域をローカルデバイスとして消費します。

※2　プロジェクト新規作成時に、ローカルデバイスとして使用可能なデバイスの領域を一括で確保します。(モジュールを10個使用して、マクロを50回呼び出すと想定して計算しています。)
各デバイスの予約個数は任意に変更できます。

※3　チャンネル(ch)単位で使用個数を設定します。

※4　タイマ･カウンタ命令のオペランドとしてデバイス番号を指定する場合のみ、「@＋(デバイス番号)」と記述します。　(ニモニック入力例:　TMR @0 #100)。

ポイント

- **リレー(R)とテンポラリデータメモリ(TM)をローカルデバイスとして使用する場合は、使用個数を設定する必要があります。**
- **以下のデバイスは、グローバルデバイスとして扱う必要があります。**
  - **《KV-5500/5000/3000》の入出力リレー**
  - **拡張ユニットに割り当てられているデバイス**
  - **タッチパネルなどの外部接続機器で参照するデバイス**
  - **プロジェクト全体で共通して使用するデバイス**
  - **モジュール間でデータの受け渡しが必要なデバイス**
- **ローカルデバイスは、0番を先頭にできるだけ詰めて使用してください。**
- **R や DM は拡張ユニットに割り当てられるため、グローバルデバイスとして使用する領域が必要です。ローカルデバイスは、MRやEMを優先して使用してください。**
- **使用可能範囲外でローカルデバイス / グローバルデバイスを使用すると、変換エラーになります。**
- **オペランドに先頭デバイスを指定する各命令語、インデックス修飾、間接指定を使用する場合は、ローカルデバイス/グローバルデバイスの使用可能範囲を超えないようにしてください。**

ローカルデバイス一覧（KV-1000の場合）

プロジェクト作成時に確保されるローカルデバイス/ グローバルデバイスの個数･範囲の初期値を下記に示します。ローカルデバイスの予約個数は、[ワークスペース] 上の [CPU システム設定] →[ローカル全体割付設定] で変更できます。

デバイス名 ※1	ローカルデバイス プロジェクト全体での予約個数 ※2	グローバルデバイス 個数・範囲	合計個数
リレー（R）※3	なし	9600 点（600ch） R00000 ～ R59915	9600 点 （600ch）
内部補助リレー（MR）※3	8000 点（500ch）	7200 点（450ch） MR00000 ～ MR44915	16000 点 （1000ch）※4
ラッチリレー（LR）※3	8000 点（500ch）	8000 点（500ch） LR00000 ～ LR49915	16000 点 （1000ch）
タイマ（T）※5	1500 点	タイマ・カウンタ合計で 1000 点 T0000 ～ T0999 C0000 ～ C0999	タイマ・カウンタ合計で 4000 点
カウンタ（C）※5	1500 点		
データメモリ（DM）	25000 点	40535 点 DM00000 ～ DM40534	65535 点
拡張データメモリ（EM）	45000 点	19535 点 EM00000 ～ EM19534	65535 点 ※4
拡張データメモリ（FM）	20000 点	11767 点 FM00000 ～ FM11766	32767 点 ※4
テンポラリデータメモリ（TM）	なし	512 点 TM000 ～ TM511	512 点

※1　高速カウンタ（CTH）、高速カウンタコンパレータ（CTC）、コントロールリレー（CR）、コントロールメモリ（CM）、インデックスレジスタ（Z）は、ローカルデバイスとして使用できません。
※2　プロジェクト新規作成時に、ローカルデバイスとして使用可能なデバイスの領域を一括で確保します（モジュールを10個使用して、マクロを50回呼び出すと想定して計算しています）。
各デバイスの予約個数は任意に変更できます。
※3　チャンネル（ch）単位で使用個数を設定します。
※4　スクリプト用ワークデバイスとして確保されている領域を含みます。
※5　タイマ･カウンタ命令のオペランドとしてデバイス番号を指定する場合のみ、「@＋（デバイス番号）」と記述します　（ニモニック入力例：　TMR @0 #100）。

ポイント

- **リレー（R）とテンポラリデータメモリ（TM）をローカルデバイスとして使用する場合は、使用個数を設定する必要があります。**
- **以下のデバイスは、グローバルデバイスとして扱う必要があります。**
  - **《KV-1000》の入出力リレー**
  - **拡張ユニットに割り当てられているデバイス**
  - **タッチパネルなどの外部接続機器で参照するデバイス**
  - **プロジェクト全体で共通して使用するデバイス**
  - **モジュール間でデータの受け渡しが必要なデバイス**
- **ローカルデバイスは、0番を先頭にできるだけ詰めて使用してください。**
- **R や DM は拡張ユニットに割り当てられるため、グローバルデバイスとして使用する領域が必要です。ローカルデバイスは、MRやEM/FMを優先して使用してください。**
- **使用可能範囲外でローカルデバイス / グローバルデバイスを使用すると、変換エラーになります。**
- **オペランドに先頭デバイスを指定する各命令語、インデックス修飾、間接指定を使用する場合は、ローカルデバイス/グローバルデバイスの使用可能範囲を超えないようにしてください。**

デバイス一覧 （KV Nanoシリーズの場合）

■KV Nanoシリーズのビットデバイス

デバイス名	範囲 ※1	点数	主な機能	運転開始時		ローカルデバイス（@）※1	インデックス修飾（:Z/:#）	間接指定（＊）※2	
				電源ON	PROG→RUN			参照	格納
入力リレー ※3	R000 ～ R203※4（R1000 ～ 59915 ※5）	入力・出力・内部補助リレー合計で9600点	外部機器からON/OFFの情報を取り込むためのデバイス	クリア	クリア	○	○	○	×
出力リレー ※3	R500 ～ R607※6（R1000 ～ 59915 ※5）		外部機器へON/OFF情報を送り出すためのデバイス			○	○	○	×
内部補助リレー	R1000 ～ R59915※3		CPU内部と拡張ユニットで使用できるデバイス			○	○	○	×
リンクリレー	B00 ～ B1FFF※8	8192点	CPU 内部でのみ使用できるデバイス	クリア	クリア	×	○	○	×
内部補助リレー	MR000 ～ MR59915	9600点		クリア	クリア	○	○	○	×
ラッチリレー	LR000 ～ LR19915	3200点		保持	保持	○	○	○	×
タイマ（接点）	T0 ～ T511	512点	10μs、1ms、10ms、100msの減算タイマ・10msのアップダウンタイマの接点デバイス	T0~T9のうち、UDT命令で使用されているデバイスのみ保持	T0~T9のうち、UDT命令で使用されているデバイスのみ保持	○	○	○	×
カウンタ（接点）	C0 ～ C255	256点	加算カウンタ・アップダウンカウンタの接点デバイス	クリア/保持 ※7	保持	○	○	○	×
高速カウンタコンパレータ（接点）	CTC0 ～ CTC7※9	8点 ※9	高速カウンタの接点デバイス	保持	保持	×	×	×	×
コントロールリレー	CR000 ～ CR8915	1440点	PLC の機能を制御したり、状態を取得するためのデバイス	保持	クリア	×	×	×	×

※1　ローカルデバイスとして使用するために、ローカルデバイスに予約した個数分、グローバルデバイスの使用可能な範囲が制限されます。「ローカルデバイス一覧」（3-20ページ）
※2　「格納」は、「＊（デバイス）」と記述できるデバイスです。「参照」は、間接指定できるデバイスです。
『KV Nano シリーズ ユーザーズマニュアル』「間接指定」
※3　ダイレクト処理する場合は、「DR」で表記します。
※4　ユニットによる使用デバイスの範囲が異なります。
KV-N14□□：R000～R007、KV-N24□□：R000～R013
KV-N40□□：R000～R107、KV-N60□□：R000～R203
KV-NC32T:R000～R015
※5　R1000 ～ R59915 は、《ユニットエディタ》で拡張入出力リレーに設定できます。
R1000未満の入力リレー・出力リレーに割り当たっていないリレーはシステム予約されているため使用できません。
※6　ユニットによる使用デバイスの範囲が異なります。
KV-N14□□：R500～R505、KV-N24□□：R500～R509
KV-N40□□：R500～R515、KV-N60□□：R500～R607
KV-NC32T:R500～R515
※7　CPUシステム設定の［停電保持設定］でクリア/ 保持の設定が可能です。（接点、現在値のみ）
※8　デバイス番号は、16 進数で指定します。
※9　ユニットによる使用デバイスの範囲が異なります。
KV-N14□□：CTC0~CTC3（4点）、KV-N24□□：CTC0~CTC3（4点）
KV-N40□□：CTC0~CTC5（6点）、KV-N60□□：CTC0~CTC7（8点）
KV-NC32T:CTC0～CTC5（6点）

■KV Nanoシリーズのワードデバイス

デバイス名	範囲 ※1	点数	主な機能	運転開始時		ローカルデバイス(@) ※1	インデックス修飾(:Z/:#)	間接指定(＊) ※2	
				電源ON	PROG→RUN			参照	格納
データメモリ	DM0～DM32767	32768点	数値データを格納できるデバイス(16ビット)	クリア/保持 ※3	クリア/保持 ※3	○	○	○	○
リンクレジスタ	W0～W3FFF※5	16384点		クリア/保持 ※3	保持	×	○	○	○
テンポラリメモリ	TM0～TM511	512点	数値データを格納できるデバイス(16ビット)	クリア	クリア	○	○	○	○
タイマ(現在値/設定値)	T0～T511	512点	10μs、1ms、10ms、100ms の減算タイマ・10msのアップダウンタイマの現在値/設定値(32ビット)	T0~T9のうち、UDT命令で使用されているデバイスのみ保持	T0~T9のうち、UDT命令で使用されているデバイスのみ保持	○	○	○	×
カウンタ(現在値/設定値)	C0～C255	256点	加算カウンタ・アップダウンカウンタの現在値/設定値(32ビット)	クリア/保持 ※4	クリア/保持 ※4	○	○	○	×
高速カウンタ(現在値)	CTH0～CTH3※6	4点 ※6	高速カウンタの現在値(32ビット)	保持	保持	×	×	×	×
高速カウンタコンパレータ(設定値)	CTC0～CTC7※6	8点 ※6	高速カウンタの設定値(32ビット)	保持	保持	×	×	×	×
インデックスレジスタ	Z1～Z12※7	12点	インデックス修飾で使用するデバイス(32ビット)	クリア	保持	×	×	×	×
コントロールメモリ	CM0～CM8999	9000点	PLCの機能を制御したり、状態を取得するためのデバイス(16ビット)	保持 ※8	保持 ※8	×	○	○	×

※1　ローカルデバイスとして使用するために、ローカルデバイスに予約した個数分、グローバルデバイスの使用可能な範囲が制限されます。
「ローカルデバイス一覧」(3-20ページ)

※2　「格納」は、「＊(デバイス)」と記述できるデバイスです。「参照」は、間接指定できるデバイスです。

※3　『KV Nanoシリーズ ユーザーズマニュアル』「間接指定」

※4　CPUシステム設定の[停電保持設定]でクリア/保持の設定が可能です。

※5　CPUシステム設定の[停電保持設定]で保持/クリアの設定が可能です。(接点、現在値のみ)

※6　デバイス番号は、16進数で指定します。

※7　ユニットによる使用デバイスの範囲が異なります。
KV-N14□□:CTH0~CTH1(2点)CTC0~CTC3(4点)、KV-N24□□:CTH0~CTH1(2点)CTC0~CTC3(4点)
KV-N40□□:CTH0~CTH2(3点)CTC0~CTC5(6点)、KV-N60□□:CTH0~CTH3(4点)CTC0~CTC7(8点)
KV-NC32T:CTH0～CTH2(3点)、CTC0～CTC5(6点)

※8　Z11, Z12は、システム予約されているため使用できません。

※9　保持が必要なデバイスのみです。対象デバイスは『KV Nanoシリーズ ユーザーズマニュアル』「CR/CM一覧」を参照してください。

■マクロ引数デバイス

マクロ内のみで使用されるマクロ引数デバイスは以下のとおりです。

属性		記号	範囲	説明	間接指定（＊）	ローカルデバイス（@）	インデックス修飾（：）
デバイス		P	P0 ～ P9	デバイスを引数として受け取ります。	×	×	○
値		V	V0 ～ V9	デバイスの値または定数を引数として受け取ります。	×	×	×
ユニット	先頭 R	UR	UR0 ～ UR9	拡張ユニットに割り付けられた先頭リレー（R）を引数として受け取ります。	×	×	○
	先頭 DM	UM	UM0 ～ UM9	拡張ユニットに割り付けられた先頭データメモリ（DM）を引数として受け取ります。	×	×	○
	番号	UV	UV0 ～ UV9	拡張ユニットのユニット番号を引数として受け取ります。	×	×	×

■定数

定数	範囲	主な機能
10 進数 (#)	#0～#65535（16ビット符号無し） －32768～+32767（16ビット符号有り） #0～#4294967295（32ビット符号無し） －2147483648～+2147483647（32ビット符号有り）	10 進定数を指定する時に使用します。 ※ XYM 表示モードにすると「K」で表示します。
単精度浮動小数点型実数	－3.4E+38 ≦N≦－1.2E－38 N＝0 +1.2E－38 ≦N≦+3.4E+38 （有効桁数：約7桁）	単精度浮動小数点型定数を指定する時に使用します。 ※ 非正規化数（－1.2E-38≦N≦－1.4E－45、+1.4E－45≦N≦+1.2E－38）は指定できません。演算結果が非正規化数になる場合、0を返します。
倍精度浮動小数点型実数	－1.79E+308 ≦ N ≦ －2.23E－308 N ＝ 0 +2.23E－308 ≦ N ≦ +1.79E+308 （有効桁数：約16桁）	倍精度浮動小数点型定数を指定する時に使用します。
16 進数 ($)	$0000～$FFFF（16ビット） $00000000～$FFFFFFFF（32ビット）	16 進定数を指定する時に使用します。 ※ XYM 表示モードにすると「H」で表示します。
固定文字列 (" ")	（例） " 型式 ABC"、" 月 /08:30:15"	固定文字列を指定する時に使用します。

■内部レジスタ

デバイス名	範囲	点数	主な機能	運転開始時（電源 ON、PROG → RUN）
内部レジスタ	－	1 点	データを一時的に格納するデバイス（16 ビット /32 ビット /64 ビット）	クリア

ローカルデバイス一覧

プロジェクト作成時に確保されるローカルデバイス/グローバルデバイスの個数･範囲の初期値を下記に示します。ローカルデバイスの予約個数は、[ワークスペース]上の[CPUシステム設定]→[ローカル全体割付設定]で変更できます。

デバイス名 ※1	ローカルデバイス	グローバルデバイス	合計個数
	プロジェクト全体での予約個数 ※2	個数・範囲	
リレー（R）※3	なし	9600 点（600ch） R000 ～ R59915	9600 点 （600ch）
内部補助リレー（MR）※3	1600 点 （100ch）	8000 点（500ch） MR000 ～ MR49915	9600 点 （600ch）
ラッチリレー（LR）※3	320 点 （20ch）	3560 点（180ch） LR000 ～ LR17915	3200 点 （200ch）
タイマ（T）※4	100 点	412 点 T0 ～ T411	512 点
カウンタ（C）※4	50 点	206 点 C0 ～ C205	256 点
データメモリ（DM）	5000 点	27768 点 DM0 ～ DM27767	32768 点
テンポラリデータメモリ（TM）	なし	512 点 TM000 ～ TM511	512 点

※1　リンクリレー(B)、リンクレジスタ(W)、高速カウンタ(CTH)、高速カウンタコンパレータ(CTC)、コントロールリレー(CR)、コントロールメモリ(CM)、インデックスレジスタ(Z)は、ローカルデバイスとして使用できません。

※2　プロジェクト新規作成時に、ローカルデバイスとして使用可能なデバイスの領域を一括で確保します。(モジュールを10個使用して、マクロを50回呼び出すと想定して計算しています。)
各デバイスの予約個数は任意に変更できます。

※3　チャンネル(ch)単位で使用個数を設定します。

※4　タイマ･カウンタ命令のオペランドとしてデバイス番号を指定する場合のみ、「@＋(デバイス番号)」と記述します。　(ニモニック入力例:　TMR @0 #100)。

ポイント

- **リレー(R)とテンポラリデータメモリ(TM)をローカルデバイスとして使用する場合は、使用個数を設定する必要があります。**
- **以下のデバイスは、グローバルデバイスとして扱う必要があります。**
  - **《KV Nano シリーズ》の入出力リレー**
  - **拡張ユニットに割り当てられているデバイス**
  - **タッチパネルなどの外部接続機器で参照するデバイス**
  - **プロジェクト全体で共通して使用するデバイス**
  - **モジュール間でデータの受け渡しが必要なデバイス**
- **ローカルデバイスは、0番を先頭にできるだけ詰めて使用してください。**
- **Rは拡張ユニットに割り当てられるため、グローバルデバイスとして使用する領域が必要です。ローカルデバイスは、MRを優先して使用してください。**
- **使用可能範囲外でローカルデバイス / グローバルデバイスを使用すると、変換エラーになります。**
- **オペランドに先頭デバイスを指定する各命令語、インデックス修飾、間接指定を使用する場合は、ローカルデバイス/グローバルデバイスの使用可能範囲を超えないようにしてください。**

デバイスの表記方法

●インデックス修飾（インデックスレジスタによる修飾、インデックス定数による修飾）の記述

・インデックスレジスタによる修飾

デバイス（R（DR）/ MR / LR / B / T / C / DM / EM / FM / ZF / W / TM / ＊DM / ＊EM / ＊FM / ＊ZF / ＊ W / ＊TM）の後に：Z□（□：1～10の数値）を続けて入力します。

例

DM0:Z01　（Z1の値が100なら、DM100を意味します）
R1000:Z01　（Z1の値が100なら、R1604を意味します）

ポイント
- **KV Nanoシリーズでは EM、FMは使用できません。**
- **KV Nanoシリーズでのみ、CMに対してインデックスレジスタによる修飾ができます。**

・インデックス定数による修飾

デバイス（R（DR）/ MR / LR / B / T / C / DM / EM / FM / ZF / W / TM / ＊DM / ＊EM / ＊FM / ＊ZF / ＊ W / ＊TM）の後に ：#□（□：－524287～524287）を続けて入力します。

例

DM0:#1　（DM1を意味します）
R1000:#1　（R1001を意味します）

ポイント
- **KV Nanoシリーズでは EM、FMは使用できません。**
- **KV Nanoシリーズでのみ、CMに対してインデックス定数による修飾ができます。**

・演算式によるインデックス修飾

デバイス（R（DR）/ MR / LR / B / T / C / DM / W / TM / CM / ＊DM / ＊ W / ＊TM）の後に演算式を続けて入力します。

例

DM0:（DM100＋20）　（DM100の値が50なら、DM70を意味します）
R1000:（DM0＊4）　（DM0の値が2なら、R1008を意味します）

ポイント
- **演算式によるインデックス修飾はKVスクリプト内でのみ使用できます。**
- **KV Nanoシリーズでは EM、FMは使用できません。**
- **KV Nanoシリーズでのみ、CMに対して演算式によるインデックス修飾ができます。**

●間接指定（＊）の記述

デバイス（DM/EM/FM/ZF/W/TM）の前に ＊ を入力します。間接指定命令とあわせて使用します。

例

＊DM0　＊TM100

「間接指定」3-57ページ

ポイント
KV Nanoシリーズでは EM、FMは使用できません。

●ローカルデバイスの記述

デバイス(R/MR/LR/T/C/DM/EM/FM/TM)の前に @ を入力します。
モジュール･マクロ内のみで有効なデバイスです。間接指定やインデックス修飾も可能です。

例

@DM0　@R10000

ポイント　**KV Nanoシリーズでは EM、FMは使用できません。**

●XYM表記について

デバイス(R(DR)/MR/LR/DM/EM/FM)については、下表のXYM表記によるデバイスの指定が可能です。

デバイス名	キーエンス表記	XYM表記	XYM表記時のデバイス番号
リレー	R(DR)	X(DX)	0000～999F※1
		Y(DY)	0000～999F※1
内部補助リレー	MR	M	00000～63999※1 ※2
ラッチリレー	LR	L	0000～999F※1
データメモリ	DM	D	00000～65534※1
拡張データメモリ	EM	E	00000～65534
ファイルレジスタ	FM	F	00000～32767

※1　KV Nanoの場合使用できるデバイス番号の範囲が異なります。
X(DX):0000～599F
Y(DY):0000～599F
M:00000～15999
L:0000～199F
D:00000～32767

※2　KV-5500/5000/3000シリーズの場合使用できるデバイス番号の範囲が異なります。
M:00000～15999

XYM表記にするには、《KV STUDIO》のメニューから【ツール】→【オプション】を選択して、[表示モードの設定]で「XYMで表示」にチェックを付けます。

ポイント　**KV Nanoシリーズでは EM、FMは使用できません。**

●ワードデバイスのビット指定の記述

デバイス(R(DR) / MR / LR / B / T / C / DM / EM / FM / ZF / W / TM / CM)の後に.□(□:0~15の数値)を続けて入力します。

例

DM0.0　(DM0の1ビット目を意味します)
EM10.4　(EM10の5ビット目を意味します)

「ワードデバイスのビットデバイス扱い」(3-46ページ)

ポイント　**KV Nanoシリーズでは EM、FMは使用できません。**

●サフィックスの記述

デバイス(R(DR)/ MR / LR / B / T / C / DM / EM / FM / ZF / W / TM / CM / ＊DM / ＊EM / ＊FM / ＊ZF / ＊ W / ＊TM)の後に.□(□:U,S,D,L,F,DF,B,T)を続けて入力します。

例

DM0.F	**(DM0～DM1の値を浮動小数点型データとして演算します)**
R1000.U	**(R1000～R1015までの16ビットデータとして演算します)**

ポイント

- **サフィックスの記述はKVスクリプト内でのみ使用できます。**
- **KV NanoシリーズではEM、FMは使用できません。**

●定数値の型指定

KVスクリプトでは、10進定数にサフィックスを付けることで、定数がもてる範囲を設定することができます。

型	サフィックス	データの範囲	記述例
16 ビット符号無し	定数 U	0 ～ 65535	910U, 505U
16 ビット符号付き	定数 S	－ 32768 ～＋ 32767	23S, － 600S
32 ビット符号無し	定数 D	0 ～ 4294967295	1234567D, 5D
32 ビット符号付き	定数 L	－ 2147483648 ～＋ 2147483647	123L, － 89335963L
32 ビット単精度浮動小数点型	定数 F	－ 3.4E ＋ 38 ≦ N ≦－ 1.4E － 45 N ＝ 0 ＋ 1.4E － 45 ≦ N ≦＋ 3.4E ＋ 38 (※KV Nano シリーズの場合、絶対値が 1.4E － 45 ～ 1.2E － 38 の範囲は使用できません。)	100F, － 3.4E38F, 0.12F
64 ビット倍精度浮動小数点型	定数 DF	－ 1.79E ＋ 308 ≦ N ≦－ 2.23E － 308 N ＝ 0 ＋ 2.23E － 308 ≦ N ≦＋ 1.79E ＋ 308	100DF, － 1.7E123DF 0.0256DF

※ KVスクリプトで扱う16進定数の型は、桁数により決まります。(サフィックスは記述できません)
4桁以下…16ビット符号無し(.U)
5桁以上…32ビット符号無し(.D)

ビットデバイス

■ビットの状態

ビットデバイスの状態を示すのに、以下の表記を用いることができます。

ビット状態	表　記	記述例
ON（真）	ON、1、TRUE	MR1000=ON、MR1001=1、MR1002=TRUE
OFF（偽）	OFF、0、FALSE	MR2000=OFF、MR2001=0、MR2002=FALSE

参 考　これら2状態を「ブール値」といいます。

入力/出力リレー　R

概要　入力リレーは、外部機器からPLCにON/OFF情報を取り込むためのデバイスです。
出力リレーは、PLCからPLCの外部機器へON/OFF指令や演算結果を送り出すためのデバイスです。

表記　デバイス番号は、次のような構成になっています。

例）

※1　CPUユニットがKV-1000、KV Nanoシリーズの場合、範囲は「000～599」になります。
※2　CPUユニットがKV-1000の場合、範囲は「000～59915」になります。

参 考
- ゼロサプレスして記述できます。
  例）R12
- サフィックスを付けて記述できます。
  例）R12.B
- サフィックスを省略して記述したとき、型宣言がなければ.B（ビット型）として扱います。
- ワードデバイスとして扱えます。
  例）R2000.U（R2000を先頭に16ビットを占有します）
- CPUユニットがKV-8000/7500/7300/5500/5000/3000、KV Nanoの場合、デバイス名に「D」を付けるとダイレクト入出力リレーとして動作します。
  例）DR500（R500はダイレクト出力リレーとして動作します）
  「ダイレクト入出力リレー」（3-47ページ）

範囲

- R000～R015※（入力専用）
- R500～R507（出力専用）
- R1000～R199915※（拡張入力ユニットの番号として割り付け可能）

※機種により使用できるリレー番号は異なります。

	入力専用	出力専用	拡張入力ユニットの番号
KV-8000	－	－	R0000～R199915
KV-7500/7300	－	－	R0000～R199915※1
KV-5500/5000/3000	R000～R015	R500～R507	R1000～R99915
KV-1000	R000～R015	R500～R507	R1000～R59915
KV-N14□□	R000～R007	R500～R505	R1000～R59915
KV-N24□□	R000～R013	R500～R509	R1000～R59915
KV-N40□□	R000～R107	R500～R515	R1000～R59915
KV-N60□□	R000～R203	R500～R607	R1000～R59915
KV-NC32T	R000～R015	R500～R515	R1000～R99915

※1　CPU機能バージョン2.2以前の場合は、R0000～R99915です。

用途

入力リレー

- 接続する外部機器として、ボタンやスイッチ（切り換えスイッチ、リミットスイッチ、光電スイッチ、デジタルスイッチ）などがあります。
- 割り込み･高速カウンタ･位置決め機能の各高速入力として使えます。
（使用できる入力リレーはユニットごとに異なります。）

KV-5500/5000/3000/1000	R000～R015
KV-N14□□	R000～R007
KV-N24□□	R000～R013
KV-N40□□/KV-N60□□	R000～R107
KV-NC32T	R000～R015

出力リレー

- 接続する外部機器として、ソレノイドや電磁開閉器、表示器（信号灯、デジタル表示器）などがあります。
- 高速カウンタ位置決め機能の各高速出力として使えます。
（使用できる出力リレーはユニットごとに異なります。）

KV-5500/5000/3000/1000	R500～R503
KV-N14□□/KV-N24□□	R500～R503
KV-N40□□	R500～R505
KV-N60□□	R500～R507
KV-NC32T	R500～R505

詳細

- 入力端子1点につき入力リレーが1ビット割り当てられます。
- 出力端子1点につき出力リレーが1ビット割り当てられます。
- a接点（ノーマルオープン）、b接点（ノーマルクローズ）のどちらも指定できます。
- 同じデバイス番号の入力リレーを、1プロジェクト内で何度でも記述できます。
- R000～R915のうち、入力/出力で使用されていないリレーはシステム予約されているため使用しないでください。

初期状態

出力リレーは、デフォルトで、電源ON時、PROG→RUN時に状態がクリアされてOFFになります。

- 対応機種（使用するCPUユニット）がKV8000/-7500/7300/5500/5000/3000、KV Nanoシリーズの場合、CPUシステム設定の［停電保持設定］により状態を保持する設定が可能です。
- 対応機種（使用するCPUユニット）がKV-1000の場合、MEMSW命令により状態を保持する設定が可能です。

参 考

・リレー(R)のユニット用デバイス表記について
《KV STUDIO》では、通常の表記以外に、ユニット番号を基準としたリレー表記ができます。ユニット用デバイス表記を使用するには、メニューから【表示(V)】→【ユニット用デバイス表示(Y)】を選択します。

ユニット番号　：CPUを先頭に00、01、02、03、04…と、ユニットの接続順に割り付けられます。

ユニットリレー番号：00000から占有するリレー点数分がリレー番号として割り付けられます。

ユニット番号	00※	01	02	03	04	05
KV-U7	CPU ユニット	KV-B16XA 入力16点	KV-C32XA 入力32点	KV-B16TA 出力16点	KV-C32TA 出力32点	KV-H20 10CH占有
ユニット リレー番号	R00000～R00015（入力） R00500～R00507（出力）	00000～00015	00000～00115	00000～00015	00000～00115	00000～00915

※ CPU(基本)ユニットで使用されるR000～R915までのデバイスはユニット用デバイス表記されません。

リンクリレー　B

概要

対応機種(使用するCPUユニットや基本ユニット)が、KV-1000の場合、使用できません。
EtherNet/IPやFL-net接続時などで、リンクリレーとして使用します。他のリレーと異なり、16進表記でデバイス番号を指定します。

表記

デバイス番号は、次のような構成になっています。

例）

参 考

・ゼロサプレスして記述できます。
例）B7A
・サフィックスを付けて記述できます。
例）B7A.B
・サフィックスを省略して記述したとき、型宣言がなければ.B(ビット型)として扱います。
・ワードデバイスとして扱えます。
例）B80.U(B80を先頭に16ビットを占有します)

ポイント

- **16進数(0～F)で指定してください。**
- **ローカルデバイス(@)は、使用できません。**

範囲

B0000～B7FFF※

用途

EtherNet/IPやFL-net接続時などでは、リンクリレーとして使用できます。

詳細
- 拡張ユニットのユニットデバイスに割り付けできません。
- 同じデバイス番号のリンクリレーBを、1プロジェクト内で何度でも記述できます。

初期状態 リンクリレーは、デフォルトで、電源ON時、PROG→RUN時に状態がクリアされてOFFになります。CPUシステム設定の[停電保持設定]により状態を保持する設定が可能です。

※ 機種により使用できるリレー番号は異なります。

KV-8000	B0000～B7FFF
KV-7500/7300	B0000～B7FFF
KV-5500/5000/3000	B0000～B3FFF
KV-1000	---
KV Nanoシリーズ	B0000～B1FFF

内部補助リレー　R・B・MR・LR

■R

概要 実入出力リレーと拡張ユニットリレーに割り当てていない R を内部補助リレーとして使用できます。

表記 デバイス番号は、次のような構成になっています。

例）

チャンネル番号（000～999※1）　接点番号（00～15）

R00012

デバイス名　デバイス番号（00000～99915※2）

※1　CPUユニットがKV-1000、KV Nanoシリーズの場合、範囲は「000～599」になります。

※2　CPUユニットがKV-1000、KV Nanoシリーズの場合、範囲は「000～59915」になります。

参考
- ゼロサプレスして記述できます。
  例）R12
- サフィックスを付けて記述できます。
  例）R12.B
- サフィックスを省略して記述したとき、型宣言がなければ.B（ビット型）として扱います。
- ワードデバイスとして扱えます。
  例）R2000.U（R2000を先頭に16ビットを占有します）

範囲
- R1000～R199915※（拡張入力ユニットの番号として割り付け可能）

  ※ 機種により使用できるリレー番号は異なります。

KV-8000	R000～R199915
KV-7500/7300	R000～R199915※1
KV-5500/5000/3000	R1000～R99915
KV-1000	R1000～R59915
KV Nanoシリーズ	R1000～R59915

※1　CPU機能バージョン2.2以前の場合は、R0000～R99915です。

■B

概要 対応機種（使用するCPUユニット）がKV-1000の場合使用できません。
EtherNet/IPやFL-netに接続しない場合や、リンクリレーに割り当てていないBを内部補助リレーとして使用できます。

表記 デバイス番号は、次のような構成になっています。
例）

チャンネル番号(000～7FF)　接点番号(0～F)

B1EE7

デバイス名　デバイス番号(0000～7FFF)

参考
- ゼロサプレスして記述できます。
  例）B7A
- サフィックスを付けて記述できます。
  例）B7A.B
- サフィックスを省略して記述したとき、型宣言がなければ.B（ビット型）として扱います。
- ワードデバイスとして扱えます。
  例）B80.U（B80を先頭に16ビットを占有します）

ポイント
- **16進数（0～F）で指定してください。**
- **ローカルデバイス（@）は、使用できません。**

範囲 B0000～B7FFF※

詳細
- 拡張ユニットのユニットデバイスに割り付けできません。
- 同じデバイス番号のリンクリレーBを、1プロジェクト内で何度でも記述できます。

初期状態 デフォルトでは、電源ON時、PROG→RUN時に状態がクリアされてOFFになります。CPUシステム設定の［停電保持設定］により状態を保持する設定が可能です。

※ 機種により使用できるリレー番号は異なります。

KV-8000	B0000~B7FFF
KV-7500/7300	B0000~B7FFF
KV-5500/5000/3000	B0000~B3FFF
KV-1000	---
KV Nanoシリーズ	B0000~B1FFF

■MR

概要 CPU内部のみで使用できるデバイスです。

表記 デバイス番号は、次のような構成になっています。

例）

チャンネル番号（000～999）　接点番号（00～15）

MR02809

デバイス名　デバイス番号（00000～99915）

参考
- ゼロサプレスして記述できます。
  例）MR2809
- サフィックスを付けて記述できます。
  例）MR2809.B
- サフィックスを省略したとき、型宣言がなければ.B（ビット型）として扱います。
- ワードデバイスとして扱えます。
  例）MR2900.U（MR2900を先頭に16ビットを占有します）

範囲 MR00000～MR399915※

ポイント
初期状態で KV-8000/7000 シリーズの場合：MR200000 ～ MR399915、KV-5000/3000/1000、KV Nanoシリーズの場合：MR50000～MR99915※は、ローカルデバイスに割り当てられているため使用できません。
（CPUシステム設定の［ローカル全体割付設定］で予約領域を変更できます）

詳細
- 拡張ユニットのユニットデバイスに割り付けできません。
- 同じデバイス番号の内部補助リレーMRを、1プロジェクト内で何度でも記述できます。

初期状態 電源ON時、PROG←→RUN時に状態がクリアされてOFFになります。

※ 機種により使用できるリレー番号は異なります。
KV-8000:
MR00000～MR399915（初期状態でMR200000~MR399915はローカルデバイス）
KV-7500/7300:
MR00000～MR399915（初期状態でMR200000~MR399915はローカルデバイス）
KV-5500/5000/3000/1000:
MR00000～MR99915（初期状態でMR50000~MR99915はローカルデバイス）
KV Nanoシリーズ:
MR00000～MR59915（初期状態でMR50000~MR59915はローカルデバイス）

■LR（ラッチリレー）

概要 CPU内部のみで使用でき、状態を保持するデバイスです。

表記 デバイス番号は、次のような構成になっています。

例）

参考
- ゼロサプレスして記述できます。
  例）LR2308
- サフィックスを付けて記述できます。
  例）LR2308.B
- サフィックスを省略したとき、型宣言がなければ.B（ビット型）として扱います。
- ワードデバイスとして扱えます。
  例）LR2400.U（LR2400を先頭に16ビットを占有します）

範囲 LR00000～LR99915※

詳細
- 拡張ユニットのユニットデバイスに割り付けできません。
- 同じデバイス番号のラッチリレーLRを、1プロジェクト内で何度でも記述できます。

初期状態 前回電源をOFFしたときの直前の状態が保持されます。

※ 機種により使用できるリレー番号は異なります。
KV-8000/7500/7300/5500/5000/3000/1000: LR00000～LR99915
KV Nanoシリーズ: LR00000～LR19915

タイマ（接点） T

概要 タイマは、接点（1ビット）、現在値（32ビット）を持ちます。減算式タイマの現在値が0と一致（タイムアップ）すると、接点がONするデバイスです。

表記 デバイス番号は、次のような構成になっています。

例）

デバイス番号（0000～3999※）

T0501.B

デバイス名　サフィックス

参考
- ゼロサプレスして記述できます。
  例）T501.B
- サフィックスを付けて記述できます。タイマ接点として扱うときは.Bを、タイマの現在値を扱うときは.Dを付けてください。
- サフィックスを省略したとき、型宣言がなければ.B（ビット型）として扱います。

範囲 T0000.B～T3999.B※

詳細 タイマ関数には10μsタイマ･1msタイマ･10msタイマ･100msタイマの4種類があります。

初期状態 接点は電源ON時、PROG→RUN時に状態がクリアされてOFFになります。
現在値は電源ON時、PROG→RUN時に設定値と同じ値になります。
設定値は保持されます。

※ 機種により使用できるリレー番号は異なります。
KV-8000/7500/7300/5500/5000/3000/1000: T0000.B～T3999.B
KV Nanoシリーズ: T000.B～T511.B

■スクリプトでタイマ関数を動作させる時の記述方法

タイマ関数の各引数に、使用するタイマの実行条件（タイマのリセット用リレー）、タイマ番号、設定値（タイムアップするまでの時間）を指定することで、タイマの動作を記述します。

例）

タイマデバイス（T0000～T3999※）　設定値（0～4294967295, デバイス番号）

TMR (R1000, T501, 1000)

タイマ関数名（100ms減算タイマ）　タイマ実行条件

【タイマの種類による動作の違い】

タイマの種類（関数）	デバイス番号	設定値	ONするまでの時間
100ms減算タイマ（TMR）	T0～T3999※1	0～4294967295	n×100ms
10ms減算タイマ（TMH）			n×10ms
1ms減算タイマ（TMS）			n× 1ms
10μs減算タイマ（TMU）※2			n×10μs

※1　機種により使用できるデバイス番号は異なります。
KV-8000/7500/7300/5500/5000/3000/1000: T0000.B～T3999.B
KV Nanoシリーズ: T000.B～T511.B
※2　KV-1000では使用できません。

参考 設定値にはデバイス（32ビット符号無しデータ）や式を指定できます。
「TMR関数」（5-22ページ）

■タイマの誤差

タイマの誤差は、次のとおりです。

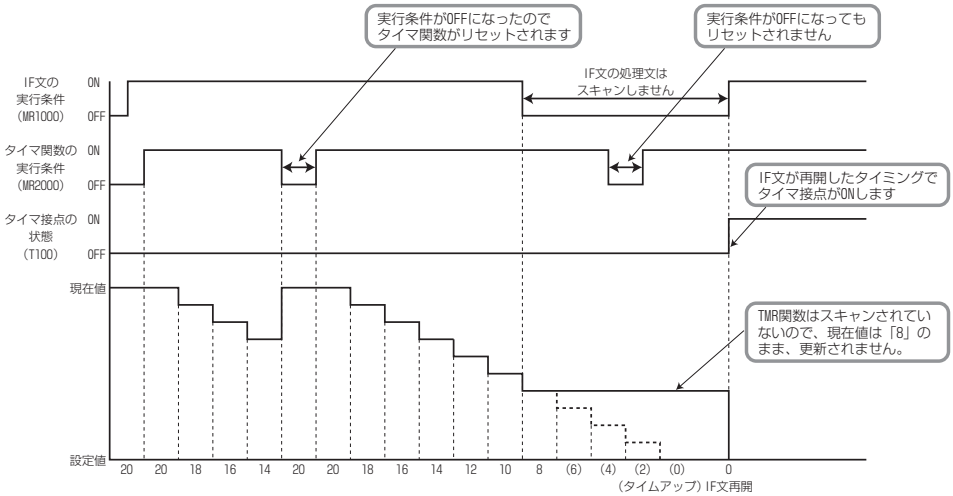
TMR（100msタイマ）　：±（100ms+1スキャンタイム）以内
TMH（10msタイマ）　：±（10ms+1スキャンタイム）以内
TMS（1msタイマ）　：±（1ms+1スキャンタイム）以内
TMU（10μsタイマ）　：±（10μs+1スキャンタイム）以内

■タイマ関数使用時の注意

IF文、ボックススクリプトでタイマ関数を使用する場合、タイマ関数実行中にIF文、ボックススクリプトを実行しなくすると、タイマの現在値は更新されません。
時間計測は引き続きされており、各命令間の実行が再開した時に経過した時間がタイマの現在値に反映されます。設定時間が経過していた場合には、通常は実行を再開した直後にタイマ接点がONします。

スクリプト記述

```
IF MR1000 = ON THEN
    TMR (MR2000 = ON, T100, 200)
END IF
```



■タイマの設定値・現在値を扱う方法

ラダー中のタイマの設定値･現在値をスクリプトで読み出したり、変更する方法を以下の表に示します。

		読み出し方法	変更方法
現在値		格納データメモリ＝タイマデバイス 例）DM1000.D＝T100.D	タイマデバイス＝変更する値（定数、デバイス） 例）T100.D＝DM2000.D
設定値	デバイス指定	格納データメモリ＝設定値格納デバイス	設定値格納デバイス＝変更する値（定数、デバイス）
	定数	設定値は読み出せません。	設定値は変更できません。

ポイント　**変更時に現在値＞設定値となる場合、現在値は設定値と同じ値に変更されます。**

カウンタ（接点） C

概要 カウンタは、接点（1ビット）、現在値（32ビット）を持ちます。
カウンタの現在値が設定値と同じになる（カウントアップする）と、接点がONするデバイスです。

表記 デバイス番号は、次のような構成になっています。
例）

デバイス番号（0000～3999※）

C0551.B

デバイス名　サフィックス

参考
- ゼロサプレスして記述できます。
  例）C551.B
- サフィックスを付けて記述できます。タイマ接点として扱うときは.Bを、タイマの現在値を扱うときは.Dを付けてください。
- サフィックスを省略したとき、型宣言がなければ.B（ビット型）として扱います。

範囲 C0000.B～C3999.B※

※ 機種により使用できるデバイス番号は異なります。
KV-8000/7500/7300/5500/5000/3000/1000: C0000.B～C3999.B
KV Nanoシリーズ: C000.B～C255.B

■スクリプトでカウンタ関数を動作させる時の記述方法

カウンタ関数の引数に使用するカウンタの実行条件（カウンタリセット）、カウンタ番号、設定値、カウント入力リレーを指定することでカウンタの動作を記述します。

カウンタデバイス番号　設定値

CNT（R1000，C600，50，MR1000）

カウンタ関数　カウンタ実行条件　カウント入力リレー

カウンタ関数は、カウント入力リレーがOFF→ONに変化（微分型）すると現在値を1ずつ加算し、現在値が設定値に達したらカウンタ接点をONします。

参考 設定値にはデバイス（32ビット符号無しデータ）や式を指定できます。
「CNT関数」（5-32ページ）

■カウンタの設定値・現在値を扱う方法

ラダー中のカウンタの設定値･現在値をスクリプトで読み出したり、変更する方法を以下の表に示します。

		読み出し方法	変更方法
現在値		格納データメモリ＝タイマデバイス 例）DM1000.D＝C200.D	タイマデバイス＝変更する値（定数、デバイス） 例）C200.D＝DM2000.D
設定値	デバイス指定	格納データメモリ＝設定値格納デバイス	設定値格納デバイス＝変更する値（定数、デバイス）
	定数	設定値は読み出せません。	設定値は変更できません。

ポイント **変更時に現在値＞設定値となる場合、現在値は設定値と同じ値に変更されます。**

高速カウンタコンパレータ（接点） CTC

概要 高速カウンタコンパレータは、接点（1ビット）と設定値を持ちます。
高速カウンタの現在値が設定値と一致した時にONするデバイスです。

ポイント **CTCを割込要因として使用している場合、CTC（接点）はONしません。**

範囲 CTC0～CTC7※

参考
- サフィックスを付けて記述できます。接点として扱うときは.Bを、設定値として扱うときは.Dを付けてください。
- サフィックスを省略したとき、型宣言がなければ.B（ビット型）として扱います。

詳細 CTCを割り込み要因として、割り込みプログラムを実行することもできます。

初期状態 デフォルトでは、値が保持されます。
- 対応機種（使用するCPUユニットや基本ユニット）がKV-5500/5000/3000、KV Nanoシリーズの場合、CPUシステム設定の［停電保持設定］により状態をクリアする（保持しない）設定が可能です。
- 対応機種（使用するCPUユニット）がKV-1000の場合、MEMSW命令により状態をクリアする（保持しない）設定が可能です。

※ 機種により使用できるデバイス番号は異なります。

KV-5500/5000/3000/1000	CTC0～CTC3
KV-N14□□/KV-N24□□	CTC0～CTC3
KV-N40□□	CTC0～CTC5
KV-N60□□	CTC0～CTC7
KV-NC32T	CTC0～CTC5

『KV-5500/5000/3000シリーズ ユーザーズマニュアル』「2-8 高速カウンタ」
『KV Nanoシリーズユーザーズマニュアル』「4-4高速カウンタ」

コントロールリレー（接点） CR

概要 PLCの機能を制御したり、PLCの状態を知るために使用する特殊な内部補助リレーです。

範囲 CR0000～CR7915※

参考
- サフィックスを付けて記述できます。
  例）CR2002.B
- サフィックスを省略したとき、型宣言がなければ.B（ビット型）として扱います。

詳細 「CR/CM一覧」（付-2ページ）

※ 機種により使用できるデバイス番号は異なります。

KV-8000	CR0000～CR7915
KV-7500/7300	CR0000～CR7915
KV-5500/5000/3000/1000	CR0000～CR3915
KV Nanoシリーズ	CR0000～CR8915

ワードデバイス

データメモリ/拡張データメモリ　DM/EM/FM

概要

データメモリと拡張データメモリは、関数が演算する際に参照するデータを保存したり、演算の結果を格納するための1点当たり16ビットのデバイスです。
対応機種（使用するCPUユニット）がKV-5500/5000/3000の場合、FMはファイルレジスタとして扱います。
「ファイルレジスタ　ZF/FM」（3-37ページ）

ポイント　**対応機種（使用する基本ユニット）が KV Nano シリーズの場合、EM、FM は使用できません。**

表記

デバイス番号は、次のような構成になっています。
例）

デバイス名　デバイス番号
DM06143

参考
- ゼロサプレスして記述できます。
  例）DM6143
- サフィックスを付けて記述できます。
  例）DM1000.S（16ビット符号付き）、DM2000.D（32ビット符号なし）
- サフィックスを省略して記述したとき、型宣言がなければ .U（16 ビット符号なし）として扱います。
- ビットデバイスとして扱えます。
  例）DM1100.B DM1100の最下位ビットを扱います
- 対応機種（使用する CPU ユニットや基本ユニット）が KV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合、ビット位置を指定したビットデバイスとして扱えます。
  例）DM1200.12 .....DM1200の13ビット目を扱います

範囲

- DM00000～DM65534
- EM00000～EM65534※1
- FM00000～FM32766※1、※2

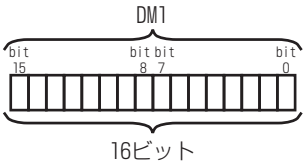
※1　対応機種（使用する基本ユニット）がKV Nanoシリーズの場合、使用できません。
※2　対応機種（使用するCPUユニット）がKV-5500/5000/3000の場合、範囲はFM00000～FM32767になります。
「ファイルレジスタ　ZF/FM」（3-37ページ）

詳細

- データメモリ（DM）のみ拡張ユニットのユニットデバイスに割り付けできます。

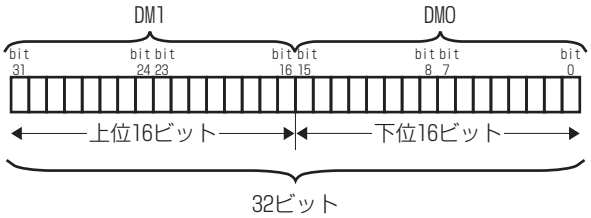
●16ビットデータを扱うとき

- デバイス1点のビット幅は16ビットで、符号無し10進数として扱う時は0～65535、符号付き10進数として扱う時は–32768～+32767、16進数として扱う時は$0～$FFFFの値が扱えます。
- 16ビットデータを扱う時は、デバイス番号に.U(省略可能)あるいは.Sのサフィックスを付けます。



●32ビットデータを扱うとき

- デバイス2点を使用して32ビットのデータを扱うこともできます。32ビットの場合、指定したデバイス番号のデータメモリに下位16ビット、その次のデバイス番号のデータメモリに上位16ビットを格納します。符号無し10進数として扱う時は0～4294967295、符号付き10進数として扱う時は–2147483648～+2147483647、16進数として扱う時は$0～$FFFFFFFFの値が扱えます。
- 32ビットデータを扱う時は、下位16ビットを示しているデバイス番号に.Dあるいは.Lのサフィックスを付けます。



初期状態 デフォルトでは、値が保持されます。

- 対応機種(使用する CPU ユニット)が KV-8000/7500/7300/5500/5000/3000、KV Nano シリーズの場合、CPUシステム設定の[停電保持設定]により状態をクリアする(保持しない)設定が可能です。
- 対応機種(使用するCPUユニット)がKV-1000の場合、MEMSW命令により状態をクリアする(保持しない)設定が可能です。

参考

- データメモリ(DM)のユニット用デバイス表記について
《KV STUDIO》では、通常の表記以外に、ユニット番号を基準とした表記ができます。
ユニット用デバイス表記を使用するには、メニューから【表示(V)】→【ユニット用デバイス表示(Y)】を選択します。

ユニット番号: CPUを先頭に00、01、02、03、04･･･と、ユニットの接続順に割り付けられます。

ユニットDM番号: 0000から占有するデータメモリ分が割り付けられます。

サフィックス: サフィックスを付けて記述できます。省略した場合は16ビット符号無し(.U)として扱われます。

例）

ユニット番号	00	01	02	03	04	05
KV-U7	KV-5000 CPU	KV-AD40 22ワード	KV-DA40 18ワード	KV-MC20V 402ワード	KV-MC20V 402ワード	KV-MC20V 402ワード
ユニット DM番号	0000～0019	0000～0021	0000～0017	0000～0401	0000～0401	0000～0401

KV-AD40の5番目 .....DM01_0004
KV-MC40V（ユニット番号03）の53番目 .....DM03_0052
KV-MC40V（ユニット番号05）の1番目 .....DM05_0000

ファイルレジスタ　ZF/FM

概要

対応機種（使用するCPUユニット）がKV-8000/7500/7300/5500/5000/3000の場合に使用できます。
ファイルレジスタは、すべての領域を連番方式で扱うZFと、領域を複数（16または4）のバンクに分割してバンク切り換え方式で扱うFMがあります。
詳細は使用しているCPUユニットのユーザーズマニュアルの「ファイルレジスタ」を参照してください。

ポイント

対応機種（使用するCPUユニット）がKV-1000の場合、FM（FM00000～FM32766）のみ扱えます。
「データメモリ/拡張データメモリ　DM/EM/FM」（3-35ページ）
対応機種（使用する基本ユニット）がKV Nanoシリーズの場合、ZF/FMは使用できません。

表記

デバイス番号は、次のような構成になっています。
例）

デバイス名　デバイス番号
ZF0372

参考

- ゼロサプレスして記述できます。
  例）ZF372
- サフィックスを付けて記述できます。
  例）ZF1000.S（16ビット符号付き）、FM12300.D（32ビット符号なし）
- サフィックスを省略して記述したとき、型宣言がなければ .U（16 ビット符号なし）として扱います。
- ビットデバイスとして扱えます。
  例）FM1100.B FM1100の最下位ビットを扱います
- 対応機種（使用するCPUユニット）がKV-8000/7500/7300/5500/5000/3000の場合、ビット位置を指定したビットデバイスとして扱えます。
  例）ZF1200.12 .....ZF1200の13ビット目を扱います

ポイント

- **ZFのローカルデバイス（@ZFxxx）は、使用できません。**
- **FMのローカルデバイス（@FM0～@FM32767）は、バンク切り換えできません。**
- **FMのローカルデバイス（@FM0～@FM32767）は、ワーク領域を使用します。**

範囲

- ZF000000～ZF524287※
- FM00000～FM32767※

※ 機種により使用できるデバイス番号は異なります。

KV-8000	ZF000000~ZF524287, FM00000~FM32767
KV-7500/7300	ZF000000~ZF524287, FM00000~FM32767
KV-5500/5000/3000	ZF000000~ZF131071, FM00000~FM32767
KV-1000	FM00000~FM32766
KV Nanoシリーズ	（使用不可）

「データメモリ/拡張データメモリ　DM/EM/FM」（3-35ページ）

詳細

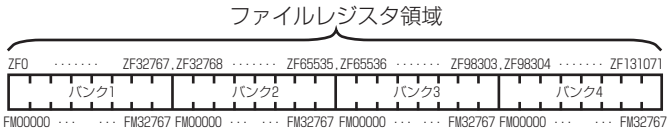
- 拡張ユニットのユニットデバイスに割り付けできません。
- 連番方式とバンク切り換え方式では、デバイスの種別と範囲が異なります。

●ZF：連番方式でファイルレジスタを扱うとき

ファイルレジスタの全領域を、1つのデバイスとして扱います。
対応機種（使用するCPUユニット）がKV-1000の場合、使用できません。

●FM：バンク切り換え方式でファイルレジスタを扱うとき

ファイルレジスタの領域を16分割※し、バンクを切り換えて扱います。
対応機種（使用するCPUユニット）がKV-1000の場合、バンク1のFM0～FM32766のみ使用できます。

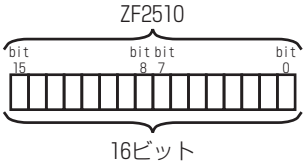


バンクを切り換えることで、1つのデバイスを16※の異なるデバイスとして使用できます。
ただし、ZFデバイスと重複して使用できるため、データの取り扱いには注意が必要です。
バンクの切り換え方法については、使用しているCPUユニットのユーザーズマニュアルの「ファイルレジスタ」を参照してください。

※ 機種により分割できるバンク数は異なります。
KV-8000:16
KV-7500/7300:16
KV-5500/5000/3000：4

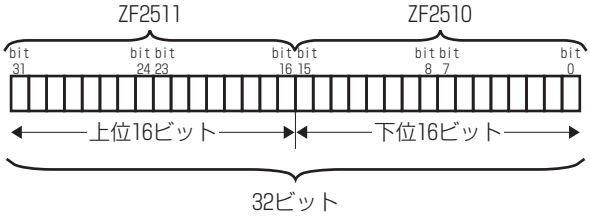
●16ビットデータを扱うとき

- デバイス1点のビット幅は16ビットで、符号無し10進数として扱う時は0～65535、符号付き10進数として扱う時は–32768～+32767、16進数として扱う時は$0～$FFFFの値が扱えます。
- 16ビットデータを扱う時は、デバイス番号に.U（省略可能）あるいは.Sのサフィックスを付けます。



●32ビットデータを扱うとき

- デバイス2点を使用して32ビットのデータを扱うこともできます。32ビットの場合、指定したデバイス番号のデータメモリに下位16ビット、その次のデバイス番号のデータメモリに上位16ビットを格納します。符号無し10進数として扱う時は0～4294967295、符号付き10進数として扱う時は–2147483648～+2147483647、16進数として扱う時は$0～$FFFFFFFFの値が扱えます。
- 32ビットデータを扱う時は、下位16ビットを示しているデバイス番号に.Dあるいは.Lのサフィックスを付けます。



初期状態 デフォルトでは、値が保持されます。

- 対応機種（使用するCPUユニット）がKV-8000/7500/7300/5500/5000/3000の場合、CPUシステム設定の［停電保持設定］により状態をクリアする（保持しない）設定が可能です。
- 対応機種（使用するCPUユニット）がKV-1000の場合、MEMSW命令により状態をクリアする（保持しない）設定が可能です。

リンクレジスタ　W

概要 対応機種（使用するCPUユニット）がKV-1000の場合使用できません。
EtherNet/IPやFL-net接続時などで、リンクレジスタとして使用します。他のワードデバイスと異なり、16進表記でデバイス番号を指定します。

表記 デバイス番号は、次のような構成になっています。
例）

デバイス名　デバイス番号（0～3FFF）

W3A6F

参考

- ゼロサプレスして記述できます。
  例）W7A
- サフィックスを付けて記述できます。
  例）W7A.S（16ビット符号付き）、W3E76.D（32ビット符号無し）
- サフィックスを省略して記述したとき、型宣言がなければ.U（16ビット符号無し）として扱います。
- ビットデバイスとして扱えます。ビット位置の指定も可能です。
  例）W1200.12 .....W1200の13ビット目を扱います

ポイント

- **16進数（0～F）で指定してください。**
- **ローカルデバイス（@）は、使用できません。**

範囲　W0000～W7FFF※

※　機種により使用できるデバイス番号は異なります。

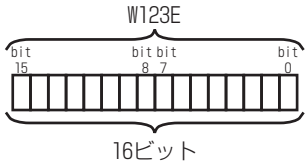
KV-8000	W0000～W7FFF
KV-7500/7300	W0000～W7FFF
KV-5500/5000/3000シリーズ	W0000～W3FFF
KV Nanoシリーズ	W0000～W3FFF

用途　EtherNet/IPやFL-net接続時などでは、リンクレジスタとして使用できます。

詳細　拡張ユニットのユニットデバイスに割り付けできません。

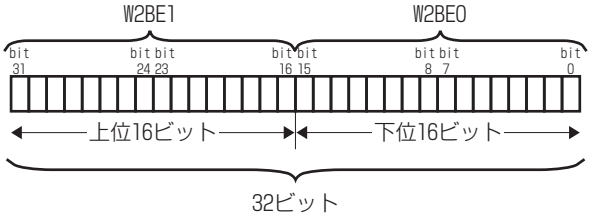
●16ビットデータを扱うとき

- デバイス1点のビット幅は16ビットで、符号無し10進数として扱う時は0～65535、符号付き10進数として扱う時は–32768～+32767、16進数として扱う時は$0～$FFFFの値が扱えます。
- 16ビットデータを扱う時は、デバイス番号に.U(省略可能)あるいは.Sのサフィックスを付けます。



●32ビットデータを扱うとき

- デバイス2点を使用して32ビットのデータを扱うこともできます。32ビットの場合、指定したデバイス番号のデータメモリに下位16ビット、その次のデバイス番号のデータメモリに上位16ビットを格納します。符号無し10進数として扱う時は0～4294967295、符号付き10進数として扱う時は–2147483648～+2147483647、16進数として扱う時は$0～$FFFFFFFFの値が扱えます。
- 32ビットデータを扱う時は、下位16ビットを示しているデバイス番号に.Dあるいは.Lのサフィックスを付けます。



初期状態　デフォルトでは、値が保持されます。
CPUシステム設定の[停電保持設定]により状態をクリアする(保持しない)設定が可能です。

テンポラリデータメモリ　TM

概要　テンポラリデータメモリは、データメモリと同じように扱うことができる1点当たり16ビットのワードデバイスです。一部のデバイス領域は、演算のためのデータや演算の結果データを一時格納するためにも使用します。

表記　デバイス番号は、次のような構成になっています。

例）

デバイス名　デバイス番号

TM010

参考
- TM000 ～ TM003 はラダー中の DIV、MUL などの演算命令で、また、TM000、TM001 は 32 ビットを超えるデータを扱うときに使用するため、それ以外の用途で使用しないことをおすすめします。
- ゼロサプレスして記述できます。
  例）TM10
- サフィックスを付けて記述できます。
  例）TM10.D（32ビット符号無し）
- サフィックスを省略したとき、型宣言がなければ.U（16ビット符号無し）として扱います。

範囲　TM000～TM511

TM番号	用　途
TM000･TM001	64ビットデータ用
TM002･TM003	除算の余り格納用
TM004～TM511	汎用※

※スクリプトで簡易間接指定は使用できません。

初期状態　電源ON時、PROG→RUN時に値がクリアされて0になります。

タイマ（現在値）　T

概要　タイマデバイスにサフィックス.Dを付けることで、現在値を扱います。

範囲　T0000.D～T3999.D※

参考　タイマデバイスを現在値として扱うときは、サフィックス.Dを省略できません。サフィックスを省略した場合、型宣言がなければ.B（タイマ接点）として扱います。

詳細　タイマデバイスの現在値は、0～4294967295の範囲で変更できます。

ポイント　**現在値変更時に現在値＞設定値となる場合、現在値は設定値と同じ値に変更されます。**

「タイマ（接点）　T」（3-31ページ）

※ 機種により使用できるデバイス番号は異なります。
KV-8000/7500/7300/5500/5000/3000/1000: T0000.D～T3999.D
KV Nanoシリーズ: T000.D～T511.D

カウンタ（現在値） C

概要 カウンタデバイスにサフィックス.Dを付けることで、現在値を扱います。

範囲 C0000.D～C3999.D※

参考 カウンタデバイスを現在値として扱うときは、サフィックス.Dを省略できません。サフィックスを省略した場合、型宣言がなければ.B（カウンタ接点）として扱います。

詳細 カウンタデバイスの現在値は、0～4294967295の範囲で変更できます。

ポイント **現在値変更時に現在値＞設定値となる場合、現在値は設定値と同じ値に変更されます。**

※ 機種により使用できるデバイス番号は異なります。
KV-8000/7500/7300/5500/5000/3000/1000: C0000.D～C3999.D
KV Nanoシリーズ: C000.D～C255.D

「カウンタ（接点） C」（3-33ページ）

高速カウンタ（現在値） CTH

概要 高速カウンタは、カウンタ関数では取り込めない、間隔の短い信号を取り込むための32ビットカウンタです。

範囲 CTH0～CTH3※

参考
- サフィックスを付けて記述できます。
  例）CTH0.D
- サフィックスを省略したとき、型宣言がなければ.D（32ビット符号無し）として扱います。

詳細
- 0～4294967295の値が扱えます。
- 外部入力のカウントのほかに、内部クロックをカウントできます。
- 最大応答周波数は、デューティー比50%で、単相100kHz、位相差50kHzです。

初期状態 デフォルトで、電源ON時、PROG→RUN時に状態がクリアされてOFFになります。
- 対応機種（使用するCPUユニットや基本ユニット）がKV-5500/5000/3000、KV Nanoシリーズの場合、CPUシステム設定の［停電保持設定］により状態を保持する設定が可能です。
- 対応機種（使用するCPUユニット）がKV-1000の場合、MEMSW命令により状態を保持する設定が可能です。

※ 機種により使用できるデバイス番号は異なります。
KV-5500/5000/3000/1000:CTH0~CTH1
KV-N14□□/KV-N24□□:CTH0~CTH1
KV-N40□□:CTH0~CTH2
KV-N60□□:CTH0~CTH3
KV-NC32T:CTH0～CTH2

『KV-5500/5000/3000シリーズ ユーザーズマニュアル』「2-8 高速カウンタ」
『KV Nanoシリーズユーザーズマニュアル』「4-4高速カウンタ」

高速カウンタコンパレータ（設定値） CTC

概要 CTCにサフィックス.Dを付けることで、高速カウンタの設定値（32ビット）を扱います。

範囲 CTC0～CTC7※

参考 高速カウンタコンパレータデバイスを設定値として扱うときは、サフィックス .D を省略できません。サフィックスを省略した場合、型宣言がなければ.B（接点）として扱います。

詳細
- 0～4294967295の範囲で設定できます。
- 高速カウンタ（CTH）1点につき、2点の設定値（CTC）が設定できます。
  CTC0、CTC1 ･･･ CTH0用
  CTC2、CTC3 ･･･ CTH1用
  CTC4、CTC5 ･･･ CTH2用※
  CTC6、CTC7 ･･･ CTH3用※

初期状態 値が保持されます。

※ 機種により使用できるデバイス番号は異なります。
KV-5500/5000/3000/1000:CTC0~CTC3
KV-N14□□/KV-N24□□:CTC0~CTC3
KV-N40□□:CTC0~CTC5
KV-N60□□:CTC0~CTC7
KV-NC32T:CTC0～CTC5

『KV-5500/5000/3000シリーズ ユーザーズマニュアル』「2-8 高速カウンタ」
『KV Nanoシリーズユーザーズマニュアル』「4-4高速カウンタ」

インデックスレジスタ　Z

概要　インデックス修飾する際に、対象デバイス番号に加える数を指定するための1点当たり32ビットのワードデバイスです。データメモリと同じように、データを保存したり、演算の結果を格納できます。

範囲　CPU機能バージョン2.0以降のKV-8000 ：Z1～Z23
上記以外の機種 ：Z1～Z12

参考
- サフィックスを付けて記述できます。
  例）Z01.L（32ビット符号付き）
  Z02.D（32ビット符号なし）
- サフィックスを省略して記述したとき、型宣言がなければ対応機種が KV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合.L（32ビット符号付き）として、対応機種がKV-1000の場合.S（16ビット符号付き）として扱います。

ポイント
- **Z11以降は、システムで予約されているため使用できません。**
- **サフィックスに.F、.DF、.T、.Bは指定できません。**
- **ワードデバイスのビット指定はできません。**
- **ローカルデバイスはありません。ZPUSH、ZPOP関数を使用してください。**
  「ZPUSH関数」（6-20ページ）
  「ZPOP関数」（6-24ページ）

初期状態　電源ON時、PROG→RUN時に値がクリアされて0になります。

「インデックス修飾」（3-55ページ）

コントロールメモリ　CM

概要　PLCの機能を制御したり、PLCの状態を知るために使用する特殊なワードデバイスです。

範囲　CM0000～CM7599※

参考
- サフィックスを付けて記述できます。
  例）CM1720.U
- サフィックスを省略したとき、型宣言がなければ.U（16ビット符号無し）として扱います。

※ 機種により使用できるデバイス番号は異なります。

KV-8000	CM0000～CM7599
KV-7500/7300/5500/5000/3000	CM0000～CM5999
KV-1000	CM00000～CM11998
KV Nanoシリーズ	CM0000～CM8999

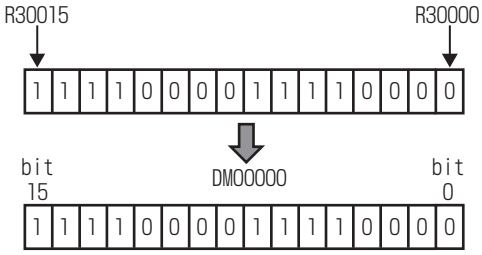
詳細　「CR/CM一覧」（付-2ページ）

ビットデバイスのワード単位扱い

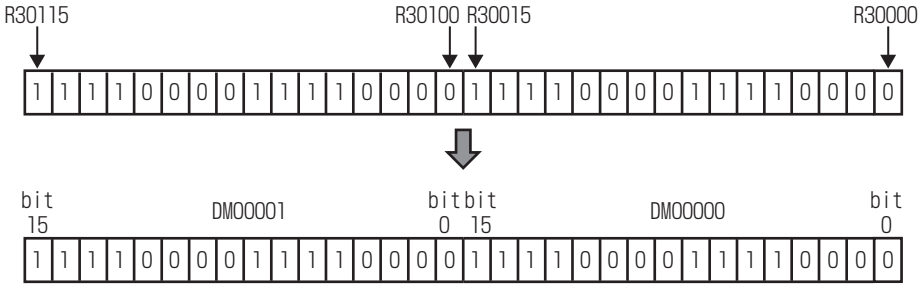
ワードデバイスを使用する関数や代入文などに、.B以外のサフィックスを付加したビットデバイスを指定すると、指定したビットデバイスを先頭としたワード単位に扱われます。

例）

DM0000.U = R3000.U

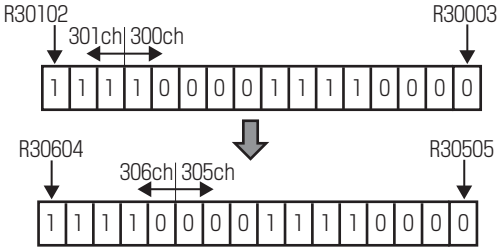


DM0000.D = R3000.D



サフィックスを付加したビットデバイスがチャンネルの先頭以外の時は、次のチャンネルにまたがったワード単位として扱われます。

R30505.U = R30003.U



ワードデバイスのビットデバイス扱い

通常、ワードデバイスは、16ビット/32ビットのデバイスとして処理されますが、ビット型のサフィックスを付加することで、ビットデバイスとして扱えます。

ポイント　**高速カウンタ(CTH)とインデックスレジスタ(Z)は、ビットデバイス扱いできません。**

●最下位ビットでビットデバイスとして扱う

ワードデバイスに.Bサフィックスを付加したり、ビットデバイスを処理する接点･出力関数の引数や代入文などにワードデバイスを指定すると、指定したワードデバイスの最下位ビットの状態でビットデバイスとして扱います。

DM1000の値が#46563($B5E3)の場合

DM1000＝#46563($B5E3)

1	0	1	1	0	1	0	1	1	1	1	0	0	0	1	1

最下位ビット
1：接点ON
0：接点OFF

DM1000の最下位ビットが1なので、DM1000.Bは「ON」として扱います。

●任意のビット位置でビットデバイスとして扱う

対応機種(使用するCPUユニットや基本ユニット)がKV-1000の場合使用できません。
ワードデバイスに「.0～.15」のビット位置を指定すると、ワードデバイスの指定したビット位置の状態でビットデバイスとして扱います。

表記　例)

DM1234.12

任意のワードデバイス　ビット位置（.0～.15）

参考
- DM123.BとDM123.0は、同じビットを扱います。
- デバイスコメントは、ワードデバイスに対して設定されたコメントが表示されます。

ポイント

- **対応機種(使用するCPUユニット)がKV-1000の場合、使用できません。**
- **ビット位置は、10進数(0～15)で指定してください。**
- **ローカルデバイス(@)は、使用できません。**
- **ビット指定可能なデバイスは、「DM、W、EM、FM、ZF、TM、CM」のみです。**
- **ビット指定可能な変数は、データ型が「INT、DINT、UINT、UDINT」の変数、およびこれらのデータ型に設定した配列型変数の要素、構造体型変数のメンバのみです。**
- **ラベルに対して、ビット位置の指定はできません。(グローバルラベル登録時に、ビット指定されたワードデバイスを指定することはできます)**
- **マクロ引数デバイスに対して、ビット位置指定によるビット扱いはできません。**
- **ビット位置指定されたデバイスは、マクロ関数の引数として設定できません。**
- **ビット位置指定されたデバイスは、間接参照やインデックス修飾できません。**

DM123の値が#59($3B)の場合

bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	0	0	0	0	0	0	0	0	0	1	1	1	0	1	1

DM123をビット位置指定でビットデバイスとして扱った場合、各ビットの状態は以下の通りです。

デバイス	ビットの状態
DM123.0	ON
DM123.1	ON
DM123.2	OFF
DM123.3	ON
DM123.4	ON
DM123.5	ON
DM123.6～DM123.15	OFF

ダイレクト入出力リレー

対応機種(使用するCPUユニットや基本ユニット)がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合、入出力リレー(R)のデバイス番号の先頭に「D」を付けることで、ダイレクト入出力リレーとして動作します。

表記　例)

ダイレクト入出力リレーの表記

DR001

入出力リレー

ポイント

- **対応機種(使用するCPUユニット)がKV-1000の場合、ダイレクト入出力リレーは使用できません。**
- **対応機種(使用する基本ユニット)がKV　Nanoシリーズの場合、拡張ユニットにはダイレクト入出力リレーは使用できません。**
- **.B以外のサフィックスは付加できません。**

範囲　入出力リレー

KV-5500/5000/3000	DR000～DR015/	DR500～DR503/ 拡張入出力リレー
KV-N14□□	DR000～DR007/	DR500～DR505
KV-N24□□	DR000～DR013/	DR500～DR509
KV-N40□□	DR000～DR107/	DR500～DR515
KV-N60□□	DR000～DR203/	DR500～DR607
KV-NC32T	DR000～DR015/	DR500～DR515

ダイレクト入出力リレーの詳細は、『KV-5500/5000/3000シリーズ ユーザーズマニュアル』、『KV Nanoシリーズユーザーズマニュアル』を参照してください。

変数

変数とは、データ型と名前が定義された、プログラミングで使用できる要素のことを言います。KV STUDIOでは、プログラミングにデバイスもしくは変数を使用できます。
変数は、デバイスと異なりKV STUDIOで定義した名前でプログラミングでき、定義した変数は自動的にCPUユニットのワーク領域に割り付きます。そのため、各機能が割り当たっているデバイス番号を管理するためだけにデバイスマップを作成するなどの作業が不要になります。また、VT5シリーズおよびSoft-VTも変数に対応しているため、PLC側のデバイス番号に合わせてタッチパネル側のデバイス番号を変更する作業も不要になります。

ポイント

- **変数を使用できるのは、KV STUDIO Ver11.0以降で、対応機種がCPU機能バージョン2.0以降のKV-8000の場合のみです。**
- **タッチパネルから変数を読み書きできるのは、VT STUDIO Ver.8.0以降で、対応機種がVT5シリーズで、PLC機種がKV-8000（イーサネット）の場合のみです。**

変数の設定方法については、「変数プログラミングと配列・構造体」（2-25ページ）を参照してください。

変数とデータ型一覧

変数を使用する場合、あらかじめデータ型を定義する必要があります。CPUユニットは定義されたデータ型に応じて必要なデータサイズを自動的にワーク領域に割り付けます※1。
KV-8000で使用できるワーク領域のサイズと、変数の仕様は以下のとおりです。

項　目		内　容
ワーク領域のサイズ	ビット	64000点
	ワード	589824ワード
	タイマ	4000点
	カウンタ	4000点
変数の仕様	定義可能数	100000
	変数名の最大長	128文字

※1　ワーク領域ではなく、CPU ユニットのデバイスに手動で割り付けることも可能です。詳細は『KV-8000シリーズ　ユーザーズマニュアル』を参照してください。

ポイント

BOOL型の配列の先頭はchの先頭に割りつかない場合があります。そのため、chの先頭を指定する必要があるオペランドにBOOL型配列を指定する場合は、以下のいずれかの方法でchの先頭に割り付けた状態で指定してください。

- **CPUユニットのデバイスに割り付ける。**
- **構造体のメンバにBOOL型配列を作成し、chの先頭に割り付ける。**

KV-8000で使用できる変数のデータ型と、データサイズは以下のとおりです。

データ型		データサイズ（ビット単位）
型　名	説　明	
BOOL	ビット	1
UINT	1ワード符号なし整数	16
INT	1ワード符号あり整数	16
UDINT	2ワード符号なし整数	32
DINT	2ワード符号あり整数	32
REAL	単精度浮動小数点型	32
LREAL	倍精度浮動小数点型	64
STRING	文字列	（文字数+1）×8 を16の整数倍に切り上げ
タイマ	タイマ	―
カウンタ	カウンタ	―
配列	―	データ型 × 配列サイズ
構造体	―	定義による※1
ファンクションブロック	ファンクションブロック	定義による※1

※1　構造体およびファンクションブロックは複数のデータ型を設定して使用するデータ型です。そのため、設定内容によりデータサイズが決まります。詳細は『KV-8000シリーズ　ユーザーズマニュアル』を参照してください。

定数

使用する関数やデバイスに付属するサフィックス、変数のデータ型によって、表記可能な定数やその範囲が変わります。

(#)10進定数

概要　16ビットと32ビットのビット幅で表される数値を10進表記で示します。

表記　10進表記の数値をそのまま表記します。(符号付き表記も可能です)

例)

数値
06143

参考
- ゼロサプレスして記述できます。
- 「#」を付けて記述できます。符号付きの場合、「+」のみ符号の省略が可能です。
- XYM表示モードでは、「K」を付けて記述できます。

範囲
- 16ビット符号無しの場合:0～65535
- 32ビット符号無しの場合:0～4294967295
- 16ビット符号付きの場合:−32768～+32767
- 32ビット符号付きの場合:−2147483648～+2147483647

参考　定数にサフィックスを付けて記述することで、定数の範囲が拡張します。(通常記述では定数にサフィックスを付ける必要はありません)

300U	:16ビット符号無し(0～65535)
500S	:16ビット符号付き(−32768～+32767)
3000D	:32ビット符号無し(0～4294967295)
5000L	:32ビット符号付き(−2147483648～+2147483647)
123F	:単精度浮動小数点型(−3.4E+38≦N≦1.4E−45 N＝0 1.4E−45≦N≦3.4E+38) (有効桁数:約7桁)
1.2345678DF	:倍精度浮動小数点型(−1.79E+308≦N≦−2.23E−308 N＝0 2.23E−308≦N≦1.79E+308) (有効桁数:約16桁)

ポイント
- **定数に付けるサフィックスは、「.」(ピリオド)を付けずに記述します。**
- **KV Nanoシリーズの場合、非正規化された単精度浮動小数点型を使用時に演算エラーが発生します。**

$16進定数

概要 16ビットと32ビットの2種類のビット幅で表される数値を16進表記で示します。

表記 先頭に「$」、続いて16進表記の数値を記述します。

例）

16進数を表す記号　数値

$1AF7

参考
- ゼロサプレスして記述できます。
- XYM表示モードでは、「H」を付けて記述します。

範囲
- 16ビットの場合：$0～$FFFF
- 32ビットの場合：$0～$FFFFFFFF

参考 16進数にはサフィックスはなく、16ビット/32ビットの判別は＄以下のケタ数で判断されます。
＄00FF（4桁以下）‥‥16ビット
＄0000 00FF（5桁以上）‥‥32ビット

ポイント
- **「$」をつけずにA～Fを含む16進数を記述した場合はエラーになり、A～Fを含まない場合は10進数と認識されます。**
- **16進定数は符号を付けて記述できません。**

単精度浮動小数点型定数

概要 単精度浮動小数点型実数として記述できます。

表記
- 符号を付けて10進表記の数値をそのまま表記します。

例）

数値

1234.56

- 8桁以上で指数表記になります。
- 10進定数にサフィックスを付けて表記します。

例）

数値　単精度浮動小数点型サフィックス

1234F

参考 サフィックスを付けずに表現することも可能です。

例）

1234.0

単精度浮動小数点型を示す

参考 「#」を付けて記述できます。「#」を付けた場合「+」のみ符号の省略が可能です。

範囲 −3.4E+38≦ N≦ −1.4E−45
N=0
1.4E−45≦ N≦ +3.4E+38（有効桁数：約7桁）
（KV Nanoシリーズの場合、絶対値が1.4E−45～1.2E−38の範囲は使用できません。）

倍精度浮動小数点型定数

概要 倍精度浮動小数点型実数として記述できます。

表記
- 符号を付けて10進表記の数値をそのまま表記します。

例）

数値

12345.6789

- 17桁以上で指数表記になります。
- 10進定数にサフィックスを付けて表記します。

例）

数値　倍精度浮動小数点型サフィックス

1234DF

参考 「#」を付けて記述できます。「#」を付けた場合「+」のみ符号の省略が可能です。

範囲 －1.79E+308≦N≦－2.23E－308
N=0
+2.23E－308≦N≦+1.79E+308（有効桁数：約16桁）
（KV Nanoシリーズの場合、絶対値が1.4E-45～1.2E-38の範囲は使用できません。）

固定文字列（文字列型定数）

概要 任意の文字列を「"」で囲むことで記述できます。

表記 「" "」の間に文字列を表記します。

例）

文字列

"演算結果"

参考
- 「"」を文字として扱うには、「""」と記述します。
- 「"」を他の文字に付けて扱う場合は、文字を「+」で結合して使用します。

DM1000.T = """" + " **演算結果** " + """" （DM1000.Tには「"演算結果"」が格納されます。）

ポイント **「" "」を付けないとラベル扱いになります。**

範囲 扱える文字列の最大は1999文字です。

インデックス修飾と間接指定

インデックス修飾と間接参照で使用できるデバイスは以下のとおりです。

デバイス			KV-8000/7500/7300/5500/5000/3000		KV-1000		KV Nano	
種類	記号	名称	インデックス修飾	間接指定	インデックス修飾	間接指定	インデックス修飾	間接指定
ビットデバイス	R	リレー	○	○※	○	○※	○	○※
	CR	コントロールリレー	–	–	–	–	–	–
	T	タイマ(接点)	○	○※	○	○※	○	○※
	C	カウンタ(接点)	○	○※	○	○※	○	○※
	MR	内部補助リレー	○	○※	○	○※	○	○※
	LR	ラッチリレー	○	○※	○	○※	○	○※
	B	リンクリレー	○	○※	–	–	○	○※
ワードデバイス	DM	データメモリ	○	○	○	○	○	○
	EM	拡張データメモリ	○	○	○	○	–	–
	FM	ファイルレジスタ(バンク方式)	○	○	○	○	–	–
	ZF	ファイルレジスタ(連番方式)	○	○	–	–	–	–
	W	リンクレジスタ	○	○	–	–	○	○
	TM	テンポラリメモリ	○	○	○	○	○	○
	TS	タイマ(設定値)	○	○	○	○	○	○
	TC	タイマ(現在値)	○	○	○	○	○	○
	CS	カウンタ(設定値)	○	○	○	○	○	○
	CC	カウンタ(現在値)	○	○	○	○	○	○
	CM	コントロールメモリ	–	–	–	–	○	○※

※ 間接指定はできますが、「*(デバイス)」と記述してアドレスの格納はできません。

ポイント

- **CR、CTC、CTH、Z、ラベル(ラベル配列)、変数(変数配列)、およびワードデバイスのビット指定デバイスはインデックス修飾できません。**
- **CR、CTC、CTH、Zおよびワードデバイスのビット指定デバイスは間接指定できません。**

インデックス修飾

インデックス修飾とは、各デバイスに、インデックスレジスタ(Z)の値や定数を加算した番号のデバイスを指定する方法です。
加算できる値の範囲は、符号付き32ビット(-2147483648～+2147483647)です。
使用できるデバイスはユニットごとに異なります。「インデックス修飾と間接参照で使用できるデバイス」(3-54ページ)を参照してください。
間接指定(*)、ローカルデバイス(@)と併用できます。

ポイント
- **ラベル(ラベル配列)およびワードデバイスのビット指定デバイスは、インデックス修飾できません。**
- **変数はインデックス修飾できません。ただし、グローバルデバイスに手動で割り付けた変数はインデックス修飾できます。**
- **インデックス修飾して使用可能範囲外のデバイスを引数に指定した場合は、コントロールリレーCR2012がONし、その関数は実行されません。運転は継続します。**

■インデックスレジスタによる修飾

デバイス番号を運転中に変更する場合に使用します。
インデックスレジスタによるデバイス修飾の記述方法は
(デバイス番号):Z(01～10)
となります。

例) DM1000:Z01 ―― Z01の値が-10の場合、DM0990を参照します。
R0000:Z05 ―― Z05の値が+16の場合、R0100を参照します。

ポイント **Z11以降は、システム予約されているため使用できません。**

例

DM1000に格納される現在値を、R0001の立ち上がりをトリガ信号として保存します。
運転開始時にインデックスレジスタを初期化し、R0001の立上り(OFF→ONの変化)で、DM0000から順番にデータを保存していきます。

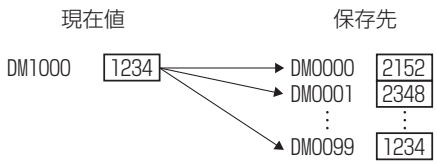
スクリプト記述

```
IF LDP(CR2008) THEN
    Z01=0
END IF
IF LDP(R0001) THEN
    DM0000:Z01 = DM1000
    INC(Z01)
END IF
```

運転開始時、インデックスレジスタZ01を0にします。

R0001の立ち上がりで、インデックス修飾されている対象デバイスにDM1000の値を書き込みます。

インデックスレジスタZ01をインクリメント(+1)します。



ポイント **インデックス修飾するデバイス(値)は、32ビット符号付きデータとして扱うため、型宣言をおこなっていないインデックスレジスタは自動的に「.L(32ビット符号付き)」サフィックスとして実行されます。**

参考 インデックスレジスタ (Z) をプロジェクト内で 10 個以上使用する必要がある場合は、インデックスレジスタ関数(ZPUSH/ZPOP)を使用して、モジュールやマクロごとにインデックスレジスタを一括で退避・復帰させます。

「ZPUSH関数」(6-20ページ)
「ZPOP関数」(6-24ページ)

■インデックス定数による修飾

定数を使用してインデックス修飾する場合の記述方法は

(デバイス番号):(#)(10進数表記の定数)

となります。(#は省略可能です)

例)

DM1000:10 ―― DM1000+10ワードでDM1010になります。

R0000:16 ―― R0000+16ビット(1チャンネル分)でR0100になります。

参考 ユニットデバイスを使用したラダープログラムを作成する場合に使用すると、割り付けられた先頭デバイスから何番目かがわかりやすくなります。また、マクロ作成時に、引数デバイスとしてUR/UMを使用する場合にも有効です。

例)

UR0:2

UM1:32

■デバイスや式による修飾

スクリプトでは、インデックスレジスタ以外でもインデックス修飾ができます。
この場合、ラダー変換時に自動的にシステムが予約しているインデックスレジスタ(Z11、Z12)を使用します。

スクリプト記述 DM1000 : (DM10 + 2) = 100

ラダー展開

※式でインデックス修飾するときは、必ず式を( )でくくってください。

■サフィックスの付いたデバイスの修飾

インデックス修飾するデバイスのサフィックスは以下のように記述してください。

DM1000.D:Z01.S

DM1100.F:(DM10+10)

■不正なインデックス修飾

次のような場合、インデックス修飾はできません。

- (DM1000:Z01):Z02・・・・・・ インデックス修飾したデバイスの修飾
- (DM1000+3):Z02.・・・・・・・ 演算結果に対する修飾
- DM1000:Z01+5・・・・・・・・・ インデックス修飾する式に( )がない
- Z01:Z02・・・・・・・・・・・・・・ インデックス修飾できないデバイスの修飾

間接指定

■間接指定(*)とは

間接指定(*)は、デバイスを直接指定するのではなく、デバイスが指し示す値(アドレス)が格納されたデバイスを記述することで、間接的に指定することです。

●間接指定の仕組み

各デバイスはCPUユニット内のメモリ上で管理され、それぞれに個別のアドレスを持っています。

- 間接指定(*)は、そのアドレスを専用関数で操作して参照するデバイスを変更します。
- インデックス修飾が「基準デバイス+相対オフセット値(インデックス定数)」で表記するのに対して、間接指定は「“*”+参照先アドレス格納デバイス」で表記します。
- インデックス修飾(:Z/:(#)10進定数)、ローカルデバイス(@)と併用できます。
  アドレスは32ビットで管理されるため、アドレス格納用にデバイスが2ワード必要になります。

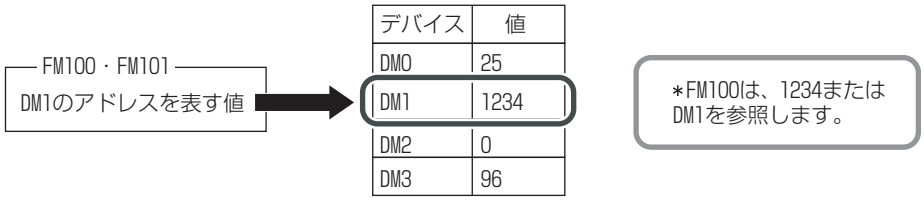
ポイント
- **間接指定して使用可能範囲外のデバイスを引数に指定した場合、その関数の動作は保証されません。**
- **演算式の項に使用可能範囲外のデバイスを指定した場合、演算結果は保証されません。**

●間接指定の記述方法

「*(デバイス)」と記述すると、そのデバイスに格納されているアドレスに該当するデバイスまたは、デバイス値を参照します。

例)

＜DM1のアドレスがFM100にセットされていて、*FM100と記述した場合＞



使用できるデバイスはユニットごとに異なります。「インデックス修飾と間接参照で使用できるデバイス」を参照してください。

ポイント
- **変数、ラベルも間接指定できます。ただし、以下の場合は間接指定できません。**
  - **配列型変数の添え字**
  - **配列型変数の要素**
  - **構造体型変数のメンバ**
- **ワードデータのビット位置指定は間接指定できません。**

参考
- 間接指定するデバイスがタイマ(T)、カウンタ(C)の場合に、接点、現在値、設定値は使用する関数の引数や付加するサフィックスによって決まります。
- ローカルデバイス使用時の記述は、「*@(デバイス)」になります。
- 間接指定したデバイスは.U(16ビット符号無し)として扱います。
- 間接指定に対しても型宣言できます。

●間接指定の使用方法

次の手順で間接指定します。

1 **アドレスセット関数(ADRSET)を使用して、参照するデバイスのアドレスを書き込みます。**

例)

ADRSET(R1000,DM0,FM100.D)

実行条件(R1000)がONのとき
DM0を示すアドレスをFM100、FM101に格納します。

ポイント **アドレスを格納できるデバイスはユニットごとに異なります。「インデックス修飾と間接参照で使用できるデバイス」を参照してください。**

「ADRSET関数」(6-26ページ)

2 **関数の各引数(デバイス)を間接指定するには、アドレスを格納しているデバイスの先頭に「*」を付けて使用します。**

例)

EM1000.U = AVG(*FM100.U,10)

*FM100(DM0)から10ワード分のデータの平均値を
EM1000.Uに格納します。

ポイント **アドレスを格納するデバイスに「*」を付けて使用する前に、必ずADRSET関数で間接指定するデバイスのアドレスを格納してください。**

3 **参照するデバイスを変更する場合は、間接指定専用関数(ADRINC/ADRDEC/ADRADD/ADRSUB)を使用して、格納されているアドレスを変更します。**

例)

ADRSET(EM300,FM100.D) — EM300のアドレスをFM100にセットします。

ADRINC(R1001,FM100.D) — 実行条件(R1001)がONのとき、FM100、FM101に格納されているDM0のアドレスをインクリメント(+1)します。

*FM100 = DM1000 — * FM100(EM300)にDM1000の値を格納します。

ポイント **参照するアドレスを変更する場合に、通常の演算子(+, − ,*,/など)を使用しないでください。ラダープログラムが正しく動作しません。**

●間接指定デバイスの型

間接指定デバイス(*[デバイス])は、その他のデバイスと同様に扱えるデータの範囲がサフィックスにより変化します。

- 型は、間接指定デバイスにサフィックスを付加して表します。

スクリプト記述
```
*DM1000.F = 123.567        ' *DM1000で間接指定されるデバイスは、.F(単精度浮
                           ' 動小数点型)として扱います。
```

- 型宣言(TYPE)を使用して間接指定デバイスの型を指定すると、サフィックスを省略して記述できます。

スクリプト記述
```
TYPE *DM1000.F             ' 型宣言:*DM1000は.Fとして扱います。
*DM1000 = 123.567          ' サフィックスを省略しても、*DM1000で間接指定さ
                           ' れるデバイスは、.Fとして扱います。
```

- 型宣言をおこなわなかった間接指定デバイスのサフィックスを省略して記述すると、.Uとして扱われます。

スクリプト記述
```
*DM1000 = 65535            ' 型宣言していない*DM1000で間接指定されるデバ
                           ' イスは、.U(16ビット符号無し)として扱います。
```

このとき間接指定されるデバイスの型が.U以外でも、間接指定デバイスのサフィックスを省略すると.Uとして扱われます。

スクリプト記述
```
ADRSET (MR2000, DM1000.D)  ' DM1000.DにMR2000のアドレスを格納します。
*DM1000 = 123              ' MR2000はビットデバイスですが、サフィックスを
                           ' 省略すると、*DM1000(=MR2000)は、.U(ワードデバ
                           ' イス)として扱われます。
```

- 間接指定デバイスの型宣言をすると、アドレスを格納するデバイスの型が.D(32ビット符号無し)として宣言したことになります。

スクリプト記述
```
TYPE *DM1000.F             ' *DM1000は.Fとして、またDM1000は.Dとして宣言
                           ' したことになります。
TYPE *DM2000.S             ' *DM2000は.Sとして、またDM2000は.Dとして宣言
                           ' したことになります。
ADRSET (EM5000, DM1000)    ' DM1000は.Dとして扱います。
*DM1000 = 123.567          ' *DM1000(=EM5000)は.Fとして扱います。
ADRSET (FM6000, DM2000)    ' DM2000は.Dとして扱います。
*DM2000 = –555             ' *DM2000(=FM6000)は.Sとして扱います。
```

●間接指定の使用例

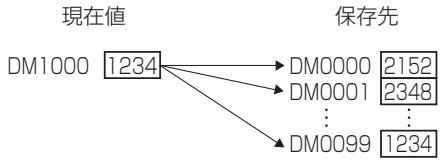
運転開始の最初のスキャンで、DM0を示すアドレスをFM100、FM101に格納します。
R0001の立ち上がりをトリガにして、DM1000の現在値をDM0から順番に格納していきます。

```
ADRSET(LDP(CR2008),DM0,FM100)
IF LDP(R1) THEN
    *FM100 = DM1000
    ADRINC(FM100)
END IF
```

運転開始時、DM0のアドレスをFM100･FM101に書き込みます。

R1の立ち上がりで間接指定されている対象デバイスにDM1000の値を書き込みます。

FM100･FM101に格納されているアドレスをインクリメント(+1)します。



インデックス修飾と間接指定を併用する

■記述方法について

間接指定デバイスにインデックス修飾をおこなう場合、以下のように記述します。

例) (*DM1000.U):Z01 ―― *DM1000をZ01でインデックス修飾します。

*(DM1000.U:Z01) ―― Z01でインデックス修飾されたDM1000を間接参照します。

(　)を省略して記述すると、以下の意味になります。

例) *DM1000.U:Z01 ―― * DM1000をZ01でインデックス修飾します。

●インデックス修飾と間接指定の併用例

R1の立ち上がりをトリガ信号として、DM1000に格納される現在値を保存します。
R0の立ち上がり時にR2がONの場合はDM0から、OFFの場合はEM0から順番にデータを保存していきます。

```
IF LDP(R0000) THEN
    Z01 = 0 ---------------------------------------- ①
    ADRSET(R2,DM0,FM100.D) ------------------- ②
    ADRSET(R2 = OFF,EM0,FM100.D) ------------ ③
END IF
IF LDP(R1) THEN
    *FM100:Z01 = DM1000 ------------------------- ④
    Z01 += 1 ---------------------------------------- ⑤
END IF
```

［測定開始時］

①インデックスレジスタZ01に0を書き込みます。(インデックスレジスタの初期化)
②R2がONの場合は、FM100･FM101にDM0のアドレスを書き込みます。
③R2がOFFの場合は、FM100･FM101にEM0のアドレスを書き込みます。

［トリガ発生時］

④FM100･FM101に格納されているアドレスのデバイスを、Z01でインデックス修飾(Z01の値を加算)したデバイスにDM1000の値を書き込みます。

例) R2がOFFの時、

*FM100→EM0
Z01= 23

→ *FM100:Z01→EM23

⑤Z01をインクリメント(＋1)します。

■簡易間接指定

KVスクリプトでは簡易間接指定(#TM)は使用できません。簡易間接指定を利用する際はラダープログラムで記述してください。

データの型 （サフィックス）

KVスクリプトではデバイスにサフィックスを指定することにより、デバイスや定数の型を表現します。末尾に「.U .S .D .L .F .DF .B .T」の記号"サフィックス"がつくデバイスは、サフィックスの違いにより扱うデータの種類が異なります。

サフィックス

サフィックスとはデバイスに接尾語としてつくもので以下の8種類があります。

□.U　□.S　□.D　□.L　□.F　□.DF　□.B　□.T

※□にはサフィックス対応のデバイスが入ります。

各デバイスはサフィックスに応じて、データメモリなどの値を以下のように扱います。

□.U 16ビット符号無しデータとして扱います。（0～65535）
□.S 16ビット符号付きデータとして扱います。（–32768～32767）
□.D 32ビット符号無しデータとして扱います。（0～4294967295）
□.L 32ビット符号付きデータとして扱います。（–2147483648～2147483647）
□.F 単精度浮動小数点型実数データとして扱います。

−3.4E+38 ≦ N ≦ −1.4E−45
N = 0
1.4E−45 ≦ N ≦ 3.4E+38
（有効桁数:約7桁）
（KV Nanoシリーズの場合、絶対値が1.4E−45～1.2E−38の範囲は使用できません。）

□.DF 倍精度浮動小数点型実数データとして扱います。

−1.79E+308 ≦ N ≦ −2.23E−308
N = 0
2.23E−308 ≦ N ≦ 1.79E+308
（有効桁数:約16桁）

□.B ビット型データとして扱います。（ON:真, OFF:偽）
□.T 文字列型データとして扱います。

参 考　ラダープログラムで指定するサフィックスは、命令語にのみ付き、デバイスには付きません（スクリプトではデバイスのみに付き、関数には付きません）。

例

```
DM1000 (.U)
DM1000.S
DM1000.D
DM1000.L
DM1000.F
DM1000.DF
DM1000.B
DM1000.T
```

ポイント　**変数、ラベルの場合も「AnalogData.L」のようにサフィックスを使用できます。ただし、変数編集ウィンドウ・ラベル編集ウィンドウで設定したデータ型よりもサフィックスが扱うデータサイズの方が大きい場合は変換エラーが発生します。**

●サフィックスの省略

デバイスのサフィックスを省略して記述した場合、デバイスの種別により扱われる型が決まります。

デバイスの種類	サフィックスを省略した場合の型(デフォルトの型)
R(DR),B,MR,LR,T,C,CTC,CR,UR	.B(ビット型)
DM,W,EM,ZF,FM,TM,CM,V,P,UM,UV,間接指定(*)するデバイス	.U(16ビット符号無しデータ型)
Z※	KV-8000/7500/7300/5500/5000/3000：.L(32ビット符号付きデータ型) KV-1000　：.S(16ビット符号付きデータ型)
CTH	.D(32ビット符号無しデータ型)

※ 使用するCPUユニットにより、デフォルトの型が異なります。

ポイント　**変数でサフィックスを省略した場合は、変数編集ウィンドウで設定したデータ型で扱われます。**

スクリプト記述

```
DM1000 = 2000                    '省略した場合16ビット符号無し(.U)扱い
DM1100.D = 75535                 'DM1100は32ビット符号無し(.D)扱い
IF DM1000 = 3000 THEN            '
   DM3000.L = DM1100 + 1000      '右辺は32ビット符号付きで演算します。
END IF                           '
                                 '
DM1000.L = –21474800             'DM1000は32ビット符号付き(.L)扱い
IF DM1000 > 0 THEN               '
   DM3000.D = DM1000.S * –1      '右辺は.Lに拡張して演算し、解を.Dに変換し
END IF                           'て代入します。※
```

※ 32 ビット符号付きデータ(.L)を 32 ビット符号無しデータ(.D)に代入するため、「スクリプト警告レベル設定」を「高」にしている場合、スクリプト変換時に警告が出ます。

参考　「TYPE」を記述することで、指定デバイスの型をあらかじめ宣言しておくこともできます。
型宣言は、宣言をおこなったスクリプト内でのみ有効です。
詳細は、「型宣言(TYPE)について」を参照してください。

■.Uと.Sについて

□.UはBINデータを16ビット符号無しデータとして扱います。
□.SはBINデータを16ビット符号付きデータとして扱います。
したがって、BINデータとサフィックスに応じた10進表記の関係は、次のようになります。

デバイス(DM1000)	定数の種類	1111111111111111というビットデータの場合
DM1000.U	符号無し10進	65535
DM1000.S	符号付き10進	–1

- DM1000に格納されている16ビットBINデータ（1111111111111111）を10進定数100と比較する場合、サフィックスが.Uのとき、DM1000の値は65535と解釈されます。

スクリプト記述
```
IF DM1000.U > 100 THEN
INC (DM2000)
END IF
```

したがって、DM1000 > 100 が成り立つので、DM2000に1が加算されます。

- サフィックスが.Sのとき、DM1000の値は–1と解釈されます。

スクリプト記述
```
IF DM1000.S > 100 THEN
DEC (DM2000)
END IF
```

したがって、DM1000 < 100 となるので、DM2000は減算されません。
制御文の詳細については、「制御文」を参照してください。

■.Dと.Lについて

□.D はBINデータを32ビット符号無しデータとして扱います。
□.L はBINデータを32ビット符号付きデータとして扱います。

したがって、BINデータとサフィックスに応じた10進表記の関係は、次のようになります。

デバイス（DM1000）	定数の種類	11111111111111111111111111111111というビットデータの場合
DM1000.D	符号無し10進	4294967295
DM1000.L	符号付き10進	–1

- DM1000･DM1001に格納されている32ビットBINデータ（11111111111111111111111111111111）を10進定数100と比較する場合サフィックスが.Dのとき、[DM1000･DM1001]の値は4294967295と解釈されます。

スクリプト記述
```
IF DM1000.D > 100 THEN
DM2000 = DM2000 + 1
END IF
```

したがって、DM1001（上位）･DM1000（下位）（4294967295）> 100となり、接点はONされ、DM2000に1が加算されます。

- サフィックスが.Lのとき、[DM1000･DM1001]の値は–1として扱われます。

スクリプト記述
```
IF DM1000.L > 100 THEN
DM2000 = DM2000 – 1
END IF
```

したがって、DM1001（上位）･DM1000（下位）（–1）< 100 となり、接点はOFFされ、DM2000は減算されません。

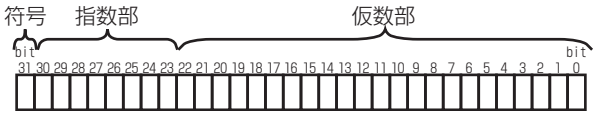
ポイント **32ビット型のデータ（.D、.L、.F）を扱うときは、偶数デバイス番号を使用してください。**

■.Fについて

□.FはBINデータを単精度浮動小数点型実数データとして扱います。データは32ビットです。
単精度浮動小数点型実数データは、特殊なビット構成で値を表現（内部表現）しています。
したがって、内部表現を解析して直接ビットを操作することは、できるだけ避けるようにしてください。
単精度浮動小数点型実数値は、次のように表記される数値です。

$$-1.234\times2^{-3}$$

符号　仮数部　指数部



符号（1ビット）
0＝正
1＝負

仮数部（23ビット）
0～8388607

指数部（8ビット）
00000000（＄00）＝未使用
00000001（＄01）＝–126
00000010（＄02）＝–125
00000011（＄03）＝–124
⋮
01111110（＄7E）＝–1
01111111（＄7F）＝0
10000000（＄80）＝1
10000001（＄81）＝2

11111100（＄FC）＝125
11111101（＄FD）＝126
11111110（＄FE）＝127
11111111（＄FF）＝未使用

ポイント

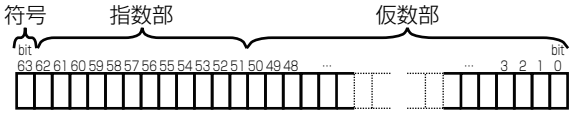
DM/W/EM/FM/TM/ZFなどのワードデバイスを使用して32ビットデータを扱う場合は、デバイスにサフィックス.D、.L、.Fを付加します。
このときデバイスは番号が連続する2ワードを使用します。
指定するデバイスは下位16ビットにサフィックスを付けたものになります。
例）DM1000.D→[DM1001,DM1000]を使用します。

■.DFについて

□.DFはBINデータを倍精度浮動小数点型実数データとして扱います。データは64ビットです。
倍精度浮動小数点型実数データは、特殊なビット構成で値を表現（内部表現）しています。
したがって、内部表現を解析して直接ビットを操作することは、できるだけ避けるようにしてください。
倍精度浮動小数点型実数値は、次のように表記される数値です。

$$-1.2345678 \times 2^{-102}$$

符号　仮数部　指数部



符号（1ビット）
0＝正
1＝負

仮数部（51ビット）
0～2251799813685248

指数部（12ビット）
000000000000（＄000）＝未使用
000000000001（＄001）＝－2046
000000000010（＄002）＝－2045
000000000011（＄003）＝－2044
⋮
011111111110（＄000）＝－1
011111111111（＄7FF）＝0
100000000000（＄800）＝1
100000000001（＄801）＝2
⋮
000000000000（＄000）＝2045
000000000000（＄000）＝2046
000000000000（＄000）＝2047
111111111111（＄FFF）＝未使用

ポイント

DM/W/EM/FM/TM/ZFなどのワードデバイスを使用して倍精度浮動小数点型実数を扱う場合は、デバイスにサフィックス.DFを付加します。
このときデバイスは番号が連続する4ワードを使用します。.DFを扱うときは、偶数番号のデバイスを用いてください。
指定するデバイスは最下位16ビットにサフィックスを付けたものになります。
例）DM1000.DF→［DM1003・DM1002・DM1001・DM1000］を使用します。
□.DFは、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットで使用できます。

■.Bについて

□.BはBINデータをビット型データとして扱えます。

例）ビットデバイスMR1000をON(1, TRUE)にします。

スクリプト記述　SET(MR1000.B)

参考　ワードデバイスにサフィックスに.Bを付加すると、最下位ビットがビット型判定の対象になります。上記例の場合、SET関数の引数にDM10.Bを指定すると、DM10の最下位ビットが1になります。ワードデバイスの真偽判定は、以下のようになります。

```
例）IF DM10.B = ON THEN
        処理文
    END IF
```

例）ビットデバイスMR1000をOFF(0, FALSE)にします。

スクリプト記述　RES(MR1000.B)

■.Tについて

□.TはBINデータを文字列型データとして扱います。

例）DM100を先頭とする連続したデバイスに "KEYENCE" の文字列を格納します。

スクリプト記述　DM100.T = "KEYENCE" + CHR ($0D) + CHR ($0A)

上記例の場合、データ格納状態は以下のようになります。

データ格納状態

bit 15	bit 0	
K(4BH)	E(45H)	:DM100
Y(59H)	E(45H)	:DM101
N(4EH)	C(43H)	:DM102
E(45H)	C_R(0DH)	:DM103
L_F(0AH)	NUL(00H)	:DM104

参考　文字列の終端には自動的にNUL(00H)が格納されます。

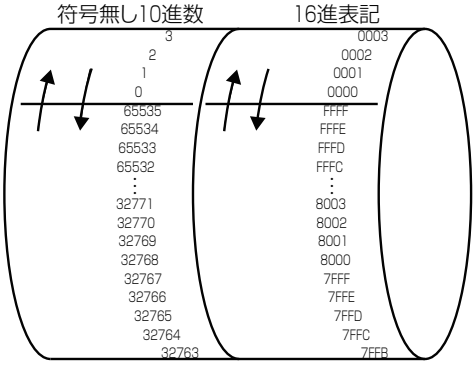
ポイント　**扱える文字列の最大は1999文字です。**

■最大値・最小値を超える時の数値の変化

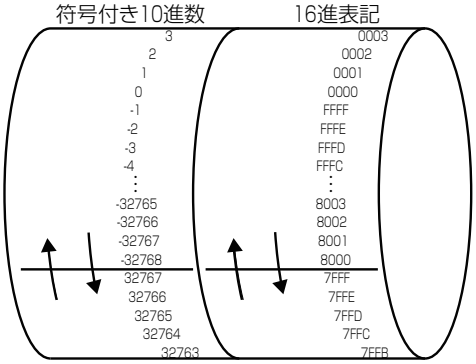
算術演算結果が最大値または最小値を超えた場合、データ型式によりそれぞれ次のように数値が変化します。

● 16ビットデータの場合

□.Uの65535を超える時の値の変化

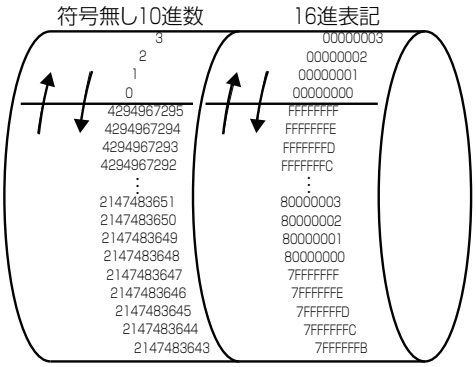


□.Sの最大値、最小値を超える時の値の変化

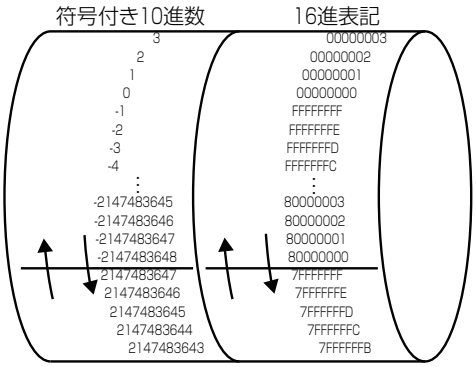


● 32ビットデータの場合

□.Dの4294967295を超える時の値の変化

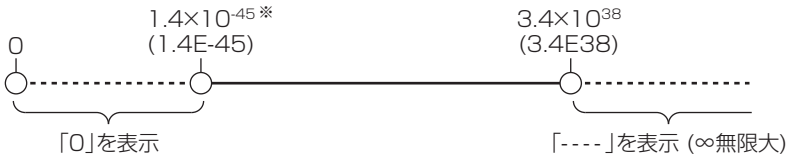


□.Lの最大値、最小値を超える時の値の変化



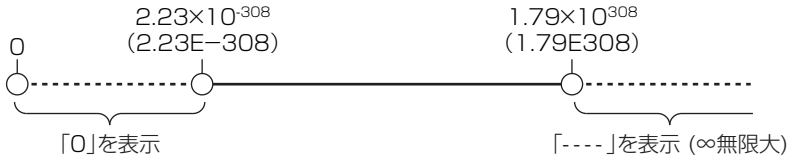
参 考

・単精度浮動小数点型データの場合
オーバーフロー：絶対値が3.4×10^{38}より大きくなる場合、「∞(無限大)」になります。
アンダーフロー：絶対値が1.4×10^{-45}※より小さくなる場合、「0」になります。



※ KV Nanoシリーズの場合1.2×10−38(1.2E−38)です。

・倍精度浮動小数点型データの場合
オーバーフロー：絶対値が1.79×10^{308}より大きくなる場合、「∞(無限大)」になります。
アンダーフロー：絶対値が2.23×10^{-308}より小さくなる場合、「0」になります。



型宣言(TYPE)について

KVスクリプトではサフィックスを省略して記述したデバイスは、あらかじめ決められた型(デバイスにより異なる)として扱われます。型宣言(TYPE)をおこなうと、サフィックスを省略して記述したデバイスの型(デフォルトの型)を変更できます。
この型宣言は、宣言がおこなわれたボックススクリプト内(またはエリアスクリプト)のみ有効です。

- 「デバイス種別＋サフィックス（ 例：DM.F)」で宣言すると、その種類のデバイスは全て宣言されたサフィックスの型を持つデバイスとして取り扱われます。

スクリプト記述

```
TYPE DM1000.D                 '以後このスクリプト内で、DM1000は.Dと
                              'して扱われます
TYPE EM.F                     '以後このスクリプト内で、EMnnnnはすべて
                              '.Fとして扱われます
DM1000 = 10000 * DM2000.D     'サフィックスを省略しても、DM1000は.D扱い
                              'です
EM3000 = 123.456 / DM1200.F   'EMデバイスは全て.F扱いです
```

- 特定範囲のデバイスの型宣言をおこなうこともできます。

スクリプト記述

```
TYPE DM1200 – DM1600.F        'DM1200～DM1600は全て.Fで扱われます
TYPE DM2000.L – DM2100        '範囲指定の別の記述方法です
TYPE DM2200.D – DM2300.D      '範囲指定の別の記述方法です
DM1200 = 12.5 * DM1100        'DM1200はサフィックスを省略しても.F扱い
DM1210 = SIN(DM1300)          '引数や戻り値は.Fとして扱われます
```

- 型宣言をおこなったデバイスでも、別のサフィックスを指定すると、そのサフィックスが優先されます。

スクリプト記述

```
TYPE EM.F                     '以後このスクリプト内で、EMはすべて.Fです
EM1000.S = DM1100 * –10       'ただし、別サフィックスを指定すると、指定
EM3000.D = 1234 * DM1200      'サフィックスが優先されます
EM3000 = EM3000 + 10000       'サフィックスを省略するとTYPEで宣言した
                              '型として扱われます
```

ポイント

- **型宣言は各スクリプトの先頭にのみ記述できます。**
- **型宣言がおこなわれたスクリプトを抜けると、一切の型宣言は無効になります。**
  よって、別のスクリプトでは再度型を宣言しなおすか、サフィックスを指定して対応する必要があります。
- **グローバルラベルを使用している場合、ラベル登録時に定義したデータ形式よりも型宣言が優先されるので注意が必要です。**

例)TYPE DM.D
上記スクリプトを記述すると、DMを割り当てているすべてのグローバルラベルのデータ形式が.Dになります。

演算の型について

KVスクリプトは、数値演算をおこなう場合、ラダー言語のように専用の命令語を使うことなく、演算子(+, − , ＊ ,/)を使って記述できます。

■演算式で扱えるデータの範囲

演算式の各項は、すべて.L(32ビット符号付き)、.F(単精度浮動小数点型)もしくは.DF(倍精度浮動小数点型)として扱われます。このため式のすべてのデバイスは、自動的に.L、.Fもしくは.DFに符号拡張されてから演算されます。

●演算式に浮動小数点型データ(.F, .DF, 定数値)が含まれない場合

例) DM3000.L = (DM1000.U + 30) ＊ DM2000.U
演算式内のデバイスは、いったん.Lに符号拡張されますので上記演算式は
DM3000.L = (TOL(DM1000.U) + 30) ＊ TOL(DM2000.U)
として計算されます。

例) DM3100.L = (DM1100.D + 30) ＊ DM2100.U
式内に、.D(32ビット符号無し)データがあった場合、.Lに符号拡張したときにデータに精度落ちが発生する場合がありますので、注意が必要です。
上記演算式は、
DM3100.L = (DM1100.L + 30) ＊ DM2100.L
として計算され、「スクリプト警告レベル設定」が「高」の場合、アウトプットウィンドウに以下の警告が表示されます。
警告 : "+"演算によりデータが欠落した可能性があります。(符号無し32ビット型->符号付き32ビット型)

●演算式に浮動小数点型データ(.F,.DF,定数値)が含まれる場合

例) DM3000.F = (DM1000.S + 30) ＊ DM2000.F + DM2500.U
演算式内に、.DF(倍精度浮動小数点型)がなく、.F(単精度浮動小数点型)が含まれる場合、.Fが演算要素として使用されるまでは.Lで演算し、.Fが演算要素になった時点で、.Fとして演算されます。
式内の.L、.F以外のデータは、いったん.Lまたは.Fに符号拡張されますので、上記演算式は
DM3000.F = (TOL(DM1000.S) + 30) ＊ DM2000.F + TOF(DM2500.U)
として計算されます。

例)DM3000.DF = (DM1000.S + 30) ＊ DM2000.F + DM2500.U + DM2600.DF - DM2700.U
演算式内に、.DF(倍精度浮動小数点型)があった場合、.DFが演算要素として使用されるまでは.Lもしくは.Fで演算し、.DFが演算要素になった時点で、.DFとして演算されます。
式内の.L、.F、.DF以外のデータは、いったん.L、.Fまたは.DFに符号拡張されますので、上記演算式は
DM3000.DF = TOF(TOL(DM1000.S) + 30) ＊ DM2000.F + TOF(DM2500.U) + DM2600.DF - TODF(DM2700.U)
として計算されます。

●結果の格納先が.L以外の場合

例) DM3200.S = (DM1200.U + 30) ＊ DM2200.U

浮動小数点型データを含まない演算式の結果は必ず.L型になるので、左辺の結果格納先の型に.L以外を指定すると、データに精度落ちが発生する場合がありますので注意が必要です。

上記演算式は、

DM3200.S =TOS ((DM1200.U + 30) ＊ DM2200.U)

として計算され、「スクリプト警告レベル設定」が「高」の場合、アウトプットウィンドウに以下の警告が表示されます。

警告 : "="代入によりデータが欠落した可能性があります。(符号付き32ビット型->符号付き16ビット型)
結果を格納するデバイスが.Fの場合も、右辺に.DFが含まれる場合は精度落ちが発生します。

ポイント

演算式で発生する精度落ちに対する警告は、デフォルト状態ではアウトプットウィンドウに表示されません。
警告メッセージを表示させるには、スクリプト警告レベルを「高」に設定してください。
「スクリプト変換時のメッセージ」(2-52ページ)

PRAGMA指示命令

演算式に単精度浮動小数点型データ(.F, 定数)が含まれる場合、単精度浮動小数点型(.F)に型変換をして演算が実行されますが、PRAGMA指示命令を活用することにより、倍精度浮動小数点型(.DF)に型変換をして演算をするようコンパイル指示できます。
倍精度で演算することにより、途中の演算結果を含めて精度落ちを抑制できます。

ポイント **PRAGMA指示命令はKV-1000では使用できません。**

■ PRAGMA指示命令の使用方法と動作説明

PRAGMA指示命令は、IF文やFOR文などの制御文と同様に、#PRAGMA DOUBLE ON と #PRAGMA DOUBLE OFFとを対で記述します。#PRAGMA DOUBLE ON から始まり #PRAGMA DOUBLE OFFで終わる区間が倍精度演算の有効範囲となります。

●演算式の動作

スクリプト記述

```
DM1000.F = 123.45
DM1004.S = -1998

DM2000.DF = DM1000.F * 3.141592        ' ...①
DM2004.DF = DM1004.S / 25              ' ...②

#PRAGMA DOUBLE ON
   DM2010.DF = DM1000.F * 3.141592     ' ...③
   DM2014.DF = DM1004.S / 25           ' ...④
#PRAGMA DOUBLE OFF
```

PRAGMA DOUBLEの有効区間

動作説明

①②と③④の右辺は全く同じ演算式ですが、③④はPRAGMA指示命令で浮動小数点型の演算は倍精度に変換しておこなうことを指定しています。①②は単精度浮動小数点型で演算をおこない、③④は倍精度浮動小数点型で演算をおこないます。

① 演算式に単精度浮動小数点型(.F, 定数)のみが含まれているので、そのまま単精度浮動小数点型で演算をおこない、演算結果を左辺の型(倍精度浮動小数点型(.DF))に変換してDM2000～DM2003(4ワード)に格納します。

② 演算式に整数型(.S, 定数)のみが含まれているので、整数型の値を32ビット符号付き整数(.L)に変換した後に演算をおこない、演算結果を倍精度浮動小数点型(.DF)に変換してDM2004～DM2007(4ワード)に格納します。

③ 演算式に単精度浮動小数点型(.F)のみが含まれていますが、PRAGMA DOUBLEが有効なので、それぞれ倍精度浮動小数点型(.DF)に変換した後に演算をおこない、演算結果をDM2010～DM2013(4ワード)に格納します。

④PRAGMA DOUBLEが有効ですが、演算式に浮動小数点型のデータを含んでいないので、②と同じルールで演算をおこない、演算結果を倍精度浮動小数点型(.DF)に変換してDM2014～DM2017(4ワード)に格納します。

ウォッチウィンドウ

デバイス	現在値	表示形式
DM1000	+123.45	FLOAT
DM1004	-1998	±10進数16BIT
DM2000	+387.829528808594	DOUBLE FLOAT
DM2004	-79	DOUBLE FLOAT
DM2010	+387.829522812622	DOUBLE FLOAT
DM2014	-79	DOUBLE FLOAT

①(**DM2000**)と③(**DM2010**)の演算結果は異なります。

ポイント

- **倍精度浮動小数点型(.DF)は、4ワード(64ビット)のデータサイズです。**
- **整数どうしの演算の場合は、PRAGMA指示の有効/無効に関わらず、32ビット符号付き整数(.L)に変換して演算されます。整数どうしの演算も浮動小数点型で演算をしたい場合は、上記の④の例であれば、**

  DM2014.DF = TODF(DM1004.S) / 25

  のように、対象の一部を浮動小数点型に拡張してください。

  また、以下のように定数を小数表記にしても同じ効果が得られます。

  DM2014.DF = DM1004.S / 25.0
- **倍精度型で演算することにより演算精度が上がりますが、演算の実行速度は単精度型よりも遅くなります。演算精度の必要な個所に絞ってPRAGMA指令命令を利用することがポイントです。**

ポイント

PRAGMA指示命令は、下記のような他の制御文と交わる(制御ブロックが交差する)記述はできません。

「ネスティングを使用する時の注意事項」(4-32ページ)

【FOR文(繰り返し制御文)とPRAGMA指示命令が交差した場合】

```
FOR <ワードデバイス①の値> TO <終了値>                    '制御文①
    処理文①
    FOR <ワードデバイス②の値> TO <終了値>                '制御文②
        処理文②
        #PRAGMA DOUBLE ON                               'PRAGMA指示命令
            演算式
    NEXT
        #PRAGMA DOUBLE OFF
NEXT
```

制御ブロックの交差

●関数の動作

スクリプト記述

```
    DM1000.DF = SIN(DM100.F)          ' ...①

#PRAGMA DOUBLE ON                     }PRAGMA DOUBLEの有効区間
    DM1010.DF = SIN(DM100.F)          ' ...②
#PRAGMA DOUBLE OFF
```

動作説明

①と②の右辺は全く同じ演算式ですが、②はPRAGMA指示命令で浮動小数点型の演算は倍精度に変換しておこなうことを指定しています。PRAGMA指示をおこなうと、関数の引数に単精度浮動小数点型(.F)を指定している場合であっても、関数の仕様として倍精度演算をサポートしていれば、倍精度浮動小数点型に変換されて実行されます。

① 引数が単精度浮動小数点型(.F)なので、そのまま単精度(SIN.F)で演算をおこない、演算結果を左辺の型(倍精度浮動小数点型(.DF))に変換してDM1000～DM1003(4ワード)に格納します。

② 引数の単精度浮動小数点型(.F)を、倍精度浮動小数点型(.DF)に変換して倍精度(SIN.DF)で演算をおこない、演算結果をDM1010～DM1013(4ワード)に格納します。

上記②のスクリプト記述を、PRAGMA指示命令なしで記述すると以下のようになります。

スクリプト記述

```
DM1010.DF = SIN(TODF(DM100.F))
```

ポイント

PRAGMA指示命令内の関数であっても下記の関数については、倍精度浮動小数点型データに拡張されません。

- **TOU関数(.U(16ビット符号無し整数)に変換)**
- **TOS関数(.S(16ビット符号付き整数)に変換)**
- **TOD関数(.D(32ビット符号無し整数)に変換)**
- **TOL関数(.L(32ビット符号付き整数)に変換)**
- **TOF関数(.F(単精度浮動小数点型データ)に変換)**

4

スクリプトの文法と注意

代入文..... 4-2
演算子..... 4-6
制御文..... 4-10
スクリプトを記述するときの注意 4-38

代入文

代入文は、左辺と右辺が等号(=)によって結ばれた式のみで構成された文で、右辺の結果あるいは値を左辺に代入(格納)します。

代入文の使用方法

代入文の右辺として記述できるもの(デバイスや変数、式)には、次に示すように数値、文字列、ビットがあり、それぞれ定数か演算式、または(戻り値をもつ)関数で記述します。

●数値代入文

右辺の数値定数または数値演算の結果を、左辺に代入します。

スクリプト記述

```
DM1000.U = 123                          '数値定数の代入
DM1100.L = DM2000.U + 123               '数値演算結果の代入
DM1200.U = DCNT(DM2100,EM1000,20)       '関数が処理した結果の代入
```

●文字列代入文

右辺の文字列定数または文字列演算の結果を、左辺に代入します。

スクリプト記述

```
DM100.T = "ABCD"                        '文字列定数の代入
DM200.T = "現在" + DM300.T + "です"      '文字列連結結果の代入
```

●ビット代入文

- 右辺のビット定数またはビット演算(条件式の評価)の結果を、左辺に代入します。

スクリプト記述

```
R1000 = ON                              'ビット定数(ブール値)の代入
R1000 = R1100 OR (R1200 AND R1300)      'ビット演算結果の代入
R1000 = DM100.S > 100                   '条件式評価の代入
R1000 = R2000 = R3000                   '比較結果の代入
```

- 1行に等号(＝)を複数回使用した場合、式の最初(一番左)に出現する等号(＝)を代入の「＝」として扱い、以降の等号(＝)はすべて比較の等号(＝)として扱います。
  このとき、代入の等号(＝)の左側を左辺として、右側をすべて右辺として扱います。

（例） **R1000 ＝ R2000 ＝ R3000 ＝ R4000 → R1000 ＝ (R2000 ＝ (R3000 ＝ R4000))**

↑ 代入　↑ 比較　↑ 比較

左辺←：→右辺

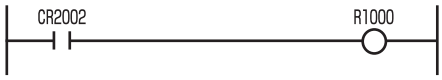
- ビット状態は、ON、OFF以外に1、0やTRUE、FALSEで表せます。

スクリプト記述

```
R1000 = 1
```

動作説明　R1000をONにします。

ラダー変換



- ビットデータはブール値のみ代入できます。

  R1000 = TRUE . . . . . R1000 = ON
  R1000 = FALSE . . . . . R1000 = OFF

- ビットデータには1、0以外の定数の代入や数値演算式の結果の代入はできません。

  × :R1000 = DM1000 + DM2000 . . . . . **“+”は不可**

- ビットデータに論値演算式の結果や比較演算の結果(どちらも結果はブール値)の代入は可能です。

  ○: R1000 = R2000 AND R2100 . . . . . **論理演算子の使用は可能**
  R1000 = DM1000 >= 500 . . . . . **等号、不等号(演算結果がブール値しかもたない式)は可能**

●演算代入文

演算代入文とは、代入文の左辺にあるデバイスに対して右辺で四則演算をおこなっている場合、右辺のデバイスを省略して記述できる代入文です。
演算代入文には次の4種類があります。

+=, –=, ∗=, /=

左辺	右辺	演算代入文
DM1000.U = DM1000.U + 1	→	DM1000.U += 1

また演算代入文を使った場合、右辺の演算は左辺の型にあわせて演算します。

スクリプト記述

```
DM1000.U += 3                    'DM1000.U = DM1000.U + 3(右辺は.Uとして演算)※
DM1100.S –= 3                    'DM1100.S = DM1100.S – 3(右辺は.Sとして演算)※
DM1200.D ∗= 3                    'DM1200.D = DM1200.D ∗ 3(右辺は.Dとして演算)※
DM1300.L /= 3                    'DM1300.L = DM1300.L / 3(右辺は.Lとして演算)※
DM2000.F += DM1500–3             'DM2000.F = DM2000.F + TOF (DM1500 – 3)
DM2100.D += _                    'DM2100.D = DM2100.D + TOD (SRA (EM1000, 8)∗2)
SRA (EM1000, 8)∗ 2
```

※ 演算式は通常 .L として演算しますが、演算代入文は特別に左辺の型にあわせて演算します。

以下のスクリプトは演算結果が同じ(.Lの範囲)ですが、変換されるラダーは異なります。

スクリプト記述

```
DM1000.U = 10
DM1000.U = DM1000.U + 10         '記述方法A:演算結果は20
DM1000.U += 10                   '記述方法B:演算結果は20
```

文字列は加算だけできます。

スクリプト記述

```
DM3000.T +="ABC"                 'DM3000.T = DM3000.T + "ABC"と同じ意味です
```

■代入文を使用するときの注意

●右辺と左辺で型が異なる場合

- 代入文は、右辺と左辺で型が異なる(型変換ができない)場合成立しません。

スクリプト記述　DM1000.T = DM2000.U　'エラーになります

動作説明　右辺のDM2000.U(定数：16ビット符号無し)を、DM1000.T(文字列型)に代入しようとしています(変換エラー)

- ただし型が異なる場合でも .U.S.D.L.F 間での型の違いは、変換できます (精度落ちが発生する場合があります)。

スクリプト記述　DM1100.U = DM2100.D　'スクリプト警告レベル「高」で警告になります

動作説明　右辺のDM2100.D(32ビット符号無し)を、DM1100.U(16ビット符号無し)に代入します。DM2100.Dの下位16ビットだけの代入になります。

「警告レベルの設定方法」(2-53ページ)

●左辺に代入できない場合

- 下記のスクリプトは、左辺の「3」が数値であり、結果や値を代入できないため、代入文として正しくありません。

スクリプト記述　3 = DM1000.U　'エラーになります(定数には代入できません)

動作説明　右辺のDM1000.U(16ビット符号無し)を、3(定数)に代入しようとしています。(変換エラー)

- 左辺を式や関数で記述しているときも、代入文として正しくありませんので、エラーになります。

スクリプト記述　DM1100.U + 30 = DM1000.U　'エラーになります(式には代入できません)

動作説明　右辺のDM1000.U(16ビット符号無し)を、左辺の演算式に代入しようとしています。(変換エラー)

MEMO

演算子

演算子とはデータに処理を加える記号のことで、式の中で使われる足し算(＋)や、掛け算(*)などのことをいいます。

演算子一覧

■演算子の種類

演算子には、下記に示す「算術演算子」、「比較演算子」、「文字列連結演算子」、および「論理演算子」の4種類があります。

ポイント　**演算子は半角で記述してください。**

●算術演算子

2つの値の算術計算の結果を返します。

記　号	処　理　内　容	記　述　例
^※1	累乗を求めます	DM500 = DM100 ^ 2
*	2つの値の積を求めます(乗算)	DM502 = DM100 * 3
/	2つの値の商を求めます(除算)	DM504 = DM100 / 4
MOD※2	2つの値の除算を行い、その剰余を返します	DM506 = DM100 MOD 5
+	2つの値の和を求めます(加算)	DM508 = DM100 + 6
－	2つの値の差を求めます(減算)	DM510 = DM100－7

※1　KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでは、.F,.DFのサフィックスに対しても累乗を用いることができます。
※2　記号の前後に半角スペースが必要です。

●文字列連結演算子

2つの文字列の連結結果を返します。

記　号	処　理　内　容	記　述　例
&, +※3	2つの文字列を連結します	DM600.T = "KEY" + "ENCE"

※3　「+」は記述内容に応じて算術演算と文字列連結に自動的に判断されます。

●比較演算子

2つの値の大小関係を評価した結果をブール値(ONまたはOFF)で返します。

記　号	処　理　内　容	記　述　例
<	より小さい	MR500 = DM100 < 10
<=	小さいか等しい(以下)	MR501 = DM100 <= 20
>	より大きい	MR502 = DM100 > 30
>=	大きいか等しい(以上)	MR503 = DM100 >= 40
=	等しい(等価)	MR504 = DM100 = 50
<>	等しくない(不等価)	MR505 = DM100 <> 60

● 論理演算子

2つの値の論理演算の結果を、ビット同士の演算の場合はブール値(ONまたはOFF)で、ワード同士の演算の場合はワード値で返します。

記　号	処　理　内　容	記　述　例
NOT※4	値の論理否定値を求めます	MR600 = NOT R000
AND※4	2つのデータ(ビットデータ同士、ワードデータ同士)の論理積を求めます	MR601 = R000 AND R001
OR※4	2つのデータ(ビットデータ同士、ワードデータ同士)の論理和を求めます	MR602 = R000 OR R001
XOR※4	2つのデータ(ビットデータ同士、ワードデータ同士)の排他的論理和を求めます	MR603 = R000 XOR R001

※4　記号の前後に半角スペースが必要です。

ポイント

- **演算するデータ(演算要素)は、ビットデータ同士またはワードデータ同士を指定してください。ビットデータとワードデータなどの論理演算はできません。**
- **演算するデータ(演算要素)の型の大きさが異なる場合、大きい方の型に合わせて演算をおこないます。**
  例)　DM1000.U AND DM2000.D ・・・・32ビット型で演算します

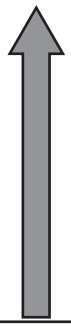
演算子の優先順位

1つの式の中に複数の演算子が使われている場合、式の各演算部分は決められた順序にしたがって評価(計算)されます。この順序を演算子の優先順位といいます。

■優先順位について

異なる種類の演算子が1つの式の中で使われている場合、算術演算子、比較演算子、論理演算子の順序で評価され、また算術演算子および論理演算子については、優先順位の高い順で評価されます。

優先順位　高



優先順位　低

算術演算子	累乗	^
	マイナス符号	−
	乗算と除算	*, /
	剰余演算	MOD
	加算と減算	+, −
	文字列連結	+
比較演算子	等しい, 等しくない, より小さい, より大きい, 以下, 以上	=, <>, <, >, <=, >=
論理演算子	論理否定	NOT
	論理積	AND
	論理和	OR
	排他的論理和	XOR

●乗除算・加減算の優先順位

1つの式の中で乗算と除算(もしくは、加算と減算)の両方が記述されている場合、演算子があらわれる順(左から右)に計算されます。ただし、乗除算と加減算の両方が記述されている場合は、乗除算が優先されます。

例）DM1000 + DM2000 ∗ 30 − DM3000/10　の場合

演算順1　：　DM2000 ∗ 30 …… 結果①
演算順2　：　DM3000/10 …… 結果②
演算順3　：　DM1000 + 結果① …… 結果③
演算順4　：　結果③ − 結果②　…… 最終演算結果

●比較演算子の優先順位

比較演算子の優先順位はすべて同じです。したがって、1つの式の中で複数の比較演算子が指定されている場合、演算子が現れる順(左から右)に評価されます。

例）R000 = DM2 > 100 = R1 = FALSE　の代入文の場合
R000 = (((DM2 > 100) = R001) = FALSE)　として演算されます。

比較順1　：　DM2 > 100 …… 結果①
比較順2　：　結果① = R001 …… 結果②
比較順3　：　結果② = FALSE …… 比較結果
演算1　　：　比較結果(成り立つときON、成り立たないときOFF)をR000 に代入

●論理演算子の優先順位

実デバイスを使用した場合の例を、下記に示します。

例）R1000 = R000 OR R001 AND DM0 = 100 の代入文の場合
R1000 = (R000 OR (R001 AND (DM0 = 100))) として演算されます。

演算順1　：　DM0 = 100 の比較（ON or OFF)　…… 比較結果①
演算順2　：　比較結果①と R001 の論理積　…… 結果②
演算順3　：　結果②と R000 の論理和　…… 結果③
演算順4　：　結果③を R1000 に代入

●優先順位の入れ換え

( )括弧を使用すると、“( )”の内側の演算は外側の演算よりも常に先に実行されますので、優先順序を入れ換えたり、式のある部分を他の部分よりも先に強制的に評価できます。ただし、“( )”内では演算子の優先順序は保持されます。

例）DM5000 = DM3000 ∗ (DM3001 + (DM3002 + DM3003 ∗ 5) / 10)の場合

演算順1　：　DM3003 ∗ 5　…… 結果①
演算順2　：　DM3002 + 結果①　…… 結果②
演算順3　：　結果② / 10…… 結果③
演算順4　：　DM3001 + 結果③　…… 結果④
演算順5　：　DM3000 ∗ 結果④　…… 演算結果
演算順6　：　演算結果を DM5000 に代入

参 考　複雑な演算や順番の紛らわしい演算を記述する場合には、演算子の優先順位に関わらず、“( )”を使用して演算手順を明確に記述しておくと、後々の保守(可読)性が向上します。

代入文と比較演算子の違い

代入文で使用する(=)と比較演算子で使用する(=)では、同じ記号を使用しますが、それぞれが持つ意味(使用方法)は異なります。

- 代入としての(=)･ ･･････････代入を含む文だけで(文が)成り立つ
  意味:右辺の結果(値)を左辺に代入する

例)
```
1 SELECT CASE DM2000          ' SELECT制御文
2 CASE 1
3         DM1000 = 100        ' 処理文として単独使用
4 CASE 2
5         DM3000 = DM1000 + 5 ' 処理文として単独使用
6 END SELECT
```

- 比較演算子としての(=)･ ･･･比較演算子を含む文だけでは文が成り立たない
  (条件分岐文などの制御文の一部として取り扱われる)
  意味:右辺と左辺を比較し、ON(真)かOFF(偽)を判定する

例)
```
7 IF DM1000 = 100 THEN        ' 制御文(IF文)の一部
8         BMOV(DM10,DM20,4)   ' 処理文
9 END IF
```

※ 代入文の例3行目と比較演算子の例7行目は同じ「DM1000 = 100」と記述されていますが、それぞれ意味が異なります。

制御文

制御文とはプログラムの流れを変えるもので、条件により処理の流れを分岐させたり、繰り返しをおこない、実行の選別や回数を変えていきます。

制御文一覧

制御文には、次に示す「条件分岐文」と「繰り返し制御文」の2種類があります。

■条件分岐文

条件分岐文は、指定した条件で処理の流れを分岐させます。

制御文	制御内容	
IF文	①条件式に従って処理を実行する分岐制御文	
	IF　条件式A　THEN **　　処理文B** **END　IF**	• 条件式 A が成り立つとき、処理文 B を実行します。条件式 A が成り立たないとき、何も処理しません。
	②条件式に従って別々の処理を実行する2分岐制御文	
	IF　条件式A　THEN **　　処理文B** **ELSE** **　　処理文C** **END　IF**	• 条件式 A が成り立つとき、処理文 B を実行します。条件式 A が成り立たないとき、処理文 C を実行します。
	③複数の条件式に従って別々の処理を実行する多分岐制御文	
	IF　条件式A　THEN **　　処理文B** **ELSE IF　条件式C　THEN** **　　処理文D** ⋮ **END　IF**	• 条件式 A が成り立つとき、処理文 B を実行します。条件式 A が成り立たないで条件式 C が成り立つとき、処理文 D を実行します。 • 条件式 A,C･･･のいずれも成り立たないのとき、何も処理しません。
SELECT文	条件値に従って対応する処理を選択して実行する多分岐制御文	
	SELECT CASE　比較元 **CASE 条件値A** **　　処理文B** **CASE 条件値C** **　　処理文D** ⋮ **CASE ELSE** **　　処理文E** **END SELECT**	• 比較元と条件値 A が等しいとき、処理文 B を実行します。比較元と条件値 C が等しいとき、処理文 D を実行します。 • 比較元が条件値 A,C･･･以外のとき、処理文 E を実行します。
MC文	マスターコントロールの制御文	
	MC　条件式A　THEN **　　処理文B** **MCR**	• 条件式 A が成り立つとき、処理文 B を実行します。条件式 A が成り立たないとき、処理文 B を OFF で実行します。

■繰り返し制御文

繰り返し制御文は、指定した条件で繰り返して処理を実行させます。

制御文	制御内容	
FOR文	指定した回数の条件式が成立するまで処理を繰り返すループ制御文	
	FOR 初期値 **TO** 最終値　**STEP** 増分値 　　処理文**A** **NEXT**	• 初期値から最終値まで、増分値を加算しながら、処理文を繰り返し実行します。
WHILE文	条件式が成立している（真である）間、処理を繰り返す不定回数ループ制御文	
	WHILE 条件式 　　処理文 **END WHILE**	• 条件式が成立している間、処理文を繰り返し実行します。
DO文	条件式が成立する（真になる）までの間、処理を繰り返す不定回数ループ制御文	
	DO 　　処理文 **UNTIL** 条件式	• 条件式が成り立たない間、処理文を繰り返し実行します。
BREAK文	繰り返し制御文の中で、繰り返し条件とは関係なくループ制御から抜ける	
	<繰り返し処理文> 　　⋮ 　　**BREAK** 　　⋮ **<繰り返し処理文>**	• BREAKが実行されると、条件に関係なくBREAKを含む繰り返し制御を終了します。

IF ～ 文(条件分岐 ①)

1つの条件式で処理をおこなうかどうかを判断する、最も簡単な条件分岐文です。

IF ～ THEN ～ ELSE ～ END␣IF	
基本書式	**書式説明**
IF␣＜条件式＞␣THEN ↵ ＜処理文1＞ ↵ END␣IF ↵	もし条件式が成り立つ時、 処理文1を実行します。 IF文はここまで

※ ␣は半角スペース挿入を表します。
↵マークがあるところは、必ず改行を入れてください。

書式例	解説

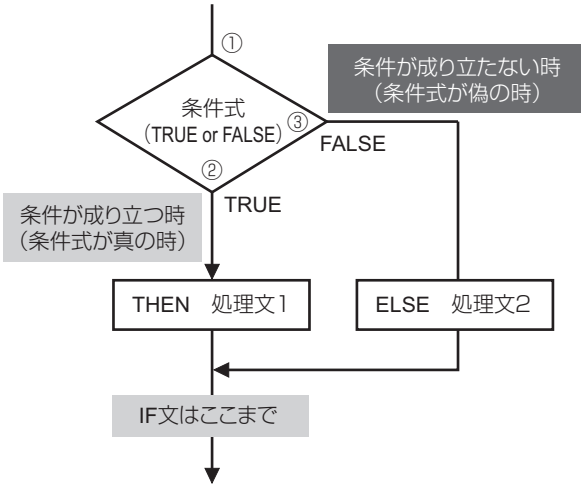
IF R1000 = ON THEN
DM1000 = DM1000 + 1
END IF
	'R1000がONのとき
	'DM1000に定数1を加算します。

ワンポイント

- 条件式には、結果がビット型になる(ブール値を返す)式であれば、どのような式でも使用できます。
- 処理文は、THEN～END IFの間に記述する一連の処理で、必要なだけ複数行に渡って記述できます。
- IF文はネスト(入れ子)も可能ですので、処理文の中にさらにIF文を記述することにより、AND(かつ)条件で絞り込みながらの多重分岐処理がおこなえます。

参 考　IF文のネスティング(IF文の中にIF文を記述する)は、エリアスクリプトでの記述で最大200重、ボックススクリプトでの記述で最大199重まで可能です。
「制御文のネスティング(入れ子)について」(4-32ページ)

処理の流れ



①条件式で「TRUE(真)」か「FALSE(偽)」を判断します。

②条件式が「TRUE(真)」の場合、THEN以下の処理文を実行し、END IFを実行します。

③条件式が「FALSE(偽)」の場合、END␣IFまでジャンプし、何も実行せずにEND␣IF以降の処理に移ります。

サンプル記述例

スクリプト記述

```
IF (10 <= DM1000) AND (DM1000 <=_
    50) THEN                          'DM1000が10以上で50以下の
                                      'とき
FM1000.T ="範囲内"                     'FM1000.Tに指定の文字列を
                                      '格納します。
END IF                                'IF文おわり
```

動作説明

DM1000の値が10以上で50以下の場合、FM1000.Tに文字列"範囲内"を格納します。
値が範囲外の場合、何も実行しません。

参考

「'(シングルクォーテーション)」(または「REM」)の記述以降、行末(改行が入力されるまで)の部分はコメントとして扱われますので、実際の動作には何も影響しません

「コメント機能について」(2-69ページ)

IF ～ ELSE文(条件分岐 ②)

1つの条件式で処理方法を2通りに分けることのできる、簡単な条件分岐文です。

IF ～ THEN ～ ELSE ～ END␣IF	
基本書式	**書式説明**
IF␣＜条件式＞␣THEN ↵	もし条件式が成り立つ時、
＜処理文1＞ ↵	処理文1を実行します。
ELSE ↵	条件式が成り立たない時、
＜処理文2＞ ↵	処理文2を実行します。
END␣IF ↵	IF文はここまで

※ ␣は半角スペース挿入を表します。
↵マークがあるところは、必ず改行を入れてください。

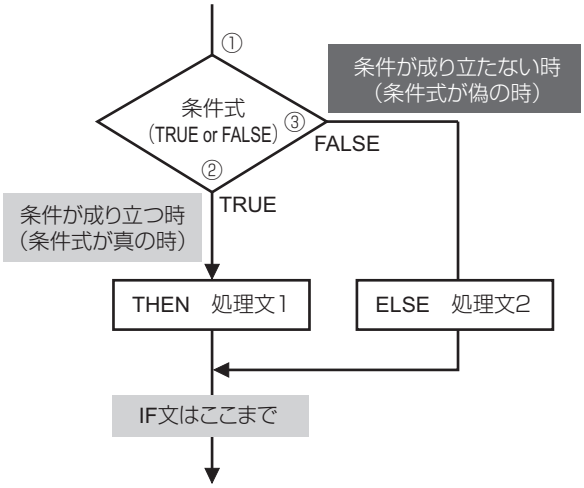
書式例	解説
IF R1000 = ON THEN	'R1000がONのとき
DM2000 = 50	'DM2000に定数50を格納します。
ELSE	'R1000がOFFのとき
DM2000 = 10	'DM2000に定数10を格納します。
END IF	

ワンポイント

- 条件式には、結果がビット型になる(ブール値を返す)式であれば、どのような式でも使用できます。
- 処理文は、THEN～END IFの間に記述する一連の処理で、必要なだけ複数行に渡って記述できます。
- IF文はネスト(入れ子)も可能ですので、処理文の中にIF文を記述することで、AND(かつ)条件で絞り込みながらの多重分岐処理がおこなえます。

参 考　IF文のネスティング(IF文の中にIF文を記述する)は、最大200重まで可能です。
「制御文のネスティング(入れ子)について」(4-32ページ)

処理の流れ



①条件式で「TRUE(真)」か「FALSE(偽)」を判断します。

②条件式が「TRUE(真)」の場合、THEN以下の処理文1を実行し、END␣IFまでジャンプします(途中の式は実行しません)。

③条件式が「FALSE(偽)」の場合、THEN～ELSE間は実行されず、ELSE以下の処理文2を実行します。

サンプル記述例

スクリプト記述

```
IF R1000 THEN                          'R1000がONのとき(R1000=ONの
                                       '省略記法)
   DM1000.F = COS(DM1100)              'COS関数を処理
ELSE                                   'R1000がONでないとき(OFFのとき)
   IF R0002 THEN                       '入れ子のIF文:R0002がONのとき
      DM1000.F = SIN(DM1100)           'SIN関数を処理
   ELSE                                'R0002がOFFのとき
      BMOV(DM5000,DM5500,10)           'BMOV関数を処理
   END IF                              '入れ子のIF文おわり
END IF                                 'IF文おわり
```

動作説明

接点R1000がONのとき、DM1000.FにCOS(DM1100)の結果を格納します。
接点R1000がON以外(OFF)のときで、接点R0002がONのとき、DM1000.FにSIN(DM1100)の戻り値を格納します。
R1000がON以外(OFF)のときで、R0002もON以外(OFF)のとき、BMOV関数を使用して、DM5000のデータを10ワード分DM5500に転送します。

「SIN関数」(7-140ページ)
「COS関数」(7-142ページ)
「BMOV関数」(7-2ページ)

IF ～ ELSE␣IF ～ 文(条件分岐 ③)

最初の条件式を満たさない場合、次の条件式にて再度判定をおこないます。条件をいくつも持てる条件分岐文です。

IF ～ THEN ～ ELSE␣IF ～ END␣IF		
基本書式		**書式説明**
IF␣<条件式1>␣THEN	↵	条件式1が成り立つ時、
<処理文1>	↵	処理文1を実行します。
ELSE␣IF␣<条件式2>␣THEN	↵	条件式1は成り立たないが、条件式2が成り立つ時
<処理文2>	↵	処理文2を実行します。
ELSE␣IF␣<条件式3>␣THEN	↵	条件式1,2は成り立たないが、条件式3が成り立つ時 } ※2
⋮	⋮	⋮
ELSE※1	↵	全ての条件式が成り立たない時、
<処理文n>※1	↵	処理文nを実行します。
END␣IF	↵	IF文はここで終わりです。

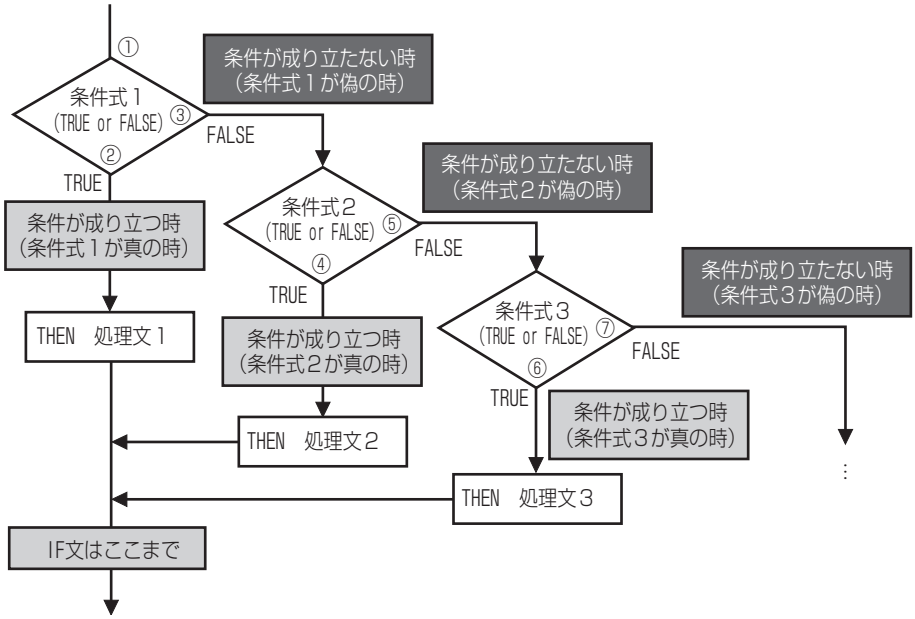
※ ␣は半角スペース挿入を表します。
↵マークがあるところは、必ず改行を入れてください。
※1　省略可能です。
※2　任意の数の条件式と処理文を設定できます。上限は有りません。

書式例	解説
IF R1000 THEN	'R1000がONのとき(R1000=ONの省略記法)
DM1000 = 1000	'DM1000に1000を格納します。
ELSE IF R1001 THEN	'R1000がONでなく、R1001がONのとき
DM1000 = 1001	'DM1000に1001を格納します。
ELSE IF R1002 THEN	'R1000,R1001がONでなく、R1002がONのとき
DM1000 = 1002	'DM1000に1002を格納します。
ELSE IF R1003 THEN	'R1000,R1001,R1002がONでなく、R1003がONのとき
DM1000 = 1003	'DM1000に1003を格納します。
ELSE IF R1004 THEN	'R1000,R1001,R1002,R1003がONでなく、R1004がONの時
DM1000 = 1004	'DM1000に1004を格納します。
ELSE	'R1000,R1001,R1002,R1003,R1004の全てがONでない時
DM1000 = 1005	'DM1000に1005を格納します。
END IF	

ワンポイント

- 条件式には、結果がビット型になる(ブール値を返す)式であれば、どのような式でも使用できます。
- 処理文は、制御文の間に記述する一連の処理で、必要なだけ複数行に渡って記述できます。
- ELSE IF文はIF文と異なり、複数行記述してもネスト構造にはなりません。

処理の流れ



①条件式1で「TRUE（真）」か「FALSE（偽）」を判断します。

②条件式1が「真」の場合、THEN以下の処理文1を実行し、END␣IFまでジャンプします(途中の式は実行しません)。

③条件式が「偽」の場合、条件式2で「真」か「偽」を判断します。

④条件式2が「真」の場合、処理文2を実行し、END␣IFまでジャンプします(途中の式は実行しません)。

⑤条件式2が「偽」の場合、条件式3で「真」か「偽」を判断します。

⑥条件式3が「真」の場合、処理文3を実行し、END␣IFまでジャンプします(途中の式は実行しません)。

⑦条件式3が「偽」の場合、条件式4で「真」か「偽）」を・・・・以下同じ

SELECT ␣ CASE ～ 文(多分岐制御)

基準となるワードデータに格納されている値により、処理を選択できる条件分岐文です。

SELECT␣CASE ～	
基本書式	**書式説明**
SELECT␣CASE␣ワードデータ ↵	ワードデータの値を基準に条件分岐
CASE␣定数1 ※3 ↵	定数1とワードデータの値が等しい時
処理文1 ↵	処理文1を実行します。
CASE␣定数2 ↵	定数2とワードデータの値が等しい時
処理文2 ↵	処理文2を実行します。
CASE␣定数3 ↵	定数3とワードデータの値が等しい時 ※2
⋮ ⋮	⋮
CASE␣ELSE※1 ↵	定数とワードデータの値が何れも等しくない場合、
処理文n※1 ↵	処理文nを実行します。
END␣SELECT ↵	セレクト文は、ここで終わります。

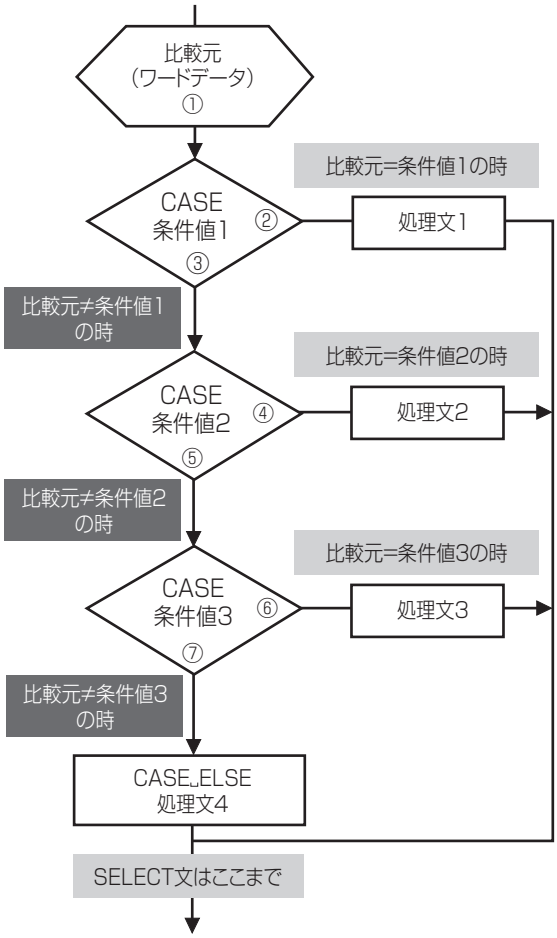
※ ␣は半角スペース挿入を表します。
↵マークがあるところは、必ず改行を入れてください。

※1　省略可能です。
※2　《CASE定数(ELSE)》は最大200個まで記述可能です。
※3　定数1に、文字列や文字列を扱うデバイスおよび変数、ビット型デバイスやBOOL型の変数は指定できません。

書式例	解説
SELECT CASE DM1000	'DM1000に格納されている値を基準とします。
CASE 1	'DM1000 = 1のとき
DM2000 = 1000	'DM2000に1000を格納します。
CASE 2 TO 10	'DM1000 = 2～10のとき
DM2000 = 2000	'DM2000に2000を格納します。
CASE 11,15,60	'DM1000 = 11 or 15 or 60のとき
DM2000 = 3000	'DM2000に3000を格納します。
CASE IS > 100	'DM1000が100より大きい時※
DM2000 = 4000	'DM2000に4000を格納します。
CASE ELSE	'DM1000の値が上記以外のとき
DM2000 = 5000	'M2000に5000を格納します。
END SELECT	

※ 定数部には比較演算子を使用した条件式を書くことができます。その場合、「IS」を用いて比較します。
(例) IS >= 100(100以上)、IS < > 30(30以外)

処理の流れ



①以降の条件値との判断基準となる値を宣言します。

②条件値1と比較元を比較して、「条件値1=比較元」が成り立つ時、処理文1を実行します。処理文1を実行した後、END␣SELECTにジャンプします。

③「条件値1=比較元」が成り立たなかった時、次の条件値比較に移ります。

④条件値2と比較元を比較して、「条件値2=比較元」が成り立つ時、処理文2を実行します。処理文2を実行した後、END␣SELECTにジャンプします。

⑤「条件値2=比較元」が成り立たなかった時、次の条件値比較に移ります。

⑥条件値3と比較元を比較して、「条件値3=比較元」が成り立つ時、処理文3を実行します。処理文3を実行した後、END␣SELECTにジャンプします。

⑦「条件値3=比較元」が成り立たなかった時、処理文4を実行します。

ワンポイント

- 選択される定数は、単一のものでも、複数指定･範囲指定を含んだものでも可能です。

  例）　CASE 1 …… 単一指定（ 1のみ ）
  　　CASE 3,9,12,50 …… 複数指定（ 3 or 9 or 12 or 50 ）
  　　CASE 8 TO 20 …… 範囲指定（8以上、20以下の範囲）
  　　CASE IS>=100 …… 比較演算子を用いた範囲指定（100以上）

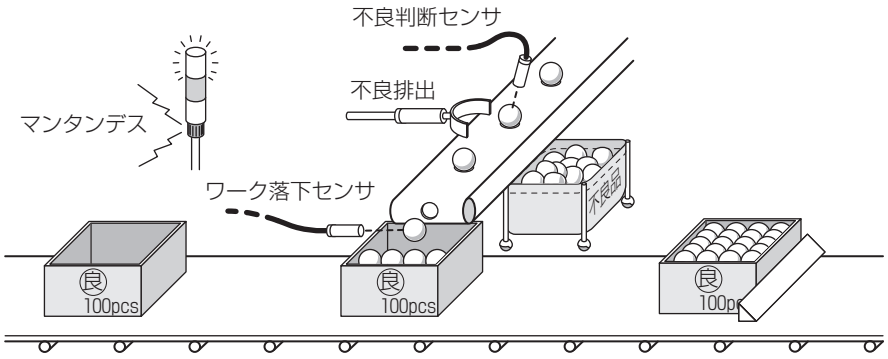
- CASE ELSE　<処理文>の記述は省略可能です。
  ただし、この『もしCASE条件が全て成り立たなかったとき』というCASE ELSEは、想定外の事象が発生した場合に問題解決の一助になります。

スクリプト参考例

- コンベアから流れ落ちるワークの数をカウントします。ワークが箱に落ちる時、良品カウンタ(DM1000)に1を加算します。
  良品が100個に達したら、箱を入れ替えます。良品が200個に達したら、チャイムを鳴らしてから箱を入れ替えます。良品が300個に達した時、作業者を呼ぶ為にアラームを鳴らして、運転を停止させ、良品カウンタの値をリセットします。

【スクリプト参考例】

```
IF LDP(R000) = ON THEN          'ワーク落下検出センサ(R000)がONのとき
   DM1000 += 1                  '良品カウンタ(DM1000)に1を加算します
END IF
SELECT CASE DM1000              '比較元をDM1000にします
CASE 100                        'DM1000の値が100になったとき
   《箱を入れ替える》
CASE 200                        'DM1000の値が200になったとき
   《チャイムを鳴らす》
   《箱を入れ替える》
CASE 300                        'DM1000の値が300になったとき
   《アラームを鳴らす》
   《運転停止》
   DM1000 = 0                   'DM1000の値の値をリセットします
CASE ELSE                       'DM1000の値が100、200、300以外のとき
   《運転中表示》
   《警告灯の青点灯》
END SELECT                      'SELECT CASE文はここで終わり
```



ポイント

選択される定数が複数存在する場合、上行に書かれている処理文を優先的に実行し、以降に書かれている処理文は実行しません。

スクリプト記述

```
SELECT CASE DM1000      '選択基準をDM1000とします。
CASE 1                  'DM1000の値が1のとき
  処理文1
CASE 3 TO 20            'DM1000の値が3以上20以下のとき
  処理文2
CASE IS >= 15           'DM1000の値が15以上のとき
  処理文3
CASE ELSE               'DM1000の値が上記以外の値のとき
  処理文4
END SELECT
```

※ DM1000(比較元)に格納されている値が17のとき、2つ目のCASEと3つ目のCASEに該当しますが、この場合、2つ目のCASEが優先的に選択されて処理文2が実行されます。処理文実行後は、END␣SELECTにジャンプするため、処理文3は実行されません。

MC ～ MCR文(マスターコントロール)

マスターコントロールをスクリプトで記述します。
マスターコントロールの動作については 『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』を参照してください。

MC ～ THEN ～ MCR	
基本書式	**書式説明**
MC␣＜MC実行条件＞␣THEN ↵	MC実行条件が成り立つ時、マスターコントロール処理をおこないます。
＜処理文1＞ ↵	処理文を実行します。
MCR ↵	MC実行条件が成り立たない時は、処理文をOFFの状態で実行します。

※ ␣は半角スペース挿入を表します。
↵マークがあるところは、必ず改行を入れてください。

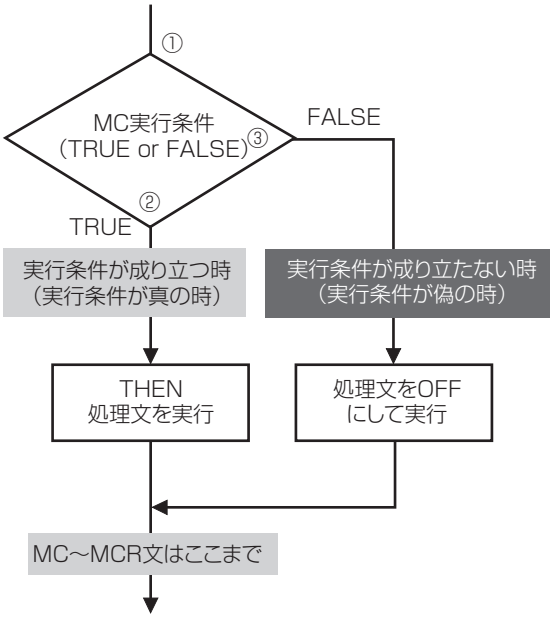
書式例	解説
MC R1000 THEN	'R1000がONのときマスターコントロール処理を行います。
R500 = ON	'R500をONにして実行します。
MCR	'R1000がOFFのとき、R0500をOFFにして実行します。

MC実行条件がOFFのときは、MC～MCR間の関数をOFFの状態で実行します。
詳細は、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「MC、MCR命令」を参照してください。

ワンポイント

- MC実行条件には、結果がビット型になる(ブール値を返す)式であれば、どのような式でも使用できます。
- 処理文は、MC～MCRの間に記述する一連の処理で、必要なだけ複数行に渡って記述できます。

処理の流れ



①MC実行条件で「TRUE(真)」か「FALSE(偽)」を判定し、MC～MCR間の処理方法を決めます。

②実行条件が「真」の場合、THEN以下の処理文を実行します。

③条件式が「偽」の場合、THEN以下の処理文をOFFにして実行します。

ポイント

MC～MCR文だけで構成されるネスティング(MC～MCR文の中にMC～MCR文を記述する)は、最大16重まで可能です。

「制御文のネスティング(入れ子)について」(4-32ページ)

FOR ～ NEXT文(指定回数の繰り返し制御)

繰返し条件により、処理文の実行回数を決めることのできる制御文です。

FOR ～ NEXT	
基本書式	
1	FOR␣<ワードデータ>=<初期値>␣TO␣<最終値>[␣STEP␣<増分値>]※1 ↵
2	<処理文> ↵
3	NEXT ↵
書式説明	
1	・繰返し制御の回数を制限するワードデータの初期化と、 終了値を決めます。STEPで処理毎にワードデータの値をいくつ増加させるかを決めます(この行では増加させません)。
2	・ワードデータが最終値を超えるまで、 毎回処理文を実行します。
3	・ワードデータに増分値を加算します。

※1 [ ]内の記述は省略可能です。省略した場合STEP␣1として実行します。
␣は半角スペース挿入を表します。
↵マークがあるところは、必ず改行を入れてください。

書式例	解説
FOR DM1000 = 0 TO 100 STEP 5	'DM1000の値が0から100を超えるまで、繰り返します。
DM2000 = DM2000 + 5	'DM2000に5を加えた結果をDM2000に格納します。
NEXT	'DM1000の現在値に増分値5を加算して、FOR～に戻ります。

ワンポイント

書式例の1行目では、多くの要素が記述されています。それらを個別に解説します。

1行目： $\underbrace{\text{FOR␣}}_{④}\underbrace{\text{DM1000=0}}_{①}\underbrace{\text{␣TO␣100}}_{②}\underbrace{\text{␣STEP␣5}}_{③}$

①指定するワードデータは何を使用してもかまいません。まず、初期値を格納する必要があります。この初期値の格納は、FOR～NEXT文が実行される最初の1回だけ処理されます。

②最終値を指定します。FOR～NEXTは指定したワードデータの値が最終値を超えたかどうかを判断し、「処理文を実行する(最終値を超えていない場合)のか」、「繰返し制御を終了する(最終値を超えた場合)のか」、という分岐をします。

③増分値は省略可能です。省略した場合は「STEP␣1」という処理をおこないます。指定したワードデータの増分処理は、最初の1回目は実行しません。実行のタイミングは、NEXT文(書式例の3行目)を実行する直前になります。

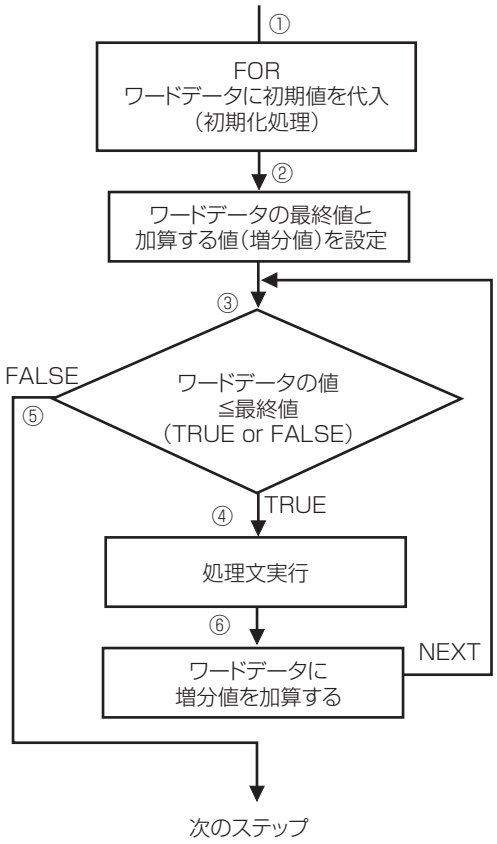
④毎回、指定したワードデータの値が要件を満たしているかどうかの判定をします。

参考 STEPで記述した定数の符号により、「ワードデータの値>最終値」か「ワードデータの値<最終値」の判断を自動的に切替えます。

例） FOR DM1100 = 100 TO 0 STEP － 5

DM1100に格納されている値を100から5づつ減算して0未満になるまで実行します。

処理の流れ



①ワードデータに初期値を代入します。
②繰返し処理の終わりを表す最終値を設定します。また、ワードデータの値に加算する増分値も設定します。
③ワードデータの値が最終値よりも小さいかどうかを判定します。
④③の条件が「TRUE（真）」のとき、処理文を実行します。
⑤「FALSE（偽）」のときは繰返し処理文を終了します。
⑥ワードデータの値に増分値を加算して、③にもどります。

ポイント

加算する値(増分値)に単精度浮動小数点型実数を記述する場合、指定するワードデータには .Fサフィックスが必要です。

例）加算する値を「0.5」にする場合

```
FOR DM2000.F = 0 TO 100 STEP 0.5      'DM2000は.F型でなければならない
        ⋮
```

参考

FOR～NEXT文は、NEXT文実行時に以下の処理を実行します。

- 指定ワードデータ＋＝増分値

FOR～NEXT文を使用するときの注意

●STEPで指定できる範囲

FOR～NEXT文の記述で、STEPで指定する増分値は以下のとおりです。

STEPで指定できる範囲：－32767～32767(－32768は指定できません)

範囲外の値を指定した場合、スクリプト変換時にエラーが出ます。

●STEPで単精度浮動小数点型実数を指定する場合

STEPで指定する増分値には、小数点付き定数も指定できますが注意が必要です。

小数点付き定数は、自動的に単精度浮動小数点型実数に変換されます。そのときに若干の誤差が生じるため、繰り返し回数にも誤差が生じる場合があります。

サンプル記述例

●FOR～NEXT文を強制終了させる場合

FOR～NEXT文で設定した条件を満たす前に、別の条件で繰返し制御文を強制的に終了させたい場合、「BREAK文」を使用します。

スクリプト記述

```
FOR DM1000 = 0 TO 1000 STEP 10      'DM1000の値が1000を超えるまで
                                    '繰り返します。
DM2000 += 9                         'DM2000の値を毎回9加算します。
IF DM2000 > 100 THEN                'DM2000の値が100より大きく
                                    'なったら
BREAK                               'この繰返し制御を終了します。
END IF                              'IF文終了
NEXT                                '増分値を加算してFOR～に戻る。
```

詳細は、「BREAK処理」(4-31ページ)を参照してください。

ポイント

繰返し制御文では、必ず繰返し処理が終了するようにスクリプトを記述してください。繰返し処理が終了しないと「無限ループ(永久に繰返し処理をおこなう)」となり、プログラム実行時にスキャンタイムオーバーとなります(無限ループは変換時にエラーとして検出されません)。

「繰返し制御文と処理時間」(4-35ページ)

参考

FOR～NEXT文のネスティングは、最大100重まで可能です。

「制御文のネスティング(入れ子)について」(4-32ページ)

WHILE文(前判定繰返し制御)

繰返し条件が「TRUE(真)」の間、繰返し処理文を実行する制御文です。

WHILE ～ END␣WHILE	
基本書式	**書式説明**
WHILE␣<条件式>↵	条件式が成り立つ時、※1
<処理文>↵	処理文を実行します。
END␣WHILE↵	文頭(WHILE)に戻ります。

※ ␣は半角スペース挿入を表します。
↵マークがあるところは、必ず改行を入れてください。
※1　条件式が成り立たない時、繰返し処理を終了します。

書式例	解説
WHILE DM1000 < 100	'DM1000の値が100未満の場合、※2
DM1000 += 5	'DM1000に5を加算します。
END WHILE	'文頭(WHILE)に戻る。

※2　DM1000の値が100以上だったら、処理文を実行せずにEND WHILEの次の行から実行します。

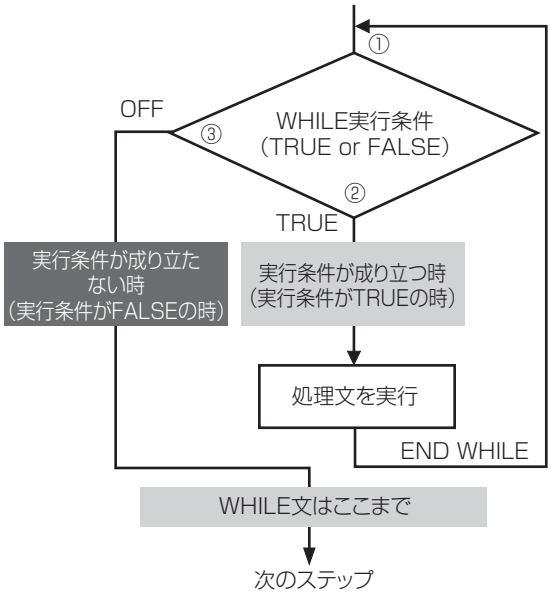
ワンポイント

- 条件式には、結果がビット型になる(ブール値を返す)式であれば、どのような式でも使用できます。
- 処理文は、WHILE～END　WHILEの間に記述する一連の処理で、必要なだけ複数行に渡って記述できます。

参考

WHILE文･DO～UNTIL文の繰返し制御文では、それらの組み合せで構成されるネスティングが、最大199重まで記述可能です。
マスターコントロールやIF文と組み合わせたネスティングも可能です。
「制御文のネスティング(入れ子)について」(4-32ページ)

処理の流れ



①WHILE実行条件で「TRUE(真)」か「FALSE(偽)」を判定し、WHILE～END」WHILE間の処理方法を決めます。

②実行条件が「TRUE」の場合、処理文を実行し、①(WHILE文の文頭)に戻ります。

③条件式が「FALSE」の場合、END WHILE文の次のステップに進みます。

サンプル記述例

●WHILE文を強制終了させる場合

WHILE～END」WHILE文で、繰り返し終了条件を満たす前に、別の条件で繰返し制御文を強制的に終了させたい場合、「BREAK文」を使用します。

スクリプト記述

```
'繰返し制御文からBREAK文で抜ける
WHILE R1000 = OFF                    'R1000がOFF(条件がON(真))の間、
                                     '繰り返します。
  IF (DM2000 MOD 8) = 0 THEN         'DM2000が8の倍数になったら
    ZRES(DM1000, DM1100)             'DM1000～DM1100をリセット
                                     'します。
  ELSE IF DM2000 > 100 THEN          'DM2000の値が100より
                                     '大きくなったら
    BREAK                            'この繰返し制御を終了します。
  END IF                             'IF文終了
  DM2000 += 1                        'DM2000に1を加算します。
END WHILE                            'WHILE～に戻ります。
```

詳細は、「BREAK処理」(4-31ページ)を参照してください。

ポイント

繰返し制御文では、必ず繰返し処理が終了するようにスクリプトを記述してください。繰返し処理が終了しないと「無限ループ(永久に繰返し処理をおこなう)」となり、プログラム実行時にスキャンタイムオーバーとなります(無限ループは変換時にエラーとして検出されません)。

「繰返し制御文と処理時間」(4-35ページ)

DO～UNTIL文(後判定繰返し制御)

繰返し条件が「FALSE(偽)」の間、繰返し処理文を実行する制御文です。

DO ～ UNTIL	
基本書式	**書式説明**
DO ↵	繰り返し制御をおこないます。
＜処理文＞ ↵	処理文を実行します。
UNTIL␣＜条件式＞ ↵	条件式が成り立たない時、文頭(DO)に戻ります。※1

※ ␣は半角スペース挿入を表します。
↵マークがあるところは、必ず改行を入れてください。

※1　条件式が成り立つ時、繰返し処理を終了します。

書式例	解説
DO	'次ステップからの処理をおこないます。
DM1000 += 5	'DM1000に5を加算します。
UNTIL DM1000 >= 100	'DM1000の値が100以上でない場合、文頭(DO)に戻ります。※2

※2　DM1000の値が100以上だったら、処理文を実行せずにUNTILの次の行から実行します。

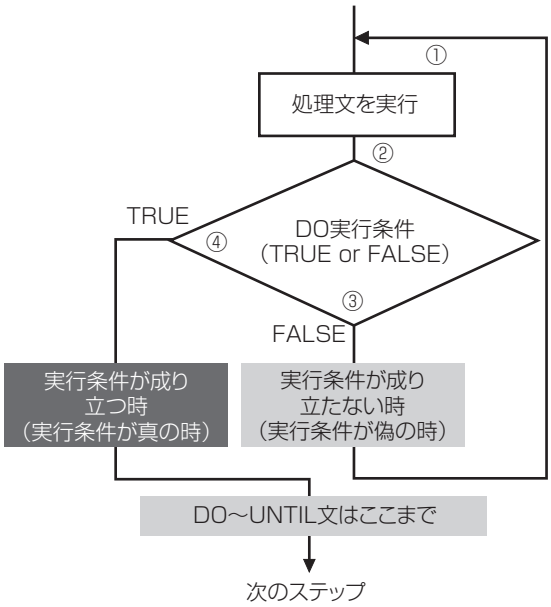
ワンポイント

- 条件式には、結果がビット型になる(ブール値を返す)式であれば、どのような式でも使用できます。
- 処理文は、DO～UNTILの間に記述する一連の処理で、必要なだけ複数行に渡って記述できます。
- DO～UNTIL文と同じような動作(繰り返し処理)をする制御文として、WHILE文があります。
- DO ～ UNTIL 文は、先に処理をおこなった後、条件式の判定をおこないます。繰り返す為の判定条件は、FALSE(偽)です。
  詳細は、「WHILE文(前判定繰返し制御)」(4-27ページ)を参照してください。

ポイント

WHILE文・DO～UNTIL文の繰返し制御文では、それらの組み合せで構成されるネスティングが、最大199重まで記述可能です。
マスターコントロールやIF文と組み合わせたネスティングも可能です。
「制御文のネスティング(入れ子)について」(4-32ページ)

処理の流れ



①処理文を実行します。

②DO実行条件で「TRUE(真)」か「FALSE(偽)」を判定し、DO～UNTIL間の処理方法を決めます。

③条件式が「FALSE(偽)」の場合、①(DO～UNTIL文の文頭)に戻ります。

④条件式が「TRUE(真)」の場合、次のステップに進みます。

WHILE文とDO文の違いについて

DO～UNTIL文と同じような動作(繰り返し処理)をする制御文としてWHILE文があります。
違いは下記のとおりです

	WHILE文	DO文
実行順序の違い	1. 条件式の判定(真か偽か) 2. 繰り返し処理文の実行 3. 文頭に戻る	1. 繰り返し処理文の実行 2. 条件式の判定(真か偽か) 3. 文頭に戻る
条件式の判定の違い	条件が「TRUE(真)」のとき繰り返す	条件が「FALSE(偽)」のとき繰り返す

詳細は、「WHILE文(前判定繰返し制御)」(4-27ページ)を参照してください。

WHILE文･･･････前判定繰り返し制御文(条件によっては処理文を1回も実行しない)
DO文･････････後判定繰り返し制御文(必ず1回は処理文を実行する)

ポイント

繰返し制御文では、必ず繰返し処理が終了するようにスクリプトを記述してください。繰返し処理が終了しないと「無限ループ(永久に繰返し処理をおこなう)」となり、プログラム実行時にスキャンタイムオーバーとなります(無限ループは変換時にエラーとして検出されません)。

「繰返し制御文と処理時間」(4-35ページ)

BREAK処理

繰返し制御を強制的に終了(繰返しから抜ける)する際に使用します。

BREAK	
基本書式	**書式説明**
＜繰り返し制御文の始まり＞ ↵	繰返し処理の中に記述します。※1
BREAK ↵	この繰返し処理を、強制的に終了します。
＜繰り返し制御文の終わり＞ ↵	繰返し制御文の終わり

※ ↵マークがあるところは、必ず改行を入れてください。

※1　BREAKは必ず、繰返し処理文の中に記述してください。

書式例	解説
FOR DM1000 = 0 TO 50 STEP 2	'DM1000が0から50になるまで、繰り返します。
DM2000 += 5	'DM2000に5を加算します。
IF DM2000 > 70 THEN	'もし、DM2000の値が70以上になったなら、
BREAK	'この繰り返し制御(FOR～NEXT文)を強制終了します。
END IF	'IF文の終わりです。
NEXT	'DM1000に5を加算して、FOR文の文頭に戻ります。

※ 繰返し処理文はFOR～NEXT以外にも、WHILE文、DO～UNTIL文の文中にも使用できます。

ポイント　**BREAKは必ず繰返し処理文の内側に書かなければいけません。**
繰返し処理文の外に書くと、変換エラーとなります。

サンプル記述例

ネスティングしている繰返し制御文にBREAK文を記述した場合、BREAKが記述されている繰り返し制御文のみ終了します。

スクリプト記述

```
FOR DM1000 = 1 TO 100 STEP 3                    'FOR文①
  FOR DM2000 = 1 TO 100 STEP 3                  'FOR文②
    FOR DM3000 = 1 TO 100 STEP 3                'FOR文③
      DM4000 = DM1000 + DM2000 + DM3000         '処理文
      IF DM4000 > 100 THEN                      'BREAK実行条件
        BREAK
      END IF
    NEXT                                        'BREAK実行時FOR文③のみ
                                                'いったん終了します。
  NEXT
NEXT
```

動作説明　BREAK文が含まれる制御文はFOR文③なので、BREAK実行時繰り返し処理を終了するのはFOR文③のみです。

「制御文を使う時の注意」(4-32ページ)

制御文を使う時の注意

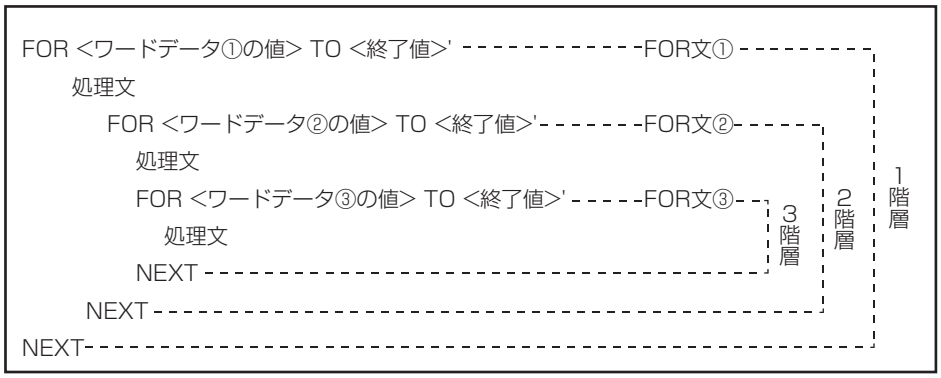
制御文のネスティング(入れ子)について

制御文は、条件分岐した処理の中や繰り返し処理の中で、また別の条件分岐や繰り返し処理をおこなうような、階層化した処理を記述できます。

■ 制御文のネスティング(入れ子)とは

ネスティング(入れ子)とは、箱の中に箱を入れるようなことで、繰り返し制御文の中にまた別の繰り返し制御文を記述して、複数の制御ブロックを多重構造にできます。以下の例は、FOR文①の中にFOR文②が、さらにその中にFOR文③が入れ子になった3重のネスティングになります。

【ネスティングの記述の例】



● ネスティングを使用する時の注意事項

すべての条件分岐文や繰り返し制御文は、ネスティングできますが、次のような制限があります。

- FOR文は、単独で階層の制限があり、最大100重までネスティングできます。
- WHILE文、DO文は、単独で階層制限があり、合わせて最大199重までネスティングできます。
- 条件分岐文(IF文、SELECT文)は単独で階層の制限があり、合わせて最大200重までネスティングできます。
- マスターコントロールは単独で階層の制限があり、16重までネスティングできます。
- ネスティングは、制御ブロック※(階層) 単位に完結させた形で入れ子にする必要があり、制御ブロック同士が交わるような記述はできません。

※ IF～END IFあるいは、FOR～NEXTなど制御文の開始から終わりまでのスクリプトを制御ブロックといいます。

ポイント **ネスティングを重ねると、プログラムがわかりにくくなります。また、スキャンタイムも伸びますので、多重のネスティングはできるだけ避けてください。**

ポイント

下図のような繰り返し制御文同士もしくは条件分岐同士、または繰り返し制御文と条件分岐文が交わるような記述はできません。

【FOR文(繰り返し制御文)とWHILE文(繰り返し制御文)が交差した場合】

```
FOR <ワードデータ①の値> TO <終了値>                    '制御文①
    処理文①
    FOR <ワードデータ②の値> TO <終了値>                '制御文②
        処理文②
        WHILE <条件式>                          '制御文③
            処理文③
    NEXT
        END WHILE
NEXT
```

制御ブロックの交差

【FOR文(繰り返し制御文)とIF文(条件分岐文)が交差した場合】

```
FOR <ワードデータ①の値> TO <終了値>                    '制御文①
    処理文①
    FOR <ワードデータ②の値> TO <終了値>                '制御文②
        処理文②
        IF <条件式> THEN                        '条件分岐文
            処理文③
    NEXT
        END IF
NEXT
```

制御ブロックの交差

■ネスティングの実動作

下記のような処理をおこなうプログラムを作成してみます。

例

繰り返し制御文(FOR文、WHILE文、DO文)を使用した3重のネスティングを構成し、実際に処理がおこなわれる回数をカウントします。

スクリプト記述

```
DM1000 = 0                      '処理回数用のカウンタを0クリアします
FOR DM2000 = 1 TO 10            'DM2000が1から10になるまで繰り
                                '返す
  '---FORループ:10回
  DM3000 = 0                    'WHILEループ用のカウンタを0クリア
                                'します
  WHILE DM3000 < 10             'DM3000が10未満の間、繰り返す
    '---WHILEループ:10回
    DM3000 += 1                 'WHILEループ用のｶｳﾝﾀに1を加算
    DM4000 = 0                  'DOループ用のカウンタを0クリアします
    DO
      '---DOループ:10回
      'ネスティングの最深部
      DM4000 += 1               'DOループ用のカウンタに1を加算
      DM1000 += 1               '処理回数用のカウンタに1を加算
    UNTIL DM4000 >= 10          'DM4000が10以上になるまでの間
                                '繰り返す
  END WHILE
NEXT
```

動作説明

DM2000で10回のFORループを構成し、その中にDM3000で10回のWHILEループを構成し、またその中にDM4000で10回のDOループを構成して、その中(階層の最深部)で処理回数(DM01000)をカウントアップしていきます。

このプログラムを実行するとFORループ10回×WHILEループ10回×DOループ10回の繰り返しが発生しますので、ネスティングの最深部の処理は、1000回実行することになります。

繰返し制御文と処理時間

KVスクリプトでは、繰返し制御文(FOR～NEXT、WHILE～、DO～)を使うことにより、繰り返し処理を簡単に記述することができますが、PLCの許容スキャンタイムを越えない範囲で繰返し処理をおこなう必要があります。

■処理時間について

繰り返し回数が多いプログラムを記述した場合のスキャンタイムを検討します。
例) 1+2+3+‥‥200,000を計算して1～200,000の総和を求めます。

スクリプト記述

```
DM1000.D = 0
DM1002.D = 0
WHILE DM1000.D < 200000                  '①～②の処理を200,000回
                                         '繰り返します
   DM1000.D += 1                         '①
   DM1002.D = DM1002.D + DM1000.D        '②
END WHILE
```

詳細は、「WHILE文(前判定繰返し制御)」(4-27ページ)を参照してください。

上記サンプルプログラムの処理時間は以下のようになります。

1 回の繰り返し時間	→	約 1.65 μs
200,000 回の繰り返し時間	→	約 1.65 μs×200,000= 約 **330,000 μs**（**330ms**）

※上記処理時間はあくまで参考例です。実際の処理時には、接続している拡張ユニットの台数や使用環境により異なります。

この場合、繰り返し処理が1回だけだと問題ありませんが、200,000回の繰り返し処理をおこなうようになっているため、総処理時間が許容スキャンタイム(KV-8000/7500/7300/5500/5000/3000/1000の場合300ms)を超えてしまいます。

KVスクリプトでプログラムを使用すると、少ない記述で複雑なプログラムや繰り返し処理が簡単に記述できますが、スキャンタイムに注意することが必要です。

参考　ラダープログラムが許容スキャンタイム(KV-8000/7500/7300/5500/5000/3000/1000、KV Nanoシリーズの場合300ms)を越えた場合スキャンタイムオーバー(エラー番号:30)が発生します。

繰り返し制御文内の処理時間が長い(複雑な演算処理をおこなったり、マクロを含む処理を実行する)場合は、少ない繰り返し回数でも、スキャンタイムオーバーが発生する場合があります。
スキャンタイムオーバーを回避するためには、数スキャンに渡って(あるいはスキャン毎に)処理を実行するように、記述を変更する必要があります。

例）1+2+3+‥‥200,000を計算し、1～200,000の総和を求めます。
(スキャンタイムオーバーを回避する記述)

スクリプト記述

```
IF R1004 = OFF THEN
  SET(R1004)                          '演算中フラグ
   DM1000.D = 0                       'DM1000.Dの初期化
   DM1002.D = 0                       'DM1002.Dの初期化
END IF
IF R1004 AND DM1000.D < 200000 THEN
   DM1000.D  += 1                     'カウンタ
   DM1002.D = DM1002.D + DM1000.D     '値1～200,000の総和の演算
END IF
```

動作説明　R1004がOFFのとき、DM1000.DとDM1002.Dに0を格納（初期化）し、R1004をセットします。
R1004がONで、DM1000.Dが200000未満のとき、DM1000.Dの現在値に1を加算し、1～200000の値を順に加算しDM1002.Dに格納します（演算はスキャン毎に実行されます）。

詳細は、「IF ～ 文(条件分岐 ①)」（4-12ページ）を参照してください。

■無限ループについて

繰り返し処理を終了できない状態を、無限ループといいます。

【実行させてはいけない記述】

```
WHILE MR1000 = OFF
   DM1001 += 1
   DM1000 = DM1000 + DM1001
END WHILE
```

繰返し制御文を終了させる記述が処理文中にない。（MR1000がONになる条件が記述されていないので終了しません。）

無限ループ

強制終了条件の追加

スクリプト記述

```
WHILE MR1000 = OFF                    'MR1000がONすれば繰り返し処理は終了
   DM1001 += 1
   DM1000 = DM1000 + DM1001
   IF DM1000 > 1000 THEN
      BREAK
   END IF
END WHILE
```

繰返し制御文の終了条件を追加する。DM1000が1000以上になったら、BREAKを実行し、繰返し制御を終了します。

「BREAK処理」（4-31ページ）

参考　繰り返し制御を強制終了させる場合、通常はBREAK文を記述します。
上記スクリプトの場合、「BREAK」の代わりに「MR1000=ON」と記述しても繰り返し制御を終了することもできますが「プログラムの可読性を高める」という意味でも、繰り返し制御を終了させる場合は、BREAK文の記述をお奨めします。

■ 繰返し制御文中のタイマ動作について

繰返し制御文の強制終了にタイマを使用することはできません。
繰り返し制御中はタイマ･カウンタ接点が変化しないため、下記のスクリプトは無限ループになります。

【実行させてはいけないスクリプト記述】

```
TMH(T001,1)                           '10msタイマ
WHILE MR1000 = OFF                    'MR1000がONすれば繰り返し処理は終了
   DM1001 += 1
   DM1000 = DM1000 + DM1001
   IF T001.B THEN                     'BREAKはタイマ接点T001がONしたら実行
      BREAK
   END IF
END WHILE
```

タイマの接点が変化しないので処理文は実行されません。

「TMR関数」（5-22ページ）

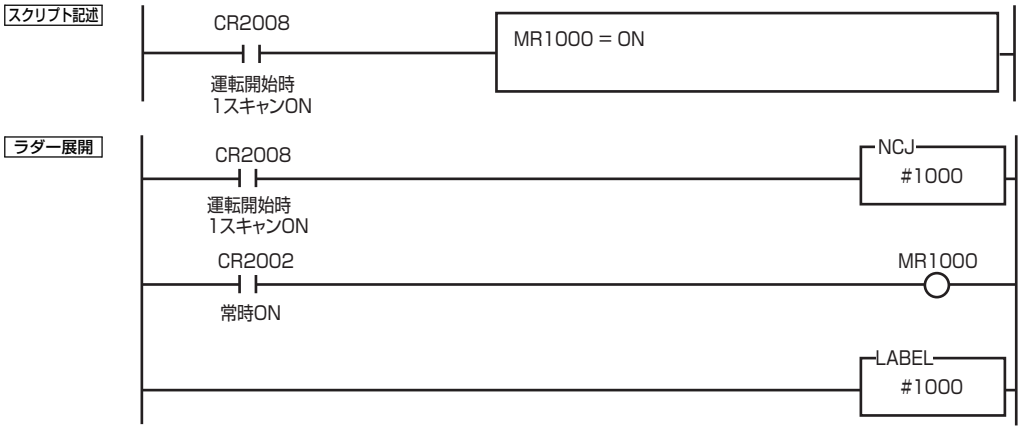
スクリプトを記述するときの注意

制御したビットデータの状態保持

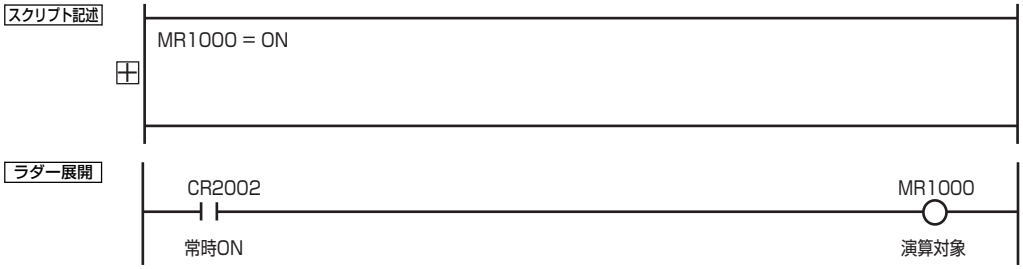
KVスクリプトでは、直前のビットデータの状態を変化させる記述(ON/OFFの代入)ができます。

例）MR1000をON状態にします(ONを代入する)。

【ボックススクリプトの場合】



【エリアスクリプトの場合】



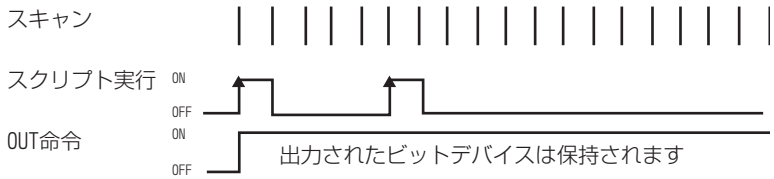
KVスクリプトで記述された「MR1000=ON」は変換後、常時ON(CR2002)を実行条件とするOUT命令に変換されます。そのため一度このスクリプトを実行すると、以降MR1000がONし続けることになります。

参 考

保持されているビットデータやワードデータの値をクリアしたいときは、以下の方法があります。

- RES関数でリセットする(ビットデータの場合)
- 代入文で0を代入する(ワードデータの場合)
- メニューの【デバッグ(D)】→【PLCのデバイスをオールクリア(L)】からデバイスを選択してクリアする

■実行するタイミング



■利用方法

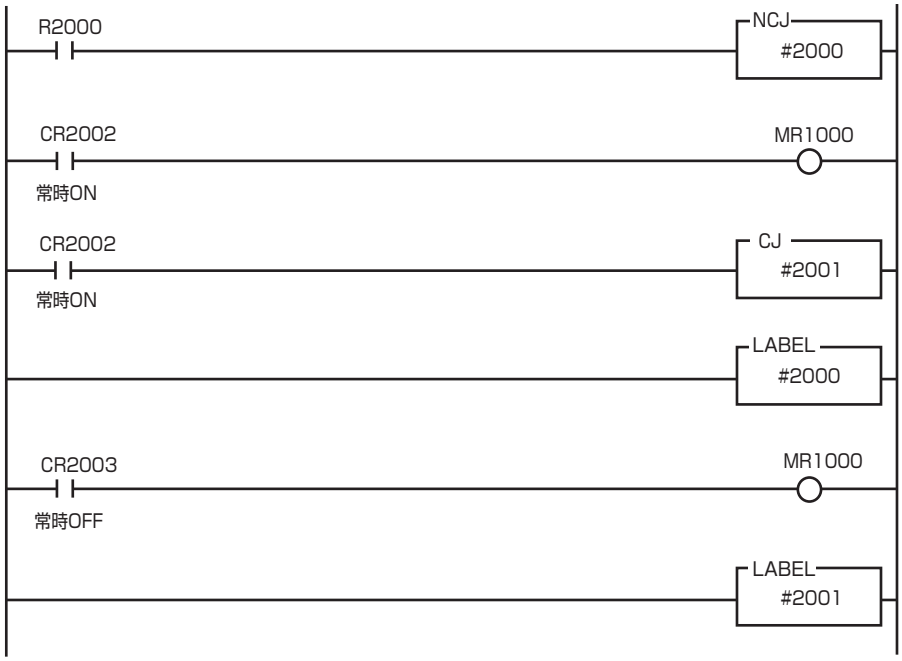
「MR1000=ON」は ON 状態が保持されるため、OFF 状態にするためには「MR1000=OFF」もしくはRES(MR1000)を実行してください。

例）R2000(実行条件)がONのとき、MR1000をONします。R2000がONでない(OFF)のときMR1000をOFFにします。

スクリプト記述

```
IF  R2000 = ON THEN
      MR1000 = ON
ELSE
      MR1000 = OFF
END IF
```

ラダー展開



参 考　ビットデータの状態は、SET/RES関数でも制御できます。

タイマ/カウンタを使う時の注意

タイス/カウンタは現在値、接点状態を同じデバイス番号で扱います。
接点状態は.B(ビット型)として他のビットデータと同じように扱えます(.Bサフィックスは省略可能です)。
現在値は.D(32ビット符号無し)として他のワードデータと同じように扱えますが、必ず.Dサフィックスを指定してください。

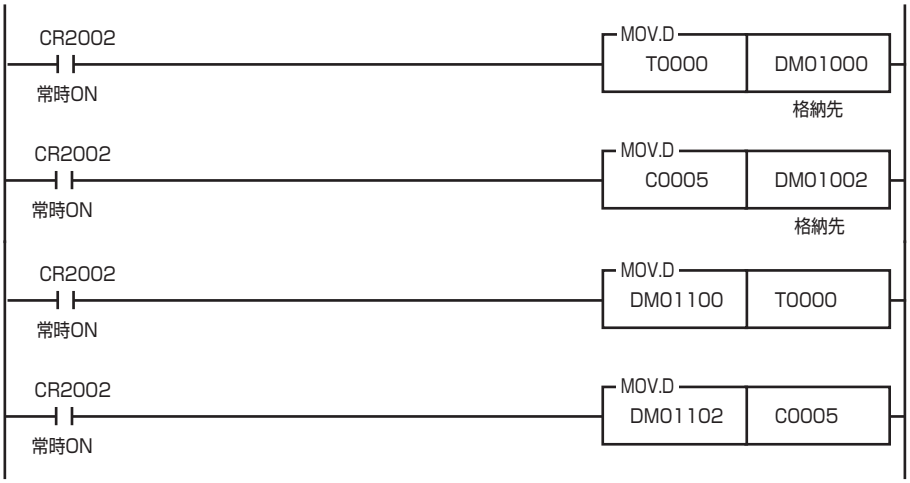
「タイマ(接点)　T」(3-31ページ)
「カウンタ(接点)　C」(3-33ページ)

●タイマ/カウンタ現在値の読み出し

スクリプト記述

```
DM1000.D = T0.D      'タイマT0の現在値をDM1000.Dに格納します。
DM1002.D = C5.D      'カウンタC5の現在値をDM1002.Dに格納します。
T0.D = DM1100.D      'タイマT0の現在値をDM1100.Dに格納されている値に
                     '書き換えます。
C5.D = DM1102.D      'カウンタC5の現在値をDM1102.Dに格納されている値に
                     '書き換えます。
```

ラダー記述



ポイント

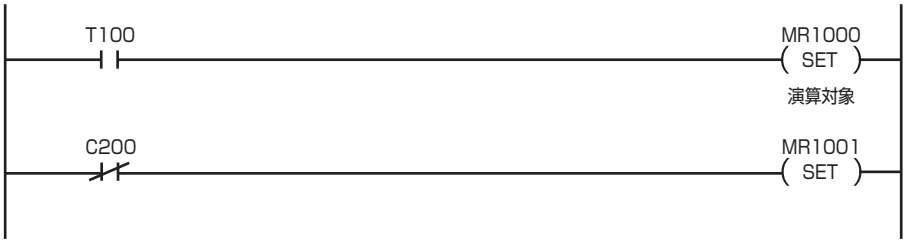
- **タイマ/カウンタの現在値は32ビットで扱います。**
- **タイマ/カウンタを現在値として扱うとき、.Dサフィックスを指定するか、型宣言をおこなってください。**
- **対応機種(使用するCPUユニット)がKV-1000の場合、タイマとカウンタで同じデバイス番号は使用できません。**
  例)T100(タイマ)をすでに使用している場合、C100(カウンタ)は使用できません

●タイマ/カウンタ接点

スクリプト記述

```
SET(T100.B, MR1000)          'タイマ接点T100がONのときMR1000を
                             'ON（保持）します。
SET(C200.B = OFF, MR1001)    'カウンタ接点C200がOFFのときMR1001を
                             'ON（保持）します。
```

ラダー記述



参考　タイマ/カウンタのデフォルトの型はビット型です。

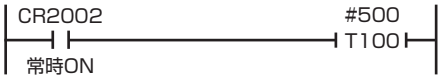
■タイマ関数を使用するときの注意

タイマ関数は、書式に実行条件を持ちますが他の関数と異なり実行条件はタイマのリセット命令をかねています。タイマ関数の実行条件は、省略して記述することができますが、その場合リセットすることができず、常時実行型関数として動作し続けます。

スクリプト記述

```
TMR(T100,500)                ' 実行条件を省略した記述
```

ラダー展開



動作説明　タイマデバイスT100はプログラム起動後5秒経過したら、T100接点をONし続けます。
（RES(T100)を実行するか、PLCの電源OFFもしくは「RUN⇒PROG」切り換え時にしか、接点はクリアできません。）

■ボックススクリプトやIF文中にタイマ関数を使用するときの注意

ボックススクリプトやIF文はNCJ命令～LABELにラダー変換されます。
ボックススクリプトやIF文の実行条件が成り立たないとき、NCJ～LABEL間はジャンプされるため、タイマ関数の使用には注意が必要です。

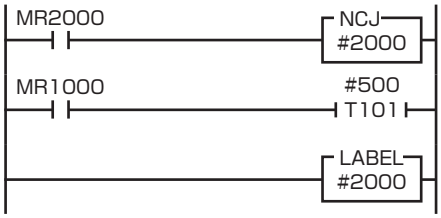
●エリアスクリプトのIF文の中でタイマ関数を使用する

エリアスクリプト使用時、タイマ関数をIF文の中で使用する場合は、以下の注意が必要です。

スクリプト記述

```
IF MR2000 = ON THEN                    ' IF文の中に記述
    TMR (MR1000 = ON, T101, 500)
END IF
```

ラダー展開

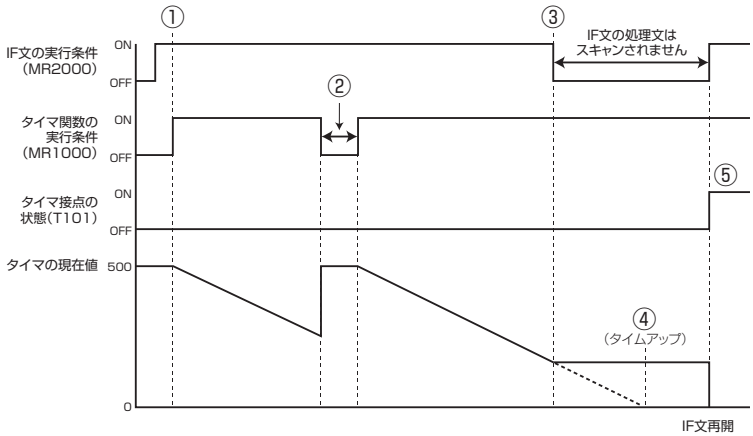


注意事項

IF文の実行条件が成り立たない場合、NCJ命令により処理文はジャンプされます。このとき、各接点の前状態（NCJでジャンプする前の状態）が保持されるため、タイマ関数の実行条件が前状態でONのとき、タイマはリセットされません。

一度動作を開始したタイマは、リセットされるまで動作し続けますが、処理文はNCJ命令でジャンプしているので、タイムアップしタイマ接点がPLC内部でONになっても、出力することはありません。この状態のときにIF文の実行条件が成り立つと、処理文が実行されますので、タイマ関数の動作を一切伴わずにタイマ接点はONが出力されます。

動作説明



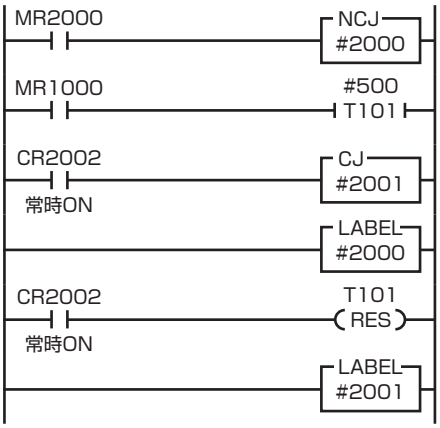
①IF文の実行条件（MR2000）がONして、かつタイマの実行条件（MR1000）がONしている間、タイマT101が動作します。動作開始後5秒経過したら、タイマ接点T101をONする設定です。
②IF文の実行条件（MR2000）がON、かつタイマの実行条件（MR1000）がOFFになると、タイマはリセットされます。
③いったん、タイマが動作を開始しタイムアップする前に、タイマの実行条件ONにもかかわらず、IF文の実行条件がOFFになったとき、IF文の処理文（タイマ関数）はスキャンされません。このときこのときタイマ関数は見かけ上一時停止しているように見えますが、実際は動作を続けます（リセット入力が無いため）。
④ただしタイムアップしても処理文はスキャンされないのでタイマ接点は出力せず、待機状態のようになります。
⑤次のIF文の実行条件が成立したスキャンで、タイマ接点はタイマ動作なしにONします。

回避方法　IF 文の実行条件が OFF の場合、処理文に含まれるタイマ関数をリセットすることで、この問題は回避できます。
スクリプトの記述を以下のように変更します。※1

スクリプト記述

```
IF MR2000 = ON THEN
   TMR (MR1000 = ON, T101, 500)      ' IF文中に記述
ELSE
   RES (T101)                        ' タイマをリセット
END IF
```

ラダー展開



動作説明　IF文の実行条件（MR2000）がONして、かつタイマの実行条件（MR1000）がONしている間、タイマT101が動作します。
動作開始後5秒経過したら、タイマ接点T101をONします。

IF文の実行条件（MR2000）がOFFになると、タイマ（T101）をリセットします。

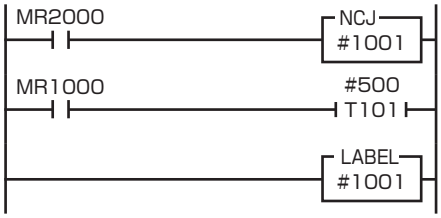
※1　タイマ関数を含んだエリアスクリプトを、CJ～LABEL命令間に記述したり、サブルーチン内に記述すると、タイマ関数が正常に動作しなくなる恐れがあり、上記回避方法でも回避できません。
詳細は、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』を参照してください。

● ボックススクリプトでタイマ関数を指定する

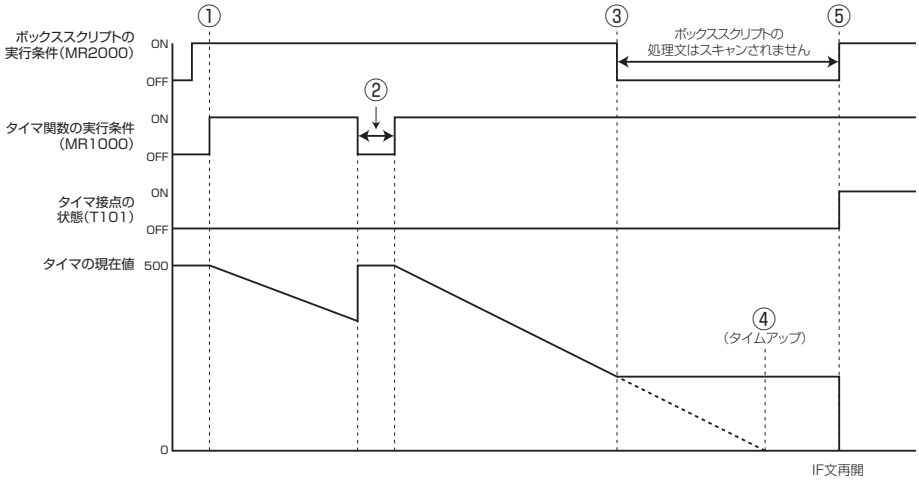
ボックススクリプトで使用するとき、タイマ関数はIF文中で使用する場合と同様の注意が必要です。

スクリプト記述

ラダー展開



動作説明



①ボックススクリプトの実行条件(MR2000)がONして、かつタイマの実行条件(MR1000)がONしている間、タイマT101が動作します。動作開始後5秒経過したら、タイマ接点T101をONする設定です。

②ボックススクリプトの実行条件(MR2000)がON、かつタイマの実行条件(MR1000)がOFFになるとタイマはリセットされます。

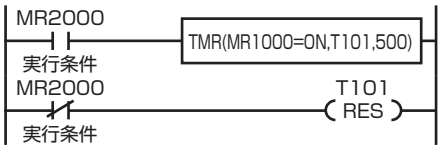
③いったん、タイマが動作を開始しタイムアップする前に、タイマの実行条件ONにもかかわらず、ボックススクリプトの実行条件がOFFになったとき、ボックススクリプトの処理文(タイマ関数)はスキャンされません。このときタイマ関数は見かけ上一時停止しているように見えますが、実際は動作を続けます(リセット入力が無いため)。

④ただし設定時間に到達しても処理文はスキャンされないのでタイマ接点は出力せず、待機状態のようになります。

⑤次のボックススクリプトの実行条件が成立したスキャンで、タイマ接点はタイマ動作無しにONします。

回避方法　ボックススクリプトの実行条件がOFFの場合、ボックススクリプトに含まれるタイマ関数をリセットすることで、この問題は回避できます。
スクリプトの記述を以下のように変更します。※1

スクリプト記述



動作説明　ボックススクリプトの実行条件のB接点でタイマをリセットします。

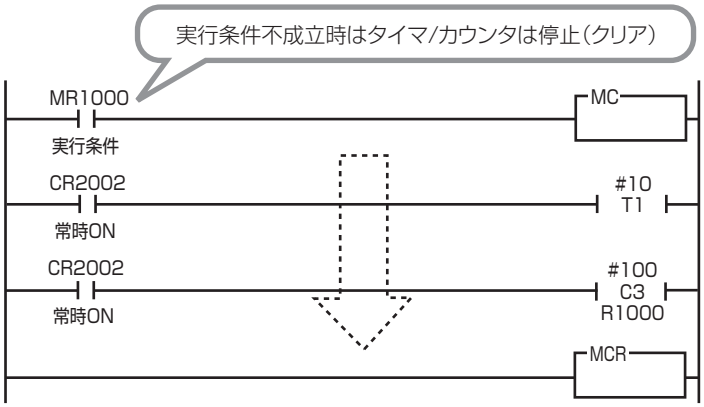
※1　タイマ関数を含んだボックススクリプトを、CJ～LABEL命令間に記述したり、サブルーチン内に記述すると、タイマ関数が正常に動作しなくなる恐れがあり、上記回避方法でも回避できません。
詳細は、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』を参照してください。

●MC～MCRにタイマ/カウンタ関数を使用する場合

スクリプト記述

```
MC MR1000 THEN          'MR1000がONのときタイマ/カウンタ起動
    TMR(T1 , 10)
    CNT(C3 , 100 , R1000)
MCR
```

ラダー記述



条件不成立時(MR1000がOFF)のときにOFFで実行されるため、タイマ/カウンタはリセットされます。

詳細は 「MC ～ MCR文(マスターコントロール)」(4-22ページ)を参照してください。

MEMO

5

基本関数

接点関数 5-2
ビット接点関数 5-8
出力関数 5-12
ビット出力関数 5-14
タイマ･カウンタ関数 5-22
ダイレクト入出力関数 5-36

LDP 指定ビットの立ち上がり微分

戻り値※1 = **LDP** (立ち上がり検出対象)
ロードパルス

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B※3※4	.T	#/$		
S	立ち上がり検出対象※2	立ち上がりを検出する対象を指定します。	–	–	–	–	–	–	.B	–	–	○	○
R	戻り値	動作させる対象を指定します。	–	–	–	–	–	–	.B	–	–	○	–

※1 戻り値は省略できます。省略した場合、演算結果はワークデバイス/ワークリレーに格納されます。
※2 立ち上がり検出対象にワードデータを指定した場合、最下位ビットの状態を検出します。
※3 対応機種がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合、ダイレクト入出力リレー(DR)が指定できます。
※4 対応機種がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合、ワードデータのビット位置を指定できます。

動作説明

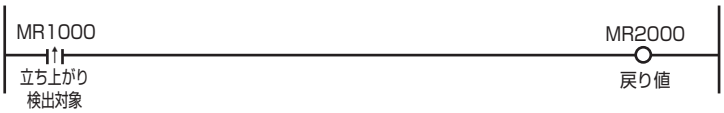
S の立ち上がり(OFFからONの変化)を検出すると、R を1スキャンだけONします。

「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「接点命令」を参照してください。

●書式例

MR2000 = **LDP** (MR1000)



ポイント **ボックススクリプト内やIF文内で使用するときは注意が必要です。**
「ボックススクリプトを使用するときの注意」(2-13ページ)

サンプルプログラム

●演算式を使用する場合

スクリプト記述　MR2000 = **LDP** (DM1000 + DM1001 = 1000)

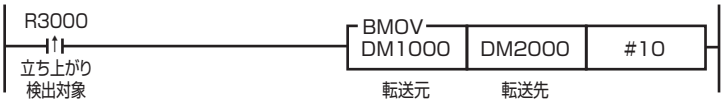
動作説明　DM1000とDM1001の合計が1000以外の値から1000と等しくなった(演算結果がFALSEからTRUEに変化した)ときの1スキャンのみ、MR2000をONします。

●他の関数に組み込んで使用する場合

スクリプト記述　BMOV (**LDP** (R3000), DM1000, DM2000, 10)

動作説明　R3000の立ち上がり(OFFからON)を検出したスキャンのみ、DM1000を先頭とする10ワード分の内容を、DM2000を先頭とする10ワードにブロック転送します。

ラダー変換



参考　組み込む関数の引数によっては、使用できない場合がありますので注意してください。
「BMOV関数」(7-2ページ)
KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

LDF 指定ビットの立ち下がり微分

戻り値※1 = **LDF** (立ち下がり検出対象)
ロードパルフ

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B※3※4	.T	#/$		
S	立ち下がり検出対象※2	立ち下がりを検出する対象を指定します。	–	–	–	–	–	–	.B	–	–	○	○
R	戻り値	動作させる対象を指定します。	–	–	–	–	–	–	.B	–	–	○	–

※1 戻り値は省略できます。省略した場合、演算結果はワークデバイス/ワークリレーに格納されます。
※2 立ち下がり検出対象にワードデータを指定した場合、最下位ビットの状態を検出します。
※3 対応機種がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合、ダイレクト入出力リレー(DR)が指定できます。
※4 対応機種がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合、ワードデータのビット位置を指定できます。

動作説明

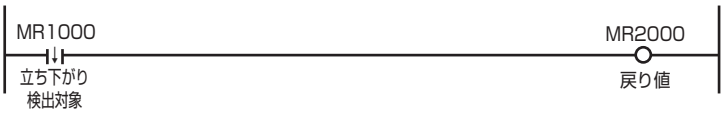
S の立ち下がり(ONからOFFの変化)を検出すると、R を1スキャンだけONします。

「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「接点命令」を参照してください。

●書式例

MR2000 = **LDF** (MR1000)



ポイント **ボックススクリプト内やIF文内で使用するときは注意が必要です。**
「ボックススクリプトを使用するときの注意」(2-13ページ)

サンプルプログラム

●演算式を使用する場合

スクリプト記述　MR2000 = **LDF** (DM1000 + DM1001 = 1000)

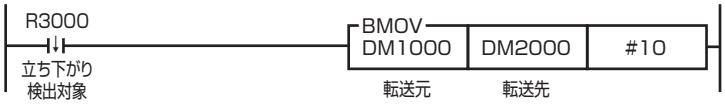
動作説明　DM1000とDM1001の合計が1000から1000以外の値(演算結果がTRUEからFALSE)に変化した1スキャンのみ、MR2000をONします。

●他の関数に組み込んで使用する場合

スクリプト記述　BMOV (**LDF** (R3000), DM1000, DM2000, 10)

動作説明　R3000の立ち下がり(ONからOFFの変化)を検出した1スキャンのみ、DM1000を先頭とする10ワード分の内容を、DM2000を先頭とする10ワードにブロック転送します。

ラダー変換



参考　組み込む関数の引数によっては、使用できない場合がありますので注意してください。
「BMOV関数」(7-2ページ)
KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

LDPB 指定ビットの立ち上がり微分の反転

戻り値※1 = **LDPB** (立ち上がり検出対象)
ロードパルスバー

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B※3※4	.T	#/$		
S	立ち上がり検出対象※2	立ち上がりを検出する対象を指定します。	–	–	–	–	–	–	.B	–	–	○	○
R	戻り値	動作させる対象を指定します。	–	–	–	–	–	–	.B	–	–	○	–

※1　戻り値は省略できます。省略した場合、演算結果はワークリレーに格納されます。
※2　立ち上がり検出対象にワードデータを指定した場合、最下位ビットの状態を検出します。
※3　ダイレクト入出力リレー(DR)が指定できます。
※4　ワードデータのビット位置を指定できます。

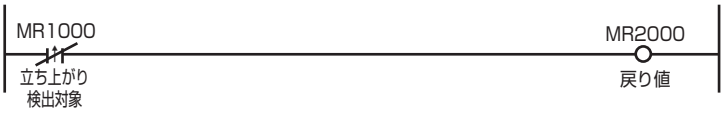
ポイント　**LDPB関数は、KV-1000シリーズでは使用できません。**

動作説明

S の立ち上がり(OFFからONの変化)を検出すると、R を1スキャンだけOFFします。
「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「接点命令」を参照してください。

●書式例

MR2000 = **LDPB** (MR1000)



ポイント　**ボックススクリプト内やIF文内で使用するときは注意が必要です。**
「ボックススクリプトを使用するときの注意」(2-13ページ)

LDFB 指定ビットの立ち下がり微分の反転

戻り値※1 = **LDFB** (立ち下がり検出対象)
ロードパルフバー

引数/戻り値		説明	認識する型							定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.B※3※4	.T	#/$		
S	立ち下がり検出対象※2	立ち下がりを検出する対象を指定します。	—	—	—	—	—	.B	—	—	○	○
R	戻り値	動作させる対象を指定します。	—	—	—	—	—	.B	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果はワークリレーに格納されます。
※2　立ち下がり検出対象にワードデータを指定した場合、最下位ビットの状態を検出します。
※3　ダイレクト入出力リレー(DR)が指定できます。
※4　ワードデータのビット位置を指定できます。

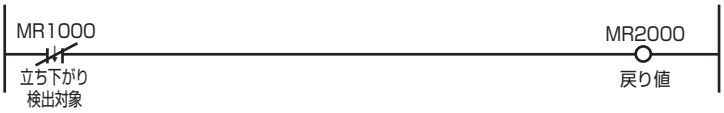
ポイント　**LDFB関数は、KV-1000シリーズでは使用できません。**

動作説明

S の立ち下がり(ONからOFFの変化)を検出すると、R を1スキャンだけOFFします。
「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「接点命令」を参照してください。

●書式例

MR2000 = **LDFB** (MR1000)



ポイント　**ボックススクリプト内やIF文内で使用するときは注意が必要です。**
「ボックススクリプトを使用するときの注意」(2-13ページ)

BLD　ワードデータの指定ビット情報

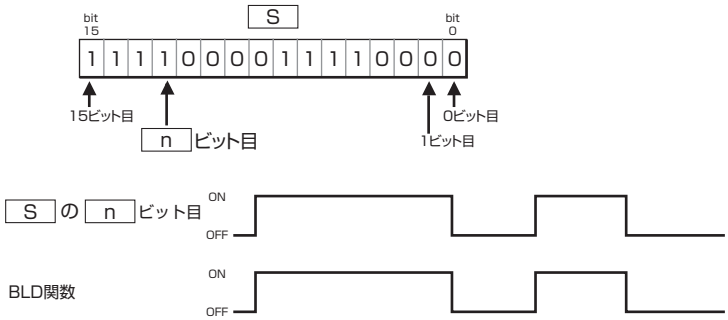
戻り値※1 = **BLD** (読み出し対象, ビット位置)
ビットロード

引数/戻り値	説明	対応サフィックス								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B※3※4	.T	#/$		
S 読み出し対象※2	ビットの状態を読み出す対象を指定します。	.U	.U	.D	.D	—	—	—	—	—	○	○
n ビット位置	対象のON/OFFを判定するビット位置を指定します。(16ビットバイナリデータ)	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	動作させる対象を指定します。	—	—	—	—	—	—	.B	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果はワークデバイス/ワークリレーに格納されます。
※2　S にビットデータを指定した場合、指定ビット＋ n の位置のビットが指定されます。チャンネルをまたぐ場合も正常に動作します。
※3　ビット位置の値は、読み出し対象のサフィックスが.Uのとき0～15、.Dのときは0～31を指定します。
※4　対応機種がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合、ダイレクト入出力リレー(DR)が指定できます。
※5　対応機種がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合、ワードデータのビット位置を指定できます。

動作説明

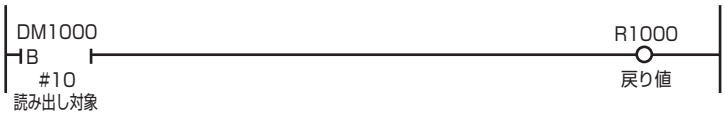
S の n で指定したビット位置のON/OFF状態を、R に伝えます。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ビット接点命令」を参照してください。

●書式例

R1000 = **BLD** (DM1000, 10)



ポイント　**n に定数以外を記述するとき、演算結果が S のサフィックスに従った範囲を(.Uのときは0～15、.Dのときは0～31)超えた場合、次のようになります。**
S のサフィックスが.U: n の下位4ビットの値が使用されます。
S のサフィックスが.D: n の下位5ビットの値が使用されます。

サンプルプログラム

●ビットデータを指定する場合

スクリプト記述　R1000 = **BLD** (MR2009.U, 10)

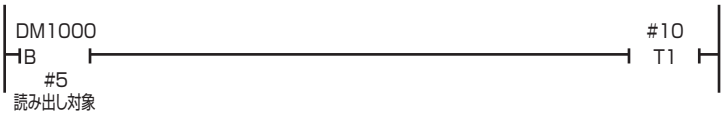
動作説明　MR2009.Uのビット10(MR2103)がONのとき、R1000をONします。

●他の関数に組み込んで使用する場合

スクリプト記述　TMR (**BLD** (DM1000, 5), T001, 10)

動作説明　DM1000のビット5がONのとき、タイマ(T1)を起動します。

ラダー変換



参考　組み込む関数の引数によっては、使用できない場合がありますので注意してください。
「TMR関数」(5-22ページ)
KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

BLDB　ワードデータの指定ビット反転情報

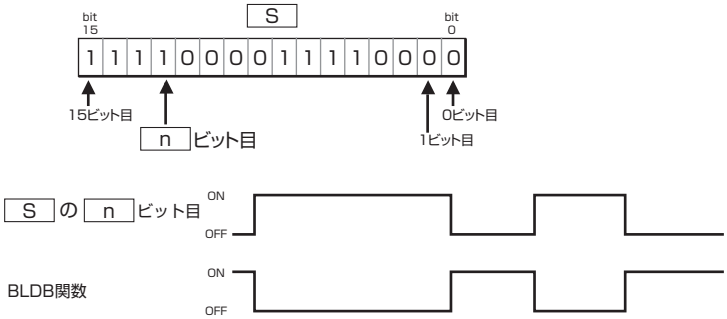
戻り値※1 = **BLDB** (読み出し対象, ビット位置)

引数/戻り値	説明	対応サフィックス								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B※3※4	.T	#/$		
S 読み出し対象※2	ビットの状態を読み出す対象を指定します。	.U	.U	.D	.D	—	—	—	—	—	○	○
n ビット位置	対象のON/OFFを判定するビット位置を指定します。(16ビットバイナリデータ)	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	動作させる対象を指定します。	—	—	—	—	—	—	.B	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果はワークデバイス/ワークリレーに格納されます。
※2　S にビットデータを指定した場合、指定ビット＋ n の位置のビットが指定されます。チャンネルをまたぐ場合も正常に動作します。
※3　ビット位置の値は、読み出し対象のサフィックスが.Uのとき0～15、.Dのときは0～31を指定します。
※4　対応機種がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合、ダイレクト入出力リレー(DR)が指定できます。
※5　対応機種がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合、ワードデータのビット位置を指定できます。

動作説明

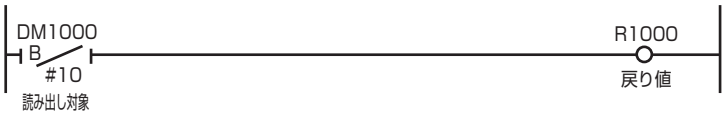
S の n で指定したビット位置のON/OFF状態を、反転して R に伝えます。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ビット接点命令」を参照してください。

●書式例

R1000 = **BLDB** (DM1000, 10)



ポイント　**n に定数以外を記述するとき、演算結果が S のサフィックスに従った範囲を(.Uのときは0～15、.Dのときは0～31)超えた場合、次のようになります。**
S のサフィックスが.U: n の下位4ビットの値が使用されます。
S のサフィックスが.D: n の下位5ビットの値が使用されます。

サンプルプログラム

●読み出し対象にビットデータを指定する場合

スクリプト記述　R1000 = **BLDB** (MR2009.U, 10)

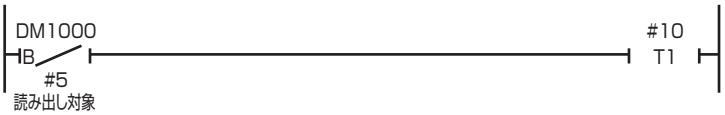
動作説明　MR2009.Uのビット10(MR2103)がOFFのとき、R1000をONします。

●他の関数に組み込んで使用する場合

スクリプト記述　TMR (**BLDB** (DM1000, 5), T1, 10)

動作説明　DM1000のビット5がOFFのとき、タイマ(T1)を起動します。

ラダー変換



参考　組み込む関数の引数によっては、使用できない場合がありますので注意してください。
「TMR関数」(5-22ページ)
KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

SET 指定ビットの出力ON

セット
SET ([実行条件,]※1 対象)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B※2※3	.T	#/$		
D 対象	ONして保持する対象を指定します。	—	—	—	—	—	—	.B	—	—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 対応機種がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合、ダイレクト入出力リレー(DR)が指定できます。
※3 対応機種がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合、ワードデータのビット位置を指定できます。

動作説明

実行条件が成り立つ(TRUE)とき、D をONします。一度ONすると、実行条件が不成立になってもONのまま保持されます。(ONした対象はRES関数でOFFします)
「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「出力命令」を参照してください。
「RES関数」(5-13ページ)

●書式例

SET (R3000, MR1000)

サンプルプログラム

●対象にワードデータを指定する場合

スクリプト記述 **SET** (MR1000, DM2000.B)

動作説明 MR1000がONのとき、DM2000の最下位ビットをONにして保持します。

参考 KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する 手順5」(2-10ページ)を参照してください。

RES 指定ビットを出力OFF

リセット
RES ([実行条件,]※1 対象)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B※2※3	.T	#/$		
D 対象	OFFする対象を指定します。	—	—	—	—	—	—	.B	—	—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 対応機種がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合、ダイレクト入出力リレー(DR)が指定できます。
※3 対応機種がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合、ワードデータのビット位置を指定できます。

動作説明

実行条件が成り立つ(TRUE)とき、D をOFFします。(SET関数でONした対象はRES関数でOFFします)

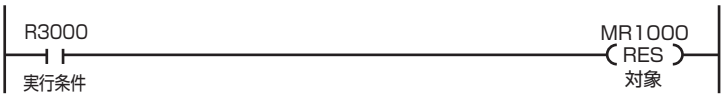
「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「出力命令」を参照してください。

「SET関数」(5-12ページ)

●書式例

RES (R3000, MR1000)



サンプルプログラム

●対象にワードデータを指定する場合

スクリプト記述 **RES** (MR1001, DM2000.B)

動作説明 MR1001がONのとき、DM2000の最下位ビットをOFFにします。

●タイマをリセットする場合

スクリプト記述 **RES** (MR1002, T100)

動作説明 MR1002がONのとき、T100の現在値を設定値と同じ値に変更します。

参考 KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する 手順5」(2-10ページ)を参照してください。

BOUT　ワードデータの指定ビット出力

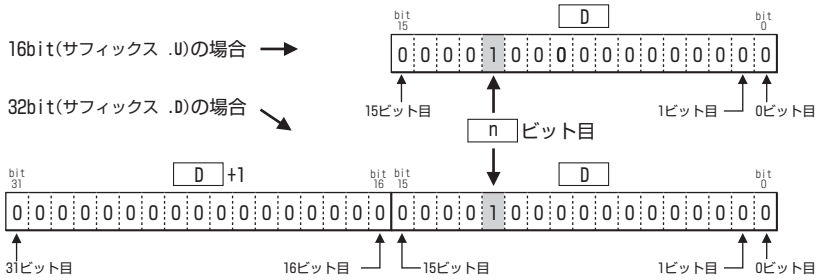
ビットアウト
BOUT ([実行条件,]※1 対象, ビット位置)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D 対象※2	入力状態を出力する対象を指定します。	.U	.U	.D	.D	—	—	—	—	—	○	—
n ビット位置※3	出力するビット位置を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1　[　]は省略可能です。(実行条件を省略した場合、CR2002(常時ON)が指定されます)
※2　D にビットデータを指定した場合、対象＋ n の値で計算される対象のリレーに出力されます。チャンネルをまたぐ場合も正常に動作します。
※3　ビット位置の値は、対象のサフィックスが.Uのとき0～15、.Dのときは0～31を指定します。

動作説明

実行条件が成り立つ(TRUE)とき、D の指定された n のビットをONし、実行条件が成り立たない(FALSE)のとき、OFFにします。

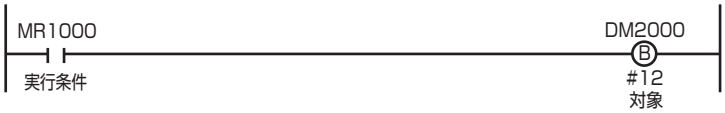


「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ビット出力命令」を参照してください。

●書式例

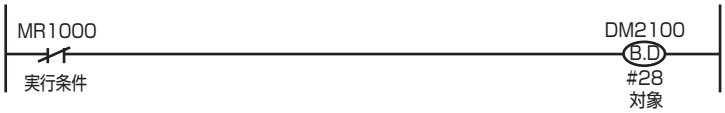
- D :対象が16ビットの場合

BOUT (MR1000, DM2000, 12)



- D :対象が32ビットの場合

BOUT (MR1000 = OFF, DM2100.D, 28)



ポイント　**n に定数以外を記述するとき、演算結果が D のサフィックスに従った範囲を(.Uのときは0～15、.Dのときは0～31)超えた場合、次のようになります。**
D のサフィックスが.U: n の下位4ビットの値が使用されます。
D のサフィックスが.D: n の下位5ビットの値が使用されます。

サンプルプログラム

●対象にビットデータを指定する場合

スクリプト記述 **BOUT** (DM1000 <= 100, MR2002.U, 14)

動作説明 DM1000が100以下のとき、MR2002.Uのビット14(MR2100)をONし、DM1000が100より大きいときOFFします。

参考 KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

BOUB　ワードデータの指定ビット反転出力

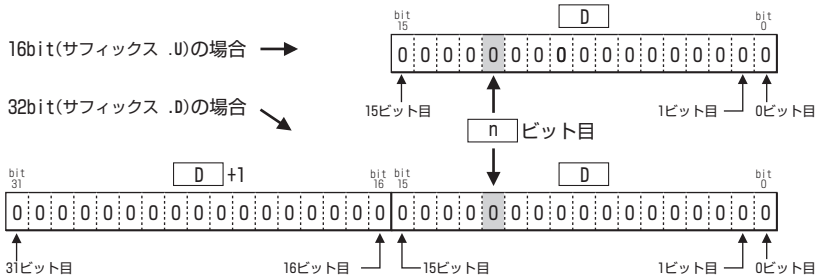
ビットアウトバー
BOUB ([実行条件,]※1 対象, ビット位置)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D 対象※2	入力状態を反転出力する対象を指定します。	.U	.U	.D	.D	—	—	—	—	—	○	—
n ビット位置※3	出力するビット位置を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1　[　]は省略可能です。(実行条件を省略した場合、CR2002(常時ON)が指定されます)
※2　D にビットデータを指定した場合、指定ビット＋ n の位置のビットに出力されます。チャンネルをまたぐ場合も正常に動作します。
※3　ビット位置の値は、対象のサフィックスが.Uのとき0～15、.Dのときは0～31を指定します。

動作説明

実行条件が成り立つ(TRUE)とき、D の指定された n のビットをOFFにし、実行条件が成り立たない(FALSE)とき、ONにします。

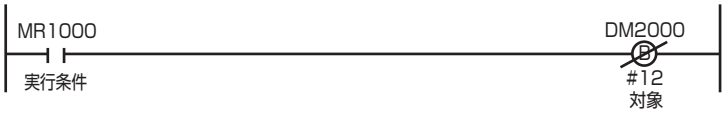


「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ビット出力命令」を参照してください。

●書式例

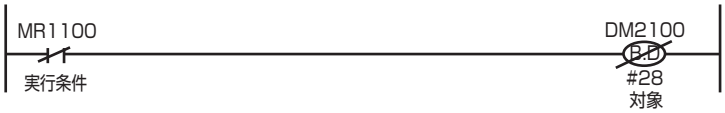
- D :対象が16ビットの場合

BOUB (MR1000, DM2000, 12)



- D :対象が32ビットの場合

BOUB (MR1100 = OFF, DM2100.D, 28)



ポイント　n に定数以外を記述するとき、演算結果が D のサフィックスに従った範囲を(.Uのときは0～15、.Dのときは0～31)超えた場合、次のようになります。
D のサフィックスが.U: n の下位4ビットの値が使用されます。
D のサフィックスが.D: n の下位5ビットの値が使用されます。

サンプルプログラム

●対象にビットデータを指定する場合

スクリプト記述 **BOUB** (DM1000 <= 100, MR2002.U, 14)

動作説明 DM1000が100以下のとき、MR2002.Uのビット14(MR2100)をOFFにし、DM1000が100より大きいときONにします。

参考 KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

BSET ワードデータの指定ビット出力ON

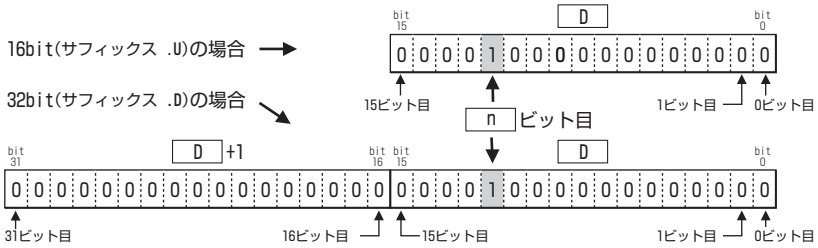
ビットセット
BSET ([実行条件,]※1 対象, ビット位置)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D 対象※2	ONして保持する対象を指定します。	.U	.U	.D	.D	—	—	—	—	—	○	—
n ビット位置※3	ONして保持するビット位置を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 D にビットデータを指定した場合、指定ビット＋ n の位置のビットに出力されます。チャンネルをまたぐ場合も正常に動作します。
※3 ビット位置の値は、対象のサフィックスが.Uのとき0～15、.Dのときは0～31を指定します。

動作説明

実行条件が成り立つ(TRUE)とき、D の指定された n のビットをONします。一度ONすると、実行条件が不成立になってもONのまま保持されます。
(ONした対象はBRES関数でOFFできます)

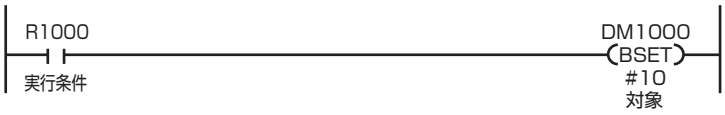


「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ビット出力命令」を参照してください。
「BRES関数」(5-20ページ)

●書式例

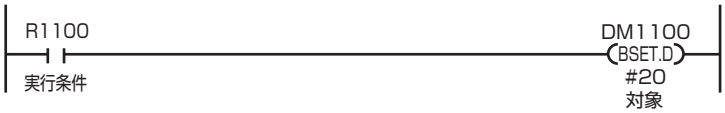
- **D :対象が16ビットの場合**

BSET (R1000, DM1000, 10)



- **D :対象が32ビットの場合**

BSET (R1100 = OFF, DM1100.D, 20)



ポイント

n に定数以外を記述するとき、演算結果が D のサフィックスに従った範囲を(.Uのときは0～15、.Dのときは0～31)超えた場合、次のようになります。
D のサフィックスが.U：n の下位4ビットの値が使用されます。
D のサフィックスが.D：n の下位5ビットの値が使用されます。

サンプルプログラム

●対象にビットデータを指定する場合

スクリプト記述　**BSET** (DM1000 = 100, MR2008.U, 10)

動作説明　DM1000が100のとき、MR2008.Uのビット10(MR2102)をONして保持します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

BRES ワードデータの指定ビット出力OFF

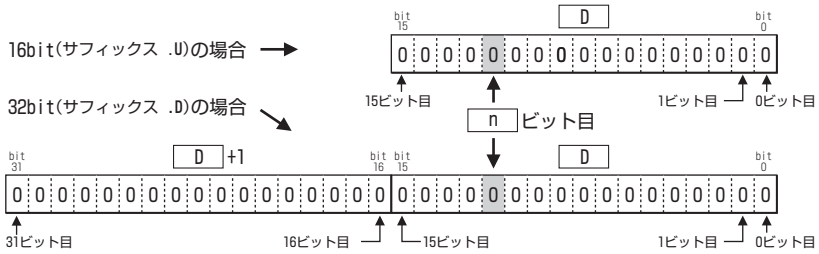
ビットリセット
BRES ([実行条件,]※1 対象, ビット位置)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D 対象※2	OFFする対象を指定します。	.U	.U	.D	.D	—	—	—	—	—	○	—
n ビット位置※3	OFFするビット位置を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 D にビットデータを指定した場合、指定ビット＋ n の位置のビットに出力されます。チャンネルをまたぐ場合も正常に動作します。
※3 ビット位置の値は、対象のサフィックスが.Uのとき0～15、.Dのときは0～31を指定します。

動作説明

実行条件が成り立つ(TRUE)とき、D の指定された n のビットをOFFします。

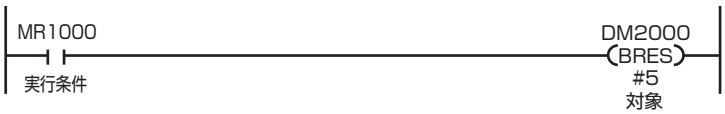


「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ビット出力命令」を参照してください。
「BSET関数」(5-18ページ)

●書式例

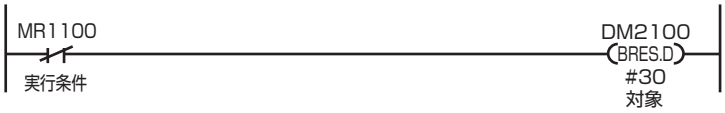
- D :対象が16ビットの場合

BRES (MR1000, DM2000, 5)



- D :対象が32ビットの場合

BRES (MR1100 = OFF, DM2100.D, 30)



ポイント
n に定数以外を記述するとき、演算結果が D のサフィックスに従った範囲を(.Uのときは0～15、.Dのときは0～31)超えた場合、次のようになります。
D のサフィックスが.U: n の下位4ビットの値が使用されます。
D のサフィックスが.D: n の下位5ビットの値が使用されます。

サンプルプログラム

●対象にビットデータを指定する場合

スクリプト記述 **BRES** (DM1000 >= 100, MR2008.U, 10)

動作説明 DM1000が100以上のとき、MR2008.Uのビット10(MR2102)をOFFします。

参考 KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

TMR 100ms32ビット減算式タイマ

TMR ([タイマ実行条件,]※1 タイマ, タイマ設定値)

引数/戻り値	説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
D タイマ	タイマを指定します。(T0～T3999※3)	—	—	—	—	—	—	—	—	—	—	—
n タイマ設定値※2	タイマの設定値を100ms単位で指定します。(0～4294967295)	.D	.D	.D	.D	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

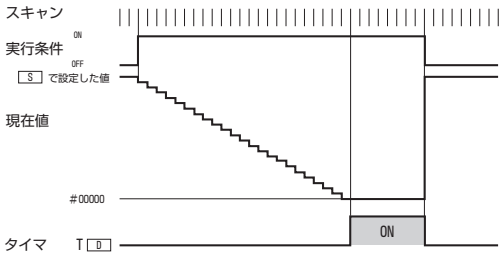
※1 • [ ]は省略可能です。実行条件はリセットを兼ねています。(実行条件がOFFのときTMR関数をリセットします)
• 実行条件を省略した場合、別途RES関数を使ってタイマをリセットしてください。

※2 タイマ設定値にビットデータを指定した場合は連続する32ビットが、ワードデータを指定した場合は連続する2ワードが占有されます。

※3 KV Nanoシリーズの場合、指定できるデバイス番号はT0～T511です。

動作説明

実行条件が成り立つ(TRUE)とき、D で指定したタイマを起動します。S で指定した値が現在値となり、100msおきに減算(-1)されます。現在値が0になると指定したタイマ(接点)がONします。実行条件が不成立となったとき、指定したタイマをOFFして現在値をリセットします。

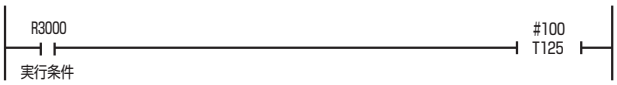


「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「タイマ･カウンタ命令」を参照してください。

●書式例

TMR (R3000, T125, 100)



参考 TMRの誤差は、「±(100ms+1スキャンタイム)以内」となります。

ポイント

- 対応機種がKV-8000/7500/7300/5500/5000/3000、KV　Nanoシリーズの場合、他のTMR、TMH、TMS、TMU関数(命令)、UDT命令と同じデバイス番号は使用できません。
- 対応機種がKV-1000の場合、他のTMR、TMH、TMS、CNT 関数(命令)、OUTC、UDC、UDT命令と同じデバイス番号は使用できません。
- 設定値には演算式を指定できます。演算式が扱える範囲は－2147483648～+2147483647の範囲になりますので以下の場合に注意が必要です。
- 演算式の結果の値が負の場合、演算式の結果に4294967295を加算した値が設定値としてセットされます。
- ボックススクリプトで使用する場合や、サブルーチン内に記述する場合は注意が必要です。

「タイマ/カウンタを使う時の注意」(4-40ページ)

タイマ現在値の読み出しと変更

タイマは、サフィックスを.D(32ビット符号無し)に指定することで現在値を扱います。

●現在値の読み出し

代入文を用い、タイマを右辺に、読み出したタイマの現在値の格納先を左辺に指定します。

スクリプト記述　DM1000.D = T125.D

動作説明　T125.Dの現在値を、DM1000.Dに格納します。

●現在値の書き換え

代入文を用い、書き換える値の格納先を右辺に、タイマを左辺に指定します。

スクリプト記述　T125.D = DM1100.D

動作説明　T125.Dの現在値を、DM1100.Dに格納されている値に書き換えます。

ポイント

- 現在値書き換え時に「現在値＞設定値」となる場合、現在値は設定値と同じ値に変更されます。
- TMR 関数のカウンタ設定値及び現在値は 32 ビット符号無しデータを使用するので、現在値や設定値の格納先や変更するデータは、必ず32ビット符号無し(.D)を使用してください。
- タイマの設定値を変更する場合や設定値を読み出したい場合は、設定値に対象 (32 ビット符号無し)を指定してください。

『KV-8000/7000/5000/3000/1000 シリーズ・KV Nano シリーズ命令語リファレンスマニュアル』「タイマ・カウンタ命令」

タイマ接点の読み出し

タイマのサフィックスを.B(ビット型)に指定することで、接点を扱います。(.Bは省略できます)

スクリプト記述　MR1000 = T125.B

動作説明　T125.Bの接点状態をMR1000に代入します。

ポイント

タイマ接点に他のビット情報を代入できません。
タイマを強制的にリセットするときは、タイマ接点をRES関数でリセットしてください。
タイマのサフィックスを省略すると、タイマはビット型として扱われます。

参考

KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

MEMO

TMH 10ms32ビット減算式タイマ

高速タイマ
TMH ([タイマ実行条件,]※1 タイマ, タイマ設定値)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D タイマ	タイマを指定します。(T0～T3999※3)	—	—	—	—	—	—	—	—	—	—	—
n タイマ設定値※2	タイマの設定値を10ms単位で指定します。(0～4294967295)	.D	.D	.D	.D	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

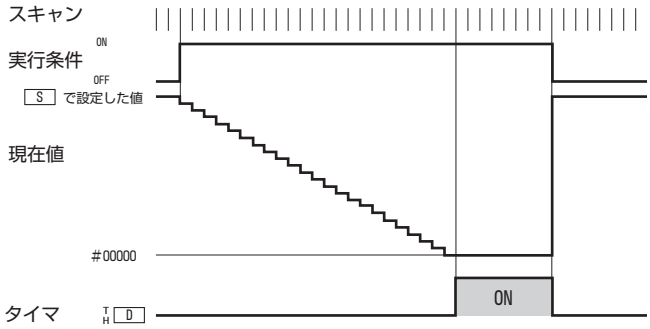
※1 • [ ]は省略可能です。実行条件はリセットを兼ねています。(実行条件がOFFのときTMH関数をリセットします)
• 実行条件を省略した場合、別途RES関数を使ってタイマをリセットしてください。

※2 タイマ設定値にビットデータを指定した場合は連続する32ビットが、ワードデータを指定した場合は連続する2ワードが占有されます。

※3 KV Nanoシリーズの場合、指定できるデバイス番号はT0～T511です。

動作説明

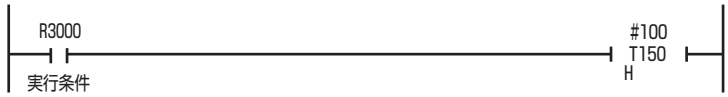
実行条件が成り立つ(TRUE)とき、n で指定したタイマを起動します。S で指定した値が現在値となり、10msおきに減算(-1)されます。現在値が0になると指定したタイマ(接点)がONします。実行条件が不成立となったとき、指定したタイマをOFFして現在値をリセットします。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「タイマ･カウンタ命令」を参照してください。

●書式例

TMH (R3000, T150, 100)



参考 TMHの誤差は、「±(10ms+1スキャンタイム)以内」となります。

ポイント

- **対応機種がKV-8000/7500/7300/5500/5000/3000、KV　Nanoシリーズの場合、他のTMR、TMH、TMS、TMU関数(命令)、UDT命令と同じデバイス番号は使用できません。**
- **対応機種がKV-1000の場合、他のTMR、TMH、TMS、CNT 関数(命令)、OUTC、UDC、UDT命令と同じデバイス番号は使用できません。**
- **設定値には演算式を指定できます。演算式が扱える範囲は－2147483648～+2147483647の範囲になりますので以下の場合に注意が必要です。**
  - **演算式の結果の値が負の場合、演算式の結果に4294967295を加算した値が設定値としてセットされます。**
- **ボックススクリプトで使用する場合や、サブルーチン内に記述する場合は注意が必要です。**
  「タイマ/カウンタを使う時の注意」(4-40ページ)

「TMR関数」(5-22ページ)

サンプルプログラム

●設定値に演算式を使用する場合

スクリプト記述　**TMH** (R1000, T150, DM1000 + 100)

動作説明　実行条件R1000が成り立っている(TRUE)あいだ、高速タイマT150は動作します。実行条件が成り立たなくなった(FALSE)の場合、タイマを停止し現在値をリセットします。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

TMS　1ms32ビット減算式タイマ

高速1msタイマ
TMS ([タイマ実行条件,]※1 タイマ, タイマ設定値)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D タイマ	タイマを指定します。(T0～T3999※3)	—	—	—	—	—	—	—	—	—	—	—
n タイマ設定値※2	タイマの設定値を1ms単位で指定します。(0～4294967295)	.D	.D	.D	.D	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

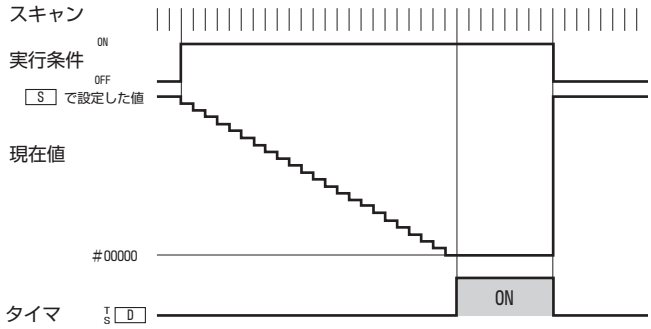
※1 • [ ]は省略可能です。実行条件はリセットを兼ねています。(実行条件がOFFのときTMS関数をリセットします)
• 実行条件を省略した場合、別途RES関数を使ってタイマをリセットしてください。

※2 タイマ設定値にビットデータを指定した場合は連続する32ビットが、ワードデータを指定した場合は連続する2ワードが占有されます。

※3 KV Nanoシリーズの場合、指定できるデバイス番号はT0～T511です。

動作説明

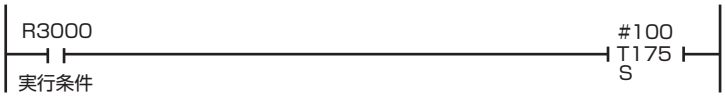
実行条件が成り立つ(TRUE)とき、D で指定したタイマを起動します。S で指定した値が現在値となり、1msおきに減算(-1)されます。現在値が0になると指定したタイマ(接点)がONします。実行条件が不成立となったとき、指定したタイマをOFFして現在値をリセットします。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「タイマ･カウンタ命令」を参照してください。

●書式例

TMS (R3000, T175, 100)



参 考　TMSの誤差は、「±(1ms+1スキャンタイム)以内」となります。

ポイント

- 対応機種がKV-8000/7500/7300/5500/5000/3000、KV　Nanoシリーズの場合、他のTMR、TMH、TMS、TMU関数(命令)、UDT命令と同じデバイス番号は使用できません。
- 対応機種がKV-1000の場合、他のTMR、TMH、TMS、CNT 関数(命令)、OUTC、UDC、UDT命令と同じデバイス番号は使用できません。
- 設定値には演算式を指定できます。演算式が扱える範囲は－2147483648～+2147483647の範囲になりますので以下の場合に注意が必要です。
  - 演算式の結果の値が負の場合、演算式の結果に4294967295を加算した値が設定値としてセットされます。
- ボックススクリプトで使用する場合や、サブルーチン内に記述する場合は注意が必要です。

「タイマ/カウンタを使う時の注意」(4-40ページ)

「TMR関数」(5-22ページ)

TMU　10μs32ビット減算式タイマ

高速10μsタイマ
TMU ([タイマ実行条件,]※1 タイマ, タイマ設定値)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D タイマ	タイマを指定します。(T0～T3999※3)	—	—	—	—	—	—	—	—	—	—	—
n タイマ設定値※2	タイマの設定値を1μs単位で指定します。(0～4294967295)	.D	.D	.D	.D	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 • [ ]は省略可能です。実行条件はリセットを兼ねています。(実行条件がOFFのときTMU関数をリセットします)
• 実行条件を省略した場合、別途RES関数を使ってタイマをリセットしてください。

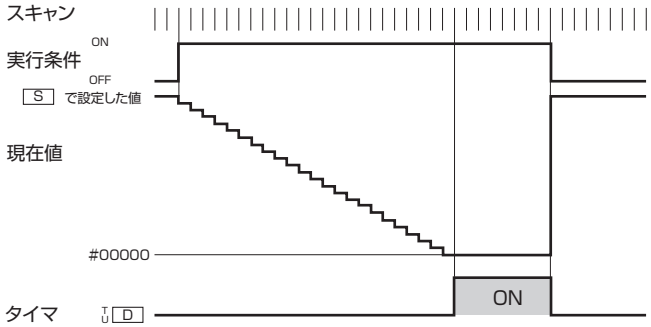
※2 タイマ設定値にビットデータを指定した場合は連続する32ビットが、ワードデータを指定した場合は連続する2ワードが占有されます。

※3 KV Nanoシリーズの場合、指定できるデバイス番号はT0～T511です。

ポイント **TMU関数は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ使用できます。**
「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

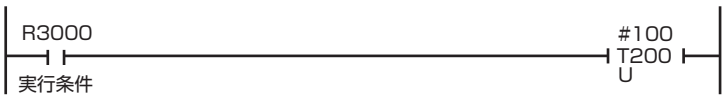
実行条件が成り立つ(TRUE)とき、D で指定したタイマを起動します。S で指定した値が現在値となり、10μsおきに減算(-1)されます。現在値が0になると指定したタイマ(接点)がONします。実行条件が不成立となったとき、指定したタイマをOFFして現在値をリセットします。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「タイマ･カウンタ命令」を参照してください。

●書式例

TMU (R3000, T200, 100)



参考 TMUの誤差は、「±(10μs+1スキャンタイム)以内」となります。

ポイント

- 他のTMR、TMH、TMS、TMU関数(命令)、UDT命令と同じデバイス番号は使用できません。
- 設定値には演算式を指定できます。演算式が扱える範囲は－2147483648～+2147483647の範囲になりますので以下の場合に注意が必要です。
  - 演算式の結果の値が負の場合、演算式の結果に4294967295を加算した値が設定値としてセットされます。
- ボックススクリプトで使用する場合や、サブルーチン内に記述する場合は注意が必要です。

「タイマ/カウンタを使う時の注意」(4-40ページ)

「TMR関数」(5-22ページ)

CNT 32ビット加算式カウンタ

CNT ([カウンタ実行条件,]※1 カウンタ, カウンタ設定値, 入力対象)

引数/戻り値	説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
D カウンタ	カウンタを指定します。(C0～C3999)※3	—	—	—	—	—	—	—	—	—	—	—
n カウンタ設定値※2	カウンタの設定値を指定します。(0～4294967295)	.D	.D	.D	.D	—	—	—	—	○	○	○
S 入力対象	カウント入力に使用する対象を指定します。	—	—	—	—	—	—	.B	—	—	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

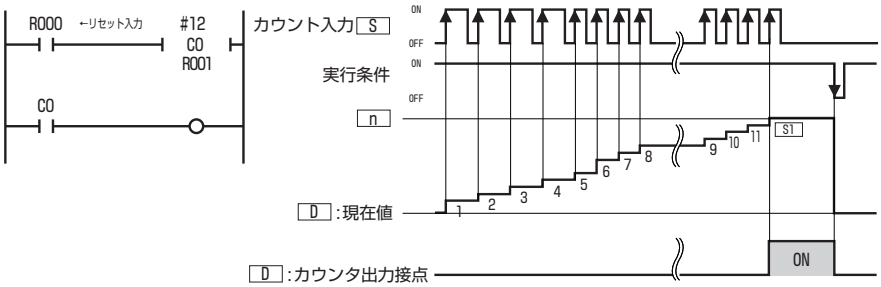
※1 • [ ]は省略可能です。実行条件はリセットを兼ねています。(実行条件がOFFのときCNT関数をリセットします)
• 実行条件を省略した場合、別途RES関数を使ってカウンタをリセットしてください。

※2 カウンタ設定値にビットデータを指定した場合は連続する32ビットが、ワードデータを指定した場合は連続する2ワードが占有されます。

※3 KV Nanoシリーズの場合、指定できるデバイス番号はC0～C255です。

動作説明

カウンタ実行条件が成立したとき、S で指定したカウント入力の立ち上がりで、指定したカウンタの現在値が加算(+1)されます。現在値が n で指定した設定値に達すると、D がONします。実行条件が不成立となったとき、指定したカウンタをOFFして現在値をリセットします。

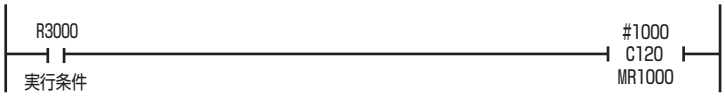


「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「タイマ･カウンタ命令」を参照してください。

●書式例

CNT (R3000, C120, 1000, MR1000)



ポイント

- **OUTC、UDC命令と同じデバイス番号は使用できません。**
- **対応機種がKV-1000の場合、他のTMR、TMH、TMS、CNT 関数(命令)、OUTC、UDC、UDT命令と同じデバイス番号は使用できません。**
- **設定値には演算式を指定できます。演算式が扱える範囲は－2147483648～+2147483647の範囲になりますので以下の場合に注意が必要です。**
  - **演算式の結果の値が負の場合、演算式の結果に4247483648を加算した値が設定値としてセットされます。**

カウンタ現在値の読み出しと変更

カウンタはサフィックスを.D(32ビット符号無し)に指定することで現在値を扱います。

●現在値の読み出し

代入文を用い、カウンタを右辺に、読み出したカウンタの現在値をの格納先を左辺に指定します。

スクリプト記述　DM1000.D = C100.D

動作説明　C100.Dの現在値を、DM1000.Dに格納します。

●現在値の書き換え

代入文を用い、読み出したカウンタの現在値の格納先を右辺に、カウンタを左辺に指定します。

スクリプト記述　C100.D = DM1100.D

動作説明　C100.Dの現在値を、DM1100.Dに格納されている値に書き換えます。

ポイント

- **現在値書き換え時に「現在値＞設定値」となる場合、現在値は設定値と同じ値に変更されます。**
- **CNT 関数のカウンタ設定値及び現在値は 32 ビット符号無しデータを使用するので、現在値や設定値の格納先や変更するデータは、必ず32ビット符号無し(.D)を使用してください。**
- **カウンタの設定値を変更する場合や設定値を読み出したい場合は、設定値に対象（32ビット符号無し）を指定してください。**
  『KV-8000/7000/5000/3000/1000 シリーズ・KV Nano シリーズ命令語リファレンスマニュアル』「タイマ・カウンタ命令」

カウンタ接点の読み出し

カウンタのサフィックスを.B(ビット型)に指定することで、接点を扱います。(.Bは省略できます)

スクリプト記述　MR1000 = C100.B

動作説明　C100.Bの接点状態をMR1000に代入します。

ポイント

- **カウンタ接点に他のビット情報を代入できません。**
  カウンタを強制的にリセットするときは、カウンタ接点をRES関数でリセットしてください。
- **カウンタのサフィックスを省略すると、カウンタはビット型として扱われます。**

参考

KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

MEMO

RFSX 入力リフレッシュ

リフレッシュX
RFSX ([実行条件**,]**※1 入力リフレッシュする先頭, リレー点数 **)**

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D 入力リフレッシュを行う先頭	入力状態をリフレッシュする対象の先頭を指定します。	—	—	—	—	—	—	.B	—	—	○	—
n リレー点数	入力状態をリフレッシュする対象のリレー点数をを指定します。(1～256)※2	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ビットデータを指定し場合、連続した16ビットを扱います。
チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16ビットを扱います。
リレー点数の指定可能範囲は、1～256です。

動作説明

通常、ラダープログラムのEND命令実行時後に一括しておこなうリフレッシュ処理を、1スキャン内の任意のタイミングで入力リフレッシュする場合に使用します。

実行条件が成り立つ(TRUE)とき、D を先頭に n で指定した点数分のビットデータの状態を入力リフレッシュします。

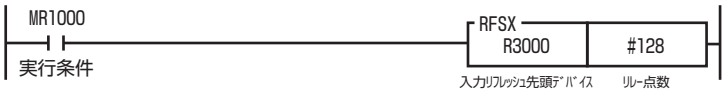
「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ダイレクト入出力命令」を参照してください。

ポイント **KV Nanoシリーズの場合、拡張ユニットに割り付いたリレーに対しては使用できません。**

●書式例

RFSX (MR1000, R3000, 128)



RFSY 出力リフレッシュ

リフレッシュY
RFSY ([実行条件**,]**※1 出力リフレッシュする先頭, リレー点数 **)**

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D 出力リフレッシュを行う先頭	出力状態をリフレッシュする対象の先頭を指定します。	—	—	—	—	—	—	.B	—	—	○	—
n リレー点数	出力状態をリフレッシュする対象のリレー点数をを指定します。(1～256)※2	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ビットデータを指定し場合、連続した16ビットを扱います。
チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16ビットを扱います。
リレー点数の指定可能範囲は、1～256です。

動作説明

通常、ラダープログラムのEND命令実行時後に一括しておこなうリフレッシュ処理を、1スキャン内の任意のタイミングで出力リフレッシュする場合に使用します。

実行条件が成り立つ(TRUE)とき、D を先頭に n で指定した点数分のビットデータの状態を出力リフレッシュします。

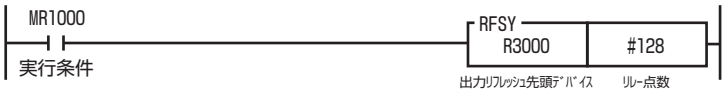
「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ダイレクト入出力命令」を参照してください。

ポイント **KV Nanoシリーズの場合、拡張ユニットに割り付いたリレーに対しては使用できません。**

●書式例

RFSY (MR1000, R3000, 128)



MEMO

6

応用関数

フロー関数 6-2
マクロ関数 6-6
ファンクションブロック関数 6-10
モジュール関数 6-18
インデックスレジスタ関数 6-20
間接指定関数 6-26
ファイルレジスタ関数 6-36
ユニットプログラム関数 6-42
バッファメモリ関数 6-52
ユニット種別判定関数 6-58
配列要素数取得関数 6-60
ワードサイズ取得関数 6-62

CALL 指定サブルーチンの起動

CALL (コール) **([**実行条件**,]**※1 サブルーチン**No.)**

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
n サブルーチンNo.	起動するサブルーチンNo.を指定します。(0～99)	—	—	—	—	—	—	—	—	※2 ○	—	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

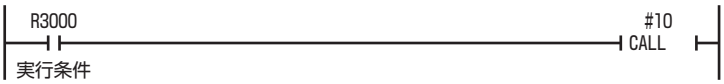
※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 サブルーチンNo.は0～99の範囲で指定します。$(16進数)では指定できません。

動作説明

実行条件が成り立つ(TRUE)とき、指定された n のサブルーチンを起動します。
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「フロー命令」を参照してください。

●書式例

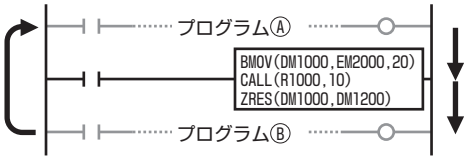
CALL (R3000, 10)



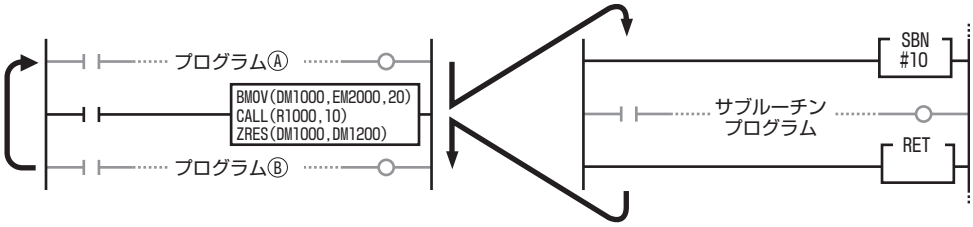
参考 サブルーチンは16重までネスティング(入れ子構造に)できます。

ポイント
- **サブルーチン番号は定数のみ指定可能で、対象や対象を含む式は使用できません。**
- **サブルーチン番号の定数は、$(16進数)での指定はできません。**
- **SBN 命令や RET 命令はスクリプト関数で用意されていないので、ラダーで記述してください。**

- **CALL関数を実行しない場合(R1000＝OFF)**



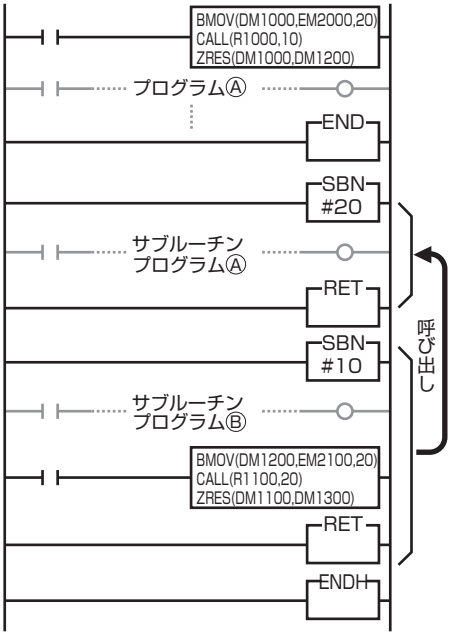
- **CALL関数を実行した場合(R1000＝ON)**



ワンポイント

■サブルーチンのネスティング構造について

- サブルーチン関数は、最大16重までネスティング（入れ子構造に）できます。
- サブルーチンの中より別のサブルーチンを呼び出すとき、呼び出されるサブルーチンはすでに定義（上の行に記述）されている必要があります。



ECALL モジュール間サブルーチンコール

Eコール
ECALL ([実行条件,]※1 “モジュール名(文字列)”)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	モジュール名(文字列)	モジュール名を文字列で指定します。									※2 ○		
n	サブルーチンNo.	呼び出すサブルーチンの番号を指定します。	—	—	—	—	—	—	—	—	※2 ○	—	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 《KV STUDIO》で設定したモジュール名を" "で囲って入力してください。
※3 サブルーチンNo.は0～99の範囲で指定します。$(16進数)では指定できません。

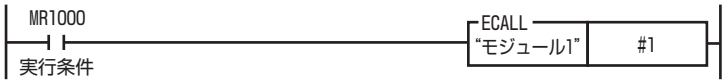
ポイント **ECALL関数は、KV-1000シリーズでは使用できません。**

動作説明

実行条件が成り立つ(TRUE)とき、S で指定したモジュールの n で指定したサブルーチンプログラムを起動します。
すべての種類のモジュール(毎スキャン、初期化、待機、定期)を指定できますが、ファンクション、ファンクションブロック、マクロは指定できません。
「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「フロー命令」を参照してください。

●書式例

ECALL (MR1000, “モジュール1”, 1)



ポイント
- **サブルーチン番号の定数は、$(16進数)での指定はできません。**
- **SBN 命令や RET 命令はスクリプト関数で用意されていないので、ラダーで記述してください。**
- **KV Nanoシリーズの場合、定周期モジュールは指定できません。**

MEMO

MCALL サブルーチン型マクロの起動

マクロコール
**MCALL ([実行条件,]※1 “マクロ名(文字列)”〔 ,マクロ引数1, マクロ引数2,
マクロ引数3…… ,マクロ引数10〕)**

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 マクロ名(文字列)	マクロ名を文字列で指定します。	—	—	—	—	—	—	—	—	※3 ○	—	—
S2 マクロ引数 ※2※4	マクロ引数を指定します。(マクロにより引数の数は異なります)	—	—	—	—	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 マクロ引数の個数や属性は、実行するマクロに設定されている引数設定によって異なります。
『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「MCALL命令」
※3 《KV STUDIO》で設定したマクロ名を" "で囲って入力してください。
※4 識別番号を使用する場合は、「(マクロ名)_(識別番号)」になります。 例)一括送信_2
詳細は、使用しているCPUユニットのユーザーズマニュアルの「マクロ」を参照してください。

動作説明

実行条件が成り立つ(TRUE)とき、S1 で指定したサブルーチン型マクロを実行します。
S2 で指定した引数をマクロに引き渡します。マクロは引数設定にしたがい、番号(P0～)や定数(V0～)として引数を扱います。
「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「マクロ命令」を参照してください。

●書式例

MCALL (R3000, "マクロ1", DM1000, 10)

ポイント
- **マクロ名には" "で囲んだサブルーチン型マクロ名以外の記述はできません。**
- **MCALL関数をボックススクリプトやIF文中で扱う場合、停止時のOFF処理の動作が命令語で実行したときと異なる場合があるなどの注意が必要です。**
  詳細は、使用しているCPUユニットのユーザーズマニュアルの「マクロ」を参照してください。

MSTRT 自己保持型マクロの起動

マクロスタート
MSTRT ([実行条件,]※1 "マクロ名(文字列)"〔,マクロ引数1, マクロ引数2, マクロ引数3…… ,マクロ引数10〕※2)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	マクロ名(文字列)	マクロ名を文字列で指定します。	—	—	—	—	—	—	—	—	※4 ○	—	—
S2	マクロ引数 ※3※5	マクロ引数を指定します。(マクロにより引数の数は異なります)	—	—	—	—	—	—	—	—	○	○	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

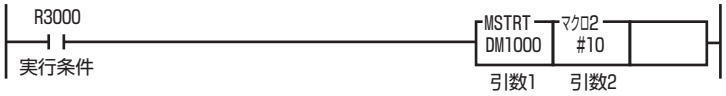
※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 実行するマクロにより引数の数が異なります。
※3 マクロ引数の個数や属性は、実行するマクロに設定されている引数設定によって異なります。
『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「MSTRT命令」
※4 《KV STUDIO》で設定したマクロ名を" "で囲って入力してください。
※5 識別番号を使用する場合は、「(マクロ名)_(識別番号)」になります。 例)一括送信_2
詳細は、使用しているCPUユニットのユーザーズマニュアルの「マクロ」を参照してください。

動作説明

実行条件が成り立つ(TRUE)とき、S1 で指定した自己保持型マクロを起動します。
S2 で指定した引数をマクロに引き渡します。マクロは引数設定にしたがい、番号(P0～)や定数(V0～)として引数を扱います。
「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「マクロ命令」を参照してください。

●書式例

MSTRT (R3000, "マクロ2", DM1000, 10)



ポイント
- **マクロ名には" "で囲んだ自己保持型マクロ名以外の記述はできません。**
- **MSTRT関数をボックススクリプトやIF文中で扱う場合、停止時のOFF処理の動作が命令語で実行したときと異なる場合があるなどの注意が必要です。**
  詳細は、使用しているCPUユニットのユーザーズマニュアルの「マクロ」を参照してください。

MEND 自己保持型マクロの終了

マクロエンド
MEND ([実行条件,]※1)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

動作説明

実行条件が成り立つ(TRUE)とき、MEND関数が記述されている自己保持型マクロを停止します。
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「マクロ命令」を参照してください。

●書式例

MEND (R3000 = OFF)

参 考 「MSTRT関数」(6-7ページ)

ポイント
- **MEND関数は、自己保持型マクロ内をスクリプトで記述する場合のみ使用可能です。**
- **自己保持型マクロ内には、少なくとも一つのMEND関数(命令)を記述する必要があります。**

MEMO

FB ファンクションブロックの起動

ファンクションブロック
FB ([実行条件,]※1 "ファンクションブロック名(文字列)"〔,引数1, 引数2, 引数3…… ,引数64〕※2)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	ファンクションブロック名(文字列)	ファンクションブロック名を文字列で指定します。	—	—	—	—	—	—	—	—	※4 ○	—	—
S2	引数※3※5	引数を指定します。(ファンクションブロックにより引数の数は異なります)	—	—	—	—	—	—	—	—	○	○	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 実行するファンクションブロックにより引数の数が異なります。
※3 引数の個数や属性は、実行するファンクションブロックに設定されている引数設定によって異なります。
『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「ファンクションブロック命令」
※4 《KV STUDIO》で設定したファンクションブロック名を" "で囲って入力してください。
詳細は『KV-8000シリーズユーザーズマニュアル』および『KV-7000シリーズユーザーズマニュアル』の「ファンクションブロック」を参照してください。

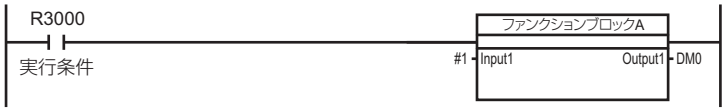
- FB関数は、KV STUDIO Ver.9.0以降のバージョンでのみ使用できます。
- FB関数は、KV-8000シリーズCPUユニットおよびCPU機能バージョン2.0以降のKV-7000シリーズCPUユニットでのみ使用できます。
「KV-7500/7300 CPU機能バージョンについて」(3ページ)

動作説明

実行条件が成り立つ(TRUE)とき、S1 で指定したファンクションブロックを起動します。
S2 で指定した引数をファンクションブロックに引き渡します。ファンクションブロックは引数設定のデータ型に従い、引数を扱います。
「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「ファンクションブロック命令」を参照してください。

●書式例

FB (R3000, "ファンクションブロックA", 1, DM0)



ポイント
- **ファンクションブロック名には" "で囲んだファンクションブロック名以外の記述はできません。**
- **「実行停止時にOFF処理する」設定を有効にしたFB関数をボックススクリプトやIF文中で扱う場合、停止時のOFF処理の動作が命令語で実行したときと異なる場合があるなどの注意が必要です。**
詳細は『KV-8000シリーズユーザーズマニュアル』および『KV-7000シリーズユーザーズマニュアル』の「ファンクションブロック」を参照してください。

FBCALL ファンクションブロックコール

ファンクションブロックコール
FBCALL ([実行条件,]※1 "ファンクションブロック名(文字列)" , インスタンス〔,引数1, 引数2, 引数3..... ,引数64〕※2)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	ファンクションブロック名(文字列)	ファンクションブロック名を文字列で指定します。	—	—	—	—	—	—	—	—	※4 ○	—	—
S2	インスタンス(文字列)	インスタンスを文字列で指定します。	—	—	—	—	—	—	—	—	※4 ○	—	—
S3	引数※3	引数を指定します。	—	—	—	—	—	—	—	—	○	○	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 実行するファンクションブロックにより引数の数が異なります。
※3 引数の個数や属性や、実行するファンクションブロックに設定されている引数設定により異なります。
『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「ファンクションブロック命令」
※4 《KV STUDIO》で設定したファンクションブロック名を" "で囲って入力してください。

- FBCALL関数は、KV STUDIO Ver.11.0以降でのみ使用できます。
- FBCALL関数は、CPU機能バージョン2.0以降のKV-8000でのみ使用できます。

「KV-8000 CPU機能バージョンについて」(2ページ)

動作説明

実行条件が成り立つ(TRUE)とき、S1 で指定したファンクションブロックの S2 で指定したインスタンスを実行します。実行条件をOFFするとインスタンスの動作は停止します。S3 で指定した引数をファンクションブロックに引き渡します。ファンクションブロックは引数設定のデータ型に従い、引数を扱います。
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「ファンクションブロック命令」を参照してください。

●書式例

FBCALL (R3000, "ファンクションブロックA", インスタンスB, #1, DM0)

参 考

名前付き引数で記述すると、ファンクションブロックの引数設定で引数の順番が変わった場合でも引数の指定順を変更する必要がなくなります。名前付き引数は以下のとおり記述します。

FBCALL(R3000, "ファンクションブロックA", インスタンスB, Input1:=#1, OutPut1=>DM10, ENO=>MR000)

IN引数、IN-OUT引数は:=、OUT引数は=>で引数を指定します。ENOは使用しない場合、省略可能です。

名前付き引数では、ユニット引数は記述できません。ユニット引数を設定しているファンクションブロックは、ラダープログラムで記述してください。

ポイント

- **ファンクションブロック名には" "で囲んだファンクションブロック名以外の記述はできません。**
- **インスタンスは""で囲む必要はありません。**
- **「実行停止時にOFF処理する」設定を有効にしたFBCALL関数をボックススクリプトやIF文中で扱う場合、停止時のOFF処理の動作が命令語で実行したときと異なる場合があるなどの注意が必要です。**
- **詳細は『KV-8000 シリーズユーザーズマニュアル』「ファンクションブロック」を参照してください。**

FBSTRT ファンクションブロックスタート

ファンクションブロックスタート

FBSTRT ([実行条件,]※1 "ファンクションブロック名(文字列)" , インスタンス〔 ,引数1, 引数2, 引数3..... ,引数64〕※2)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	ファンクションブロック名(文字列)	ファンクションブロック名を文字列で指定します。	—	—	—	—	—	—	—	—	※4 ○	—	—
S2	インスタンス(文字列)	インスタンスを文字列で指定します。	—	—	—	—	—	—	—	—	○	—	—
S3	引数※3	引数を指定します。	—	—	—	—	—	—	—	—	○	○	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 実行するファンクションブロックにより引数の数が異なります。
※3 引数の個数や属性や、実行するファンクションブロックに設定されている引数設定により異なります。
『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「ファンクションブロック命令」
※4 《KV STUDIO》で設定したファンクションブロック名を" "で囲って入力してください。

- FBSTRT関数は、KV STUDIO Ver.11.0以降でのみ使用できます。
- FBSTRT関数は、CPU機能バージョン2.0以降のKV-8000でのみ使用できます。

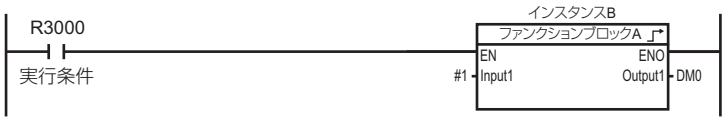
「KV-8000 CPU機能バージョンについて」(2ページ)

動作説明

実行条件が成り立つ(TRUE)とき、S1 で指定したファンクションブロックの S2 で指定したインスタンスを実行します。実行条件をOFFしてもインスタンスの動作は継続します。ファンクションブロック内でFEND命令が実行されると動作を停止します。
S3 で指定した引数をファンクションブロックに引き渡します。ファンクションブロックは引数設定のデータ型に従い、引数を扱います。
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「ファンクションブロック命令」を参照してください。

●書式例

FBSTRT (R3000, "ファンクションブロックA", インスタンスB, #1, DM0)



参 考

名前付き引数で記述すると、ファンクションブロックの引数設定で引数の順番が変わった場合でも引数の指定順を変更する必要がなくなります。名前付き引数は以下のとおり記述します。

FBSTRT(R3000, "ファンクションブロックA", インスタンスB, Input1:=#1, OutPut1=>DM10, ENO=>MR000)

IN引数、IN-OUT引数は:=、OUT引数は=>で引数を指定します。ENOは使用しない場合、省略可能です。

名前付き引数では、ユニット引数は記述できません。ユニット引数を設定しているファンクションブロックは、ラダープログラムで記述してください。

ポイント

- **ファンクションブロック名には" "で囲んだファンクションブロック名以外の記述はできません。**
- **インスタンスは""で囲む必要はありません。**
- **「実行停止時にOFF処理する」設定を有効にしたFBSTRT関数をボックススクリプトやIF文中で扱う場合、停止時のOFF処理の動作が命令語で実行したときと異なる場合があるなどの注意が必要です。**
- **詳細は『KV-8000 シリーズユーザーズマニュアル』「ファンクションブロック」を参照してください。**

FEND　ファンクションブロックの終了

ファンクションブロックエンド
FEND ([実行条件,]※1)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1　[　]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

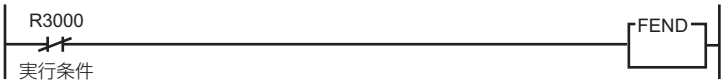
- FEND関数は、KV STUDIO Ver.9.0以降のバージョンでのみ使用できます。
- FEND関数は、KV-8000シリーズCPUユニットおよびCPU機能バージョン2.0以降のKV-7000シリーズCPUユニットでのみ使用できます。
  「KV-7500/7300　CPU機能バージョンについて」(3ページ)

動作説明

実行条件が成り立つ(TRUE)とき、FEND関数が記述されているファンクションブロックを停止します。詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ファンクションブロック命令」を参照してください。

●書式例

FEND (R3000 = OFF)



参考　「FBSTRT関数」(6-13ページ)

ポイント　**FEND関数は、ファンクションブロック内でのみ使用可能です。**

FUN　ファンクションの起動

ファンクション
FUN ([実行条件,]※1 **"**ファンクション名(文字列)**"**〔 **,**引数**1,** 引数**2,**
引数**3…… ,**引数**64**〕**)**

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	ファンクション名(文字列)	ファンクション名を文字列で指定します。	—	—	—	—	—	—	—	—	※3 ○	—	—
S2	引数※2	引数を指定します。(ファンクションにより引数の数は異なります)	—	—	—	—	—	—	—	—	○	○	○
R	戻り値	FUN関数は引数設定で設定したデータ型を持ちます。※4	○	○	○	○	○	○	○	○	—	○	—

※1　[　]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2　引数の個数や属性は、実行するファンクションに設定されている引数設定によって異なります。
『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「FUN命令」
※3　《KV STUDIO》で設定したファンクション名を"　"で囲って入力してください。
詳細は『KV-8000シリーズユーザーズマニュアル』および『KV-7000シリーズユーザーズマニュアル』の「ファンクションブロック」を参照してください。
※4　CPU 機能バージョン 2.0 以降の KV-8000 の場合のみ戻り値を持ちます。戻り値に格納する値はファンクション内でのプログラムで指定します。

- FUN関数は、KV STUDIO Ver.9.0以降のバージョンでのみ使用できます。
- FUN関数は、KV-8000シリーズCPUユニットおよびCPU機能バージョン2.0以降のKV-7000シリーズCPUユニットでのみ使用できます。
「KV-7500/7300　CPU機能バージョンについて」(3ページ)

動作説明

実行条件が成り立つ(TRUE)とき、S1 で指定したファンクションを実行します。
S2 で指定した引数をファンクションに引き渡します。ファンクションは引数設定のデータ型に従い、引数を扱います。
「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ファンクションブロック命令」を参照してください。

●書式例

FUN (R3000, "ファンクションB", EM0, MR000)

参 考

参考　名前付き引数で記述すると、ファンクションの引数設定で引数の順番が変わった場合でも引数の指定順を変更する必要がなくなります。名前付き引数は以下のとおり記述します。

FUN(R3000, "ファンクションB", Input1:=EM0, OutPut1=>MR000, ENO=>MR100)

IN引数、IN-OUT引数は:=、OUT引数は=>で引数を指定します。ENOは使用しない場合、省略可能です。

名前付き引数では、ユニット引数は記述できません。ユニット引数を設定しているファンクションブロックは、ラダープログラムで記述してください。

ポイント

- **ファンクション名には" "で囲んだファンクション名以外の記述はできません。**
- **名前付き引数で記述できるのは、KV STUDIO Ver.11.0以降で、機種がCPU機能バージョン2.0以降のKV-8000の場合のみです。**

MDSTRT モジュール実行開始

モジュールスタート
MDSTRT ([実行条件,]※1 “モジュール名(文字列)”)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S モジュール名(文字列)	モジュール名を文字列で指定します。	—	—	—	—	—	—	—	—	※2 ○	—	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 《KV STUDIO》で設定したモジュール名を" "で囲って入力してください。

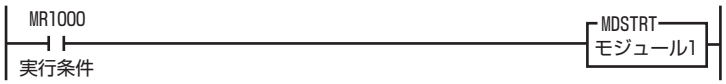
ポイント **MDSTRT関数は、KV STUDIO Ver.4.00以降のバージョンでのみ使用できます。**

動作説明

実行条件が成り立つ(TRUE)とき、S で指定したモジュールを起動します。
起動したモジュールは、次のスキャンから動作を開始します。
すでにモジュールが起動している場合、何もしません。
「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「モジュール命令」を参照してください。

●書式例

MDSTRT (MR1000, “モジュール1”)



ポイント
- **モジュール実行開始時における微分実行型命令の初期状態は、立ち上がり検出の場合がON、立ち下がり検出の場合がOFFになります。実行開始時に微分を直ちに検出することはありません。**
- **同じモジュールに対して同じスキャン内にMDSTRT関数(命令)、MDSTOP関数(命令)を複数回実行した場合は最後に実行したモジュール命令に従います。**
- **初期化モジュールは実行開始･停止できません。**

MDSTOP モジュール実行停止

モジュールストップ
MDSTOP ([実行条件,]※1 “モジュール名(文字列)”)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S モジュール名(文字列)	モジュール名を文字列で指定します。	—	—	—	—	—	—	—	—	※2 ○	—	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 《KV STUDIO》で設定したモジュール名を" "で囲って入力してください。

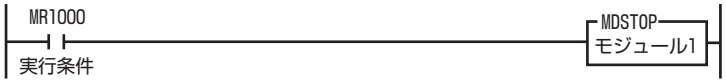
ポイント **MDSTOP関数は、KV STUDIO Ver.4.00以降のバージョンでのみ使用できます。**

動作説明

実行条件が成り立つ(TRUE)とき、S で指定したモジュールを停止します。
すでに停止しているモジュールに対しては、何もしません。
自分自身のモジュールを停止することも可能です。
「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「モジュール命令」を参照してください。

●書式例

MDSTOP (MR1000, “モジュール1”)



ポイント
- **同じモジュールに対して同じスキャン内にMDSTRT関数(命令)、MDSTOP関数(命令)を複数回実行した場合は最後に実行したモジュール命令に従います。**
- **全てのモジュールを停止しないでください。全モジュールを停止した場合、以下のいずれかの方法で復帰してください。(一旦プログラムを停止しなければ復帰できません。)**
  - **CPUユニットの電源を一度切り、再度、電源を入れてください。**
  - **CPUユニットの「RUN-PROG切換スイッチ」を一度、"PROGモード" に切り換えて再度、RUNモードにしてください。**
  - **《KV STUDIO》でプログラムのRUNを一度OFFし、再度ONしてください。**
- **初期化モジュールは実行開始・停止できません。**

ZPUSH インデックスレジスタの一括退避

Zプッシュ
ZPUSH ([実行条件,]※1 退避先)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D 退避先※2	退避先データブロックの先頭を指定します。	※3 .S	※3 .S	※4 .L	※4 .L	—	—	—	—	—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 T、C、CR、CTC、CTH、Zは使用できません。
※3 退避先の型を.Sで指定した場合:
- 連続する13ワードを占有します。ネスティングをする場合、(退避する回数×12ワード＋1)ワードを占有します。
- 対応機種がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合、下位16ビットのみを扱います。

※4 退避先の型を.Lで指定した場合:
- 連続する25ワードを占有します。ネスティングをする場合、(退避する回数×24ワード＋1)ワードを占有します。
- 対応機種がKV-1000の場合、.Sとして実行します。

ポイント **Z(インデックスレジスタ)は、対応機種がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合.L(32ビット符号付き)として扱い、対応機種がKV-1000の場合.S(16ビット符号付き)として扱います。**

動作説明

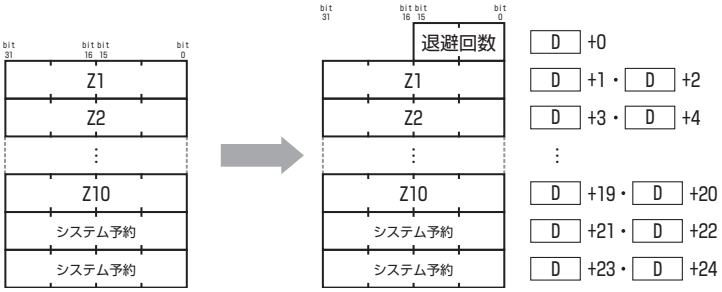
実行条件が成り立つ(TRUE)とき、D で指定したデータブロックにZ1～Z10のインデックスレジスタの値を格納(退避)します。関数実行後[D +0]の値に+1します。

ポイント
- **最初の「ZPUSH 関数」の実行前に、D +0 に 0 をセットしてください。2 回目以降の「ZPUSH関数」の場合は必要ありません。**
- **「ZPUSH関数」と「ZPOP関数」は対で使用します。**
- **「ZPUSH関数」で占有するの範囲には、他の関数、命令等で書き込まないでください。**
- **データブロックの範囲がデバイスの範囲や変数として設定した範囲を超えた場合、関数は実行されません。**

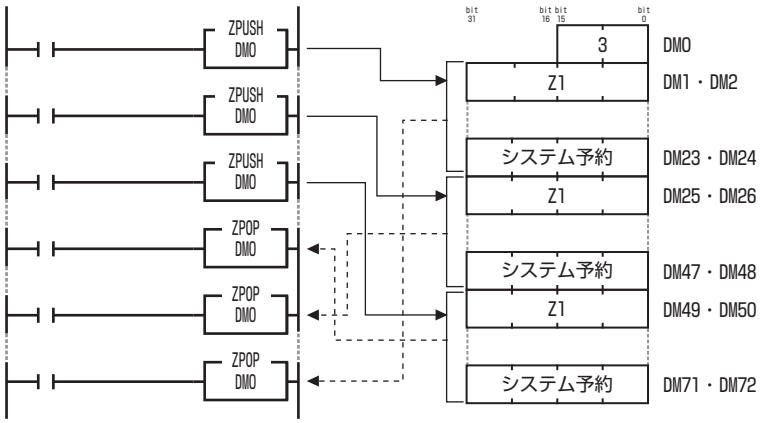
「演算フラグについて(CR2009～CR2012)」(1-18ページ)

■対応機種がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズ

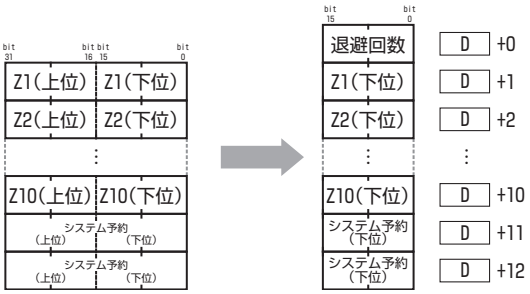
- D :退避先が32ビットの場合



参 考 「ZPUSH関数」と「ZPOP関数」はネスティング構造にできます。ネスティングをおこなう場合、同じ先頭を指定します。

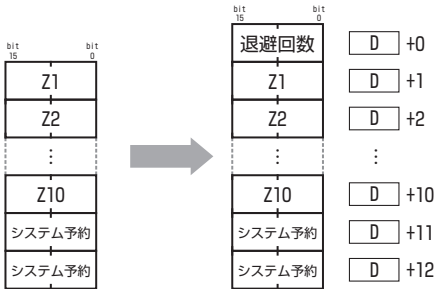


- D :退避先が16ビットの場合



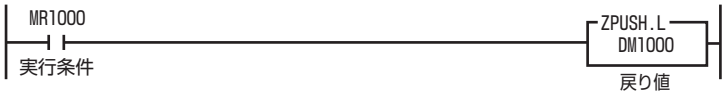
ポイント **KV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズでは、Z(インデックスレジスタ)を.L(32ビット符号付き)として扱うため、D に16ビット型サフィックス(.S、.U)を指定した場合、全てのZの下位16ビットのみを退避先に格納し、上位16ビットを切り捨てます。**
この場合、ZPOP関数(命令)を使用しても退避時の値は復帰できません。
Zに.Sの範囲(－32768～+32767)を超える値を格納するプログラムを使用している場合は、D に.Sを指定しないでください。

■対応機種がKV-1000



●書式例

ZPUSH (MR3000, DM1000.L)



詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「インデックスレジスタ命令」を参照してください。

MEMO

ZPOP インデックスレジスタの一括復帰

Zポップ
ZPOP ([実行条件,]※1 退避先)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D 退避先※2	退避先データブロックの先頭を指定します。	※3 .S	※3 .S	※4 .L	※4 .L	—	—	—	—	—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 T、C、CR、CTC、CTH、Zは使用できません。
※3 退避先の型を.Sで指定した場合:
- 連続する13ワードを占有します。ネスティングをする場合、(退避する回数×12ワード＋1)ワードを占有します。
- 対応機種がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合、各Zの下位16ビットに値を格納します。

※4 退避先の型を.Lで指定した場合:
- 連続する25ワードを占有します。ネスティングをする場合、(退避する回数×24ワード＋1)ワードを占有します。
- 対応機種がKV-1000の場合、.Sとして実行します。

ポイント **Z(インデックスレジスタ)は、対応機種がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズの場合.L(32ビット符号付き)として扱い、対応機種がKV-1000の場合.S(16ビット符号付き)として扱います。**

動作説明

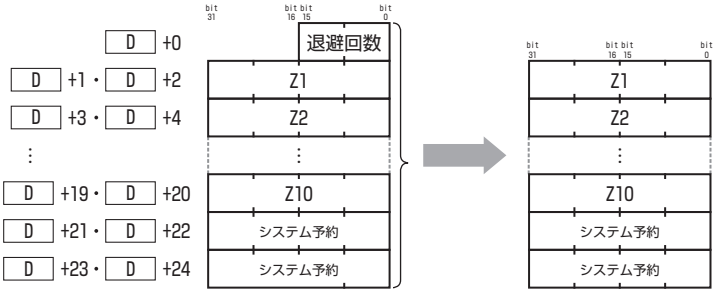
実行条件が成り立つ(TRUE)とき、D で指定したデータブロック(「ZPUSH関数」で格納したインデックスレジスタ)の値を、Z1～Z10のインデックスレジスタに読み出します。関数実行後[D +0]の値を-1します。

ポイント
- **「ZPUSH関数」と「ZPOP関数」は対で使用します。**
- **次のいずれかの条件が成立した場合、関数は実行されません。**
  - **[D +0]が0の場合**
  - **データブロックの範囲がデバイスの範囲や変数として設定した範囲を超えた場合**

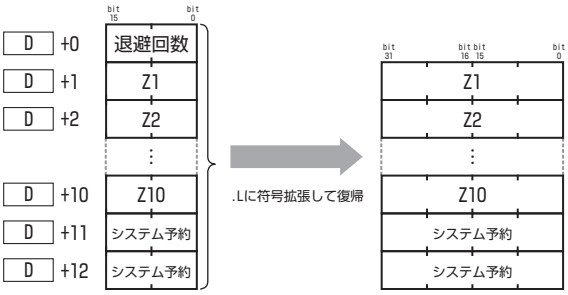
「演算フラグについて(CR2009～CR2012)」(1-18ページ)

■対応機種がKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズ

- D :退避先が32ビットの場合



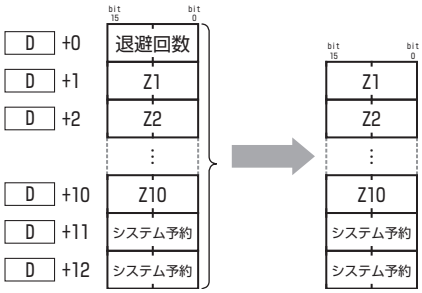
- D :退避先が16ビットの場合



ポイント

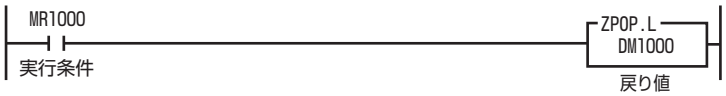
KV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズでは、Z(インデックスレジスタ)を.L(32ビット符号付き)として扱うため、D に16ビット型サフィックス(.S、.U)を指定した場合、全てのZの下位16ビットのみを退避先に格納し、上位16ビットを切り捨てます。
この場合、ZPOP関数(命令)を使用しても退避時の値は復帰できません。
Zに.Sの範囲(-32768～+32767)を超える値を格納するプログラムを使用している場合は、D 退避先に.Sを指定しないでください。

■対応機種がKV-1000



●書式例

ZPOP (MR1000, DM1000.L)



詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「インデックスレジスタ命令」を参照してください。

ADRSET 間接指定対象のアドレス格納

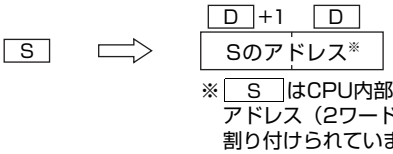
アドレスセット
ADRSET ([実行条件,]※1 間接指定対象, 格納先)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	間接指定対象	間接指定される対象を指定します。	.U	.S	.D	.D	.F	※4 .DF	.B	.T	—	○	—
D	格納先※2 ※3	アドレス格納先を指定します。	—	—	.D	.D	—	—	—	—	—	○	—
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 連続する2ワードを使用します。
※3 CMは指定できません。
※4
- 対応機種が KV-8000/7000 シリーズ CPU ユニット、CPU 機能バージョン 2.0 以降の KV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ、.DFサフィックスが使用できます。
- .DFサフィックスでは、対象にZを指定できません。

動作説明

実行条件が成り立つ(TRUE)とき、S のアドレスを、D を先頭とする連続した2ワードに格納します。格納後、間接指定(*格納先)で記述すると、間接指定を直接指定するのと同じ結果になります。

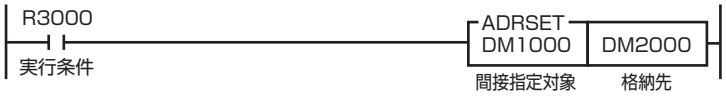


[D・D+1] には、S の値ではなく S の“アドレス”が格納されます。

「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「間接指定命令」を参照してください。

●書式例

ADRSET (R3000, DM1000, DM2000.D)



ポイント
- **アドレス格納先は、対象番号を先頭とする連続した2ワードを必ず占有します。**
- **間接指定(*格納先)を使用する前には、必ずADRSET関数で間接指定アドレスを格納先にセットしてください。**
- **アドレスの生成、および操作は、必ずADRSET, ADRINC, ADRDEC, ADRADD, ADRSUB関数を使用しておこなってください。**

サンプルプログラム

●間接指定にビットデータを指定する場合

スクリプト記述 **ADRSET** (R3000, MR1000, DM2000.D)

動作説明 R3000がONのとき、ビットデータMR1000のアドレスをDM2000.Dに格納します。
(ワンポイント参照)

●インデックスレジスタを使用する場合

スクリプト記述 **ADRSET** (DM1000:Z1, DM2000.D:Z1)

動作説明 毎スキャン(実行条件無し)、DM(1000+Z1)のアドレスをDM(2000+Z1).Dに格納します。

参考 KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ワンポイント

ビットデータのアドレスを格納して間接指定したとき、サフィックスを省略するとワードデータとして扱われます。ビットデータとして扱う場合には、.Bサフィックスで修飾します。

スクリプト記述 **ADRSET** (R1000, DM1000.D)
MR2000 = LDP (∗DM1000.B)

動作説明 R1000のアドレスをDM1000.Dに格納します。
*DM1000.B(R1000)の立ち上がりを検出したらMR2000を1スキャンONします。

参考 「LDP関数」(5-2ページ)

ADRINC 格納アドレスに1を加算

アドレスインクリメント
ADRINC ([実行条件,]※1 アドレス格納対象)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D	アドレス格納対象※2 ※3	間接指定のアドレスの格納先を指定します。	—	—	.D	.D	—	—	—	—	—	○	—
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 連続する2ワードを使用します。
※3 CMは指定できません。

動作説明

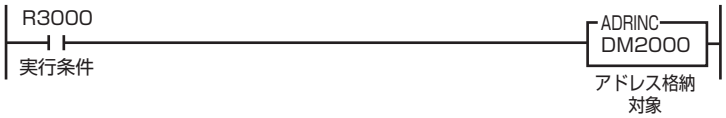
実行条件が成り立つ(TRUE)とき、D に格納されている間接指定アドレスを1つ次のアドレスにします。ワードデータアドレスが格納されているときは、次のワードデータのアドレスを示し、またビットデータアドレスが格納されているときは、次のビットデータのアドレスを示します。

D に格納されているアドレスに1を加算します。

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「間接指定命令」を参照してください。

●書式例

ADRINC (R3000, DM2000.D)



ポイント
- **アドレス格納対象は、連続した2ワードを必ず占有します。**
- **ADRINC 関数を使用する前には、必ず ADRSET 関数で間接指定アドレスをアドレス格納対象にセットしてください。**
- **アドレスの生成、および操作は、必ずADRSET, ADRINC, ADRDEC, ADRADD, ADRSUB関数を使用しておこなってください。**
- **次のいずれかの条件が成立した場合、関数は実行されません。**
  - **D に格納されているアドレスが不適切な場合。**
  - **加算後のアドレスが不適切な場合。**

サンプルプログラム

●インデックスレジスタを使用する場合

スクリプト記述　**ADRINC** (R3000, DM2000.D:Z1)

動作説明　R3000がONのとき、DM(2000+Z1).Dに格納されている間接指定アドレスを1つ次のアドレスにします。※

※実行前にADRSET関数(命令)でDM2000.Dにアドレスを格納しておく必要があります。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ワンポイント

格納されているアドレスがビットデータのとき、ADRINC関数は1つ次のビットデータを示します。このときチャンネルをまたいだ場合は、次のチャンネルのアドレスになります。

スクリプト記述　ADRSET (R3015, DM1000.D)

```
ADRINC ( *DM1000.B = ON, DM1000.D )
SET ( *DM1000.B )
```

動作説明　DM1000にビットデータR3015のアドレスを格納します。
*DM1000.B(R3015)がONのとき、DM1000に格納されたアドレスに1を加え、
*DM1000.B(R3100)をセットします。

参考　「ADRSET関数」(6-26ページ)
「SET関数」(5-12ページ)

ADRDEC 格納アドレスから1を減算

アドレスデクリメント
ADRDEC ([実行条件,]※1 アドレス格納対象)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D アドレス格納対象※2 ※3	間接指定のアドレスの格納先を指定します。	—	—	.D	.D	—	—	—	—	—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 連続する2ワードを使用します。
※3 CMは指定できません。

動作説明

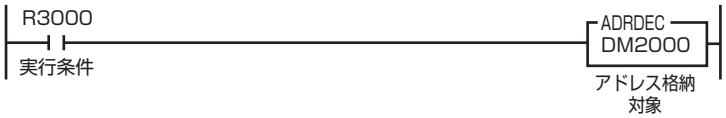
実行条件が成り立つ(TRUE)とき、D に格納されている間接指定アドレスを1つ前のアドレスにします。ワードデータアドレスが格納されているときは、前のワードデータのアドレスを示し、またビットデータアドレスが格納されているときは、前のビットデータのアドレスを示します。

D に格納されているアドレスから1を減算します。

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「間接指定命令」を参照してください。

●書式例

ADRDEC (R3000, DM2000.D)



ポイント
- **アドレス格納対象は、連続した2ワードを必ず占有します。**
- **ADRDEC関数を使用する前には、必ずADRSET関数で間接指定アドレスをアドレス格納対象にセットしてください。**
- **アドレスの生成、および操作は、必ずADRSET, ADRINC, ADRDEC, ADRADD, ADRSUB関数を使用しておこなってください。**
- **次のいずれかの条件が成立した場合、関数は実行されません。**
  - **D に格納されているアドレスが不適切な場合。**
  - **減算後のアドレスが不適切な場合。**

サンプルプログラム

●インデックスレジスタを使用する場合

スクリプト記述　**ADRDEC** (R3000, DM2000.D:Z1)

動作説明　R3000がONのとき、DM(2000+Z1).Dに格納されている間接指定アドレスを1つ前のアドレスにします。※

※実行前にADRSET関数(命令)でDM2000.Dにアドレスを格納しておく必要があります。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ワンポイント

格納されているアドレスがビットデータのとき、ADRDEC関数は1つ前のビットデータを示します。このときチャンネルをまたいだ場合は、前のチャンネルのアドレスになります。

スクリプト記述　ADRSET (R3100, DM1000.D)

ADRDEC (*DM1000.B, DM1000.D)

SET (*DM1000.B)

動作説明　DM1000.Dにビットデータ R3100のアドレスを格納します。

*DM1000.B(R3100)がONのとき、DM1000.Dに格納されたアドレスから1を減算し、

*DM1000.B(R3015)をセットします。

参考　「ADRSET関数」(6-26ページ)
「SET関数」(5-12ページ)

ADRADD 格納アドレスを加算

アドレスアッド
ADRADD ([実行条件,]※1 アドレス加算値, アドレス格納対象)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	アドレス加算値	アドレス加算データ、または格納先を指定します。	.S	.S	.L	.L	—	—	—	—	○	○	○
D	アドレス格納対象※2	間接指定のアドレスの格納先を指定します。	—	—	.D	.D	—	—	—	—	—	○	—
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 CMは指定できません。

動作説明

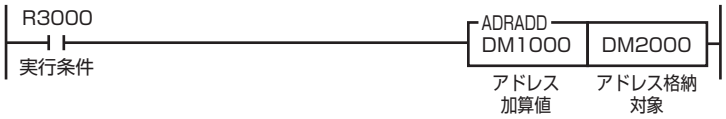
実行条件が成り立つ(TRUE)とき、D に格納されている間接指定アドレスを S で指定された値だけ加算したアドレスにします。ワードデータのアドレスが格納されているときは、アドレス加算値だけ加算したワードデータのアドレスを示し、またビットデータのアドレスが格納されているときは、アドレス加算値だけ加算したビットデータを示します。

D に格納されているアドレスを S の値だけ加算します。

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「間接指定命令」を参照してください。

●書式例

ADRADD (R3000, DM1000.S, DM2000.D)



ポイント
- **アドレス格納対象は、連続した2ワードを必ず占有します。**
- **ADRADD関数を使用する前には、必ずADRSET関数で間接指定アドレスをアドレス格納対象にセットしてください。**
- **アドレスの生成、および操作は、必ずADRSET, ADRINC, ADRDEC, ADRADD, ADRSUB関数を使用しておこなってください。**
- **次のいずれかの条件が成立した場合、関数は実行されません。**
  - **D に格納されているアドレスが不適切な場合。**
  - **加算後のアドレスが不適切な場合。**

サンプルプログラム

アドレスを格納して間接指定したとき、間接指定(*DM1000等)は16ビットデータとして扱われます。32ビットデータとして扱う場合には、.D(.L)サフィックスを付加します。
ADRADD関数は、アドレス加算値で指定した値だけ加算した番号を示しますので、32ビットのデータを扱う場合に便利です。

スクリプト記述

```
ADRSET ( DM2000, DM1000.D )
*DM1000.D += 1
ADRADD ( 2, DM1000.D )
*DM1000.D += 1
```

動作説明

DM2000のアドレスをDM1000.Dへ格納します。
格納されたアドレスを32ビットデータとして 1 を加算します。
DM1000.Dへ格納されたアドレスに2を加算し、次の32ビットデータ(DM2002.D)に1を加算します。

参 考

「ADRSET関数」(6-26ページ)

ADRSUB 格納アドレスを減算

ADRSUB (アドレスサブ) **([**実行条件**,]**※1 アドレス減算値**,** アドレス格納対象**)**

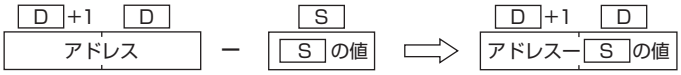
引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	アドレス減算値	アドレス減算データ、または格納先を指定します。	.S	.S	.L	.L	—	—	—	—	○	○	○
D	アドレス格納対象※2	間接指定のアドレスの格納先を指定します。	—	—	.D	.D	—	—	—	—	—	○	—
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1　[　]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2　CMは指定できません。

動作説明

実行条件が成り立つ(TRUE)とき、D に格納されている間接指定アドレスを S で指定された値だけ減算したアドレスにします。ワードデータのアドレスが格納されているときは、アドレス減算値だけ減算したワードデータのアドレスを示し、またビットデータのアドレスが格納されているときは、アドレス減算値だけ減算したビットデータを示します。

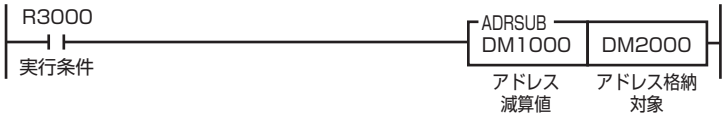
D に格納されているアドレスを S の値だけ減算します。



「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「間接指定命令」を参照してください。

●書式例

ADRSUB (R3000, DM1000.S, DM2000.D)



ポイント

- **アドレス格納対象は、連続した2ワードを必ず占有します。**
- **ADRSUB関数を使用する前には、必ずADRSET関数で間接指定アドレスをアドレス格納対象にセットしてください。**
- **アドレスの生成、および操作は、必ずADRSET, ADRINC, ADRDEC, ADRADD, ADRSUB関数を使用しておこなってください。**
- **次のいずれかの条件が成立した場合、関数は実行されません。**
  - **D に格納されているアドレスが不適切な場合。**
  - **減算後のアドレスが不適切な場合。**

サンプルプログラム

アドレスを格納して間接指定したとき、間接指定(*DM1000等)は16ビットデータとして扱われます。32ビットデータとして扱う場合には、.D(.L)サフィックスを付加します。
ADRSUB関数は、アドレス減算値で指定した値だけ減算した番号を示しますので、32ビットのデータを扱う場合に便利です。

スクリプト記述

```
ADRSET ( DM2000, DM1000.D )
*DM1000.D += 1
ADRSUB ( 2, DM1000.D )
*DM1000.D += 1
```

動作説明

DM2000のアドレスをDM1000.Dへ格納します。
格納されたアドレスを32ビットデータとして 1 を加算します。
DM1000.Dへ格納されたアドレスから2を減算し、前の32ビットデータ(DM1998.D)に1を加算します。

参 考

「ADRSET関数」(6-26ページ)

FRSET ファイルレジスタのバンク切り換え

ファイルレジスタバンク切換

FRSET ([実行条件,]※1 バンク番号)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
n バンク番号.	切り換え後のカレントバンク番号を指定します。(0～15※2)	.U	.U	.U	.U	—	—	—	—	※2 ○	—	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

※2 バンク番号は、KV-8000/7500/7300の場合：0～15、KV-5500/5000/3000の場合：0～3の範囲で指定します。

動作説明

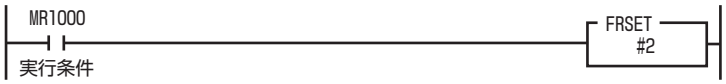
実行条件が成り立つ(TRUE)とき、ファイルレジスタのカレントバンク番号をnで指定した番号に切り換えます。

「演算フラグについて（CR2009～CR2012）」（1-18ページ）

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ファイルレジスタ命令」を参照してください。

●書式例

FRSET (MR1000, 2)



MEMO

FRSTM ファイルレジスタの一括保存

ファイルレジスター括保存
戻り値※1 =**FRSTM (ドライブ番号**※1 **,**バンク番号**,**ファイル番号**,**保存結果格納対象**)**

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
n1	ドライブ番号※2	保存するドライブ番号を指定します。※2 (0～1)	.U	.U	.U	.U	—	—	—	—	※2 ○	○	○
n2	バンク番号.	一括保存するファイルレジスタのバンク番号を指定します。(0～15※3)	.U	.U	.U	.U	—	—	—	—	※3 ○	○	○
S	ファイル番号	データを保存するときのファイル番号を指定します。(0～999)	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
D	保存結果格納対象	保存結果を通知するビットデータの先頭を指定します。	—	—	—	—	—	—	.B	—	—	○	—
R	戻り値	関数の起動に成功したらONし、失敗のときはOFFになります。	—	—	—	—	—	—	.B	—	—	○	—

※1 戻り値は省略できます。省略した場合、演算結果はワークリレーに格納されます。
※2 KV-8000/7500/7300のみ、ドライブ番号(0:メモリカード、1:CPUメモリ)を設定します。KV-5500/5000/3000の場合、設定できません。
※3 バンク番号は、KV-8000/7500/7300の場合:0～15、KV-5500/5000/3000の場合:0～3の範囲で指定します。
※4 ファイル番号は、0～999の範囲で指定します。
※5 T,C,CTC,CTH,Zは指定できません。
※6 保存結果格納対象にビットデータを指定した場合、連続する2ビットが通知ビットとして使用され、ワードデータを指定した場合は、連続する2ワードの各々最下位ビットが通知ビットデータとして使用されます。

動作説明

実行条件が成り立つ(TRUE)とき、n1 に n2 のファイルレジスタのデータを、S で指定した番号のファイル名で、メモリカードにバイナリ形式で一括保存します。
D に書き込み完了通知と異常終了アラーム通知が格納されます。
FRSTM関数が正常に起動できた場合、戻り値をONし、失敗した場合はOFFします。

参考
- メモリカードまたはCPUメモリに保存されるファイル名は、「¥FR¥FR(ファイル番号3桁).FRU」となります(KV-5500/5000/3000の場合、ファイルの拡張子がFRBです)。
  (例) S に10を指定した場合、「¥FR¥FR010.FRU」がメモリカードまたはCPUメモリに作成されます。
- D の通知ビットは、以下の意味を持ちます。
  (例)MR2000を指定した場合

通知ビット		ON (1、TRUE)	OFF (0、FALSE)
MR2000	完了通知	完了	未完了
MR2001	異常通知	異常終了	正常終了

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ファイルレジスタ命令」を参照してください。

●書式例

MR1000 = FRSTM (1, 10, MR2000)

FRLDM ファイルレジスタの一括読み出し

ファイルレジスター括読出
戻り値※1 =**FRLDM (ドライブ番号**※1 **,**バンク番号**,**ファイル番号**,**読出結果格納対象**)**

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
n1	ドライブ番号※2	一括読み出しするドライブ番号を指定します。※2 (0～1)	.U	.U	.U	.U	—	—	—	—	※2 ○	○	○
n2	バンク番号.	一括読み出しするファイルレジスタのバンク番号を指定します。(0～15※3)	.U	.U	.U	.U	—	—	—	—	※3 ○	○	○
S	ファイル番号	データを読み出しするときのファイル番号を指定します。(0～999)	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
D	読出結果格納対象	読出結果を通知するビットデータの先頭を指定します。	—	—	—	—	—	—	.B	—	—	○	—
R	戻り値	関数の起動に成功したらONし、失敗のときはOFFになります。	—	—	—	—	—	—	.B	—	—	○	—

※1 戻り値は省略できます。省略した場合、演算結果はワークリレーに格納されます。
※2 KV-8000/7500/7300のみ、ドライブ番号(0:メモリカード、1:CPUメモリ)を設定します。KV-5500/5000/3000の場合、設定できません。
※3 バンク番号は、KV-8000/7500/7300の場合:0～15、KV-5500/5000/3000の場合:0～3の範囲で指定します。
※4 ファイル番号は、0～999の範囲で指定します。
※5 T,C,CTC,CTH,Zは指定できません。
※6 読出結果格納対象にビットデータを指定した場合、連続する2ビットが通知ビットとして使用され、ワードデータを指定した場合は、連続する2ワードの各々最下位ビットが通知ビットデータとして使用されます。

動作説明

実行条件が成り立つ(TRUE)とき、n1に保存されている、Sで指定したデータ(¥FR¥FR(ファイル番号3桁).FRB)を読み出し、n2のファイルレジスタに格納します。
Dに書き込み完了通知と異常終了アラーム通知が格納されます。
FRLDM関数が正常に起動できた場合、戻り値をONし、失敗した場合はOFFします。

参考
- メモリカードまたはCPUメモリから読み出すファイル名は、「¥FR¥FR(ファイル番号3桁).FRU」となります(KV-5500/5000/3000の場合、ファイルの拡張子がFRBです)。
  (例)Sに10を指定した場合、「¥FR¥FR010.FRU」がメモリカードまたはCPUメモリから読み出します。
- D読出結果格納対象の通知ビットは、以下の意味を持ちます。
  (例)MR2000を指定した場合

通知ビット		ON(1、TRUE)	OFF(1、FALSE)
MR2000	完了通知	完了	未完了
MR2001	異常通知	正常終了	異常終了

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ファイルレジスタ命令」を参照してください。

●書式例

MR1000 = FRLDM (1, 10, MR2000)

UPSTRT ユニットプログラム開始

ユニットプログラム開始

UPSTRT (実行条件, ユ ニット番号, 要求詳細, 実行開始スロット番号, 実行結果)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	実行条件	実行条件に使用するビットデータを指定します。	—	—	—	—	—	—	.B	—	—	○	○
n1	ユニット番号※1	ユニット番号、または格納先を指定します。(1～48)	.U	.U	.U	.U	—	—	—	—	○	○	○
S2	要求詳細※2	実行を要求するユニットプログラムの詳細を格納した先頭を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
D1	実行開始スロット番号	開始したスロット番号の格納先を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
D2	実行結果※3	結果を格納する先頭を指定します。	—	—	—	—	—	—	—	—	—	○	—
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1　$(16進数指定)は使用できません。
※2　連続した4ワードを占有します。
※3　連続した2ワードを占有します。

ポイント

- **UPSTRT関数は、KV STUDIO Ver.9.0以降のバージョンでのみ使用できます。**
- **UPSTRT関数は、KV-8000シリーズCPUユニットおよびCPU機能バージョン2.0以降のKV-7000シリーズCPUユニットでのみ使用できます。**
  「KV-7500/7300　CPU機能バージョンについて」(3ページ)

動作説明

S1 の立ち上がりで、n1 番目のユニットの S2 で指定したユニットプログラムを S2 +2で指定したパラメータに従い開始します。
S2 +3の0ビット目がOFFの場合、自動的に割り当てられる空いているスロットを使用します。
S2 +3の0ビット目がONの場合、S2 +1で指定したスロットを使用します。
関数の完了時、使用するスロットの番号を D1 に、実行結果を D2 、D2 +1に格納します。

S2 ～：要求詳細

要求詳細			内容
S2	プログラム番号		開始するユニットプログラム番号を指定します。
S2 ＋1	スロット番号		S2 +3の0ビット目がONの場合、使用するスロット番号を指定します。
S2 ＋2	パラメータ		開始するユニットプログラムのパラメータを指定します。 フローの場合：最初に実行するブロック番号を指定します。
S2 ＋3	0ビット	スロット指定方法	ON　：S2 +1で指定したスロットを使用します。 OFF ：自動で割り当てられる空いているスロットを使用します。

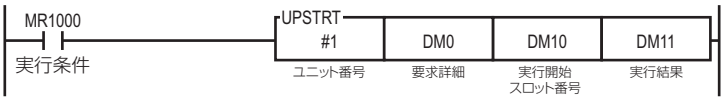
D2 ～：実行結果

実行結果			内容
D2	0ビット	完了ビット	関数実行時にOFF、完了時にONします。関数が正常完了、異常完了のどちらでもONします。
	1ビット	実行失敗ビット	関数の実行に失敗した場合、完了ビットと同時にONします。関数実行時にOFFします。
D2 ＋1	完了コード		関数完了時に完了コードを格納します。 0　　：正常に開始しました。 1　　：指定したユニットプログラムが存在しません。 2　　：空いているスロットがありません。 3　　：動作許可がOFFしています。 4　　：ユニットプログラムエラーが発生中です。 5　　：指定したスロットが存在しません。 6　　：指定したスロットが使用中です。 7　　：すでに実行中のユニットプログラムです。 8　　：指定したブロックが存在しません。 9　　：すでに実行停止中のユニットプログラムです。 10　　：指定したユニットプログラムがフローではありません。 100　：ユニットでエラーが発生しました。 1000 ：指定したユニットが存在しません。 1001 ：指定したユニットが不正です。

ユニットプログラム開始関数は、実行条件の立ち上がりで関数が実行されると、次のスキャンで入力条件がOFFであっても処理が継続されます。
結果格納先の完了ビット、実行失敗ビットは、完了時にONし、関数実行時にOFFします。
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ユニットプログラム命令」を参照してください。

●書式例

UPSTRT(MR1000, #1, DM0, DM10, DM11)



UPSTOP ユニットプログラム強制終了

ユニットプログラム強制終了

UPSTOP (実行条件, ユ ニット番号, スロット番号, 実行結果)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 実行条件	実行条件に使用するビットデータを指定します。	—	—	—	—	—	—	.B	—	—	○	○
n1 ユニット番号※1	ユニット番号、または格納先を指定します。(1～48)	.U	.U	.U	.U	—	—	—	—	○	○	○
n2 スロット番号	強制終了するスロット番号の格納先を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
D1 実行結果※2	結果を格納する先頭を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1　$(16進数指定)は使用できません。
※2　連続した2ワードを占有します。

ポイント
- **UPSTOP関数は、KV STUDIO Ver.9.0以降のバージョンでのみ使用できます。**
- **UPSTOP関数は、KV-8000シリーズCPUユニットおよびCPU機能バージョン2.0以降のKV-7000シリーズCPUユニットでのみ使用できます。**
  「KV-7500/7300　CPU機能バージョンについて」(3ページ)

動作説明

S の立ち上がりで、n1 番目のユニットの n2 で指定したスロットで実行しているユニットプログラムを強制終了し、スロットを空き状態にします。
関数完了時、実行結果を D1 、D1 +1に格納します。

参 考　「UPSTRT関数」(6-42ページ)

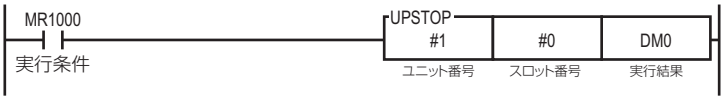
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ユニットプログラム命令」を参照してください。

D1 ～：実行結果格納先

実行結果			内容
D1	0ビット	完了ビット	関数実行時にOFF、完了時にONします。関数が正常完了、異常完了のどちらでもONします。
	1ビット	実行失敗ビット	関数の実行に失敗した場合、完了ビットと同時にONします。関数実行時にOFFします。
D1 ＋ 1	完了コード		関数完了時に完了コードを格納します。 0　　：正常に開始しました。 1　　：指定したユニットプログラムが存在しません。 2　　：空いているスロットがありません。 3　　：動作許可がOFFしています。 4　　：ユニットプログラムエラーが発生中です。 5　　：指定したスロットが存在しません。 6　　：指定したスロットが使用中です。 7　　：すでに実行中のユニットプログラムです。 8　　：指定したブロックが存在しません。 9　　：すでに実行停止中のユニットプログラムです。 10　　：指定したユニットプログラムがフローではありません。 100　：ユニットでエラーが発生しました。 1000：指定したユニットが存在しません。 1001：指定したユニットが不正です。

●書式例

UPSTOP(MR1000, #1, #0, DM0)



UFSUS　フローー時停止

フローー時停止
UFSUS (実行条件, ユ ニット番号, スロット番号, 実行結果)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 実行条件	実行条件に使用するビットデータを指定します。	—	—	—	—	—	—	.B	—	—	○	○
n1 ユニット番号※1	ユニット番号、または格納先を指定します。(1～48)	.U	.U	.U	.U	—	—	—	—	○	○	○
n2 スロット番号	一時停止するスロット番号の格納先を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
n3 要求詳細	一時停止方法を格納した対象を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	—
D1 実行結果※2	結果を格納する先頭を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1　$(16進数指定)は使用できません。
※2　連続した2ワードを占有します。

ポイント
- **UFSUS関数は、KV STUDIO Ver.9.0以降のバージョンでのみ使用できます。**
- **UFSUS関数は、KV-8000シリーズCPUユニットおよびCPU機能バージョン2.0以降のKV-7000シリーズCPUユニットでのみ使用できます。**
  「KV-7500/7300　CPU機能バージョンについて」(3ページ)

動作説明

S の立ち上がりで、n1 番目のユニットの n2 で指定したスロットで実行しているフローを n3 で指定した方法に従い一時停止します。
関数完了時、実行結果を D1 、D1 +1に格納します。

D1 ～：実行結果格納先

実行結果			内容
D1	0ビット	完了ビット	関数実行時にOFF、完了時にONします。関数が正常完了、異常完了のどちらでもONします。
	1ビット	実行失敗ビット	関数の実行に失敗した場合、完了ビットと同時にONします。関数実行時にOFFします。
D1＋1	完了コード		関数完了時に完了コードを格納します。 0　　：正常に開始しました。 1　　：指定したユニットプログラムが存在しません。 2　　：空いているスロットがありません。 3　　：動作許可がOFFしています。 4　　：ユニットプログラムエラーが発生中です。 5　　：指定したスロットが存在しません。 6　　：指定したスロットが使用中です。 7　　：すでに実行中のユニットプログラムです。 8　　：指定したブロックが存在しません。 9　　：すでに実行停止中のユニットプログラムです。 10　　：指定したユニットプログラムがフローではありません。 100　：ユニットでエラーが発生しました。 1000：指定したユニットが存在しません。 1001：指定したユニットが不正です。

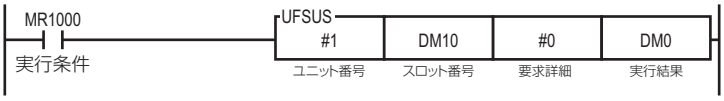
n3 ：要求詳細

要求詳細			内容
n3	0ビット	一時停止方法	ON　：実行中のブロックがあれば強制終了して、フローを一時停止します。 OFF：実行中のブロックが完了したタイミングでフローを一時停止します。

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ユニットプログラム命令」を参照してください。

●書式例

UFSUS(MR1000, #1, DM10, #0, DM0)



UFRSM フロー再開

フロー再開
UFRSM (実行条件, ユ ニット番号, スロット番号, 実行結果)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 実行条件	実行条件に使用するビットデータを指定します。	—	—	—	—	—	—	.B	—	—	○	○
n1 ユニット番号※1	ユニット番号、または格納先を指定します。(1～48)	.U	.U	.U	.U	—	—	—	—	○	○	○
n2 スロット番号	再開するスロット番号の格納先を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
D1 実行結果※2	結果を格納する先頭を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1　$(16進数指定)は使用できません。
※2　連続した2ワードを占有します。

ポイント
- **UFRSM関数は、KV STUDIO Ver.9.0以降のバージョンでのみ使用できます。**
- **UFRSM関数は、KV-8000シリーズCPUユニットおよびCPU機能バージョン2.0以降のKV-7000シリーズCPUユニットでのみ使用できます。**
  「KV-7500/7300　CPU機能バージョンについて」(3ページ)

動作説明

S の立ち上がりで、n1 番目のユニットの n2 で指定したスロットの一時停止しているフローを再開します。
命令完了時、実行結果を D1 、D1 +1に格納します。

参考 「UFSUS関数」(6-46ページ)

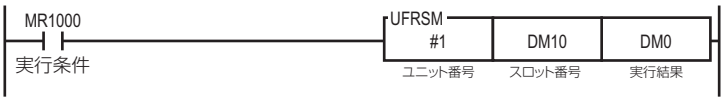
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ユニットプログラム命令」を参照してください。

D1 ～：実行結果格納先

実行結果			内容
D1	0ビット	完了ビット	関数実行時にOFF、完了時にONします。関数が正常完了、異常完了のどちらでもONします。
	1ビット	実行失敗ビット	関数の実行に失敗した場合、完了ビットと同時にONします。関数実行時にOFFします。
D1＋1	完了コード		関数完了時に完了コードを格納します。 0　　：正常に開始しました。 1　　：指定したユニットプログラムが存在しません。 2　　：空いているスロットがありません。 3　　：動作許可がOFFしています。 4　　：ユニットプログラムエラーが発生中です。 5　　：指定したスロットが存在しません。 6　　：指定したスロットが使用中です。 7　　：すでに実行中のユニットプログラムです。 8　　：指定したブロックが存在しません。 9　　：すでに実行停止中のユニットプログラムです。 10　　：指定したユニットプログラムがフローではありません。 100　：ユニットでエラーが発生しました。 1000：指定したユニットが存在しません。 1001：指定したユニットが不正です。

●書式例

UFRSM(MR1000, #1, DM10, DM0)



UMALLOC バッファメモリ確保

バッファメモリ確保
UMALLOC (実行条件, ユニット番号, 割付サイズ, 実行結果, 割付結果格納先)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 実行条件	実行条件に使用するビットデータを指定します。	—	—	—	—	—	—	.B	—	—	○	○
n1 ユニット番号※1	ユニット番号、または格納先を指定します。(1～48)	.U	.U	.U	.U	—	—	—	—	○	○	○
n2 割付サイズ	確保するワード数の格納先を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
D1 実行結果	結果を格納する先頭を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	—
D2 割付結果格納先	確保した領域の先頭アドレスの格納先を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1　$(16進数指定)は使用できません。

ポイント
- **UMALLOC関数は、KV STUDIO Ver.9.0以降のバージョンでのみ使用できます。**
- **UMALLOC関数は、KV-8000シリーズCPUユニットおよびCPU機能バージョン2.0以降のKV-7000シリーズCPUユニットでのみ使用できます。**
  「KV-7500/7300　CPU機能バージョンについて」(3ページ)

動作説明

S の立ち上がりで、n1 番目のユニットのバッファメモリから n2 で指定したワード数の領域を確保します。確保した領域の先頭アドレスを D2 に、実行結果を D1 に格納します。
確保に失敗した場合、D2 には#0が格納されます。

参考
- 領域は自動で割り当てられます。
- 確保した領域は、自由に値を読み書きできます。
- ユニットの動作許可リレーOFFで、確保した領域は解放されます

D1 : 実行結果

実行結果			内容
D1	0ビット	完了ビット	関数実行時にOFF、完了時にONします。関数が正常完了、異常完了のどちらでもONします。
	1ビット	実行失敗ビット	領域の確保に失敗した場合、完了ビットと同時にONします。関数実行時にOFFします。

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ユニットプログラム命令」を参照してください。

●書式例

UMALLOC(MR1000, #1, DM10, DM11, DM20)

UREAD バッファメモリ読み出し

バッファメモリ読み出し

UREAD ([実行条件,]※1 ユニット番号,バッファメモリ先頭アドレス,
読出データ格納先頭 ,読出データ数)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
n1	ユニット番号※2	バッファメモリデータを読み出す拡張特殊ユニット･CPUユニットのユニット番号を指定します。(0～48)	.U	.U	.U	.U	—	—	—	—	※3 ○	○	○
n2	バッファメモリ先頭アドレス※3	データを読み出すバッファメモリの先頭アドレスを指定します。(0～59999※3)	.U	.U	.U	.U	—	—	—	—	※3 ○	○	○
D	読出データ格納先頭※5	読み出したデータを格納する先頭を指定します。	.U	.S	.D	.L	.F	—	—	.T	—	○	—
n3	読出データ数※4	バッファメモリから読み出すデータ数を指定します。(1～60000※4)	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	○	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ユニット番号は、0～48(Nanoシリーズでは0～15)の範囲で指定します。$(16進数)では指定できません。
※3 バッファメモリの先頭アドレスは、KV-8000/7500/7300の場合：0～59999、そのほかの場合：0～32767の範囲で指定します。
※4 読出データ数は、KV-8000/7500/7300の場合：1～60000、そのほかの場合：1～32768の範囲で指定します。
※5 CTC,CTH,Zは指定できません。

動作説明

実行条件が成り立つ(TRUE)とき、CPUユニットに接続された n1 の拡張特殊ユニット･CPUユニットのバッファメモリを、n2 で指定されたアドレスから、n3 で指定された個数分のデータ(16ビット型ワードデータ)を読み出し、D に格納します。

「演算フラグについて（CR2009～CR2012）」（1-18ページ）

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「バッファメモリ命令」を参照してください。

●書式例

UREAD (MR1000, 3, DM1200, DM1000, DM1100)

MEMO

UWRIT バッファメモリ書き込み

バッファメモリ書き込み

UWRIT ([実行条件,]※1 ユニット番号,バッファメモリ先頭アドレス,
書込データ格納先頭 ,書込データ数)

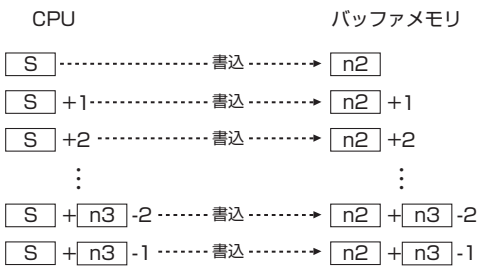
引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
n1	ユニット番号※2	バッファメモリに書き込む拡張特殊ユニット･CPUユニットのユニット番号を指定します。(0～48)	.U	.U	.U	.U	—	—	—	—	※3 ○	○	○
n2	バッファメモリ先頭アドレス※3	データを書き込むバッファメモリの先頭アドレスを指定します。(0～59999※3)	.U	.U	.U	.U	—	—	—	—	※3 ○	○	○
D	書込データ格納先頭※5	書き込むデータの先頭を指定します。	.U	.S	.D	.L	.F	—	—	.T	○	○	※6 ○
n3	書込データ数※4	バッファメモリに書き込むデータ数を指定します。(1～60000※4)	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ユニット番号は、0～48(Nanoシリーズでは0～15)の範囲で指定します。$(16進数)では指定できません。
※3 バッファメモリの先頭アドレスは、KV-8000/7500/7300の場合：0～59999、そのほかの場合：0～32767の範囲で指定します。
※4 読出データ数は、KV-8000/7500/7300の場合：1～60000、そのほかの場合：1～32768の範囲で指定します。
※5 CTC,CTH,Zは指定できません。
※6 文字列演算のみ指定できます。文字列演算以外の演算(DM0+DM2など)は記述できません。

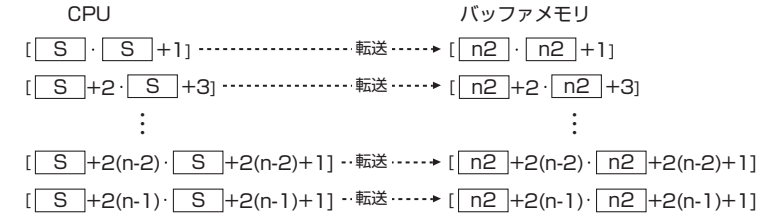
動作説明

実行条件が成り立つ(TRUE)とき、CPUユニットに接続された n1 の拡張特殊ユニット･CPUユニットのバッファメモリの、n2 で指定されたアドレスに、S に格納されている n3 で指定された個数分のデータを書き込みます。

- S :転送元が16ビット(.U/.S)の場合



- [S]:転送元が32ビット(.D/.L/.F)の場合



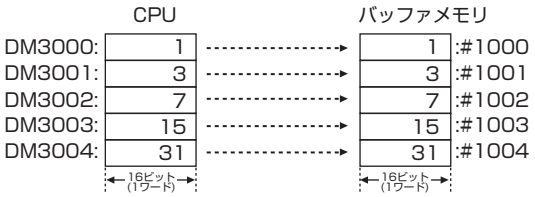
参 考

- [S]書込データ格納先頭に定数を指定した場合、UWRIT関数はUFILL関数と同じ動作をおこないます。
  「UFILL関数」(6-56ページ)

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「バッファメモリ命令」を参照してください。

●書式例

UWRIT (MR1000, 3, 1000, DM3000.U, 5)



UFILL バッファメモリのデータ列に同じ値を挿入

バッファメモリフィル

UFILL ([実行条件,]※1 ユニット番号,バッファメモリ先頭アドレス,
書込データ格納対象 ,書込データ数)

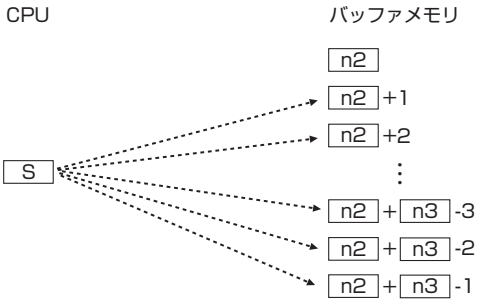
引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
n1	ユニット番号※2	バッファメモリに書き込む拡張特殊ユニット･CPUユニットのユニット番号を指定します。(0～48)	.U	.U	.U	.U	—	—	—	—	※3 ○	○	○
n2	バッファメモリ先頭アドレス※3	データを書き込むバッファメモリの先頭アドレスを指定します。(0～59999※3)	.U	.U	.U	.U	—	—	—	—	※3 ○	○	○
D	書込データ格納対象※5	書き込むデータまたは、データの格納先を指定します。	.U	.S	.D	.L	.F	—	—	.T	○	○	○
n3	書込データ数※4	バッファメモリに書き込むデータ数を指定します。(1～60000※4)	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ユニット番号は、0～48(Nanoシリーズでは0～15)の範囲で指定します。$(16進数)では指定できません。
※3 バッファメモリの先頭アドレスは、KV-8000/7500/7300の場合：0～59999、そのほかの場合：0～32767の範囲で指定します。
※4 書込データ数は、KV-8000/7500/7300の場合：1～60000、そのほかの場合：1～32768の範囲で指定します。
※5 CTC,CTH,Zは指定できません。

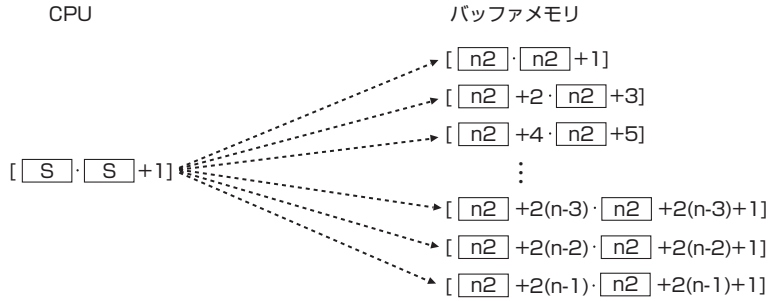
動作説明

実行条件が成り立つ(TRUE)とき、CPUユニットに接続された n1 の拡張特殊ユニット･CPUユニットのバッファメモリの、n2 で指定されたアドレスから n3 で指定された個数分の領域に、S に格納されているデータを一括で書き込みます。

- S ：転送元が16ビット(.U/.S)の場合



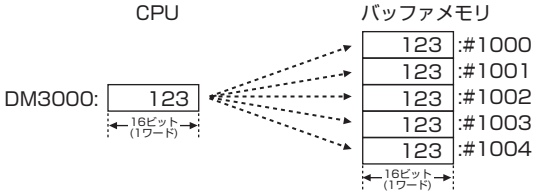
• [S]:転送元が32ビット(.D/.L/.F)の場合



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「バッファメモリ命令」を参照してください。

●書式例

UFILL (MR1000, 3, 1000, DM3000.U, 5)



IS_□□□ ユニット種別判定

ユニット種別判定
IS_□□□ ([実行条件], ユニット番号, 実行結果)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
n1 ユニット番号※1	ユニット番号、または格納先を指定します。(1～48)	.U	.U	.U	.U	—	—	—	—	○	○	○
D 実行結果	結果通知の格納先を指定します。	.B	.B	.B	.B	—	—	—	—	—	○	—

※1 $(16進数指定)は使用できません。
※2 ワードデータを指定した場合、最下位ビットを通知ビットデータとして使用します。最下位ビット以外は変化しません。

ポイント
- **ユニット種別判定関数は、KV STUDIO Ver.9.1以降のバージョンでのみ使用できます。**
- **ユニット種別判定関数は、KV-8000シリーズCPUユニットおよびCPU機能バージョン2.1以降のKV-7000シリーズCPUユニットでのみ使用できます。**
  「KV-7500/7300 CPU機能バージョンについて」(3ページ)

動作説明

実行条件の立ち上がりで、n1番目のユニットのユニット種別を確認し、命令語に対応したユニット種別と一致したらDをONします。不一致または未接続の場合、DをOFFします。

命令語とユニット種別
命令語の「IS_」を「KV-」に置き換えると、対応するユニット種別になります。
例)
(命令語) (ユニット種別)
IS_XH16ML → KV-XH16ML

●書式例

IS_XH16ML(@CR2008, #1,MR000.U)

MEMO

ACNT 配列要素数取得

戻り値 = **ACNT** (対象配列, 次元数)
配列要素数取得

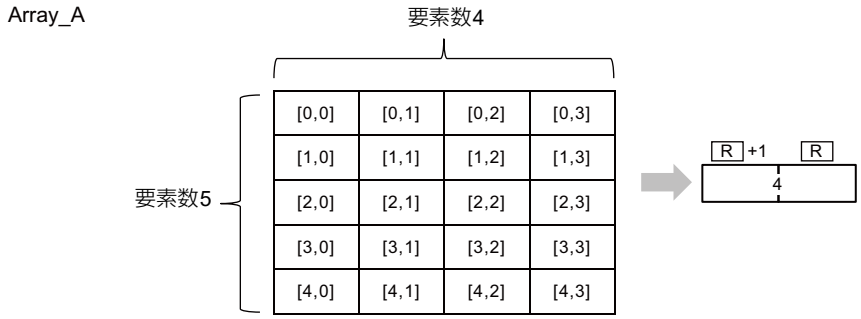
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 対象配列	要素数を取得する配列型変数を指定します。※1	—	—	—	—	—	—	—	—	—	○	—
n 次元数	次元数を指定します。	.U	—	—	—	—	—	—	—	○	○	○
R 戻り値	ACNT関数は、.Dの型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	—	—

※1　配列型以外の変数は指定できません。また、可変長配列は指定できません。
※2　戻り値に.D以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

動作説明

S の n 次元目の要素数を数え、結果を32ビットバイナリデータとして[R・R+1]に格納します。

例　S =Array_A(5×4の2次元配列)、n =2の場合



「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「配列要素数取得」を参照してください。

ポイント　**ACNT命令で取得するのはワードサイズではなく要素数です。例えば、UDINT(2ワード符号なし整数)が4個の次元を指定した場合、[R・R+1]には4が格納されます。**

●書式例

duSizeOfDimension = ACNT(Array_A, 2)

MEMO

WSIZE ワードサイズ取得

戻り値※1 = **WSIZE** (対象変数)
ワードサイズ取得

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 対象変数	ワードサイズを取得する変数を指定します。※2	—	—	—	—	—	—	—	—	—	○	—
R 戻り値	WSIZE関数は、.Dの型を持ちます。※3	.U	.S	.D	.L	.F	※4 .DF	—	—	—	—	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス / ワークメモリに格納されます。

※2 以下の変数は指定できません。
- ·TIMER型変数
- ·COUNTER型変数
- ·ファンクションブロック型変数
- ·可変長配列
- ·可変長文字列

その他の配列は指定できます。

※3 戻り値に.D以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。

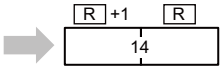
※4 デバイスZに「.DF」サフィックスは指定できません。

動作説明

S のワードサイズを32ビットバイナリデータとして[R ·R +1]に格納します。

例 S =以下の構造体型変数を指定した場合

メンバ名	データ型
member1	UINT
member2	STRING[20]
member3	REAL



→STRING[20]のワード数は(20+1)/2を切り上げて11ワードなので、1 + 11 + 2 = 14となります。

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ·KV Nanoシリーズ命令語リファレンスマニュアル』「ワードサイズ取得」を参照してください。

ポイント
- **WSIZE命令で取得するのはワードサイズです。例えば、UDINT型のデータ(2ワード符号なし整数)を4つ持つ配列型変数や構造体型変数を指定した場合、2ワード×4=8ワードなので、[R ·R +1]には8が格納されます。要素数やメンバの個数が格納されるわけではありません。**
- **[R ·R +1]にはオフセットも含めたワードサイズが格納されます。オフセットの詳細は使用しているCPUユニットのユーザーズマニュアルを参照してください。**
- **[R ·R +1]にはワードサイズに切り上げた結果が格納されます。例えば、BOOL[33]の場合は3が格納されます。**

●書式例

duSizeOfTag = WSIZE(tagAxis_A)

7

演算関数

データ転送関数 7-2
算術関数 7-14
論理演算関数 7-22
データシフト関数 7-38
データ制御関数 7-62
データ変換関数 7-80
浮動小数点関数 7-114
文字処理関数 7-154

BMOV　ワードブロックの一括データ転送

ブロックムーブ
BMOV ([実行条件,]※1 転送元, 転送先, 転送データ数)

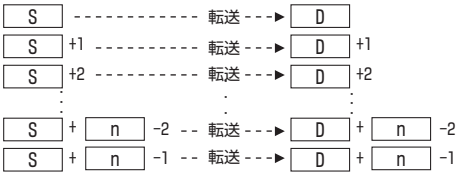
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 転送元※2※3	転送元データブロックの先頭を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
D 転送先※2※3	転送先データブロックの先頭を指定します。	S と同じ型として扱われます。				—	—	—	—	—	○	—
n 転送データ数	転送するデータ数の格納先、または転送データ数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1　[　]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2　デバイスT,Cを指定すると現在値が転送元となります。
※3　ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。

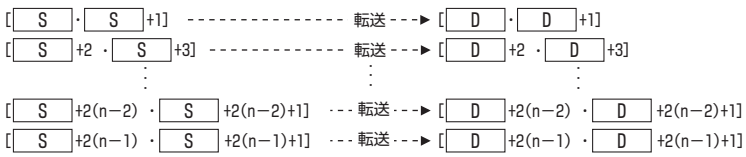
動作説明

実行条件が成り立つ(TRUE)とき、S を先頭とする n 分のワードブロックの内容を、D を先頭とする転送データ数分のワードブロックへブロック転送します。(転送対象が32ビットデータのとき、ワードブロックは指定された転送データ数×2ワードになります。)

- **S :転送元が16ビット(.U/.S)の場合**



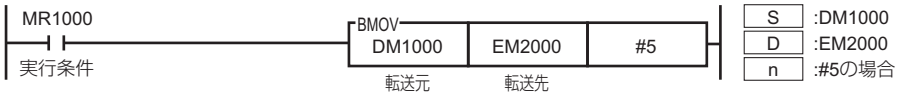
- **S :転送元が32ビット(.D/.L/.F)の場合**

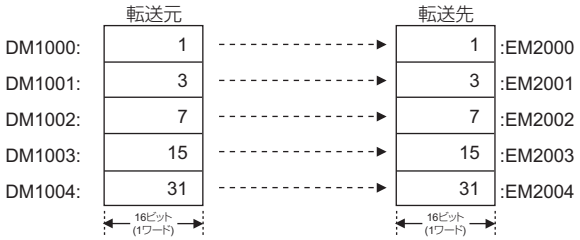


「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ転送命令」を参照してください。

●書式例

BMOV (MR1000, DM1000.U, EM2000, 5)



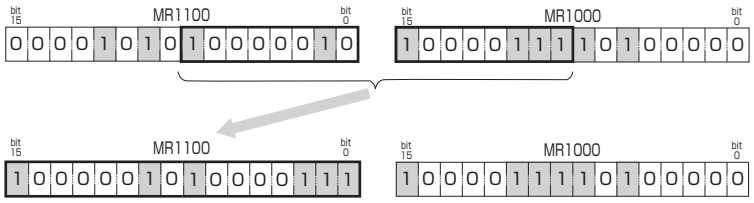


ポイント　n =0の場合、関数は実行されません。

サンプルプログラム

スクリプト記述　**BMOV** (MR1008.U, MR1100.U, 1)

動作説明　MR1008を先頭とする1ワード(16ビット)分の内容を、MR1100を先頭とする1ワード(16ビット)にブロック転送します



参考
- 転送元と転送先のブロック(ビット)がチャンネルを越えて重複していても、正しく転送されます。
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

FMOV データ列に同じ値を代入

フィルムーブ
FMOV ([実行条件,]※1 転送元, 転送先, 転送データ数)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 転送元※2	転送元、または転送元データの格納先を指定します。	.U	.U	.D	.D	.D	※3 .DF	—	—	○	○	○
D 転送先※2	転送先データブロックの先頭を指定します。	S と同じ型として扱われます。						—	—	—	○	—
n 転送データ数	転送するデータ数の格納先、または転送データ数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

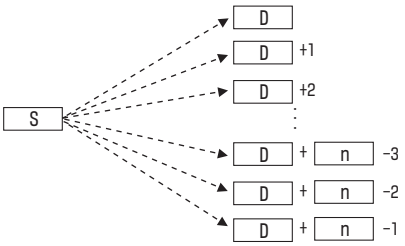
※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 デバイスT, C, CTHを指定すると、それぞれの現在値が指定の対象となります。デバイスCTCを指定すると、設定値が指定の対象となります。
※3 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

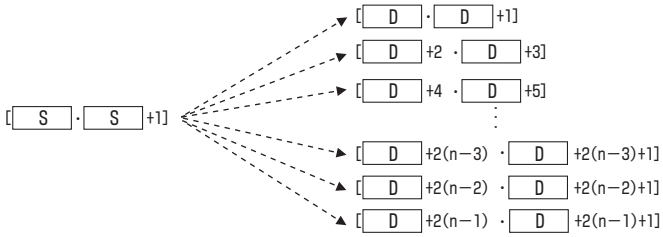
実行条件が成り立つ(TRUE)とき、S 格納された16ビット/32ビットデータ(符号無し/符号付き)を、D を先頭とする n 分のワードブロックへブロック転送します。(転送対象が32ビットデータのとき、ワードブロックは指定された転送データ数×2ワードになります。)

- **S :転送元が16ビット(.U/.S)の場合**

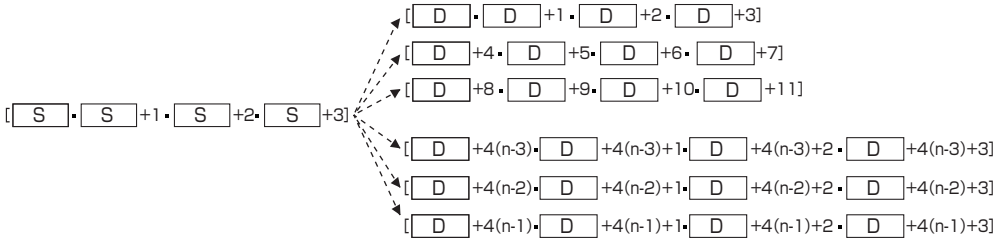


ポイント
FMOV(-1, DM0, 10)のように転送元に負の数を指定すると、FMOV.L -1 DM0 10 というラダーに展開され、転送対象が32ビットデータとみなされるため、20ワードを転送します。転送元を16ビットデータとするときはFMOV(TOS(-1), DM0, 10)と記述してください。

- **S :転送元が32ビット(.D/.L/.F)の場合**



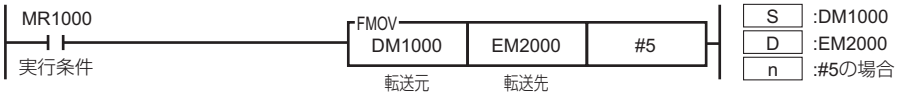
- S :転送元が64ビット(.DF)の場合

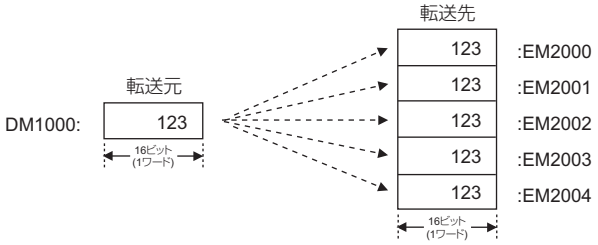


「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ転送命令」を参照してください。

●書式例

FMOV (MR1000, DM1000.U, EM2000, 5)



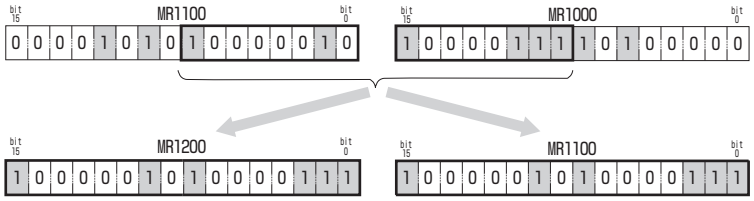


ポイント　n =0の場合、関数は実行されません。

サンプルプログラム

スクリプト記述　**FMOV** (MR1008.U, MR1100.U, 2)

動作説明　FMOV関数は、転送元、転送先にビットデータを使用できます。
MR1008を先頭とする1ワード(16ビット)分の内容を、MR1100を先頭とする2ワード(32ビット)にブロック転送します



参考
- 転送元と転送先のブロック(ビット)がチャンネルを越えて重複していても、正しく転送されます。
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

PMOV 連続ビット情報を指定位置に転送

パーシャルムーブ
PMOV ([実行条件,]※1 転送元, 転送元先頭ビット位置,
転送先, 転送先先頭ビット位置,転送ビット数)

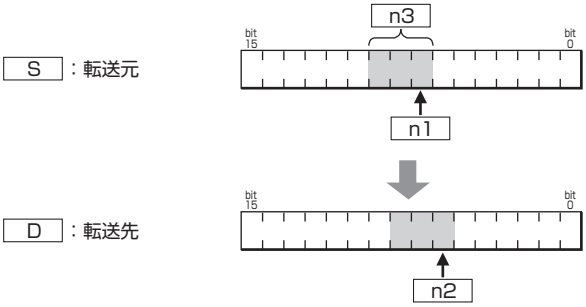
引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	転送元※2※3	転送元データの格納先を指定します。	.U	.U	.D	.D	.D	—	—	—	○	※6 ○	○
n1	転送元先頭ビット位置※4※5	転送元の先頭ビット位置の格納先、または先頭ビット位置を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
D	転送先※2※3	転送先を指定します。	S と同じ型として扱われます。						—	—	—	○	—
n2	転送先先頭ビット位置※4※5	転送先の先頭ビット位置の格納先、または先頭ビット位置を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
n3	転送ビット数	転送するデータ数の格納先、または転送データ数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ビットデータを指定した場合、連続する16ビット(.Uのとき)/32ビット(.Dのとき)を占有します。
※3 S と D は同じ対象でも構いません。
※4 例えば、サフィックスが .Uで、n1 (n2)にDM1000が指定され、その値が20(2進数:0000 0000 0001_0100)の場合、下位4ビットの値は4(0100)なので n1 (n2)の値として4が使用されます。
※5 S 転送元が16ビット(.U/.S)の場合は0～15を、32ビット(.D/.L)の場合は0～31を指定します。
※6 転送元に定数が指定できるのは、以下の条件を満たすときだけです。
- KV STUDIOのバージョンがVer.6.00以上のとき
- 対応機種が KV-8000/7000 シリーズ CPU ユニット、CPU 機能バージョン 2.0 以降の KV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットのとき

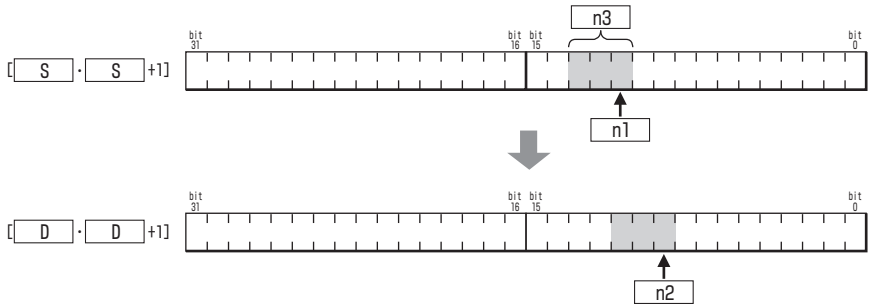
動作説明

実行条件が成り立つ(TRUE)とき、S に格納された16ビット/32ビットデータの n1 で指定されたビット位置以降のビットを、D の n2 で指定されたビット位置以降のビットに、n3 で指定されたビット数分ブロック(ビット)転送します。

- **S :転送元が16ビット(.U/.S)の場合**



- [S]:転送元が32ビット(.D/.L/.F)の場合

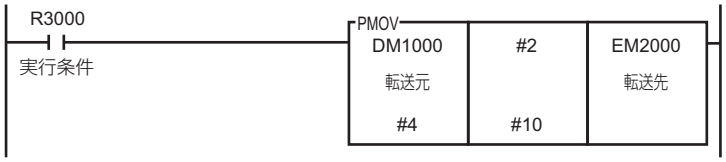


「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ転送命令」を参照してください。

参 考　転送されていない部分(転送対象外)のビットは、元の値が保持されます。

●書式例

PMOV (R3000, DM1000.U, 2, EM2000, 4, 10)



ポイント　**次のいずれかの条件が成立した場合、関数は実行されません。**

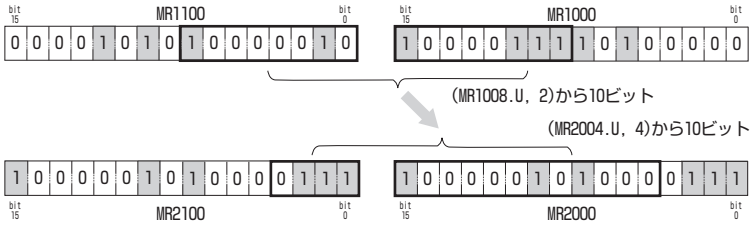
- **[S]のサフィックスが.Uで、[n3]=0または16<[n1]+[n3]または16<[n2]+[n3]の場合。**
- **[S]のサフィックスが.Dで、[n3]=0または32<[n1]+[n3]または32<[n2]+[n3]の場合。**

サンプルプログラム

●ビットデータを転送する場合

スクリプト記述　**PMOV** (MR1008.U, 2, MR2004.U, 4, 10)

動作説明　MR1008.Uの2ビット目を先頭とする10ビット(MR1010～MR1103)を、MR2004.Uの4ビット目を先頭とする10ビット(MR2008～MR2101)にブロック(ビット)転送します。



●他の関数を組み込んで使用する場合

スクリプト記述　**PMOV** (DM3000, EM3001, DM3100, EM3002, ANDA (EM3000, $7))

動作説明　毎スキャン(実行条件無し)、DM3000に格納された16ビットデータのEM3001で示されるビット位置以降のビットを、DM3001のEM3002で示されるビット位置以降のビットに、EM3000の下位3ビット(抽出)で示される転送ビット数分だけブロック(ビット)転送します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「ANDA関数」(7-22ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

MEMO

BYBMOV 連続バイト列情報を指定位置に転送(H→L)

バイBムーブ
BYBMOV ([実行条件,]※1 転送元バイト列先頭, 転送元バイトオフセット,,
転送先バイト列格納先先頭, 転送先バイトオフセット,転送バイト数)

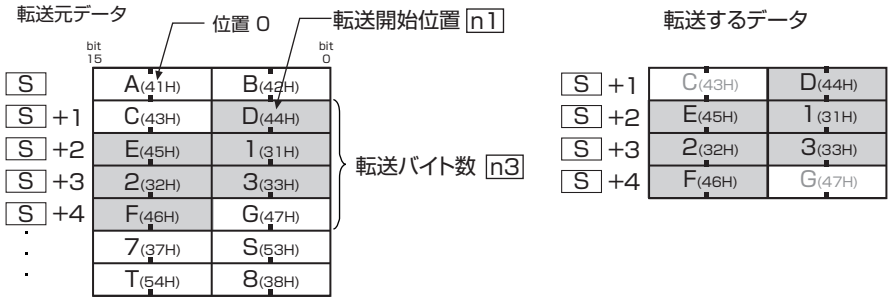
引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	転送元バイト列先頭※2	転送元バイト列データを格納している先頭を指定します。	.U	.S	.D	.L	.F	※3 .DF	.B	.T	—	○	—
n1	転送元バイトオフセット	転送元バイト列先頭からの位置の格納先、または位置を指定します(0～65535)。	.U	.U	.U	.U	—	—	—	—	○	○	○
D	転送先バイト列先頭※2	転送先バイト列の先頭を指定します。	.U	.S	.D	.L	.F	※3 .DF	.B	.T	—	○	—
n2	転送先バイトオフセット	転送先バイト列先頭からの位置の格納先、または位置を指定します(0～65535)。	.U	.U	.U	.U	—	—	—	—	○	○	○
n3	転送バイト数	転送するデータ数の格納先、または転送データ数を指定しますす(1～65535)。	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 • ビットデータではチャンネルの先頭を指定してください。
• S と D は同じ対象でも構いません。
※3 デバイスZに「.DF」サフィックスは指定できません。

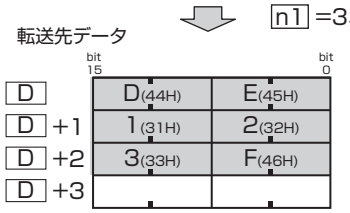
ポイント
- **BYBMOV関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニットで使用できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

実行条件が成り立つ(TRUE)とき、S の(上位バイトから数えて) n1 バイト目から n3 分を、D の(上位バイトから数えて) n2 バイト目から(n2 + n3 −1)バイト目に転送します。転送がおこなわれない部分のバイトは元の値が保存されます。



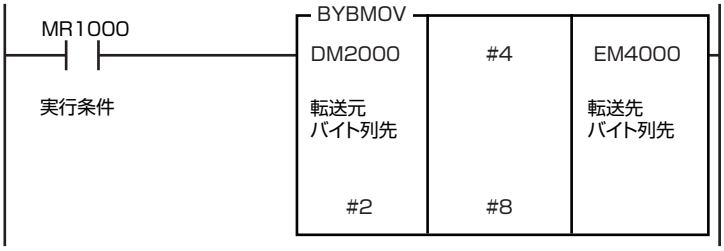
n1 =3、n2 =0 n3 =6 のとき、転送先は次のようになります。



「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ転送命令」を参照してください。

●書式例

BYBMOV (MR1000, DM2000, 4, EM4000, 2, 8)



ポイント　n3 =0の場合、関数は実行されません。

サンプルプログラム

スクリプト記述　**BYBMOV** (DM200, 4, DM400, 2, 8)

動作説明　DM200を先頭とする転送元バイト列データの(上位バイトから数えて)4バイト目から8バイト分を、DM400を先頭とする転送先バイト列データの(上位バイトから数えて)2～9バイトに転送します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」（2-10ページ）を参照してください。

BYLMOV 連続バイト列情報を指定位置に転送(L→H)

BYLMOV (バイLムーブ) **([**実行条件**,]**※1 転送元バイト列先頭**,** 転送元バイトオフセット**,,**
転送先バイト列格納先先頭**,** 転送先バイトオフセット**,**転送バイト数**)**

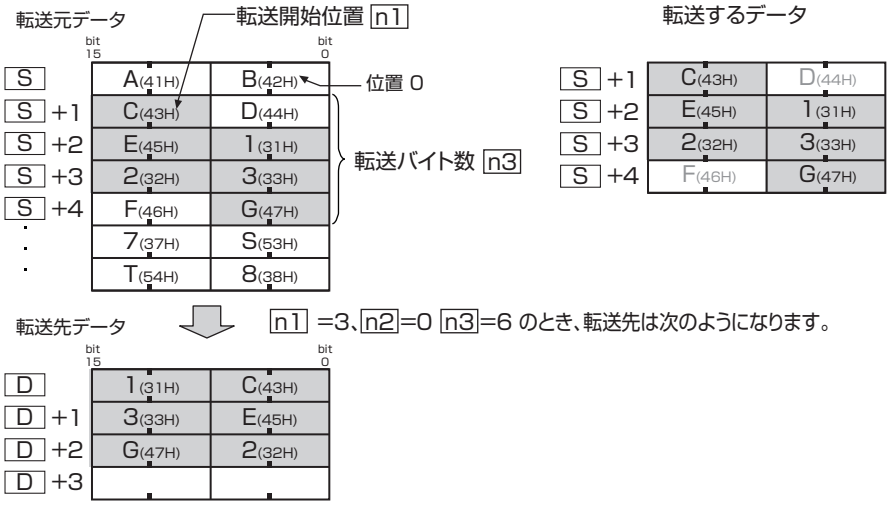
引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	転送元バイト列先頭※2	転送元バイト列データを格納している先頭を指定します。	.U	.S	.D	.L	.F	※3 .DF	.B	.T	—	○	—
n1	転送元バイトオフセット	転送元バイト列先頭からの位置の格納先、または位置を指定します(0～65535)。	.U	.U	.U	.U	—	—	—	—	○	○	○
D	転送先バイト列先頭※2	転送先バイト列の先頭を指定します。	.U	.S	.D	.L	.F	※3 .DF	.B	.T	—	○	—
n2	転送先バイトオフセット	転送先バイト列先頭からの位置の格納先、または位置を指定します(0～65535)。	.U	.U	.U	.U	—	—	—	—	○	○	○
n3	転送バイト数	転送するデータ数の格納先、または転送データ数を指定しますす(1～65535)。	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 • ビットデータではチャンネルの先頭を指定してください。
• S と D は同じ対象でも構いません。
※3 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **BYLMOV関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ使用できます。**
「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

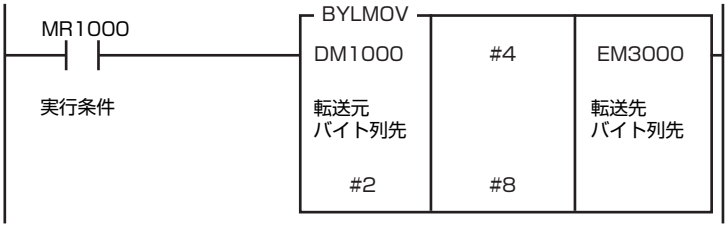
実行条件が成り立つ(TRUE)とき、S の(下位バイトから数えて)n1 バイト目から n3 分を、D の(下位バイトから数えて)n2 バイト目から(n2+n3−1)バイト目に転送します。転送がおこなわれない部分のバイトは元の値が保存されます。



「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ転送命令」を参照してください。

●書式例

BYLMOV (MR1000, DM1000, 4, EM3000, 2, 8)



ポイント　n3 **=0の場合、関数は実行されません。**

サンプルプログラム

スクリプト記述　**BYLMOV** (DM100, 4, DM200, 2, 8)

動作説明　DM100を先頭とする転送元バイト列データの(下位バイトから数えて)4バイト目から8バイト分を、DM200を先頭とする転送先バイト列データの(下位バイトから数えて)2～9バイトに転送します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」（2-10ページ）を参照してください。

INC　1を加算

インクリメントメモリ
INC ([実行条件,]※1 加算対象)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D 加算対象※2	インクリメント(1を加算)する対象を指定します。	.U	.S	.D	.L	—	—	—	—	—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1　[　]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2　ビットデータを指定した場合、連続した16/32ビットを扱います。
チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。

動作説明

D のデータに1を加算し、結果を D に格納します。

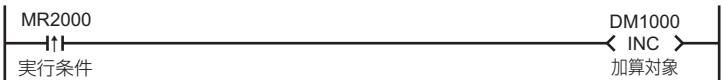
- **D :加算対象が16ビット符号無しデータ(.U)の場合**
  演算結果が65535より大きくなるとオーバーフローが発生します。
  (演算前が65535の場合、演算後は0となり、オーバーフローになります)
- **D :加算対象が16ビット符号付きデータ(.S)の場合**
  演算結果が＋32767より大きくなるとオーバーフローが発生します。
  (演算前が＋32767の場合、演算後は–32768となり、オーバーフローになります)
- **D :加算対象が32ビット符号無しデータ(.D)の場合**
  演算結果が4294967295より大きくなるとオーバーフローが発生します。
  (演算前が4294967295の場合、演算後は0となり、オーバーフローになります)
- **D :加算対象が32ビット符号付きデータ(.L)の場合**
  演算結果が＋2147483647より大きくなるとオーバーフローが発生します。
  (演算前が＋2147483647の場合、演算後は–2147483648となり、オーバーフローになります)

ポイント　**オーバーフローが発生しても正しく演算されます。**

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「算術/比較命令」を参照してください。

●書式例

INC (LDP (MR2000), DM1000.U)



サンプルプログラム

対象のデータに1を加算する場合、演算式を使う方法(DM1000 = DM1000 + 1)やINC関数を使う方法があります。
演算式を使う方法では、サフィックスに関係なく右辺は.Lとして演算されるため、精度落ちなどに注意する必要がありますが、INC関数を使う方法では、対象に指定したサフィックスで演算されるので精度落ちなどを意識せずに記述できます。

スクリプト記述

```
IF LDP ( MR1000 ) THEN
    DM1000 = DM1000 + 1
    INC ( DM1100 )
END IF
```

動作説明　MR1000の立ち上がり(OFF→ONに変化)で、DM1000、DM1100の現在値に1を加算します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

DEC　1を減算

デクリメントメモリ
DEC ([実行条件,]※1 減算対象)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D 減算対象※2	デクリメント(1を減算)する対象を指定します。	.U	.S	.D	.L	—	—	—	—	—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1　[　]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2　ビットデータを指定した場合、連続した16/32ビットを扱います。
チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。

動作説明

D のデータから1を減算し、結果を D に格納します。

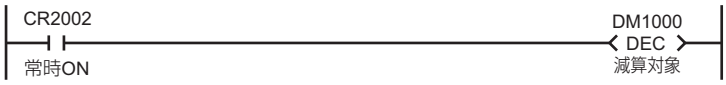
- **D :減算対象が16ビット符号無しデータ(.U)の場合**
  演算結果が0より小さくなるとアンダーフローが発生します。
  (演算前が0の場合、演算後は65535となり、アンダーフローになります)
- **D :減算対象が16ビット符号付きデータ(.S)の場合**
  演算結果が–32768より小さくなるとアンダーフローが発生します。
  (演算前が–32768の場合、演算後は＋32767となり、アンダーフローになります)
- **D :減算対象が32ビット符号無しデータ(.D)の場合**
  演算結果が0より小さくなるとアンダーフローが発生します。
  (演算前が0の場合、演算後は4294967295となり、アンダーフローになります)
- **D :減算対象が32ビット符号付きデータ(.L)の場合**
  演算結果が–2147483648より小さくなるとアンダーフローが発生します。
  (演算前が–2147483648の場合、演算後は＋2147483647となり、アンダーフローになります)

ポイント　**アンダーフローが発生しても正しく演算されます。**

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「算術/比較命令」を参照してください。

●書式例

DEC (DM1000.U)



サンプルプログラム

対象のデータに1を減算する場合、演算式を使う方法(DM1000 = DM1000 －1)やDEC関数を使う方法があります。
演算式を使う方法では、サフィックスに関係なく右辺は.Lとして演算されるため、精度落ちなどに注意する必要がありますが、DEC関数を使う方法では、対象に指定したサフィックスで演算されるので精度落ちなどを意識せずに記述できます。

スクリプト記述

```
IF LDP ( MR1000 ) THEN
    DM1000 = DM1000 - 1
    DEC ( DM1100 )
END IF
```

動作説明

MR1000の立ち上がり(OFF→ONに変化)で、DM1000、DM1100の現在値から1を減算します。

参 考

KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ROOT 平方根

戻り値※1 = **ROOT** (演算対象)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.U	.S	.D	.L	.F	※3 .DF	—	—	○	○	○
R 戻り値	ROOT関数は、引数と同じ型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

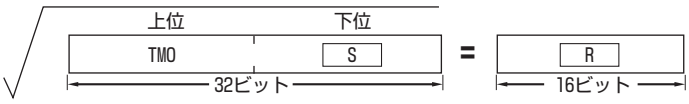
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

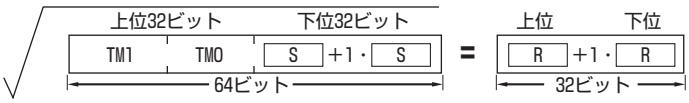
動作説明

S に格納されたデータの平方根を求め、結果を R に格納します。

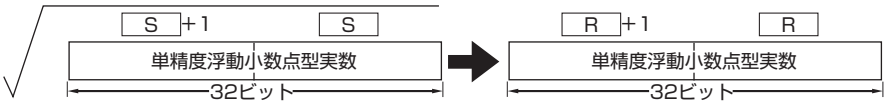
- **S :演算対象が16ビット(.U/.S)の場合**



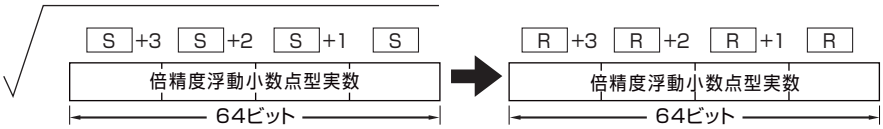
- **S :演算対象が32ビット(.D/.L)の場合**



- **S :演算対象が単精度浮動小数点型実数(.F)の場合**



- **S :演算対象が倍精度浮動小数点型実数(.DF)の場合**



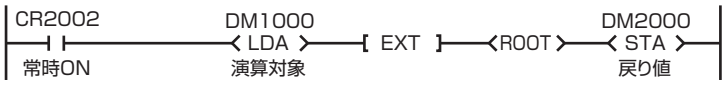
参 考　演算対象データに整数値を指定した場合：

- 変換対象データの16ビット/32ビットバイナリデータは、それぞれデータ拡張された結果の32ビット/64ビットバイナリデータとして平方根が求められ、結果はそれぞれ再度16ビット/32ビットバイナリデータとして格納されます。
- 演算結果の小数点以下の値は切り捨てられます。

「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「算術/比較命令」を参照してください。

●書式例

DM2000.U = ROOT (DM1000.U)



サンプルプログラム

●演算式を使用する場合

スクリプト記述　DM2000.L = **ROOT** (DM1000.S * EM1000.S)

動作説明　DM1000.SとEM1000.Sに格納されたデータの積(乗算)の平方根を求め、結果をDM2000.Lに格納します。

参 考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」（2-10ページ）を参照してください。

SQRT 平方根

戻り値※1 = **SQRT** (演算対象)
平方根

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.U	.S	.D	.L	.F	※3 .DF	—	—	○	○	○
R 戻り値	SQRT関数は、引数と同じ型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **SQRT関数は、KV STUDIO Ver.4.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

本関数は「ROOT関数」と関数名が異なるだけで、動作等に違いはありません。

「ROOT関数」(7-18ページ)

MEMO

ANDA 論理積演算

戻り値※1 = **ANDA** (演算対象, 演算データ)
アンドA

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 演算対象※2	演算対象、または演算対象データを指定します。	.U	.U	.D	.D	—	—	—	—	○	○	○
S2 演算データ※2	論理積データの格納先、または論理積データを指定します。	S1 と同じ型として扱われます。				—	—	—	—	○	○	○
R 戻り値	ANDA関数は、引数と同じ型を持ちます。※3	.U	.S	.D	.L	.F	※4 .DF	—	—	—	○	—

※1 戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2 ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。
※3 戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※4 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

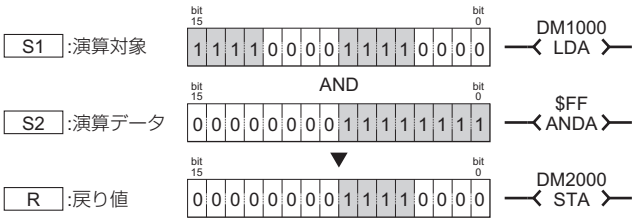
動作説明

S1 に格納されたデータと、S2 に格納されたバイナリデータで論理積演算をおこないます。

●書式例

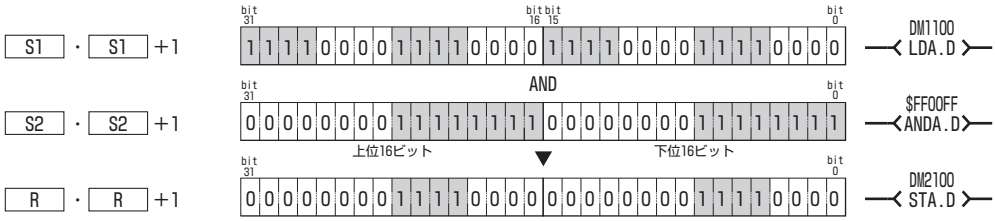
- **S1 :演算対象が16ビット(.U/.S)の場合**

DM2000.U = **ANDA** (DM1000.U, $00FF)



• S1 :演算対象が32ビット(.D/.L)の場合

DM2100.D = **ANDA** (DM1100.D, $00FF00FF)



「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「論理演算命令」を参照してください。

● 論理積演算真理値表

演算対象	0	0	1	1
演算データ	0	1	0	1
戻り値	0	0	0	1

サンプルプログラム

● ビットデータの論理積を求める場合

スクリプト記述　DM2000.D = **ANDA** (MR1000.D, DM1100.D)

動作説明　MR1000.Dに格納されたデータと、DM1100.Dに格納されたデータで論理積をおこないます。
結果をDM2000.Dに格納します。

● 他の関数に組み込んで使用する場合

スクリプト記述　FMOV (**ANDA** (DM1000, $FF), DM2000, 10)

動作説明　DM1000に格納されたデータと演算データ$FF(16進)で論理積をおこないます。
演算結果のデータをDM2000を先頭とする10ワードに同一データでブロック転送します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「FMOV関数」(7-4ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ORA 論理和演算

戻り値※1 = **ORA** (演算対象, 演算データ)

オアA

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 演算対象※2	演算対象、または演算対象データを指定します。	.U	.U	.D	.D	—	—	—	—	○	○	○
S2 演算データ※2	論理和データの格納先、または論理和データを指定します。	S1 と同じ型として扱われます。				—	—	—	—	○	○	○
R 戻り値	ORA関数は、引数と同じ型を持ちます。※3	.U	.S	.D	.L	.F	※4 .DF	—	—	—	○	—

※1 戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2 ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。
※3 戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※4 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

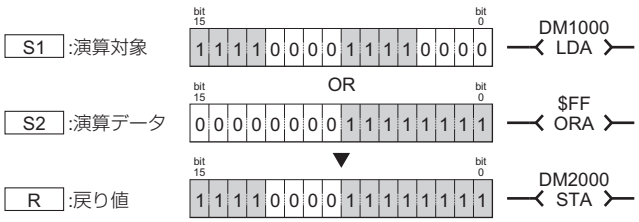
動作説明

S1 に格納されたデータと、S2 に格納されたバイナリデータで論理和演算をおこないます。

●書式例

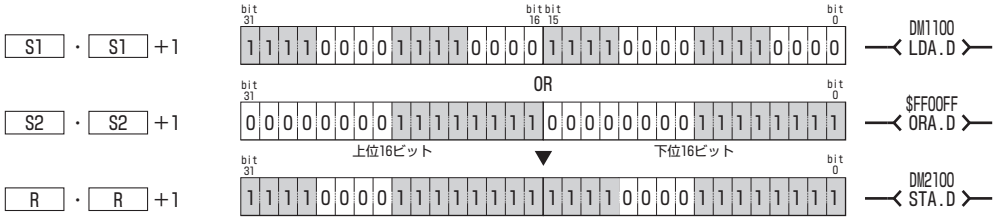
- **S1 :演算対象が16ビット(.U/.S)の場合**

DM2000.U = **ORA** (DM1000.U, $00FF)



• S1 :演算対象が32ビット(.D/.L)の場合

DM2100.D = **ORA** (DM1100.D, $00FF00FF)



「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「論理演算命令」を参照してください。

●論理和演算真理値表

演算対象	0	0	1	1
演算データ	0	1	0	1
戻り値	0	1	1	1

サンプルプログラム

●ビットデータの論理和を求める場合

スクリプト記述　DM2000.D = **ORA** (MR1000.D, DM1100.D)

動作説明　MR1000.Dに格納されたデータと、DM1100.Dに格納されたデータで論理和をおこないます。
結果をDM2000.Dに格納します。

●他の関数に組み込んで使用する場合

スクリプト記述　R5000.U = **ORA** (TBCD (DM1000), $1111)

動作説明　DM1000に格納されたデータを4桁BCDデータに変換します。変換結果の4桁BCDデータと$1111(16進)で論理和(4桁BCDの各桁が偶数の時は奇数(+1)に変換)をおこない、演算結果のデータを、出力リレーR5000～R5015に出力します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「TBCD関数」(7-80ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

EORA 排他的論理和演算

戻り値※1 = **EORA** (演算対象, 演算データ)
エクスクルーシブオアA

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 演算対象※2	演算対象、または演算対象データを指定します。	.U	.U	.D	.D	—	—	—	—	○	○	○
S2 演算データ※2	排他的論理和データの格納先、または排他的論理和データを指定します。	S1と同じ型として扱われます。				—	—	—	—	○	○	○
R 戻り値	EORA関数は、引数と同じ型を持ちます。※3	.U	.S	.D	.L	.F	※4 .DF	—	—	—	○	—

※1 戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2 ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。
※3 戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※4 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

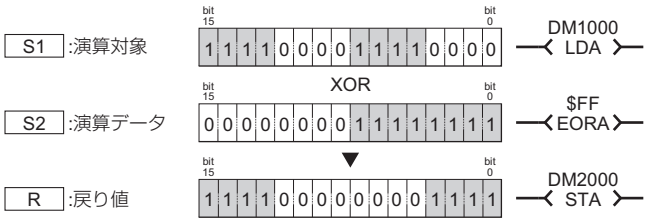
動作説明

S1 に格納されたデータと、S2 に格納されたバイナリデータで排他的論理和演算をおこないます。

●書式例

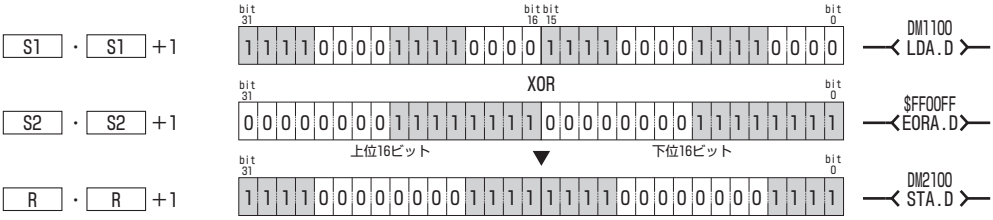
- **S1 :演算対象が16ビット(.U/.S)の場合**

DM2000.U = **EORA** (DM1000.U, $00FF)



- S1 :演算対象が32ビット(.D/.L)の場合

DM2100.D = **EORA** (DM1100.D, $00FF00FF)



「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「論理演算命令」を参照してください。

●排他的論理和演算真理値表

演算対象	0	0	1	1
演算データ	0	1	0	1
戻り値	0	1	1	0

サンプルプログラム

●ビットデータの排他的論理和を求める場合

スクリプト記述　DM2000.D = **EORA** (MR1000.D, DM1100.D)

動作説明　MR1000.Dに格納されたデータと、DM1100.Dに格納されたデータで排他的論理和をおこないます。
結果をDM2000.Dに格納します。

●使用例

EORA関数は、1(ON)に指定した任意のビットに対してのみ論理反転できるため、演算対象で指定されたデータを、演算データで指定したビットについてのみONとOFFを反転させる場合などに用いられます。

スクリプト記述　R5000.U = **EORA** (R5000.U, $00FF)

動作説明　出力リレーR5000～R5015について、現在のR5000～R5007の出力状態(ON/OFF)を、スキャン毎に反転(交互にON/OFF)して出力します。(R5008～R5015は変化しません

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ENRA 排他的論理和否定演算

戻り値※1 = **EORA** (演算対象, 演算データ)
エクスクルーシブオアA

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 演算対象※2	演算対象、または演算対象データを指定します。	.U	.U	.D	.D	—	—	—	—	○	○	○
S2 演算データ※2	排他的論理和否定データの格納先、または排他的論理和否定データを指定します。	S1 と同じ型として扱われます。				—	—	—	—	○	○	○
R 戻り値	EORA関数は、引数と同じ型を持ちます。※3	.U	.S	.D	.L	.F	※4 .DF	—	—	—	○	—

※1 戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2 ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。
※3 戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※4 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

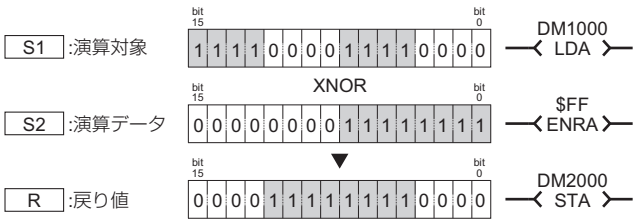
動作説明

S1 に格納されたデータと、S2 に格納されたバイナリデータで排他的論理和否定演算をおこない、結果を R に格納します。

●書式例

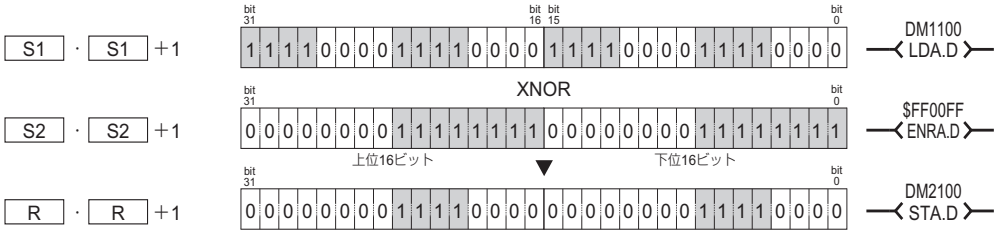
- **S1 :演算対象が16ビット(.U/.S)の場合**

DM2000.U = **ENRA** (DM1000.U, $00FF)



• S1 :演算対象が32ビット(.D/.L)の場合

DM2100.D = **ENRA** (DM1100.D, $00FF00FF)



「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「論理演算命令」を参照してください。

●排他的論理和否定演算真理値表

演算対象	0	0	1	1
演算データ	0	1	0	1
戻り値	1	0	0	1

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　DM2000 = BCNT (**ENRA** (MR1000.U, MR1100.U))

動作説明　MR1000.UとMR1100.Uに格納されている16ビットバイナリデータの一致しているビットの数を求めて、結果をDM2000に格納します。

参考　組み込む関数の引数によっては、使用できない場合がありますので注意してください。

「BCNT関数」(8-6ページ)

●使用例

ENRA関数は、ON:ON、またはOFF:OFFの一致したビットが抽出できるため、前回値と今回値でビットの変化を捉える場合などに用いられます。

スクリプト記述　R1000.U = **ENRA** (R2000.U, MR2000.U)

動作説明　現在のR2000～R2015の入力状態(ON/OFF)で、MR2000に格納されている前回の入力状態と一致しているビットを抽出し、出力リレーR1000～R1015の対応するビットをONにします。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

COM　ビット反転

戻り値※1 = **COM** (演算対象)
コンプリメント

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.U	.U	.D	.D	—	—	—	—	○	○	○
R 戻り値	COM関数は、引数と同じ型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

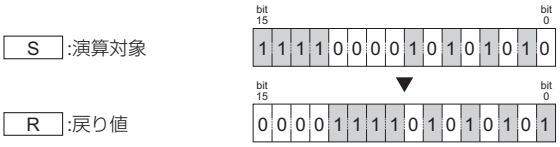
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

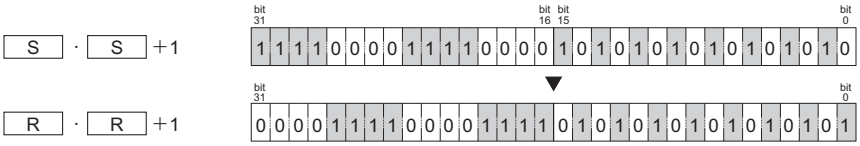
動作説明

S に格納されたバイナリデータの、各ビットを反転します。

- **S :演算対象が16ビットの場合**



- **S :演算対象が32ビットの場合**



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「論理演算命令」を参照してください。

●書式例

DM2000.U = **COM** (DM1000.U)

サンプルプログラム

●使用例

COM関数は、演算対象で指定されたバイナリデータを論理反転できるため、現在のビット状態のONとOFFを反転させる場合などに用いられます。

スクリプト記述　R5000.U = **COM** (R5000.U)

動作説明　出力リレーR5000～R5015について、現在の出力状態を、スキャン毎に反転(交互にON/OFF)して出力します。

参考

- 上記スクリプトは、
  R5000.U = **EORA** (R5000.U, $FFFF)
  と同等です。
  「EORA関数」(7-26ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

NEG 符号反転

戻り値※1 = **NEG** (演算対象)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.U	.U	.D	.D	.F	※3 .DF	—	—	○	○	○
R 戻り値	NEG関数は、引数と同じ型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

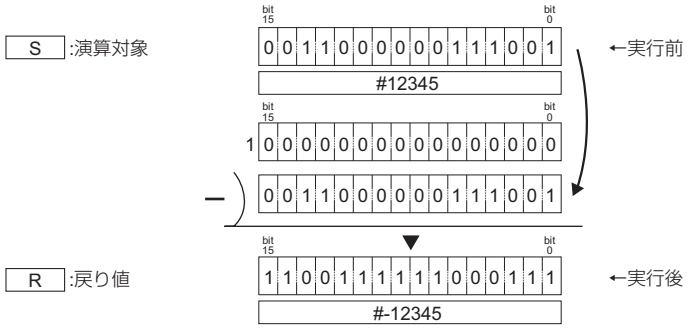
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

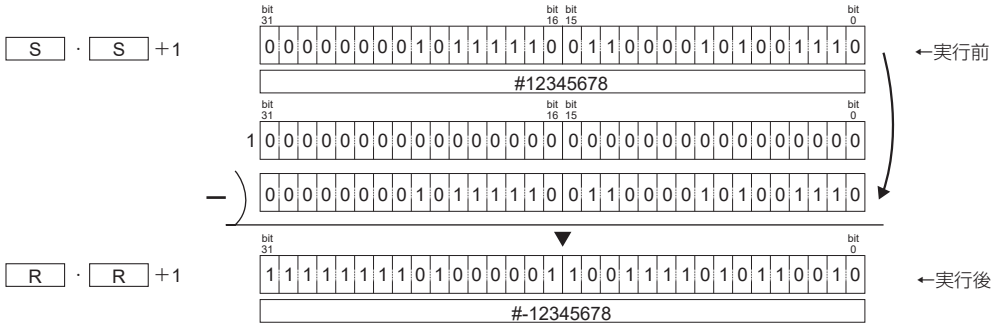
動作説明

S に格納されたバイナリデータの、正負の符号を反転(2の補数演算)します。

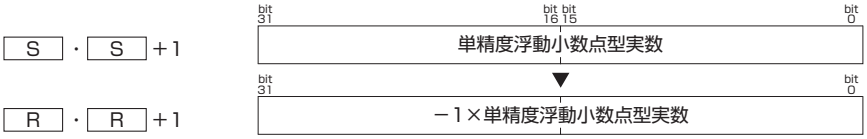
- **S :演算対象が16ビットの場合**



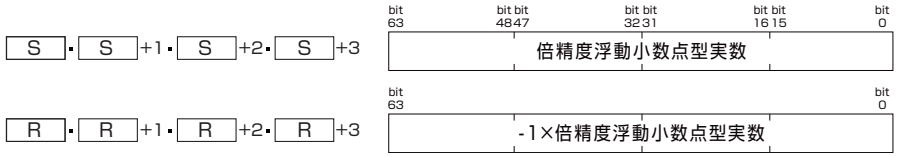
- **S :演算対象が32ビットの場合**



- S :演算対象が単精度浮動小数点型実数の場合



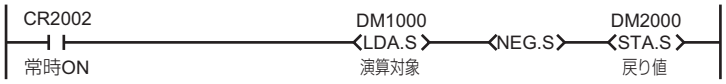
- S :演算対象が倍精度浮動小数点型実数の場合



詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「論理演算命令」を参照してください。

●書式例

DM2000.S = **NEG** (DM1000.S)



サンプルプログラム

●ビットデータを符号反転する場合

スクリプト記述　DM2000.U = **NEG** (MR1000.U)

動作説明　MR1000に格納されたバイナリデータの、正負の符号を反転します。
結果をDM2000に格納します。

参 考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ワンポイント

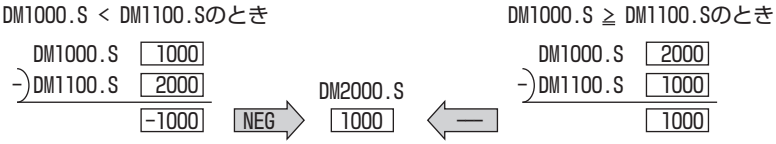
NEG関数は、演算対象で指定されたバイナリデータの正負の符号を反転できるため、負の値を符号反転して絶対値を求める場合などに用いられます。

スクリプト記述

```
IF DM1000.S < DM1100.S THEN
DM2000.S = NEG ( TOS ( DM1000.S – DM1100.S ) )'演算結果を符号反転
ELSE
DM2000.S = TOS ( DM1000.S – DM1100.S )
END IF
```

動作説明

DM1000.S–DM1100.Sの演算結果の絶対値をDM2000に格納します。



ポイント

16ビット符号付きデータ(.S)に対してNEG関数を用いる場合、扱える範囲は–32768～32767のため、–32768をNEG関数で演算した場合、結果は –32768になります。このとき、TOL関数を用いて32ビット符号付きデータ(.L)に拡張してから符号反転させることで+32768を求めることが可能です。
また、NEG関数を使う代わりに対象の直前にマイナス演算子(–)をつけると式として扱われますので、自動的に32ビット符号付きデータ(.L)に拡張され、–32768が符号反転され+32768が求められます。

MEMO

ABS 絶対値

戻り値※1 = **ABS** (演算対象)
アブソリュート

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.U	.S	.D	.L	.F	※3 .DF	—	—	○	○	○
R 戻り値	ABS関数は、引数と同じ型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

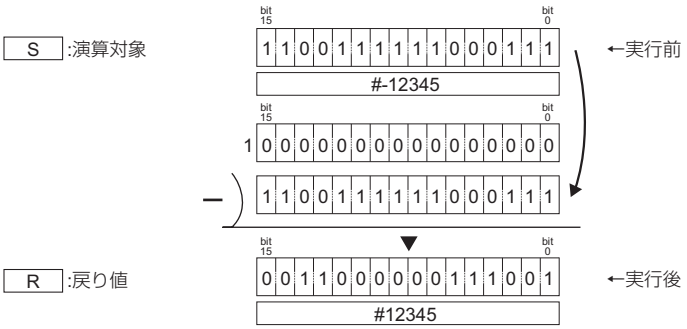
ポイント
- **ABS関数は、KV STUDIO Ver.4.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

S に格納されたバイナリデータの、絶対値を R に格納します。

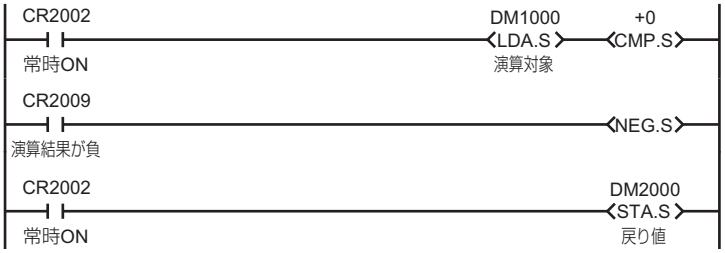
- **S :演算対象が、符号無し16ビット/32ビットデータ (.U/.D) の場合**
  S は、無処理のまま R に格納されます。

- **S :演算対象が、符号付き16ビット/32ビットデータ (.S/.L) または単精度/倍精度浮動小数点型実数データ (.F/.DF) の場合**
  S ≧0の場合、無処理のまま R に格納されます。
  S ＜0の場合、符号を反転(2の補数演算)してから R に格納されます。



●書式例

DM2000.S = **ABS** (DM1000.S)



SRA 右シフト(ビット単位)

戻り値※1 = **SRA** (演算対象, 右シフト数)
シフトライトA

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.U	.U	.D	.D	—	—	—	—	○	○	○
n 右シフト数	右シフト数の格納先、または右シフト数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	SRA関数は、第1引数と同じ型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

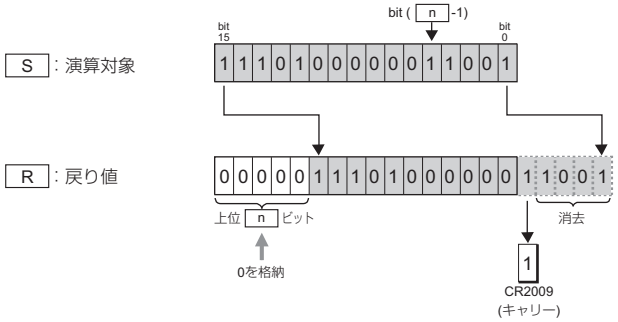
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

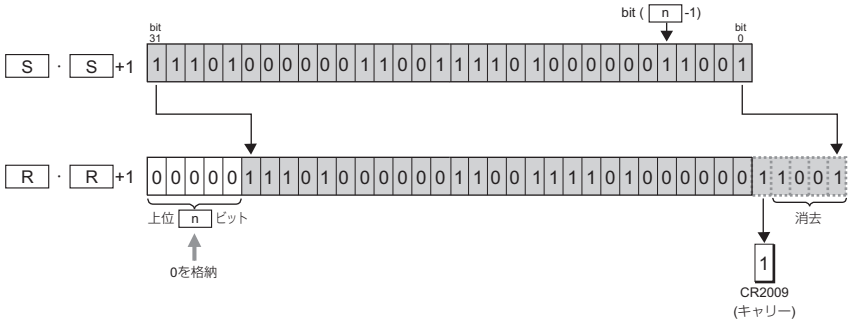
動作説明

S に格納されたデータを、n で指定したビット数分右へシフトし、上位ビットからの右シフトしたビット数分には0を格納します。

- **S :演算対象が16ビットの場合**



- **S :演算対象が32ビットの場合**



「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データシフト命令」を参照してください。

●書式例

DM2000.U = SRA (DM1000.U, 5)

ポイント
- **n が0の場合、R には S の値が格納されます。**
- **S が.Uで n ≧17、S が.Dで n ≧33の場合、演算結果は0になります。**

サンプルプログラム

●ビットデータを右シフトする場合

スクリプト記述　DM2000.U = **SRA** (R1000.U, 10)

動作説明　R1000.Uに格納されているデータを、10ビット分右にシフトします。
結果をDM2000.Uに格納します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」（2-10ページ）を参照してください。

SLA　左シフト(ビット単位)

戻り値※1 = **SLA** (演算対象, 左シフト数)
シフトレフトA

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.U	.U	.D	.D	—	—	—	—	○	○	○
n 左シフト数	左シフト数の格納先、または左シフト数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	SLA関数は、第1引数と同じ型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

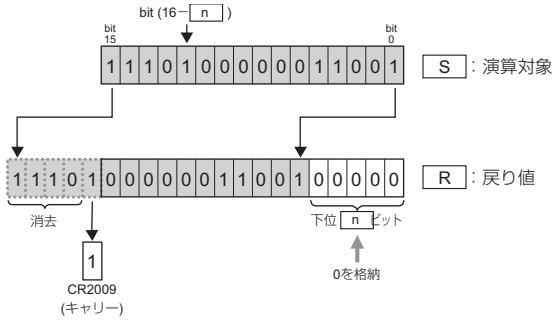
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

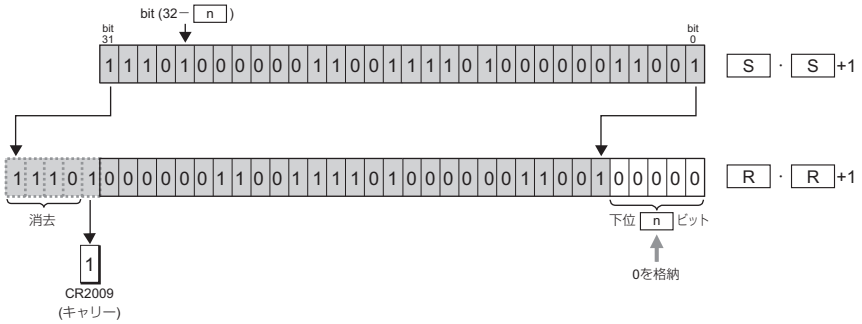
動作説明

S に格納されたデータを、n で指定したビット数分左へシフトし、下位ビットからの左シフトしたビット数分には0を格納します。

- **S :演算対象が16ビットの場合**



- **S :演算対象が32ビットの場合**



「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データシフト命令」を参照してください。

●書式例

DM2000.U = **SLA** (DM1000.U, 5)

ポイント
- **n が0の場合、R には S の値が格納されます。**
- **S が.Uで n ≧17、S が.Dで n ≧33の場合、演算結果は0になります。**

サンプルプログラム

●ビットデータを左シフトする場合

スクリプト記述　DM2000.U = **SLA** (MR1000.U, 10)

動作説明　MR1000.Uに格納されているデータを、10ビット分左にシフトします。
結果をDM2000に格納します。

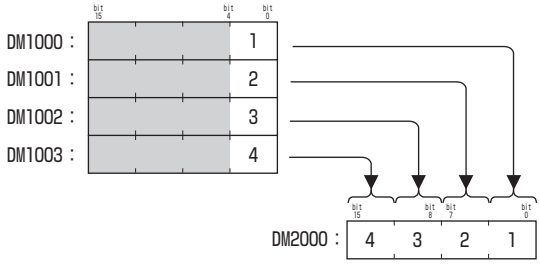
参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ワンポイント

SLA関数は、任意のビット数を左にシフトできるため、ワードデータなどに格納されたデータの分離、結合などに用いられます。

スクリプト記述
DM2000 = ANDA (DM1000, $F)
DM2000 = ORA (DM1000, **SLA** (ANDA (DM1001, $F), 4))
DM2000 = ORA (DM1000, **SLA** (ANDA (DM1002, $F), 8))
DM2000 = ORA (DM1000, **SLA** (DM1003, 12))

動作説明　下位から1桁ずつDM1000、DM1001、DM1002、DM1003に格納されたBCDデータを、4桁のBCDとして結合し、DM2000に格納します。



参考　上記スクリプトは、データ変換命令UNIN(ニブル結合)と同等の機能です。
「ANDA関数」(7-22ページ)
「ORA関数」(7-24ページ)
「UNIN関数」(7-94ページ)
『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ変換命令」「UNIN命令」

ASRA 算術右シフト(ビット単位)

戻り値※1 = **ASRA** (演算対象, 右シフト数)
算術シフトライトA

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.U	.S	.D	.L	—	—	—	—	○	○	○
n 算術右シフト数	右シフト数の格納先、または右シフト数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	ASRA関数は、第1引数と同じ型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

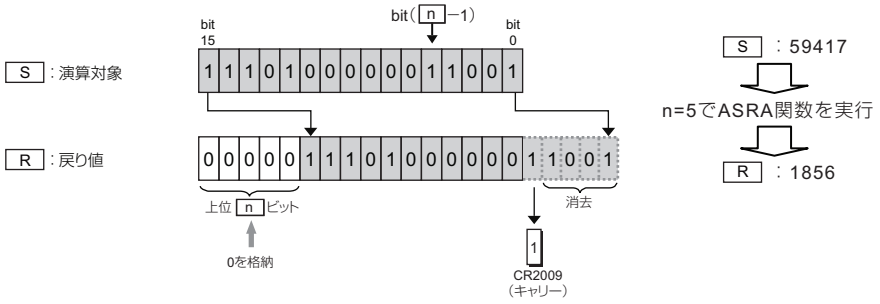
※1 戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2 戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **ASRA関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000 シリーズ CPU ユニット、CPU 機能バージョン 2.0 以降の KV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ使用できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

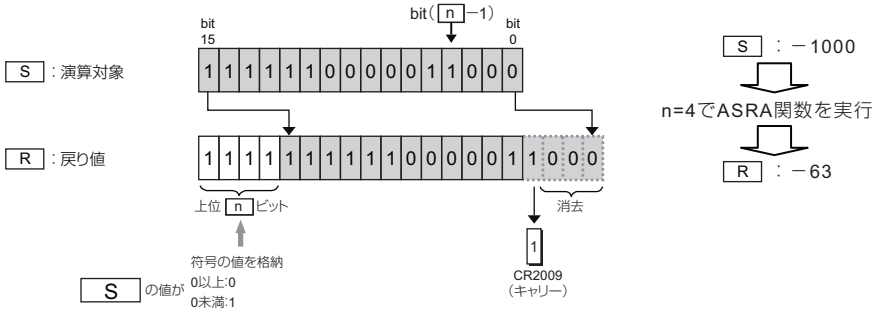
動作説明

S に格納されたデータを、n で指定したビット数分右へシフトし、上位ビットからの右シフトしたビット数分には S ≧0のとき「0」が格納され、S ＜0のとき「1」が格納されます。

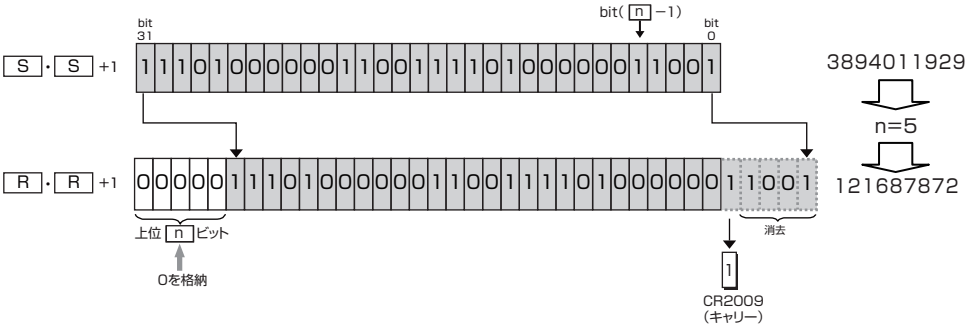
- **S :演算対象が符号無し16ビットデータの場合**



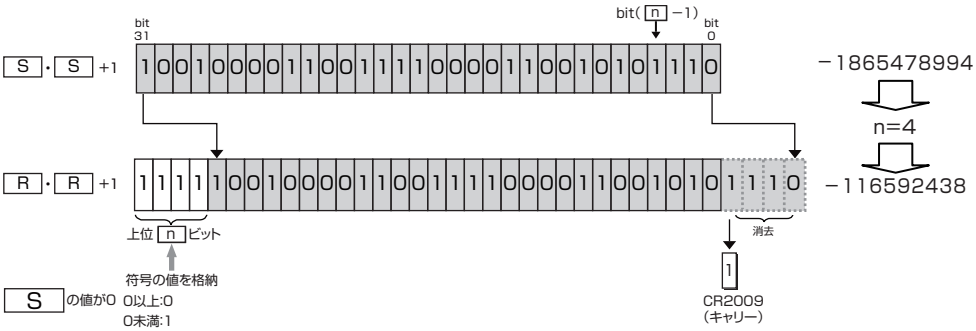
- **S :演算対象が符号付き16ビットデータの場合**



- S :演算対象が符号無し32ビットデータの場合



- S :演算対象が符号付き32ビットデータの場合



「演算フラグについて(CR2009〜CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データシフト命令」を参照してください。

●書式例

DM2000.U = ASRA (DM1000.U, 5)

ポイント
- **n が0の場合、R には S の値が格納されます。**
- **S が.Uで n ≧17、S が.Dで n ≧33の場合、演算結果は0になります。**

サンプルプログラム

スクリプト記述　DM2000.U = **ASRA** (R1000.U, 10)

動作説明　R1000.Uに格納されているデータを、10ビット分右にシフトします。
結果をDM2000.Uに格納します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ASLA 算術左シフト(ビット単位)

戻り値※1 = **ASLA** (演算対象, 左シフト数)
算術シフトレフトA

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.U	.S	.D	.L	—	—	—	—	○	○	○
n 算術左シフト数	左シフト数の格納先、または左シフト数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	ASLA関数は、第1引数と同じ型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

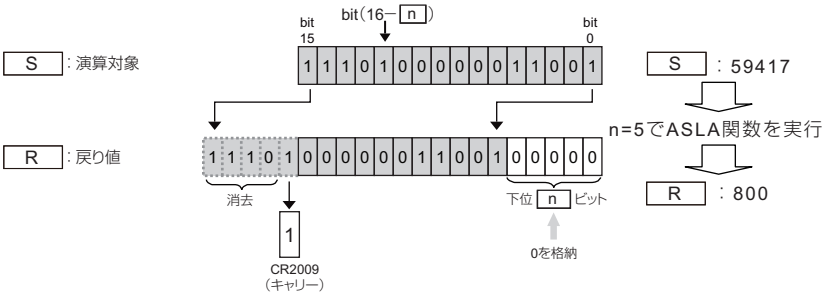
※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **ASLA関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000 シリーズ CPU ユニット、CPU 機能バージョン 2.0 以降の KV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ使用できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

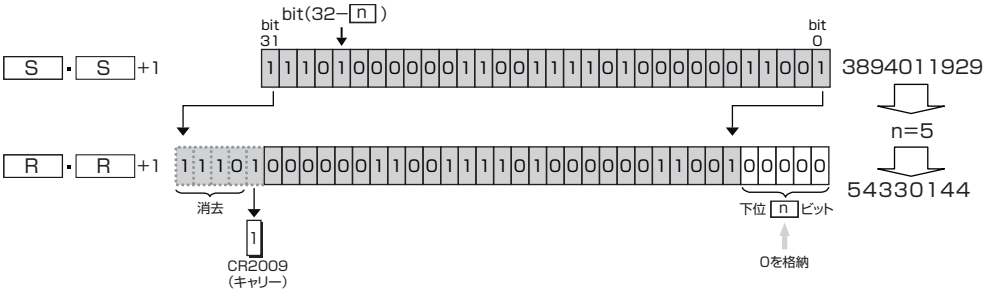
動作説明

S に格納されたデータを、n で指定したビット数分左へシフトし、下位ビットからの左シフトしたビット数分には0が格納されます。

- **S :演算対象が16ビットデータの場合**



- **S :演算対象が32ビットデータの場合**



「演算フラグについて(CR2009〜CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データシフト命令」を参照してください。

●書式例

DM2000.U = **ASLA** (MR1000.U, 5)

ポイント

- **[n]が0の場合、[R]には[S]の値が格納されます。**
- **[S]が.Uで[n]≧17、[S]が.Dで[n]≧33の場合、演算結果は0になります。**

サンプルプログラム

スクリプト記述　DM2000.U = **ASLA** (R1000.U, 10)

動作説明　R1000.Uに格納されているデータを、10ビット分左にシフトします。
結果をDM2000.Uに格納します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

RRA 右ローテート(ビット単位キャリ付)

戻り値※1 = **RRA** (演算対象, 右ローテート数)

ローテートライトA

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.U	.U	.D	.D	—	—	—	—	○	○	○
n 右ローテート数	右ローテート数の格納先、または右ローテート数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	RRA関数は、第1引数と同じ型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

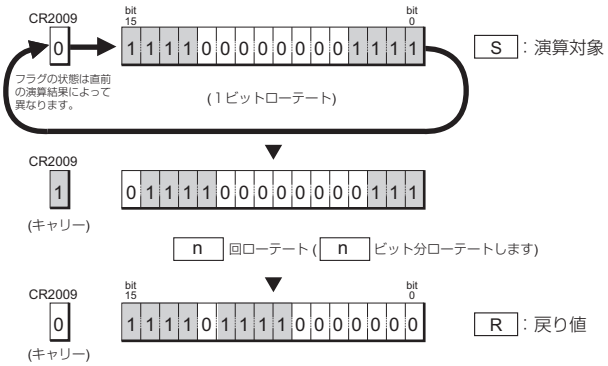
※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

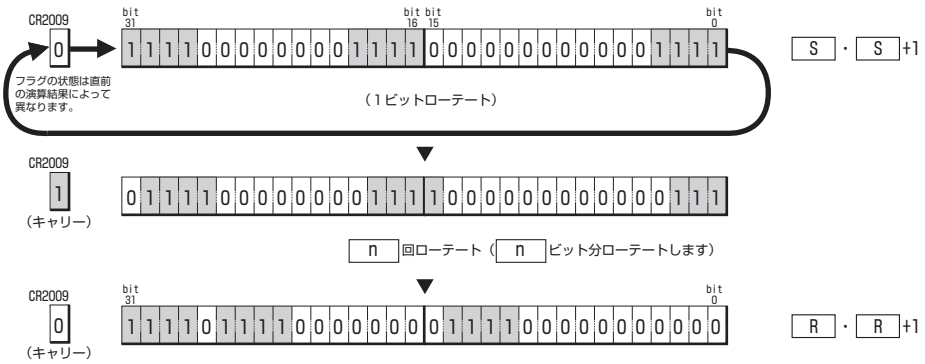
動作説明

S に格納されたデータを、n で指定したビット数分キャリーフラグ(CR2009)を含めて右へローテートします。

- **S :演算対象が16ビットの場合**



- **S :演算対象が32ビットの場合**



「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データシフト命令」を参照してください。

●書式例

DM2000.U = **RRA** (DM1000.U, 10)

ポイント　**右へローテートしたとき、最下位ビットはキャリーフラグ(CR2009)に格納されます。**

サンプルプログラム

●ビットデータを右ローテートする場合

スクリプト記述　DM2000 = **RRA** (R1000.U, 10)

動作説明　R1000.Uに格納されているデータを、10ビット分右にローテートします。
結果をDM2000に格納します。

ワンポイント

SRA関数はシフト毎に最上位ビットが0になっていきますが、RRA関数の場合は最下位ビットが、キャリーフラグを経由して最上位ビットへ回り込んでいきます。

スクリプト記述　R5000.D = **RRA** (R5000.D, 1)

動作説明　出力リレーR5000～R5115について、現在の出力状態をスキャン毎に1ビット分右ローテートしながら、繰り返し流れるように出力します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」（2-10ページ）を参照してください。

RLA 左ローテート(ビット単位キャリ付)

戻り値※1 = **RLA** (演算対象, 左ローテート数)
ローテートレフトA

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.U	.U	.D	.D	—	—	—	—	○	○	○
n 左ローテート数	左ローテート数の格納先、または左ローテート数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	RLA関数は、第1引数と同じ型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

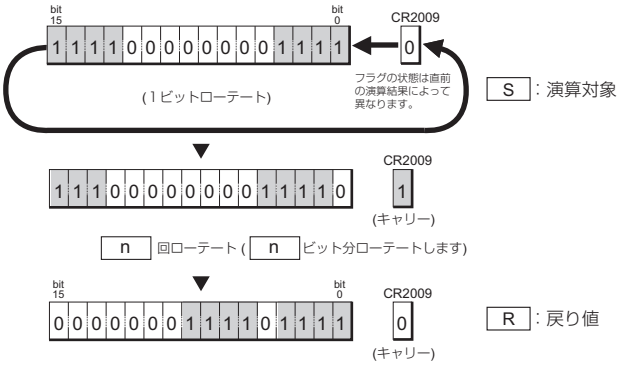
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

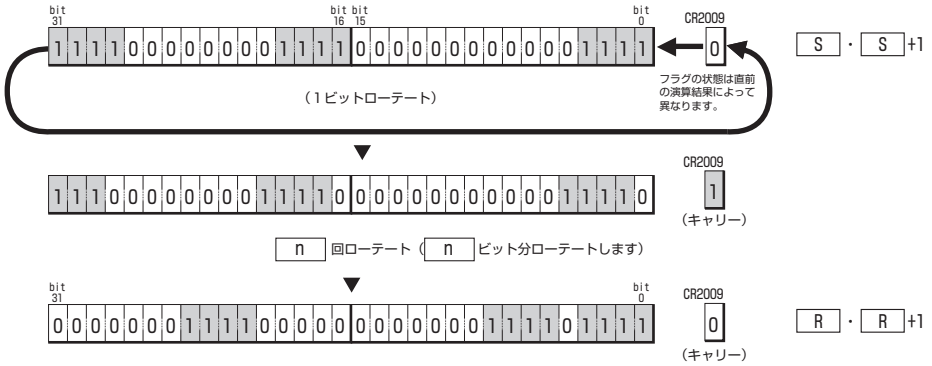
動作説明

S に格納されたデータを、n で指定したビット数分キャリーフラグ(CR2009)を含めて左へローテートします。

- **S :演算対象が16ビットの場合**



- S :演算対象が32ビットの場合



「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データシフト命令」を参照してください。

●書式例

DM2000.U = **RLA** (DM1000.U, 10)

ポイント **左へローテートしたとき、最上位ビットはキャリーフラグ(CR2009)に格納されます。**

サンプルプログラム

●ビットデータを左ローテートする場合

スクリプト記述 DM2000.U = **RLA** (R1000.U, 10)

動作説明 R1000.Uに格納されているデータを、10ビット分左にローテートします。
結果をDM2000に格納します。

ワンポイント

SLA関数はシフト毎に最下位ビットが0になっていきますが、RLA関数の場合は最上位ビットが、キャリーフラグを経由して最下位ビットへ回り込んでいきます。

スクリプト記述 R5000.D = **RLA** (R5000.D, 1)

動作説明 出力リレーR5000～R5031について、現在の出力状態をスキャン毎に1ビット分左ローテートしながら、繰り返し流れるように出力します。

参考 KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」（2-10ページ）を参照してください。

RRNCA 右ローテート(ビット単位キャリ無)

キャリーなしローテートライトA
戻り値※1 = **RRNCA** (演算対象, 右ローテート数)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象の格納先を指定します。	.U	.U	.D	.D	—	—	—	—	○	○	○
n 右ローテート数※2	右ローテート数、またはその格納先を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	RRNCA関数は、第1引数と同じ型を持ちます。※3	.U	.S	.D	.L	.F	※4 .DF	—	—	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果は内部レジスタに格納されます。
※2 n に対象を指定した場合、n の値が16以上(演算対象が.Uのとき)/32以上(演算対象が.Dのとき)のときは以下のようになります。
.Uのとき:n の値を16で割った余りの値をローテートするビット数とします。
.Dのとき:n の値を32で割った余りの値をローテートするビット数とします。
※3 戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※4 デバイスZに「.DF」サフィックスは指定できません。

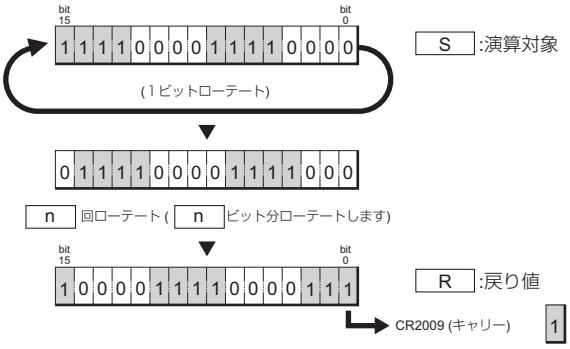
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

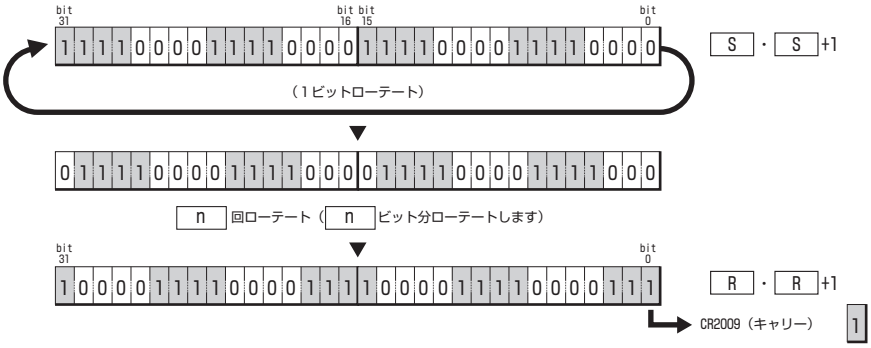
動作説明

S に格納されたデータを、n で指定したビット数分、キャリーを含まずに右へローテートし、結果を R に格納します。

- **S :演算対象が16ビットの場合**



- S :演算対象が32ビットの場合



「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データシフト命令」を参照してください。

参考　ローテート結果の最下位ビットの状態がキャリーフラグ(CR2009)に格納されます。

●書式例

DM2000.U = **RRNCA** (DM1000.U, 10)

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　DM2100.D = TBIN (**RRNCA** (DM1100.D, 16))

動作説明　DM1100.Dに格納されている8桁BCDデータを、16ビット右へキャリーを含まずローテートして、上位･下位ワード(8桁BCDの上位4桁、下位4桁)を入れ換え、結果を32ビットバイナリデータに変換してDM2000.Dに格納します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「TBIN関数」（7-82ページ）
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」（2-10ページ）を参照してください。

RLNCA　左ローテート(ビット単位キャリ無)

キャリーなしローテートレフトA
戻り値※1 = **RLNCA** (演算対象, 左ローテート数)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象の格納先を指定します。	.U	.U	.D	.D	—	—	—	—	○	○	○
n 左ローテート数※2	左ローテート数、またはその格納先を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	RLNCA関数は、第1引数と同じ型を持ちます。※3	.U	.S	.D	.L	.F	※4 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、変換結果は内部レジスタに格納されます。
※2　n に対象を指定した場合、n の値が16以上(演算対象が.Uのとき)/32以上(演算対象が.Dのとき)のときは以下のようになります。
.Uのとき: n の値を16で割った余りの値をローテートするビット数とします。
.Dのとき: n の値を32で割った余りの値をローテートするビット数とします。
※3　戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※4　デバイスZに「.DF」サフィックスは指定できません。

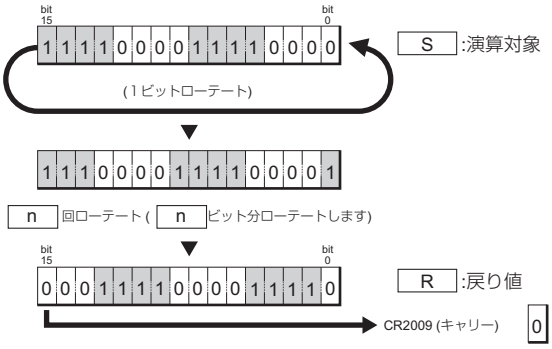
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

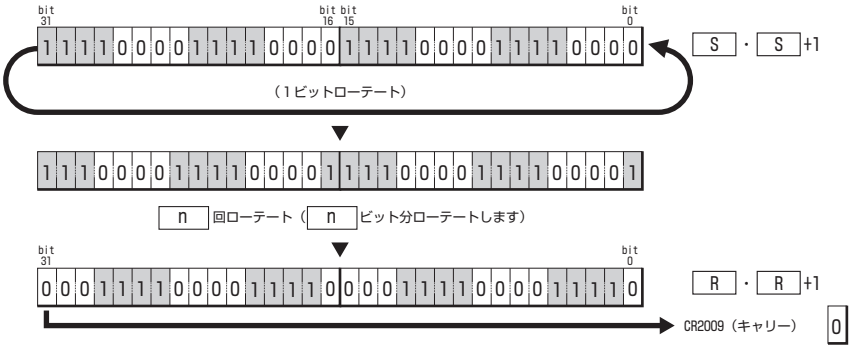
動作説明

S に格納されたデータを、n で指定したビット数分、キャリーを含まずに左へローテートし、結果を R に格納します。

- **S :演算対象が16ビットの場合**



- S :演算対象が32ビットの場合



「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データシフト命令」を参照してください。

参 考　ローテート結果の最上位ビットの状態がキャリーフラグ(CR2009)に格納されます。

●書式例

DM2000.U = **RLNCA** (DM1000.U, 10)

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　DM2100.D = TBIN (**RLNCA** (DM1100.D, 16))

動作説明　DM1100.Dに格納されている8桁BCDデータを、16ビット左へキャリーを含まずローテートして、上位･下位ワード(8桁BCDの上位4桁、下位4桁)を入れ換え、結果を32ビットバイナリデータに変換してDM2000.Dに格納します。

参 考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「TBIN関数」（7-82ページ）
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」（2-10ページ）を参照してください。

WSR　ワードデバイス右シフト

ワードシフトライト
WSR ([実行条件,]※1 演算対象, 演算範囲データ数, シフト数)

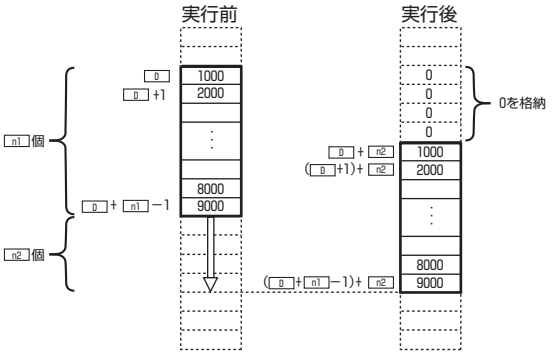
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D 演算対象※2	演算対象、または演算対象データを指定します。	.U	.U	.D	.D	.D	—	—	—	—	○	—
n1 演算範囲データ数	右シフトする範囲のデータ数の格納先、または右シフト範囲データ数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
n2 シフト数	右シフト数の格納先、または右シフト数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1　[　]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2　ビットデータを指定した場合、連続した16/32ビットを扱います。
チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。

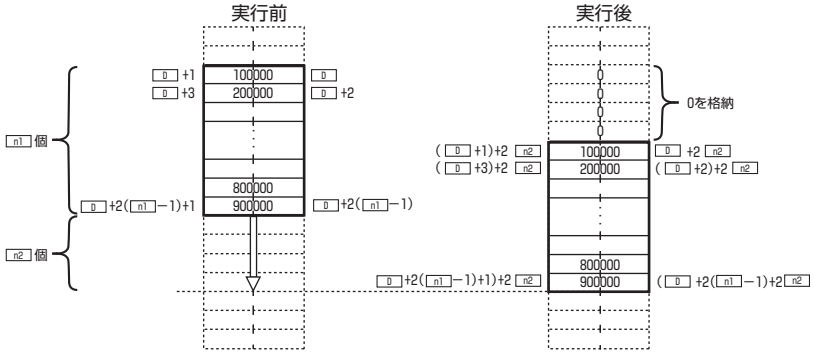
動作説明

実行条件が成り立つ(TRUE)とき、D を先頭として、n1 で指定されたデータ数の範囲に格納されたデータ(16ビット/32ビット)を、n2 で指定されたデータ数分だけ右へ(デバイス番号/配列の添え字の大きい方へ)シフトします。
シフト元の範囲の内、シフト先の範囲と重ならない部分には0が格納されます。

- **D :演算対象が16ビットの場合**



- **D :演算対象が32ビットの場合**



「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データシフト命令」を参照してください。

参 考　演算対象が32ビットデータのとき、演算範囲データ数、およびシフト数は指定されたデータ数×2ワードとなります。

●書式例

WSR (R3000, DM1000.U, 32, 10)

ポイント
- **n1 ＝0の場合、関数は実行されません。**
- **n2 ＝0の場合、演算結果は演算前と変わりません。**

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述　**WSR** (DM3000, EM3000, ANDA (EM3001, $F))

動作説明　毎スキャン(実行条件無し)、DM3000を先頭としてEM3000で示されるワードの範囲に格納されている16ビットデータを、EM3001の下位4ビット(抽出)で示されるワード数分右へシフトします。
シフト元の範囲の内、シフト先の範囲と重ならない部分には0が格納されます。

参 考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「ANDA関数」
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」（2-10ページ）を参照してください。

WSL ワードデバイス左シフト

ワードシフトレフト
WSL ([実行条件,]※1 演算対象, 演算範囲データ数, シフト数)

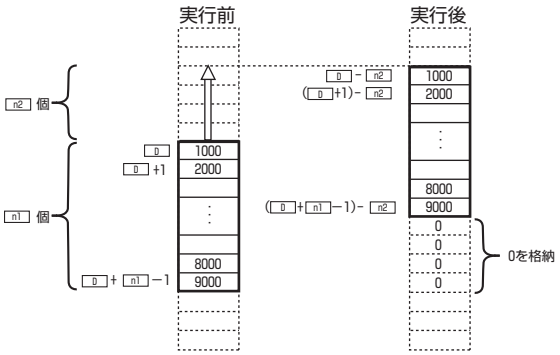
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象※2	左シフトする範囲の先頭を指定します。	.U	.U	.D	.D	.D	—	—	—	—	○	—
n1 演算範囲データ数	左シフトする範囲のデータ数の格納先、または左シフト範囲データ数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
n2 シフト数	左シフト数の格納先、または左シフト数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ビットデータを指定した場合、連続した16/32ビットを扱います。
チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。

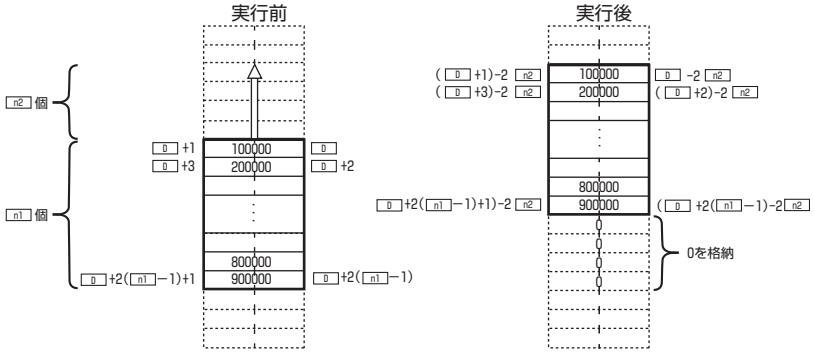
動作説明

実行条件が成り立つ(TRUE)とき、D を先頭として、n1 で指定されたデータ数の範囲に格納されたデータ(16ビット/32ビット)を、n2 で指定されたデータ数分だけ左へ(デバイス番号/配列の添え字の小さい方へ)シフトします。
シフト元の範囲の内、シフト先の範囲と重ならない部分には0が格納されます。

- **D :演算対象が16ビットの場合**



- D :演算対象が32ビットの場合



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データシフト命令」を参照してください。

参 考　演算対象が32ビットデータのとき、演算範囲データ数、およびシフト数は指定されたデータ数×2ワードとなります。

●書式例

WSL (R3000, DM1000.U, 32, 10)

ポイント
- n1 =0の場合、関数は実行されません。
- n2 =0の場合、演算結果は演算前と変わりません。

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　**WSL** (DM3000, EM3000, ANDA (EM3001, $F))

動作説明　毎スキャン(実行条件無し)、DM3000を先頭としてEM3000で示されるワードの範囲に格納されている16ビットデータを、EM3001の下位4ビット(抽出)で示されるワード数分左へシフトします。
シフト元の範囲の内、シフト先の範囲と重ならない部分には0が格納されます。

参 考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
「ANDA関数」
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

BSR ビットデバイス右シフト

ワードシフトライト
BSR ([実行条件,]※1 演算対象, 演算範囲対象数, シフト数)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象※2※3	右シフトする範囲の先頭を指定します。	—	—	—	—	—	—	.B	—	—	○	—
n1 演算範囲データ数	右シフトする範囲の対象数の格納先、または右シフト範囲対象数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
n2 シフト数	右シフト数の格納先、または右シフト数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

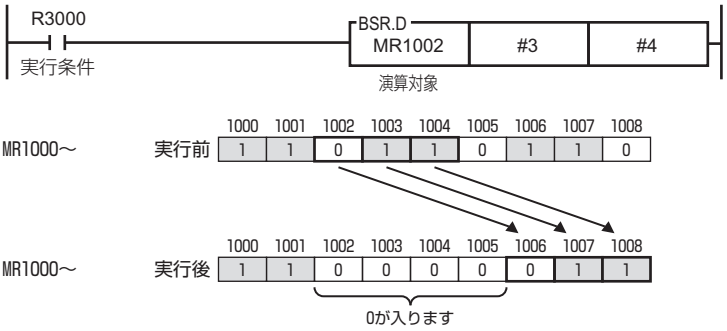
※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ビットデータを指定した場合、チャンネルをまたがっても正しく動作します。
ワードデータを指定した場合、n1 個のワードデータの最下位ビットがシフトする対象になります。最下位ビット以外は変化しません。
※3 T, C, CR, CTC, CTH, Z, CMは使用できません。

動作説明

実行条件が成り立つ(TRUE)とき、D を先頭とする、n1 で指定された個数の状態を、n2 で指定された回数分だけ右へ(デバイス番号/配列の添え字の大きい方へ)シフトします。D がワードデータのとき、各ワードデータの最下位ビットのみがシフト対象となります。(最下位ビット以外は変化しません)
シフト元の範囲の内、シフト先の範囲と重ならない部分(ワードデータのときは最下位ビット)にはOFFが格納されます。

●書式例

BSR (R3000, MR1002, 3, 4)



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データシフト命令」を参照してください。

ポイント
- **n2 =0の場合、演算結果は演算前と変わりません。**
- **次のいずれかの条件が成立した場合、関数は実行されません。**
  - **シフト先がデバイスや変数の範囲を超える場合**
  - **n1 が0の場合。**

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述 **BSR** (DM3000.B, EM3000, ANDA (EM3001, $F))

動作説明 毎スキャン(実行条件無し)、DM3000を先頭とするEM3000で示される個数分のワードデータの最下位ビットの状態を、EM3001の下位4ビット(抽出)で示される回数分だけ右へシフトします。
シフト元の範囲の内、シフト先の範囲と重ならない部分の最下位ビットにはOFFが格納されます。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
「ANDA関数」(7-22ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

BSL ビットデバイス左シフト

ビットシフトライト
BSL ([実行条件,]※1 演算対象, 演算範囲対象数, シフト数)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象※2※3	左シフトする範囲の先頭を指定します。	—	—	—	—	—	—	.B	—	—	○	—
n1 演算範囲データ数	左シフトする範囲の対象数の格納先、または左シフト範囲対象数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
n2 シフト数	左シフト数の格納先、または左シフト数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

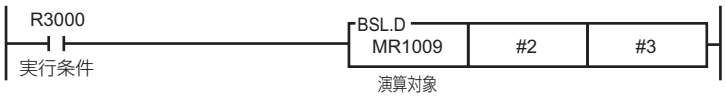
※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ビットデータを指定した場合、チャンネルをまたがっても正しく動作します。
ワードデータを指定した場合、n1 個のワードデータの最下位ビットがシフトする対象になります。最下位ビット以外は変化しません。
※3 T, C, CR, CTC, CTH, Z, CMは使用できません。

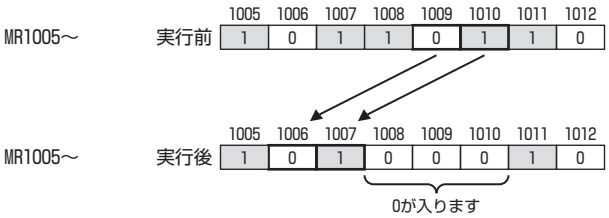
動作説明

実行条件が成り立つ(TRUE)とき、D を先頭とする、n1 で指定された個数の状態を、n2 で指定された回数分だけ左へ(デバイス番号/配列の添え字の小さい方へ)シフトします。D がワードデータのとき、各ワードデータの最下位ビットのみがシフト対象となります。(最下位ビット以外は変化しません)
シフト元の範囲の内、シフト先の範囲と重ならない部分(ワードデータのときは最下位ビット)にはOFFが格納されます。
シフト元の範囲の内、シフト先の範囲と重ならない部分には0が格納されます。

●書式例

BSL (R3000, MR1009, 2, 3)





「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データシフト命令」を参照してください。

ポイント

- n2 ＝0の場合、演算結果は演算前と変わりません。
- 次のいずれかの条件が成立した場合、関数は実行されません。
  - D －n1 ＞0となる場合。
  - n1 が0の場合。

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述

BSL (DM3000.B, EM3000, ANDA (EM3001, $F))

動作説明

毎スキャン(実行条件無し)、DM3000.Bを先頭とするEM3000で示される個数分のワードデータの最下位ビットの状態を、EM3001の下位4ビット(抽出)で示される回数分だけ左へシフトします。
シフト元の範囲の内、シフト先の範囲と重ならない部分の最下位ビットにはOFFが格納されます。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「ANDA関数」(7-22ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

LIMIT 上下限リミット制御

戻り値※1 = **LIMIT** (下限リミット値, 上限リミット値, 入力値)

引数/戻り値	説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
S1 下限リミット※2（最小出力限界値）	下限値、または下限値の格納先を指定します。	.U	.S	.D	.L	—	—	—	—	○	○	○
S2 上限リミット※2（最大出力限界値）	上限値、または上限値の格納先を指定します。	S1と同じ型として扱われます。				—	—	—	—	○	○	○
S3 入力値※2	上下限リミット制御の入力値、または入力値の格納先を指定します。	S1と同じ型として扱われます。				—	—	—	—	○	○	○
R 戻り値	LIMIT関数は、第1引数と同じ型を持ちます。※3	.U	.S	.D	.L	.F	※4 .DF	—	—	—	○	—

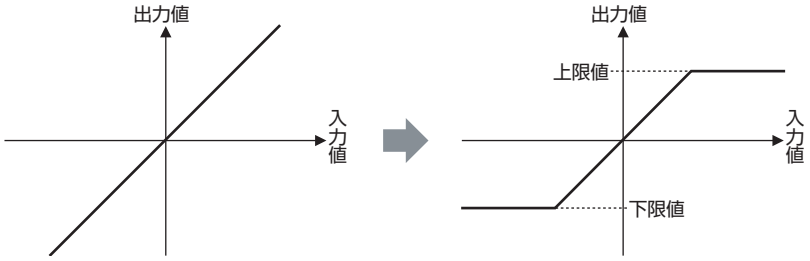
※1　戻り値は省略できます。省略した場合、演算結果はワークメモリに格納されます。
※2　• ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。
• CTHを指定した場合、現在値を扱います。CTCを使用した場合、設定値を扱います。
※3　戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※4　デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **KV-1000を使用する場合、システムマクロ「LIMITS」「LIMITL」を使用してください。**
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

S3 に対して、S2 、S1 で上下限制御をおこない、結果を R に格納します。

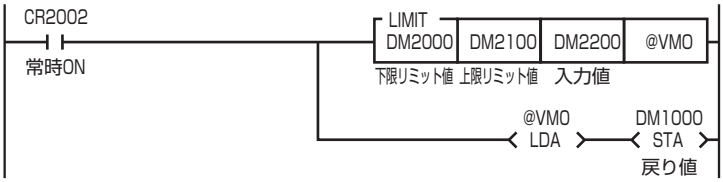


「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ制御命令」を参照してください。

●書式例

- S1 :下限リミット値が16ビット(.U/.S)の場合

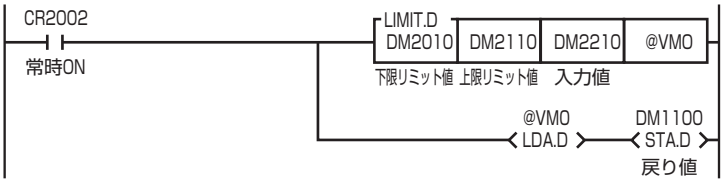
DM1000.U = LIMIT (DM2000.U, DM2100, DM2200)



条件	R に格納される値
S1 (下限値) > S3 (入力値)	S1 (下限値)
S2 (上限値) < S3 (入力値)	S2 (上限値)
S1 (下限値) ≦ S3 (入力値) ≦ S2 (上限値)	S3 (入力値)

- S1 :下限リミット値が32ビット(.D/.L)の場合

DM1100.D = LIMIT (DM2010.D, DM2110, DM2210)



条件	[R・R+1]に格納される値
[S1・S1+1] > [S3・S3+1] (下限値) (入力値)	[S1・S1+1] (下限値)
[S2・S2+1] > [S3・S3+1] (上限値) (入力値)	[S2・S2+1] (上限値)
[S1・S1+1] ≦ [S3・S3+1] ≦ [S2・S2+1] (下限値) (入力値) (上限値)	[S3・S3+1] (入力値)

BANDC 不感帯制御

戻り値※1 = **BANDC** (不感帯下限値, 不感帯上限値, 入力値)
不感帯制御

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	不感帯(無出力帯域)下限値※2	下限値、または下限値の格納先を指定します。	.S	.S	.L	.L	—	—	—	—	○	○	○
S2	不感帯(無出力帯域)上限値※2	上限値、または上限値の格納先を指定します。	S1と同じ型として扱われます。				—	—	—	—	○	○	○
S3	入力値※2	不感帯制御の入力値、または入力値の格納先を指定します。	S1と同じ型として扱われます。				—	—	—	—	○	○	○
R	戻り値	BANDC関数は、第1引数と同じ型を持ちます。※3	.U	.S	.D	.L	.F	※4 .DF	—	—	—	○	—

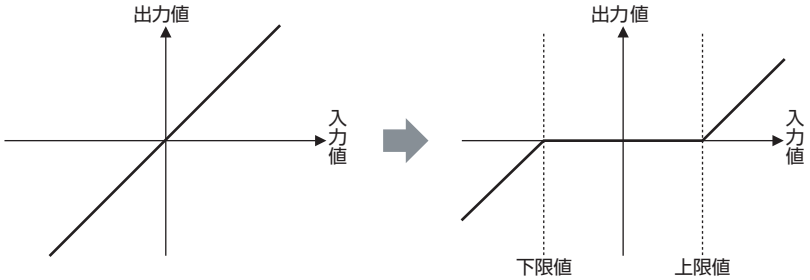
※1 戻り値は省略できます。省略した場合、演算結果はワークメモリに格納されます。
※2 • ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。
• CTHを指定した場合、現在値を扱います。CTCを使用した場合、設定値を扱います。
※3 戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※4 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **KV-1000を使用する場合、システムマクロ「BANDS」「BANDL」を使用してください。**
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

S3 に対して、S2 、S1 で不感帯制御をおこない、結果を R に格納します。

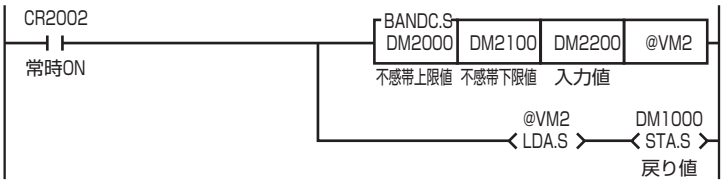


「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「データ制御命令」を参照してください。

●書式例

- [S1]:不感帯(無出力帯域)下限値が16ビット(.S)の場合

DM1000.S = BANDC (DM2000.S, DM2100, DM2200)



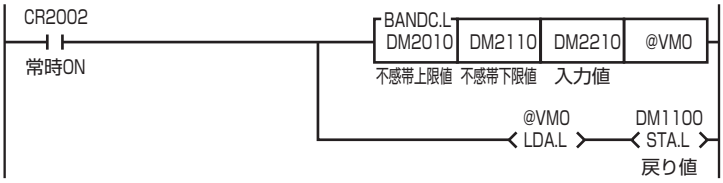
条件	[R]に格納される値
[S1](下限値)＞[S3](入力値)	[S3](入力値)－[S1](下限値)
[S2](上限値)＜[S3](入力値)	[S3](入力値)－[S2](上限値)
[S1](下限値)≦[S3](入力値)≦[S2](上限値)	0

※[R]には、16ビット符号付きBINデータの範囲内で値が格納されます。

- [R]≦－32768→[R]＝－32768
- [R]≧+32767→[R]＝+32767

- [S1]:不感帯(無出力帯域)下限値が32ビット(.L)の場合

DM1100.L = BANDC (DM2010.L, DM2110, DM2210)



条件	[[R]・[R]+1]に格納される値
[[S1]・[S1]+1]＞[[S3]・[S3]+1] (下限値)　(入力値)	[[S1]・[S1]+1] (下限値)
[[S2]・[S2]+1]＞[[S3]・[S3]+1] (上限値)　(入力値)	[[S2]・[S2]+1] (上限値)
[[S1]・[S1]+1]≦[[S3]・[S3]+1]≦[[S2]・[S2]+1] (下限値)　(入力値)　(上限値)	[[S3]・[S3]+1] (入力値)

※[[R]・[R]+1]には、32ビット符号付きBINデータの範囲内で値が格納されます。

- [[R]・[R]+1]≦－2147483648→[[R]・[R]+1]＝－2147483648
- [[R]・[R]+1]≧+2147483647→[[R]・[R]+1]＝+2147483647

ZONE ゾーン制御

戻り値※1 = **ZONE** (ゾーン) (負のバイアス値, 正のバイアス値, 入力値)

引数/戻り値	説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
S1 負のバイアス値※2	入力値に加算する負のバイアス値、または負のバイアス値の格納先を指定します。	.S	.S	.L	.L	—	—	—	—	○	○	○
S2 正のバイアス値※2	入力値に加算する正のバイアス値、または正のバイアス値の格納先を指定します。	S1と同じ型として扱われます。				—	—	—	—	○	○	○
S3 入力値※2	ゾーン制御の入力値、または入力値の格納先を指定します。	S1と同じ型として扱われます。				—	—	—	—	○	○	○
R 戻り値	ZONE関数は、第1引数と同じ型を持ちます。※3	.U	.S	.D	.L	.F	※4 .DF	—	—	—	○	—

※1 戻り値は省略できます。省略した場合、演算結果はワークメモリに格納されます。

※2
- ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。
- CTHを指定した場合、現在値を扱います。CTCを使用した場合、設定値を扱います。

※3 戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。

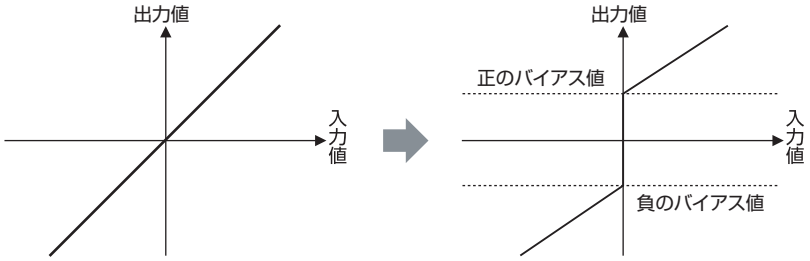
※4 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **ZONE関数は、KV-5500/5000/3000の専用関数です。KV-1000を使用する場合、システムマクロ「ZONES」「ZONEL」を使用してください。**
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

S3 に対して、S2 、S1 でゾーン制御をおこない、結果を R に格納します。



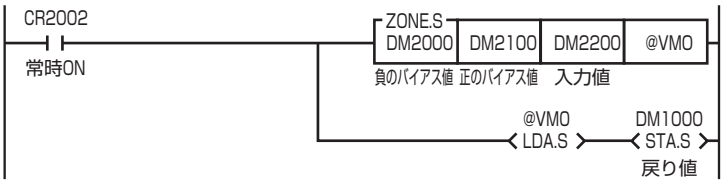
「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ制御命令」を参照してください。

●書式例

- S1 :負のバイアス値が16ビット(.S)の場合

DM1000.S = **ZONE** (DM2000.S, DM2100, DM2200)



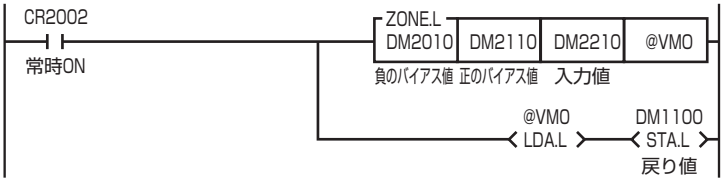
条件	R に格納される値
S3 (入力値)＜0	S3 (入力値)+ S1 (負のバイアス値)
S3 (入力値)＝0	0
S3 (入力値)＞0	S2 (入力値)+ S2 (正のバイアス値)

※ R には、16ビット符号付きBINデータの範囲内で値が格納されます。

- R ≦－32768→ R ＝－32768
- R ≧+32767→ R ＝+32767

- S1 :負のバイアス値が32ビット(.L)の場合

DM1100.L = **ZONE** (DM2010.L, DM2110, DM2210)



条件	[R ・ R +1]に格納される値
[S1 ・ S1 +1]＞[S3 ・ S3 +1] (負のバイアス値) (入力値)	[S1 ・ S1 +1] (負のバイアス値)
[S2 ・ S2 +1]＞[S3 ・ S3 +1] (正のバイアス値) (入力値)	[S2 ・ S2 +1] (正のバイアス値)
[S1 ・ S1 +1]≦[S3 ・ S3 +1]≦[S2 ・ S2 +1] (負のバイアス値) (入力値) (正のバイアス値)	[S3 ・ S3 +1] (入力値)

※[R ・ R +1]には、32ビット符号付きBINデータの範囲内で値が格納されます。

- [R ・ R +1]≦－2147483648→[R ・ R +1]＝－2147483648
- [R ・ R +1]≧+2147483647→[R ・ R +1]＝+2147483647

APR 折れ線近似(スケーリング)

戻り値※1 = **APR** (入力値, データテーブル先頭,データレコード数)
折れ線近似

引数/戻り値	説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
S1 入力値※2	折れ線近似の入力値、または入力値の格納先を指定します。	.U	.S	.D	.L	.F	※5 .DF	—	—	○	○	○
S2 データテーブル先頭※3	折れ線データテーブルを格納している先頭を指定します。	S1と同じ型として扱われます。				—	—	—	—	—	○	—
n データレコード数	折れ線データレコード数、またはレコード数の格納先を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	APR関数は、第1引数と同じ型を持ちます。※4	.U	.S	.D	.L	.F	※5 .DF	—	—	—	○	—

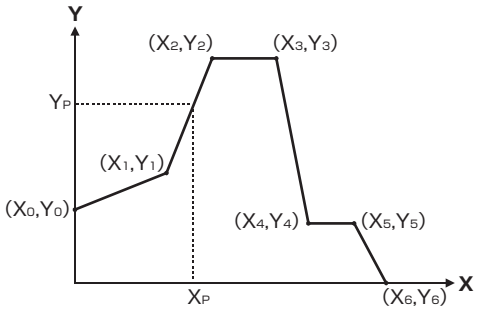
※1 戻り値は省略できます。省略した場合、演算結果はワークメモリに格納されます。
※2 • ビットデータを指定した場合、連続した16/32/64ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32/64ビットを扱います。
• T,C,CTHを指定した場合、現在値を扱います。CTCを使用した場合、設定値を扱います。
※3 S1の型が.Fまたは.DFの場合、T,Cは指定できません。
※4 戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※5 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **KV-1000を使用する場合、システムマクロ「APRS」「APRL」「APRF」を使用してください。**
- **引数の「.DF」は、KV STUDIO Ver.9.0以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000 シリーズ CPU ユニットおよび CPU 機能バージョン 2.0 以降のKV-7000シリーズCPUユニットでのみ指定できます。**
- **戻り値の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **戻り値の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-7500/7300 CPU機能バージョンについて」(3ページ)
「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

- **S1:入力値が16ビット(.U/.S)の場合**
  S1に格納されている折れ線近似の「入力値」に対し、S2を先頭にnレコード分格納されている「折れ線データテーブル」にしたがって近似計算をおこないます。
  折れ線近似計算は、S1が、折れ線データテーブルのどの範囲内にあるのかをサーチし、前後の2点で折れ線近似をおこないます。計算結果は、Rに格納されます。



$$Y_P = \frac{Y_2 - Y_1}{X_2 - X_1}(X_P - X_1) + Y_1$$

折れ線データテーブル

S2 テーブル先頭

Index	X	Y
0	0	10
1	8	13
2	14	20
3	20	20
4	23	6
5	26	6
6	28	0

n レコード数

Xのデータは必ず昇順に格納してください。

折れ線データテーブルのXの値は、昇順に設定してください。($X_0 < X_1 \ldots < X_k \ldots < X_n$)
折れ線データテーブルは、(X_0, Y_0)、(X_1, Y_1)…(X_{n-1}, Y_{n-1})、(X_n, Y_n)の順に[S2]から格納してください。

Index	X	Y
0	[S2]	[S2]+1
1	[S2]+2	[S2]+3
2	[S2]+4	[S2]+5
3	[S2]+6	[S2]+7
4	[S2]+8	[S2]+9
5	[S2]+10	[S2]+11
6	[S2]+12	[S2]+13

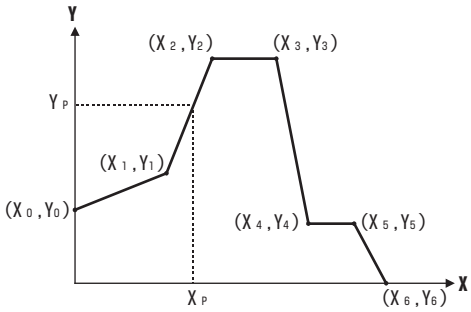
[S2] → テーブル先頭

[n]には、折れ線データテーブルのレコード数を格納してください。また、[n]は2以上の値としてください。

[S1](入力値)<X_0の場合は、[R]にはY_0が格納されます。
[S1](入力値)<X_nの場合は、[R]にはY_nが格納されます。
[S1](入力値)<X_kの場合は、[R]にはY_kが格納されます。

- **[S1]:入力値が32ビット(.D/.L)の場合**

[[S1]・[S1]+1]に格納されている折れ線近似の「入力値」に対し、[[S2]・[S2]+1]を先頭に[[n]]レコード分格納されている「折れ線データテーブル」にしたがって近似計算をおこないます。
折れ線近似計算は、[[S1]・[S1]+1]が、折れ線データテーブルのどの範囲内にあるのかをサーチし、前後の2点で折れ線近似をおこないます。
計算結果は、[[R]・[R]+1]に格納されます。



$$Y_P = \frac{Y_2 - Y_1}{X_2 - X_1}(X_P - X_1) + Y_1$$

折れ線データテーブル

[[S2]・[S2]+1] → テーブル先頭

Index	X	Y
0	0	10
1	8	13
2	14	20
3	20	20
4	23	6
5	26	6
6	28	0

[[n]] → レコード数

折れ線データテーブルは、(X_0, Y_0)、(X_1, Y_1)…(X_{n-1}, Y_{n-1})、(X_n, Y_n)の順に[[S2]・[S2]+1]から格納してください。
[[n]](レコード数)には、折れ線データテーブルのレコード数を格納してください。
また、[[n]]は2個以上としてください。

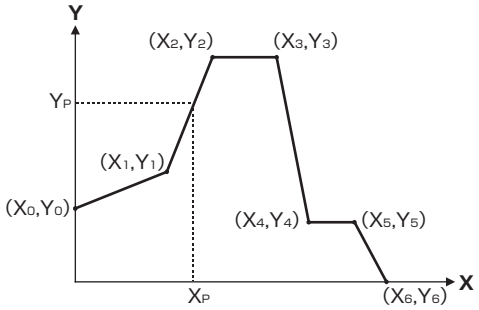
[[S1]・[S1]+1]<X_0の場合は、[[R]・[R]+1]にはY_0が格納されます。
[[S1]・[S1]+1]<X_nの場合は、[[R]・[R]+1]にはY_nが格納されます。
[[S1]・[S1]+1]<X_kの場合は、[[R]・[R]+1]にはY_kが格納されます(k=1～n)。

- **[S1]:入力値が32ビット(.F)の場合**

[S1]・[S1]+1]に格納されている折れ線近似の「入力値」に対し、[S2]・[S2]+1]を先頭に[n]レコード分格納されている「折れ線データテーブル」にしたがって近似計算(単精度浮動小数点型)をおこないます。

折れ線近似計算は、[S1]・[S1]+1]が、折れ線データテーブルのどの範囲内にあるのかをサーチし、前後の2点で折れ線近似をおこないます。

計算結果は、[R]・[R]+1]に格納されます。



$$Y_P = \frac{Y_2 - Y_1}{X_2 - X_1}(X_P - X_1) + Y_1$$

折れ線データテーブル

[S2]・[S2]+1]
テーブル先頭

Index	X	Y
0	0	5.2
1	3.3	6.8
2	5.7	13.9
3	7.9	13.9
4	9.9	2.1
5	12.2	2.1
6	15.5	0

[n]
レコード数

折れ線データテーブルは、(X_0, Y_0)、(X_1, Y_1)…(X_{n-1}, Y_{n-1})、(X_n, Y_n)の順に[S2]・[S2]+1]から格納してください。

折れ線データテーブルのXの値は、昇順に設定してください。$(X_0 < X_1 \ldots < X_k \ldots < X_n)$

[n]には、折れ線データテーブルのレコード数を格納してください。

また、[n]は2個以上としてください。

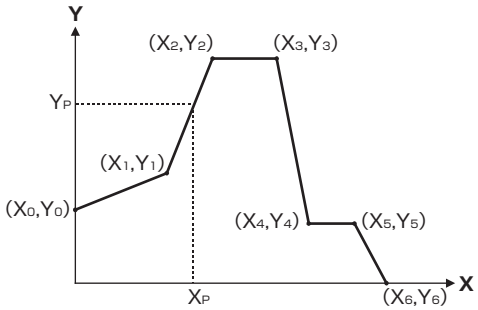
[S1]・[S1]+1]$<X_0$の場合は、[R]・[R]+1]にはY_0が格納されます。

[S1]・[S1]+1]$<X_n$の場合は、[R]・[R]+1]にはY_nが格納されます。

[S1]・[S1]+1]$=X_k$の場合は、[R]・[R]+1]にはY_kが格納されます(k=1～n)。

- **[S1]:入力値が64ビット(.DF)の場合**

[S1]・[S1]+1・[S1]+2・[S1]+3]に格納されている折れ線近似の「入力値」に対し、[S2]・[S2]+1・[S2]+2・[S2]+3]を先頭に[n]レコード分格納されている「折れ線データテーブル」にしたがって近似計算(倍精度浮動小数点型)をおこないます。
折れ線近似計算は、[S1]・[S1]+1・[S1]+2・[S1]+3]が、折れ線データテーブルのどの範囲内にあるのかをサーチし、前後の2点で折れ線近似をおこないます。
計算結果は、[R]・[R]+1・[R]+2・[R]+3]に格納されます。



$$Y_P = \frac{Y_2 - Y_1}{X_2 - X_1}(X_P - X_1) + Y_1$$

折れ線データテーブル

[S2]・[S2]+1
・[S2]+2
・[S2]+3]
テーブル先頭

Index	X	Y
0	0	5.2
1	3.3	6.8
2	5.7	13.9
3	7.9	13.9
4	9.9	2.1
5	12.2	2.1
6	15.5	0

[n]
レコード数

折れ線データテーブルは、(X_0, Y_0)、(X_1, Y_1)…(X_{n-1}, Y_{n-1})、(X_n, Y_n)の順に[S2]・[S2]+1・[S2]+2・[S2]+3]から格納してください。
折れ線データテーブルのXの値は、昇順に設定してください。($X_0 < X_1 \ldots < X_k \ldots < X_n$)

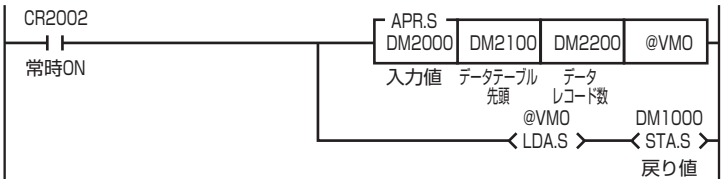
[n]には、折れ線データテーブルのレコード数を格納してください。
また、[n]は2個以上としてください。

[S1]・[S1]+1・[S1]+2・[S1]+3]<X_0の場合は、[R]・[R]+1・[R]+2・[R]+3]にはY_0が格納されます。
[S1]・[S1]+1・[S1]+2・[S1]+3]<X_nの場合は、[R]・[R]+1・[R]+2・[R]+3]にはY_nが格納されます。
[S1]・[S1]+1・[S1]+2・[S1]+3]=X_kの場合は、[R]・[R]+1・[R]+2・[R]+3]にはY_kが格納されます(k=1～n)。

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「データ制御命令」を参照してください。

●書式例

DM1000.S = APR (DM2000.S, DM2100, DM2200)



サンプルプログラム

●スケーリング機能として使用する場合

スクリプト記述

```
DM0.S=0
DM1.S=-1000
DM2.S=+255
DM3.S=+1000
DM100.S=APR(CM1630.s,DM0.S,2)
```

動作説明

KV Nanoシリーズのアナログボリューム CM1630(0~255)を－1000～+1000にスケーリングし、DM100に格納します。

参 考

KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

MEMO

RAMP 傾斜信号

傾斜信号
RAMP ([実行条件,]※1 入力値, 変化率設定, 時間単位設定, 出力値)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 入力値※2	入力値、または入力値の格納先を指定します(−32768〜32767)。	.S	.S	.S	.S	—	—	—	—	○	○	○
S2 変化率設定※2	時間単位あたりの変化率、または変化率の格納先を指定します。(0〜65535)	.U	.U	.U	.U	—	—	—	—	○	○	○
S3 時間単位設定※2※3	時間単位を指定します(0〜2)。	.U	.U	.U	.U	—	—	—	—	○	○	○
D 出力値※2	指定した間隔で、入力値への傾斜量を加算した結果を格納します。	.S	.S	.S	.S	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

※2
- ビットデータを指定した場合、連続した16ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16ビットを扱います。
- CTHを指定した場合、現在値を扱います。CTCを使用した場合、設定値を扱います。
- T, C, CTH, CTC, Zは下位16ビットを扱います。

※3 時間単位設定は、0〜2の範囲で指定してください。

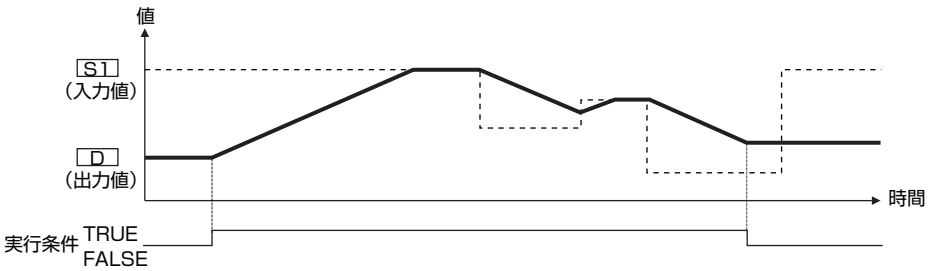
S3 時間単位設定の説明

値	時間単位
0	1秒単位
1	1分単位
2	1時間単位

ポイント
- **RAMP関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

実行条件が成り立つ(TRUE)とき、S1 に一致するまで、指定の傾き(S2 / S3)で D の値を近づけていきます。

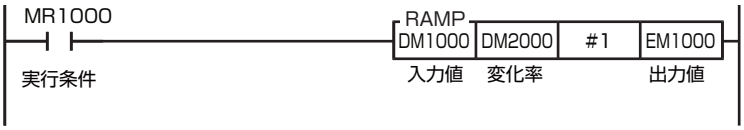


「演算フラグについて(CR2009〜CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「データ制御命令」を参照してください。

●書式例

RAMP (MR1000, DM1000.S, DM2000.U, 1, EM1000.S)



ポイント　S3 ≧3の場合、関数は実行されません。

サンプルプログラム

スクリプト記述　**RAMP** (DM1500.S, DM2500.U, 2, EM1500.S)

動作説明　EM1500.Sに格納されている出力値を、指定の傾き(DM2000/時間)で、入力値に近づけていきます。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

TPOUT 時分割比例出力

時分割比例出力

TPOUT ([実行条件,]※1 ON時間, 周期, 出力リレー)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 ON時間※2	出力リレーをONにする時間をms単位で指定します(0～65535)。	.U	.U	.U	.U	—	—	—	—	○	○	○
S2 周期※2	出力リレーのON/OFF周期をms単位で指定します(1～65535)。	.U	.U	.U	.U	—	—	—	—	○	○	○
D 出力リレー※3	ON/OFFさせる対象を指定します。	—	—	—	—	—	—	.B	—	—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

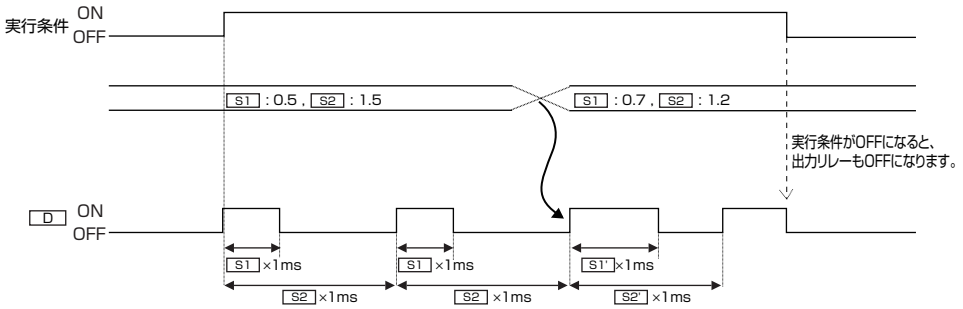
※2
- ビットデータを指定した場合、連続した16ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16ビットを扱います。
- CTHを指定した場合、現在値を扱います。CTCを使用した場合、設定値を扱います。
- T, C, CTH, CTC, Zは下位16ビットを扱います。

※3
- ワードデータを指定した場合、最下位ビットが結果の格納先になります。(最下位ビット以外は変化しません)
- チャンネルの先頭以外(R1002, R1005等)を指定した場合でも、次のチャンネルにまたがって正常に機能します。
- CTC, CTH, Zは指定できません。

ポイント
- **TPOUT関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

実行条件が成り立つ(TRUE)とき、S2(ms)でDをS1(ms)だけONにします。

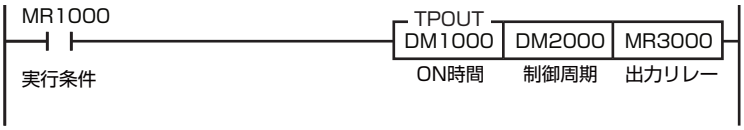


「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ制御命令」を参照してください。

●書式例

TPOUT (MR1000, DM1000, DM2000, MR3000)



ポイント

- **次のいずれかの条件が成立した場合、関数は実行されません。**
  - S2 < S1 の場合
  - S2 =0の場合
- **±10μs+1スキャンタイム以内の誤差が生じます。**
- S1 **に0を指定した場合、1スキャンもONしません。**
- **RUN中書き込みをおこなった場合、周期(** S2 **)が初期化されてRUN中書き込み終了時からON時間(** S1 **)を計測します。**
- **割り込みプログラム内および初期化モジュールでは、使用できません。**
- **サブルーチンプログラム内、STP～STE命令間、STG～JMP/ENDS命令間、CJ(NCJ)～LABEL命令間で使用するときは注意が必要です。**
  使用しているCPUユニットのユーザーズマニュアルの「タイマ命令使用時の注意」を参照してください。
- **RUN中書き込みをおこなう場合は、注意が必要です。**
  使用しているCPUユニットのユーザーズマニュアルの「RUN中書き込みについて」を参照してください。

サンプルプログラム

スクリプト記述 **TPOUT** (DM1500, EM1000, MR2000)

動作説明 毎スキャン(実行条件無し)、MR2000のON(DM1500ms)/OFF(EM1000-DM1500 ms)をEM1000 msの周期で切り換えます。

参考 KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

LLFLT 進み遅れフィルタ

進み遅れフィルタ
LLFLT ([実行条件,]※1 入力値, パラメータ, 出力値)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 入力値※2※3	入フィルタの入力値、または入力値の格納先を指定します(－32768～32767)。	.S	.S	.S	.S	—	—	—	—	○	○	○
D1 パラメータ※2※4	パラメータを格納している先頭を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
D2 出力値※2	出力値の格納先を格納します(－32768～32767)。	.S	.S	.S	.S	—	—	—	—	—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ビットデータを指定した場合、連続した16ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16ビットを扱います。
※3
- CTHを指定した場合、現在値を扱います。CTCを使用した場合、設定値を扱います。
- T, C, CTH, CTC, Zは下位16ビットを扱います。

※4
- ビットデータを指定した場合、連続する48ビットを占有します。
- ビットデータでは、チャンネルの先頭を指定してください。
- ワードデータを指定した場合、連続する3ワードを占有します。

ポイント
- **LLFLT関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000 シリーズ CPU ユニット、CPU 機能バージョン 2.0 以降の KV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

実行条件が成り立つ(TRUE)とき、S に対して D1 ごとに進み遅れ演算を実行して、D2 に格納します。
実行条件の立ち上がり時、D2 は S の値で初期化されます。

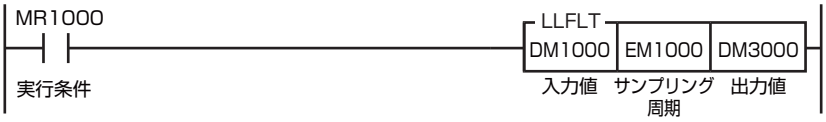
D1 :パラメータの内容

パラメータ		説明
D1	サンプリング周期(T_S)	命令を実行する周期時間を指定します。(1～60000ms) 遅れ時間、進み時間よりも少ない値を指定してください。
D1 **+1**	遅れ時間(T1)	遅れ要素の時定数を指定します。(0～65535ms) 0を指定した場合、遅れがなくなります。
D1 **+2**	進み時間(T2)	進み要素の時定数を指定します。(0～65535ms) 0を指定した場合、進みがなくなります。

進み遅れ演算の詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「LLFLT命令」を参照してください。
「演算フラグについて(CR2009～CR2012)」(1-18ページ)

●書式例

LLFLT (MR1000, DM1000.S, EM1000, DM3000.S)



ポイント

- **RUN 中書き込みをおこなった場合、サンプリング周期 (D1 +1) が初期化され、出力値 (D2)は入力値(S)の値で初期化されます。**
- **割り込みプログラム内および初期化モジュールでは、使用できません。**
- **サブルーチンプログラム内、STP ～ STE 命令間、STG ～ JMP/ENDS 命令間、CJ(NCJ) ～ LABEL命令間で使用するときは注意が必要です。**
  使用している CPU ユニットのユーザーズマニュアルの「タイマ命令使用時の注意」を参照してください。
- **RUN中書き込みをおこなう場合は、注意が必要です。**
  使用しているCPUユニットのユーザーズマニュアルの「RUN中書き込みについて」を参照してください。

サンプルプログラム

スクリプト記述

LLFLT (DM1500.S, EM2000, DM2000.S)

動作説明

毎スキャン(実行条件無し)、DM1500に格納されている値に対して、EM1000のサンプリング周期でEM1001(遅れ要素の時定数)とEM1002(進み要素の時定数)から進み遅れ演算をおこないます。
演算結果は、DM3000に格納されます。

参 考

KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

TBCD　BINデータ→BCDデータ変換

戻り値※1 = **TBCD** (変換対象)
トランスファBCD

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象	変換対象、または変換対象データを指定します。	.U	.U	.D	.D	—	—	—	—	○	○	○
R 戻り値	TBCD関数は、引数と同じ型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

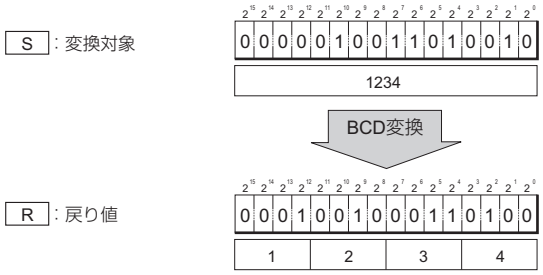
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

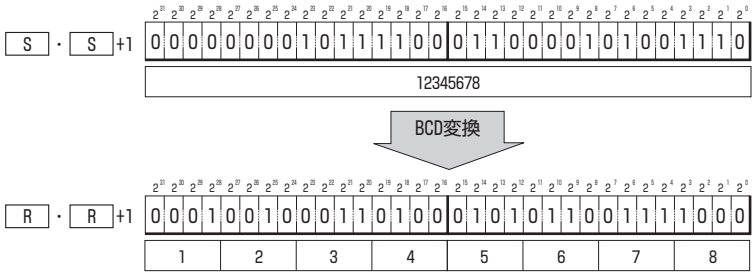
動作説明

S に格納された16ビットのバイナリデータを4桁BCD(16ビット)データに、あるいは32ビットのバイナリデータを8桁BCD(32ビット)データに変換し、R に格納します。

- **S :変換対象が16ビットの場合**



- **S :変換対象が32ビットの場合**



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ変換命令」を参照してください。

●書式例

DM2000.U = TBCD (DM1000.U)

ポイント　**S が16ビットの時は0～9999、32ビットの時は0～99999999の範囲を超えた場合は、R に S の値が格納されます。**

サンプルプログラム

●ビットデータを指定する場合

スクリプト記述　DM2000.D = **TBCD** (MR1000.D)

動作説明　MR1000.Dに格納されたバイナリデータを、8桁BCD(32ビット)データに変換します。
変換結果をDM2000.Dに格納します。

●使用例

TBCD関数は、変換対象で指定されたバイナリデータをBCDデータに変換できるため、現在の値をそのままBCDで表示出力させる場合などに用いられます。

スクリプト記述　R1000.U = **TBCD** (TOU (DM1000 + 1000))

動作説明　DM1000の内容に1000を加算した結果の値を4桁のBCDに変換し、出力リレーR1000～R1015に出力します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

TBIN　BCDデータ→BINデータ変換

戻り値※1 = **TBIN** (変換対象)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象	変換対象、または変換対象データを指定します。	.U	.U	.D	.D	—	—	—	—	○	○	○
R 戻り値	TBIN関数は、第1引数と同じ型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

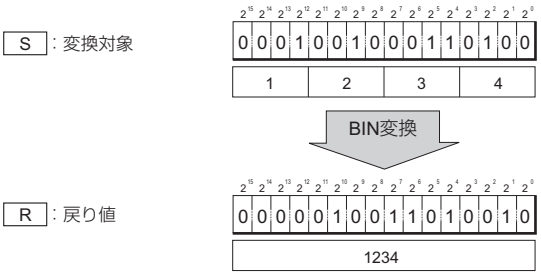
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

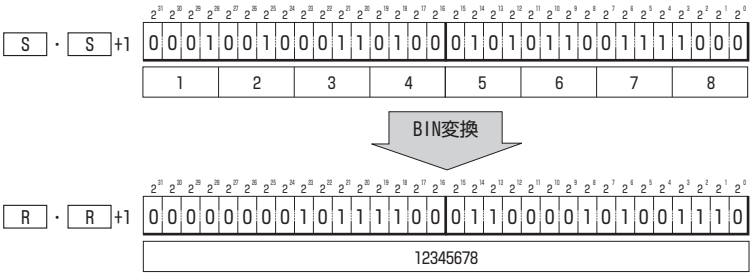
動作説明

S に格納された4桁BCD(16ビット)データを16ビットのバイナリデータに、あるいは8桁BCD(32ビット)データを32ビットのバイナリデータに変換し、R に格納します。

- **S :変換対象が16ビットの場合**



- **S :変換対象が32ビットの場合**



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「データ変換命令」を参照してください。

●書式例

DM2000.U = TBIN (DM1000.U)

ポイント　S がBCDデータでないとき、R には S の値が格納されます。

サンプルプログラム

●ビットデータを指定する場合

スクリプト記述　DM2000.D = **TBIN** (MR1000.D)

動作説明　MR1000.Dに格納された8桁BCD(32ビット)データを、バイナリデータに変換します。
変換結果をDM2000.Dに格納します。

●使用例

TBIN関数は、変換対象で指定されたBCDデータをバイナリデータに変換できるため、現在のBCDの値を入力して、そのままバイナリデータとして扱う場合などに用いられます。

スクリプト記述　DM2000.U = TOU (**TBIN** (R2000.U) + 1000)

動作説明　4桁のBCDデータが出力されているリレーR2000～R2015の状態(値)を入力し、バイナリデータに変換した結果に1000を加算してDM2000に格納します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

MPX 4ビットデータデコード

戻り値※1 = **MPX (変換**対象, 変換桁位置**)**

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象※3※4	変換対象、または変換対象データを指定します。	.U	.U	.D	.D	—	—	—	—	—	○	○
n 変換桁位置※2	変換桁位置を指定します。(0～3)	—	—	—	—	—	—	—	—	○	—	—
R 戻り値	MPX関数は、.Uの型を持ちます。※5	.U	.S	.D	.L	.F	※6 .DF	—	—	—	○	—

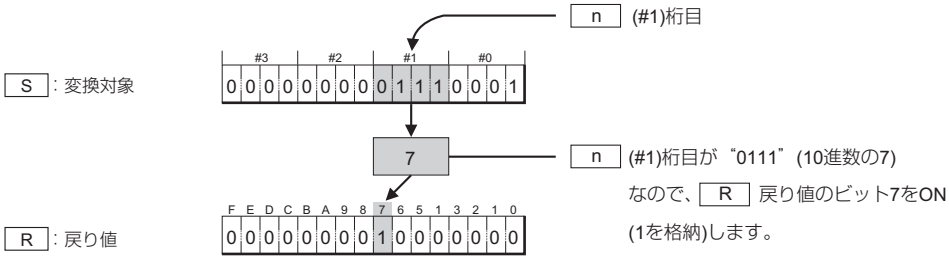
※1 戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2 変換桁位置には、0～3(4ビット/1桁)の桁数を定数で指定します。(対象の指定はできません)
※3 変換対象に32ビットデータが指定された場合、下位16ビット(下位ワード)が変換対象(有効変換桁)となります。
※4 変換対象にビットデータのチャンネルの先頭以外(R1002, R1005等)を指定した場合でも、次のチャンネルにまたがって正常に機能します。
※5 戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※6 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

S に格納された16ビットデータの、n で指定された桁数の4ビット(0桁～3桁)を数値化(0～15)します。数値化された値が示すビットのみをONにした結果を R に格納します。



詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ変換命令」を参照してください。

●書式例

DM2000 = **MPX** (DM1000, 2)

サンプルプログラム

●ビットデータを16ビットBINデータにデコードする場合

スクリプト記述　DM2000 = **MPX** (MR1000.U, 3)

動作説明　MR1000.Uに格納されたデータの、3桁目(ビット12～15の4ビット)の値をデコードして、結果をDM2000に格納します。

ラダー変換

●他の関数を組み込んで使用する場合

スクリプト記述　R2000.U = **MPX** (TBIN (DM1000), 1)

動作説明　DM1000に格納されている16ビットバイナリデータを4桁のBCD(16ビット)データに変換し、変換結果の10の桁(1桁目)の値をデコードして、対応する出力リレー(R2000～R2015)をONします。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「TBIN関数」(7-82ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

TBCD
TBIN
MPX
DMX
GRY
RGRY
DISN
UNIN
DISB
UNIB
SWAP
BSWAP
DECO
ENCO
TOU
TOS
TOD
TOL
CPMSET
CPMGET

DMX 4ビットデータエンコード

戻り値※1 = **DMX (変換対象)**

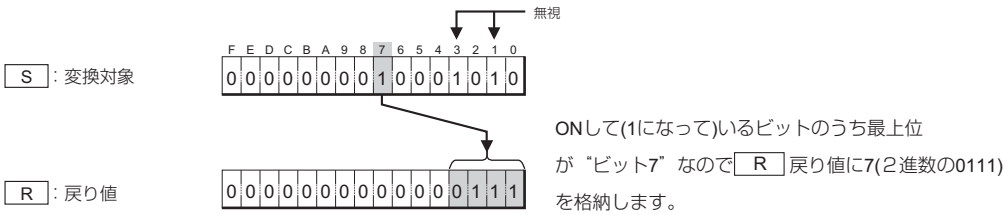
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象※2※3	変換対象、または演算対象データを指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	DMX関数は、.Uの型を持ちます。※4	.U	.S	.D	.L	.F	※5 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　変換対象に32ビットデータが指定された場合、下位16ビット(下位ワード)が変換対象(有効ビット)となります。
※3　演算対象にビットデータのチャンネルの先頭以外(R1002, R1005等)を指定した場合でも、次のチャンネルにまたがって正常に機能します。
※4　戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※5　デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

S に格納された16ビットデータの、ONしているビットの内で最上位のビット位置(0～15)を、16ビット(下位4ビットに格納)データとして R に格納します。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ変換命令」を参照してください。

●書式例

DM2000 = DMX (DM1000)

ポイント
S の値が0の場合、R には0が格納されます。

サンプルプログラム

●ビットデータを4ビットの数値データにエンコードする場合

スクリプト記述　DM2000 = **DMX** (MR1000.U)

動作説明　MR1000.Dに格納されたデータの、ONしているビットの内で最上位のビット位置を、DM2000に格納します。

ラダー変換

●他の関数に組み込んで使用する場合

スクリプト記述　R2000.U = TBCD (**DMX** (MR1000.U))

動作説明　センサー入力リレー(MR1000～MR1015)の状態で、ONしているビットの内で最上位のビット位置(センサー番号)を、4桁のBCD(16ビット)データに変換して結果をBCD出力リレー(R2000～R2015)に出力します。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「TBCD関数」(7-80ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

TBCD
TBIN
MPX
DMX
GRY
RGRY
DISN
UNIN
DISB
UNIB
SWAP
BSWAP
DECO
ENCO
TOU
TOS
TOD
TOL
CPMSET
CPMGET

GRY　BINデータ→グレイコード変換

戻り値※1 = **GRY** (**変換**対象)
グレイコード変換

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象	変換対象、または演算対象データを指定します。	.U	.U	.D	.D	—	—	—	—	○	○	○
R 戻り値	GRY関数は、引数と同じ型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

S に格納された16ビット、あるいは32ビットのバイナリデータを、グレイコードに変換し、R に格納します。
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ変換命令」を参照してください。

●書式例

DM2000.U = GRY (DM1000.U)

グレイコード表

10進	bit														
	16	15	14	13	12	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
4	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
≀															
31	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
32	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
≀															
255	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
256	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
≀															
65535	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

サンプルプログラム

●ビットデータを指定する場合

スクリプト記述　DM2000.D = **GRY** (MR1000.D)

動作説明　MR1000.Dに格納されたバイナリデータを、グレイコードに変換します。
変換結果をDM2000.Dに格納します。

●使用例

GRY関数は、変換対象で指定されたバイナリデータをグレイコードに変換できるため、現在の値をそのままグレイコードで出力させる場合などに用いられます。

スクリプト記述　R1000.U = **GRY** (TOU (DM1000 + 128))

動作説明　DM1000の内容に128を加算した結果の値をグレイコードに変換し、出力リレーR1000～R1015に出力します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

RGRY グレイコード→BINデータ変換

戻り値※1 = **RGRY** (**変換**対象)
グレイコード逆変換

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象※2	変換対象、または演算対象データを指定します。	.U	.U	.D	.D	—	—	—	—	○	○	○
R 戻り値	RGRY関数は、引数と同じ型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV8000/-7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

S に格納されたグレイコードを、16ビット、あるいは32ビットのバイナリデータに変換し、R に格納します
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ変換命令」を参照してください。

●書式例

DM2000.U = **RGRY** (DM1000.U)

グレイコード表

10進	bit														
	16	15	14	13	12	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
4	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
≀															
31	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
32	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
≀															
255	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
256	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
≀															
65535	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

サンプルプログラム

●ビットデータを指定する場合

スクリプト記述　DM2000.D = **RGRY** (MR1000.D)

動作説明　MR1000.Dに格納されたグレイコードを、バイナリデータに変換します。
変換結果をDM2000.Dに格納します。

●使用例

RGRY関数は、変換対象で指定されたグレイコードをバイナリデータに変換できるため、現在のグレイコードの値を入力して、そのままバイナリデータとして扱う場合などに用いられます。

スクリプト記述　DM2000.U = TOU (**RGRY** (R2000.U) – 128)

動作説明　グレイコードが出力されているリレーR2000～R2015の状態(値)を入力し、バイナリデータに変換した結果から128を減算してDM2000に格納します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

DISN ニブル分離

ディスN
DISN ([実行条件,]※1 演算対象, 分離データ格納対象, 分離データ数)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	演算対象※2※4	ニブル分離するデータの格納先を指定します。	.U	.U	.D	.D	—	—	.B	—	—	○	—
D	分離データ格納対象※2※4	ニブル分離したデータを格納する先頭を指定します。	S と同じ型として扱われます。					—	—	—	—	○	—
n	分離データ数※3	分離するニブルデータの個数を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

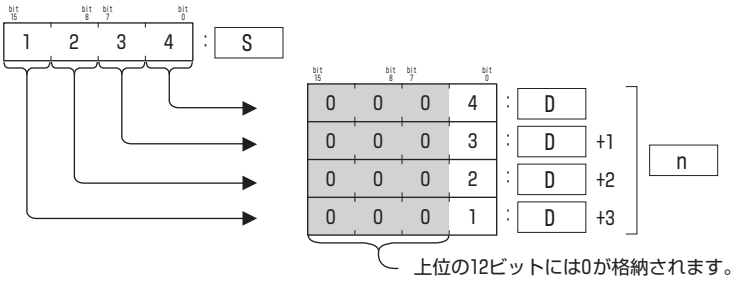
※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ビットデータを指定した場合、S は連続した16/32ビットを扱います。D は連続した64/128ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって扱います。
ワードデータを指定した場合、S は連続した1/2ワードを扱います。D は4/8ワードを扱います。
※3 S で指定したデータのサフィックスが.Uもしくは.Bの場合0～4、.Dの場合0～8の範囲で指定します。
※4 CTC, CTH, Zは指定できません。

ポイント **KV-1000で使用する場合は、ラダープログラムでDISN命令を使用してください。**

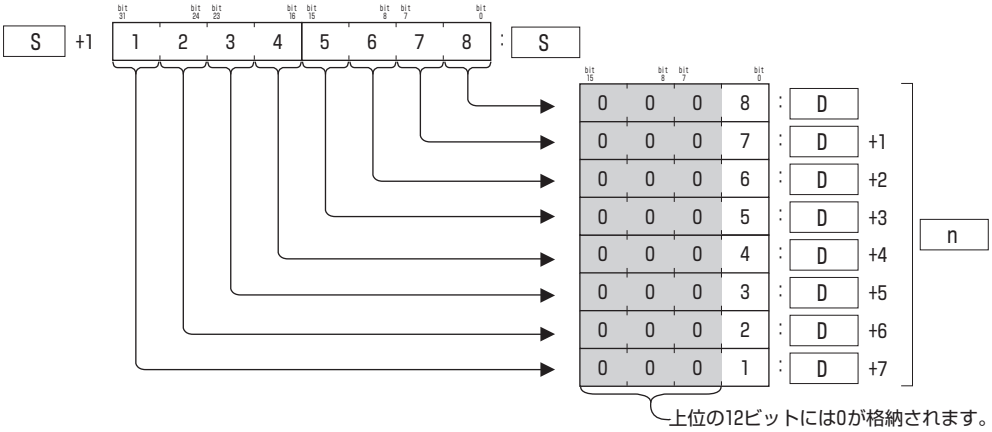
動作説明

実行条件が成り立つ(TRUE)とき、S で指定されたデータを n で指定された個数にニブル(4ビット)単位で分離し、D へ格納します。
4ビットより上位には、0が格納されます。

- **S :演算対象が16ビット符号なし(.U)の場合**



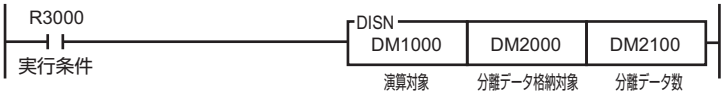
- S :演算対象が32ビット符号なし(.D)の場合



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ変換命令」を参照してください。

●書式例

DISN (R3000, DM1000.U, DM2000, DM2100.U)



ポイント　n が0の場合、関数は動作しません。

TBCD
TBIN
MPX
DMX
GRY
RGRY
DISN
UNIN
DISB
UNIB
SWAP
BSWAP
DECO
ENCO
TOU
TOS
TOD
TOL
CPMSET
CPMGET

UNIN ニブル結合

ユニN
UNIN ([実行条件,]※1 演算対象, 結合データ格納対象, 結合データ数)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	演算対象※2※4	ニブル結合するデータの格納先を指定します。	.U	.U	.D	.D	—	—	.B	—	—	○	—
D	結合データ格納対象※2※4	ニブル結合したデータを格納する先頭を指定します。	S と同じ型として扱われます。					—	—	—	—	○	—
n	結合データ数※3	結合するニブルデータの個数を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

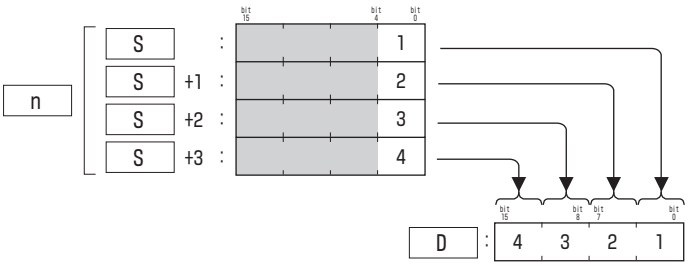
※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ビットデータを指定した場合、S は連続した64/128ビットを扱います。D は連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって扱います。
ワードデータを指定した場合、S は連続した4/8ワードを扱います。D は1/2ワードを扱います。
※3 S で指定したデータのサフィックスが.Uもしくは.Bの場合0～4、.Dの場合0～8の範囲で指定します。
※4 CTC, CTH, Zは指定できません。

ポイント **KV-1000で使用する場合は、ラダープログラムでUNIN命令を使用してください。**

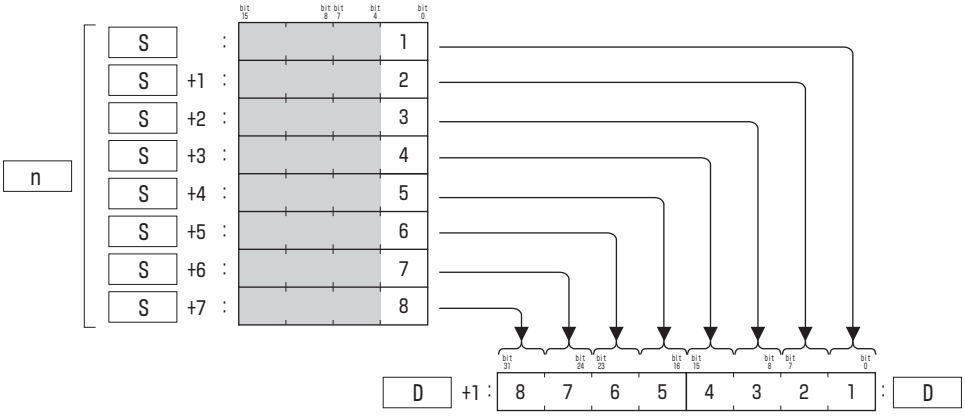
動作説明

実行条件が成り立つ(TRUE)とき、S を先頭とする結合データの下位4ビットを n だけBINデータに結合し、D へ格納します。

- **S :演算対象が16ビット符号なし(.U)の場合**



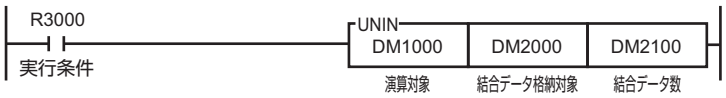
- S :演算対象が32ビット符号なし(.D)の場合



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ変換命令」を参照してください。

●書式例

UNIN (R3000, DM1000.U, DM2000, DM2100.U)



ポイント　n が0の場合、関数は動作しません。

TBCD
TBIN
MPX
DMX
GRY
RGRY
DISN
UNIN
DISB
UNIB
SWAP
BSWAP
DECO
ENCO
TOU
TOS
TOD
TOL
CPMSET
CPMGET

DISB バイト分離

ディスB
DISB ([実行条件,]※1 演算対象, 分離データ格納対象, 分離データ数)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	演算対象※2※4	バイト分離するデータの格納先を指定します。	.U	.U	.D	.D	—	—	.B	—	—	○	—
D	分離データ格納対象※2※4	バイト分離したデータを格納する先頭を指定します。	S と同じ型として扱われます。				—	—	—	—	—	○	—
n	分離データ数※3	分離するバイトデータの個数を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

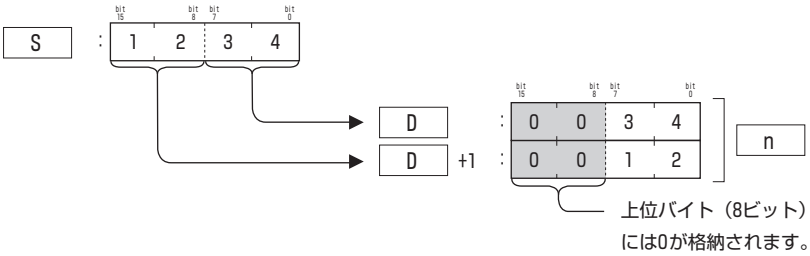
※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ビットデータを指定した場合、S は連続した16/32ビットを扱います。D は連続した32/64ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって扱います。
ワードデータを指定した場合、S は連続した1/2ワードを扱います。D は2/4ワードを扱います。
※3 S で指定したデータのサフィックスが.Uもしくは.Bの場合0～2、.Dの場合0～4の範囲で指定します。
※4 CTC, CTH, Zは指定できません。

ポイント **KV-1000で使用する場合は、ラダープログラムでDISB命令を使用してください。**

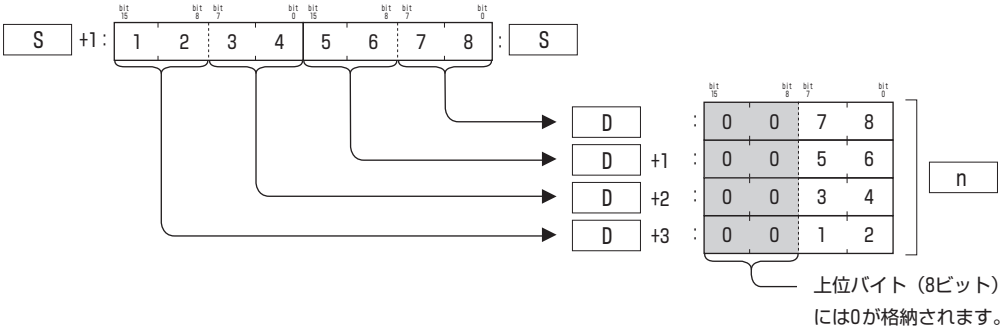
動作説明

実行条件が成り立つ(TRUE)とき、S で指定されたデータを n で指定された個数にバイト(8ビット)単位で分離し、D へ格納します。
8ビットより上位には、0が格納されます。

- **S :演算対象が16ビット符号なし(.U)の場合**



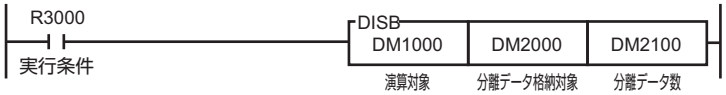
- S :演算対象が32ビット符号なし(.D)の場合



「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ変換命令」を参照してください。

●書式例

DISB (R3000, DM1000.U, DM2000, DM2100.U)



ポイント　n が0の場合、関数は動作しません。

TBCD
TBIN
MPX
DMX
GRY
RGRY
DISN
UNIN
DISB
UNIB
SWAP
BSWAP
DECO
ENCO
TOU
TOS
TOD
TOL
CPMSET
CPMGET

UNIB バイト結合

ユニB
UNIB ([実行条件,]※1 演算対象, 結合データ格納対象, 結合データ数)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	演算対象※2※4	バイト結合するデータの格納先を指定します。	.U	.U	.D	.D	—	—	.B	—	—	○	—
D	結合データ格納対象※2※4	バイト結合したデータを格納する先頭を指定します。	S と同じ型として扱われます。					—	—	—	—	○	—
n	結合データ数※3	結合するバイトデータの個数を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

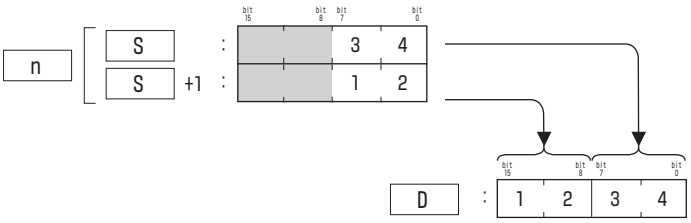
※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ビットデータを指定した場合、S は連続した32/64ビットを扱います。D は連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって扱います。
ワードデータを指定した場合、S は連続した2/4ワードを扱います。D は1/2ワードを扱います。
※3 S で指定したデータのサフィックスが.Uもしくは.Bの場合0～2、.Dの場合0～4の範囲で指定します。
※4 CTC, CTH, Zは指定できません。

ポイント **KV-1000で使用する場合は、ラダープログラムでUNIB命令を使用してください。**

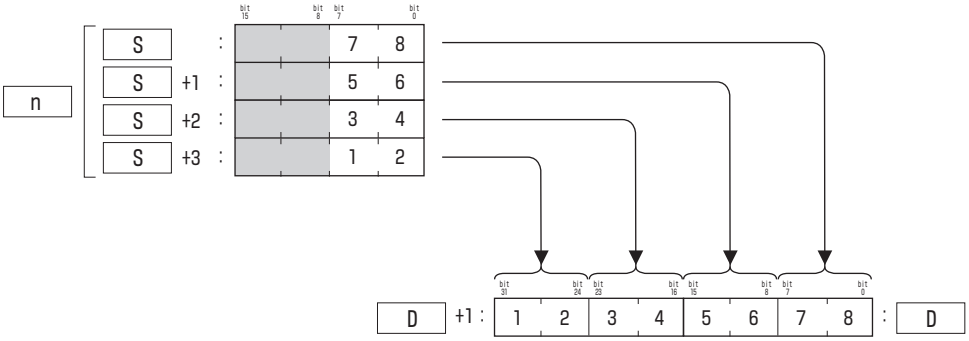
動作説明

実行条件が成り立つ(TRUE)とき、S を先頭とする結合データの下位8ビットを n だけBINデータに結合し、D へ格納します。

- **S :演算対象が16ビット符号なし(.U)の場合**



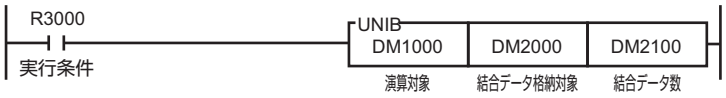
- S :演算対象が32ビット符号なし(.D)の場合



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ変換命令」を参照してください。

●書式例

UNIB (R3000, DM1000.U, DM2000, DM2100.U)



ポイント　n が0の場合、関数は動作しません。

TBCD
TBIN
MPX
DMX
GRY
RGRY
DISN
UNIN
DISB
UNIB
SWAP
BSWAP
DECO
ENCO
TOU
TOS
TOD
TOL
CPMSET
CPMGET

SWAP 上位･下位バイト入れ替え

戻り値※1 = **SWAP** (**変換**対象)
トランスファBCD

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 入替対象	入替対象、または入替対象データを指定します。	.U	.U	.D	.D	—	—	—	—	○	○	○
R 戻り値	SWAP関数は、引数と同じ型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、入替結果は内部レジスタに格納されます。
※2　戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

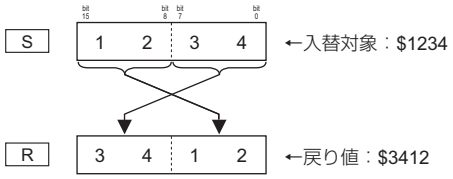
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

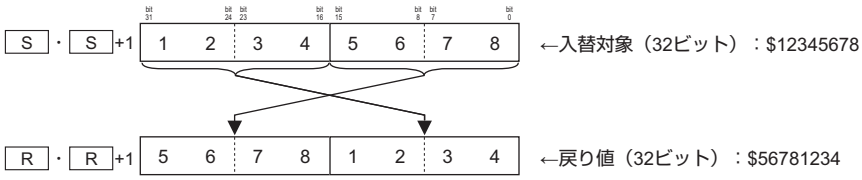
動作説明

S に格納された16ビットデータの、上位バイト(上位8ビット)と下位バイト(下位8ビット)の値を入れ替え、結果を R **に格納**します。

- **S :入替対象が16ビットの場合**



- **S :入替対象が32ビットの場合**



詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ変換命令」を参照してください。

●書式例

DM2000.U = **SWAP** (DM1000.U)

サンプルプログラム

●ビットデータを指定する場合

スクリプト記述　DM2000.D = **SWAP** (MR1000.D)

動作説明　MR1000.Dに格納された32ビットデータの、上位ワード(MR1000～MR1015)と下位ワード(MR1100～MR1115)の値を入れ替えます。
入れ替え結果をDM2000.Dに格納します。

●他の関数を組み込んで使用する場合

スクリプト記述　R2000.U = **SWAP** (TBCD (TOU (DM1000 + 100)))

動作説明　DM1000の内容に100を加算した結果の値を4桁のBCDに変換し、変換したBCDの上位2桁と下位2桁の値を入れ替えてR2000～R2015に出力します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
「TBCD関数」(7-80ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

TBCD
TBIN
MPX
DMX
GRY
RGRY
DISN
UNIN
DISB
UNIB
SWAP
BSWAP
DECO
ENCO
TOU
TOS
TOD
TOL
CPMSET
CPMGET

BSWAP ブロックスワップ

ブロックスワップ
BSWAP ([実行条件,]※1 入替対象, 対象データ数)

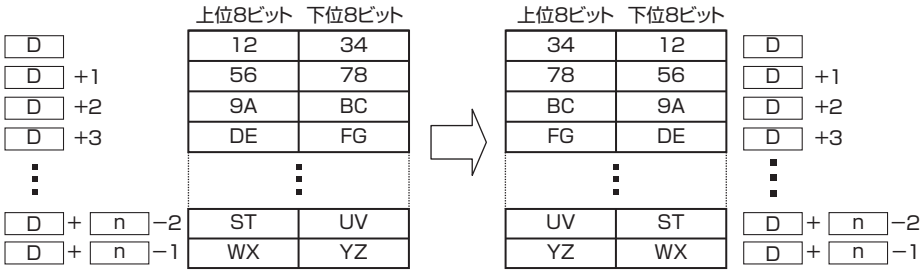
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D 入替対象	入れ替え対象を指定します。	.U	.U	.U	.U	—	—	.B	—	—	○	—
n 対象データ数	入れ替える数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

ポイント
- **BSWAP関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000 シリーズ CPU ユニット、CPU 機能バージョン 2.0 以降の KV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

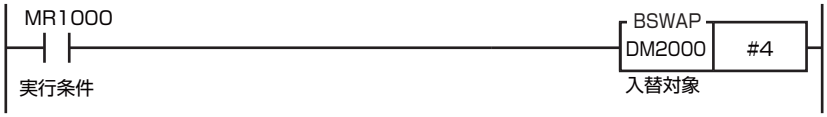
実行条件が成り立つ(TRUE)とき、D を先頭とする n 分のデータの上位バイト(上位8バイト)と下位バイト(下位8バイト)を入れ替えます。



詳細については、『KV-8000/7000/5000/3000/1000シリーズ·KV Nanoシリーズ命令語リファレンスマニュアル』「データ変換命令」を参照してください。

●書式例

BSWAP (MR1000, DM2000, 4)



ポイント
n =0の場合、関数は実行されません。

サンプルプログラム

スクリプト記述 **BSWAP** (DM3000, 8)

動作説明 DM3000～DM3007の上位バイトと下位バイトを入れ替えます。

参 考 KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

TBCD
TBIN
MPX
DMX
GRY
RGRY
DISN
UNIN
DISB
UNIB
SWAP
BSWAP
DECO
ENCO
TOU
TOS
TOD
TOL
CPMSET
CPMGET

DECO 8ビットデコード

8ビットデコード

DECO ([実行条件,]※1 変換対象, 格納先, 変換対象範囲)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象	変換する範囲の先頭を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
D 格納先※2※3	デコード結果の格納先を指定します。	—	—	—	—		—	.B	—	—	○	—
n 変換対象範囲	変換する範囲数(ビット数)を指定します(1～8)	※4 .U	※4 .U	※4 .U	※4 .U	—	—	—	—	○	※4 ○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

※2 ワードデータを指定した場合、連続する2^n個のワードデータの最下位ビットがデコード結果の格納先になります。(最下位ビット以外は変化しません)

※3 CTC, CTH, Zは使用できません。

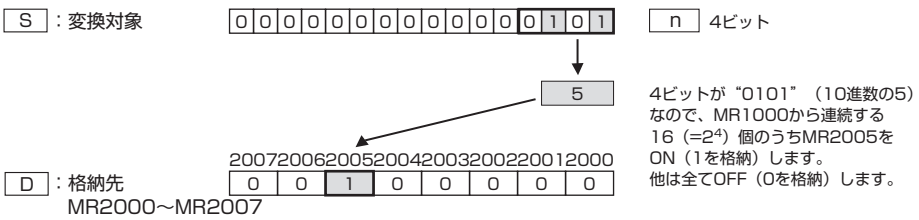
※4 変換対象範囲数に対象が指定できるのは、以下の条件を満たすときだけです。

- KV STUDIOのバージョンがVer.6.00以上のとき
- 対応機種が KV-8000/7000 シリーズ CPU ユニット、CPU 機能バージョン 2.0 以降の KV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットのとき

動作説明

実行条件が成り立つ(TRUE)とき、S で指定されたビットデータを先頭とする、n で指定された個数(ビット数)分の状態を数値化(0～255)します。D を先頭として連続する2^n 個のビットデータのうちで、先頭番号+数値化した値(番目)のビットをONし、その他のビットはすべてOFFします。

S がワードデータのとき、n は数値化するビット数を示し、また格納先がワードデータのときは、指定されたワードデータを先頭として連続する2^n 個のワードデータのうちで、先頭番号+数値化した値(番目)のワードデータの最下位ビットをONし、その他のワードデータの最下位ビットはすべてOFFします。(最下位ビット以外は変化しません)

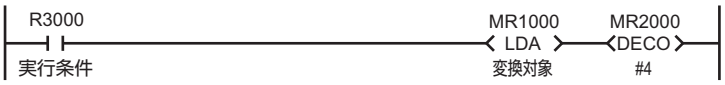


「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ変換命令」を参照してください。

●書式例

DECO (R3000, MR1000.U, MR2000, 4)



ポイント　n =0の場合、関数は実行されません。

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述　**DECO** (ORA (DM3000, DM3001), MR3000, 5)

動作説明　毎スキャン(実行条件無し)、DM3000とDM3001に格納されているデータの論理和演算の結果の値の下位5ビットをデコードします。
MR3000を先頭として連続する2^5個のビットのうち、先頭番号+数値化した値(番目)のビットをONし、その他のビットはすべてOFFします。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
「ORA関数」(7-24ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

TBCD
TBIN
MPX
DMX
GRY
RGRY
DISN
UNIN
DISB
UNIB
SWAP
BSWAP
DECO
ENCO
TOU
TOS
TOD
TOL
CPMSET
CPMGET

ENCO 8ビットエンコード

戻り値※1 = ENCO (変換対象, 変換対象範囲数)
8ビットエンコード

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象※2※3	変換する範囲の先頭を指定します。	—	—	—	—	—	—	.B	—	—	○	—
n 変換対象範囲数	変換する範囲数(ビット数)を指定します。(1～256)	※5 .U	※5 .U	※5 .U	※5 .U	—	—	—	—	○	※5 ○	○
R 戻り値	ENCO関数は.Uの型を持ちます。※4	.U	.S	.D	.L	.F	※6 .DF	—	—	—	○	—

※1 戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2 ワードデータを指定した場合、連続する n 個のワードデータの最下位ビットがエンコードする対象になります。
※3 CTC, CTH, Zは使用できません。
※4 戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※5 変換対象範囲数に対象が指定できるのは、以下の条件を満たすときだけです。
- KV STUDIOのバージョンがVer.6.00以上のとき
- 対応機種が KV-8000/7000 シリーズ CPU ユニット、CPU 機能バージョン 2.0 以降の KV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットのとき

※6 デバイスZに「.DF」サフィックスは指定できません。

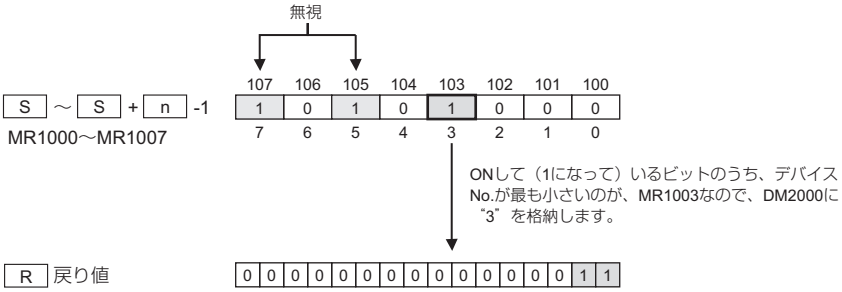
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

S で指定されたビットデータを先頭とする、n で指定された個数(ビット数)分の状態のうちで、ONしているビットの最小のデバイスの位置(0～255)を、16ビットデータとして R に格納します。

S がワードデータのとき、n はワード数を示し、変換対象範囲数分のワードデータのうちで、最下位ビットがONしているワードの最小のビットの位置(0～255)を、16ビットデータとして R に格納します。

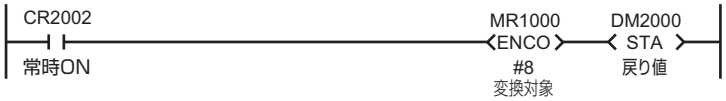


「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ変換命令」を参照してください。

●書式例

DM2000.U = **ENCO** (MR1000, 8)



ポイント　n =0の場合、関数は実行されません。

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　DM2000.U = TBCD (**ENCO** (MR3000, 32))

動作説明　MR3000を先頭とする32個のビットデータ(MR3000～MR3115)のうちで、ONしている最小のビットの位置(16ビットデータ)を、4桁BCDデータの変換してDM2000に格納します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「TBCD関数」(7-80ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

TBCD
TBIN
MPX
DMX
GRY
RGRY
DISN
UNIN
DISB
UNIB
SWAP
BSWAP
DECO
ENCO
TOU
TOS
TOD
TOL
CPMSET
CPMGET

TOU データの型を.Uに変換
TOS データの型を.Sに変換
TOD データの型を.Dに変換
TOL データの型を.Lに変換

戻り値※1 = **TOU** (変換対象)
U変換

戻り値※1 = **TOS** (変換対象)
S変換

戻り値※1 = **TOD** (変換対象)
D変換

戻り値※1 = **TOL** (変換対象)
L変換

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象	変換対象のデータの格納先、または変換データを指定します。	.U	.S	.D	.L	.F	※2 .DF	—	—	○	○	○
R 戻り値※1	TOU関数:.Uの型を持ちます。 TOS関数:.Sの型を持ちます。 TOD関数:.Dの型を持ちます。 TOL関数:.Lの型を持ちます。	.U	.S	.D	.L	.F	※2 .DF	—	—	—	○	—

※1　戻り値に関数が持つ型と異なる型を指定した場合、代入時に自動的に型変換がおこなわれます。
※2　デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

- **TOU**

S に格納されたデータを、16ビットバイナリデータ(符号無し)に変換し、結果を R に格納します。

参考
- 変換対象データに 32 ビットバイナリデータを指定した場合は、下位 16 ビットが変換対象となります。
- 変換対象データに単精度 / 倍精度浮動小数点型実数を指定した場合は、32 ビットバイナリデータ(符号無し)に変換された結果の、下位16ビットが変換対象となります。
- 戻り値の格納先のサフィックスは、.Uにしてください。

- **TOS**

  S に格納されたデータを、16ビットバイナリデータ(符号付き)に変換し、結果を R に格納します。

  参考
  - 変換対象データに 32 ビットバイナリデータを指定した場合は、下位 16 ビットが変換対象となります。
  - 変換対象データに単精度 / 倍精度浮動小数点型実数を指定した場合は、32 ビットバイナリデータ(符号付き)に変換された結果の、下位16ビットが変換対象となります。
  - 戻り値の格納先のサフィックスは、.Sにしてください。

- **TOD**

  S に格納されたデータを、32ビットバイナリデータ(符号無し)に変換し、結果を R に格納します。

  参考
  - 変換対象データに16ビットバイナリデータを指定した場合は、データ拡張された結果の32ビットバイナリデータが変換対象となります。
  - 変換対象データに単精度 / 倍精度浮動小数点型実数を指定した場合は、32 ビットバイナリデータ(符号無し)に変換された結果が変換対象となります。
  - 戻り値の格納先のサフィックスは、.Dにしてください。

- **TOL**

  S に格納されたデータを、32ビットバイナリデータ(符号付き)に変換し、結果を R に格納します。

  参考
  - 変換対象データに16ビットバイナリデータを指定した場合は、データ拡張された結果の32ビットバイナリデータが変換対象となります。
  - 変換対象データに単精度 / 倍精度浮動小数点型実数を指定した場合は、32 ビットバイナリデータ(符号付き)に変換された結果が変換対象となります。
  - 戻り値の格納先のサフィックスは、.Lにしてください。

●書式例

DM2000.L = **TOL** (DM1000.S)

サンプルプログラム

スクリプトで記述された演算式は、32ビット符号付きデータ(.L)で演算されます。
スクリプトからラダーに変換する際、演算結果の格納先のサフィックスが「.L」以外の場合、型変換警告が発生します。
型変換警告を回避するためには、演算結果を「TOU, TOS, TOD, TOL」の各関数を使って、格納先と同じ型に変換してください。

●演算結果の格納先の型が、符号無し32ビットデータ「.D」の場合

スクリプト記述　DM1000.D = **TOD** ((DM2000 + 50) * DM2100 + 10)

動作説明　“＝”の右辺の演算結果(.Lで演算)を符号無し32ビットデータ「.D」に変換して、左辺のDM1000.Dに代入(格納)します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

CPMSET CIPメッセージ作成

CPMSET ([実行条件,][※1] ,入力データ格納先,パラメータ,出力データ格納先)

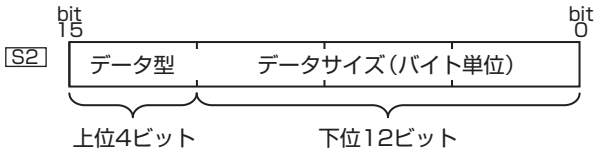
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 入力データ格納先[※2]	追加するデータの先頭を指定します。	.U	.S	.D	.L	.F	.DF	.B	.T	—	○	—
S2 パラメータ	追加する送信データのデータ型やデータサイズ(バイト単位)、または格納先を指定します。[※2]	.U	.U	.U	.U	—	—	—	—	○	○	○
D 出力データ格納先[※2]	作成する送信データの先頭を指定します。	.U	.S	.D	.L	.F	.DF	.B	.T	—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ビットデータを指定する場合は、チャンネルの先頭を指定してください。

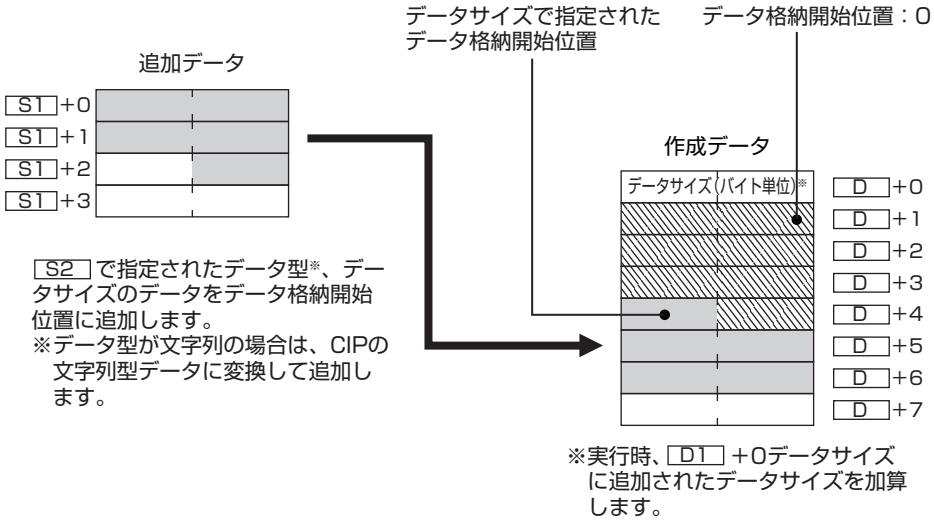
ポイント
- **CPMSET関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

実行条件が成り立つ(TRUE)とき、S1に格納されているデータをS2の上位4ビットで指定されたデータ型で、下位12ビットで指定されたデータサイズ(バイト単位)分、Dに格納されたデータサイズ(バイト単位)以降の領域に追加します。
最初にデータを追加する場合は、Dに0を格納してから、関数を実行します。
CPMSET関数は、CIPのデータ型の文字列データ、数値データを作成するための関数です。
CIPのデータ型については、『KV EtherNet/IP機能ユーザーズマニュアル』を参照してください。



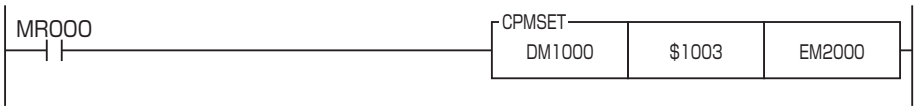
上位4ビット	下位12ビット
0:文字列以外	1～4095(001H～FFFH):データサイズを指定
1:文字列	0～1999(000H～7CFH):CIPの文字列型データに変換する文字数を指定 0を指定した場合はNUL(00H)までを変換します。 指定した文字数の途中にNUL(00H)が含まれる場合は、NUL(00H)以降の文字には、00Hを追加します。
2以上:システム予約	—



詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ変換命令」を参照してください。

●書式例

CPMSET (MR0,DM1000,$1003,EM2000)



CPMGET CIPメッセージ取得

CPMGET ([実行条件**,]**※1 **,**入力データ格納先**,**パラメータ**,**
入力データオフセット**,**出力データ格納先**)**

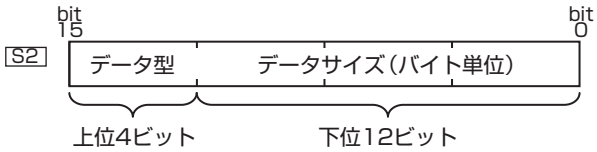
引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	入力データ格納先※2	受信データの格納先の先頭を指定します。	.U	.S	.D	.L	.F	.DF	.B	.T	—	○	—
S2	パラメータ	取得する受信データのデータ型やデータサイズ(バイト単位)、または格納先を指定します。※2	.U	.U	.U	.U	—	—	—	—	○	○	○
D1	受信データバイトオフセット※2	受信データの取得位置を指定する先頭からのバイトオフセット、または格納先を指定します。	.U	.S	.D	.L	—	—	—	—	—	○	—
D2	出力データ格納先※2	取得データを格納する対象の先頭を指定します。	.U	.S	.D	.L	.F	.DF	.B	.T	—	○	—
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ビットデータを指定する場合は、チャンネルの先頭を指定してください。

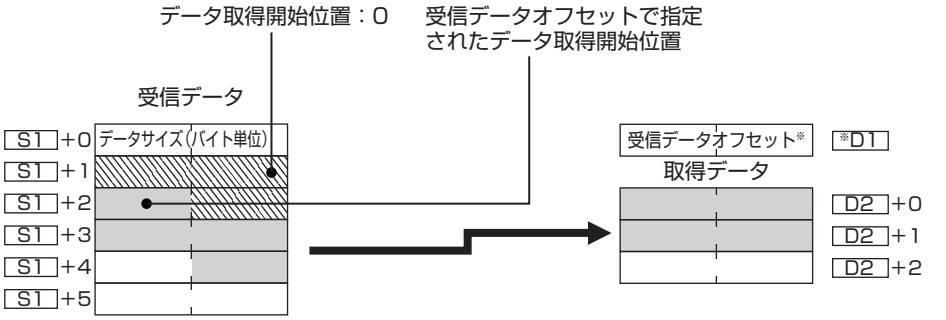
ポイント
- **CPMGET関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

実行条件が成り立つ(TRUE)とき、S1に格納されている受信データのD1で指定されたオフセット位置以降のデータを、S2の上位4ビットで指定されたデータ型で、S2の下位12ビットで指定されたデータサイズ(バイト単位)分のデータを、D2に格納します。データを格納後、D1が更新されます。
CPMGET関数は、受信したCIPのデータ型の文字列データ、数値データを取得するための関数です。
CIPのデータ型については、『KV EtherNet/IP機能ユーザーズマニュアル』を参照してください。



上位4ビット	下位12ビット
0:文字列以外	1～4095(001H～FFFH):データサイズを指定
1:文字列	0～1999(000H～7CFH):CIPの文字列型データに変換する文字数を指定 0を指定した場合はNUL(00H)までを変換します。 指定した文字数の途中にNUL(00H)が含まれる場合は、NUL(00H)以降の文字には、00Hを追加します。
2以上:システム予約	—



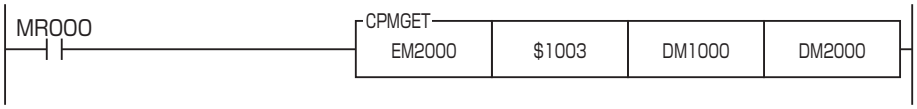
D1 のデータ取得開始位置以降のデータから、S2 で指定されたデータ型※、データサイズ分のデータを取得して、D2 以降に格納します。
※データ型が文字列の場合は、CIPの文字列型データに変換します。

※実行時、D1 受信データオフセットに取得したデータサイズを加算します。

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ変換命令」を参照してください。

●書式例

CPMGET (MR0, EM2000, $1003, DM1000, DM2000)



FLOAT　BINデータ→単精度浮動小数点型実数データ変換

戻り値※1 = **FLOAT** (変換対象)
単精度浮動小数変換

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象	変換対象、または変換対象データを指定します。	.U	.U	.D	.D	—	※3 .DF	—	—	○	○	○
R 戻り値	FLOAT関数は、.Fの型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

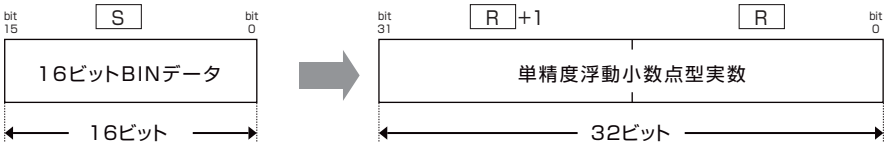
※1　戻り値の省略は可能です。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に.F以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

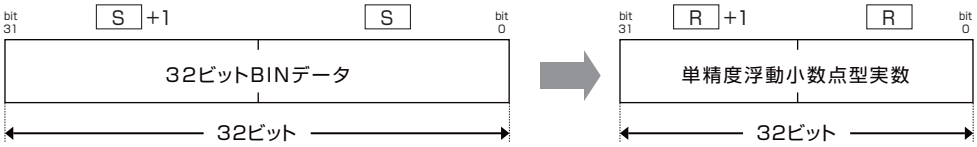
動作説明

S に格納されたバイナリデータ(16ビット/32ビット(符号無し/符号付き)、倍精度浮動小数点型実数)を、単精度浮動小数点型実数データに変換し、結果を R に格納します。

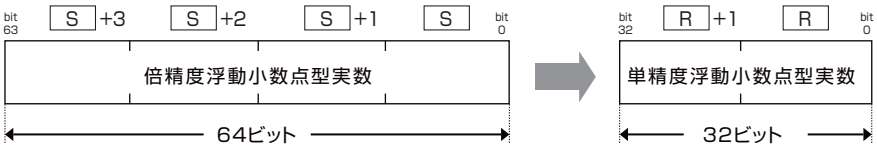
- **S :変換対象の型が16ビットの場合**



- **S :変換対象の型が32ビットの場合**



- **S :変換対象が倍精度浮動小数点型実数の場合**



詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「浮動小数命令」を参照してください。

●書式例

DM2000.F = **FLOAT** (DM1000.U)

ポイント　**S の整数値が－16777216～16777215(24ビットBIN)の範囲を超えた場合、25ビット目以降は丸められます。**

サンプルプログラム

●使用例

スクリプト記述　DM2000.F = **FLOAT** (DM1000.U + 123)

動作説明　DM1000.U+123の演算結果を単精度浮動小数点型実数データに変換し、DM2000.Fに格納します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

FLOAT
TOF
DFLOAT
TODF
INTG
INT
DISF
UNIF
EXP
LOG
LOG10
RAD
DEG
SIN
COS
TAN
ASIN
ACOS
ATAN
ATAN2

TOF　BINデータ→単精度浮動小数点型実数データ変換

戻り値 = **TOF** (変換対象)
F変換

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象	変換対象のデータを格納した対象、または変換データを指定します。	.U	.U	.D	.D	.F	※2 .DF	—	—	○	○	○
R 戻り値	TOF関数は、.Fの型を持ちます。※1	.U	.S	.D	.L	.F	※2 .DF	—	—	—	○	—

※1　戻り値に.F以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※2　デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

S に格納されたデータを、単精度浮動小数点型実数データに変換し、結果を R に格納します。
「FLOAT関数」(7-114ページ)

●書式例

DM2000.F = **TOF** (DM1000.D)

参考
- 変換対象データは、データ型 (16 ビット /32 ビット (符号無し / 符号付き)) に対応したFLOAT命令により、32ビット単精度浮動小数点型実数に変換されます。
- 変換対象データが倍精度浮動小数点型実数のとき、DFTOF 命令により 32 ビット単精度浮動小数点型実数に変換されます。
- 戻り値の格納先には、.Fサフィックスを指定する必要があります。

MEMO

FLOAT
TOF
DFLOAT
TODF
INTG
INT
DISF
UNIF
EXP
LOG
LOG10
RAD
DEG
SIN
COS
TAN
ASIN
ACOS
ATAN
ATAN2

DFLOAT BINデータ→倍精度浮動小数点実数データ変換

戻り値※1 = **DFLOAT** (変換対象)
倍精度浮動小数変換

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象	変換対象のデータを格納した対象、または変換データを指定します。	.U	.U	.D	.D	.F	—	—	—	○	○	○
R 戻り値	DFLOAT関数は、.DFの型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は64ビット内部レジスタに格納されます。
※2　戻り値に.DF以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

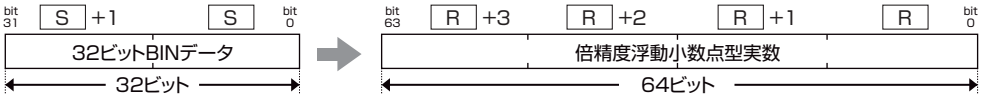
ポイント
- **DFLOAT関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000 シリーズ CPU ユニット、CPU 機能バージョン 2.0 以降の KV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ使用できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

S に格納されたバイナリデータ(16ビット/32ビット(符号無し/符号付き)、単精度浮動小数点型実数)を、倍精度浮動小数点型実数データに変換し、結果を R に格納します。

- **S :変換対象の型が16ビットの場合**

- **S :変換対象の型が32ビットの場合**



- **S :変換対象の型が単精度浮動小数点型実数の場合**

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「浮動小数命令」を参照してください。

●書式例

DM2000.DF = **DFLOAT** (DM1000.S)

サンプルプログラム

スクリプト記述　DM2500.DF = **DFLOAT** (DM1500.L + 1000)

動作説明　DM1500.L+1000の演算結果を倍精度浮動小数点型実数データに変換し、DM2500.DFに格納します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

FLOAT
TOF
DFLOAT
TODF
INTG
INT
DISF
UNIF
EXP
LOG
LOG10
RAD
DEG
SIN
COS
TAN
ASIN
ACOS
ATAN
ATAN2

TODF　BINデータ→倍精度浮動小数点実数データ変換

戻り値 = **TODF** (変換対象)
DF変換

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象	変換対象のデータを格納した対象、または変換データを指定します。	.U	.U	.D	.D	.F	※2 .DF	—	—	○	○	○
R 戻り値	TODF関数は、.DFの型を持ちます。※1	.U	.S	.D	.L	.F	※2 .DF	—	—	—	○	—

※1　戻り値に.DF以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※2　デバイスZに「.DF」サフィックスは指定できません。

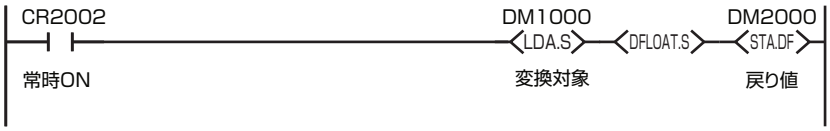
ポイント
- **TODF関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000 シリーズ CPU ユニット、CPU 機能バージョン 2.0 以降の KV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ使用できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

S に格納されたデータを、倍精度浮動小数点型実数データに変換し、結果を R に格納します。
「DFLOAT関数」(7-118ページ)

●書式例

DM2000.DF = **TODF** (DM1000.S)



参考
- 変換対象データは、データ型(16ビット/32ビット(符号無し/符号付き)、単精度浮動小数点型実数)に対応したDFLOAT命令により、64ビット倍精度浮動小数点型実数に変換されます。
- 戻り値の格納先には、.DFサフィックスを指定する必要があります。

MEMO

FLOAT
TOF
DFLOAT
TODF
INTG
INT
DISF
UNIF
EXP
LOG
LOG10
RAD
DEG
SIN
COS
TAN
ASIN
ACOS
ATAN
ATAN2

INTG 単精度/倍精度浮動小数点型実数データ→BINデータ変換

戻り値※1 = **INTG** (変換対象)
整数変換

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象	変換対象、または変換対象データを指定します。	—	—	—	—	.F	※3 .DF	—	—	○	○	○
R 戻り値	INTG関数は、.Lの型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　C, T, CTC, CTH, Zは使用できません。
※3　戻り値に.L以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※4　デバイスZに「.DF」サフィックスは指定できません。

ポイント

- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)
- **単精度浮動小数点型実数の範囲は、以下のとおりです。**

  −3.4E+38 ≦ N ≦ −1.4E−45
  N = 0
  1.4E−45 ≦ N ≦ 3.4E+38
  (有効桁数:約7桁)
  (KV Nanoシリーズの場合、絶対値が1.4E−45〜1.2E−38の範囲は使用できません。)

- **倍精度浮動小数点型実数の範囲は、以下のとおりです。**

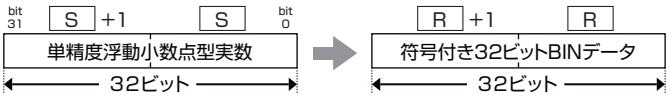
  −1.79E+308 ≦ N ≦ −2.23E−308
  N = 0
  2.23E−308 ≦ N ≦ 1.79E+308

  (有効桁数:約16桁)

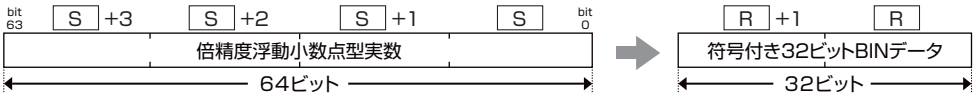
動作説明

S に格納された単精度/倍精度浮動小数点型実数データを、32ビットバイナリデータ(符号付き)に変換し、結果を R に格納します。

- **S :変換対象の型が単精度浮動小数点型実数の場合**



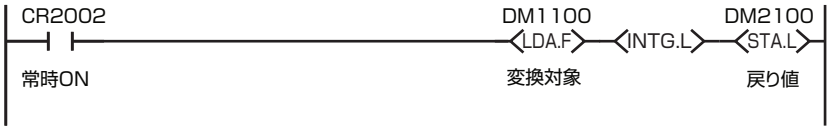
- **S :変換対象の型が倍精度浮動小数点型実数の場合**



詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「浮動小数命令」を参照してください。

●書式例

DM2100.L = **INTG** (DM1100.F)



ポイント

- **S が単精度浮動小数点型実数の範囲外の場合、また、変換したデータが16/32ビット符号付きバイナリの範囲外の場合、S のビットパターンが R に格納されます。**
- **S が倍精度浮動小数点型実数の範囲外の場合、また、変換したデータが16/32ビット符号付きバイナリの範囲外の場合、S の下位32ビットのビットパターンが R に格納されます。**

サンプルプログラム

●使用例

スクリプト記述　DM2000.L = **INTG** (DM1000.F + 1.25)

動作説明　DM1000.F+1.25の単精度浮動小数点型実数の演算結果を32ビット符号付きデータに整数化し、変換結果をDM2000.Lに格納します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

FLOAT
TOF
DFLOAT
TODF
INTG
INT
DISF
UNIF
EXP
LOG
LOG10
RAD
DEG
SIN
COS
TAN
ASIN
ACOS
ATAN
ATAN2

INT 単精度浮動小数点型実数データ→BINデータ変換

戻り値※1 = **INT** (変換対象)
整数変換

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象※2	変換対象、または変換対象データを指定します。	—	—	—	—	.F	※4 .DF	—	—	○	○	○
R 戻り値	INT関数は、.Lの型を持ちます。※3	.U	.S	.D	.L	.F	※4 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　C, T, CTC, CTH, Zは使用できません。
※3　戻り値に.L以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※4　デバイスZに「.DF」サフィックスは指定できません。

ポイント

- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)
- **単精度浮動小数点型実数の範囲、および倍精度浮動小数点型実数の範囲は、「INTG関数」(7-122ページ)を参照してください。**

動作説明

本関数は「INTG関数」と関数名が異なるだけで、動作等に違いはありません。

MEMO

FLOAT
TOF
DFLOAT
TODF
INTG
INT
DISF
UNIF
EXP
LOG
LOG10
RAD
DEG
SIN
COS
TAN
ASIN
ACOS
ATAN
ATAN2

DISF 単精度浮動小数点型実数の仮数・指数分離

ディスF
DISF ([実行条件,]※1 演算対象, 仮数格納対象, 指数格納対象)

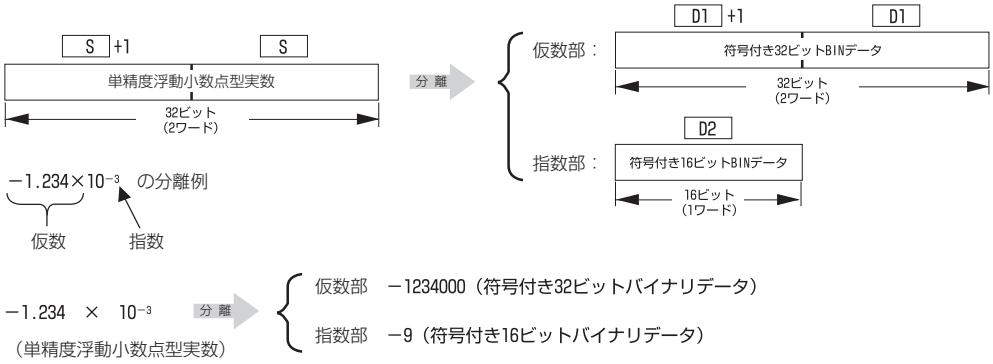
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象※2※4	仮数・指数分離する単精度浮動小数点型実数の対象を指定します。	—	—	—	—	.F	—	—	—	※3 ○	○	○
D1 仮数格納対象※2	分離した仮数部を格納する対象を指定します。	—	.L	.L	—	—	—	—	—		○	
D2 指数格納対象	分離した指数部を格納する対象を指定します。	.S	.S	—	—	—	—	—	—		○	
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ビットデータを指定した場合、連続する32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって扱います。
※3 定数の設定は、単精度浮動小数点型実数の範囲です。
※4 C, T, CTC, CTH, Zは使用できません。

ポイント **単精度浮動小数点型実数の範囲は、「INTG関数」(7-122ページ)を参照してください。**

動作説明

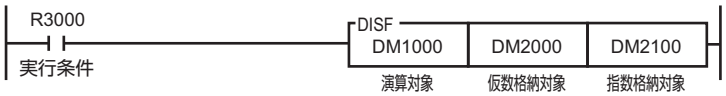
実行条件が成り立つ(TRUE)とき、S で指定された32ビット単精度浮動小数点型実数の仮数部と指数部を分離して、仮数部を指定されたD1 へ32ビットバイナリデータ(符号付き)として格納し、指数部を指定されたD2 へ16ビットバイナリデータ(符号付き)として格納します。



「演算フラグについて(CR2009〜CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「浮動小数命令」を参照してください。

●書式例

DISF (R3000, DM1000.F, DM2000.L, DM2100.S)



ポイント

- **DISF関数では、単精度浮動小数点型実数(2進数)から、仮数部(10進数)と指数部(10進数)に変換しているため、誤差が生じる場合があります。**
- **分離するデータが単精度浮動小数点型実数の範囲外の場合、関数は実行されません。**

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述

DISF (LOG (DM3000.F), EM3000.L, EM3100)

動作説明

毎スキャン(実行条件無し)、DM3000.Fに格納されているを単精度浮動小数点型実数の自然対数を求めた結果の値の仮数部と指数部を分離して、仮数部をEM3000.Lに、指数部をEM3100.Sに格納します。

参 考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「LOG関数」(7-132ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

FLOAT
TOF
DFLOAT
TODF
INTG
INT
DISF
UNIF
EXP
LOG
LOG10
RAD
DEG
SIN
COS
TAN
ASIN
ACOS
ATAN
ATAN2

UNIF 仮数・指数の単精度浮動小数点型実数への結合

戻り値※1 = **UNIF** (ユニF) (仮数格納対象, 指数格納対象)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 仮数格納対象※2	結合する仮数部を格納した対象を指定します。	—	—	.L	.L	—	—	—	—	—	○	○
S2 指数格納対象	結合する指数部を格納した対象を指定します。	.S	.S	.S	.S	—	—	—	—	—	○	○
R 戻り値	UNIF関数は、.Fの型を持ちます。※3	.U	.S	.D	.L	.F	※4 .DF	—	—	—	○	—

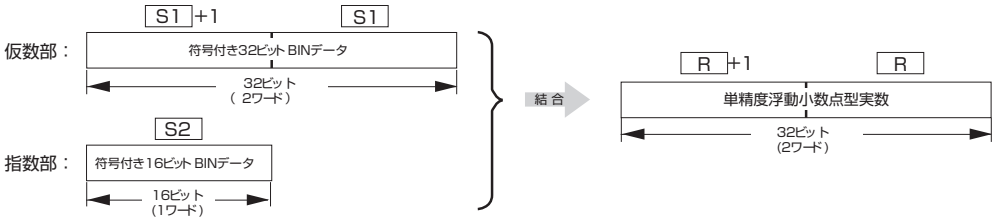
※1 戻り値は省略できます。省略した場合、演算結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータを指定した場合、連続する32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって扱います。
※3 戻り値に.F以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※4 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

S1 で指定された32ビットバイナリデータ(符号付き)を仮数とし、S2 で指定された16ビットバイナリデータ(符号付き)を指数として結合した、32ビット単精度浮動小数点型実数値を R に格納します。



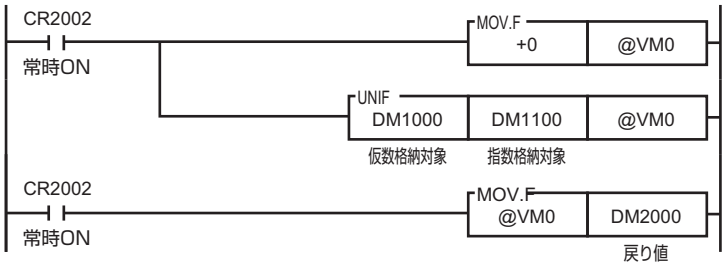
結合例
仮数部 −1234 (符号付き32ビット バイナリデータ)
指数部 −6 (符号付き16ビット バイナリデータ)
結合 → -1.234×10^{-3} (単精度浮動小数点型実数)

「演算フラグについて(CR2009〜CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「浮動小数命令」を参照してください。

●書式例

DM2000.F = **UNIF** (DM1000.L, DM1100.S)



ポイント

- **UNIF関数では、仮数部(10進数)と指数部(10進数)から、単精度浮動小数点型実数(2進数)に変換しているため、誤差が生じる場合があります。**
- **演算結果が単精度浮動小数点型実数の範囲外の場合、R には0が格納されます。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　DM2000.F = EXP (**UNIF** (EM3000.L, EM3100.S))

動作説明　EM3000(.L)に格納されている仮数部と、EM3100(.S)に格納されている指数部を結合した単精度浮動小数点型実数値を指数とした指数関数を求め、結果をDM2000.Fに格納します。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「EXP関数」(7-130ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

FLOAT
TOF
DFLOAT
TODF
INTG
INT
DISF
UNIF
EXP
LOG
LOG10
RAD
DEG
SIN
COS
TAN
ASIN
ACOS
ATAN
ATAN2

EXP 指数関数演算

戻り値※1 = **EXP** (演算対象)

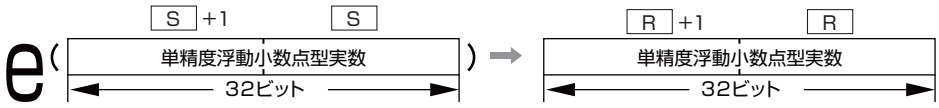
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.F	.F	.F	.F	.F	※3 .DF	—	—	○	○	○
R 戻り値	EXP関数は、引数がU/S/D/L/Fの場合.Fの型を持ち、DFの場合.DFの型を持ちます。 引数の型は、自動的に関数の型に変換されます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に関数が持つ型と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

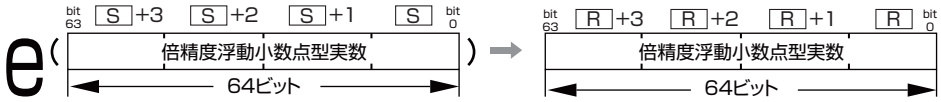
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)
- **単精度浮動小数点型実数の範囲、および倍精度浮動小数点型実数の範囲は、「INTG関数」(7-122ページ)を参照してください。**

動作説明

- **S :変換対象のサフィックスが.U/.S/.D/.L/.Fの場合**
  [S +1・S]に格納された単精度浮動小数点型実数を指数とした指数関数値を求めます。
  結果は32ビット単精度浮動小数点型実数値となり、[R +1・R]に格納されます。



- **S :変換対象のサフィックスが.DFの場合**
  [S +3・S +2・S +1・S]に格納された倍精度浮動小数点型実数を指数とした指数関数値を求めます。
  結果は64ビット倍精度浮動小数点型実数値となり、[R +3・R +2・R +1・R]に格納されます。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「浮動小数命令」を参照してください。

●書式例

DM2100.F = **EXP** (DM1100.F)

ポイント　**演算する値、または演算結果が単精度浮動小数点型実数の範囲外の場合、R には S の値が格納されます。**

サンプルプログラム

●使用例

スクリプト記述　DM2000.F = **EXP** (DM1000.D)

動作説明　DM1000.Dに格納された32ビットバイナリデータは自動的に単精度浮動小数点型実数にされ結果の値を指数とし、e^xを求めてDM2000. F に格納します。(DM1000.DをFLOAT関数等で変換する必要はありません)

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「FLOAT関数」(7-114ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

FLOAT
TOF
DFLOAT
TODF
INTG
INT
DISF
UNIF
EXP
LOG
LOG10
RAD
DEG
SIN
COS
TAN
ASIN
ACOS
ATAN
ATAN2

LOG 自然対数演算

戻り値※1 = **LOG** (演算対象)

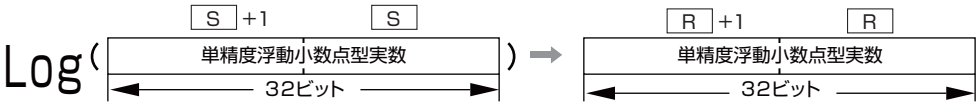
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.F	.F	.F	.F	.F	※3 .DF	—	—	○	○	○
R 戻り値	LOG関数は、引数がU/S/D/L/Fの場合.Fの型を持ち、DFの場合.DFの型を持ちます。 引数の型は、自動的に関数の型に変換されます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に関数が持つ型と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

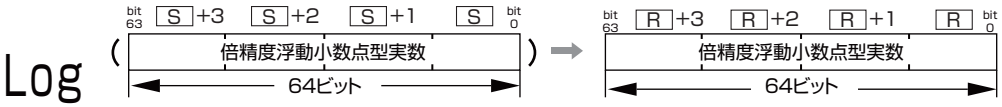
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)
- **単精度浮動小数点型実数の範囲、および倍精度浮動小数点型実数の範囲は、「INTG関数」(7-122ページ)を参照してください。**

動作説明

- **S :変換対象のサフィックスが.U/.S/.D/.L/.Fの場合**
  [S+1・S]に格納された単精度浮動小数点型実数の自然対数を求めます。
  結果は32ビット単精度浮動小数点型実数値となり、[R+1・R]に格納されます。



- **S :変換対象のサフィックスが.DFの場合**
  [S+3・S+2・S+1・S]に格納された倍精度浮動小数点型実数の自然対数を求めます。
  結果は64ビット倍精度浮動小数点型実数値となり、[R+3・R+2・R+1・R]に格納されます。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「浮動小数命令」を参照してください。

●書式例

DM2100.F = **LOG** (DM1100.F)

ポイント　**演算する値が0か負、または演算結果が浮動小数点型実数の範囲外の場合、 R には S の値が格納されます。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　DM2000.L = INTG (**LOG** (DM1000.F))

動作説明　DM1000.Fに格納された単精度浮動小数点型実数の自然対数を求めた結果の値を、32ビット符号付きバイナリデータに変換してDM2000.Lに格納します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「INTG関数」(7-122ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

FLOAT
TOF
DFLOAT
TODF
INTG
INT
DISF
UNIF
EXP
LOG
LOG10
RAD
DEG
SIN
COS
TAN
ASIN
ACOS
ATAN
ATAN2

LOG10 常用対数演算

戻り値※1 = **LOG10** (演算対象)
ログ10

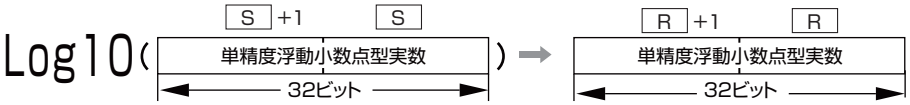
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.F	.F	.F	.F	.F	※3 .DF	—	—	○	○	○
R 戻り値	LOG10関数は、引数がU/S/D/L/Fの場合.Fの型を持ち、DFの場合.DFの型を持ちます。引数の型は、自動的に関数の型に変換されます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に関数が持つ型と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

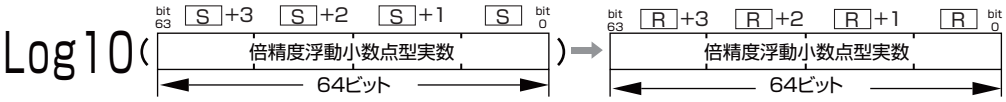
ポイント
- **LOG10関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000 シリーズ CPU ユニット、CPU 機能バージョン 2.0 以降の KV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)
- **単精度浮動小数点型実数の範囲、および倍精度浮動小数点型実数の範囲は、****「INTG関数」(7-122ページ)を参照してください。**

動作説明

- **S :変換対象のサフィックスが.U/.S/.D/.L/.Fの場合**
  [S+1・S]に格納された単精度浮動小数点型実数の常用対数を求めます。
  結果は32ビット単精度浮動小数点型実数値となり、[R+1・R]に格納されます。



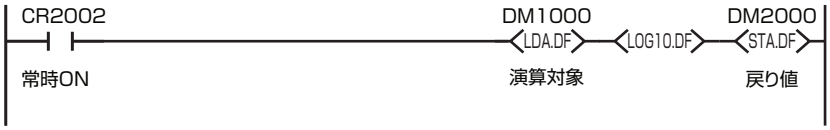
- **S :変換対象のサフィックスが.DFの場合**
  [S+3・S+2・S+1・S]に格納された倍精度浮動小数点型実数の常用対数を求めます。
  結果は64ビット倍精度浮動小数点型実数値となり、[R+3・R+2・R+1・R]に格納されます。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「浮動小数命令」を参照してください。

●書式例

DM2000.DF = **Log10** (DM1000.DF)



サンプルプログラム

スクリプト記述　DM2500.L = INTG (**LOG10** (DM1500.DF))

動作説明　DM1500.DFに格納された倍精度浮動小数点型実数の自然対数を求めた結果の値を、32ビット符号付きバイナリデータに変換してDM2500.Lに格納します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

FLOAT
TOF
DFLOAT
TODF
INTG
INT
DISF
UNIF
EXP
LOG
LOG10
RAD
DEG
SIN
COS
TAN
ASIN
ACOS
ATAN
ATAN2

RAD 度(°)→ラジアン単位変換

戻り値※1 = **RAD** (変換対象)

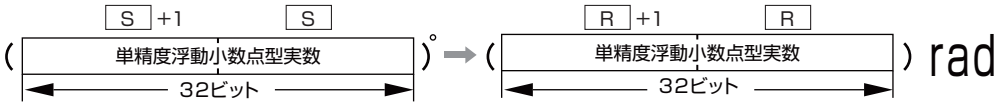
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象	変換対象、または変換対象データを指定します。	.F	.F	.F	.F	.F	※3 .DF	—	—	○	○	○
R 戻り値	RAD関数は、引数がU/S/D/L/Fの場合.Fの型を持ち、DFの場合.DFの型を持ちます。引数の型は、自動的に関数の型に変換されます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、変換結果は内部レジスタに格納されます。
※2　戻り値に関数が持つ型と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

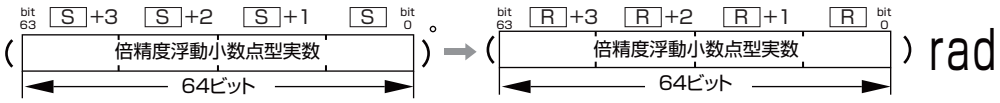
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)
- **単精度浮動小数点型実数の範囲、および倍精度浮動小数点型実数の範囲は、「INTG関数」(7-122ページ)を参照してください。**

動作説明

- **S :変換対象のサフィックスが.U/.S/.D/.L/.Fの場合**
  [S +1・S]に格納された単精度浮動小数点型実数の角度(°単位)をラジアン単位に変換します。
  結果は32ビット単精度浮動小数点型実数値となり、[R +1・R]に格納されます。



- **S :変換対象のサフィックスが.DFの場合**
  [S +3・S +2・S +1・S]に格納された倍精度浮動小数点型実数の角度(°単位)をラジアン単位に変換します。
  結果は64ビット倍精度浮動小数点型実数値となり、[R +3・R +2・R +1・R]に格納されます。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「浮動小数命令」を参照してください。

参 考　角度(°)と、(rad)は次のように計算されます。

$$1^\circ = \frac{\pi}{180} \fallingdotseq 0.01745\ (\mathrm{rad}),\quad 1\ (\mathrm{rad}) = \frac{180}{\pi} \fallingdotseq 57.23^\circ$$ です。

●書式例

DM2100.F = **RAD** (DM1100.F)

ポイント　**演算する値が単精度浮動小数点型実数の範囲外の場合、R には S の値が格納されます。**

サンプルプログラム

●使用例

スクリプト記述　DM2000.F = **RAD** (DM1000.D)

動作説明　DM1000.Dに格納された32ビットバイナリデータの角度(°単位)は自動的に単精度浮動小数点型実数に変換され、結果をラジアン単位に変換してDM2000.Fに格納します。
(DM1000.DをFLOAT関数等で変換する必要はありません)

参 考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

FLOAT
TOF
DFLOAT
TODF
INTG
INT
DISF
UNIF
EXP
LOG
LOG10
RAD
DEG
SIN
COS
TAN
ASIN
ACOS
ATAN
ATAN2

DEG　ラジアン→度(°)単位変換

戻り値※1 = **DEG** (変換対象)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象	変換対象、または変換対象データを指定します。	.F	.F	.F	.F	.F	※3 .DF	—	—	○	○	○
R 戻り値	DEG関数は、引数がU/S/D/L/Fの場合.Fの型を持ち、DFの場合.DFの型を持ちます。 引数の型は、自動的に関数の型に変換されます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、変換結果は内部レジスタに格納されます。
※2　戻り値に関数が持つ型と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)
- **単精度浮動小数点型実数の範囲、および倍精度浮動小数点型実数の範囲は、「INTG関数」(7-122ページ)を参照してください。**

動作説明

- **S :変換対象のサフィックスが.U/.S/.D/.L/.Fの場合**
  [S +1・S]に格納された単精度浮動小数点型実数の角度(ラジアン単位)を度(°)単位に変換します。
  結果は32ビット単精度浮動小数点型実数値となり、[R +1・R]に格納されます。

- **S :変換対象のサフィックスが.DFの場合**
  [S +3・S +2・S +1・S]に格納された倍精度浮動小数点型実数の角度(ラジアン単位)を度(°)単位に変換します。
  結果は64ビット倍精度浮動小数点型実数値となり、[R +3・R +2・R +1・R]に格納されます。

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「浮動小数命令」を参照してください。

参考　角度(°)と、(rad)は次のように計算されます。

$$1^\circ = \frac{\pi}{180} \fallingdotseq 0.01745\ (\mathrm{rad}),\quad 1\ (\mathrm{rad}) = \frac{180}{\pi} \fallingdotseq 57.23^\circ$$ です。

●書式例

DM2100.F = **DEG** (DM1100.F)

ポイント　**演算する値が単精度浮動小数点型実数の範囲外の場合、R には S の値が格納されます。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　DM2000.L = INTG (**DEG** (DM1000.F))

動作説明　DM1000.Fに格納された単精度浮動小数点型実数の角度(ラジアン単位)を度(°)単位に変換し、結果を32ビット符号付きバイナリデータに変換してDM2000.Lに格納します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「INTG関数」(7-122ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

FLOAT
TOF
DFLOAT
TODF
INTG
INT
DISF
UNIF
EXP
LOG
LOG10
RAD
DEG
SIN
COS
TAN
ASIN
ACOS
ATAN
ATAN2

SIN 角度(ラジアン)→正弦値

戻り値※1 = **SIN** (演算対象)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.F	.F	.F	.F	.F	※3 .DF	—	—	○	○	○
R 戻り値	SIN関数は、引数がU/S/D/L/Fの場合.Fの型を持ち、DFの場合.DFの型を持ちます。 引数の型は、自動的に関数の型に変換されます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に関数が持つ型と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

ポイント

- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)
- **この関数で扱える単精度浮動小数点型実数の範囲は、**

  −1.6777216E+7 ≦ N ≦ −1.4E−45
  N ＝ 0
  1.4E−45 ≦ N ≦ 1.6777216E+7
  (KV Nanoシリーズの場合、絶対値が1.4E−45〜1.2E−38の範囲は使用できません。)

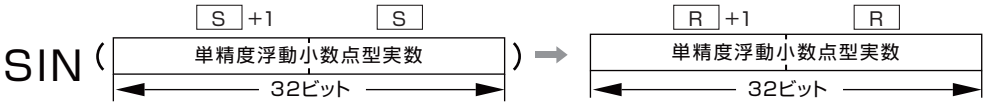
  (有効桁数:約7桁)です。
- **この関数で扱える倍精度浮動小数点型実数の範囲は、**

  −9.007199254740992E+15 ≦ N ≦ −2.23E−308
  N ＝ 0
  +2.23E−308 ≦ N ≦ +9.007199254740992E+15

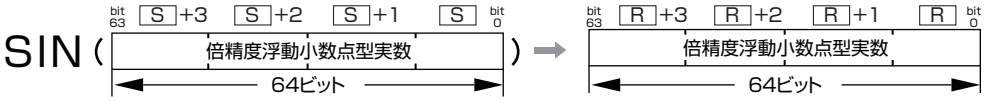
  (有効桁数:約16桁)です。

動作説明

- **S :変換対象のサフィックスが.U/.S/.D/.L/.Fの場合**
  [S +1・S]に格納された単精度浮動小数点型実数の角度(ラジアン単位)の正弦(SIN)値を求めます。
  結果は32ビット単精度浮動小数点型実数値となり、[R +1・R]に格納されます。



- **[S]:変換対象のサフィックスが.DFの場合**
  [S +3· S +2· S +1· S]に格納された倍精度浮動小数点型実数の角度(ラジアン単位)の正弦(SIN)値を求めます。
  結果は64ビット倍精度浮動小数点型実数値となり、[R +3· R +2· R +1· R]に格納されます。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「浮動小数命令」を参照してください。

参 考　角度(°)と、(rad)は次のように計算されます。

$$1^\circ = \frac{\pi}{180} \fallingdotseq 0.01745\ (\mathrm{rad}),\quad 1\ (\mathrm{rad}) = \frac{180}{\pi} \fallingdotseq 57.23^\circ$$ です。

●書式例

DM2100.F = **SIN** (DM1100.F)

ポイント　**演算する値がSIN関数で扱える単精度浮動小数点型実数の範囲外の場合、[R]には[S]の値が格納されます。**

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述　DM2000.F = **SIN** (RAD (DM1000.F))

動作説明　DM1000.Fに格納された単精度浮動小数点型実数の角度(°単位)をラジアン単位に変換し、正弦値を求めて結果をDM2000.Fに格納します。

参 考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「RAD関数」(7-136ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

COS 角度(ラジアン)→余弦値

戻り値※1 = **COS** (演算対象)
(コサイン)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.F	.F	.F	.F	.F	※3 .DF	—	—	○	○	○
R 戻り値	COS関数は、引数がU/S/D/L/Fの場合.Fの型を持ち、DFの場合.DFの型を持ちます。 引数の型は、自動的に関数の型に変換されます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に関数が持つ型と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

ポイント

- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)
- **この関数で扱える単精度浮動小数点型実数の範囲は、**

  −1.6777216E+7 ≦ N ≦ −1.4E−45
  N ＝ 0
  1.4E−45 ≦ N ≦ 1.6777216E+7
  (KV Nanoシリーズの場合、絶対値が1.4E−45〜1.2E−38の範囲は使用できません。)

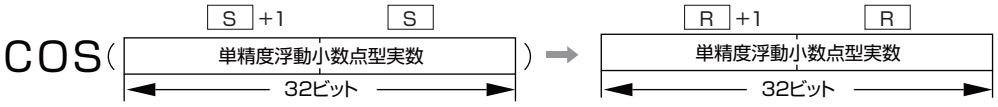
  (有効桁数:約7桁)です。
- **この関数で扱える倍精度浮動小数点型実数の範囲は、**

  −9.007199254740992E+15 ≦ N ≦ −2.23E−308
  N ＝ 0
  +2.23E−308 ≦ N ≦ +9.007199254740992E+15

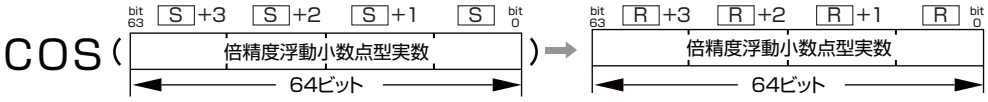
  (有効桁数:約16桁)です。

動作説明

- **S :変換対象のサフィックスが.U/.S/.D/.L/.Fの場合**
  [S +1・S]に格納された単精度浮動小数点型実数の角度(ラジアン単位)の余弦(COS)値を求めます。
  結果は32ビット単精度浮動小数点型実数値となり、[R +1・R]に格納されます。



- **[S]:変換対象のサフィックスが.DFの場合**
  [S +3· S +2· S +1· S]に格納された倍精度浮動小数点型実数の角度(ラジアン単位)の余弦(COS)値を求めます。
  結果は64ビット倍精度浮動小数点型実数値となり、[R +3· R +2· R +1· R]に格納されます。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「浮動小数命令」を参照してください。

参考　角度(°)と、(rad)は次のように計算されます。

$$1^\circ = \frac{\pi}{180} \fallingdotseq 0.01745\ (\mathrm{rad}),\quad 1\ (\mathrm{rad}) = \frac{180}{\pi} \fallingdotseq 57.23^\circ$$ です。

●書式例

DM2100.F = **COS** (DM1100.F)

ポイント　**演算する値がCOS関数で扱える単精度浮動小数点型実数の範囲外の場合、[R]には[S]の値が格納されます。**

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述　DM2000.F = **COS** (RAD (DM1000.F))

動作説明　DM1000.Fに格納された単精度浮動小数点型実数の角度(°単位)をラジアン単位に変換し、余弦値を求めて結果をDM2000.Fに格納します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「RAD関数」(7-136ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

TAN 角度(ラジアン)→正接値

戻り値※1 = **TAN** (タンジェント) **(演算対象)**

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.F	.F	.F	.F	.F	※3 .DF	—	—	○	○	○
R 戻り値	TAN関数は、引数がU/S/D/L/Fの場合.Fの型を持ち、DFの場合.DFの型を持ちます。 引数の型は、自動的に関数の型に変換されます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に関数が持つ型と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

ポイント

- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)
- **この関数で扱える単精度浮動小数点型実数の範囲は、**

  −1.6777216E+7 ≦ N ≦ −1.4E−45
  N ＝ 0
  1.4E−45 ≦ N ≦ 1.6777216E+7
  (KV Nanoシリーズの場合、絶対値が1.4E−45〜1.2E−38の範囲は使用できません。)

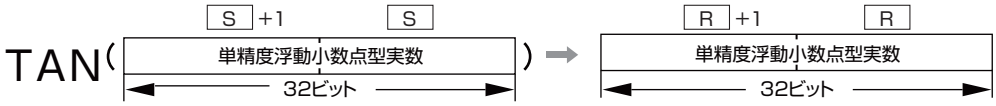
  (有効桁数:約7桁)です。
- **この関数で扱える倍精度浮動小数点型実数の範囲は、**

  −9.007199254740992E+15 ≦ N ≦ −2.23E−308
  N ＝ 0
  +2.23E−308 ≦ N ≦ +9.007199254740992E+15

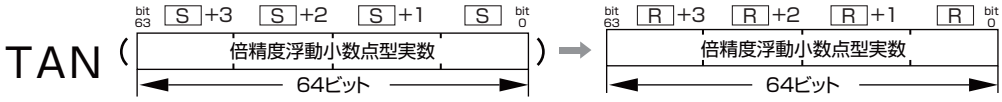
  (有効桁数:約16桁)です。

動作説明

- **S :変換対象のサフィックスが.U/.S/.D/.L/.Fの場合**
  [S +1・S]に格納された単精度浮動小数点型実数の角度(ラジアン単位)の正接(TAN)値を求めます。
  結果は32ビット単精度浮動小数点型実数値となり、[R +1・R]に格納されます。



- **[S]:変換対象のサフィックスが.DFの場合**
  [S +3· S +2· S +1· S]に格納された倍精度浮動小数点型実数の角度(ラジアン単位)の正接(TAN)値を求めます。
  結果は64ビット倍精度浮動小数点型実数値となり、[R +3· R +2· R +1· R]に格納されます。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「浮動小数命令」を参照してください。

参考　角度(°)と、(rad)は次のように計算されます。

$$1^\circ = \frac{\pi}{180} \fallingdotseq 0.01745\ (\mathrm{rad}),\quad 1\ (\mathrm{rad}) = \frac{180}{\pi} \fallingdotseq 57.23^\circ$$ です。

●書式例

DM2100.F = **TAN** (DM1100.F)

ポイント　**演算する値がTAN関数で扱える単精度浮動小数点型実数の範囲外の場合、[R]には[S]の値が格納されます。**

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述　DM2000.F = **TAN** (RAD (DM1000.F))

動作説明　DM1000.Fに格納された単精度浮動小数点型実数の角度(°単位)をラジアン単位に変換し、正接値を求めて結果をDM2000.Fに格納します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「RAD関数」(7-136ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

FLOAT
TOF
DFLOAT
TODF
INTG
INT
DISF
UNIF
EXP
LOG
LOG10
RAD
DEG
SIN
COS
TAN
ASIN
ACOS
ATAN
ATAN2

ASIN 正弦値→角度(ラジアン)

戻り値※1 = **ASIN** (演算対象)
(アークサイン)

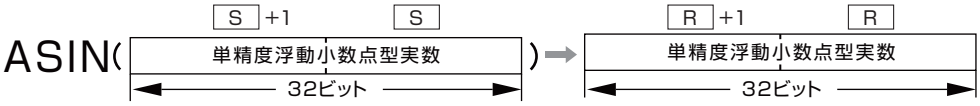
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.F	.F	.F	.F	.F	※3 .DF	—	—	○	○	○
R 戻り値	ASIN関数は、引数がU/S/D/L/Fの場合.Fの型を持ち、DFの場合.DFの型を持ちます。 引数の型は、自動的に関数の型に変換されます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1 戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2 戻り値に関数が持つ型と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3 デバイスZに「.DF」サフィックスは指定できません。

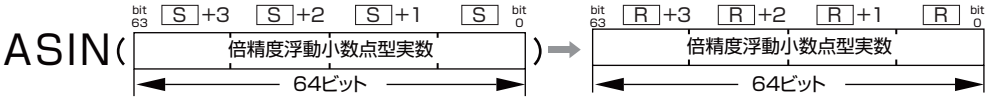
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)
- **単精度浮動小数点型実数の範囲、および倍精度浮動小数点型実数の範囲は、「INTG関数」(7-122ページ)を参照してください。**

動作説明

- **S :変換対象のサフィックスが.U/.S/.D/.L/.Fの場合**
  [S +1・S]に格納された単精度浮動小数点型実数の正弦(SIN)値から角度(ラジアン単位)を求めます。
  結果は32ビット単精度浮動小数点型実数値となり、[R +1・R]に格納されます。



- **S :変換対象のサフィックスが.DFの場合**
  [S +3・S +2・S +1・S]に格納された倍精度浮動小数点型実数の正弦(SIN)値から角度(ラジアン単位)を求めます。
  結果は64ビット倍精度浮動小数点型実数値となり、[R +3・R +2・R +1・R]に格納されます。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「浮動小数命令」を参照してください。

参 考　角度(°)と、(rad)は次のように計算されます。

$$1^\circ = \frac{\pi}{180} \fallingdotseq 0.01745\ (\mathrm{rad}),\quad 1\ (\mathrm{rad}) = \frac{180}{\pi} \fallingdotseq 57.23^\circ$$ です。

●書式例

DM2100.F = **ASIN** (DM1100.F)

ポイント　**演算する値が－1.0～1.0の範囲外の場合、R には S の値が格納されます。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　DM2000.F = DEG (**ASIN** (DM1000.F))

動作説明　DM1000.Fに格納された単精度浮動小数点型実数の正弦値から角度(ラジアン単位)を求め、結果を度(°)単位に変換してDM2000.Fに格納します。

参 考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「DEG関数」(7-138ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ACOS 余弦値→角度(ラジアン)

戻り値※1 = **ACOS** (演算対象)
アークコサイン

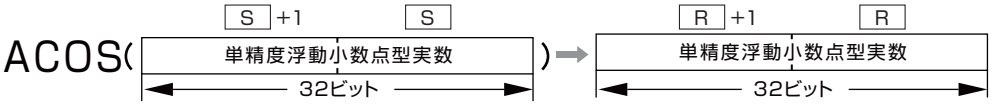
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.F	.F	.F	.F	.F	※3 .DF	—	—	○	○	○
R 戻り値	ACOS関数は、引数がU/S/D/L/Fの場合.Fの型を持ち、DFの場合.DFの型を持ちます。 引数の型は、自動的に関数の型に変換されます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1 戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2 戻り値に関数が持つ型と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3 デバイスZに「.DF」サフィックスは指定できません。

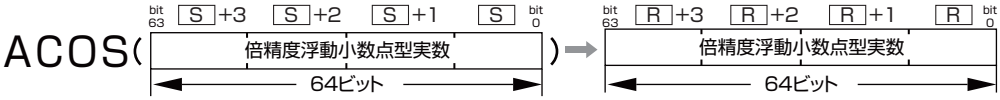
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)
- **単精度浮動小数点型実数の範囲、および倍精度浮動小数点型実数の範囲は、「INTG関数」(7-122ページ)を参照してください。**

動作説明

- **S :変換対象のサフィックスが.U/.S/.D/.L/.Fの場合**
  [S +1・S]に格納された単精度浮動小数点型実数の余弦(COS)値から角度(ラジアン単位)を求めます。
  結果は32ビット単精度浮動小数点型実数値となり、[R +1・R]に格納されます。



- **S :変換対象のサフィックスが.DFの場合**
  [S +3・S +2・S +1・S]に格納された倍精度浮動小数点型実数の余弦(COS)値から角度(ラジアン単位)を求めます。
  結果は64ビット倍精度浮動小数点型実数値となり、[R +3・R +2・R +1・R]に格納されます。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「浮動小数命令」を参照してください。

参 考　角度(°)と、(rad)は次のように計算されます。

$$1^\circ = \frac{\pi}{180} \fallingdotseq 0.01745\ (\mathrm{rad}),\quad 1\ (\mathrm{rad}) = \frac{180}{\pi} \fallingdotseq 57.23^\circ$$ です。

●書式例

DM2100.F = **ACOS** (DM1100.F)

ポイント　**演算する値が－1.0～1.0の範囲外の場合、R には S の値が格納されます。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　DM2000.F = DEG (**ACOS** (DM1000.F))

動作説明　DM1000.Fに格納された単精度浮動小数点型実数の余弦値から角度(ラジアン単位)を求め、結果を度(°)単位に変換してDM2000.Fに格納します。

参 考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「DEG関数」(7-138ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ATAN 正接値→角度(ラジアン)

戻り値※1 = **ATAN** (演算対象)

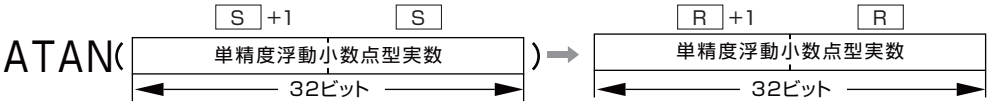
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.F	.F	.F	.F	.F	※3 .DF	—	—	○	○	○
R 戻り値	ATAN関数は、引数がU/S/D/L/Fの場合.Fの型を持ち、DFの場合.DFの型を持ちます。 引数の型は、自動的に関数の型に変換されます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1 戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2 戻り値に関数が持つ型と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3 デバイスZに「.DF」サフィックスは指定できません。

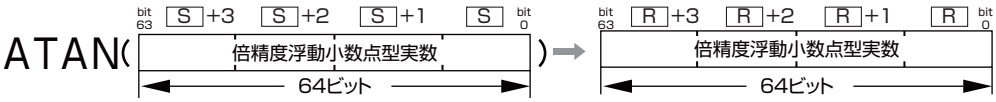
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)
- **単精度浮動小数点型実数の範囲、および倍精度浮動小数点型実数の範囲は、「INTG関数」(7-122ページ)を参照してください。**

動作説明

- **S :変換対象のサフィックスが.U/.S/.D/.L/.Fの場合**
  [S +1・S]に格納された単精度浮動小数点型実数の正接(TAN)値から角度(ラジアン単位)を求めます。
  結果は32ビット単精度浮動小数点型実数値となり、[R +1・R]に格納されます。



- **S :変換対象のサフィックスが.DFの場合**
  [S +3・S +2・S +1・S]に格納された倍精度浮動小数点型実数の正接(TAN)値から角度(ラジアン単位)を求めます。
  結果は64ビット倍精度浮動小数点型実数値となり、[R +3・R +2・R +1・R]に格納されます。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「浮動小数命令」を参照してください。

参 考　角度(°)と、(rad)は次のように計算されます。

$$1^\circ = \frac{\pi}{180} \fallingdotseq 0.01745\ (\mathrm{rad}),\quad 1\ (\mathrm{rad}) = \frac{180}{\pi} \fallingdotseq 57.23^\circ$$ です。

●書式例

DM2100.F = **ATAN** (DM1100.F)

ポイント　**演算する値が単精度浮動小数点型実数の範囲外の場合、R には S の値が格納されます。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　DM2000.F = DEG (**ATAN** (DM1000.F))

動作説明　DM1000.Fに格納された単精度浮動小数点型実数の正接値から角度(ラジアン単位)を求め、結果を度(°)単位に変換してDM2000.Fに格納します。

参 考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「DEG関数」(7-138ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ATAN2 座標→角度(ラジアン)

戻り値※1 = **ATAN2** (演算対象1, 演算対象2)
アークタンジェント2

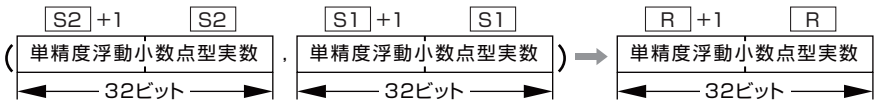
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 演算対象1	演算対象1、または演算対象データ1を指定します。	.F	.F	.DF	.DF	.F	※3 .DF	—	—	○	○	○
S2 演算対象2	演算対象2、または演算対象データ2を指定します。	.F	.F	.DF	.DF	.F	※3 .DF	—	—	○	○	○
R 戻り値	ATAN2関数は、引数がU/S/Fの場合.Fの型を持ち、D/L/DFの場合.DFの型を持ちます。 引数の型は、自動的に関数の型に変換されます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に関数が持つ型と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

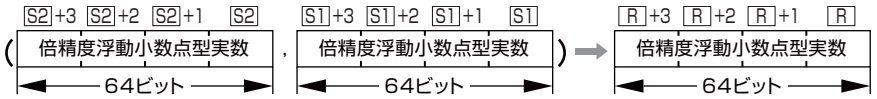
ポイント
- **ATAN2関数は、KV STUDIO Ver.9.0以降のバージョンでのみ使用できます。**
- **KV-8000シリーズCPUユニットおよびCPU機能バージョン2.0以降のKV-7000シリーズCPUユニットでのみ指定できます。**
  「KV-7500/7300　CPU機能バージョンについて」(3ページ)
- **単精度浮動小数点型実数の範囲、および倍精度浮動小数点型実数の範囲は、「INTG関数」(7-122ページ)を参照してください。**

動作説明

- **S1:変換対象1、S2:変換対象2のサフィックスが.U/.S/.Fの場合**
  [S1+1・S1]で指定した値を単精度浮動小数点型実数に変換したY座標と[S2+1・S2]で指定した値を単精度浮動小数点型実数に変換したX座標から、単精度浮動小数点型実数のX軸に対する角度(ラジアン単位)を求めます。
  結果は32ビット単精度浮動小数点型実数値となり、[R+1・R]に格納されます。

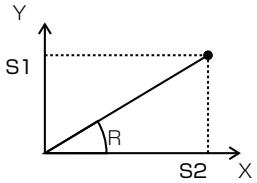


- **S1:変換対象1、S2:変換対象2のサフィックスが.D/.L/.DFの場合**
  [S1+3・S1+2・S1+1・S1]で指定した値を倍精度浮動小数点型実数に変換したY座標と[S2+3・S2+2・S2+1・S2]で指定した値を倍精度浮動小数点型実数に変換したX座標から、倍精度浮動小数点型実数のX軸に対する角度(ラジアン単位)を求めます。
  結果は64ビット倍精度浮動小数点型実数値となり、[R+3・R+2・R+1・R]に格納されます。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「浮動小数命令」を参照してください。

参 考　ATAN命令では、正接(TAN)値を使用することで、-π/2(rad)＜ R ＜ π/2(rad)までの角度を求められたのに対し、ATAN2命令では、座標(S2,S1)を使用することで、-π(rad)＜ R ≦ π(rad)までの角度を求められます。



$1^\circ = \frac{\pi}{180} \fallingdotseq 0.01745$ (rad)、 1 (rad) $= \frac{180}{\pi} \fallingdotseq 57.23^\circ$ です。

S1=0、S2≧0の場合、R(戻り値)=0°=0(rad)となります。

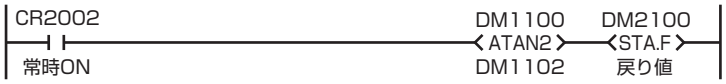
S1=0、S2<0の場合、R(戻り値)=180°=π(rad)となります。

S1>0、S2=0の場合、R(戻り値)=90°=π/2(rad)となります。

S1<0、S2=0の場合、R(戻り値)=-90°=-π/2(rad)となります。

●書式例

DM2100.F = **ATAN2** (DM1100.F, DM1102.F)



ポイント　**演算する値が単精度浮動小数点型実数の範囲外の場合、R の値は変更されません。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　DM2000.F = DEG (**ATAN2** (DM1000.F, DM1002.F))

動作説明　DM1000.Fに格納された単精度浮動小数点型実数のX座標とDM1002.Fに格納された単精度浮動小数点型実数のY座標から、単精度浮動小数点型実数のX軸に対する角度(ラジアン単位)を求め、結果を度(°)単位に変換してDM2000.Fに格納します。

参 考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「DEG関数」(7-138ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ASC　BINデータ→16進アスキー文字コード変換

16進アスキーコード変換
戻り値※1 = **ASC** (変換対象)

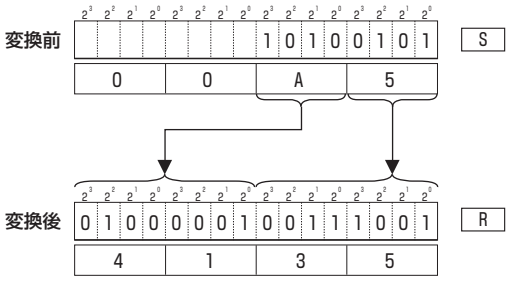
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象※2	変換対象の16進アスキー文字コードを格納した対象、または変換対象データを指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	ASC関数は、.Uの型を持ちます。※3	.U	.S	.D	.L	.F	※4 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、変換結果は内部レジスタに格納されます。
※2　変換対象に、ビットデータのチャンネルの先頭以外(R1002, R1005等)を指定した場合でも、次のチャンネルにまたがって正常に機能します。
※3　戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※4　デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

S に格納されたバイナリデータの下位8ビットの値を、16進表記のアスキー文字コードに変換し、結果の16ビットデータを R に格納します。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.U = **ASC** (DM1000.U)

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述　DM2000.U = **ASC** (SRA (DM1000.U, 8))

動作説明　DM1000に格納されているバイナリデータを8ビット分右にシフトします。
シフト結果の下位8ビット(シフト前の上位8ビット)の値を、16進表記のアスキー文字コードに変換し、結果をDM2000に格納します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「SRA関数」(7-38ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

RASC　16進アスキー文字コード→BINデータ変換

16進アスキーコード逆変換
戻り値※1 = **RASC** (変換対象)

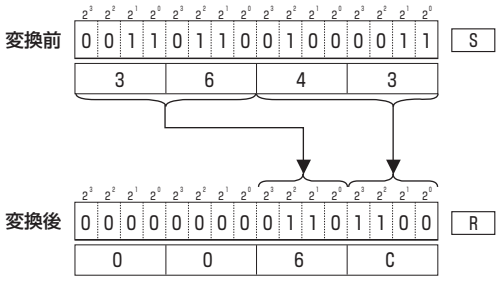
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象※2※3	変換対象のバイナリデータを格納した対象、または変換対象データを指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	RASC関数は、.Uの型を持ちます。※4	.U	.S	.D	.L	.F	※5 .DF	—	—	—	○	—

※1　戻り値は省略できます。省略した場合、変換結果は内部レジスタに格納されます。
※2　変換対象データには、32ビットデータも使用できますが、変換対象は下位16ビットのみとなります。
※3　変換対象に、ビットデータのチャンネルの先頭以外(R1002, R1005等)を指定した場合でも、次のチャンネルにまたがって正常に機能します。
※4　戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※5　デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

S に格納された16ビットの16進表記のアスキー文字コード(2文字分)を、バイナリデータに変換し、結果の16ビットデータ(下位8ビットが変換値、上位8ビットは0)を R に格納します。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
「ASCIIコード表」(付-53ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

ポイント
- **S には上位8ビット、下位8ビットとも0～9を示すアスキーコード(30H～39H)または、A～F(41H～46H)のとき、および、上位8ビットがNUL(00H)で下位8ビットが0～9(30H～39H)またはA～F(41H～46H)の値(アスキーコード)が格納されている必要があります。**
  上位8ビット、下位8ビットのどちらかがこの範囲外のとき、関数は実行しません。
- **扱えないアスキー文字コードがある場合、R には S の値が格納されます。**

●書式例

DM2000.U = **RASC** (DM1000.U)

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述　DM2000.U = **RASC** (SWAP (DM1000))

動作説明　DM1000に格納されている16進表記のアスキー文字コードの、上位バイト(上位8ビット)と下位バイト(下位8ビット)の値を入れ替えます。
入れ替えた結果で、16ビットバイナリデータに変換し、結果をDM2000に格納します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
「SWAP関数」(7-100ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

DASC　BINデータ→10進アスキー文字列変換

戻り値※1 = **DASC** (変換対象, 変換パラメータ)
10進アスキーコード変換

引数/戻り値	説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
S1 変換対象※2※3	変換対象のバイナリデータを格納した対象、または変換対象データを指定します。	.U	.S	.D	.L	—	—	—	—	※4 ○	○	○
S2 変換パラメータ※5	変換パラメータを格納した対象、またはパラメータを指定します。(0～3)	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値※3※6	DASC関数は、.Tの型を持ちます。	—	—	—	—	—	—	—	.T	—	○	—

※1　戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2　ビットデータを指定した場合、連続する16/32ビットを扱います。
チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。
※3　ビットデータではチャンネルの先頭を指定してください。
※4　定数は$で指定できません。
※5　変換パラメータは0～3の範囲で指定してください。
S2 変換パラメータの説明

値	内　容	CR2814	CR2815
0	ゼロサプレスOFF、符号省略OFF	0	0
1	ゼロサプレスON、符号省略OFF	1	0
2	ゼロサプレスOFF、符号省略ON	0	1
3	ゼロサプレスON、符号省略ON	1	1

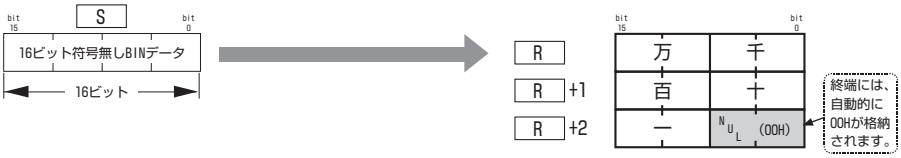
ポイント　**変換パラメータに範囲外の値を指定した場合、正しく動作しない可能性があります。**

※6　T, C, CR, CTC, CTH, Zは使用できません。

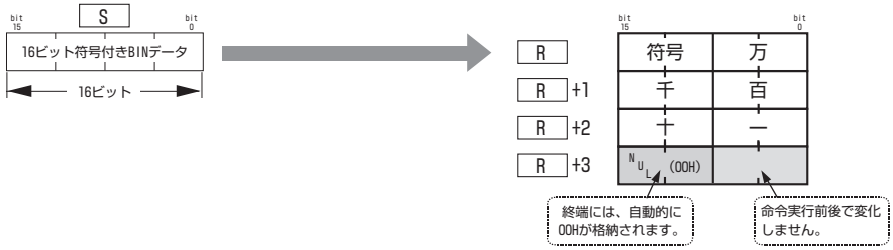
動作説明

S1 に格納されたバイナリデータを、S2 で指定された変換パラメータに従い、10進アスキーコード文字列に変換して、結果を R に格納します。

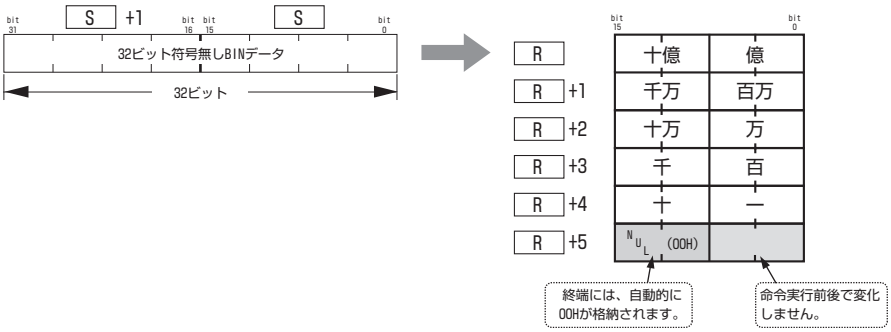
- **S1 :変換対象が16ビット符号無しデータ(.U)の場合**



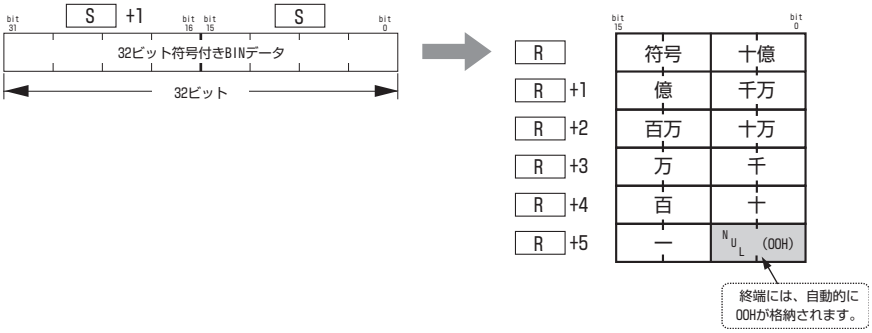
- S1 :変換対象が16ビット符号付きデータ(.S)の場合



- S1 :変換対象が32ビット符号無しデータ(.D)の場合



- S1 :変換対象が32ビット符号付きデータ(.L)の場合

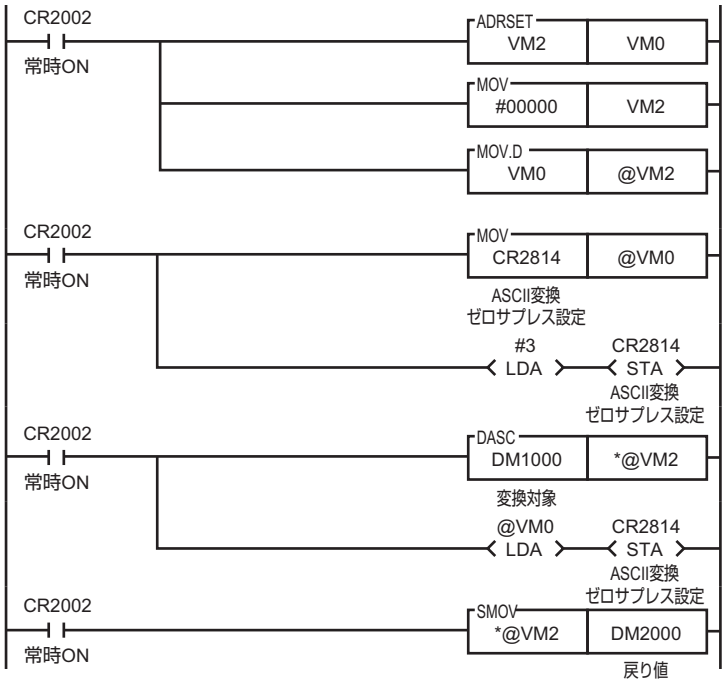


「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

●書式例

DM2000.T = **DASC** (DM1000 , 3)



ポイント

- 変換対象データに定数を使用する場合、16進($)での指定はできません。
- 変換対象データには半角文字(ASCIIコード)のみ指定できます。
- 割り込みプログラムでCR2814・CR2815を変更したり参照する場合は注意が必要です。
- 次のいずれかの条件が成立した場合、[R]には終端コード(NUL(00H))が格納されます。
  - [R]にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。
  - [R]に間接指定でタイマ・カウンタを指定した場合。

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　EM2000.T = **DASC** (TBIN (DM1000.D), 0)

動作説明　DM1000.Dに格納されている8桁BCDデータをバイナリデータに変換します。
変換した結果の値を、10進アスキーコード文字列に変換し、結果をEM2000.Tに格納します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「TBIN関数」(7-82ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

STR 数値データ→アスキー文字列変換

10進アスキーコード変換
戻り値※1 = **STR** (変換対象)

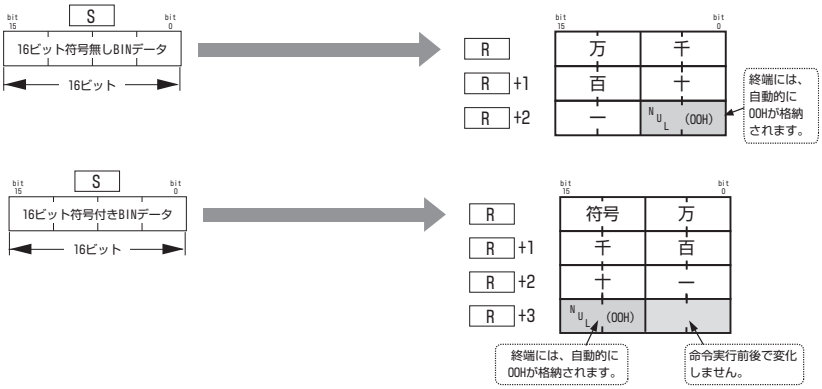
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 変換対象※2※4	変換対象のバイナリデータを格納した対象、または変換対象データを指定します。	.U	.S	.D	.L	.F	—	—	—	○	○	○
R 戻り値※3※5	STR関数は、.Tの型を持ちます。	—	—	—	—	—	—	—	.T	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータを指定した場合、連続する16/32ビットを扱います。
チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。
※3 ビットデータではチャンネルの先頭を指定してください。
※4 定数は$で指定できません。
※5 T, C, CR, CTC, CTH, Zは使用できません。

動作説明

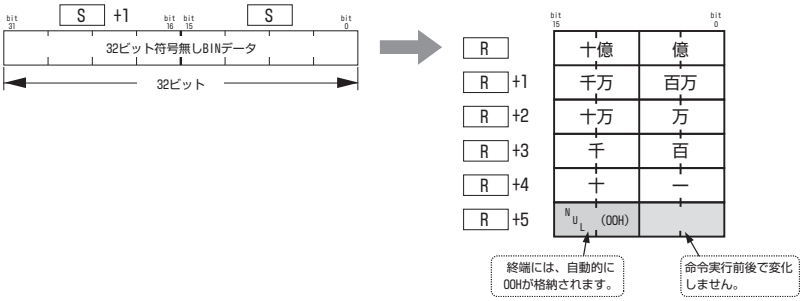
S で指定された引数の型により、10進アスキー文字列変換または単精度浮動小数点文字列変換をおこないます。

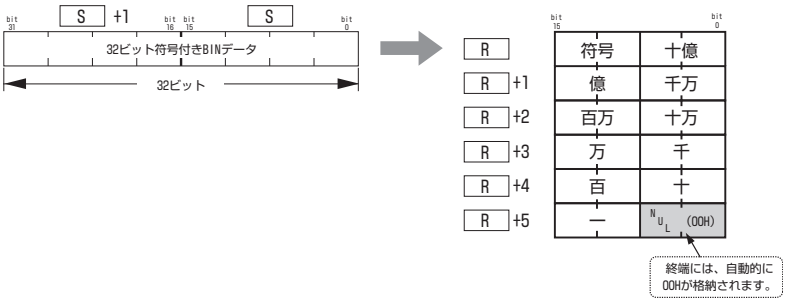
- **S 変換対象が16ビットの場合:DASC関数と同じ動作をします。**



動作の詳細については、「DASC関数」(7-158ページ)を参照してください。

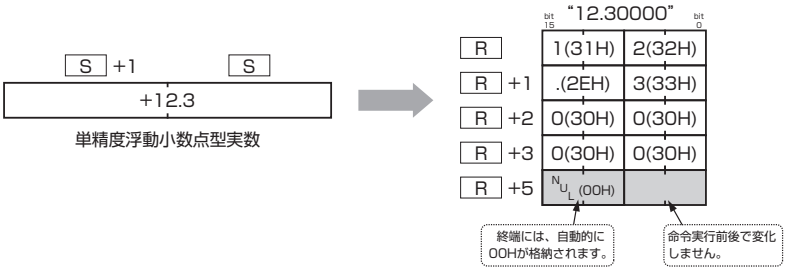
- **[S] 変換対象が32ビットの場合：DASC関数と同じ動作をします。**





動作の詳細については、「DASC関数」(7-158ページ)を参照してください。

- **[S] 変換対象が単精度浮動小数点型の場合：FASC関数※と同じ動作をします。**



※変換パラメータ＝3(CR2814＝ON、CR2815＝ON：ゼロサプレス、符号省略)として変換します。
動作の詳細については、「FASC関数」(7-174ページ)を参照してください。
「演算フラグについて(CR2009～CR2012)」(1-18ページ)

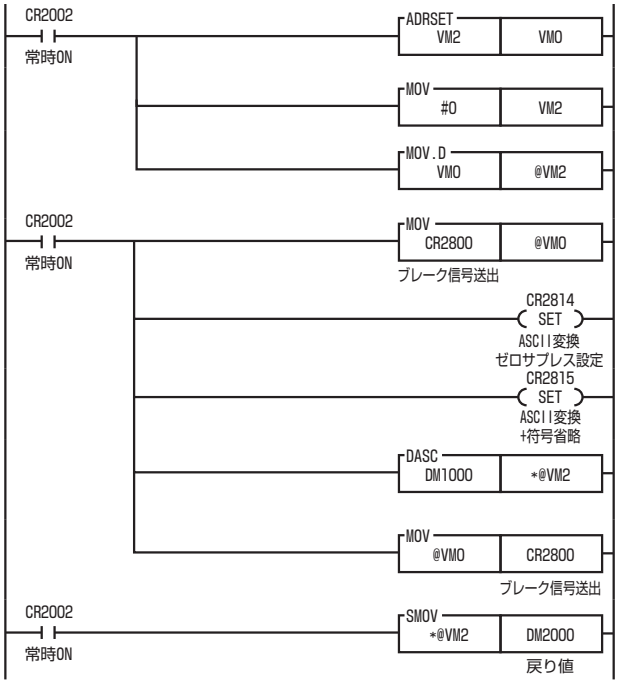
ポイント

- **変換対象データに定数を使用する場合、16進($)での指定はできません。**
- **次のいずれかの条件が成立した場合、[R]には終端コード(NUL(00H))が格納されます。**
  - **[R]にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。**
  - **[R]に間接指定でタイマ・カウンタを指定した場合。**

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

●書式例

DM2000.T = **STR** (DM1000)



MEMO

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

RDASC　10進アスキー文字列→BINデータ変換

10進アスキーコード逆変換
戻り値※1 = **RDASC** (変換対象)

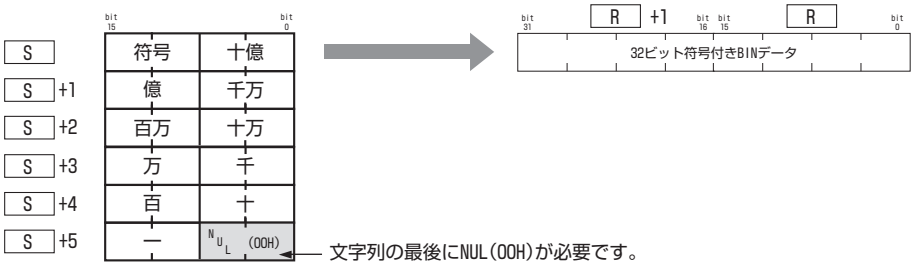
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 変換対象※2※3	変換対象の10進アスキー文字列を格納した対象、または10進アスキーコード文字列を指定します。	—	—	—	—	—	—	—	.T	○	○	○
R 戻り値※4	RDASC関数は、.Lの型を持ちます。※5	.U	.S	.D	.L	.F	※6 .DF	—	—	—	○	○

※1　戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2　ビットデータではチャンネルの先頭を指定してください。
※3　T, C, CR, CTC, CTH, Zは使用できません。
※4　ビットデータを指定した場合、連続する32ビットを扱います。
　　チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。
※5　戻り値に.L以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※6　デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

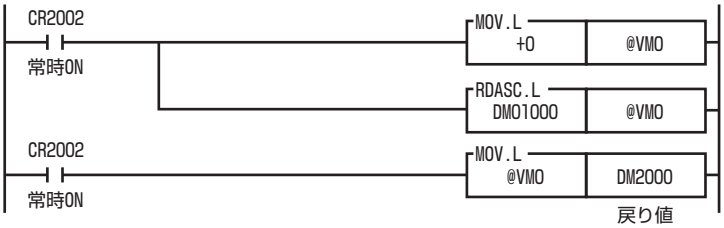
S を先頭に格納されている10進アスキー文字列を、32ビットバイナリデータ(符号付き)に変換し、結果を R に格納します。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.L = **RDASC** (DM1000.T)



ポイント

- 変換対象データで指定する10進アスキー文字列には、$30～$39(”0”～”9”)、および$2B、$2D(”±”符号：符号付変換のときのみ)で構成された文字列を指定してください。
- S に上記以外のアスキー文字列を指定した場合、関数は実行されません。
- S で指定したアスキー文字列にNUL(00H)がない場合、関数は最大11文字の文字列を変換します。
- 次のいずれかの条件が成立した場合、R には0が格納されます。
  - S に不適切な文字がある場合。
  - 変換した値が32ビット符号付きバイナリデータの範囲外の場合。
  - S にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。
  - S に間接指定でタイマ・カウンタを指定した場合。

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述　DM2000.L = **RDASC** (SLEFT (DM1000.T, 7))

動作説明　DM1000.Tを先頭に格納されている10進アスキー文字列の先頭7文字を切り出して、32ビットバイナリデータ(符号付き)に変換し、結果をDM2000.Lに格納します。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「SLEFT関数」(7-188ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

HASC　BINデータ→16進アスキー文字列変換

16進アスキーコード変換
戻り値※1 = **HASC** (変換対象, 変換桁数)

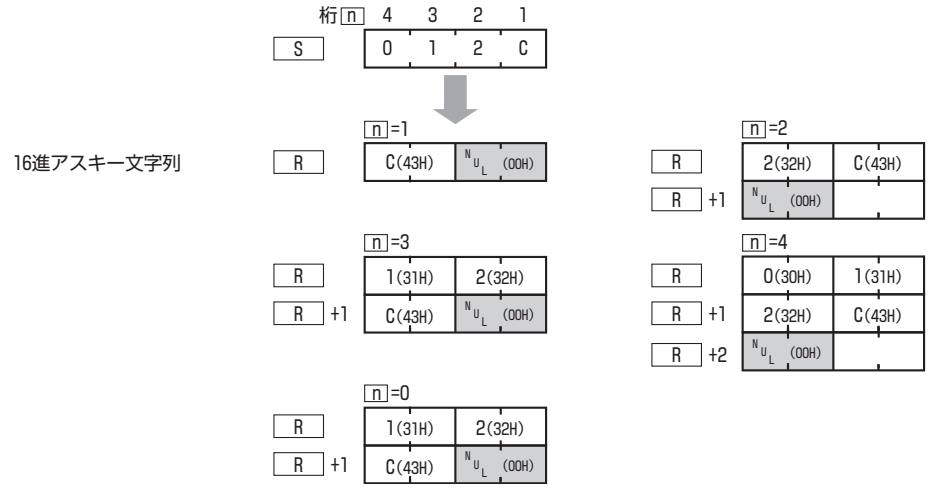
引数/戻り値	説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
S 変換対象※2	変換対象のバイナリデータを格納した対象、または変換対象データを指定します。(符号無しバイナリデータ)。	.U	.U	.D	.D	—	—	—	—	○	○	○
n 変換桁数※5	変換結果を格納する数値(16進)の桁数を指定します。(0～8)	.U	.U	.U	.U	—	—	—	—	※3 ○	○	○
R 戻り値※4※6	HASC関数は、.Tの型を持ちます。	—	—	—	—	—	—	—	.T	—	○	—

※1　戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2　ビットデータを指定した場合、連続する16/32ビットを扱います。
　　チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。
※3　定数は $ で指定してください。# で指定することはできません。
※4　戻り値を格納する対象には、.Tサフィックスを指定する必要があります。
　　また戻り値にビットデータを指定する場合は、チャンネルの先頭を指定する必要があります。
※5　変換桁数には変換結果を格納する桁数(下位桁)を、0～8の範囲で指定します。
　　(0を指定した場合は、ゼロサプレス有りで有効桁数分が格納されます)
※6　T, C, CR, CTC, CTH, Zは使用できません。

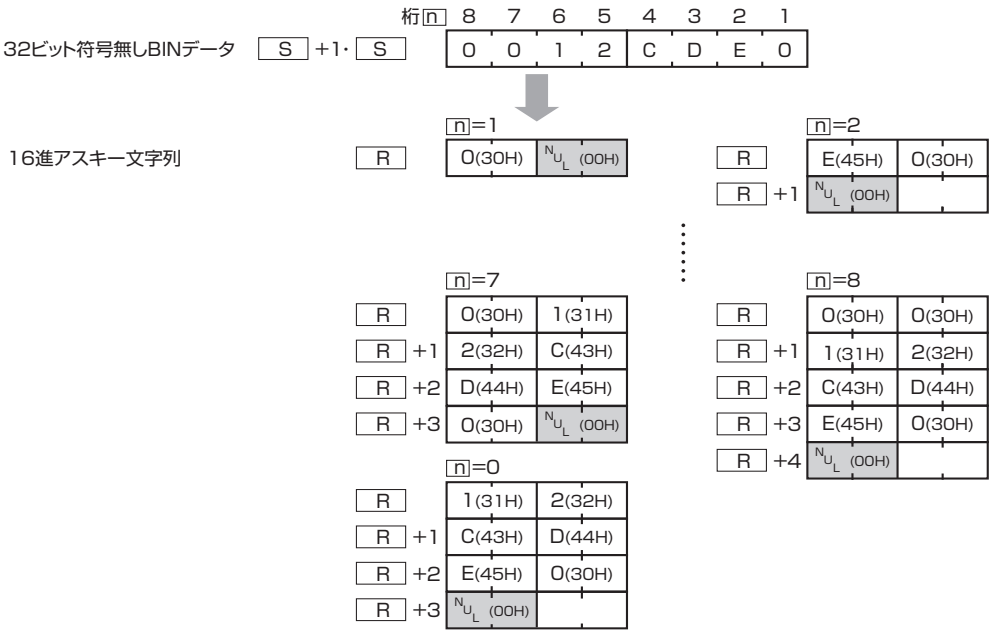
動作説明

S に格納されたバイナリデータを、終端コードNUL(00H)を付加した16進表記のアスキーコード文字列に変換し、結果を指定された n 分だけ R に格納します。(n ＝0のとき S のデータの全てをゼロサプレスして変換します)

- **S :変換対象が16ビット符号無しデータ(.U)の場合**



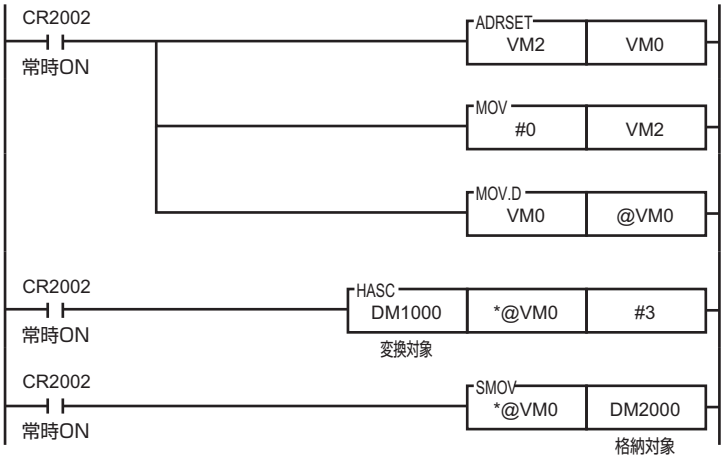
- S :変換対象が32ビット符号無しデータ(.D)の場合



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.T = HASC (DM1000, 3)



ポイント
- 変換対象データに定数を使用する場合、10進(#)での指定はできません。
- 次のいずれかの条件が成立した場合、R には終端コード(NUL(00H))が格納されます。
  - R にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。
  - R に間接指定でタイマ･カウンタを指定した場合。
  - n が.Uのとき5以上、.Dのとき9以上が指定された場合。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述　EM2000.T = **HASC** (SRA (DM1000.D, 8), 0)

動作説明　DM1000.Dに格納されているバイナリデータを8ビット分右にシフトします。
シフト結果の値(シフト前の上位24ビット)を、16進表記のアスキーコード文字列に変換し、ゼロサプレス有りで有効桁数分をEM2000.Tに格納します。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
「SRA関数」(7-38ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

MEMO

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

RHASC　16進アスキー文字列→BINデータ変換

16進アスキーコード逆変換
戻り値※1 = **RHASC** (変換対象)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 変換対象※2※3	変換対象の16進アスキー文字列を格納した対象、または6進アスキーコード文字列を指定します。	—	—	—	—	—	—	—	.T	○	○	○
R 戻り値※4	RHASC関数は、.Dの型を持ちます。※5	.U	.S	.D	.L	.F	※6 .DF	—	—	—	○	○

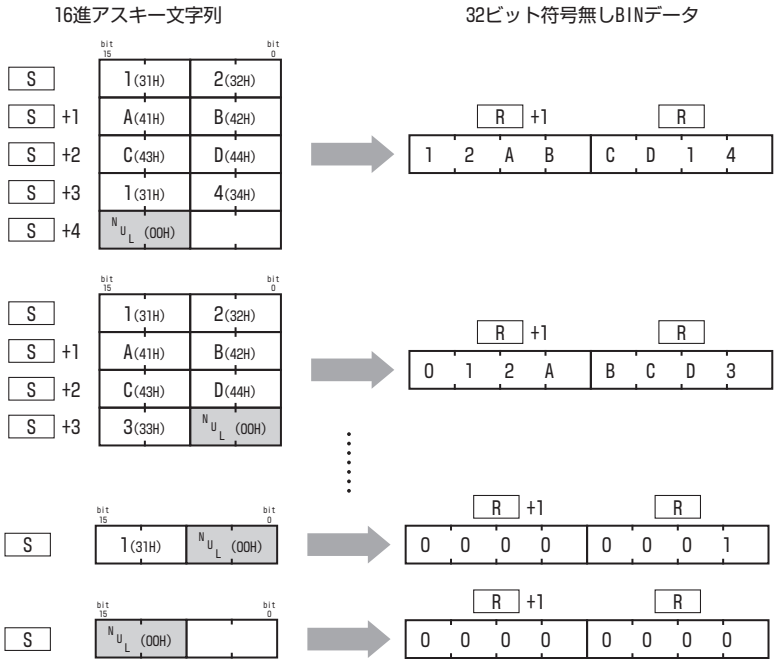
※1　戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2　ビットデータではチャンネルの先頭を指定してください。
※3　T, C, CR, CTC, CTH, Zは使用できません。
※4　ビットデータを指定した場合、連続する32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって32ビットを扱います。
※5　戻り値に.D以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※6　デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

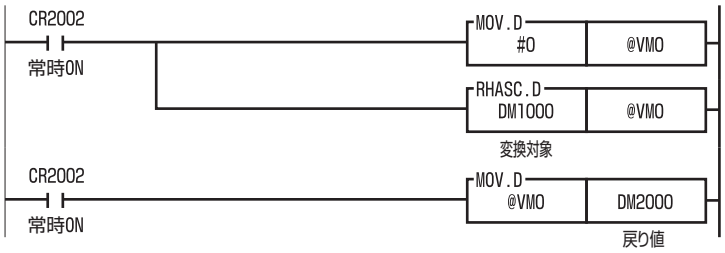
S を先頭に格納されている16進表記のアスキーコード文字列を、32ビットバイナリデータ(符号無し)に変換し、結果を R に格納します。



「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.D = **RHASC** (DM1000.T)



ポイント

- **S 変換対象データで指定する16進アスキー文字列には、$30～$39(“0”～“9”)、または$41～$46(“A”～“F”)の範囲の値が格納されている必要があります。**
- **16進アスキー文字列でA～F(41H～46H)は大文字を使用してください。小文字には対応していません。**
- **次のいずれかの条件が成立した場合、R には0が格納されます。**
  - **S に不適切な文字がある場合。**
  - **変換した値が32ビット符号無しバイナリデータの範囲外の場合。**
  - **S にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。**
  - **S に間接指定でタイマ･カウンタを指定した場合。**

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述　DM2000.D = **RHASC** (SLEFT (DM1000.T, 6))

動作説明　DM1000.Tを先頭に格納されている16進表記のアスキーコード文字列の先頭6文字を切り出して、32ビットバイナリデータ(符号無し)に変換し、結果をDM2000.Dに格納します。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「SLEFT関数」（7-188ページ）
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」（2-10ページ）を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

FASC 単精度浮動小数点型実数→文字列変換

単精度浮動小数点文字列変換
戻り値※1 = **FASC** (表示形式, 全桁数, 小数点桁数, 変換対象, 変換パラメータ)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	表示指定形式	表示指定形式(小数点形式、指数形式)を指定します。	.U	.U	.U	.U	—	—	—	.T	○	○	○
S2	全桁数	全体の桁数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
S3	小数点桁数	小数部桁数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
S4	変換対象※2※3※5	変換対象の単精度浮動小数点型実数を格納した対象、または単精度浮動小数点型実数データを指定します。	—	—	—	—	.F	—	—	—	○	○	○
S5	変換パラメータ	ゼロサプレスと符号省略を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値※4※5	FASC関数は、.Tの型を持ちます。	—	—	—	—	—	—	—	.T	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータを指定した場合、連続した32ビットを占有します。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって32ビットを占有します。
※3 単精度浮動小数点型実数の範囲は、「INTG関数」(7-122ページ)を参照してください。
※4 ビットデータでは、チャンネルの先頭を指定してください。
※5 T, C, CR, CTC, CTH, Zは使用できません。

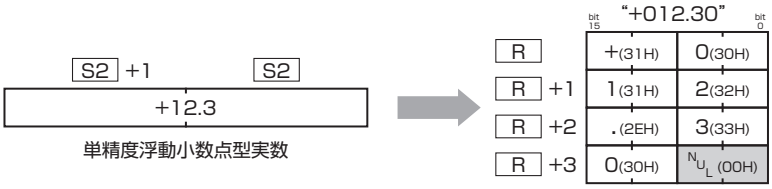
動作説明

S4 に格納された単精度浮動小数点型実数データを、S1 ～ S3 および S5 に格納された変換パラメータに従って、終端コードNUL(00H)を付加した文字列に変換し、結果を R に格納します。
変換パラメータは次のように指定します。

S1	表示指定形式	小数点形式の文字列に変換する場合は「0」、指数形式の文字列に変換する場合は「1」を指定します。
S2	全体の桁数	1～24桁の範囲で指定します。
S3	小数部桁数	0～7桁の範囲で指定します。

変換パラメータで指定する内容は次のようになります。

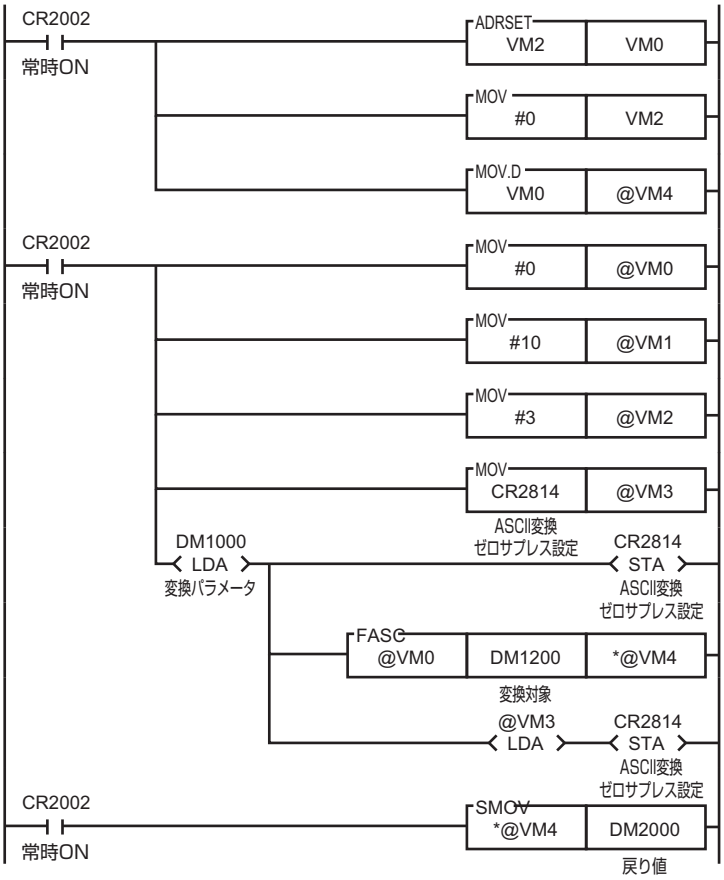
S5 値	内　容	CR2814	CR2815
0	ゼロサプレスOFF、符号省略OFF	0	0
1	ゼロサプレスON、符号省略OFF	1	0
2	ゼロサプレスOFF、符号省略ON	0	1
3	ゼロサプレスON、符号省略ON	1	1



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.T = **FASC** (0, 10, 3, DM1200.F, DM1000)



ポイント

- 変換対象データに定数を使用する場合、16進($)での指定はできません。
- 割り込みプログラムでCR2814・CR2815を変更したり参照する場合は注意が必要です。
- 次のいずれかの条件が成立した場合、[R]には終端コード(NUL(00H))が格納されます。
  - 引数の値が範囲外の場合。
  - 単精度浮動小数点型実数の範囲外を指定した場合。
  - 格納する文字列の桁数が全桁数を超えた場合。
  - [R]にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。
  - [R]に間接指定でタイマ・カウンタを指定した場合。
  - 全桁数を、小数部桁数が次の条件を満たさない場合。

	小数点形式表示	指数形式表示
[S3]=0	1≦[S2]≦24	5≦[S2]≦24
0<[S3]≦7	[S3]+2≦[S2] 3≦[S2]≦24	[S3]+6≦[S2] 7≦[S2]≦24

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述　EM2000.T = **FASC** (0, 10, 3, SIN (DM1000.F), 3)

動作説明　DM1000.Fに格納された単精度浮動小数点型実数の角度の正弦(SIN)値を求め、結果を小数点形式、全桁数10桁のうち3桁を小数点以下とした文字列に変換します。(ゼロサプレス、符号省略で変換)
変換された文字列に終端コードNUL(00H)を付加し、EM2000.Tに格納します。(FLOAT関数等を使って変換対象を単精度浮動小数点型にする必要はありません)

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「SIN関数」(7-140ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

MEMO

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

RFASC 文字列→単精度浮動小数点型実数変換

単精度浮動小数点文字列逆変換
戻り値※1 = **RFASC** (変換対象)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象※2※4※5	変換対象の単精度浮動小数点文字列を格納した対象、または単精度浮動小数点文字列を指定します。	—	—	—	—	—	—	—	.T	○	○	○
R 戻り値※3※5	RFASC関数は、.Fの型を持ちます。※6	.U	.S	.D	.L	.F	※7 .DF	—	—	—	○	—

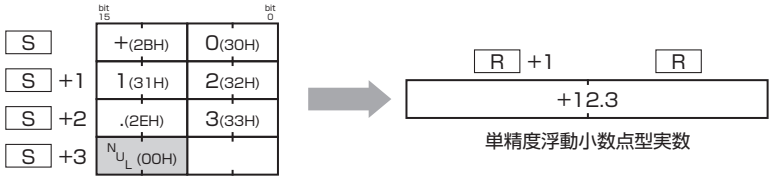
※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 ビットデータを指定した場合、連続した32ビットを占有します。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって32ビットを占有します。
※4 変換対象で指定する単精度浮動小数点文字列には、最後にNUL(00H)が必要です。
※5 T, C, CR, CTC, CTH, Zは使用できません。
※6 戻り値に.F以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※7 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

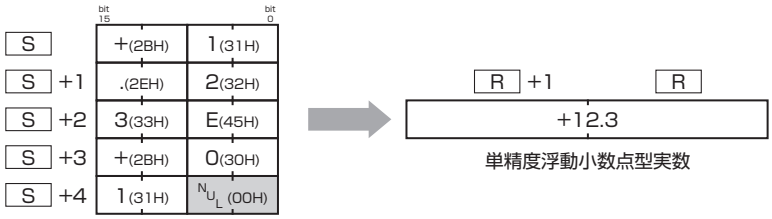
動作説明

S を先頭に格納されている単精度浮動小数点文字列を、単精度浮動小数点型実数データに変換し、結果を R に格納します。

"+012.30" の単精度浮動小数点文字列の場合



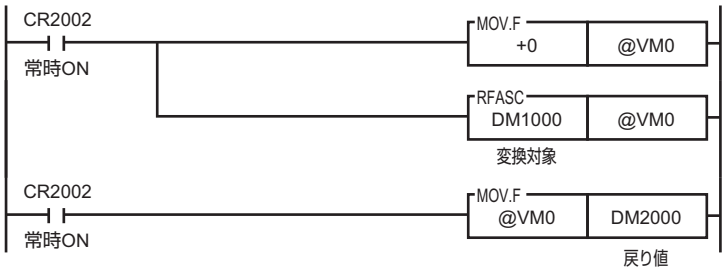
"+1.23E+01" の単精度浮動小数点文字列の場合



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

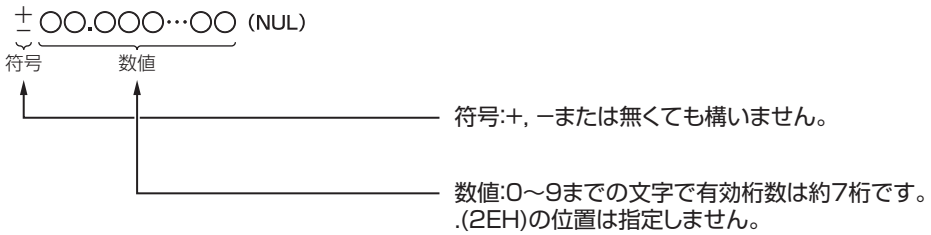
DM2000.F = **RFASC** (DM1000.T)

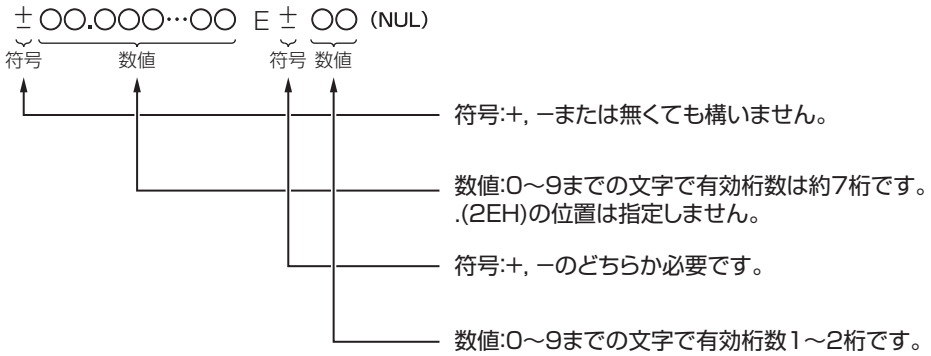


ポイント

- **変換対象データで指定する単精度浮動小数点文字列には、$30 ～ $39("0" ～ "9")、$2B("+")、$2C("－")、$45("E")、および$2E(".")の値のみが格納されている必要があります。**
- **次のいずれかの条件が成立した場合、[R]には0が格納されます。**
  - **文字列の長さが0の場合。**
  - **文字列の書式が正しくない場合。**
  - **変換後の数値が単精度浮動小数点型実数の範囲を超えている場合。**
  - **格納する文字列の桁数が全桁数を超えた場合。**
  - **[S]にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。**

●文字列の書式





サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　DM2000.F = COS (**RFASC** (DM1000.T))

動作説明　DM1000.Tを先頭に格納されている角度(ラジアン単位)の単精度浮動小数点文字列を、単精度浮動小数点型実数データに変換し、変換結果から余弦(COS)値を求め、DM2000.Fに格納します。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「COS関数」(7-142ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

VAL 文字列→単精度浮動小数点型実数変換

単精度浮動小数点文字列逆変換
戻り値※1 = **VAL** (変換対象)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象※2※3※5	変換対象の単精度浮動小数点文字列を格納した対象、または単精度浮動小数点文字列を指定します。	—	—	—	—	—	—	—	.T	○	○	○
R 戻り値※4	VAL関数は、.Fの型を持ちます。※6	.U	.S	.D	.L	.F	※7 .DF	—	—	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 変換対象で指定する単精度浮動小数点文字列には、最後にNUL(00H)が必要です。
※4 ビットデータを指定した場合、連続した32ビットを占有します。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって32ビットを占有します。
※5 T, C, CR, CTC, CTH, Zは使用できません。
※6 戻り値に.F以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※7 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

本関数は「RFASC関数」と関数名が異なるだけで、動作等に違いはありません。
動作の詳細については、「RFASC関数」(7-178ページ)を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

LEN 文字列長検出

戻り値※1 = LEN (対象文字列)
文字列長検出

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 対象文字列※3※4	文字列長を求める対象の文字列を格納した対象、または対象文字列を指定します。	—	—	—	—	—	—	—	.T	※2 ○	○	○
R 戻り値	LEN関数は、.Uの型を持ちます。※5	.U	.S	.D	.L	.F	※6 .DF	—	—	—	○	—

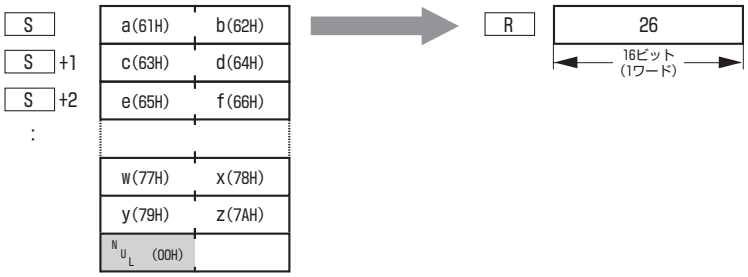
※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 定数で文字列を指定する場合は、" "で囲ってください。
※3 ビットデータではチャンネルの先頭を指定してください。
※4 T, C, CR, CTC, CTH, Zは使用できません。
※5 戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※6 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

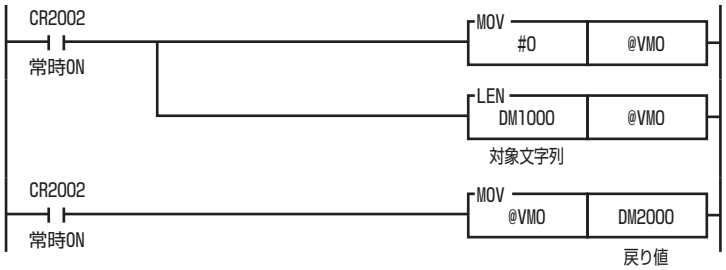
S を先頭に格納されている文字列の文字数を数え、結果を16ビットバイナリデータとして R に格納します。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000 = LEN (DM1000.T)



ポイント

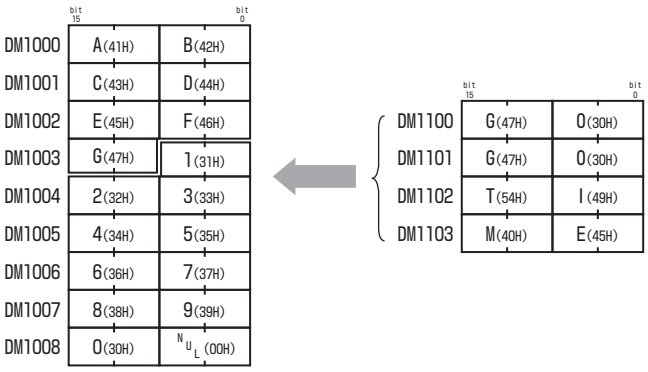
- **全角文字は2文字として扱われます。**
- **終端コードNUL(00H)は数えません。**
- **次のいずれかの条件が成立した場合、R には0が格納されます。**
  - **終端コード(NUL(00H))が無い場合。**
  - **終端コード(NUL(00H))を含む文字列が2000文字を超えた場合。**
  - **S に間接指定でビットデータを指定したときに、チャンネルの先頭以外を指定した場合。**
  - **S に間接指定でタイマ・カウンタを指定した場合。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　DM2000.T = SINS (DM1000.T, DM1100.T, **LEN** (DM1000.T) - 10)

動作説明　DM1000を先頭に格納されている文字列の文字列長(文字数)を求め最後から10文字前(LEN (DM1000.T) －10)に、DM1100を先頭に格納されている文字列を挿入します。
挿入した結果の文字列の最後には終端コードNUL(00H)を付加しDM2000に格納します。



参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「SINS関数」(7-200ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

SRGHT 右文字列切り出し

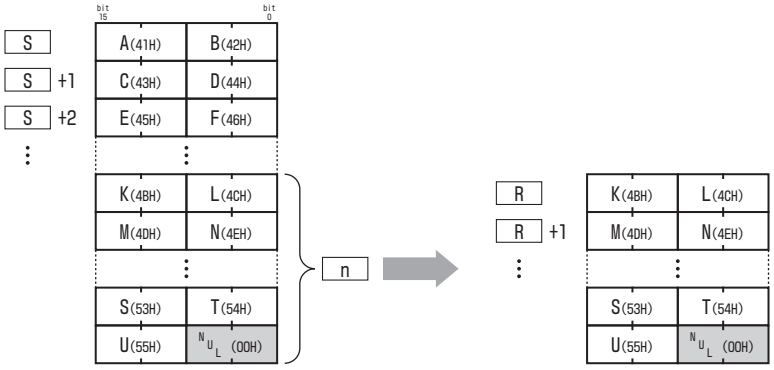
戻り値※1 = **SRGHT** (切り出し対象, 切り出し文字数)
右文字列切り出し

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 切り出し対象※2※6	切り出し対象となる文字列を格納した対象、または切り出し対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
n 切り出し文字数※5	切り出す文字数(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
R 戻り値※2 ※6	SRGHT関数は、.Tの型を持ちます。	—	—	—	—	—	—	—	.T	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 $ は使用できません。
※5 切り出し文字数には右端(終端)からの文字数(バイト数)を、0～1999の範囲で指定します。
0を指定した場合、戻り値にはNUL(00H)が1文字分だけ格納されます。
※6 T, C, CR, CTC, CTH, Zは使用できません。

動作説明

S を先頭に格納されている文字列を、右端から、n で指定された文字数(バイト数)分切り出し、終端コードNUL(00H)を付加して、結果を R に格納します。



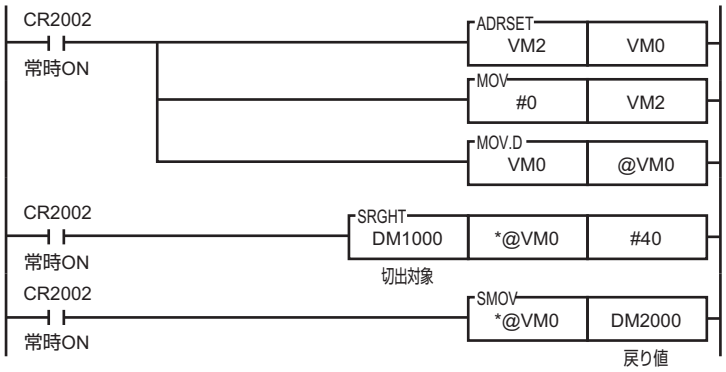
文字列の最後のNUL(00H)は含みません。

切り出し後の文字列の最後にNUL(00H)が格納されます。

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.T = SRGHT (DM1000.T, 40)



ポイント

- **次のいずれかの条件が成立した場合、終端コード(NUL(00H))が格納されます。**
  - **[S]で指定した文字列に終端コード(NUL(00H))が無い場合。**
  - **終端コード(NUL(00H))を含む文字列が2000文字を超えた場合。**
  - **[R]以降に格納される文字列がデバイス/変数の範囲を超えた場合。**
  - **[S]、[R]にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。**
  - **[S]、[R]に間接指定でタイマ・カウンタを指定した場合。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　EM2000.L = RDASC (**SRGHT** (DM1000.T, 8))

動作説明　DM1000を先頭に格納されている10進アスキーコード文字列の終端8文字(下位8桁)を切り出して、32ビットバイナリデータ(符号付き)に変換し、結果をEM2000.Lに格納します。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「RDASC関数」(7-166ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

RIGHT 右文字列切り出し

戻り値※1 = **RIGHT** (切り出し対象, 切り出し文字数)
右文字列切り出し

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 切り出し対象※2※6	切り出し対象となる文字列を格納した対象、または切り出し対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
n 切り出し文字数※5	切り出す文字数(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
R 戻り値※2 ※6	RIGHT関数は、.Tの型を持ちます。	—	—	—	—	—	—	—	.T	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 $ は使用できません。
※5 切り出し文字数には右端(終端)からの文字数(バイト数)を、0～1999の範囲で指定します。
0を指定した場合、戻り値にはNUL(00H)が1文字分だけ格納されます。
※6 T, C, CR, CTC, CTH, Zは使用できません。

動作説明

本関数は「SRGHT関数」と関数名が異なるだけで、動作等に違いはありません。
動作の詳細については、「SRGHT関数」(7-184ページ)を参照してください。

MEMO

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

SLEFT 左文字列切り出し

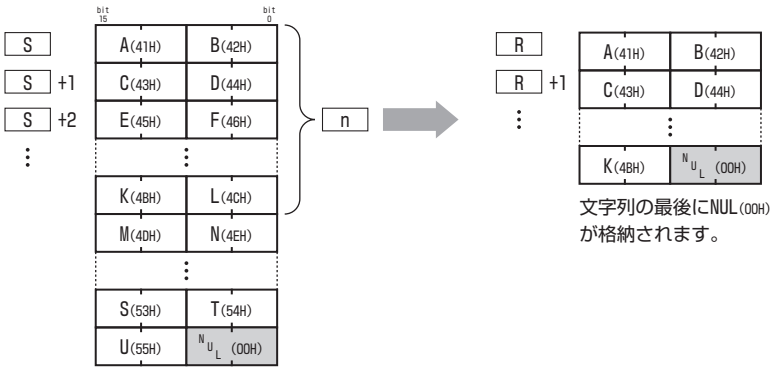
戻り値※1 = **SLEFT** (切り出し対象, 切り出し文字数)
左文字列切り出し

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 切り出し対象※2※6	切り出し対象となる文字列を格納した対象、または切り出し対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
n 切り出し文字数※5	切り出す文字数(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
R 戻り値※2 ※6	SLEFT関数は、.Tの型を持ちます。	—	—	—	—	—	—	—	.T	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 $は使用できません。
※5 切り出し文字数には左端(先端)からの文字数(バイト数)を、0～1999の範囲で指定します。
0を指定した場合、戻り値にはNUL(00H)が1文字分だけ格納されます。
※6 T, C, CR, CTC, CTH, Zは使用できません。

動作説明

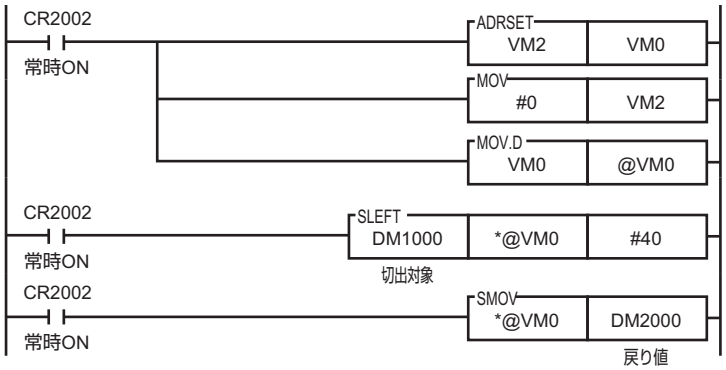
S を先頭に格納されている文字列を、左端(先端)から、n で指定された文字数(バイト数)分切り出し、終端コードNUL(00H)を付加して、結果を R に格納します。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.T = **SLEFT** (DM1000.T, 40)



ポイント

- **次のいずれかの条件が成立した場合、終端コード(NUL(00H))が格納されます。**
  - **[S]で指定した文字列に終端コード(NUL(00H))が無い場合。**
  - **終端コード(NUL(00H))を含む文字列が2000文字を超えた場合。**
  - **[R]以降に格納される文字列がデバイス/変数の範囲を超えた場合。**
  - **[S]、[R]にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。**
  - **[S]、[R]に間接指定でタイマ・カウンタを指定した場合。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　EM2000.D = RHASC (**SLEFT** (DM1000.T, 4))

動作説明　DM1000.Tを先頭に格納されている16進表記のアスキーコード文字列の先頭4文字(上位4桁)を切り出して、16ビットバイナリデータ(符号無し)に変換し、結果をEM2000に格納します。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「RHASC関数」(7-172ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

LEFT 左文字列切り出し

戻り値※1 = **LEFT** (切り出し対象, 切り出し文字数)
左文字列切り出し

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 切り出し対象※2※6	切り出し対象となる文字列を格納した対象、または切り出し対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
n 切り出し文字数※5	切り出す文字数(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
R 戻り値※2 ※6	LEFT関数は、.Tの型を持ちます。	—	—	—	—	—	—	—	.T	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 $は使用できません。
※5 切り出し文字数には左端(先端)からの文字数(バイト数)を、0～1999の範囲で指定します。
0を指定した場合、戻り値にはNUL(00H)が1文字分だけ格納されます。
※6 T, C, CR, CTC, CTH, Zは使用できません。

動作説明

本関数は「SLEFT関数」と関数名が異なるだけで、動作等に違いはありません。
動作の詳細については、「SLEFT関数」(7-188ページ)を参照してください。

MEMO

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

SMID 文字列切り出し

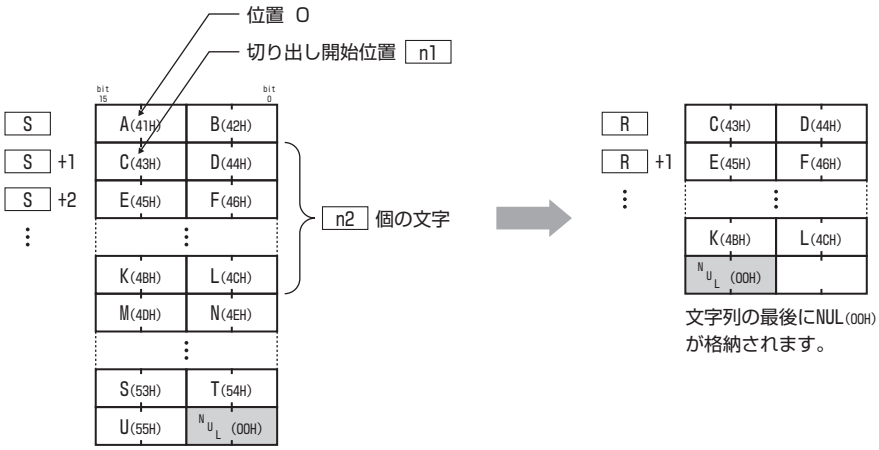
戻り値※1 = **SMID** (切り出し対象, 切り出し文字位置, 切り出し文字数)
文字列切り出し

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 切り出し対象※2※6	切り出し対象となる文字列を格納した対象、または切り出し対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
n1 切り出し文字位置※7	切り出し開始位置(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
n2 切り出し文字数※5	切り出す文字数(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
R 戻り値※2※6	SMID関数は、.Tの型を持ちます。	—	—	—	—	—	—	—	.T	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 $は使用できません。
※5 切り出し文字数には切り出し文字からの文字数(バイト数)を、0～1999の範囲で指定します。
0を指定した場合、戻り値にはNUL(00H)が1文字分だけ格納されます。
※6 T, C, CR, CTC, CTH, Zは使用できません。
※7 S で指定した文字列の先頭文字を0とします。

動作説明

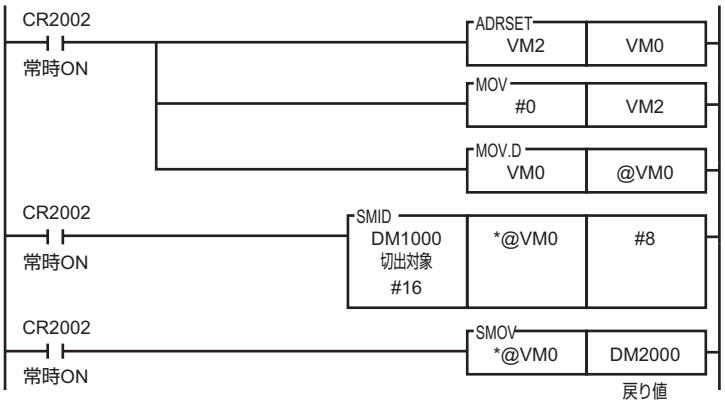
S を先頭に格納されている文字列を、n1 で指定された位置(バイト数)から、n2 で指定された文字数(バイト数)分を切り出し、終端コードNUL(00H)を付加して、結果を R に格納します。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.T = **SMID** (DM1000.T, 8, 16)



ポイント

- **次のいずれかの条件が成立した場合、終端コード(NUL(00H))が格納されます。**
  - **[S]で指定した文字列に終端コード(NUL(00H))が無い場合。**
  - **終端コード(NUL(00H))を含む文字列が2000文字を超えた場合。**
  - **[S]の文字列長＜[n1]の場合。**
  - **[R]以降に格納される文字列がデバイス/変数の範囲を超えた場合。**
  - **[S]、[R]にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。**
  - **[S]、[R]に間接指定でタイマ・カウンタを指定した場合。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　EM2000.D = RHASC (**SMID** (DM1000.T, 2, 4))

動作説明　DM1000.Tを先頭に格納されている16進表記のアスキーコード文字列の、2文字目から4文字を(中4桁)切り出して、16ビットバイナリデータ(符号無し)に変換し、結果をEM2000に格納します。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「RHASC関数」(7-172ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

MID 文字列切り出し

戻り値※1 = **MID** (切り出し対象, 切り出し文字位置, 切り出し文字数)
文字列切り出し

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 切り出し対象※2※6	切り出し対象となる文字列を格納した対象、または切り出し対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
n1 切り出し文字位置※7	切り出し開始位置(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
n2 切り出し文字数※5	切り出す文字数(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
R 戻り値※2※6	MID関数は、.Tの型を持ちます。	—	—	—	—	—	—	—	.T	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 $は使用できません。
※5 切り出し文字数には切り出し文字からの文字数(バイト数)を、0～1999の範囲で指定します。
0を指定した場合、戻り値にはNUL(00H)が1文字分だけ格納されます。
※6 T, C, CR, CTC, CTH, Zは使用できません。
※7 S で指定した文字列の先頭文字を0とします。

動作説明

本関数は「SMID関数」と関数名が異なるだけで、動作等に違いはありません。
動作の詳細については、「SMID関数」(7-192ページ)を参照してください。

MEMO

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

SRPLC 文字列部分置換

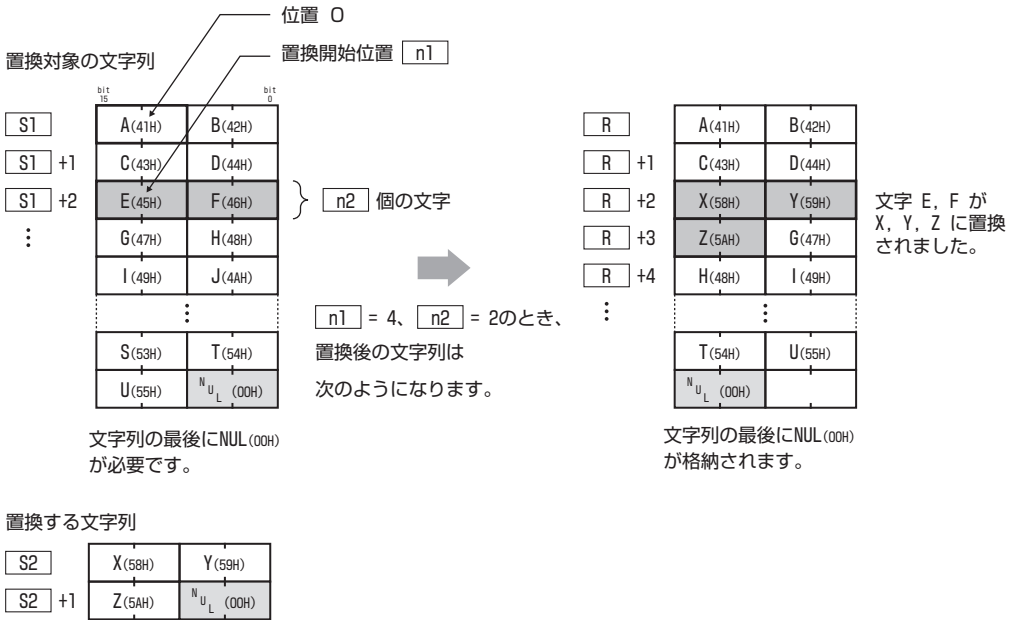
戻り値※1 = **SRPLC** (置換対象, 置換文字列対象, 置換開始位置, 置換文字数)
文字列部分置換

引数/戻り値	説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
S1 置換対象※2※6	置換対象となる文字列を格納した対象、または置換対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
S2 置換文字列対象※2※6	置換する文字列を格納した対象、または置換文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
n1 置換開始位置※7	置換開始位置(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
n2 置換文字数※5	置換する文字数(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
R 戻り値※2※6	関数が処理した結果の値を格納します。	—	—	—	—	—	—	—	.T	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 $は使用できません。
※5 置換文字数には置換開始位置からの文字数(バイト数)を、0～1999の範囲で指定します。
※6 T, C, CR, CTC, CTH, Zは使用できません。
※7 S1 で指定した文字列の先頭文字を0とします。

動作説明

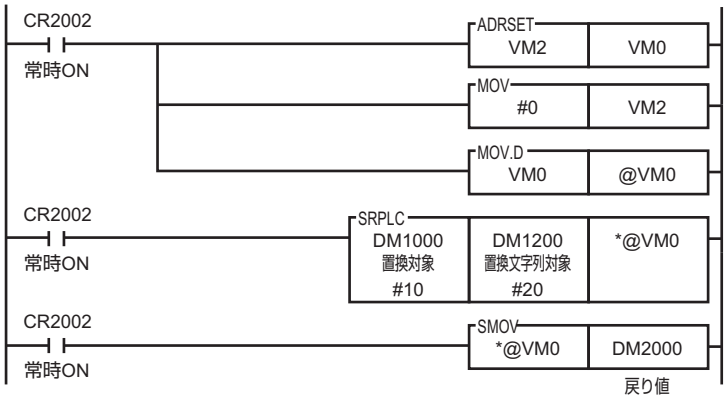
S1 を先頭に格納されている文字列の、n1 で指定された位置(バイト数)から、n2 で指定された文字数(バイト数)分の文字列を、S2 を先頭に格納されている文字列で置き換え、終端コードNUL(00H)を付加して、結果を R に格納します。



「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.T = **SRPLC** (DM1000.T, DM1200.T, 10, 20)



ポイント

- **次のいずれかの条件が成立した場合、終端コード(NUL(00H))が格納されます。**
  - **[S1]、[S2]で指定した文字列に終端コード(NUL(00H))が無い場合。**
  - **終端コード(NUL(00H))を含む文字列が2000文字を超えた場合。**
  - **[S1]の文字列長＜[n1]の場合。**
  - **置換後の文字列長が終端コード(NUL(00H))を含めて2000文字を超える場合。**
  - **[S1]、[S2]、[R]にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。**
  - **[S1]、[S2]、[R]に間接指定でタイマ･カウンタを指定した場合。**

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　EM2000.L = RDASC (**SRPLC** (DM1000.T, EM1000.T, 0, 4))

動作説明　DM1000.Tを先頭に格納されている10進アスキー文字列の、先頭から4文字(バイト数)分の文字列を、EM1000.Tを先頭に格納されている10進アスキー文字列で置き換えます。
置換した結果の10進アスキー文字列を、32ビットバイナリデータ(符号付き)に変換し、結果をEM2000.Lに格納します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
「RDASC関数」(7-166ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ワンポイント

SRPLC関数は、引数指定内容によって下記示すような動作機能となります。

例1)　置換文字数に0を指定したとき、SINS関数と同じ動作になります。
置換対象文字列の4文字目に置換文字列が挿入(置換対象範囲なし)されます。

スクリプト記述　DM2000.T = **SRPLC** (DM1000.T, DM1200.T, 4, 0)

例2)　置換文字列に空の文字列(NULのみ)を指定したとき、SDEL関数と同じ動作になります。
置換対象文字列の2文字目から4文字が削除(置換文字列なし)されます。

スクリプト記述　DM2000.T = **SRPLC** (DM1000.T, "", 2, 4)

REPLACE 文字列部分置換

戻り値※1 = **REPLACE** (置換対象, 置換文字列対象, 置換文字数, 置換開始位置)
文字列部分置換

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	置換対象※2※6	置換対象となる文字列を格納した対象、または置換対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
S2	置換文字列対象※2※6	置換する文字列を格納した対象、または置換文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
n1	置換文字数※5	置換する文字数(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
n2	置換開始位置※7	置換開始位置(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
R	戻り値※2※6	関数が処理した結果の値を格納します。	—	—	—	—	—	—	—	.T	—	○	—

※1　戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2　ビットデータではチャンネルの先頭を指定してください。
※3　定数で文字列を指定する場合は、" "で囲ってください。
※4　$は使用できません。
※5　置換文字数には置換開始位置からの文字数(バイト数)を、0～1999の範囲で指定します。
※6　T, C, CR, CTC, CTH, Zは使用できません。
※7　S1で指定した文字列の先頭文字を0とします。

動作説明

本関数は「SRPLC関数」と関数名が異なるだけで、動作等に違いはありません。
動作の詳細については、「SRPLC関数」(7-196ページ)を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

SINS 文字列挿入

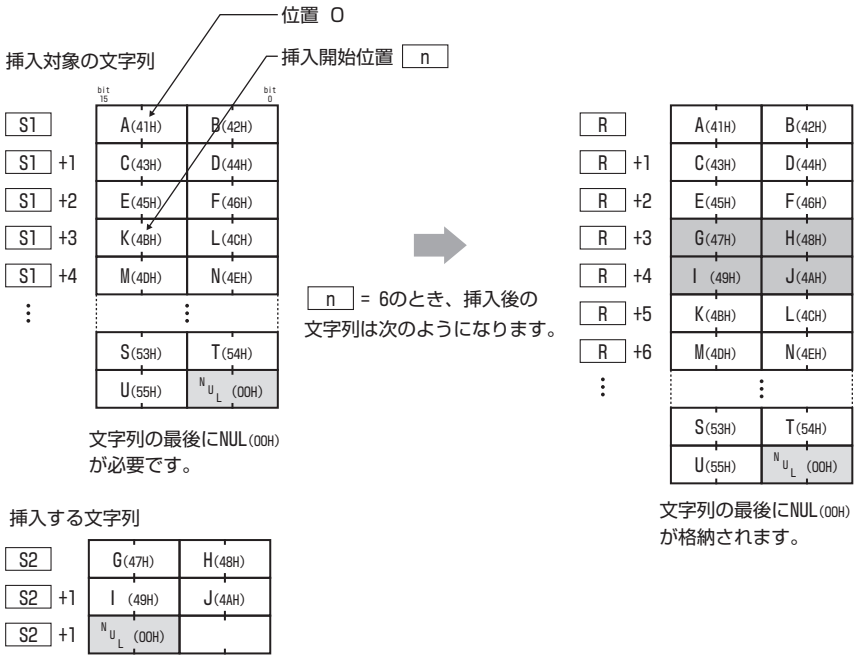
戻り値※1 = **SINS** (挿入対象, 挿入文字列対象, 挿入開始位置)
文字列挿入

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	挿入対象※2※6	挿入対象となる文字列を格納した対象、または挿入対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
S2	挿入文字列対象※2※6	挿入する文字列を格納した対象、または挿入文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
n	挿入開始位置※7	挿入開始位置(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
R	戻り値※4※6	SINS関数は、.Tの型を持ちます。	—	—	—	—	—	—	—	.T	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 $は使用できません。
※5 挿入開始位置には挿入する位置として、左端(先端)からの文字数(バイト数)を、0～1999の範囲で指定します。
※6 T, C, CR, CTC, CTH, Zは使用できません。
※7 S1 で指定した文字列の先頭文字を0とします。

動作説明

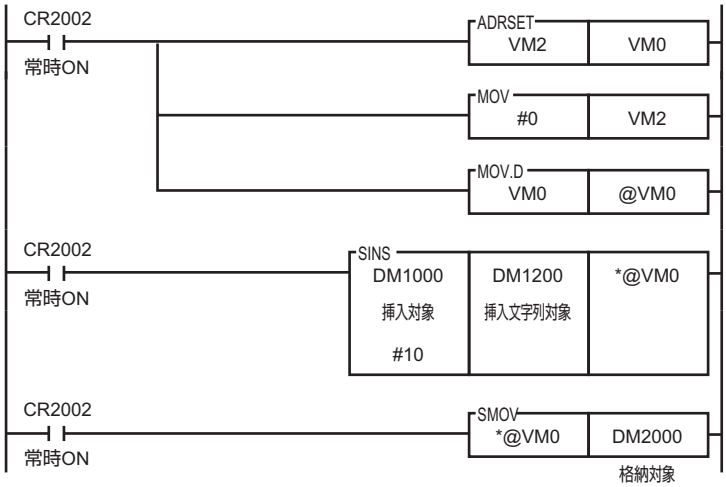
S1 を先頭に格納されている文字列の、n で指定された位置(バイト数)へ、S2 を先頭に格納されている文字列を挿入し、終端コードNUL(00H)を付加して、結果を R に格納します。



「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.T = **SINS** (DM1000.T, DM1200.T, 10)



ポイント

- **次のいずれかの条件が成立した場合、終端コード(NUL(00H))が格納されます。**
  - **S1、S2で指定した文字列に終端コード(NUL(00H))が無い場合。**
  - **終端コード(NUL(00H))を含む文字列が2000文字を超えた場合。**
  - **S1の文字列長＜nの場合。**
  - **挿入後の文字列長が終端コード(NUL(00H))を含めて2000文字を超える場合。**
  - **S1、S2、Rにビットデータを指定したときに、チャンネルの先頭以外を指定した場合。**
  - **S1、S2、Rに間接指定でタイマ･カウンタを指定した場合。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　EM2000.D = RHASC (**SINS** (DM1000.T, EM1000.T, 2))

動作説明　DM1000.Tを先頭に格納されている16進表記のアスキー文字列の、先頭から2文字(バイト数)目に、EM1000.Tを先頭に格納されている16進表記のアスキー文字列を挿入します。挿入した結果の16進表記のアスキー文字列を、32ビットバイナリデータ(符号無し)に変換し、結果をEM2000.Dに格納します。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「RDASC関数」（7-166ページ）
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」（2-10ページ）を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

INSERT 文字列挿入

戻り値※1 = INSERT (挿入対象, 挿入文字列対象, 挿入開始位置)
文字列挿入

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	挿入対象※2※6	挿入対象となる文字列を格納した対象、または挿入対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
S2	挿入文字列対象※2※6	挿入する文字列を格納した対象、または挿入文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
n	挿入開始位置※7	挿入開始位置(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
R	戻り値※2※6	INSERT関数は、.Tの型を持ちます。	—	—	—	—	—	—	—	.T	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 $は使用できません。
※5 挿入開始位置には挿入する位置として、左端(先端)からの文字数(バイト数)を、0～1999の範囲で指定します。
※6 T, C, CR, CTC, CTH, Zは使用できません。
※7 S1で指定した文字列の先頭文字を0とします。

動作説明

本関数は「SINS関数」と関数名が異なるだけで、動作等に違いはありません。
動作の詳細については、「SINS関数」(7-200ページ)を参照してください。

MEMO

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

SDEL 文字列の部分削除

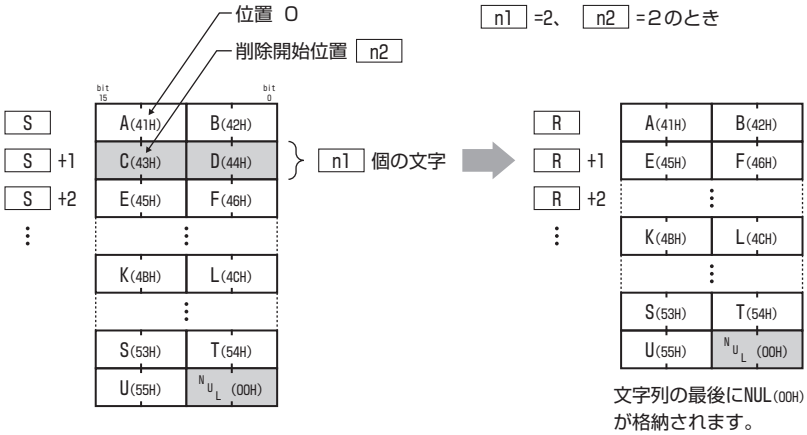
戻り値※1 = **SDEL** (削除対象, 削除開始位置, 削除文字数)
文字列部分削除

引数/戻り値	説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
S 削除対象※2※7	削除対象となる文字列を格納した対象、または削除対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
n1 削除開始位置※4※5	削除開始位置(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	○	○	○
n2 削除文字数※4※6	削除する文字数(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値※2※7	SDEL関数は、.Tの型を持ちます。	—	—	—	—	—	—	—	.T	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 $は使用できません。
※5 削除開始位置には削除する位置として、左端(先端)からの文字数(バイト数)を、0～1999の範囲で指定します。
※6 削除文字数には削除開始位置からの文字数(バイト数)を、0～1999の範囲で指定します。
※7 T, C, CR, CTC, CTH, Zは使用できません。

動作説明

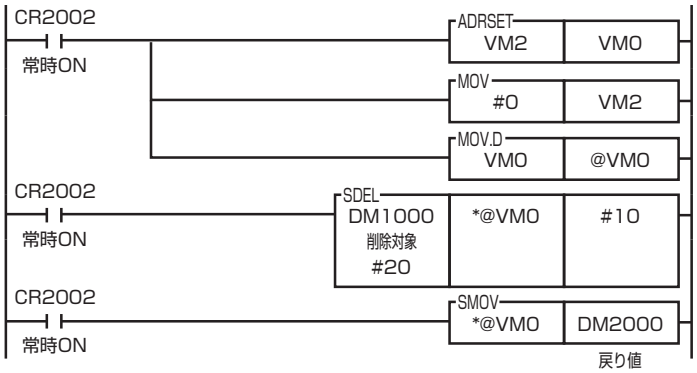
S を先頭に格納されている文字列の、n1 で指定された位置(バイト数)から n2 で指定された文字数(バイト数)分の文字列を削除し、終端コードNUL(00H)を付加して、結果を R に格納します。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.T = **SDEL** (DM1000.T, 10, 20)



ポイント

- **次のいずれかの条件が成立した場合、終端コード(NUL(00H))が格納されます。**
  - **[S]で指定した文字列に終端コード(NUL(00H))が無い場合。**
  - **終端コード(NUL(00H))を含む文字列が2000文字を超えた場合。**
  - **[S]の文字列長<[n1]の場合。**
  - **[R]以降に格納される文字列がデバイス/変数の範囲を超えた場合。**
  - **[S]、[R]にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。**
  - **[S]、[R]に間接指定でタイマ・カウンタを指定した場合。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　EM2000.D = RHASC (**SDEL** (DM1000.T, 2, 4))

動作説明　DM1000.Tを先頭に格納されている16進表記のアスキー文字列の、2文字(バイト数)目から4文字(バイト数)分の文字列を削除します。
削除した結果の16進表記のアスキー文字列を、16ビットバイナリデータ(符号無し)に変換し、結果をEM2000.Dに格納します。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「RHASC関数」(7-172ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

DELETE 文字列の部分削除

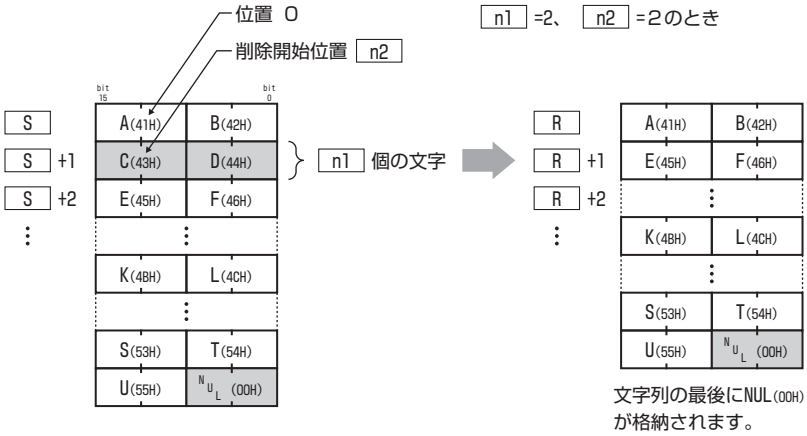
文字列部分削除
戻り値※1 = **DELETE** (削除対象, 削除文字数, 削除開始位置)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 削除対象※2※7	削除対象となる文字列を格納した対象、または削除対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
n1 削除文字数※4※6	削除する文字数(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	○	○	○
n2 削除開始位置※4※5	削除開始位置(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値※2※7	DELETE関数は、.Tの型を持ちます。	—	—	—	—	—	—	—	.T	○	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 $は使用できません。
※5 削除開始位置には削除する位置として、左端(先端)からの文字数(バイト数)を、0～1999の範囲で指定します。
※6 削除文字数には削除開始位置からの文字数(バイト数)を、0～1999の範囲で指定します。
※7 T, C, CTC, CTH, Zは使用できません。

動作説明

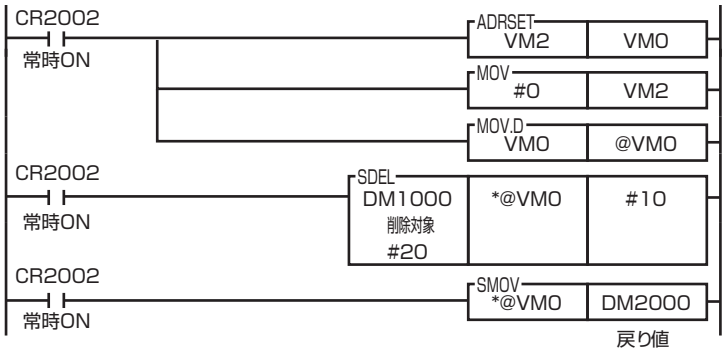
S を先頭に格納されている文字列の、n2 で指定された位置(バイト数)から n1 で指定された文字数(バイト数)分の文字列を削除し、終端コードNUL(00H)を付加して、結果を R に格納します。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.T = **DELETE** (DM1000.T, 20, 10)



ポイント

- **次のいずれかの条件が成立した場合、終端コード(NUL(00H))が格納されます。**
  - **[S]で指定した文字列に終端コード(NUL(00H))が無い場合。**
  - **終端コード(NUL(00H))を含む文字列が2000文字を超えた場合。**
  - **[S]の文字列長＜[n2]の場合。**
  - **[R]以降に格納される文字列がデバイス/変数の範囲を超えた場合。**
  - **[S]、[R]にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。**
  - **[S]、[R]に間接指定でタイマ・カウンタを指定した場合。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　EM2000.D = RHASC (**DELETE** (DM1000.T, 4, 2))

動作説明　DM1000.Tを先頭に格納されている16進表記のアスキー文字列の、2文字(バイト数)目から4文字(バイト数)分の文字列を削除します。
削除した結果の16進表記のアスキー文字列を、16ビットバイナリデータ(符号無し)に変換し、結果をEM2000.Dに格納します。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「RHASC関数」(7-172ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

TRIM 文字列両端の半角スペース・タブ削除

戻り値※1 = **TRIM** (対象文字列)
トリム

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 対象文字列※2	トリム対象となる文字列を格納した対象、またはトリム対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
R 戻り値※2※4	トリム後の文字列を格納します。	—	—	—	—	—	—	—	.T	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 T, C, CTC, CTH, Zは使用できません。

ポイント
- **TRIM関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000 シリーズ CPU ユニット、CPU 機能バージョン 2.0 以降の KV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

S を先頭に格納されている文字列の両端(文字列の先頭と末尾)にある半角スペース(20H)とタブ(09H)をすべて削除し、終端コードNUL(00H)を付加して、結果を R に格納します。

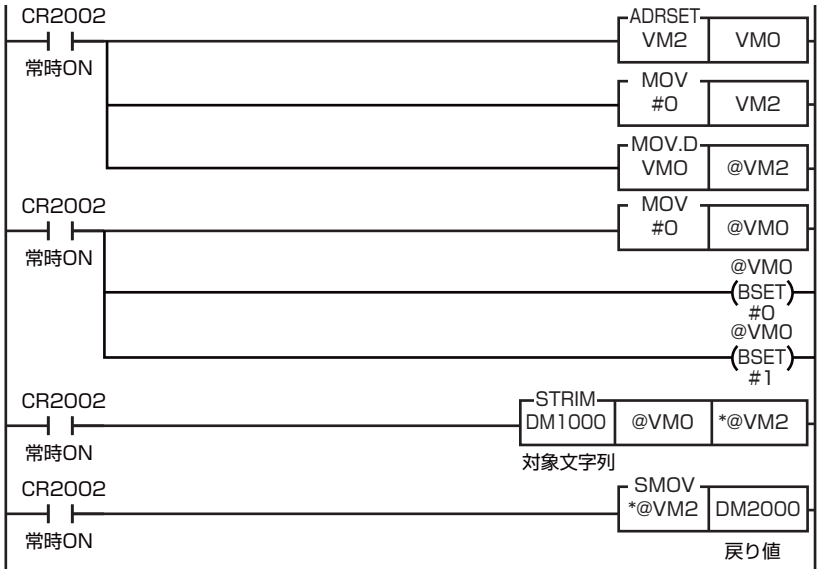
S	HT(09H)	SP(20H)
S+1	SP(20H)	K(4BH)
S+2	e(65H)	y(79H)
S+3	e(65H)	n(6EH)
⋮	c(63H)	e(65H)
	SP(20H)	SP(20H)
	HT(09H)	NUL(00H)

R	K(4BH)	e(65H)
R+1	y(79H)	e(65H)
R+2	n(6EH)	c(63H)
R+3	e(65H)	NUL(00H)

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.T = **TRIM** (DM1000.T)



サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　EM2500.D = RHASC (**TRIM** (DM1000.T))

動作説明　DM1000.Tを先頭に格納されている16進表記のアスキー文字列の両端(先頭と末尾)にある半角スペースとタブを削除します。
削除した結果の16進表記のアスキー文字列を、16ビットバイナリデータ(符号無し)に変換し、結果をEM2500.Dに格納します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

LTRIM 文字列左端の半角スペース・タブ削除

戻り値※1 = **LTRIM** (対象文字列)
左トリム

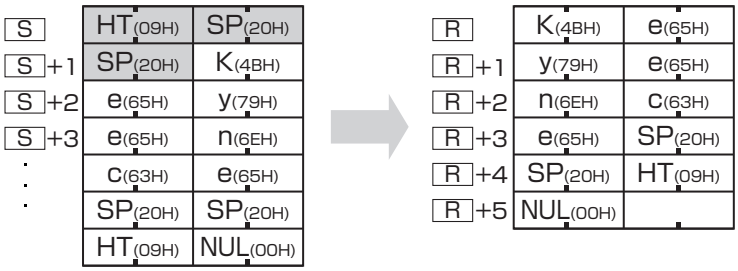
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 対象文字列※2	トリム対象となる文字列を格納した対象、またはトリム対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
R 戻り値※2※4	トリム後の文字列を格納します。	—	—	—	—	—	—	—	.T	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 T, C, CTC, CTH, Zは使用できません。

ポイント
- **LTRIM関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000 シリーズ CPU ユニット、CPU 機能バージョン 2.0 以降の KV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

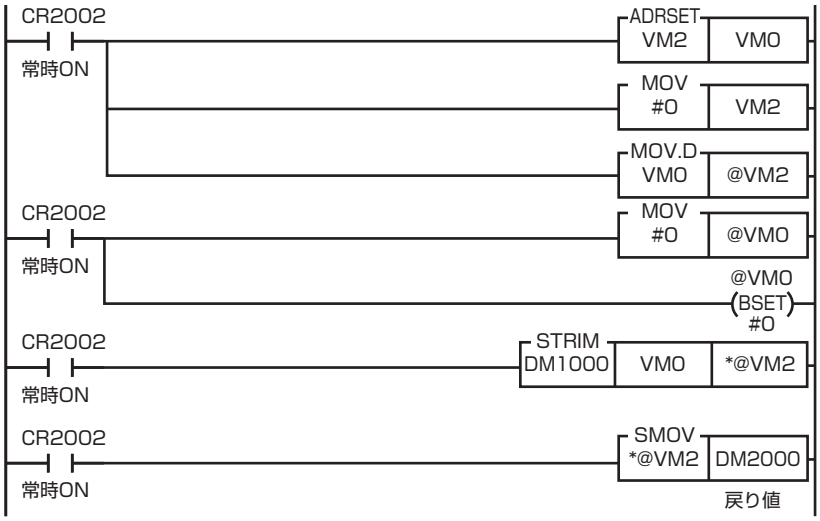
S を先頭に格納されている文字列の左端(文字列の先頭)にある半角スペース(20H)とタブ(09H)をすべて削除し、終端コードNUL(00H)を付加して、結果を R に格納します。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.T = **LTRIM** (DM1000.T)



サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　EM2500.D = RHASC (**LTRIM** (DM1000.T))

動作説明　DM1000.Tを先頭に格納されている16進表記のアスキー文字列の左端(文字列の先頭)にある半角スペースとタブを削除します。
削除した結果の16進表記のアスキー文字列を、16ビットバイナリデータ(符号無し)に変換し、結果をEM2500.Dに格納します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

RTRIM 文字列右端の半角スペース・タブ削除

戻り値※1 = **RTRIM** (対象文字列)
右トリム

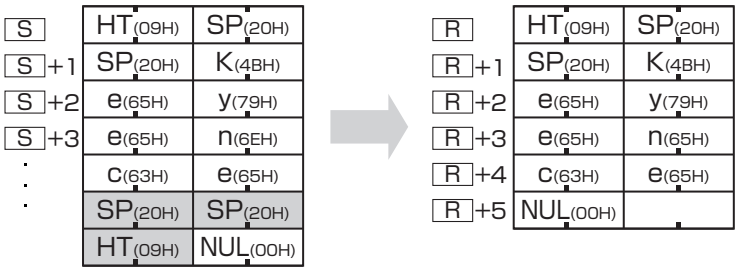
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 対象文字列※2	トリム対象となる文字列を格納した対象、またはトリム対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
R 戻り値※2※4	トリム後の文字列を格納します。	—	—	—	—	—	—	—	.T	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 T, C, CTC, CTH, Zは使用できません。

ポイント
- **RTRIM関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

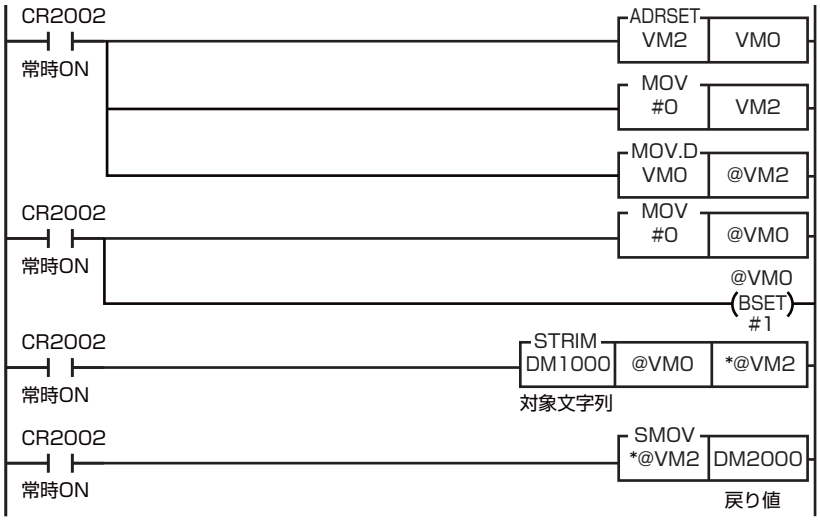
S を先頭に格納されている文字列の右端(文字列の末尾)にある半角スペース(20H)とタブ(09H)をすべて削除し、終端コードNUL(00H)を付加して、結果を R に格納します。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.T = **RTRIM** (DM1000.T)



サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　EM2500.D = RHASC (**RTRIM** (DM1000.T))

動作説明　DM1000.Tを先頭に格納されている16進表記のアスキー文字列の右端(文字列の末尾)にある半角スペースとタブを削除します。
削除した結果の16進表記のアスキー文字列を、16ビットバイナリデータ(符号無し)に変換し、結果をEM2500.Dに格納します。

参 考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

STRIM 文字列指定位置の半角スペース・タブ削除

戻り値※1 = **STRIM** (対象文字列, 削除位置指定パラメータ)
指定位置トリム

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	対象文字列※2	トリム対象となる文字列を格納した対象、またはトリム対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
S2	削除位置指定パラメータ	トリム対象となる位置を指定します(1～3)。	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値※2※4	トリム後の文字列を格納します。	—	—	—	—	—	—	—	.T	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 T, C, CTC, CTH, Zは使用できません。

ポイント
- **STRIM関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

S1を先頭に格納されている文字列から、S2で指定した位置にある半角スペースおよびタブを全て削除し、終端コードNUL(00H)を付加して、結果をRに格納します。
削除位置指定パラメータで指定する内容は次のようになります。

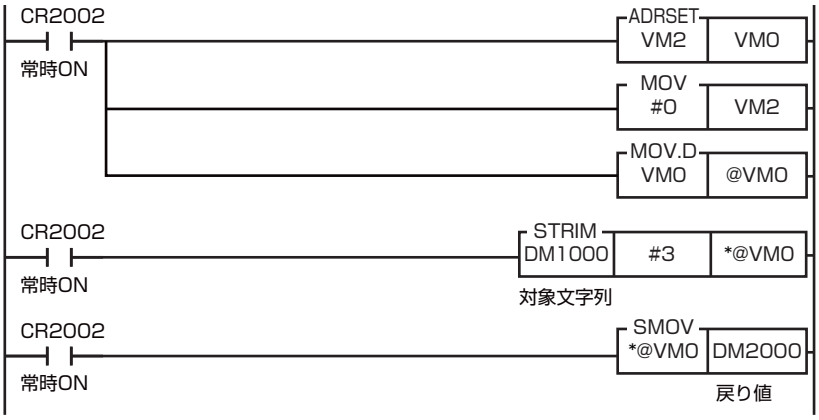
S2 値	内容
1	文字列左側の半角スペースおよびタブをすべて削除します。(LTRIM関数と同じ動作)
2	文字列右側の半角スペースおよびタブをすべて削除します。(RTRIM関数と同じ動作)
3	文字列両端の半角スペースおよびタブをすべて削除します。(TRIM関数と同じ動作)

ポイント **上記以外のパラメータを指定した場合は、関数を実行せずにS1がRに格納されます。**

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.T = **STRIM** (DM1000.T, 3)



サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　EM2500.D = RHASC (**STRIM** (DM1000.T, 3))

動作説明　DM1000.Tを先頭に格納されている16進表記のアスキー文字列の両端(先頭と末尾)にある半角スペースとタブを削除します。
削除した結果の16進表記のアスキー文字列を、16ビットバイナリデータ(符号無し)に変換し、結果をEM2500.Dに格納します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

SPLIT 文字列分離

戻り値※1 = **SPLIT** (分離元文字列, 区切り文字列, 分離後の文字列**1**,
分離後の文字列**2**,･･･, 分離後の文字列**n**)

引数/戻り値	説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
S1 分離元文字列※2	分離元となる文字列を格納した対象、または分離対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
S2 区切り文字列※2	区切り文字列を格納した対象、または区切り対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
D1 分離後の文字列1 ※2	分離後の文字列1を格納する対象を指定します。	—	—	—	—	—	—	—	.T	—	○	—
D2 分離後の文字列2 ※2	分離後の文字列2を格納する対象を指定します。	—	—	—	—	—	—	—	.T	—	○	—
⋮ ⋮	⋮	—	—	—	—	—	—	—	.T	—	○	—
Dn 分離後の文字列n ※2※4	分離後の文字列nを格納する対象を指定します。	—	—	—	—	—	—	—	.T	—	○	—
R 戻り値※2※5	分離数を格納します。SPLIT関数は、.Uの型を持ちます。※6	.U	.S	.D	.L	.F	.DF	—	—	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 引数の個数nは、1～10の範囲で指定してください。
※5 T, C, CTC, CTH, Zは使用できません。
※6 戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。

ポイント
- **SPLIT関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000 シリーズ CPU ユニット、CPU 機能バージョン 2.0 以降の KV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

S1 を S2 で分離し、結果を D1 ～ Dn 分離後文字列に格納します。
R には、分離した文字列の数が格納されます。

S1 :" 18,1A,35,E0,C3,EB"、S2 :","(カンマ)のとき

S	1(31H)	8(38H)
S+1	,(2CH)	1(31H)
S+2	A(41H)	,(2CH)
S+3	3(33H)	5(35H)
⋮	,(2CH)	E(45H)
	0(30H)	,(2CH)
	C(43H)	3(33H)
	,(2CH)	E(45H)
	B(42H)	NUL(00H)

D1	1(31H)	8(38H)
D2	1(31H)	A(41H)
D3	3(33H)	5(35H)
D4	E(45H)	0(30H)
D5	C(43H)	3(33H)
D6	E(45H)	B(42H)

R には、「6」が格納されます。

参 考

- R 分離数=0の場合、関数は実行されません。
- 0＜ R 分離数＜nの場合、余分な引数に対しては終端コード(NUL)のみ格納します。
- R 分離数 =n の場合、Dn に区切り文字を含んでいる (まだ分離できる余地が残る) 場合があります。
  分離できる余地があるか判断するだけであれば、SFIND関数で Dn 内の区切り文字の有無を調べます。
  分離できる余地があった場合、Dn を S1 に指定して再度SPLIT関数を実行することで、分離できます。

「演算フラグについて（CR2009～CR2012）」（1-18ページ）

●書式例

DM5000.U = **SPLIT** (DM2000.T, ",", EM3000.T, EM3100.T, EM3200.T)

サンプルプログラム

スクリプト記述

DM2000.T = " 日付 , データ , レコード "
DM5000.U = **SPLIT** (DM2000.T, ",", EM3000.T, EM3100.T, EM3200.T)

動作説明

DM2000.Tに格納されている文字列「日付 , データ , レコード」を、「,」(カンマ)分離してEM3000.T、EM3100.T、EM3200.Tに格納します。

参 考

KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」（2-10ページ）を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

SFIND 文字列検索

戻り値※1 = **SFIND** (検索対象, 検索文字列対象, 検索開始バイト位置)
文字列検索

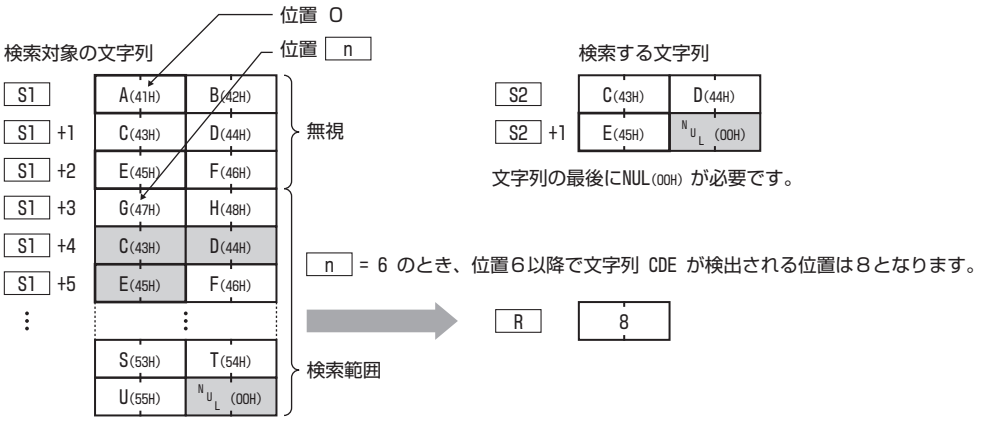
引数/戻り値		説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
S1	検索対象※2※6	検索対象となる文字列を格納した対象、または検索対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
S2	検索文字列対象※2※6	検索する文字列を格納した対象、または検索文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
n	検索開始バイト位置※4※5	検索開始位置(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
R	戻り値	SFIND関数は、.Uの型を持ちます。※7	.U	.S	.D	.L	.F	※8 .DF	—	—	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 $は使用できません。
※5 検索開始位置には検索を開始する位置として、検索対象文字列の先頭からの文字数(バイト数)を、0～1999の範囲で指定します。
※6 T, C, CTC, CTH, Zは使用できません。
※7 戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※8 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

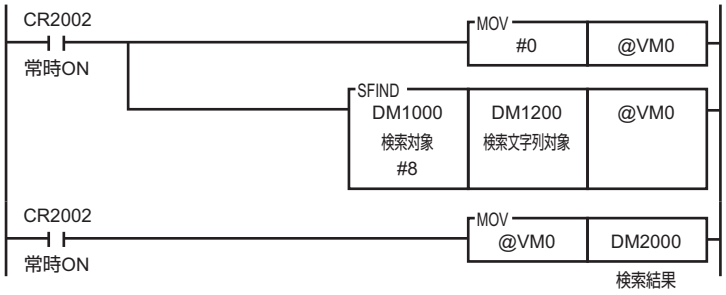
実行条件が成り立つ(TRUE)とき、S1 を先頭に格納されている検索対象文字列の、n で指定された位置から、S2 を先頭に格納されている検索文字列を検索します。
検索結果として、n 以降で初めて見つかった検索文字列の先頭文字位置(S1 からのバイト数)が、R に格納されます。
S2 で指定した文字列が見つからなかった場合、または S2 の文字列長が0の場合、R には$FFFFが格納されます。



「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.U = **SFIND** (DM1000.T, DM1200.T, 8)



ポイント

次のいずれかの条件が成立した場合、R には0が格納されます。

- **S1、S2 で指定した文字列に終端コード(NUL(00H))が無い場合。**
- **終端コード(NUL(00H))を含む文字列が2000文字を超えた場合。**
- **(S1 の文字列長)＜(n ＋ S2 の文字列長)の場合(終端コード(NUL(00H))は文字列長には含みません)。**
- **S1、S2 にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。**
- **S1、S2 に間接指定でタイマ･カウンタを指定した場合。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　DM2000.U = **SFIND** (DM1000.T, SMID (DM1200.T, 4, 6), 0)

動作説明　DM1000.Tを先頭に格納されている検索対象文字列の先頭から、DM1200.Tを先頭に格納されている文字列を、4文字(バイト数)目から6文字(バイト数)分切り出した結果の文字列を検索します。
検索結果として、検索対象文字列先頭から初めて見つかった検索文字列の先頭文字位置をDM2000.Uに格納します。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「SMID関数」（7-192ページ）
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」（2-10ページ）を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

SFINDN 文字列検索

戻り値※1 = **SFINDN** (検索対象, 検索文字列対象, 検索対象バイト数, 検索開始バイト位)
文字列検索N

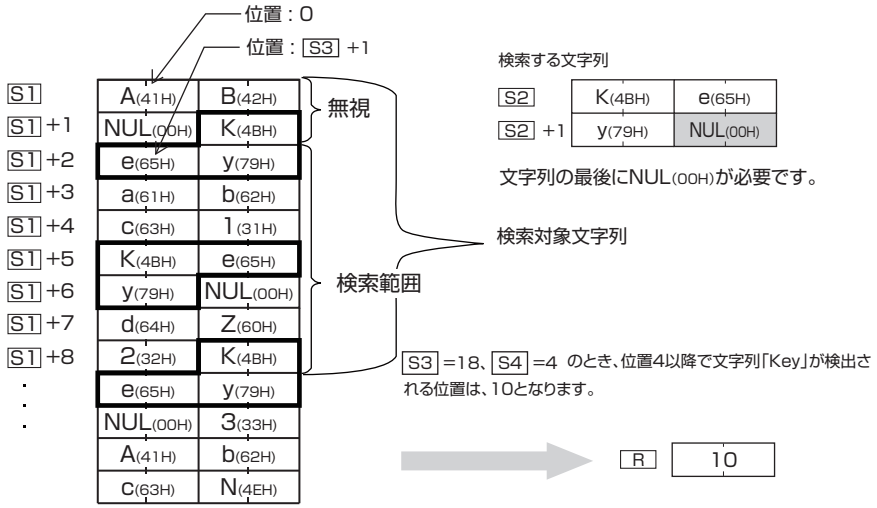
引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	検索対象※2※4	検索対象となる文字列を格納した対象、または検索対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
S2	検索文字列対象※2※4	検索する文字列を格納した対象、または検索文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
S3	検索対象バイト数	検索対象の文字列長(バイト数)を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
S4	検索開始バイト位置	検索開始バイト位置を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	SFINDN関数は、.Uの型を持ちます。※5	.U	.S	.D	.L	.F	※6 .DF	—	.T	—	○	—

※1 戻り値は省略できます。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 T, C, CTC, CTH, Zは使用できません。
※5 戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※6 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **SFINDN関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

実行条件が成り立つ(TRUE)とき、S1を先頭に、検索対象バイト数の文字列のS4から、S2を先頭に格納されている文字列を検索します。
Rには、S3で指定した文字列中のS4で指定された位置以降で、はじめて見つかったS2で指定した文字列の開始位置(バイト数)が格納されます。
S2で指定した文字列が見つからなかった場合、またはS2の文字列長が0の場合、
Rには「$FFFF」が格納されます。

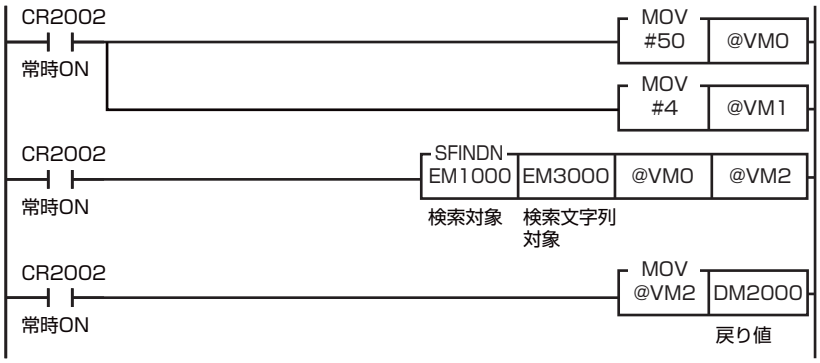


「演算フラグについて（CR2009～CR2012）」（1-18ページ）

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.U = SFINDN (EM1000.T, EM3000.T, 50, 4)



サンプルプログラム

スクリプト記述

EM1000.T = "abcdefghijklmnabcdefghijklmn"

EM3000.T = "abc"

DM2000.U = **SFINDN** (EM1000.T, EM3000.T, 50, 4)

動作説明

EM1000.Tに格納されている文字列中の50文字から、EM3000.Tの文字列(abc)を検索します。EM1000.Tの4文字目から検索を開始します。

参 考

KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」（2-10ページ）を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

INSTR 文字列検索

戻り値※1 = INSTR (検索開始位置, 検索対象, 検索文字列対象)
文字列検索N

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
n	検索開始位置※5	検索開始位置(バイト数)を指定します。(0～1999)	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
S1	検索対象※2※6	検索対象となる文字列を格納した対象、または検索対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
S2	検索文字列対象※2※6	検索する文字列を格納した対象、または検索文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
R	戻り値	INSTR関数は、.Uの型を持ちます。※7	.U	.S	.D	.L	.F	※8 .DF	—	—	—	○	—

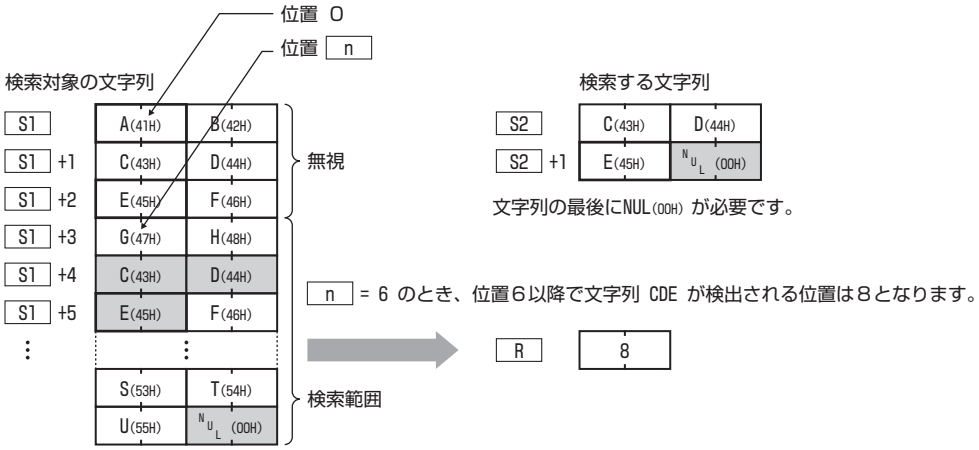
※1 戻り値は省略できます。省略した場合、演算結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 $は使用できません。
※5 検索開始位置には検索を開始する位置として、検索対象文字列の先頭からの文字数(バイト数)を、0～1999の範囲で指定します。
※6 T, C, CTC, CTH, Zは使用できません。
※7 戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※8 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

実行条件が成り立つ(TRUE)とき、S1 を先頭に格納されている検索対象文字列の、n で指定された位置(バイト数)から、S2 を先頭に格納されている検索文字列を検索します。
検索結果として、n 以降で初めて見つかった検索文字列の先頭文字位置(S1 からのバイト数)が、R に格納されます。
S2 で指定した文字列が見つからなかった場合、または S2 の文字列長が0の場合、R には$FFFFが格納されます。

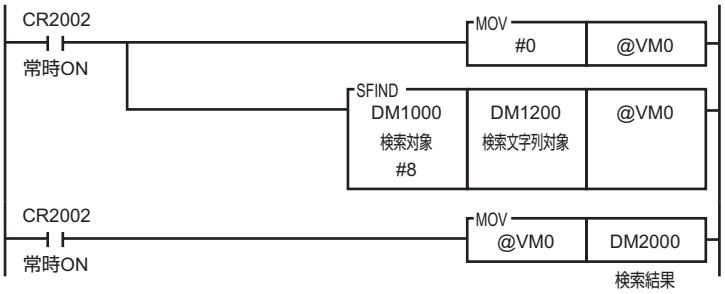


文字列の最後にNUL(00H) が必要です。

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.U = **INSTR** (8, DM1000.T, DM1200.T)



ポイント

次のいずれかの条件が成立した場合、R には0が格納されます。

- **S1、S2 で指定した文字列に終端コード(NUL(00H))が無い場合。**
- **終端コード(NUL(00H))を含む文字列が2000文字を超えた場合。**
- **(S1 の文字列長)＜(n +S2 の文字列長)の場合(終端コード(NUL(00H))は文字列長には含みません)。**
- **S1、S2 にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。**
- **S1、S2 に間接指定でタイマ･カウンタを指定した場合。**

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　DM2000.U = **INSTR** (0, DM1000.T, SMID (DM1200.T, 4, 6))

動作説明　DM1000.Tを先頭に格納されている検索対象文字列の先頭から、DM1200.Tを先頭に格納されている文字列を、4文字(バイト数)目から6文字(バイト数)分切り出した結果の文字列を検索します。
検索結果として、検索対象文字列先頭から初めて見つかった検索文字列の先頭文字位置をDM2000に格納します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
「SMID関数」(7-192ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

MEMO

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

FIND 文字列検索

戻り値※1 = **FIND** (検索対象, 検索文字列対象)
文字列検索

引数/戻り値	説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
S1 検索対象※2※4	検索対象となる文字列を格納した対象、または検索対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
S2 検索文字列対象※2※4	検索する文字列を格納した対象、または検索文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
R 戻り値	FIND関数は、.Uの型を持ちます。※5	.U	.S	.D	.L	.F	※6 .DF	—	—	—	○	—

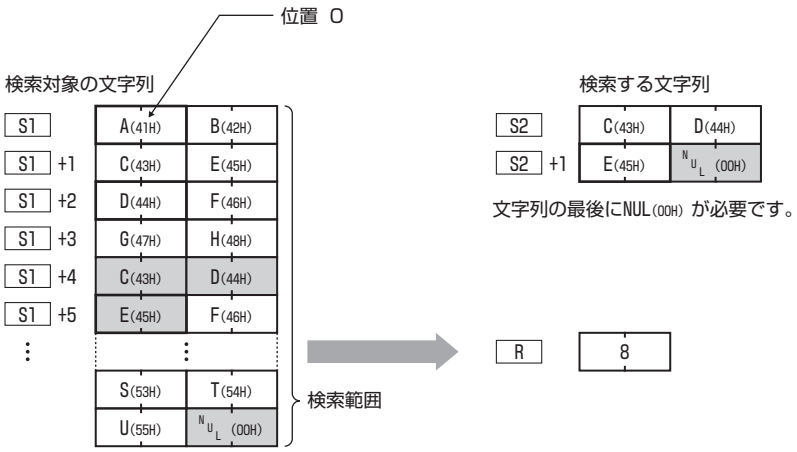
※1 戻り値は省略できます。省略した場合、演算結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください(半角31文字まで)。
※4 T, C, CTC, CTH, Zは使用できません。
※5 戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※6 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

実行条件が成り立つ(TRUE)とき、S1 を先頭に格納されている検索対象文字列から、S2 を先頭に格納されている検索文字列を検索します。
検索結果として、初めて見つかった検索文字列の先頭文字位置(S1 からのバイト数)が、R に格納されます。

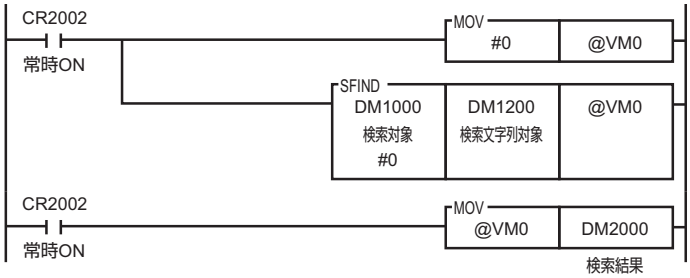


文字列の最後にNUL(00H) が必要です。

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000 = **FIND** (DM1000.T, DM1200.T)



ポイント

次のいずれかの条件が成立した場合、R には0が格納されます。

- **S1、S2 で指定した文字列に終端コード(NUL(00H))が無い場合。**
- **終端コード(NUL(00H))を含む文字列が2000文字を超えた場合。**
- **(S1 の文字列長)＜(S2 の文字列長)の場合。**
- **S1、S2 にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。**
- **S1、S2 に間接指定でタイマ・カウンタを指定した場合。**
- **S2 の文字列長が 0 の場合、または文字列がみつからなかった場合は、65535 が R に入ります。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述

DM2000.U = **FIND** (DM1000.T, SMID (DM1200.T, 4, 6))

動作説明

DM1000.Tを先頭に格納されている検索対象文字列の先頭から、DM1200.Tを先頭に格納されている文字列を、4文字(バイト数)目から6文字(バイト数)分切り出した結果の文字列を検索します。
検索結果として、検索対象文字列先頭から初めて見つかった検索文字列の先頭文字位置をDM2000.Uに格納します。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「SMID関数」(7-192ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

DISS 文字列バイト分離

文字列分離

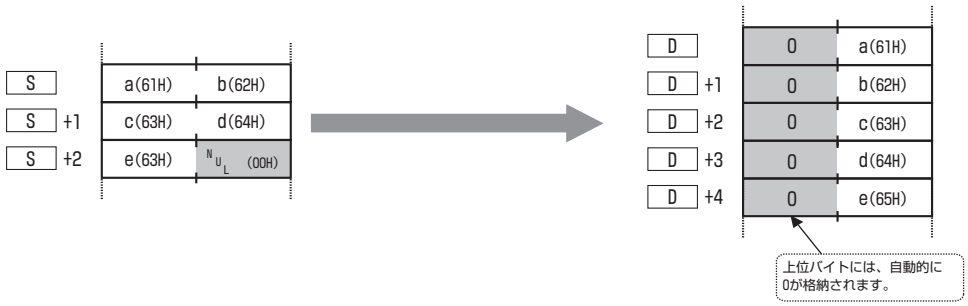
DISS ([実行条件,]※1 分離対象, 格納先)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 分離対象※2※4	分離対象となる文字列を格納した対象、または分離対象文字列を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	○
D 格納先※2※5	分離した文字列(ワード単位)を格納する対象を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ビットデータではチャンネルの先頭を指定してください。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 T, C, CR, CTC, CTH, Zは使用できません。
※5 T, C, CTC, CTH, Zは使用できません。

動作説明

実行条件が成り立つ(TRUE)とき、S を先頭に格納されているバイト単位の文字列を、D を先頭にしてワード単位に分離して格納します。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DISS (R3000, DM1000.T, DM2000)

ポイント

次のいずれかの条件が成立した場合、D には0が格納されます。

- **S で指定した文字列に終端コード(NUL(00H))が無い場合。**
- **終端コード(NUL(00H))を含む文字列が2000文字を超えた場合。**
- **S 、D にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。**
- **S 、D に間接指定でタイマ･カウンタを指定した場合。**

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述 **DISS** (SMID (DM1000.T, EM1000, EM1001), DM2000)

動作説明 毎スキャン(実行条件無し)、DM1000.Tを先頭に格納されている文字列を、EM1000で示される文字位置(バイト数)から、EM1001で示される文字数(バイト数)分切り出します。
切り出した結果のバイト単位の文字列を、DM2000を先頭にしてワード単位に分離して格納します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
「SMID関数」(7-192ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

UNIS バイト文字列結合

文字列結合

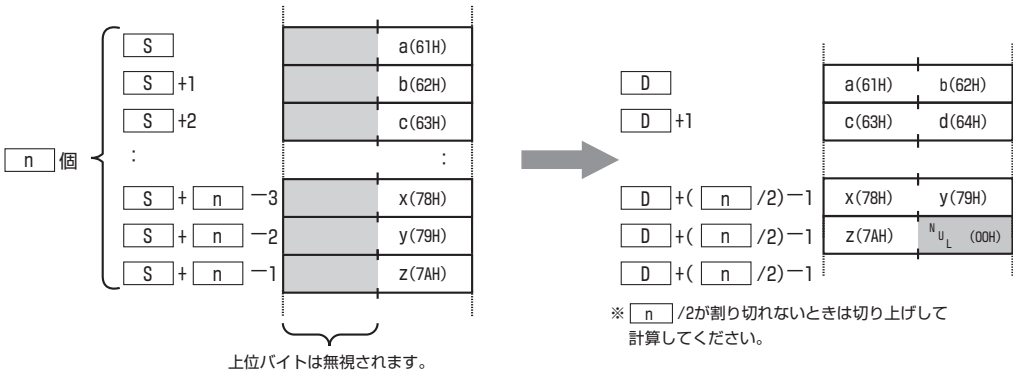
UNIS ([実行条件,]※1 結合対象, 格納先, 結合文字数)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 結合対象※2※5	結合対象となる文字列(ワード単位)を格納した対象、またはは結合対象文字列(ワード単位)を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	○
D 格納先※2※6	結合した文字列を格納する対象を指定します。	—	—	—	—	—	—	—	.T	—	○	—
n 結合文字数※4	結合する文字数(バイト数(ワード単位数))を指定します。(1～1999)	.U	.U	.U	.U	—	—	—	—	※3 ○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ビットデータではチャンネルの先頭を指定してください。
※3 $は使用できません。
※4 結合文字数には結合する文字(ワード単位)数を、1～1999の範囲で指定します。
※5 T, C, CTC, CTH, Zは使用できません。
※6 T, C, CR, CTC, CTH, Zは使用できません。

動作説明

実行条件が成り立つ(TRUE)とき、S を先頭に格納されているワード単位の文字列を、n で指定された文字数(ワード単位数)分だけバイト単位に結合し、D を先頭にして、終端コードNUL(00H)を付加して格納します。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

UNIS (R3000, DM1000, DM2000.T, 16)

ポイント

次のいずれかの条件が成立した場合、関数は実行されません。

- **[n]の値が0または2000以上の場合。**
- **終端コード(NUL(00H))を含む文字列が2000文字を超えた場合。**
- **[S]、[D]にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。**
- **[S]、[D]に間接指定でタイマ・カウンタを指定した場合。**

サンプルプログラム

●ビットデータを指定する場合

スクリプト記述 **UNIS** (MR3000.U < 10, MR1000.U, MR2000.T, 10)

動作説明 MR3000.Uが10より小さいとき、MR1000.Uを先頭に格納されているワード単位の文字列を、10文字(ワード単位数)分だけバイト単位に結合し、MR2000.Tを先頭にして終端コードNUL(00H)を付加して格納します。

●他の関数に組み込んで使用する場合

スクリプト記述 **UNIS** (DM1000, DM2000.T, ANDA (EM1000, $FF))

動作説明 毎スキャン(実行条件無し)、DM1000を先頭に格納されているワード単位の文字列を、EM1000の下位8ビット(抽出)で示される文字数(ワード単位数)分だけバイト単位に結合し、DM2000.Tを先頭にして終端コードNUL(00H)を付加して格納します。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「ANDA関数」(7-22ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

CHR　16進アスキー文字コード→文字変換

戻り値※1 = **CHR** (変換対象)
キャラクタ変換

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換対象	変換対象のASCIIコードを格納した対象、またはASCIIコードデータを指定します。(16ビットデータ)	.U	.U	.U	.U	—	—	—	—	※2 ○	○	○
R 戻り値※3	CHR関数は、.Tの型を持ちます。	—	—	—	—	—	—	—	.T	—	○	—

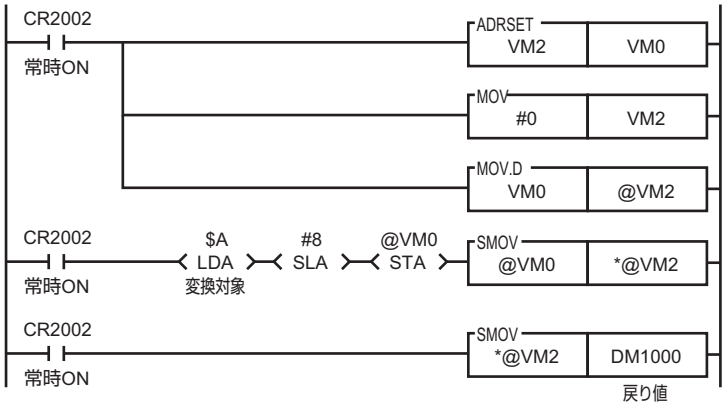
※1　戻り値は省略できます。省略した場合、演算結果はワークデバイス/ワークメモリに格納されます。
※2　定数の値は0～255の範囲。対象を指定し、その値が0～255以上のとき上位8ビットを切り捨てた値が使用されます。
※3　T, C, CR, CTC, CTH, Zは使用できません。

動作説明

S に格納されたASCIIコードデータを、終端コードNUL(00H)を付加した1バイトの文字列に変換し、結果を R に格納します。

●書式例

DM1000.T = CHR ($0A)



サンプルプログラム

スクリプト記述　DM2000.T = DM2000.T + **CHR** ($0D) + **CHR** ($0A)

動作説明　文字列DM2000.Tの最後にCRコード(0DH)とLFコード(0AH)を加えます。

参考
- 変換対象データには、ASCIIコード(1バイト)を指定します。
  「ASCIIコード表」(付-53ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

MEMO

ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

CPSASC CIP文字列変換

戻り値※1 = **CPSASC** (対象データ格納先)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	変換対象データ格納先※2	変換前のCIPの文字列型データが格納されている対象の先頭を指定します。	.U	.S	.D	.L	.F	.DF	.B	.T	—	○	—
R	戻り値	CPSASC関数は、.Tの型を持ちます。	—	—	—	—	—	—	—	.T	—	—	—

※1　戻り値の省略は可能です。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2　ビットデータを指定する場合は、チャンネルの先頭を指定してください。

ポイント
- **CPSASC関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

S を先頭に格納されているCIPの文字列型データを文字列に変換し、結果を R に格納します。
CPSASC関数は、CIPの文字列型データを変換し、文字列として処理するための関数です。
CIPのデータ型については、『KV EtherNet/IP機能ユーザーズマニュアル』を参照してください。
CPSASC関数を実行すると、CIPの文字列型データを変換して戻り値に格納します。

CIPの文字列型データ(“ABCDE”)

	bit 15	bit 0
S +0	5(0005H)	
S +1	B(42H)	A(41H)
S +2	D(44H)	C(43H)
S +3		E(45H)

CIPの文字列型データは、先頭の1ワードでデータサイズを指定し、文字列を下位バイト→上位バイトの順に格納します。

変換後の文字列データ

bit 15	bit 0
A(41H)	B(42H)
C(43H)	D(44H)
E(45H)	NULL(00H)

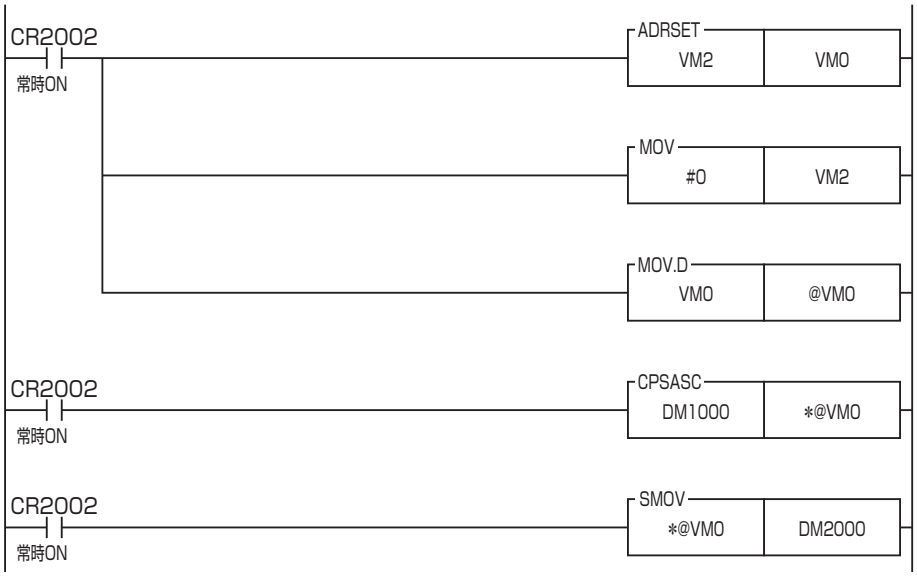
戻り値

最後にNULL(00H)を付加して、データサイズ分の文字列の上位バイト、下位バイトを入れ替えて格納します。

詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM2000.T = **CPSASC** (DM1000)



ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

RCPSASC CIP文字列逆変換

戻り値※1 = **RCPSASC** (対象データ格納先,変換後データ格納先)

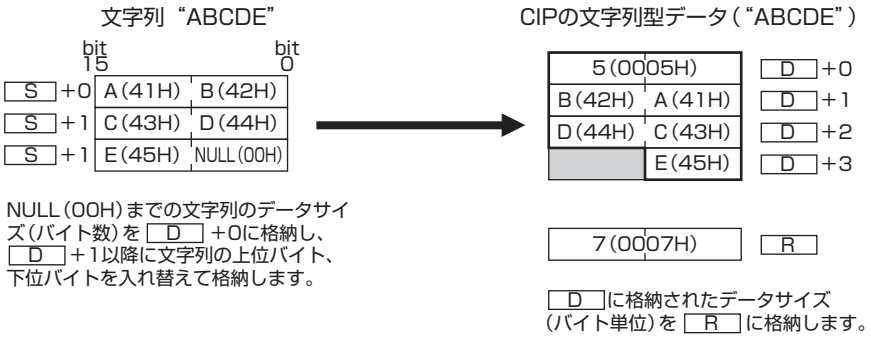
引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	変換対象データ格納先	変換前の文字列が格納されている対象の先頭を指定します。	—	—	—	—	—	—	—	.T	※3 ○	○	—
D	変換後データ格納先	変換後のCIPの文字列型データを格納する対象の先頭を指定します。	.U	.S	.D	.L	.F	.DF	.B	.T	—	○	
R	戻り値※2	D に格納された文字数(バイト単位)が格納されます。RCPSASC 関数は、.Uの型を持ちます	.U	.S	.D	.L	.F	.DF	—	—	—	—	—

※1　戻り値の省略は可能です。省略した場合、変換結果はワークデバイス/ワークメモリに格納されます。
※2　戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　定数で文字列を指定する場合は、" "で囲ってください。

ポイント
- **RCPSASC関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000 シリーズ CPU ユニット、CPU 機能バージョン 2.0 以降の KV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

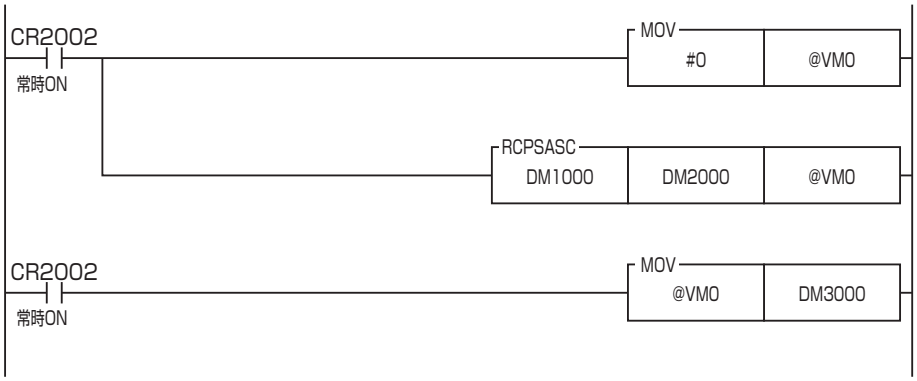
S に格納されている文字列をCIPの文字列型データに変換し、結果を D に格納します。変換後に D に格納された文字数(バイト単位)を R に格納します。
RCPSASC関数は、文字列をCIPの文字列型データに変換するための関数です。
CIPのデータ型については、『KV EtherNet/IP機能ユーザーズマニュアル』を参照してください。
RCPSASC関数を実行すると、文字列をCIPの文字列型データに変換します。



詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「文字処理命令」を参照してください。

●書式例

DM3000 = **RCPSASC** (DM1000.T, DM2000)



ASC
RASC
DASC
STR
RDASC
HASC
RHASC
FASC
RFASC
VAL
LEN
SRGHT
RIGHT
SLEFT
LEFT
SMID
MID
SRPLC
REPLACE
SINS
INSERT
SDEL
DELETE
TRIM
LTRIM
RTRIM
STRIM
SPLIT
SFIND
SFINDN
INSTR
FIND
DISS
UNIS
CHR
CPSASC
RCPSASC

MEMO

8

拡張関数

データ処理関数 8-2
テーブル処理関数 8-48
時計処理関数 8-70
高速処理関数 8-78
PID関数 8-84
ログ関数 8-86
ストレージ関数 8-90
アクセスウィンドウ関数 8-118
フリーランカウンタ関数 8-124
センサ設定関数 8-126
サイクリック通信リフレッシュ関数 8-132

SEG　4ビットデータの7セグメントデコード

戻り値※1 = **SEG** (デコード対象, 対象桁位置)
7セグメントデコード

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S デコード対象※2	デコード対象のデータを格納した対象、またはデコード対象データを指定します。	.U	.U	.D	.D	—	—	—	—	—	○	○
n 対象桁位置※3	デコードする桁位置を格納した対象、またはデコード桁位置を指定します。(.Uのとき0～3、.Dのとき0～7)	.U	.U	.U	.U	—	—	—	—	—	○	○
R 戻り値	SEG関数は、.Uの型を持ちます。※4	.U	.S	.D	.L	.F	※5 .DF	—	—	—	○	—

※1　戻り値の省略は可能です。省略した場合、演算結果は内部レジスタに格納されます。

※2　S デコード対象にビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。

※3　n 対象桁位置に対象を指定した場合で、格納された値が0～3または0～7を超えた場合は以下のようになります。
デコード対象のサフィックスが.Uの場合：指定した対象の下位2ビットの値が使用されます。
デコード対象のサフィックスが.Dの場合：指定した対象の下位3ビットの値が使用されます。
例えば、S デコード対象のサフィックスが .Uで、n にDM01000が指定され、その値が23(2進数：0000 0000 0001 01<u>11</u>)の場合、下位2ビットの値は3なので n の値として3が使用されます。

※4　戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。

※5　デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

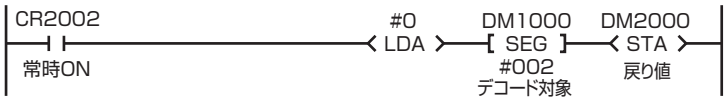
S に格納された16ビット/32ビットデータの、n で指定された桁数の4ビット(1桁)を、7セグメント表示用データ(8ビット)にデコードして、結果を R に格納します。

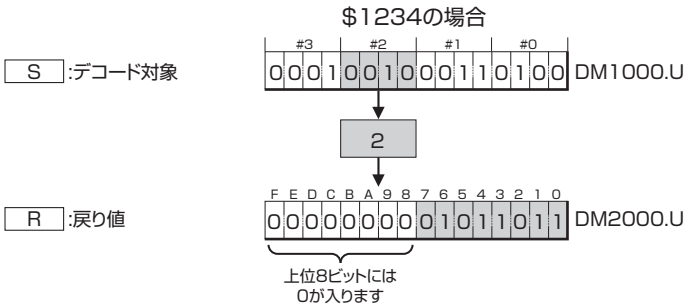
「演算フラグについて(CR2009～CR2012)」(1-18ページ)

●書式例

- S :デコード対象が16ビット(.U/.S)の場合

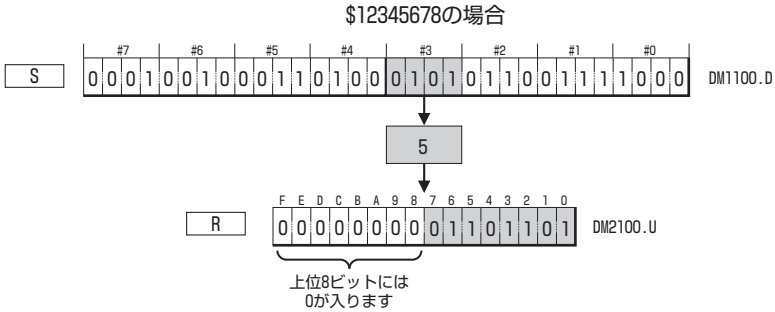
DM2000.U = **SEG** (DM1000.U, 2)





- S :デコード対象が32ビット(.D/.L)の場合

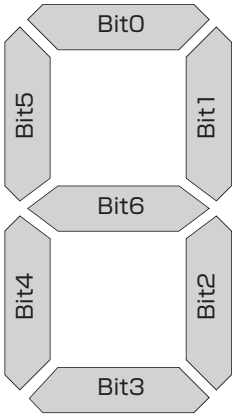
DM2100.U = **SEG** (DM1100.D, 3)



詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ処理命令」を参照してください。

●7セグメント表示用データ表

指定した桁の値	16ビット内部レジスタ							
	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$0	0	0	1	1	1	1	1	1
$1	0	0	0	0	0	1	1	0
$2	0	1	0	1	1	0	1	1
$3	0	1	0	0	1	1	1	1
$4	0	1	1	0	0	1	1	0
$5	0	1	1	0	1	1	0	1
$6	0	1	1	1	1	1	0	1
$7	0	0	1	0	0	1	1	1
$8	0	1	1	1	1	1	1	1
$9	0	1	1	0	1	1	1	1
$A	0	1	1	1	0	1	1	1
$B	0	1	1	1	1	1	0	0
$C	0	0	1	1	1	0	0	1
$D	0	1	0	1	1	1	1	0
$E	0	1	1	1	1	0	0	1
$F	0	1	1	1	0	0	0	1



サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述　R2000.U = **SEG** (TBCD (DM1000), 3)

動作説明　DM1000に格納されている16ビットバイナリデータを4桁のBCD(16ビット)データに変換し、変換結果の1000の桁(3桁目)の値を7セグメント表示用データにデコードして、対応する出力リレー(R2000～R2015(R2007))をONします。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「TBCD関数」(7-80ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

MEMO

SEG
BCNT
DCNT
SER
DSER
MAX
MIN
AVG
WSUM
BSUM
CRC
ZRES
BCMP
BCMPI
RND
SORT
SORTN

BCNT 対象のONビットカウント

戻り値※1 = **BCNT** (演算対象)
ビットカウント

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 演算対象	演算対象、または演算対象データを指定します。	.U	.U	.D	.D	—	—	—	—	—	○	○
R 戻り値	SEG関数は、.Uの型を持ちます。※2	.U	.S	.D	.L	.F	※3 .DF	—	—	—	○	—

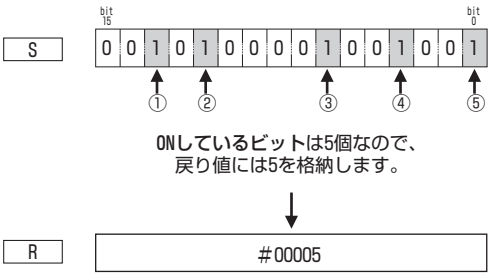
※1　戻り値の省略は可能です。省略した場合、演算結果は内部レジスタに格納されます。
※2　戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※3　デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

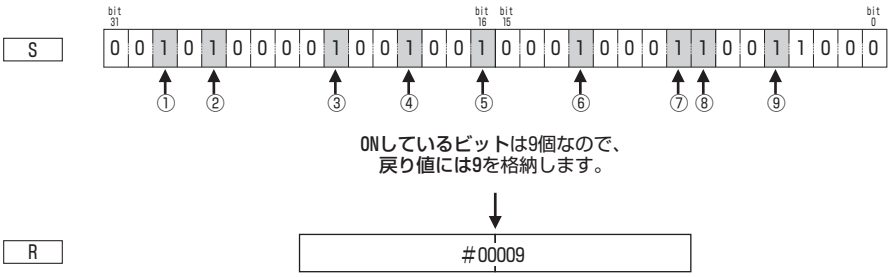
動作説明

S に格納された(16ビット/32ビット)バイナリデータの、ONしているビットの数を数え、結果を戻り値に格納します。

- **S :演算対象が16ビット(.U/.S)の場合**



- **S :演算対象が32ビット(.D/.L)の場合**



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ処理命令」を参照してください。

●書式例

DM2000.U = **BCNT** (DM1000.U)

サンプルプログラム

●ビットデータをカウントする場合

スクリプト記述　DM2000.U = **BCNT** (MR1000.D)

動作説明　MR1000.Dに格納された32ビットバイナリデータの、ONしているビットを数えます。結果をDM2000.Uに格納します。

ラダー変換

●他の関数を組み込んで使用する場合

スクリプト記述　DM2000.U = **BCNT** (ORA (DM1000.D, DM1100.D))

動作説明　DM1000.Dに格納されたデータと、DM1100.Dに格納されたデータで論理和演算をおこない、演算結果のONしているビットを数えてDM2000.Uに格納します。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「ORA関数」(7-24ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

DCNT 指定ブロック内の検出データカウント

戻り値※1 = **DCNT** (検出対象, 検出範囲先頭, 検出データ数)
データカウント

引数/戻り値	説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
S1 検出対象※2	検出する値を格納した対象、または検出値を指定します。	.U	.U	.D	.D	.D	—	—	—	○	○	○
S2 検出範囲先頭※2	検出するデータブロックの先頭を指定します。	S1 と同じ型として扱われます。					—	—	—	—	○	—
n 検出データ数	検出するブロックのデータ数の格納先、または検出データ数を指定します	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	DCNT関数は、.Uの型を持ちます。※3	.U	.S	.D	.L	.F	※4 .DF	—	—	—	○	—

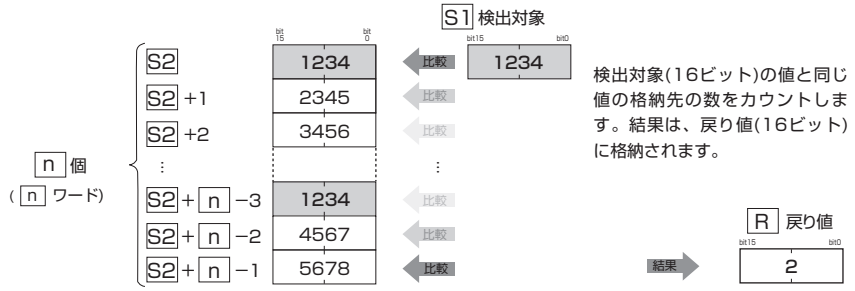
※1　戻り値の省略は可能です。省略した場合、演算結果は内部レジスタに格納されます。
※2　ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。
※3　戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※4　デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

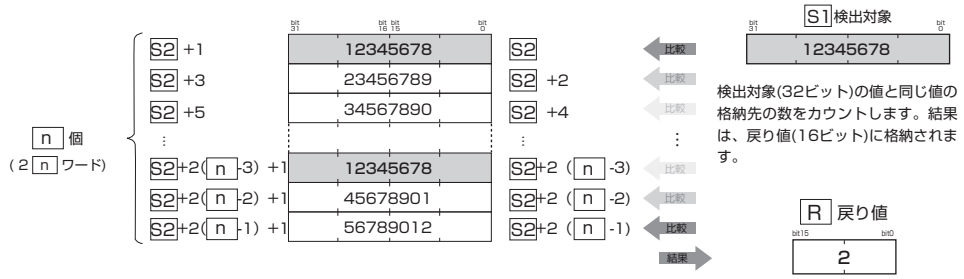
動作説明

S2 を先頭とする、n で指定された個数のブロックデータを、S1 で指定されたデータとそれぞれ比較し、一致したデータの数を R **戻り値**に格納します。

- **S1 :検出対象が16ビット(.U/.S)の場合**



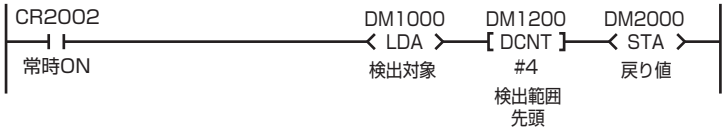
- **S1 :検出対象が32ビット(.D/.L)の場合**



「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ処理命令」を参照してください。

●書式例

DM2000.U = **DCNT** (DM1000.U, DM1200.U, 4)



ポイント　**n 検出データ数の値が0の場合、S1 検出対象の値が、R 戻り値に格納されます。**

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述　DM2000.U = **DCNT** (COM (DM1000.D), MR1000.D, EM2000)

動作説明　MR1000(.D)を先頭とする、EM2000で示される検出データ数分の32ビットバイナリデータ(符号無し)ブロックを、DM1000.Dに格納されている32ビットバイナリデータ(符号無し)をビット反転した結果の値とそれぞれ比較し、一致したデータの数をDM2000.Uに格納します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「COM関数」（7-30ページ）
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」（2-10ページ）を参照してください。

SER 指定ブロック内のデータ検出

戻り値※1 = SER (検出対象, 検出範囲先頭, 検出データ数)
検索

引数/戻り値	説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
S1 検出対象※2	検出する値を格納した対象、または検出値を指定します。	.U	.U	.D	.D	.D	—	—	—	○	○	○
S2 検出範囲先頭※2	検出するデータブロックの先頭を指定します。	S1と同じ型として扱われます。					—	—	—	—	○	—
n 検出データ数	検出するブロックのデータ数の格納先、または検出データ数を指定します	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	SER関数は、.Uの型を持ちます。※3	.U	.S	.D	.L	.F	※4 .DF	—	—	—	○	—

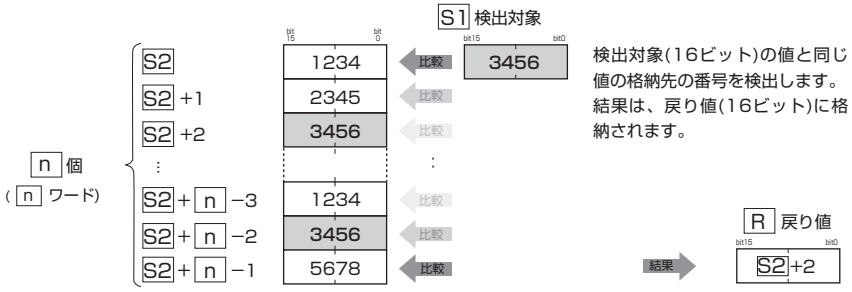
※1 戻り値の省略は可能です。省略した場合、演算結果は内部レジスタに格納されます。
※2 ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。
※3 戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※4 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

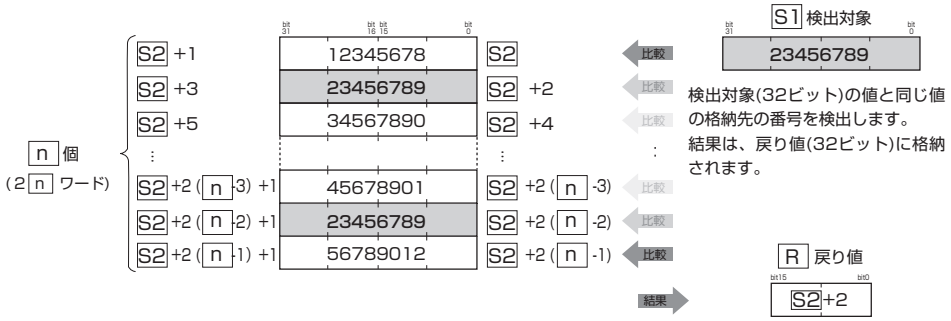
動作説明

S2 を先頭とする、n で指定された個数のブロックデータを、S1 で指定されたデータとそれぞれ比較し、一致したデータが格納されているデバイス番号を R に格納します。
一致するデータが複数個検出された場合は、一番小さい番号が格納されます。

- **S1 :検出対象が16ビット(.U/.S)の場合**



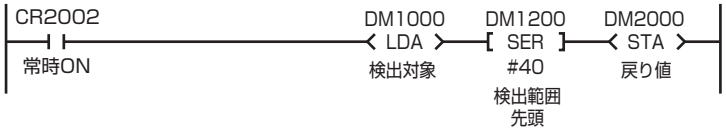
- S1 :検出対象が32ビット(.D/.L)の場合



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ処理命令」を参照してください。

●書式例

DM2000.U = **SER** (DM1000.U, DM1200.U, 40)



ポイント

- **データがみつからなかった場合や、指定した n 検出データ数が 0 だった場合は、S1 検出対象の値が R 戻り値に格納されます。**
- **検出範囲先頭にビットデータを指定した場合、検索結果(戻り値)には16ビット単位ではなくR(MR、LR)0000を基点とした10進単位に変換された値が格納されます。**
  (例)検索結果がMR82312の場合:82312→823×16+12＝13180が格納されます。

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述　DM2000.U = **SER** (NEG (DM1000.D), MR1000.D, EM2000.U)

動作説明　MR1000.Dを先頭とする、EM2000で示される検出データ数分の32ビットバイナリデータ(符号無し)ブロックを、DM1000.Dに格納されている32ビットバイナリデータ(符号無し)を符号反転した結果の値とそれぞれ比較し、一致したデータが格納されている番号をDM2000.Uに格納します。

参考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「NEG関数」(7-32ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

DSER 指定範囲内のデータ検索

戻り値※1 = **DSER** (検出対象, 検出範囲先頭, 検出対象個数, 検索個数格納対象)
データ検索

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 検出対象※2	検索データ、または検索データの格納先を指定します。	.U	.U	.D	.D	.D	—	—	—	○	○	○
S2 検出範囲先頭※2	検索する範囲の先頭を指定します。	S1 と同じ型として扱われます。					—	—	—	—	○	—
n 検出対象個数※2	検索する対象の個数、または個数の格納先を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
D 検索個数格納対象※2	検出結果(検出個数)を格納する対象を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
R 戻り値	検出位置(検出範囲先頭からのオフセット)を格納する対象を指定します。DSER関数は、.Uの型を持ちます。※3	.U	.S	.D	.L	.F	※4 .DF	—	—	—	○	—

※1 戻り値の省略は可能です。省略した場合、演算結果はワークメモリに格納されます。
※2 ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。
※3 戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※4 デバイスZに「.DF」サフィックスは指定できません。

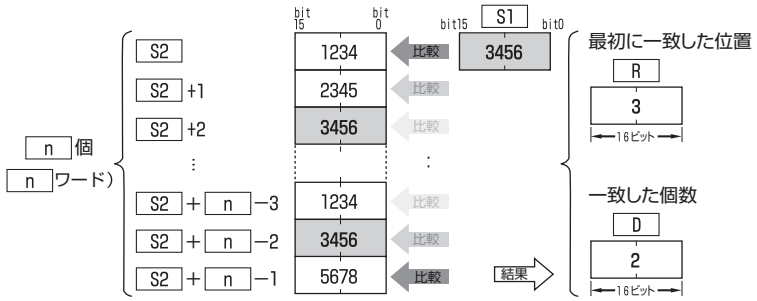
ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

S1 に格納されたバイナリデータをキーワードとして、S2 から始まる n のワードデータの範囲を検索します。

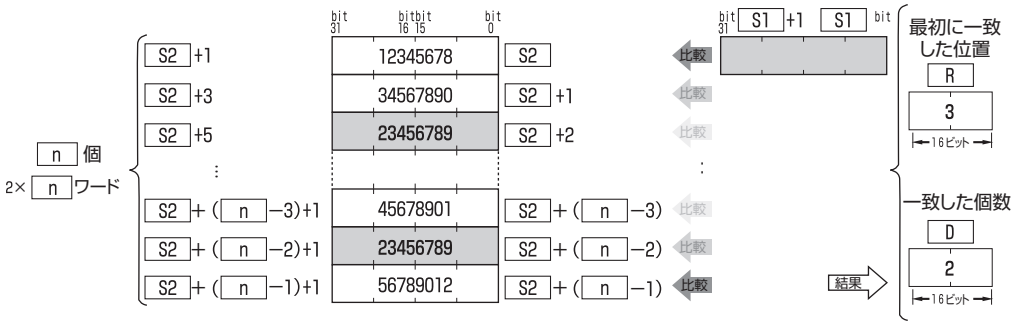
- S1 と一致したワードデータの個数が、D に格納されます。
- 最初に一致したワードデータの位置(S1 からの相対位置)を R に格納します。
- 一致するデータが見つからなかった場合、D および R には0を格納します。

- S1 :検索データが16ビット(.U)の場合



検出対象(16ビット)の値と同じ値の格納先の数をカウントします。検索結果(位置)は、戻り値(16ビット)に格納されます。

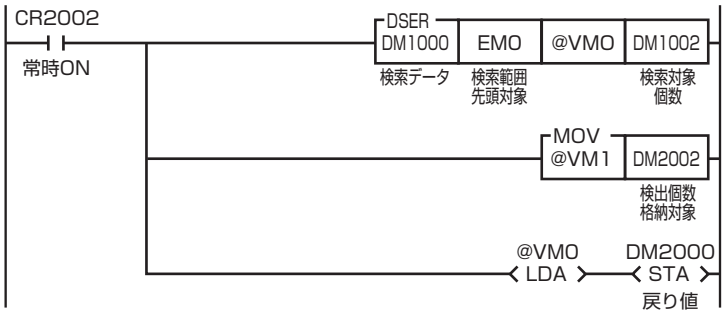
- S1 :演算対象が32ビットの場合



検出対象(32ビット)の値と同じ値の格納先の数をカウントします。検索結果(位置)は、戻り値(16ビット)に格納されます。

●書式例

DM2000.U = **DSER** (DM1000.U, EM0, DM1002, DM2002)



SEG
BCNT
DCNT
SER
DSER
MAX
MIN
AVG
WSUM
BSUM
CRC
ZRES
BCMP
BCMPI
RND
SORT
SORTN

MAX 指定ブロック内の最大値検索

戻り値※1 = **MAX** (検出対象, 検索範囲データ数)
最大値検索

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	検出対象※2※3※4	検索する範囲の先頭を指定します。	.U	.S	.D	.L	.F	※6 .DF	—	—	—	○	—
n	検索範囲データ数※2	検索する範囲のデータ数の格納先、または検索範囲データ数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	MAX関数は、第1引数と同じ型を持ちます。※5	.U	.S	.D	.L	.F	※6 .DF	—	—	—	○	—

※1 戻り値の省略は可能です。省略した場合、演算結果は内部レジスタに格納されます。
※2 サフィックスにより扱える数値の範囲が異なります。
※3 ビットデータを指定した場合、連続した16/32/64ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32/64ビットを扱います。
※4 Sの型が.Fまたは.DFの場合、T,Cは指定できません。
※5 戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※6 デバイスZに「.DF」サフィックスは指定できません。

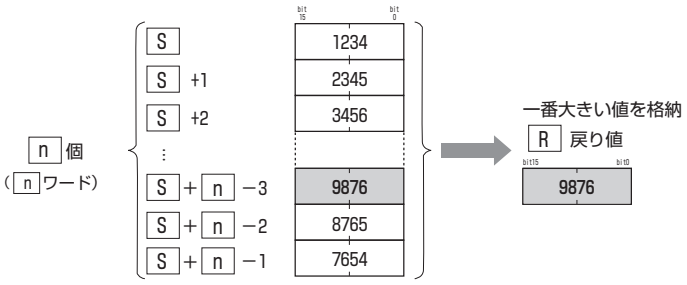
ポイント
- **引数の「.F」、「.DF」は、KV STUDIO Ver.9.0以降のバージョンでのみ使用できます。**
- **引数の「.F」、「.DF」は、KV-8000シリーズCPUユニットおよびCPU機能バージョン2.0以降のKV-7000シリーズCPUユニットでのみ指定できます。**
- **戻り値の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **戻り値の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-7500/7300　CPU機能バージョンについて」(3ページ)
「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

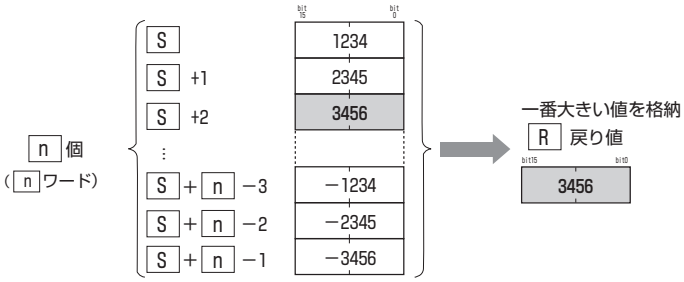
動作説明

S を先頭として、n で指定されたワード数の範囲に格納されたバイナリデータ(16ビット/32ビット/64ビット)の中で最大値を検索します。(検索対象が32ビットデータのとき、検索範囲は指定されたデータ数×2ワード、64ビットデータのとき、検索範囲は指定されたデータ数×4ワード)
検索の結果、最大値が R に格納されます。

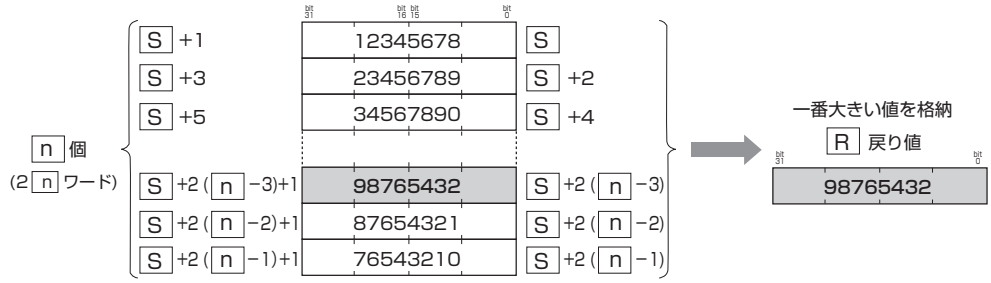
- **S :検索対象が16ビット符号無し(.U)の場合**



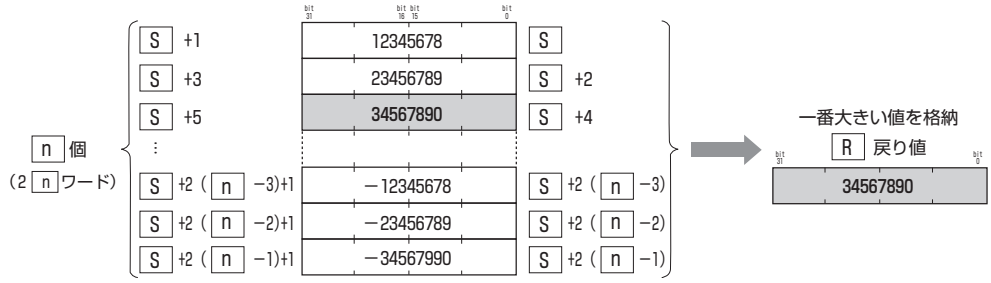
- S :検索対象が16ビット符号付き(.S)の場合



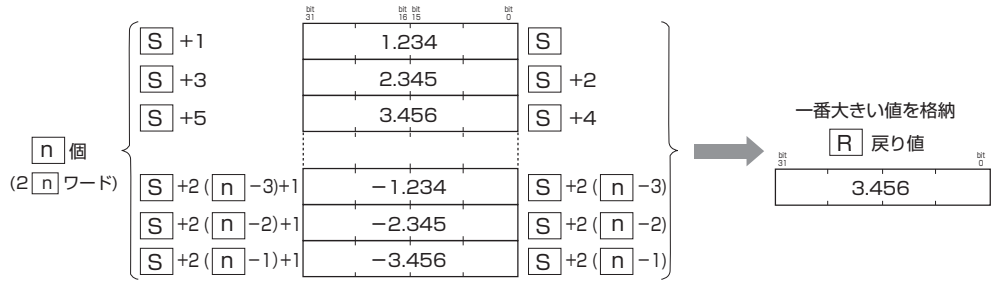
- S :検索対象が32ビット符号無し(.D)の場合



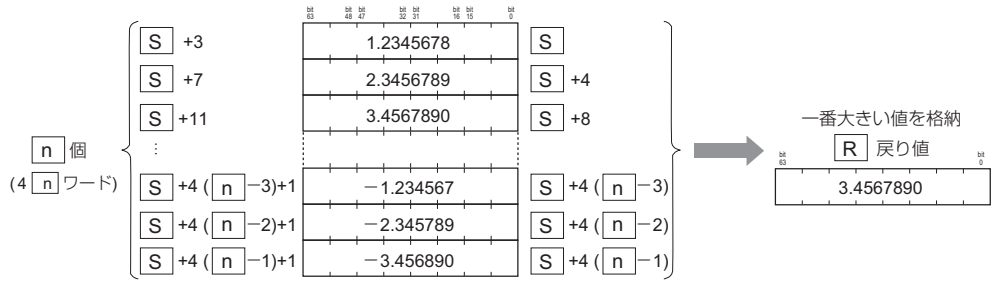
- S :検索対象が32ビット符号付き(.L)の場合



- S :検索対象が単精度浮動小数点型(.F)の場合



- S :検索対象が倍精度浮動小数点型(.DF)の場合

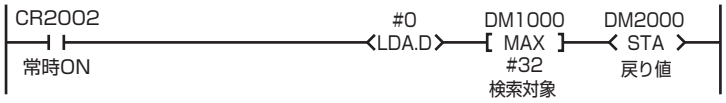


「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ処理命令」を参照してください。

●書式例

DM2000.U = **MAX** (DM1000.U, 32)



ポイント　**n の値が0の場合、R には0が格納されます。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　DM2000 = TBCD (**MAX** (DM1000.U, EM1100.U))

動作説明　DM1000を先頭とするEM1100ワードの範囲に格納されたデータの中で最大値を検索します。検索結果の最大値を4桁BCDデータに変換してDM2000に格納します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「TBCD関数」(7-80ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

MEMO

SEG
BCNT
DCNT
SER
DSER
MAX
MIN
AVG
WSUM
BSUM
CRC
ZRES
BCMP
BCMPI
RND
SORT
SORTN

MIN 指定ブロック内の最小値検索

戻り値※1 = **MIN** (検出対象, 検索範囲データ数)
最小値検索

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	検出対象※2※3※4	検索する範囲の先頭を指定します。	.U	.U	.D	.D	.F	※6 .DF	—	—	—	○	—
n	検索範囲データ数※2	検索する範囲のデータ数の格納先、または検索範囲データ数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	MIN関数は、第1引数と同じ型を持ちます。※5	.U	.S	.D	.L	.F	※6 .DF	—	—	—	○	—

※1 戻り値の省略は可能です。省略した場合、演算結果は内部レジスタに格納されます。
※2 サフィックスにより扱える数値の範囲が異なります。
※3 ビットデータを指定した場合、連続した16/32/64ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32/64ビットを扱います。
※4 Sの型が.Fまたは.DFの場合、T,Cは指定できません。
※5 戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※6 デバイスZに「.DF」サフィックスは指定できません。

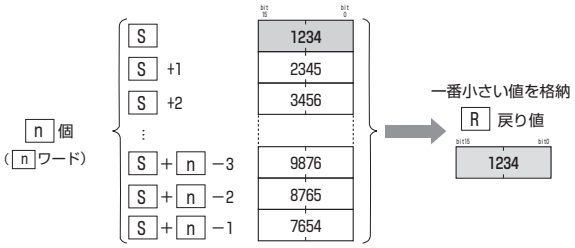
ポイント
- **引数の「.F」、「.DF」は、KV STUDIO Ver.9.0以降のバージョンでのみ使用できます。**
- **引数の「.F」、「.DF」は、KV-8000シリーズCPUユニットおよびCPU機能バージョン2.0以降のKV-7000シリーズCPUユニットでのみ指定できます。**
- **戻り値の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **戻り値の「.DF」は、KV-7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-7500/7300　CPU機能バージョンについて」(3ページ)
「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

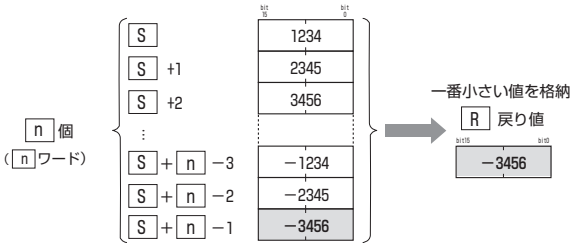
動作説明

S を先頭として、n で指定されたワード数の範囲に格納されたバイナリデータ(16ビット/32ビット/64ビット)の中で最小値を検索します。(検索対象が32ビットデータのとき、検索範囲は指定されたデータ数×2ワード、64ビットデータのとき、検索範囲は指定されたデータ数×4ワード)
検索の結果、最小値が R に格納されます。

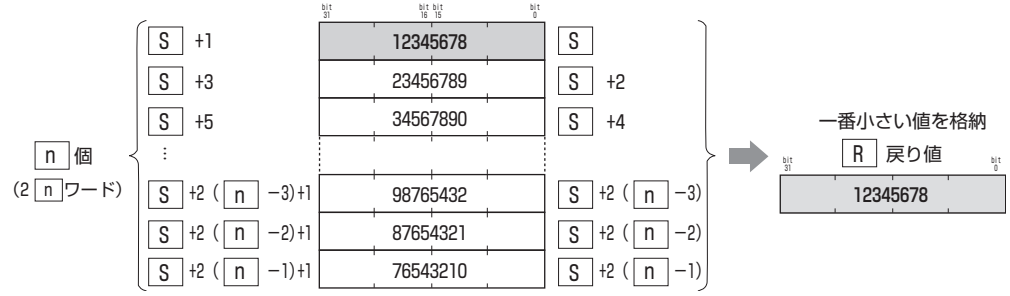
- **S :検索対象が16ビット符号無し(.U)の場合**



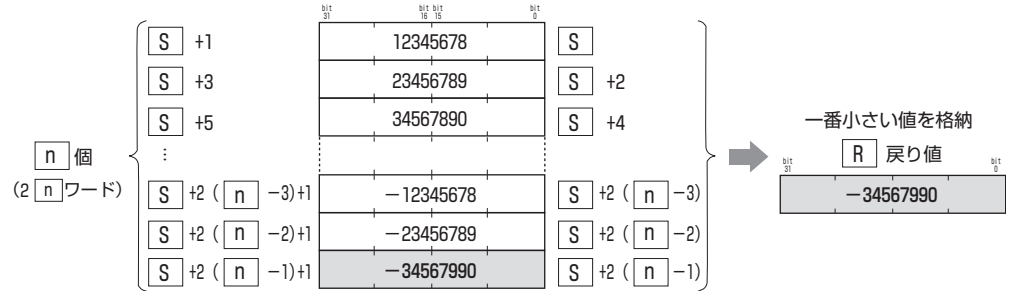
- S :検索対象が16ビット符号付き(.S)の場合



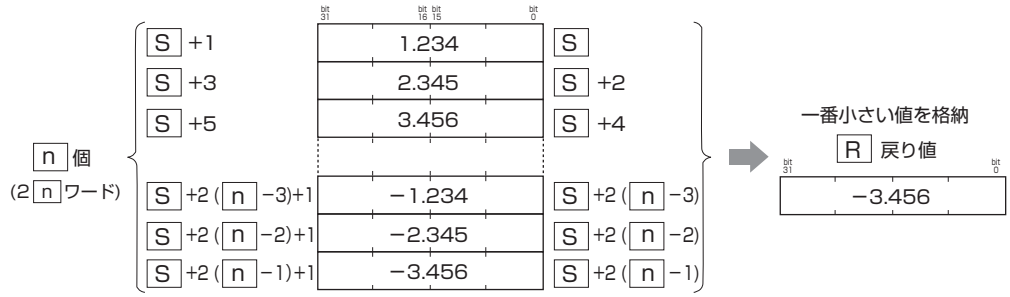
- S :検索対象が32ビット符号無し(.D)の場合



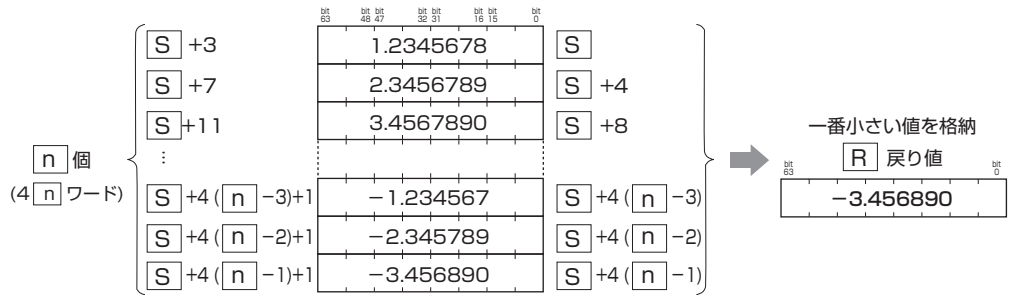
- S :検索対象が32ビット符号付き(.L)の場合



- S :検索対象が単精度浮動小数点型(.F)の場合



- S :検索対象が倍精度浮動小数点型(.DF)の場合

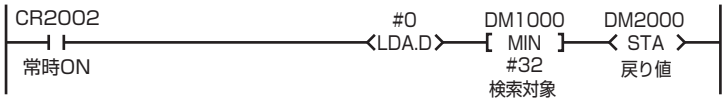


「演算フラグについて(CR2009〜CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ処理命令」を参照してください。

●書式例

DM2000.U = **MIN** (DM1000.U, 32)



ポイント　**n の値が0の場合、R には0が格納されます。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　DM2000.S = NEG (**MIN** (DM1000.S, EM1100))

動作説明　DM1000を先頭とするEM1100ワードの範囲に格納されたデータの中で最小値を検索します。検索結果の最小値を符号反転してDM2000.Sに格納します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「NEG関数」(7-32ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

MEMO

SEG
BCNT
DCNT
SER
DSER
MAX
MIN
AVG
WSUM
BSUM
CRC
ZRES
BCMP
BCMPI
RND
SORT
SORTN

AVG 指定ブロックの平均値

戻り値※1 = **AVG** (演算対象, 演算範囲データ数)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	演算対象※2※3※4	平均する範囲の先頭を指定します。	.U	.U	.D	.D	.F	※6 .DF	—	—	—	○	—
n	演算範囲データ数※2	平均する範囲のデータ数、または格納先を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	AVG関数は、第1引数と同じ型を持ちます。※5	.U	.S	.D	.L	.F	※6 .DF	—	—	—	○	—

※1 戻り値の省略は可能です。省略した場合、演算結果は内部レジスタに格納されます。
※2 サフィックスにより扱える数値の範囲が異なります。
※3 ビットデータを指定した場合、連続した16/32/64ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32/64ビットを扱います。
※4 Sの型が.Fまたは.DFの場合、T,Cは指定できません。
※5 戻り値に第1引数と異なる型を指定した場合、変換時に自動的に型変換がおこなわれます。
※6 デバイスZに「.DF」サフィックスは指定できません。

ポイント

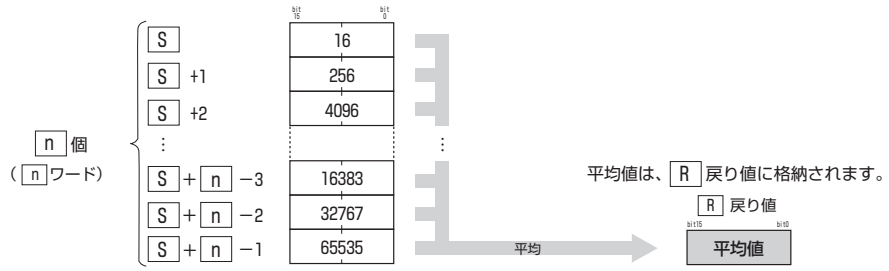
- **引数の「.F」、「.DF」は、KV STUDIO Ver.9.0以降のバージョンでのみ使用できます。**
- **引数の「.F」、「.DF」は、KV-8000シリーズCPUユニットおよびCPU機能バージョン2.0以降のKV-7000シリーズCPUユニットでのみ指定できます。**
- **戻り値の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **戻り値の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-7500/7300　CPU機能バージョンについて」(3ページ)
「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

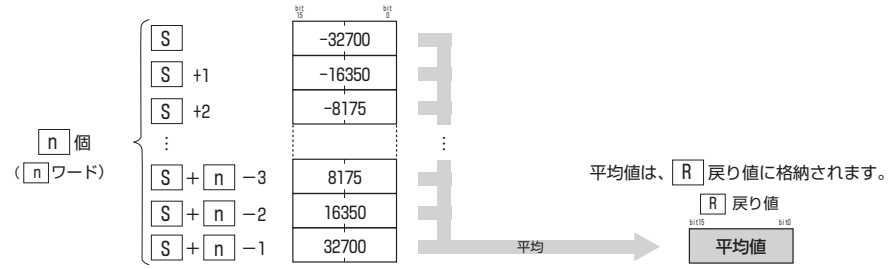
動作説明

S を先頭として、n で指定されたワード数の範囲に格納されたバイナリデータ(16ビット/32ビット(符号無し/符号付き)/64ビット)全ての平均値を求めます。(演算対象が32ビットデータのとき、演算範囲は指定されたデータ数×2ワード、64ビットデータのとき、演算範囲は指定されたデータ数×4ワード)
演算の結果、平均値を R に格納します。

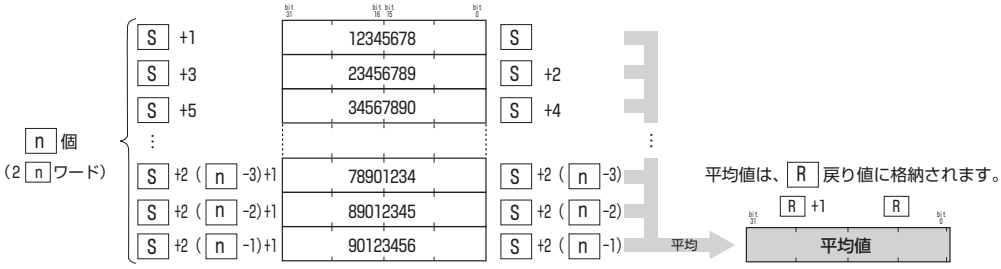
- **S :演算対象が16ビット符号無し(.U)の場合**



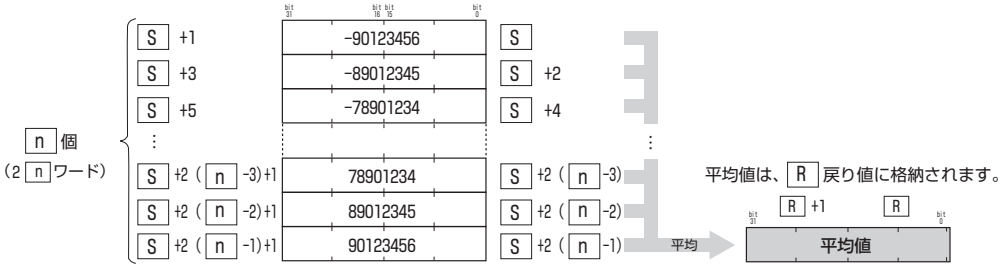
- S :演算対象が16ビット符号付き(.S)の場合



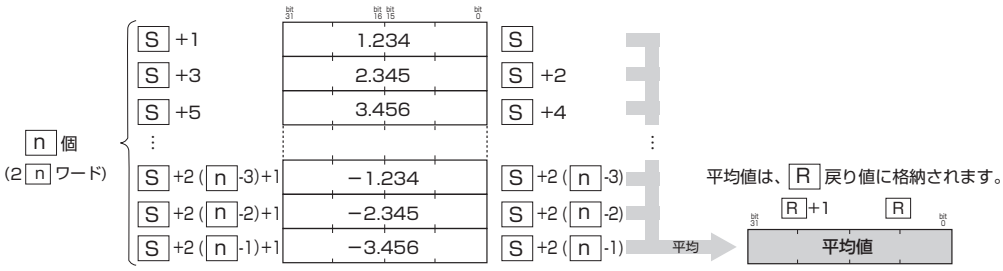
- S :演算対象が32ビット符号無し(.D)の場合



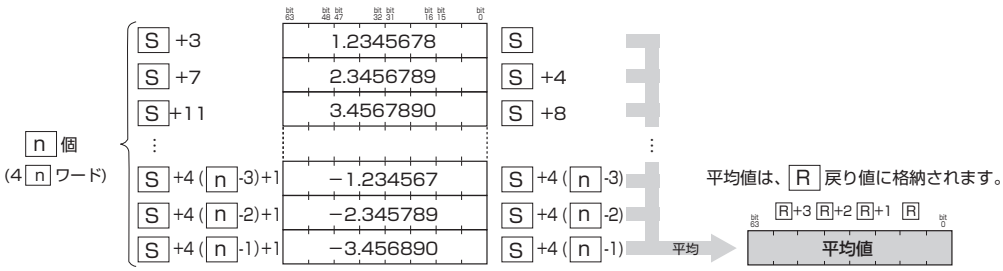
- S :演算対象が32ビット符号付き(.L)の場合



- S :検索対象が単精度浮動小数点型(.F)の場合



- S :検索対象が倍精度浮動小数点型(.DF)の場合

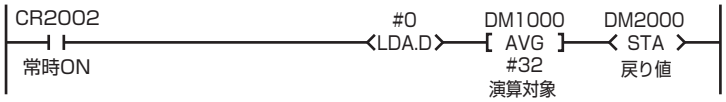


「演算フラグについて(CR2009〜CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ処理命令」を参照してください。

●書式例

DM2000.U = **AVG** (DM1000.U, 32)



ポイント　**n の値が0の場合、R には0が格納されます。**

MEMO

SEG
BCNT
DCNT
SER
DSER
MAX
MIN
AVG
WSUM
BSUM
CRC
ZRES
BCMP
BCMPI
RND
SORT
SORTN

WSUM 指定ブロックの合計値

戻り値※1 = **WSUM** (演算対象, 演算範囲データ数)
ワードサム

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	演算対象※2※3※4	合計する範囲の先頭を指定します。	.U	.U	.D	.D	.F	※6 .DF	—	—	—	○	—
n	演算範囲データ数※2	合計する範囲のデータ数の格納先、または合計範囲データを指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	WSUM関数は、.Lの型を持ちます。※5	.U	.S	.D	.L	.F	※6 .DF	—	—	—	○	—

※1 戻り値の省略は可能です。省略した場合、演算結果は内部レジスタに格納されます。
※2 サフィックスにより扱える数値の範囲が異なります。
※3 ビットデータを指定した場合、連続した16/32/64ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32/64ビットを扱います。
※4 Sの型が.Fまたは.DFの場合、T,Cは指定できません。
※5 戻り値に.L以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※6 デバイスZに「.DF」サフィックスは指定できません。

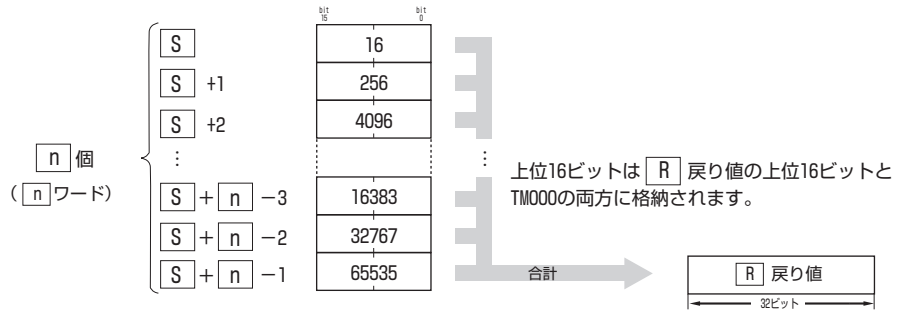
ポイント
- **引数の「.F」、「.DF」は、KV STUDIO Ver.9.0以降のバージョンでのみ使用できます。**
- **引数の「.F」、「.DF」は、KV-8000シリーズCPUユニットおよびCPU機能バージョン2.0以降のKV-7000シリーズCPUユニットでのみ指定できます。**
- **戻り値の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **戻り値の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-7500/7300 CPU機能バージョンについて」(3ページ)
「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

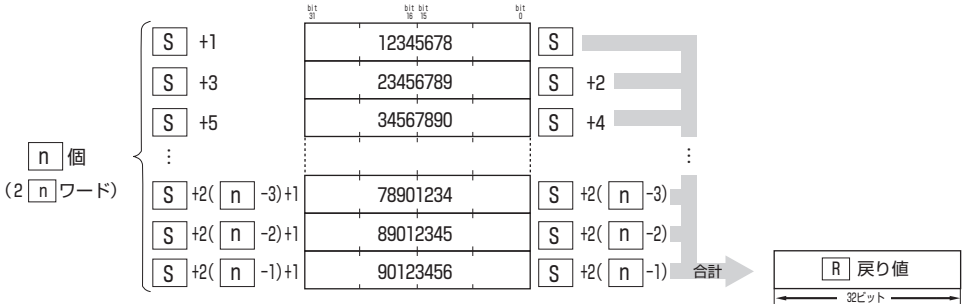
動作説明

S を先頭として、n で指定されたワード数の範囲に格納されたバイナリデータ(16ビット/32ビット(符号無し)/64ビット)全ての合計値を求めます。(演算対象が32ビットデータのとき、演算範囲は指定されたデータ数×2ワード、64ビットデータのとき、演算範囲は指定されたデータ数×4ワード)演算の結果、合計値を R に格納します。

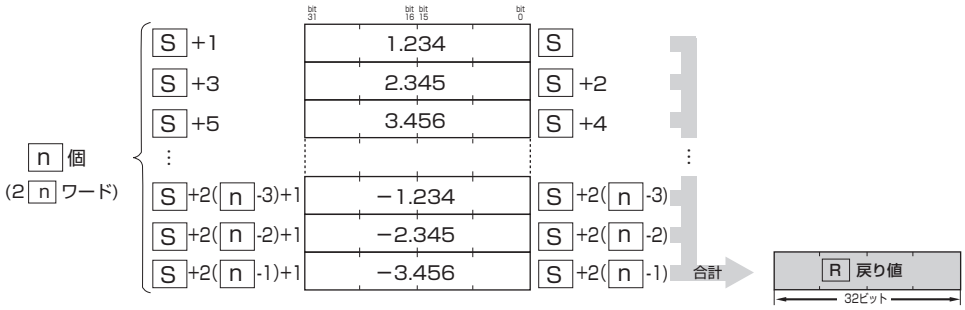
- **S :演算対象が16ビット(.U/.S)の場合**



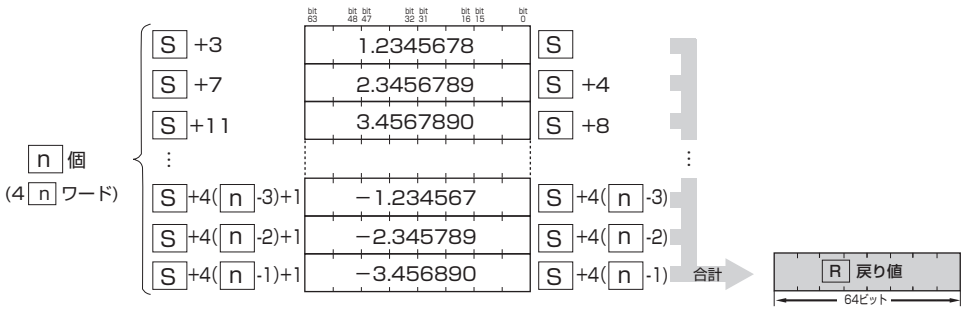
- S :演算対象が32ビット(.D/.L)の場合



- S :検索対象が単精度浮動小数点型(.F)の場合



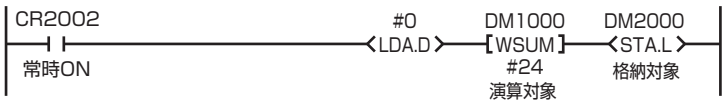
- S :検索対象が倍精度浮動小数点型(.DF)の場合



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ処理命令」を参照してください。

●書式例

DM2000.L = **WSUM** (DM1000.U, 24)



参考　演算結果が符号付き32ビットデータの範囲(－2147483648～+2147483647)を超える場合、戻り値には下位32ビットデータが格納され、上位32ビットデータはTM001、TM000に格納されます。

ポイント　**n の値が0の場合、R には0が格納されます。**

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述　DM2000.F = FLOAT (**WSUM** (DM1000.U, EM1100.U))

動作説明　DM1000.Uを先頭とするEM1100.Uに格納されている値の範囲に格納されたデータ全ての合計値を求めます。演算結果の合計値を単精度浮動小数点型実数データに変換してDM2000.Fに格納します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「FLOAT関数」(7-114ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ワンポイント

WSUM関数で演算した結果が32ビット符号付き範囲を超える場合、戻り値には下位32ビットのデータが格納され、上位32ビットのデータはテンポラリデータメモリTM0/TM1に格納されます。
WSUM関数は上位32ビット(TM0/TM1)のデータを利用して64ビットデータとして扱います。

スクリプト記述　EM2000.L = **WSUM** (DM1000, 1024)
EM2002.L = TM0000.L

DM1000を先頭とする1024ワードの範囲に格納された16ビットバイナリデータ(符号無し)全てを合計します。合計値の下位32ビットをEM2000.Lに格納し、直後に上位16ビット(TM0.L)をEM2002.Lに格納して、EM2000.L～EM2002.Lの64ビットバイナリデータ(符号付き)として扱います。

参考　TM0/TM1のテンポラリデータは、他の演算式でも使用しますので64ビットデータとして扱う場合はWSUM関数を記述した直後に別のデバイス/変数に格納しておく必要があります。

MEMO

SEG
BCNT
DCNT
SER
DSER
MAX
MIN
AVG
WSUM
BSUM
CRC
ZRES
BCMP
BCMPI
RND
SORT
SORTN

BSUM 指定ブロックのバイト単位の合計値

戻り値※1 = **BSUM** (演算対象, 演算範囲データ数)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	演算対象※2※3※4	合計値を求める範囲の先頭を指定します。	.U	.U	.U	.U	—	—	—	.U	—	○	—
n	演算範囲データ数※2	合計値を求めるデータ数(バイト単位)を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	BSUM関数は、.Dの型を持ちます。※5	.U	.S	.D	.L	.F	※6 .DF	—	—	—	○	—

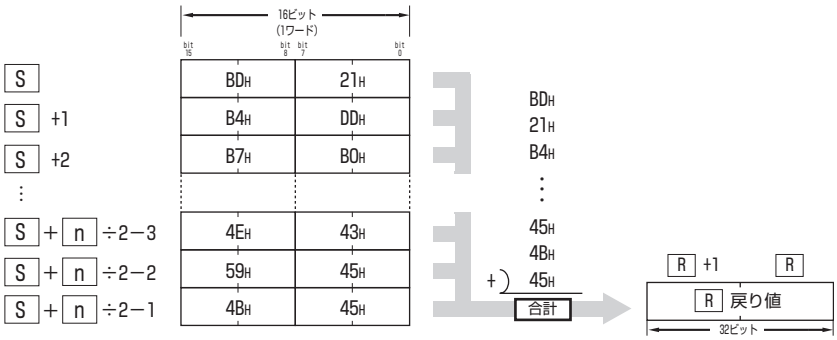
※1 戻り値の省略は可能です。省略した場合、演算結果は内部レジスタに格納されます。
※2 サフィックスにより扱える数値の範囲が異なります。
※3 ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。
※4 T, C, CTC, CTH, Zは使用できません。
※5 戻り値に.D以外の型を指定した場合、変換時に自動的に型変換がおこなわれます
※6 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

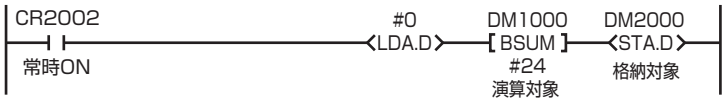
S から始まる指定範囲内に格納されているバイナリデータ上位･下位それぞれ8ビットの合計値を求め R に格納します。n が奇数の場合は最後のワードデータの上位8ビットまでが計算されます。計算は上位→下位の順におこない n が奇数のときは最後のワードデータの下位8ビットは無視されます。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ処理命令」を参照してください。

●書式例

DM2000.D = **BSUM** (DM1000, 24)



参 考　演算結果が符号付き32ビットデータの範囲を超える場合、戻り値には下位32ビットデータが格納され、上位32ビットデータはTM1、TM0に格納されます。

ポイント　**次のいずれかの条件が成立した場合、R には0が格納されます。**

- S にタイマ・カウンタを指定した場合。
- S にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。
- n の値が0の場合。

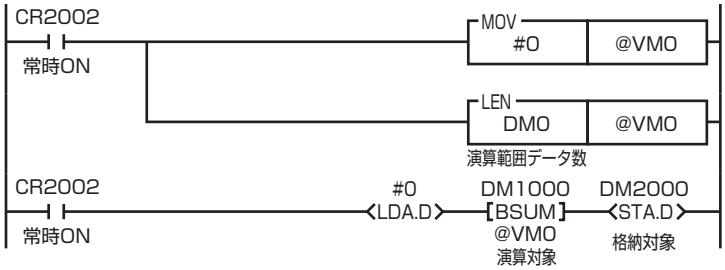
サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述　DM2000.D = **BSUM** (DM1000.T, LEN (DM0000.T))

動作説明　DM1000.Tに格納されているアスキー文字列をバイト単位で合計し、結果を〔DM2000,DM2001〕に格納します。

ラダー変換



参 考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「LEN関数」(7-182ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

CRC 指定ブロック内のCRC値算出

戻り値※1 = **CRC** (初期値, 演算方式, 演算順序, ビット演算順序反転,
データブロック先頭, データ個数)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	初期値※5	動作モードを指定します。※2	.U	.U	.U	.U	—	—	—	—	○	○	○
S2	演算方式※5		.U	.U	.U	.U	—	—	—	—	○	○	○
S3	演算順序※5		.U	.U	.U	.U	—	—	—	—	○	○	○
S4	ビット順序反転※5		.U	.U	.U	.U	—	—	—	—	○	○	○
S5	データブロック先頭※3※5	算出したいデータブロックの先頭を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
n	データ個数※4	データの個数(1～65535)を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	CRC関数は、.Uの型を持ちます。※6	.U	.S	.D	.L	.F	※7 .DF	—	—	—	○	—

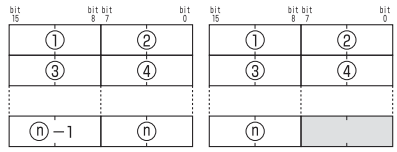
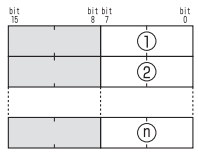
※1 戻り値の省略は可能です。省略した場合、演算結果は内部レジスタに格納されます。
※2 指定した対象または式の値が0、1以外の場合、1として扱われます。
※3 ビットデータを指定するときは、チャンネルの先頭を指定します。
※4 S3 が「0」のときは1ワード(16bit)単位で個数を指定します。
S3 が「1(0以外)」のときは1バイト(8bit)単位で個数を指定します。
※5 T, C, CTC, CTH, Zは使用できません。
※6 戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※7 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

S1 ～S4 で指定した動作モードで、S5 から始まる n 個のデータのCRC値を算出し、R に格納します。
動作モードは、次のように指定します。

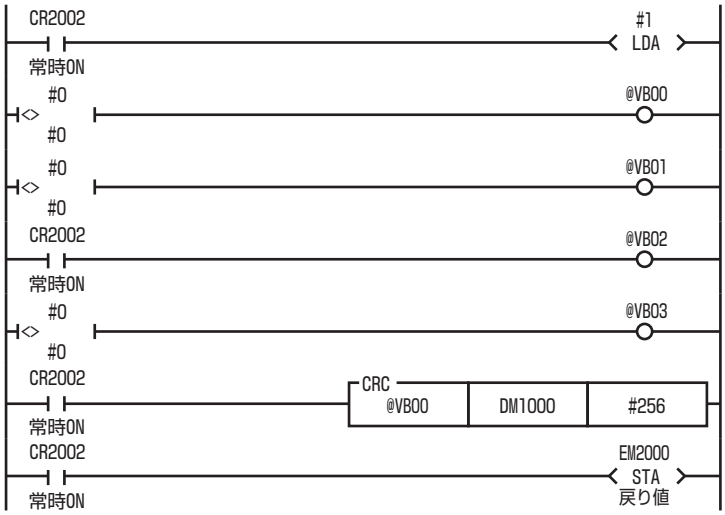
項 目		CRC演算方式	
		1を指定した時	0を指定した時
S1	CRC初期値	16bit内部レジスタにセットされている値をCRC初期値として使用します。	CRC初期値を0x0000固定とします。
S2	CRC演算方式	CRC-ITU-T(CRC-CCITT)	CRC-16(CRC-ANSI)
S3	演算順序	上位8bit→下位8bitの順に演算します。 n : 偶数　n : 奇数 	下位8bitのみ演算します。 (上位8bitは無視) 
S4	演算データビット順序反転	演算するデータのビットの並び(順序)をバイト単位で反転し、CRC値を計算します。	ビット位置は反転しません。

S4 **演算データビット順序反転**に関する例をバイト単位で次に示します。

演算データ ビット順序反転	CRC-ITU-T （S2 =1）	CRC-16 （S2 =0）
反転しない S4 :「0」	bit 7 bit 0 演算順序 →	bit 7 bit 0 ← 演算順序
反転する S4 :「1」(0以外)	bit 7 bit 0 ← 演算順序	bit 7 bit 0 演算順序 →

●書式例

EM2000.U = **CRC** (1, 0, 0, 0, DM1000.U, 256)



「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ処理命令」CRC命令を参照してください。

ポイント

次のいずれかの条件が成立した場合、R には S1 の値が格納されます。

- S1 にワードデータまたは、タイマ･カウンタを指定した場合。
- S2 に間接指定でタイマ･カウンタまたは、ビットデータを指定したとき、チャンネルの先頭以外を指定した場合。
- n の値が0の場合。

SEG
BCNT
DCNT
SER
DSER
MAX
MIN
AVG
WSUM
BSUM
CRC
ZRES
BCMP
BCMPI
RND
SORT
SORTN

ZRES 指定ブロックのリセット(0を格納)

ゾーンリセット
ZRES ([実行条件,]※1 先頭, 終端)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D1 先頭	リセット(0を格納)する範囲の先頭を指定します。	.U	.U	.U	.U	.U	※2 .U	.B	.T	—	○	—
D2 終端	リセット(0を格納)する範囲の終端を指定します	D1と同じ型として扱われます。								—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

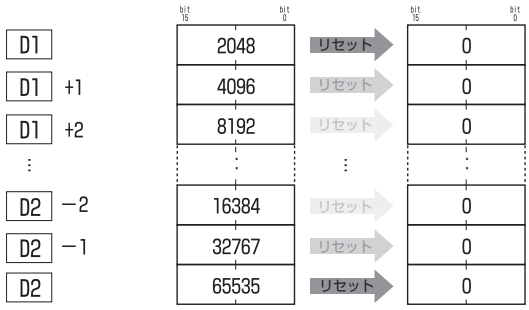
※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

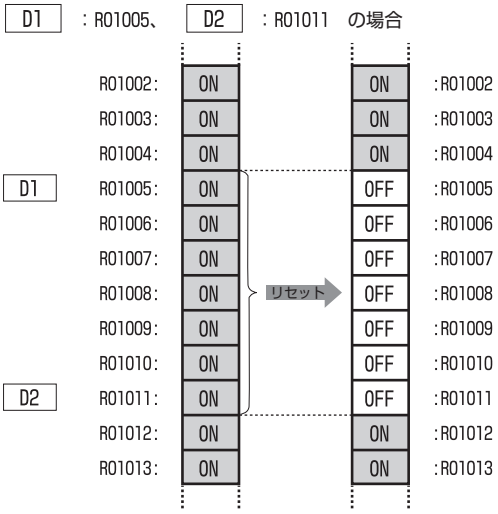
実行条件が成り立つ(TRUE)とき、D1 と D2 で指定された範囲を、すべてリセット(0を格納)します。

- **ワードデータを指定した場合**



※サフィックスに.D .L .F .DFを指定してもすべて.Uとして扱われるので、終端ワードは指定した1ワードのみ0が格納されます。

• ビットデータを指定した場合



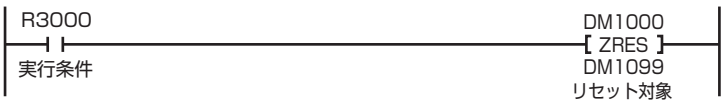
参 考　ワードデータが指定された場合はワード単位で0に、ビットデータが指定された場合はビット単位でOFFになります。

「演算フラグについて（CR2009～CR2012）」（1-18ページ）

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ処理命令」を参照してください。

●書式例

ZRES (R3000, DM1000.U, DM1099.U)



ポイント　**D1 と D2 に指定されたデバイスの種類が異なる場合、ZRES関数を実行しません。**

サンプルプログラム

●ビットデータを指定する場合

スクリプト記述　**ZRES** (R3000, MR1008, MR1207)

動作説明　R3000がONのとき、MR1008からMR1207の32ビットをすべてリセット(OFF)します。

●インデックスレジスタを使用する場合

スクリプト記述　**ZRES** (DM2000:Z01, DM2000:Z02)

動作説明　毎スキャン(実行条件無し)、DM2000:Z01からDM2000:Z02の範囲のワードをすべてリセット(0クリア)します。

参 考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」（2-10ページ）を参照してください。

BCMP 指定ブロック間データ比較

戻り値※1 = **BCMP** (比較元, 比較先,結果格納対象, 比較データ数)
ブロック間データ比較

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 比較元※2	比較元のデータブロックの先頭を指定します。	.U	.U	.D	.D	.D	—	—	—	—	○	—
S2 比較先※2	比較先のデータブロックの先頭を指定します。	S1と同じ型として扱われます。					—	—	—	—	○	—
S3 結果格納対象※2※3	比較結果を格納する先頭を指定します。	—	—	—	—	—	—	.B	—	—	○	—
n 比較データ数	比較するブロックのデータ数の格納先、または比較データ数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	全データが一致していればONし、それ以外はOFFします。	—	—	—	—	—	—	.B	—	—	○	—

※1 戻り値の省略は可能です。省略した場合、ビット情報はワークデバイス / ワークリレーに格納されます。

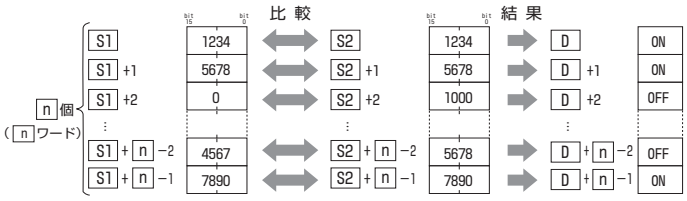
※2 ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。

※3 ビットデータを指定した場合、チャンネルをまたがっても正しく動作します。
D にワードデータを指定した場合、n 個の16ビットワードデータの最下位ビットが結果の格納先になります。最下位ビット以外は変化しません。

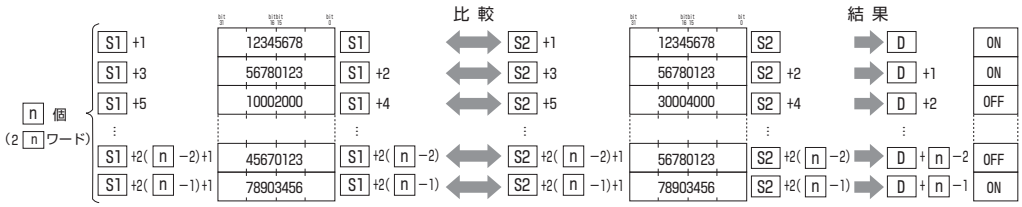
動作説明

実行条件が成り立つ(TRUE)とき、S1 を先頭とする、n で指定された個数のブロックデータと、S2 を先頭とする、n 分のブロックデータをそれぞれ比較します。
比較結果は、一致した場合はON、一致しなかった場合はOFFとして、D を先頭とする、比較データ数分のビットデータへ格納されます。
D がワードデータのとき、比較結果の格納は16ビットワードデータ単位での扱いとなり、各ワードデータの最下位ビットのみが対象となります。(最下位ビット以外は変化しません)
比較の結果、全データが一致していれば R をONし、そうでなければOFFします。

- **S1:比較元が16ビット(.U/.S)の場合**



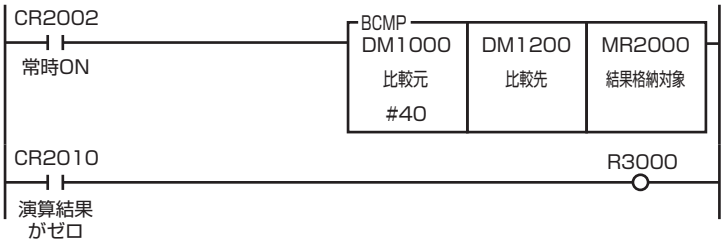
- S1 :比較元が32ビット(.D/.L)の場合



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ処理命令」を参照してください。

●書式例

R3000 = **BCMP** (DM1000.U, DM1200.U, MR2000, 40)



ポイント　n の値が0の場合、R はONします。

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述　MR2000 = **BCMP** (DM1000.L, EM2000.L, MR3000, ANDA (EM3000, $3F))

動作説明　DM1000.Lを先頭とする32ビットバイナリデータ(符号付き)ブロックと、EM2000(.L)を先頭とする32ビットバイナリデータ(符号付き)ブロックのうち、EM3000の下位6ビット(抽出)で示される比較データ数分(比較データ数×2ワード)のデータを、それぞれ比較します。
比較結果を、MR3000を先頭とする16ビットデータに格納します。
比較の結果、全データが一致していればMR2000をONし、そうでなければOFFにします。

参 考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「ANDA関数」(7-22ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

SEG
BCNT
DCNT
SER
DSER
MAX
MIN
AVG
WSUM
BSUM
CRC
ZRES
BCMP
BCMPI
RND
SORT
SORTN

BCMPI ブロックと値のデータ比較

戻り値※1 = **BCMPI** (比較元, 比較値,結果格納対象, 比較データ数)
ブロック間データ比較

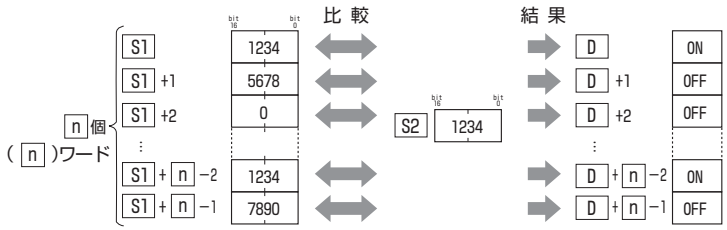
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 比較元※2	比較するデータブロックの先頭を指定します。	.U	.U	.D	.D	.D	—	—	—	—	○	○
S2 比較値	比較する値を格納した対象、または比較を指定します。	S1と同じ型として扱われます。					—	—	—	—	○	—
D 結果格納対象※3	比較結果を格納する先頭を指定します。	—	—	—	—	—	—	.B	—	—	○	—
n 比較データ数	比較するブロックのデータ数の格納先、または比較データ数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	全データが一致していればONし、それ以外はOFFします。	—	—	—	—	—	—	.B	—	—	○	—

※1 戻り値の省略は可能です。省略した場合、演算結果はワークデバイス/ワークリレーに格納されます。
※2 ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。
※3 ビットデータを指定した場合、チャンネルをまたがっても正しく動作します。
D にワードデータを指定した場合、n 個の16ビットワードデータの最下位ビットが結果の格納先になります。最下位ビット以外は変化しません。

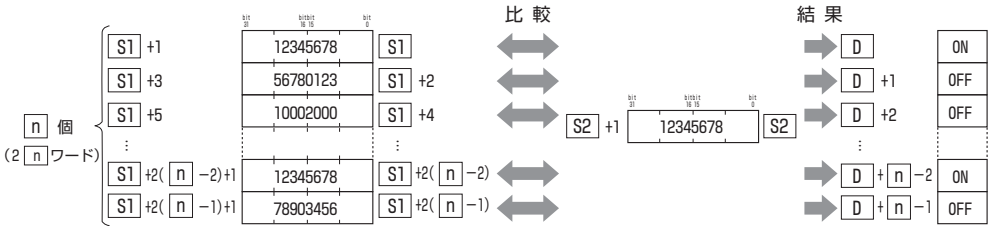
動作説明

実行条件が成り立つ(TRUE)とき、S1 を先頭とする、n で指定された個数のブロックデータを、S2 で指定されたデータとそれぞれ比較します。
比較結果は、一致した場合はON、一致しなかった場合はOFFとして、D を先頭とする、n 分の対象へ格納されます。
D がワードデータのとき、比較結果の格納は16ビットワードデータ単位での扱いとなり、各ワードデータの最下位ビットのみが対象となります。(最下位ビット以外は変化しません)
比較の結果、全データが一致していれば R をONし、そうでなければOFFします。

- **S1 :比較元が16ビット(.U/.S)の場合**



- S1 :比較元が32ビット(.D/.L)の場合

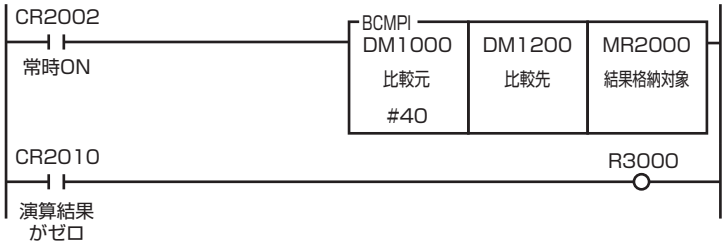


「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ処理命令」を参照してください。

●書式例

R3000 = **BCMPI** (DM1000.U, DM1200, MR2000, 40)



ポイント　**n の値が0の場合、R はONします。**

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述　MR2000 = **BCMPI** (DM1000.D, RRNCA(MR1000.D, 16), MR3000, 24)

動作説明　DM1000.Dを先頭とする24個の32ビットバイナリデータ(符号無し)ブロックを、MR1000.Dに格納されている32ビットバイナリデータ(符号無し)を16ビット右にキャリーを含まずローテート(上位/下位のワード入れ替え)した結果の値とそれぞれ比較します。
比較結果を、MR3000を先頭とする16ビットデータに格納します。
比較の結果、全データが一致していればMR2000をONし、そうでなければOFFにします。

参 考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「RRNCA関数」(7-50ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

SEG
BCNT
DCNT
SER
DSER
MAX
MIN
AVG
WSUM
BSUM
CRC
ZRES
BCMP
BCMPI
RND
SORT
SORTN

RND 乱数発生

戻り値※1 = **RND** (乱数の種)
ランダム

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 乱数の種※2	乱数発生の元となる、演算対象または演算対象データを指定しますす。	.U	.U	.U	.U	—	—	—	—	—	○	○
R 戻り値	RND関数は、.Uの型を持ちます。※3	.U	.S	.D	.L	.F	※4 .DF	—	—	—	○	—

※1 戻り値の省略は可能です。省略した場合、演算結果はワークメモリに格納されます。
※2 乱数の種(変換対象)にビットデータのチャンネルの先頭以外(R1002, R1005等)を指定した場合でも、次のチャンネルにまたがって正常に機能します。
※3 戻り値に.U以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※4 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

S (演算対象)から乱数を発生させ、乱数を R に格納します。
発生する乱数は1～65535の疑似乱数です。S =0のときは、R に1を格納します。
「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「算術/比較命令」を参照してください。

ポイント
- **乱数発生のアルゴリズムには「M系列」を用いています。**
- **RND関数を使って生成される値には周期性がある(例えば、1の後には必ず2が生成される)ため、S 乱数の種の値が変化しない場合、生成される乱数の値も変化しません。**

●書式例

DM1000.U = **RND** (DM1000.U)

MEMO

SEG
BCNT
DCNT
SER
DSER
MAX
MIN
AVG
WSUM
BSUM
CRC
ZRES
BCMP
BCMPI
RND
SORT
SORTN

SORT バイナリデータソート

SORT (ソート) **([**実行条件**,]** ※1 ソート順**,** データテーブル先頭**,** ソートデータ数**)**

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	ソート順	ソート順序を指定します。(0:昇順、1:降順)	.U	.U	.U	.U	—	—	—	—	○	○	○
D	データテーブル先頭※2	検索するバイト列の先頭を指定します。	.U	.S	.D	.L	—	—	—	—	—	○	—
n	ソートデータ数※3	ソートするデータ数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。実行条件を省略した場合、常時(毎スキャン)実行します。

※2 ビットデータを指定した場合、D の型が.U/.Sの時 n チャンネル、.D/.Lの時2× n チャンネルを占有します。
ワードデータを指定した場合、D の型が.U/.Sの時 n ワード、.D/.Lの時2× n ワードを占有します。

※3 ビットデータを指定した場合、連続する16ビットを占有します。ワードデータを指定した場合、1ワードを占有します。

ポイント

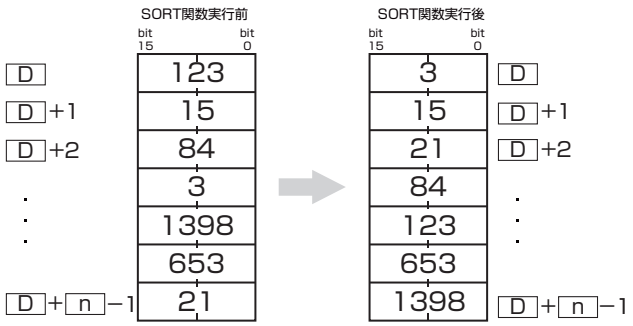
- **SORT関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000 シリーズ CPU ユニット、CPU 機能バージョン 2.0 以降の KV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

D で指定したデータテーブルから n のバイナリデータを、S で指定したソート順序でソートします。

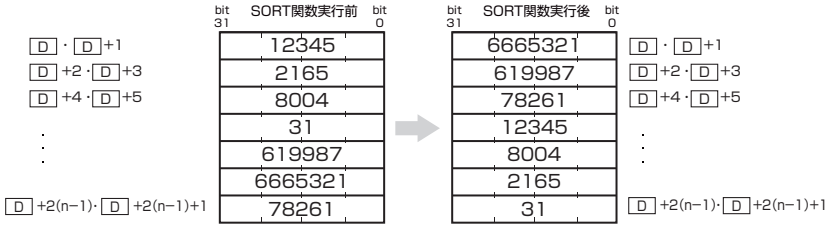
- **D :データテーブル先頭が16ビット(.U/.S)の場合**

S =0(昇順でソート)の場合



- [D]：データテーブル先頭が32ビット(.D/.L)の場合

[S]＝1（降順でソート）の場合



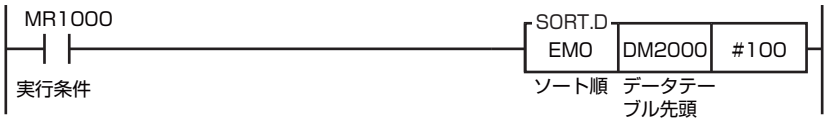
ポイント

SORT関数は、関数実行時に全データをソート処理します。そのため、[n]に大きな値を指定すると、スキャンタイムが大きく伸びることがあります。
スキャンタイムへの影響を調整する必要がある場合は、SORTN関数で複数スキャンに分割してソートをおこなってください。
「SORTN関数」（8-44ページ）

「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「データ処理命令」を参照してください。

●書式例

SORT (MR1000, EM0, DM2000.D, 100)



SEG
BCNT
DCNT
SER
DSER
MAX
MIN
AVG
WSUM
BSUM
CRC
ZRES
BCMP
BCMPI
RND
SORT
SORTN

SORTN　バイナリデータ分割ソート

戻り値※1 = **SORTN** (ソート制御フラグ, ソート順, 比較回数,
データテーブル先頭, ソートデータ数)
（分割ソート）

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	ソート制御フラグ	ソート処理を指定します。	—	—	—	—	—	—	.B	—	—	○	—
S2	ソート順	ソート順序を指定します。(0：昇順、1：降順)	.U	.U	.U	.U	—	—	—	—	○	○	○
S3	比較回数※3	1スキャンでの比較回数を指定します。(1～65535)	.U	.U	.U	.U	—	—	—	—	○	○	○
D	データテーブル先頭※2	検索するバイト列の先頭を指定します。	.U	.S	.D	.L	—	—	—	—	—	○	—
n	ソートデータ数※3	ソートするデータ数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	ソート中はOFF、ソート完了時ONになります。	—	—	—	—	—	—	.B	—	—	○	—

※1　戻り値は省略できます。省略した場合、検索結果はワークデバイスに格納されます。
※2　ビットデータを指定した場合、D の型が.U/.Sの時 n チャンネル、.D/.Lの時2× n チャンネルを占有します。
ワードデータを指定した場合、D の型が.U/.Sの時 n ワード、.D/.Lの時2× n ワードを占有します。
※3　ビットデータを指定した場合、連続する16ビットを占有します。ワードデータを指定した場合、1ワードを占有します。

ポイント
- **SORTN関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**
「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

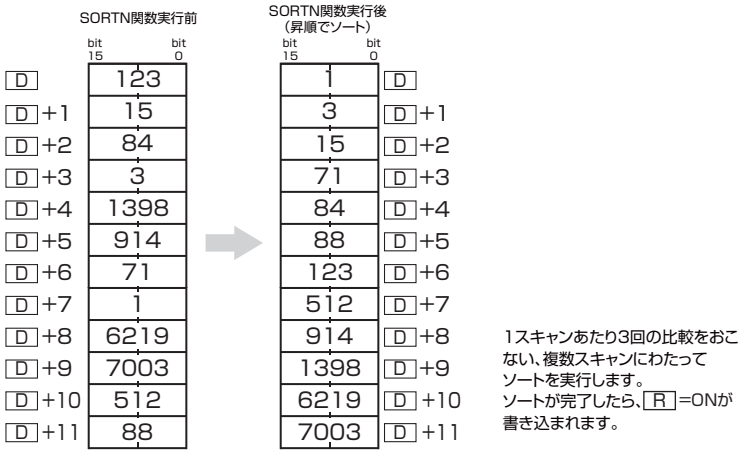
動作説明

D で指定したデータテーブルから n のバイナリデータを、S2 で指定したソート順序でソートします。
1スキャンあたりの比較回数は、S3 **比較回数**で指定します。

例

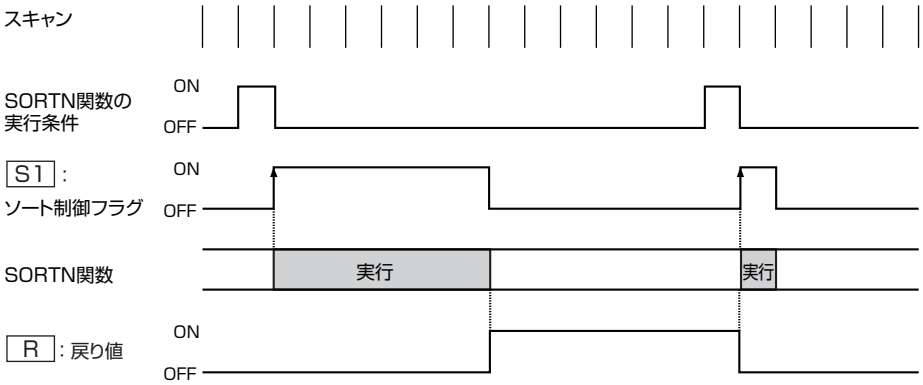
D :データテーブル先頭が16ビット(.U/.S)の場合

S2 =0、S3 =3、n =12の場合



1スキャンあたり3回の比較をおこない、複数スキャンにわたってソートを実行します。
ソートが完了したら、R =ONが書き込まれます。

SORTN関数の実行タイミングは以下のとおりです。



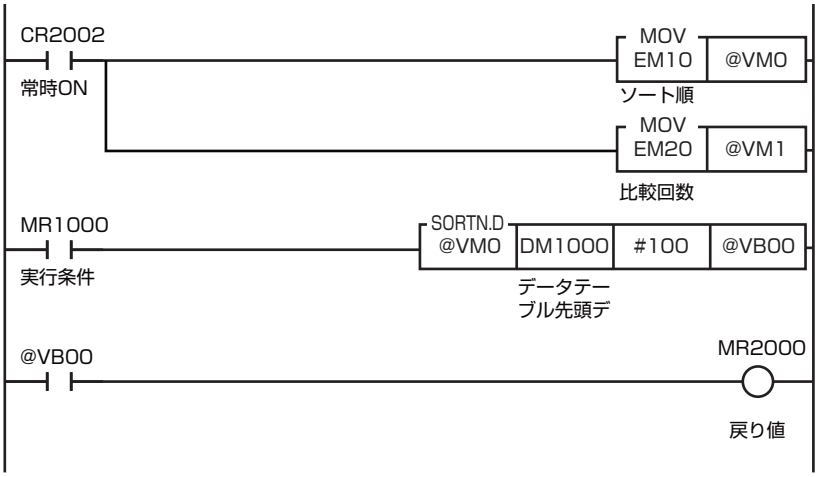
「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「データ処理命令」を参照してください。

SEG
BCNT
DCNT
SER
DSER
MAX
MIN
AVG
WSUM
BSUM
CRC
ZRES
BCMP
BCMPI
RND
SORT
SORTN

●書式例

MR2000 = **SORTN** (MR1000, EM10, EM20, DM1000.D, 100)



MEMO

SEG
BCNT
DCNT
SER
DSER
MAX
MIN
AVG
WSUM
BSUM
CRC
ZRES
BCMP
BCMPI
RND
SORT
SORTN

FIFOW FIFOテーブルへ書き込み

FIFO書き込み

FIFOW ([実行条件,]※1 データ格納対象, データテーブル先頭)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	データ格納対象※2	FIFOデータブロックに書き込むデータの格納先を指定します。	D と同じ型として扱われます。					—	—	—	—	○	○
D	データテーブル先頭	FIFOデータテーブルの先頭を指定します。	.U	.U	.D	.D	.D	—	—	—	—	○	—
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

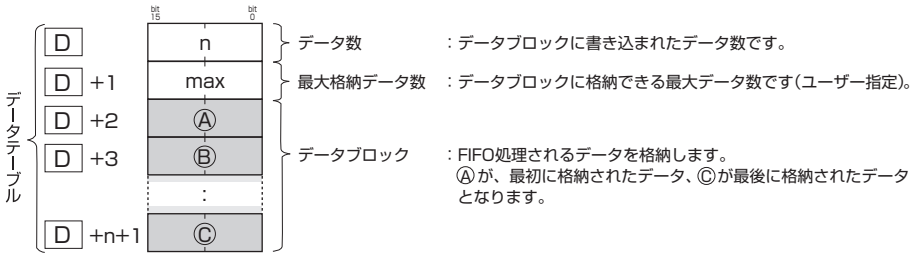
※2 ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。

動作説明

実行条件が成り立つ(TRUE)とき、S に格納されたデータ(16/32ビット)を、D で指定されたデータブロックの最後に書き込みます。

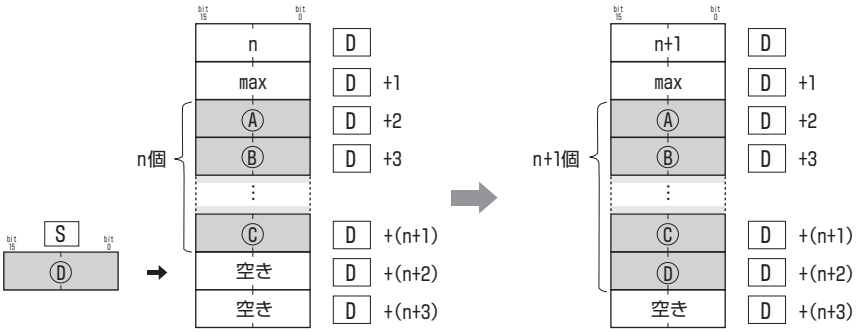
●データテーブル・データブロックについて

FIFOW, FIFOR関数は、データテーブルと呼ぶワードデータの集まりを使ってFIFO(First In First Out)処理します。データテーブルは、下図のような構造になっています。

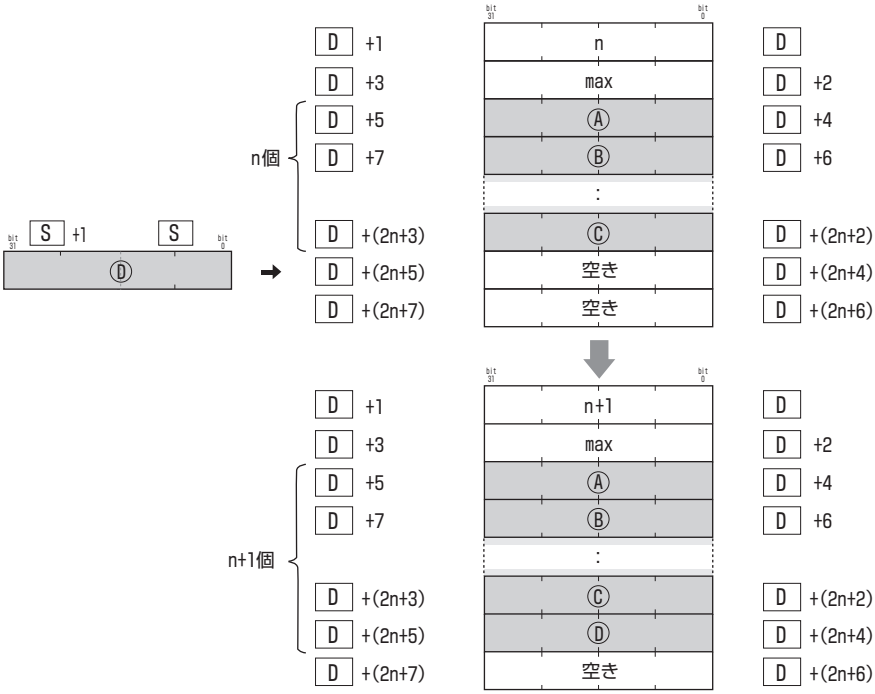


- 関数を最初に実行する前に、D に格納されるデータ数(n)をリセット(0を格納)してください。
  また、D +1に最大格納データ数を設定してください。
- データテーブルの先頭 D は、対応する FIFOR 関数のデータテーブルの先頭と一致させてください。

- S :データ格納対象が16ビット(.U/.S)の場合
  - データの格納先は D +(n+2)となります。
  - 格納後、n は1加算(+1)されます。



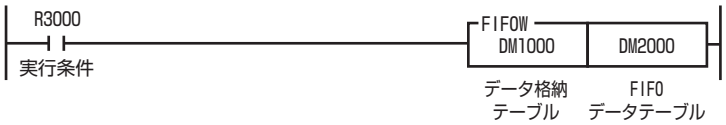
- S :データ格納対象が32ビット(.D/.L/.F)の場合
  - データの格納先は D +(2n+4)･ D +(2n+5)となります。
  - 格納後、n は1加算(+1)されます。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「テーブル処理命令」を参照してください。

●書式例

FIFOW (R3000, DM1000, DM2000.U)



ポイント

次のいずれかの条件が成立した場合、関数は実行されません。

- 最大格納データ数が0の場合。
- データ数nが最大格納データ数maxを超えた場合。
- データテーブルがデバイス/変数範囲を超えるような最大格納データ数maxの場合。

サンプルプログラム

●インデックスレジスタを使用した場合

スクリプト記述

FIFOW (R3000, DM1000, DM2000:Z1)

動作説明

R3000がONのとき、DM1000に格納されている16ビットデータを、DM(2000+Z1)を先頭とするデータブロックの最後に書き込みます。

参 考

KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ワンポイント

データテーブル先頭で指定するデータテーブルの、先頭には「格納されているデータ数」が保存されます。また先頭＋1には「最大格納データ数」をセットします。
FIFOW関数を使用する前に、格納されているデータ数をリセット(0を格納)し、最大格納データ数を設定する必要があります。(初期設定)

スクリプト記述

```
IF LDP ( R3000 ) THEN          ' R3000がONになったときデータテーブル全体をクリア
    DM1000 = 0
    DM1001 = 10                ' R3000がONになったとき最大格納データ数を10に設定
END IF
FIFOW ( LDP ( R3100 ), DM2000, DM1000 )   ' R3100がONになったときデータブロックに書き込み
```

動作説明

R3000の立ち上がり(OFF→ONの変化)を検出したとき、16ビットデータテーブル先頭DM1000から10個のデータブロックをもつデータテーブルを初期設定します。
(データテーブルは格納データ数、最大格納データ数、およびデータブロックで全12ワード)
R3100の立ち上がり(OFF→ONの変化)を検出したとき、DM2000に格納されている16ビットデータを、データブロックの最後に書き込みます。

ポイント

32ビットデータテーブルでは、格納データ数、最大格納データ数、格納データは全て32ビットデータで扱われますので、データテーブル構造(必要ワード数)には、注意が必要です。

参考

32ビットデータテーブルのとき、書き込みデータは、単精度浮動小数点型実数データも使用できます。

FIFOR FIFOテーブルから読み出し

FIFO読み出し

FIFOR ([実行条件,]※1 データテーブル先頭, データ格納対象)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D1	データテーブル先頭※2	FIFOデータテーブルの先頭を指定します。	.U	.U	.D	.D	.D	—	—	—	—	○	—
D2	データ格納対象※2	FIFOデータブロックから読み出したデータを格納する対象を指定します。	D1と同じ型として扱われます。					—	—	—	—	○	—
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

※2 ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。

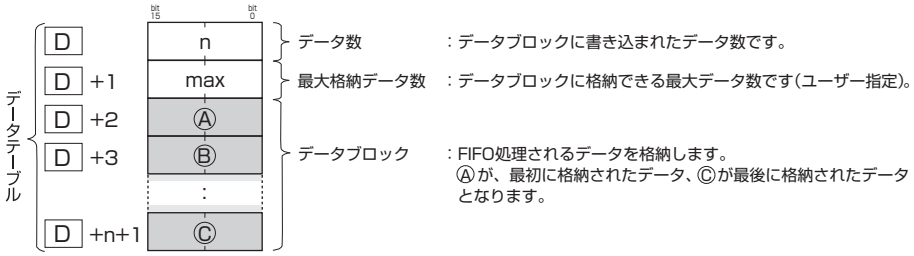
動作説明

実行条件が成り立つ(TRUE)とき、D1 で指定されたデータブロックに格納されている先頭のデータを読み出して、D2 へ格納します。

格納後、データブロックの内容が1データずつ前詰めされ、格納データ数が減算(–1)されます。

●データテーブル・データブロックについて

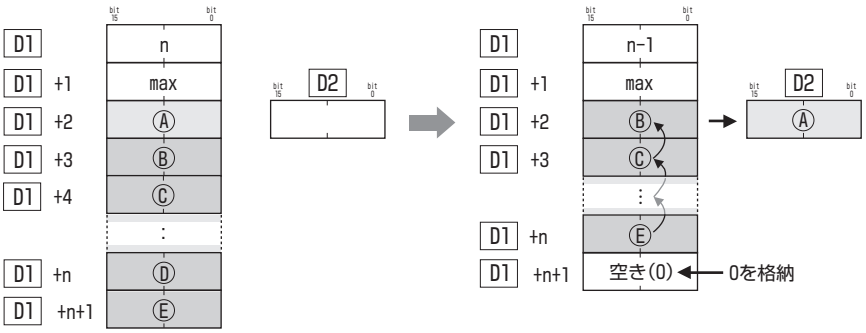
FIFOW, FIFOR関数は、データテーブルと呼ぶワードデータの集まりを使ってFIFO(First In First Out)処理します。データテーブルは、下図のような構造になっています。



- **D1 :データテーブル先頭が16ビット(.U/.S)の場合**

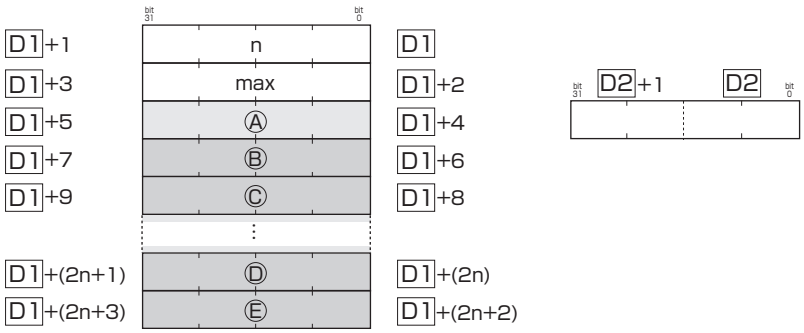
① 実行条件がONのとき、D1 +2に格納されている16ビットデータをD2 に格納します。

② 格納後データブロックの内容を1データずつ前詰めします。
前詰め後、n は1減算(－1)されます。

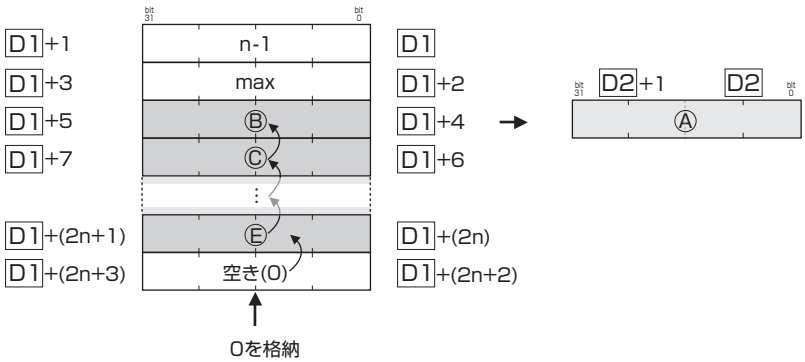


- D1 :データテーブル先頭が32ビット(.D/.L/.F)の場合

① 実行条件がONのとき、[D1+4・D1+5]に格納されている32ビットデータを[D2・D2+1]に格納します。



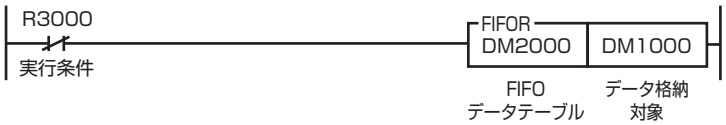
② 格納後データブロックの内容を1データずつ前詰めします。
前詰め後、nは1減算(－1)されます。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「テーブル処理命令」を参照してください。

●書式例

FIFOR (R3000 = OFF, DM2000.U, DM1000)



ポイント

次のいずれかの条件が成立した場合、関数は実行されません。

- データテーブルにデータが無い場合。
- データ数nが最大格納データ数maxより多い場合。
- データテーブルがデバイス/変数範囲を超えるような最大格納データ数maxの場合。

サンプルプログラム

●インデックスレジスタを使用した場合

スクリプト記述　**FIFOR** (R3000, DM1000.D, DM2000.D:Z1)

動作説明　R3000がONのとき、DM1000.Dを先頭とするデータブロックに格納されている先頭の32ビットデータを読み出して、DM(2000+Z1).Dに格納します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

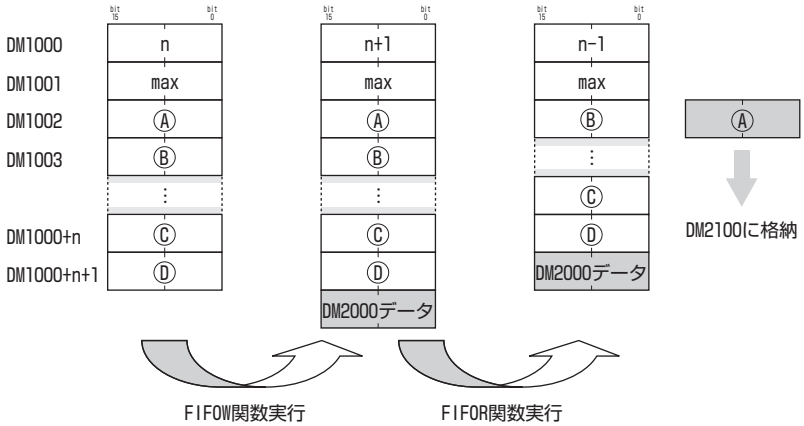
ワンポイント

FIFOR関数で読み出されるデータは、データブロックの先頭のデータが読み出されます。

スクリプト記述

```
FIFOW ( LDP ( R3100 ), DM2000, DM1000 )   ' R3100がONになったときデー
                                          ' タブロックに書き込み
FIFOR ( LDP ( R3101 ), DM1000, DM2100 )   ' R3101がONになったときデー
                                          ' タブロックから読み出し
```

動作説明　R3100の立ち上がり(OFF→ONの変化)を検出したとき、DM1000を先頭とするデータブロックにDM2000に格納されたデータを書き込みます。
R3101の立ち上がり(OFF→ONの変化)を検出したとき、テーブルブロックからデータを読み出し、DM2100に格納します。



MEMO

LIFOW LIFOテーブルへ書き込み

LIFO書き込み

LIFOW ([実行条件,]※1 データ格納対象, データテーブル先頭)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	データ格納対象※2	LIFOデータブロックに書き込むデータの格納先を指定します。	D と同じ型として扱われます。					—	—	—	—	○	○
D	データテーブル先頭	LIFOデータテーブルの先頭を指定します。	.U	.U	.D	.D	.D	—	—	—	—	○	—
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

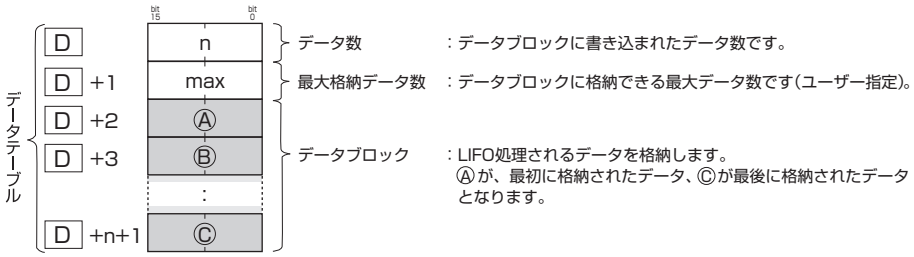
※2 ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。

動作説明

実行条件が成り立つ(TRUE)とき、S に格納されたデータ(16/32ビット)を、D で指定されたデータブロックの最後に書き込みます。

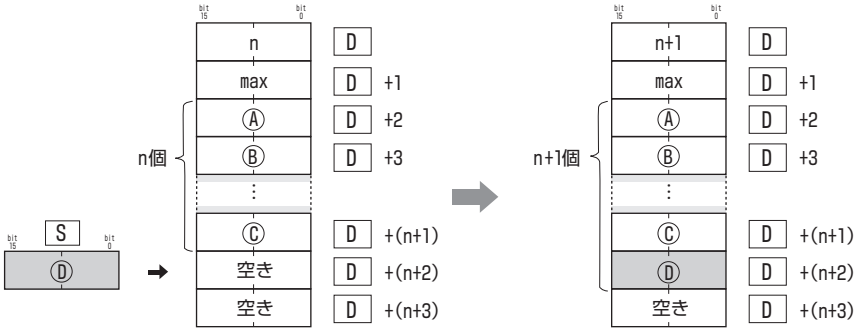
●データテーブル・データブロックについて

LIFOW, LIFOR関数は、データテーブルと呼ぶワードデータの集まりを使ってLIFO(Last In First Out)処理します。データテーブルは、下図のような構造になっています。

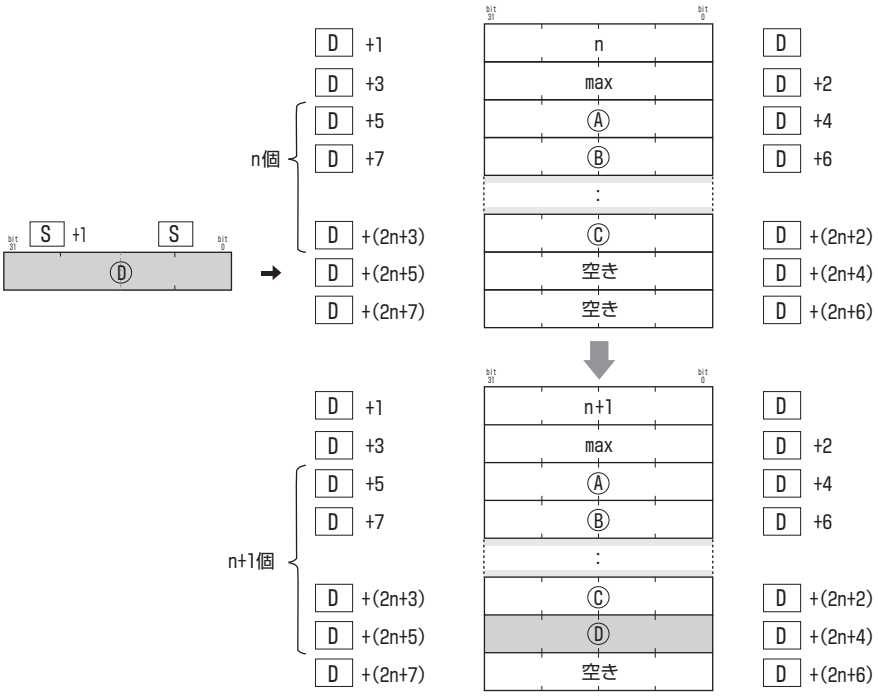


- 関数を最初に実行する前に、D に格納されるデータ数(n)をリセット(0を格納)してください。
  また、D +1に最大格納データ数を設定してください。
- データテーブルの先頭 D は、対応する LIFOR 関数のデータテーブルの先頭と一致させてください。

- S :データ格納対象が16ビット(.U/.S)の場合
  - データの格納先は、D + (n+2) となります。
  - 格納後、n は1加算(+1)されます。



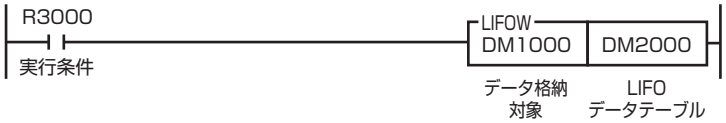
- S :データ格納対象が32ビット(.D/.L/.F)の場合
  - データの格納先は、D +(2n+4)・ D +(2n+5)となります。
  - 格納後:nは1加算(＋1)されます。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「テーブル処理命令」を参照してください。

●書式例

LIFOW (R3000, DM1000, DM2000.U)



ポイント

次のいずれかの条件が成立した場合、関数は実行されません。

- 最大格納データ数が0の場合。
- データ数nが最大格納データ数maxを超えた場合。
- データテーブルがデバイス/変数範囲を超えるような最大格納データ数maxの場合。

サンプルプログラム

●インデックスレジスタを使用した場合

スクリプト記述 **LIFOW** (R3000, DM1000:Z1, DM2000)

動作説明 R3000がONのとき、DM(1000+Z1)に格納されている16ビットデータを、DM2000を先頭とするデータブロックの最後に書き込みます。

参 考 KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

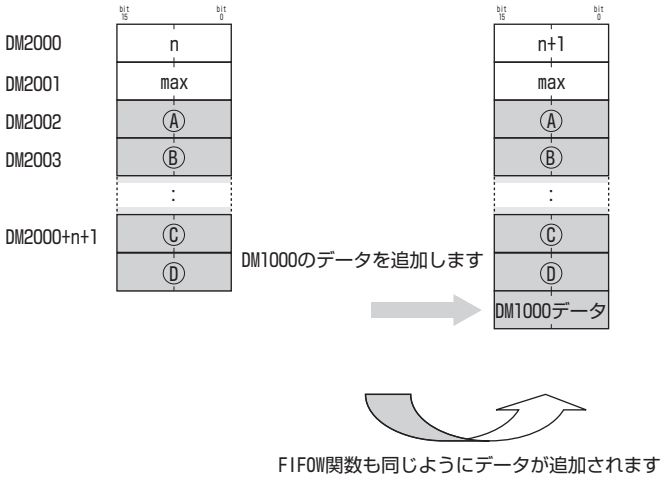
ワンポイント

LIFOW関数で使用するデータテーブルは、FIFOW関数で使用するデータテーブルと同じ構造で、同じデータテーブルを使用できます。
※詳しくはFIFOW関数を参照してください.。
またデータブロックへの書き込みも、LIFOW関数とFIFOW関数の格納方法は同じです。

スクリプト記述 **LIFOW** (DM1000, DM2000)

動作説明 毎スキャン(実行条件無し)、DM1000に格納されている16ビットデータを、DM2000を先頭とするデータブロックの最後に書き込みます。

ラダー変換



参 考 以下の記述と同じ動作をします。
FIFOW (DM1000, DM2000)
「FIFOW関数」(8-48ページ)

MEMO

LIFOR　LIFOテーブルから読み出し

FIFO読み出し

LIFOR ([実行条件,]※1 データテーブル先頭, データ格納対象)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D1	データテーブル先頭※2	LIFOデータテーブルの先頭を指定します。	.U	.U	.D	.D	.D	—	—	—	—	○	—
D2	データ格納対象※2	LIFOデータブロックから読み出したデータを格納する対象を指定します。	D1と同じ型として扱われます。					—	—	—	—	○	—
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1　[　]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

※2　ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。

動作説明

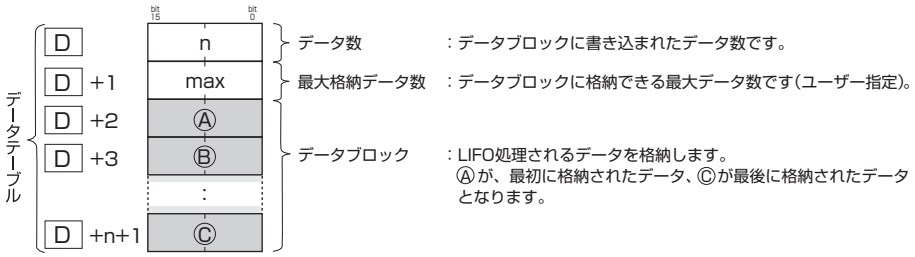
実行条件が成り立つ(TRUE)とき、D1 で指定されたデータブロックに格納されている最終のデータを読み出して、D2 へ格納します。

格納後、データブロックの内容が1データずつ前詰めされ、格納データ数が減算(-1)されます。

(読み出した最終データ部分には、0が格納されます)

●データテーブル・データブロックについて

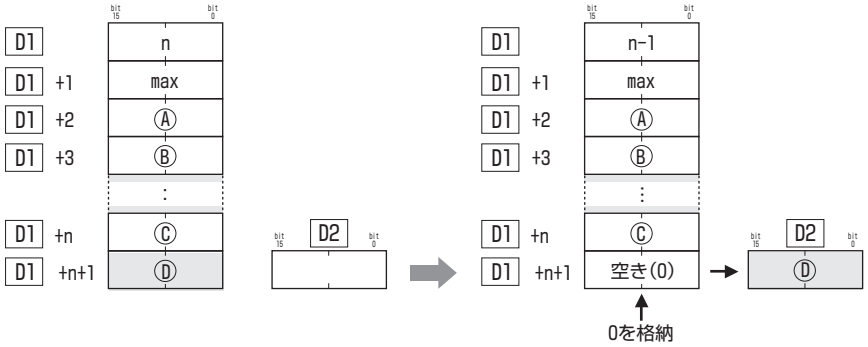
LIFOW, LIFOR関数は、データテーブルと呼ぶワードデータの集まりを使ってLIFO(Last In First Out)処理します。データテーブルは、下図のような構造になっています。



- **D1 :データテーブル先頭が16ビット(.U/.S)の場合**

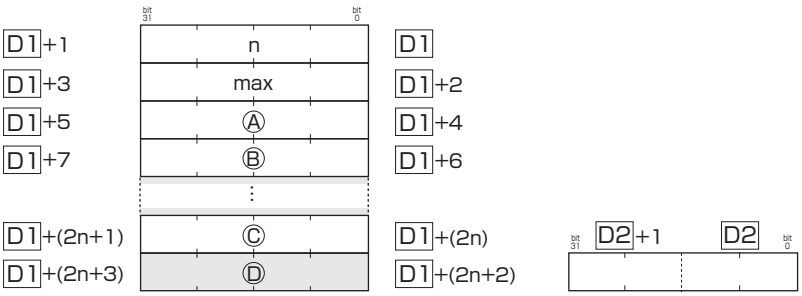
① 実行条件がONのとき、D1 +n+1に格納されている16ビットデータをD2 に格納します。

② 格納後、n は1減算(－1)されます。

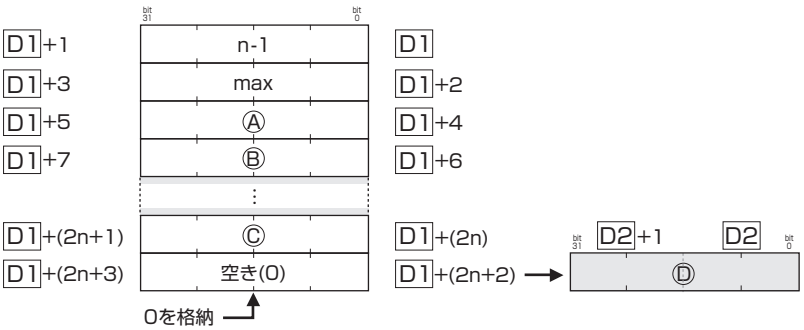


- D1 :データテーブル先頭が32ビット(.D/.L/.F)の場合

① 実行条件が ON のとき、[D1 +(2n+2)・D1 +(2n+3)] に格納されている 32 ビットデータを [D2 ・D2 +1]に格納します。



② 格納後、nは1減算(－1)されます。

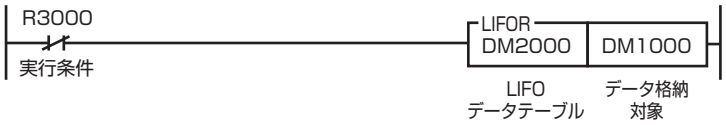


「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「テーブル処理命令」を参照してください。

●書式例

LIFOR (R3000 = OFF, DM2000.U, DM1000)



ポイント

次のいずれかの条件が成立した場合、関数は実行されません。

- データテーブルにデータが無い場合。
- データ数nが最大格納データ数maxより多い場合。
- データテーブルがデバイス/変数範囲を超えるような最大格納データ数maxの場合。

サンプルプログラム

●インデックスレジスタを使用した場合

スクリプト記述　**LIFOR** (R3000, DM1000.D, DM2000.D:Z1)

動作説明　R3000がONのとき、DM1000.Dを先頭とするデータブロックに格納されている最終の32ビットデータを読み出して、DM(2000+Z1).Dに格納します。

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

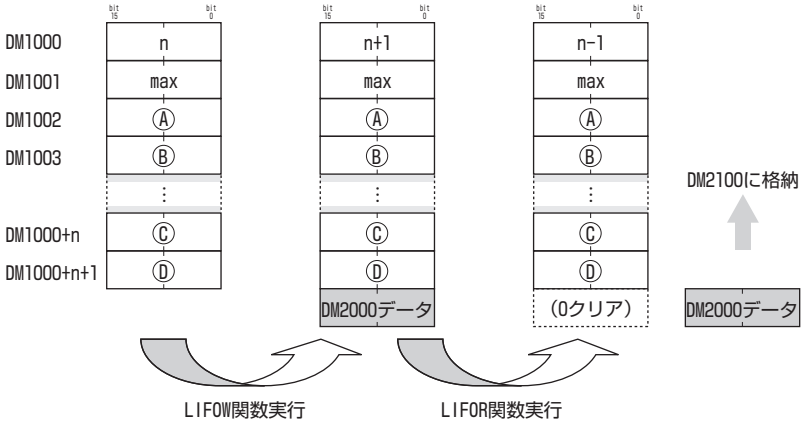
ワンポイント

LIFOR関数で読み出されるデータは、データブロックの最後のデータが読み出されます。

スクリプト記述　LIFOW (LDP (R3100), DM2000, DM1000)　' R3100がONになったときデータブロックに書き込み

LIFOR (LDP (R3101), DM1000, DM2100)　' R3101がONになったときデータブロックから読み出し

動作説明　R3100の立ち上がり(OFF→ONの変化)を検出したとき、DM1000を先頭とするデータブロックにDM2000に格納されたデータを書き込みます。
R3101の立ち上がり(OFF→ONの変化)を検出したとき、テーブルブロックからデータを読み出し、DM2100に格納します。



MEMO

FWRIT データテーブル指定位置の上書き

データテーブルの上書き

FWRIT ([実行条件,]※1 データ格納対象, データテーブル先頭, 書込ブロック位置)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	データ格納対象※2	上書きするデータの格納先、または上書きデータを指定します。	D と同じ型として扱われます。					—	—	—	—	○	○
D	データテーブル先頭※2	テーブルの先頭を指定します。	.U	.U	.D	.D	.D	—	—	—	—	○	—
n	書込ブロック位置	上書きするデータブロックの位置の格納先、または上書位置を指定します	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

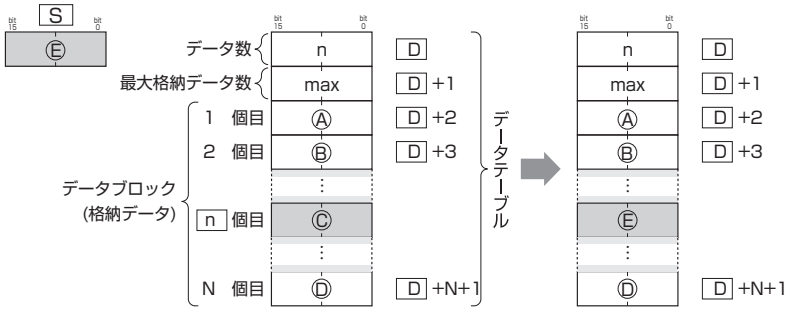
※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

※2 ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。

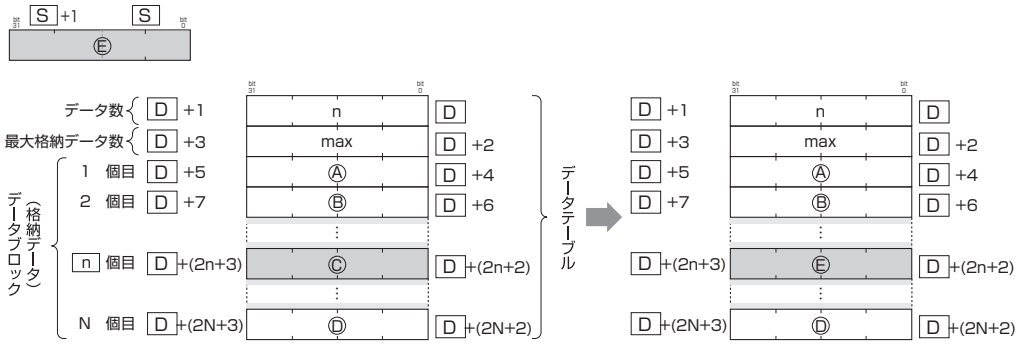
動作説明

実行条件が成り立つ(TRUE)とき、S に格納されたデータ(16/32ビット)を、D で指定されたデータブロックの、n で指定された位置(番目)に上書きします。

- **S :データ格納対象が16ビット(.U/.S)の場合**



- **S :データ格納対象が32ビット(.D/.L/.F)の場合**



「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「テーブル処理命令」を参照してください。

●書式例

FWRIT (R3000, DM1000.U, DM2000, 5)

ポイント

次のいずれかの条件が成立した場合、関数は実行されません。

- n の値が0または、データ数Nよりも大きい場合。
- 最大格納データ数maxが0の場合。
- データ数Nが最大格納データ数maxより大きい場合。
- デバイス/変数範囲を超えるような最大格納データ数maxの場合。

サンプルプログラム

●インデックスレジスタを使用した場合

スクリプト記述 **FWRIT** (R3000, DM1000:Z1, DM2000:Z2, DM2100)

動作説明 R3000がONのとき、DM(1000+Z1)に格納されている16ビットデータを、DM(2000+Z2)を先頭とするデータブロックの、DM2100で示されるブロック位置に上書きします。

参考 KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

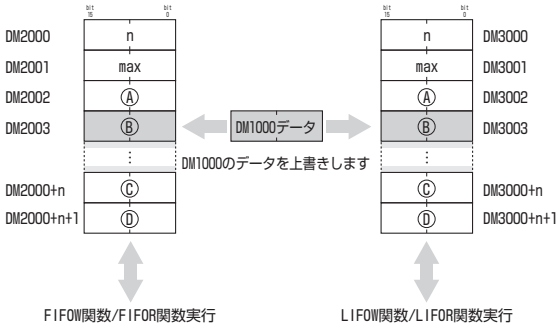
ワンポイント

FWRIT関数は、FIFOW関数/FIFOR関数、あるいはLIFOW関数/LIFOR関数で使用するデータテーブルのどちらにでも使用できます。

スクリプト記述

FIFOW (EM5000, DM2000)	'FIFOテーブル作成
LIFOW (EM5100, DM3000)	'LIFOテーブル作成
FWRIT (DM1000, DM2000, 2)	'FIFOテーブルを上書き
FWRIT (DM1000, DM3000, 2)	'LIFOテーブルを上書き

動作説明 毎スキャン(実行条件無し)、DM1000に格納されている16ビットデータを、DM2000を先頭とするFIFOデータブロックと、DM3000を先頭とするLIFOデータブロックの、それぞれ2番目に上書きします。



FINS データテーブル指定位置への挿入

データテーブルの挿入

FINS ([実行条件,]※1 データ格納対象, データテーブル先頭, 挿入ブロック位置)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	データ格納対象※2	上書きするデータの格納先、または上書きデータを指定します。	D と同じ型として扱われます。					—	—	—	—	○	○
D	データテーブル先頭※2	テーブルの先頭を指定します。	.U	.U	.D	.D	.D	—	—	—	—	○	—
n	挿入ブロック位置	挿入するデータブロックの格納先、または挿入位置を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

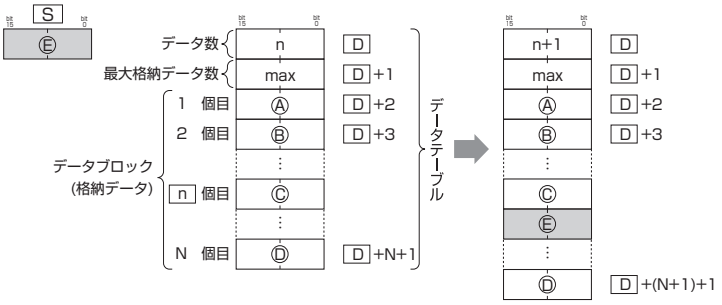
※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

※2 ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。

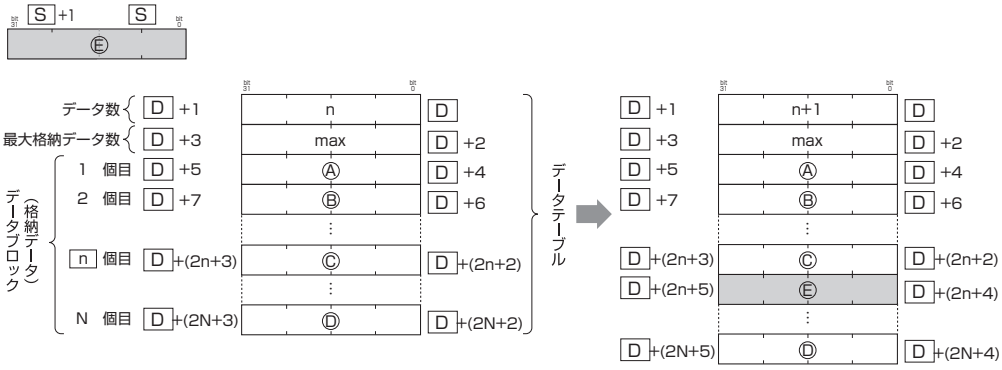
動作説明

実行条件が成り立つ(TRUE)とき、S に格納されたデータ(16/32ビット)を、D で指定されたデータブロックの、n で指定された位置(番目)に挿入します。

- **S :データ格納対象が16ビット(.U/.S)の場合**



- **S :データ格納対象が32ビット(.D/.L/.F)の場合**



「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「テーブル処理命令」を参照してください。

●書式例

FINS (R3000, DM1000.U, DM2000, 4)

ポイント

次のいずれかの条件が成立した場合、関数は実行されません。

- n の値が0または、データ数Nよりも大きい場合。
- 最大格納データ数maxが0の場合。
- データ数Nが最大格納データ数maxより大きい場合。
- デバイス/変数範囲を超えるような最大格納データ数maxの場合。

サンプルプログラム

●インデックスレジスタを使用した場合

スクリプト記述

FINS (R3000, DM1000:Z1, DM2000:Z2, DM2100)

動作説明

R3000がONのとき、DM(1000+Z1)に格納されている16ビットデータを、DM(2000+Z2)を先頭とするデータブロックの、DM2100で示されるブロック位置に挿入します。

参考

KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

ワンポイント

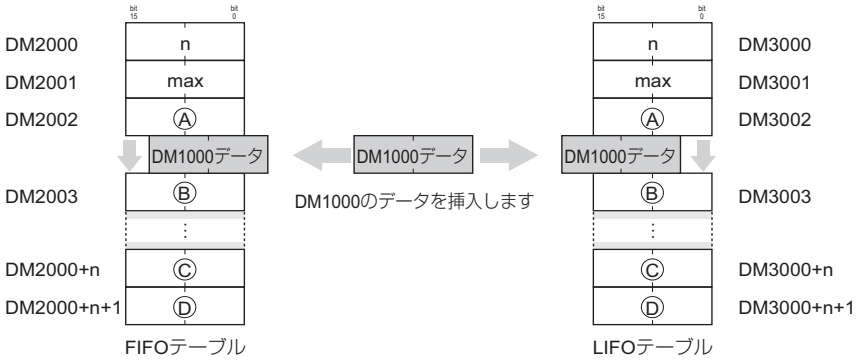
FINS関数は、FIFOW関数/FIFOR関数、あるいはLIFOW関数/LIFOR関数で使用するデータテーブルのどちらにでも使用できます。

スクリプト記述

FIFOW (EM5000, DM2000)	'FIFOテーブル作成
LIFOW (EM5100, DM3000)	'LIFOテーブル作成
FINS (DM1000, DM2000, 2)	'FIFOテーブルに挿入
FINS (DM1000, DM3000, 2)	'LIFOテーブルに挿入

動作説明

毎スキャン(実行条件無し)、DM1000に格納されている16ビットデータを、DM2000を先頭とするFIFOデータブロックと、DM3000を先頭とするLIFOデータブロックの、それぞれ2番目に挿入します。



FDEL データテーブル指定位置の削除

データテーブルの削除

FDEL ([実行条件,]※1 データテーブル先頭, 削除ブロック位置)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D	データテーブル先頭※2	テーブルの先頭を指定します。	.U	.U	.D	.D	.D	—	—	—	—	○	—
n	削除ブロック位置	削除するデータブロックの格納先、または削除位置を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

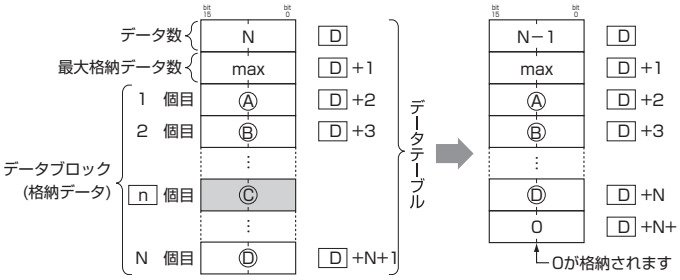
※2 ビットデータを指定した場合、連続した16/32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16/32ビットを扱います。

動作説明

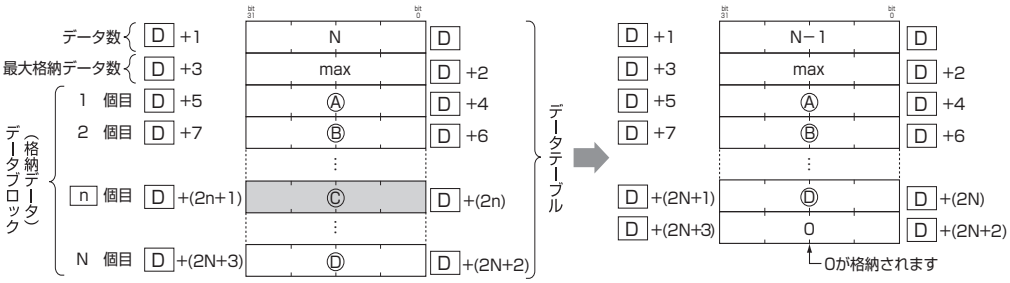
実行条件が成り立つ(TRUE)とき、D で指定されたデータブロックの、n で指定された位置(番目)のデータを削除します。

削除後、データブロックの内容が1データずつ前詰めされ、格納データ数が減算(-1)されます。

- **D :データテーブル先頭が16ビット(.U/.S)の場合**



- **D :データテーブル先頭が32ビット(.D/.L/.F)の場合**

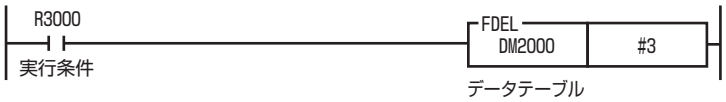


「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「テーブル処理命令」を参照してください。

●書式例

FDEL (R3000, DM2000.U, 3)



ポイント

次のいずれかの条件が成立した場合、関数は実行されません。

- n の値が0または、データ数Nよりも大きい場合。
- 最大格納データ数maxが0の場合。
- データ数Nが最大格納データ数maxより大きい場合。
- デバイス/変数範囲を超えるような最大格納データ数maxの場合。

サンプルプログラム

●インデックスレジスタを使用した場合

スクリプト記述 **FDEL** (R3000, DM2000:Z1, DM2100)

動作説明 R3000がONのとき、DM(2000+Z1)を先頭とするデータブロックの、DM2100で示されるブロック位置のデータを削除します。

参考 KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

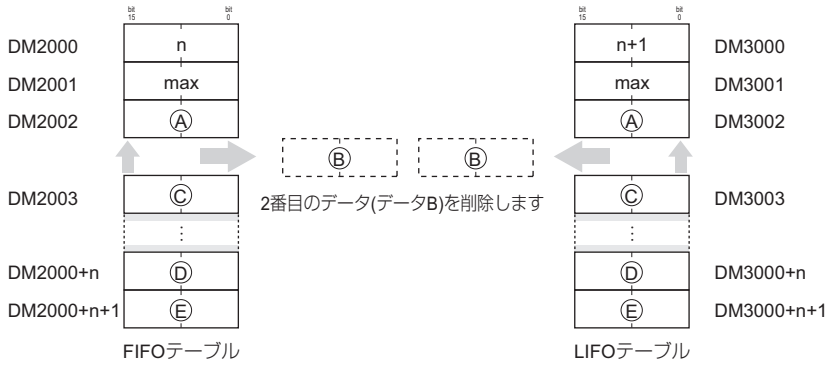
ワンポイント

FDEL関数は、FIFOW関数/FIFOR関数、あるいはLIFOW関数/LIFOR関数で使用するデータテーブルのどちらにでも使用できます。

スクリプト記述

FIFOW (EM5000, DM2000)	'FIFOテーブル作成
LIFOW (EM5100, DM3000)	'LIFOテーブル作成
FDEL (DM2000, 2)	'FIFOテーブルから削除
FDEL (DM3000, 2)	'LIFOテーブルから削除

動作説明 毎スキャン(実行条件無し)、DM2000を先頭とするFIFOデータブロックと、DM3000を先頭とするLIFOデータブロックの、それぞれ2番目のデータを削除します。



WTIME カレンダタイマに時刻データ書き込み

時刻書き込み

WTIME ([実行条件,]※1 年, 月, 日, 時, 分, 秒, 曜)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	時刻設定対象	年データ(0～99)	.U	.U	.U	.U	—	—	—	—	○	○	○
S2		月データ(1～12)	.U	.U	.U	.U	—	—	—	—	○	○	○
S3		日データ(1～31)	.U	.U	.U	.U	—	—	—	—	○	○	○
S4		時データ(0～23)	.U	.U	.U	.U	—	—	—	—	○	○	○
S5		分データ(0～59)	.U	.U	.U	.U	—	—	—	—	○	○	○
S6		秒データ(0～59)	.U	.U	.U	.U	—	—	—	—	○	○	○
S7		曜日データ(0～6)	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

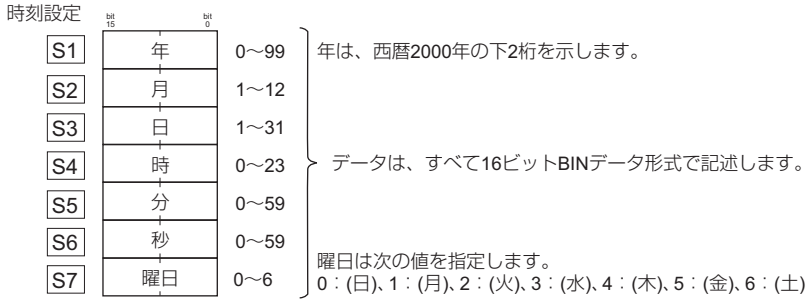
※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

参考 S1～S7にワードデータを指定した場合、すべて16ビット符号無しデータとして扱われます。(指定したサフィックスは無視されます)

※ KV-N14□□では使用できません。関数実行時エラーは発生せず実行されません。
(KV-N14□□、KV STUDIOは変換エラーになりません)

動作説明

実行条件が成り立つ(TRUE)とき、S1～S7に格納されている時刻データを内蔵カレンダタイマに書き込みます。

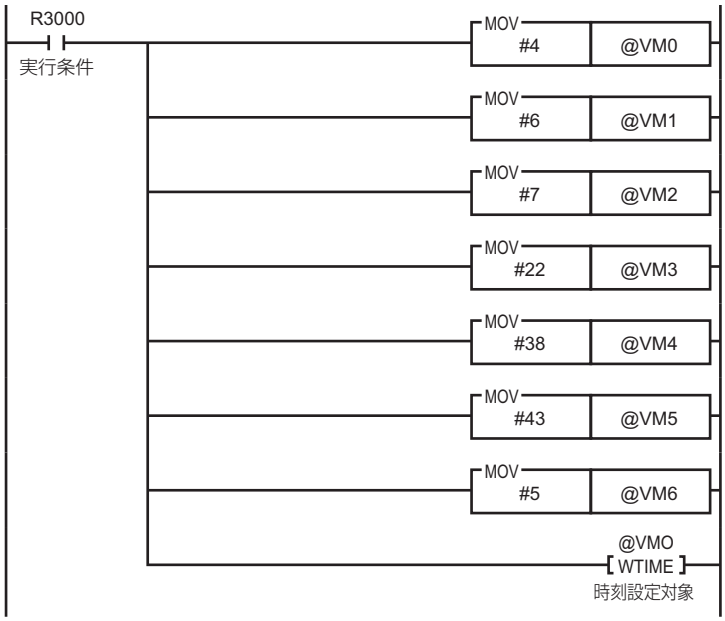


「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「時計処理命令」を参照してください。

●書式例

WTIME (R3000, 04, 06, 07, 22, 38, 43, 5)



ポイント

- 時刻データを個別に直接コントロールメモリに書き込むことはできません。
- **WTIME**関数を使用する場合、年・月・日・時・分・秒・曜日を全て正確に指定してください。年・月・日・時・分・秒に対する曜日のチェックはおこなわれません。
- 書き込むデータが時刻データとして不適切な場合、**WTIME**関数は実行しません。

SEC 日時データを秒形式に変換

戻り値※1 = **SEC** (変換元)
日時→秒変換

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 変換元※2※3	変換元の日時を格納した先頭を指定します。	.U	.U	.U	.U	.F	—	—	—	—	○	—
R 戻り値	SEC関数は、.Uの型を持ちます。※4	.U	.S	.D	.L	.F	※5 .DF	—	—	—	○	—

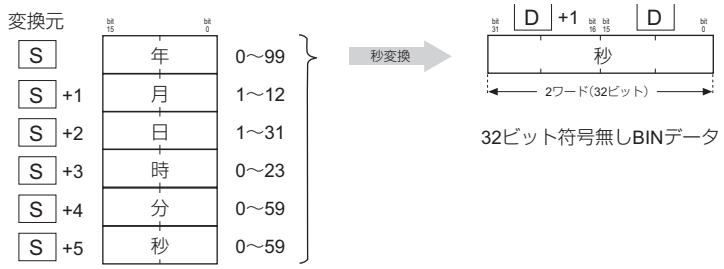
※1 戻り値の省略は可能です。省略した場合、演算結果はワークデバイス/ワークメモリに格納されます。
※2 ビットデータを指定した場合、日時データは連続した96ビット、秒データは連続した32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合でも可能です。
※3 T, C, CTC, CTH, Zは使用できません。
※4 戻り値に.D以外の型を指定した場合、変換時に自動的に型変換がおこなわれます。
※5 デバイスZに「.DF」サフィックスは指定できません。

ポイント
- **引数の「.DF」は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **引数の「.DF」は、KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)

動作説明

S を先頭に格納されている日時データの6ワード(96ビット)を、2000年1月1日0時0分0秒を基準日時とした通算秒に換算し、換算結果を32ビットバイナリデータ(符号無し)として R に格納します。



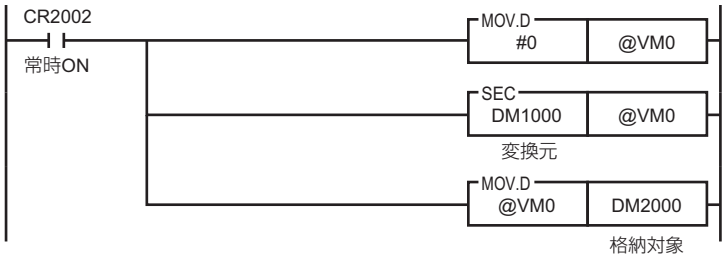
年は、西暦2000年の下2桁を指定します。

参考 日時データは、全て16ビットバイナリデータで指定し、符号無し/符号付きのどちらでも使用できます。

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「時計処理命令」を参照してください。

●書式例

DM2000.D = **SEC** (DM1000.U)



ポイント

次のいずれかの条件が成立した場合、R には0が格納されます。

- 変換する日時データが不適切な場合。
- S 、R に間接指定でタイマ･カウンタを指定した場合。

サンプルプログラム

●他の関数に組み込んで使用する場合

スクリプト記述

RSEC (TOL (**SEC** (CM0700)) + 864000, DM2000)

動作説明

現在の日時から10日後の日時を求めます。
CR0700を先頭に格納されている現在日時データを、基準日時からの通算秒に換算します。
換算結果の通算秒に864000秒(10日)を加算した10日後の通算秒を日時データに逆変換し、DM2000からの6ワードに格納します。

参 考

- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
  「RSEC関数」(8-74ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

RSEC 秒形式を日時データに変換

秒→日時変換

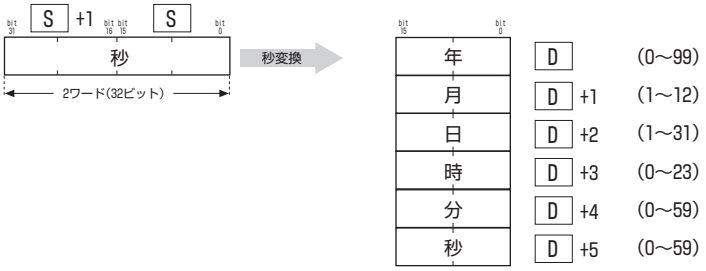
RSEC ([実行条件,]※1 変換元, 変換先)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	変換元※2※3	変換元の秒(通算秒)の格納先を指定します。	—	—	.D	.D	—	—	—	—	—	○	○
D	変換データ格納対象※2※3	変換した日時を格納する先頭を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ビットデータを指定した場合、日時データは連続した96ビット、秒データは連続した32ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合でも可能です。
※3 タイマ、カウンタは使用できません。

動作説明

S で指定された、2000年1月1日0時0分0秒を基準日時とした通算秒(32ビットバイナリデータ(符号無し))を日時データに変換し、D を先頭とする6ワード(96ビット)に格納します。

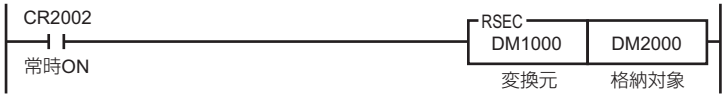


参考 日時データは、全て16ビットバイナリデータで格納されます。

「演算フラグについて(CR2009〜CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「時計処理命令」を参照してください。

●書式例

RSEC (DM1000.D, DM2000)



ポイント **次のいずれかの条件が成立した場合、関数は実行されません。**
- 変換する秒データが不適切な場合。
- S、D に間接指定でタイマ・カウンタを指定した場合。

サンプルプログラム

●他の関数を組み込んで使用する場合

スクリプト記述 **RSEC** (TOL (SEC (CM0700)) – 259200, DM2000)

動作説明 現在の日時から3日前の日時を求めます。
CM0700を先頭に格納されている現在日時データを、基準日時からの通算秒に換算します。
換算結果の通算秒から259200秒(3日)を減算した3日前の通算秒を日時データに逆変換し、DM2000からの6ワードに格納します。

参考
- 組み込む関数の引数によっては、使用できない場合がありますので注意してください。
「SEC関数」(8-72ページ)
- KV スクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

AJST　カレンダタイマ補正

アジャスト
AJST ([実行条件,]※1)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1　[　]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

※ KV-N14□□では使用できません。関数実行時エラーは発生せず実行されません。
(KV-N14□□、KV STUDIOは変換エラーになりません)

動作説明

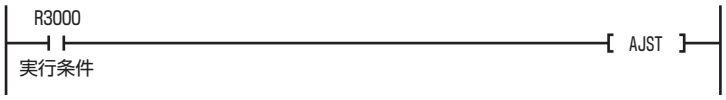
実行条件が成り立つ(TRUE)とき、内蔵カレンダタイマの現在時刻の「秒」が0～29のときは0にし、「秒」が30～59のときは「分」を桁上げして「秒」を0にして、分単位(0秒)で近い方に補正します。

実行前		実行後
12時59分15秒	→	12時59分00秒
12時59分30秒	→	13時00分00秒

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「時計処理命令」を参照してください。

●書式例

AJST (R3000)



参考　実行条件を省略した場合は、以下のような記述になります。

スクリプト記述　**AJST** (　)　　’実行条件の省略

参考　KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する」の手順5(2-10ページ)を参照してください。

MEMO

DI 割り込み禁止
EI 割り込み許可

割り込み禁止
DI ([実行条件,]※1)
割り込み許可
EI ([実行条件,]※1)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

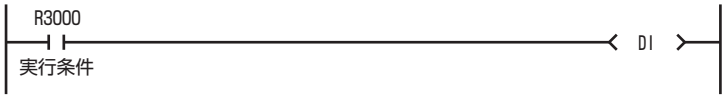
動作説明

DI関数は、実行条件が成り立つ(TRUE)とき、割り込みプログラムの実行を禁止します。
EI関数は、実行条件が成り立つ(TRUE)とき、割り込みプログラムの実行を許可します。
DI関数で一時的に禁止した割り込みプログラムの実行を、EI関数で許可(禁止の解除)します。

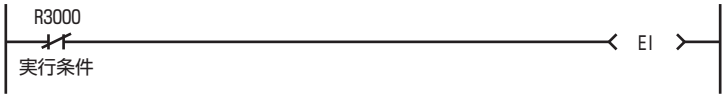
- 一度EI関数を実行すると、DI関数を実行するまでは割り込み許可となります。
- 運転開始時は常に割り込み禁止状態になっています。
- DI関数を実行中(割り込み禁止中)に割り込みが発生したときは、16重まで記憶されます。
  EI関数が実行されると記憶した割り込みが実行されます。
- 割り込みプログラムを実行中にさらに割り込みプログラムを実行したい場合は、割り込みプログラム中でEI関数を実行することにより16重まで受け付け可能となります。
  割り込み処理の詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「INT/RETI命令」「高速処理命令」を参照してください。

●書式例

DI (R3000)



EI (R3000 = OFF)



ポイント **運転開始時は常に割り込み禁止状態で、一度EI関数を実行すると、DI関数を実行するまで割り込み許可状態になります。**

参考 割り込みプログラムが実行されると、他の割り込みはその割り込みプログラムが終了するまで禁止状態になります。他の割り込みを許可したいときは、割り込みプログラムの実行中にEI関数を実行する必要があります。EI関数を実行すると、実行した割り込み要因より高い優先順位の割り込みプログラムを許可(多重割り込み)できます。ただし同じ要因の割り込みはできません。

RFSCTH 高速カウンタリフレッシュ

高速カウンタリフレッシュ

RFSCTH ([実行条件,]※1 高速カウンタ番号)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S	高速カウンタ番号	高速カウンタ番号を指定します。	—	—	—	—	—	—	—	—	※2 ○	—	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 高速カウンタ番号を数字一桁(0～3)で指定します。使用できる高速カウンタ番号はユニットにより異なります。

動作説明

実行条件が成り立つ(TRUE)とき、S で指定したCTHの現在値を最新の値にリフレッシュします。同時に、CTHのオーバーフローおよび変化方法のCR(コントロールリレー)も最新の状態にリフレッシュします。

KV-5500/5000/3000/1000の場合

高速カウンタ	リフレッシュされるコントロールリレー	
	オーバーフロー	変化方向
CTH0	CR2403	CR2412
CTH1	CR2407	CR2413

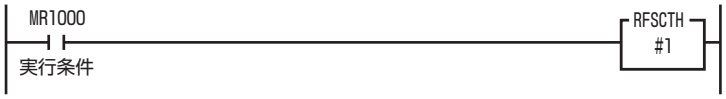
KV Nanoシリーズの場合

高速カウンタ	リフレッシュされるコントロールリレー	
	オーバーフロー	変化方向
CTH0	CR4206	CR4207
CTH1	CR4406	CR4407
CTH2	CR4606	CR4607
CTH3	CR4806	CR4807

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「高速処理命令」を参照してください。

●書式例

RFSCTH (MR1000, 1)



DIC 割り込み禁止範囲の設定

割り込み禁止

DIC ([実行条件,]※1 割込状態格納対象)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
D	割込状態格納対象※2	命令実行前の割り込み禁止状態を格納する対象を指定します。	—	—	—	—	—	—	.B	—	—	○	—
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

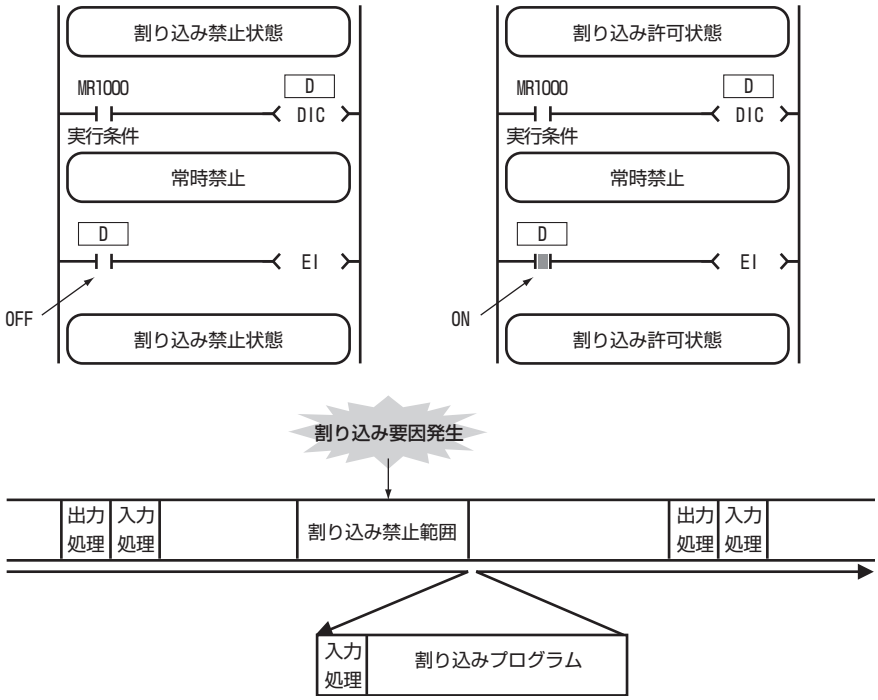
※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

※2 割込状態格納デバイスがワードデータのとき、割込状態の格納は最下位ビットのみが対象となります。(最下位ビット以外は変化しません)

動作説明

実行条件が成り立つ(TRUE)とき、割り込みプログラムの実行を禁止します。
同時に D を、関数実行前が割り込み許可状態であればセットし、割り込み禁止状態であればリセットします。

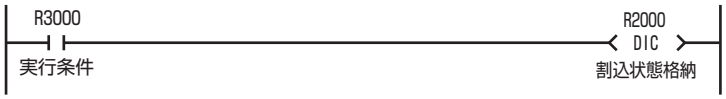
参考 割込状態格納対象、割り込みプログラムの実行を禁止する前の割り込み状態が格納されますので、EI関数(割り込み許可)の実行条件と組み合わせて割り込みをネスティング(入れ子)させる場合に使用します。



詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「高速処理命令」を参照してください。

●書式例

DIC (R3000, R2000)



サンプルプログラム

スクリプト記述

```
DIC ( R2000 )
FOR DM1000 = 0 TO 10
    DM2000.L = DM2000 + DM1000
NEXT
EI ( R2000 )
```

動作説明

毎スキャン(実行条件無し)、指定範囲での割り込みプログラムの実行を禁止します。DIC関数実行前が割り込み許可状態であればR2000をセットし、割り込み禁止状態であればR2000をリセットします。
R2000がON(割り込み処理実行前が割り込み許可状態)のとき、EI関数以下で割り込みプログラムの実行を許可します。

参 考

KVスクリプトは実行の前に、ラダープログラムに変換されています。変換内容を確認する場合は、「ボックススクリプトを新規作成する　手順5」(2-10ページ)を参照してください。

IEDGE 割り込み条件設定

割込条件設定

IEDGE ([実行条件,]※1 ユニット番号, 入力番号, 検出条件)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
n1 ユニット番号	入力割り込み検出条件を書き込む対象の拡張ユニット番号を指定します。(0～48)	.U	.U	.U	.U	—	—	—	—	※2 ○	—	—
n2 入力番号	割り込み要因番号を指定します。(0～31)	.U	.U	.U	.U	—	—	—	—	※3 ○	○	○
S 検出条件	検出条件(立ち上がり:0もしくは1、立ち下がり:2、立ち上がり・下がり両方:3)を指定します。	.U	.U	.U	.U	—	—	—	—	※4 ○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

動作説明

実行条件が成り立つ(TRUE)とき、S で指定した1ワードのデータを n1 番目のユニットの n2 番目の要因の割り込みの検出条件に設定します。

S の値により、検出条件は以下のようになります。

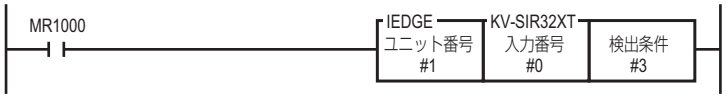
S	検出条件
0～1	立ち上がり
2	立ち下がり
3	立ち上がり・下がり両方

ポイント
- **S に 4～65535 を指定した場合は演算エラーにはなりませんが、検出条件は変更されません。**
- **本命令が実行完了したタイミングで、上記検出条件の設定が変更されます。**

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「IEDGE命令」を参照してください。

●書式例

IEDGE (MR1000,1,0,3)



MEMO

PIDAT　オートチューニング付きPID制御

AT付きPID制御

PIDAT ([実行条件,]※1 測定値, パラメータ, 操作量)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 定値	制御対象から観測された値を指定します。	.S	.S	.S	.S	—	—	—	—	—	○	○
D1 パラメータ※2	パラメータを格納している先頭を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
D2 操作量	PID演算の結果を格納する対象を指定します。	.S	.S	.S	.S	—	—	—	—	—	○	—
R 戻り値	なし	—	—	—	—	—	—	.B	—	—	—	—

※1　[　]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

※2　D1 パラメータは、連続する51ワードを占有します。パラメータの詳細はPIDAT命令語の説明を参照してください。

ポイント

- **PIDAT関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV Nanoシリーズ基本ユニットでのみ使用できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)

動作説明

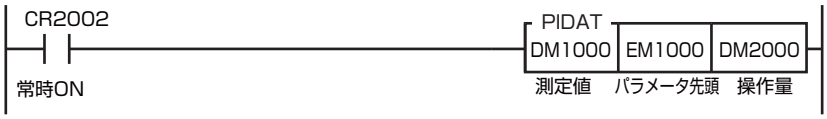
S と D1 から、D2 (PID演算結果)を取得します。
初回呼び出し時(初期化処理を実行する場合)は、実行条件ONかつ D1 のワークエリアを0クリアしてPIDAT関数を呼び出します。
演算を継続する場合は、実行条件ONを継続してPIDAT関数を呼び出します。

参考　PIDAT関数を実行条件OFFで呼び出した後、実行条件ONで呼び出すと初期化処理を実行します。

演算フラグや D1 の詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「PIDAT命令」を参照してください。

●書式例

PIDAT (DM1000, EM1000, DM2000)



MEMO

LOGE 指定IDのロギング許可
LOGD 指定IDのロギング禁止

ロギング許可
LOGE ([実行条件,]※1 ロギングID)
ロギング不許可
LOGD ([実行条件,]※1 ロギングID)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
n ロギングID	ロギングの許可/不許可をおこなうロギングIDを格納した対象、またはをロギングIDを指定します。(0～9)	.U	.U	.U	.U	—	—	—	—	※2 ○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 “$”での指定はできません。

動作説明

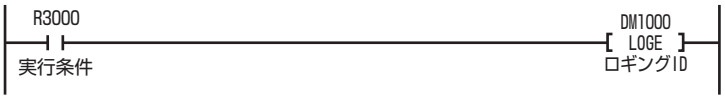
LOGE関数は、実行条件が成り立つ(TRUE)とき、n で指定したロギングIDのロギングを実行許可状態にします。関数実行後、ロギングIDごとのロギング実行リレーがセットされます。
LOGD関数は、実行条件が成り立つ(TRUE)とき、n で指定したロギングIDのロギングを実行不許可状態にします。関数実行後、ロギングIDごとのロギング実行リレーがリセットされます。

ロギングID	ロギング実行リレー
0	CR0000
1	CR0100
2	CR0200
3	CR0300
4	CR0400
5	CR0500
6	CR0600
7	CR0700
8	CR0800
9	CR0900

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、使用しているCPUユニットのユーザーズマニュアルを参照してください。

●書式例

LOGE (R3000, DM1000)



ポイント n に定数以外を設定したとき、10以上を設定した場合、関数は実行されません。

TRGD ロギングデータ取得

ロギングデータ取得
TRGD ([実行条件,]※1 ロギングID)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
n ロギングID	ロギングIDを指定します。(0～10)※2	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2 ロギングIDは、0～10を指定してください。リアルタイムチャートモニタでトリガ命令を使用する場合は、ロギングIDに10を指定してください。

動作説明

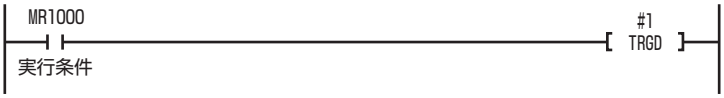
実行条件が成り立つ(TRUE)とき、 n で指定したロギングIDのデータを取得します。
通常、ロギング・トレース・リアルタイムチャートモニタはEND処理のタイミングでデータを取得しますが、TRGD命令を使用することで、ラダーの任意のタイミングでデータが取得できます。

ポイント
- **TRGD関数(命令)は、1つのプロジェクトにつき各IDで8個までしか使えません。**
- **TRGD関数(命令)でデータを取得するには、KV STUDIOのロギング・トレース・リアルタイムチャートモニタの設定で、トリガ種類を「トリガ命令」に設定する必要があります。「トリガ命令」に設定していない場合、TRGD関数(命令)を実行してもデータは取得されません。**

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、使用しているCPUユニットのユーザーズマニュアルを参照してください。

●書式例

TRGD (MR1000, 1)



TRGR デバイス/変数データ取得

デバイス/変数データ取得

TRGR ([実行条件,]※1 運転記録ID)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
n 運転記録ID	運転記録IDを指定します。(0～3)	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

動作説明

実行条件が成り立つ(TRUE)とき、 n で指定した運転記録IDのデータを取得します。
通常はEND処理のタイミングでデータを取得しますが、TRGR命令を使用することで、ラダーの任意のタイミングでデータが取得できます。

ポイント

- **TRGR命令はKV STUDIO Ver.10.0以降で対応機種がKV-8000のときのみ使用できます。**
- **TRGR関数(命令)でデータを取得するには、KV STUDIOの運転記録設定でサンプリング周期を「トリガ命令」に設定する必要があります。「トリガ命令」に設定していない場合、TRGR関数(命令)を実行してもデータは取得されません。**

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000シリーズユーザーズマニュアル』「運転記録機能」を参照してください。

●書式例

TRGR(MR1000, 0)

MEMO

MWRIT ストレージ(メモリカード/CPUメモリ)書き込み

戻り値※1 = **MWRIT** (保存ファイル名, 保存データ格納対象, 保存データ数格納対象, 保存形式格納対象, 保存結果格納対象)
ストレージ書込

引数/戻り値	説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
S1 保存ファイル名※7	データを保存するファイル名の格納先、または保存ファイル名を指定します。(1～247文字(終端コードNUL、フォルダ名を含む)の範囲の文字列を指定します)	—	—	—	—	—	—	—	.T	※3 ○	○	○
S2 保存データ格納対象※4※5	保存するデータを格納している先頭を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
n 保存データ数格納対象	保存する範囲のデータ数の格納先、または保存データ数を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	○
D1 保存形式格納対象※5※6	保存形式等に関するパラメータ、およびデータの保存状態を格納する先頭を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
D2 保存結果格納対象※2	保存結果を通知するビットデータを指定します。	—	—	—	—	—	—	.B	—	—	○	—
R 戻り値	関数の起動に成功したらONし、失敗のときOFFになります。	—	—	—	—	—	—	.B	—	—	○	—

※1 戻り値の省略は可能です。省略した場合、演算結果はワークデバイス/ワークリレーに格納されます。
※2 保存結果格納対象にビットデータを指定した場合、連続する2ビットが通知ビットとして使用され、ワードデータを指定した場合は、連続する2ワードの各々最下位ビットが通知ビットデータとして使用されます。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 インデックスレジスタZおよびアクセスウィンドウで設定したデジタルトリマ値を保存する場合は、一度データメモリDM等にコピーして、そのデータメモリDMを設定してください。
※5 ビットデータを使用する場合はチャンネルの先頭を指定してください。
※6 D1 にビットデータを指定した場合、連続する96ビットを占有します。ワードデータを指定した場合、連続する6ワードを占有します。
※7 T, C, CR, CTC, CTH, Zは使用できません。

ポイント **KV-N14□□/N24□□/N40□□/N60□□では使用できません。(《KV STUDIO》では変換エラーにならず、警告のみ表示されます。)**

動作説明

S1 で指定されたファイル名(文字列)で、S2 を先頭として、n で指定された範囲に格納されたデータをメモリカードまたはCPUメモリに書き込みます。
書き込み処理は、D1 で指定された書き込みパラメータに従っておこなわれ、D2 に書き込み完了通知と異常終了アラーム通知が格納されます。
MWRIT関数が正常に起動できた場合、R をONし、失敗した場合はOFFします。
D1 に指定する保存形式等に関するパラメータの詳細は、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「MWRIT命令」を参照してください。

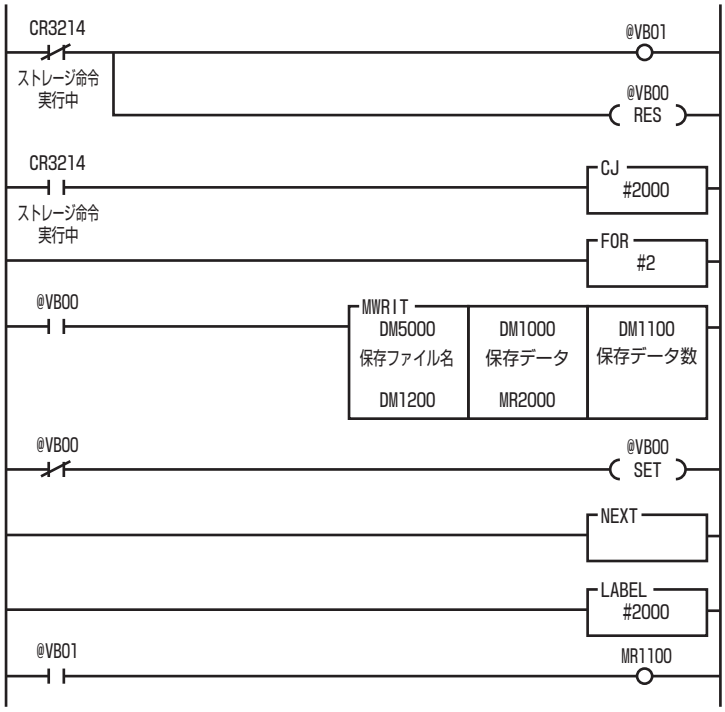
参考　D2 保存結果格納対象の通知ビットは次のような意味を持ちます。
(例) MR2000を指定した場合

通知ビット		ON(1、TRUE)	OFF(0、FALSE)
MR2000	完了通知	完了	未完了
MR2001	異常通知	異常終了	正常終了

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ストレージ命令」を参照してください。

●書式例

MR1100 = **MWRIT** (DM5000.T, DM1000.U, DM1100.U, DM1200.U, MR2000)



ポイント　**次のいずれかの条件が成立した場合、関数は実行されません。**

- S1 で指定した文字列に終端コード(NUL(00H))が無い場合。
- S1 で指定した文字列が247文字(終端コード(NUL(00H))含む)を超えた場合。
- S1 に間接指定でタイマ･カウンタを指定した場合。
- S1 、S2 、D1 にビットデータを指定したとき、チャンネルの先頭以外を指定した場合。
- 引数に範囲外の値を指定した場合。
- 引数の予約に値を指定した場合。

MREAD ストレージ(メモリカード/CPUメモリ)読み出し

戻り値※1 = **MREAD** (読出ファイル名, 読出データ格納対象, 読出データ数格納対象, 読出形式格納対象, 読出結果格納対象)
ストレージ読出

引数/戻り値	説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
S1 読出ファイル名※4※7	データを読み出すファイル名の格納先、または読出ファイル名を指定します。(1～247文字(終端コードNUL、フォルダ名を含む)の範囲の文字列を指定します)	—	—	—	—	—	—	—	.T	※3 ○	○	○
S2 読出データ格納対象※4	読み出したデータを格納している先頭を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
n 読出データ数格納対象※5	読み出す範囲のデータ数の格納先、または読み出しデータ数を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
D1 読出形式格納対象※6	読み出す形式等に関するパラメータ、およびデータの読み出し状態を格納する先頭を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
D2 読出結果格納対象※2※4	読み出し結果を通知するビットデータを指定します。	—	—	—	—	—	.B	—	—	—	○	—
R 戻り値	関数の起動に成功したらONし、失敗のときOFFになります。	—	—	—	—	—	.B	—	—	—	○	—

※1 戻り値の省略は可能です。省略した場合、演算結果はワークデバイス/ワークリレーに格納されます。
※2 読出結果格納対象にビットデータを指定した場合、連続する2ビットが通知ビットとして使用され、ワードデータを指定した場合は、連続する2ワードの各々最下位ビットが通知ビットデータとして使用されます。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 ビットデータを使用する場合は、チャンネルの先頭を指定してください。
※5 ASCII文字列データを読み出す場合、文字列の最大文字数(バイト数)を指定します。
※6 D2 にビットデータを指定した場合、連続する96ビットを占有します。ワードデータを指定した場合、連続する6ワードを占有します。
※7 T, C, CR, CTC, CTH, Zは使用できません。

ポイント **KV-N14□□/N24□□/N40□□/N60□□では使用できません。(《KV STUDIO》では変換エラーにならず、警告のみ表示されます。)**

動作説明

ストレージから S で指定されたファイル名(文字列)のデータを、D1 を先頭として、n で指定された範囲に読み出し、格納します。
読み出し処理は、D2 で指定された読み出しパラメータに従っておこなわれ、D3 に読み出し完了通知と異常終了アラーム通知が格納されます。
MREAD関数が正常に起動できた場合、R をONし、失敗した場合はOFFします。
D2 に指定する読み出す形式等に関するパラメータの詳細は、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「MREAD命令」を参照してください。

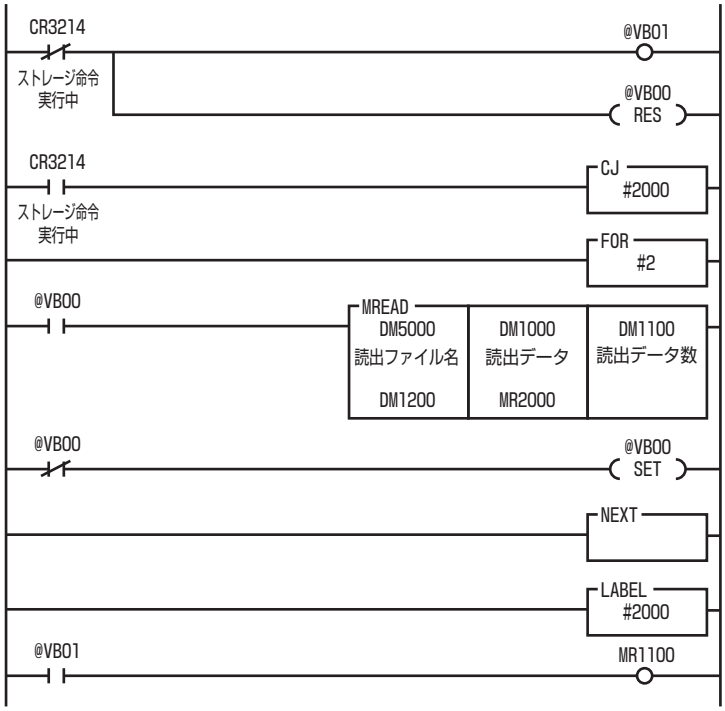
参考　D3 読出結果格納対象の通知ビットは次のような意味を持ちます。
(例) MR2000を指定した場合

通知ビット		ON	OFF
MR2000	完了通知	完了	未完了
MR2001	異常通知	異常終了	正常終了

「演算フラグについて（CR2009～CR2012）」（1-18ページ）
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ストレージ命令」を参照してください。

●書式例

MR1100 = **MREAD** (DM5000.T, DM1000.U, DM1100.U, DM1200.U, MR2000)



ポイント　**次のいずれかの条件が成立した場合、関数は実行されません。**

- S で指定した文字列に終端コード(NUL(00H))が無い場合。
- S で指定した文字列が247文字(終端コード(NUL(00H))含む)を超えた場合。
- S に間接指定でタイマ･カウンタを指定した場合。
- S 、D1 、D2 にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。
- 引数に範囲外の値を指定した場合。
- 引数の予約に値を指定した場合。

MFREE ストレージ(メモリカード/CPUメモリ)の空き容量をバイト単位で取得

ストレージ容量取得

戻り値※1 = **MFREE** ((ドライブ番号、)※4 容量格納対象、取得結果格納対象)

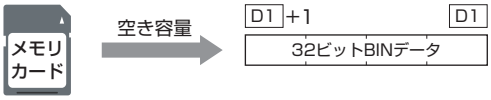
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
n ドライブ番号※4	空き容量を確認するドライブの番号(メモリカード:0、CPUメモリ:1):を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
D1 容量格納対象※2	取得したメモリカード(CPUメモリ)の空き容量(バイト単位)を格納する対象を指定します。(32ビットデータ)	—	—	.D	.D	—	—	—	—	—	○	—
D2 取得結果格納対象※3	空き容量の取得結果を通知するビットデータを指定します。	—	—	—	—	—	—	.B	—	—	○	—
R 戻り値	関数の起動に成功したらONし、失敗のときOFFになります。	—	—	—	—	—	—	.B	—	—	○	—

※1 戻り値の省略は可能です。省略した場合、演算結果はワークデバイス/ワークリレーに格納されます。
※2 ビットデータを指定した場合、チャンネルの先頭から連続した32ビットを扱います。
※3 取得結果格納対象にビットデータを指定した場合、連続する2ビットが通知ビットとして使用され、ワードデータを指定した場合は、連続する2ワードの各々最下位ビットが通知ビットデータとして使用されます。
※4 KV-8000/7000シリーズの場合は、空き容量を確認するストレージを、メモリカードまたはCPUメモリから選択できます。

ポイント **KV-N14□□/N24□□/N40□□/N60□□では使用できません。(《KV STUDIO》では変換エラーにならず、警告のみ表示されます。)**

動作説明

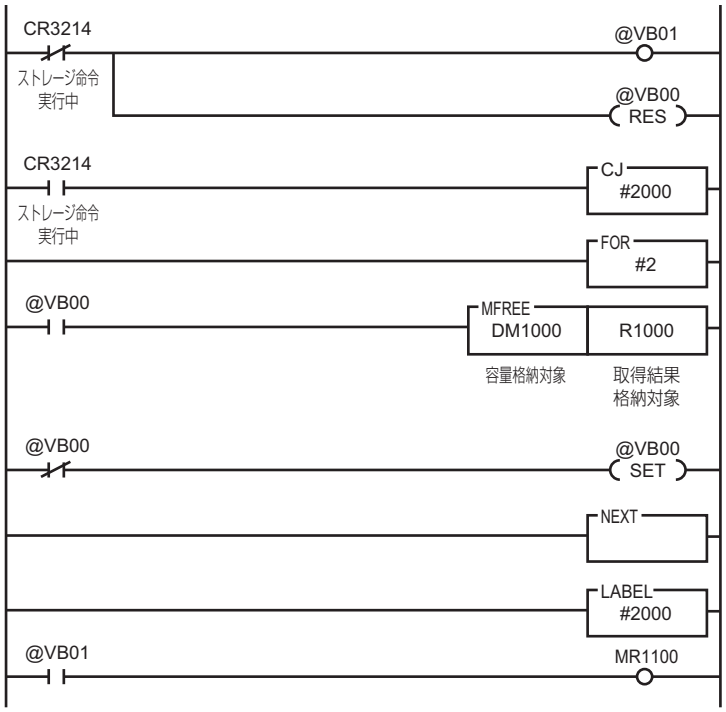
n で指定されたメモリカードまたはCPUメモリの空き容量(バイト単位)を取得して、D1 に32ビットバイナリデータ(2ワード)として格納し、D2 に空き容量取得の完了通知と異常終了アラーム通知が格納されます。



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「ストレージ命令」を参照してください。

●書式例(KV-5000の場合)

MR1100 = **MFREE** (DM1000.D, R1000)



●通知ビットデータ(D2 :完了通知、D2 +1:エラー通知)

項　目		内　容
D2	完了通知	命令実行開始時にOFF、メモリカード(またはCPUメモリ)より空き容量取得処理終了時にONします。命令が正常終了、異常終了のどちらかでもONします。
D2 +1	エラー通知	命令が正常終了でOFF、異常終了でONします。

ポイント

- **完了通知、エラー通知は命令実行完了後END処理で結果が書き込まれます。**
- **空き容量取得には、数スキャンを要する場合があります。**
- **取得処理中に FTP サーバ機能経由でファイル操作を行っていた場合や、ロギング機能を使用していた場合、取得した容量は実際の空き容量と異なる場合があります。**
- **ストレージ命令実行中(CR3214がONしている)は、MFREE関数は実行されません。**
- **D1 にチャンネルの先頭以外のリレーを指定した場合、関数は実行されません。**
- **割り込みプログラムおよび初期化モジュール内では使用できません。**

MFREEK ストレージ(メモリカード/CPUメモリ)の空き容量をキロバイト単位で取得

戻り値※1 = **MFREEK** (ストレージ容量取得 (ドライブ番号、)※4 容量格納対象、取得結果格納対象)

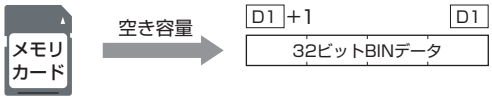
引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
n ドライブ番号※4	空き容量を確認するドライブの番号(メモリカード:0、CPUメモリ:1):を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
D1 容量格納対象※2	取得したメモリカード(CPUメモリ)の空き容量(キロバイト単位)を格納する対象を指定します。(32ビットデータ)	—	—	.D	.D	—	—	—	—	—	○	—
D2 取得結果格納対象※3	空き容量の取得結果を通知するビットデータを指定します。	—	—	—	—	—	—	.B	—	—	○	—
R 戻り値	関数の起動に成功したらONし、失敗のときOFFになります。	—	—	—	—	—	—	.B	—	—	○	—

※1 戻り値の省略は可能です。省略した場合、演算結果はワークデバイス/ワークリレーに格納されます。
※2 ビットデータを指定した場合、チャンネルの先頭から連続した32ビットを扱います。
※3 取得結果格納対象にビットデータを指定した場合、連続する2ビットが通知ビットとして使用され、ワードデータを指定した場合は、連続する2ワードの各々最下位ビットが通知ビットデータとして使用されます。
※4 KV-8000/7000シリーズの場合は、空き容量を確認するストレージを、メモリカードまたはCPUメモリから選択できます。

ポイント
- **MFREEK関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV-NC32Tでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)
- **KV-N14□□/N24□□/N40□□/N60□□では使用できません。(《KV STUDIO》では変換エラーにならず、警告のみ表示されます。)**

動作説明

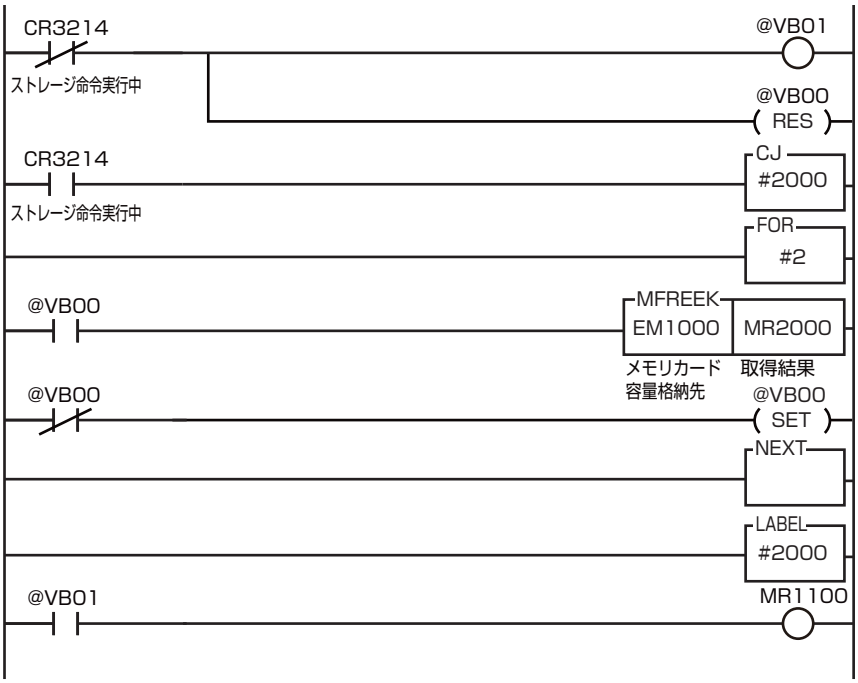
n で指定されたメモリカードまたはCPUメモリの空き容量(キロバイト単位)を取得して、D1 に32ビットバイナリデータ(2ワード)として格納し、D2 に空き容量取得の完了通知と異常終了アラーム通知が格納されます。



「演算フラグについて(CR2009〜CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「MFREEK命令」を参照してください。

● **書式例(KV-5000の場合)**

MR1100 = **MFREEK** (EM1000.D, MR2000)



● **通知ビットデータ(D2 :完了通知、 D2 +1:エラー通知)**

項　目		内　容
D2	完了通知	命令実行開始時にOFF、メモリカード(CPUメモリ)より空き容量取得処理終了時にONします。命令が正常終了、異常終了のどちらかでもONします。
D2 +1	エラー通知	命令が正常終了でOFF、異常終了でONします。

ポイント

- **完了通知、エラー通知は命令実行完了後END処理で結果が書き込まれます。**
- **空き容量取得には、数スキャンを要する場合があります。**
- **取得処理中に FTP サーバ機能経由でファイル操作をおこなっていた場合や、ロギング機能を使用していた場合、取得した容量は実際の空き容量と異なる場合があります。**
- **ストレージ命令実行中(CR3214がONしている)は、MFREEK関数は実行されません。**
- **D1 にチャンネルの先頭以外のリレーを指定した場合、関数は実行されません。**
- **割り込みプログラムおよび初期化モジュール内では使用できません。**

MMKDIR ストレージ(メモリカード/CPUメモリ)フォルダ作成

戻り値※1 = **MMKDIR** (作成フォルダ名、作成結果格納対象)
ストレージフォルダ作成

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 作成フォルダ名※3※5	作成するフォルダのパス名の格納先、または作成フォルダパス名を指定します。(1～247文字(終端コードNUL含む)の範囲の文字列を指定します)	—	—	—	—	—	—	—	.T	※2 ○	○	○
D 作成結果格納対象※4	作成結果を通知する対象を指定します。	—	—	—	—	—	—	.B	—	—	○	—
R 戻り値	関数の起動に成功したらONし、失敗のときOFFになります。	—	—	—	—	—	—	.B	—	—	○	—

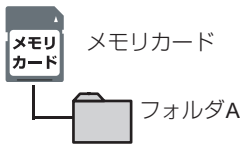
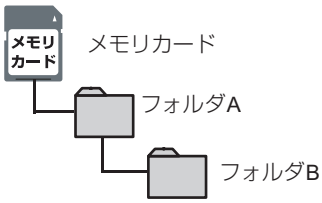
※1 戻り値の省略は可能です。省略した場合、演算結果はワークデバイス/ワークリレーに格納されます。
※2 定数で文字列を指定する場合は、" "で囲ってください。
※3 ビットデータを指定した場合、チャンネルの先頭を指定してください。
※4 ビットデータを指定した場合、連続する2ビットを占有します。ワードデータを指定した場合、連続する2ワードのそれぞれ最下位ビットを通知ビットデータとして使用します。最下位ビット以外は変化しません。
※5 T, C, CR, CTC, CTH, Zは使用できません。

ポイント
- **KV-N14□□/N24□□/N40□□/N60□□では使用できません。(《KV STUDIO》では変換エラーにならず、警告のみ表示されます。)**

動作説明

S で指定されたパス名(文字列)で、メモリカードまたはCPUメモリにフォルダを作成します。
作成結果が D に格納されます。

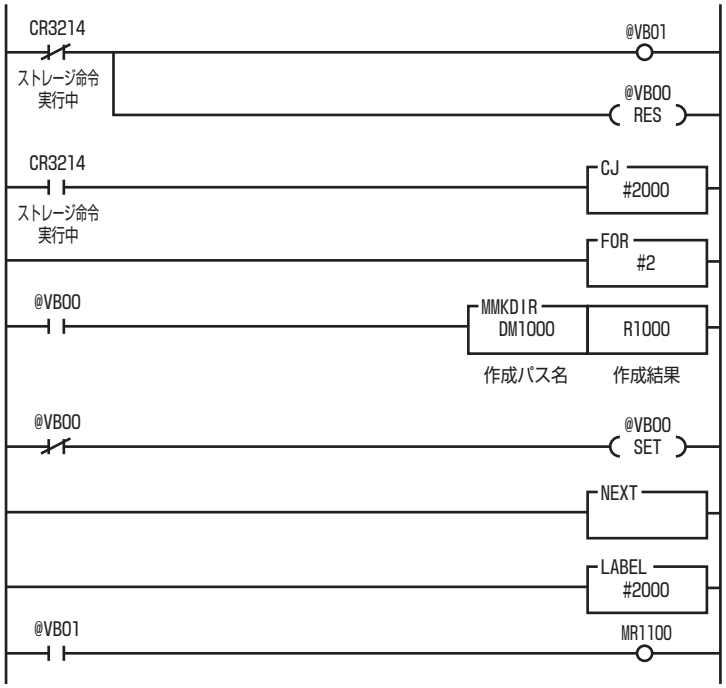
例

S	"フォルダA"	"フォルダA¥フォルダB"
フォルダ構造例		
注記	すでにフォルダが存在する場合は、何もせず正常終了し完了通知がONします。	親フォルダ(フォルダA)は事前に作成しておいてください。フォルダBと一緒に作成できません。

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「ストレージ命令」を参照してください。

●書式例

MR1100 = **MMKDIR** (DM1000.T, R1000)



●通知ビットデータ(D :完了通知、D +1:エラー通知)

項　目		内　容
D	完了通知	関数実行開始時にOFF、フォルダ作成終了時にONします。 関数が正常終了、異常終了のどちらでもONします。
D +1	エラー通知	関数が正常終了でOFF、異常終了でONします。

ポイント

- フォルダ作成には、数スキャンを要する場合があります。
- 通知ビットデータは関数実行完了後、END処理で結果が書き込まれます。
- 通知ビットデータに間接指定、インデックス修飾を指定した時の格納先は、実行条件の立ち上がり時点での格納先になります。
- ストレージ命令実行中(CR3214がONしている)は、MMKDIR関数は実行されません。
- 次のいずれかの条件が成立した場合、関数は実行されません。
  - S で指定した文字列に終端コード(NUL(00H))が無い場合。
  - S で指定した文字列が247文字(終端コード(NUL(00H))含む)を超えた場合。
  - S に間接指定でタイマ・カウンタを指定した場合。
  - S にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。
- 割り込みプログラムおよび初期化モジュール内では使用できません。

ワンポイント

MMKDIR関数は書き込みに数スキャンを要する場合があり、フォルダ作成処理中に作成パス名等の値を変更した場合、フォルダ作成結果は保証されません。
そのため、実行条件に特殊デバイスCR3214(ストレージ命令実行中)、あるいはフォルダ作成処理の作成結果(完了通知)を使用して、排他制御を行う必要があります。

スクリプト記述

```
IF MR1000 AND R5000 THEN
    RES ( R5000 )
    MR2000 = MMKDIR ( DM1000, R5000 )
END IF
IF MR2000 = OFF THEN
    SET ( R5000 )
END IF
```

動作説明

MR1000(DIR作成要求)がONで、かつR5000がON(前回DIR作成要求が完了)のとき、今回のフォルダ作成を実行します。
R5000をOFF(今回のDIR作成要求の完了判定のためにリセット)して、DM1000に格納されたパス名(文字列)で、メモリカードにフォルダを作成します。
作成結果がR5000(今回のDIR作成完了通知)とR5001(今回のDIR作成結果通知)に格納されます。
MMKDIR関数の起動に失敗(MR2000＝OFF)した場合、R5000をONして再度MMKDIR関数を起動させます。

MEMO

MWRIT
MREAD
MFREE
MFREEK
MMKDIR
MRMDIR
MDEL
MPRINT
MREADL
MCOPY
MMOV
MREN
MSTAT

MRMDIR ストレージ(メモリカード/CPUメモリ)フォルダ削除

戻り値※1 = **MRMDIR** (削除フォルダ名、削除結果格納対象)
ストレージフォルダ削除

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 削除フォルダ名※3※5	削除するフォルダのパス名の格納先、または削除フォルダパス名を指定します。(1～247文字(終端コードNUL含む)の範囲の文字列を指定します)	—	—	—	—	—	—	—	.T	※2 ○	○	○
D 削除結果格納対象※4	削除結果を通知する対象を指定します。	—	—	—	—	—	—	.B	—	—	○	—
R 戻り値	関数の起動に成功したらONし、失敗のときOFFになります。	—	—	—	—	—	—	.B	—	—	○	—

※1 戻り値の省略は可能です。省略した場合、演算結果はワークリレーに格納されます。
※2 定数で文字列を指定する場合は、" "で囲ってください。
※3 ビットデータを指定した場合、チャンネルの先頭を指定してください。
※4 ビットデータを指定した場合、連続する2ビットを占有します。ワードデータを指定した場合、連続する2ワードのそれぞれ最下位ビットを通知ビットデータとして使用します。最下位ビット以外は変化しません。
※5 T, C, CR, CTC, CTH, Zは使用できません。

ポイント **KV-N14□□/N24□□/N40□□/N60□□では使用できません。(《KV STUDIO》では変換エラーにならず、警告のみ表示されます。)**

動作説明

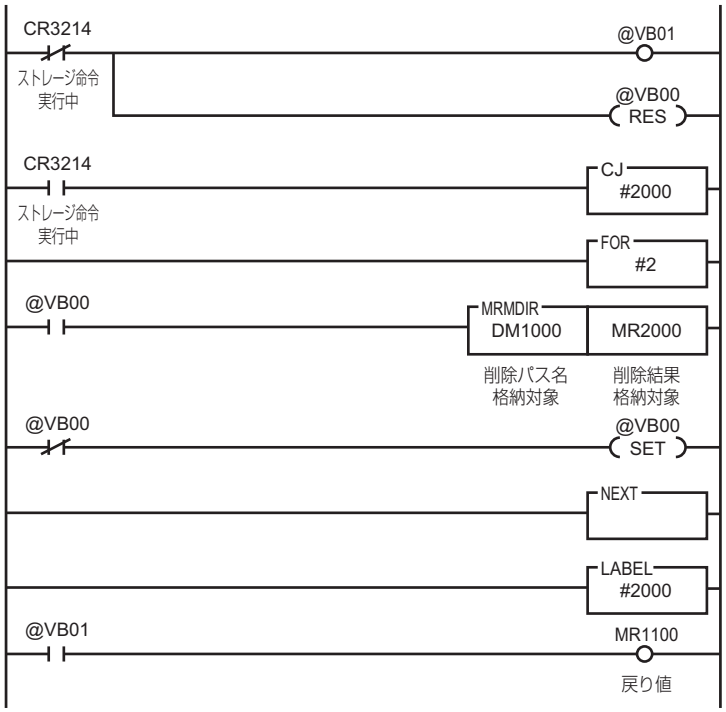
S で指定されたパス名(文字列)のフォルダをメモリカードまたはCPUメモリから削除します。
削除結果が D に格納されます。

- **S で指定したフォルダがメモリカードに存在しない場合**
  D +1がONし、CM2390に3(ファイルもしくはフォルダが存在しない)が格納されます。

- **S で指定したフォルダ内にファイルやフォルダが存在する場合**
  フォルダに存在する全てのファイル･フォルダを消去した後、指定したフォルダを消去します。
  「演算フラグについて(CR2009～CR2012)」(1-18ページ)
  詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ストレージ命令」を参照してください。

●書式例

MR1100 = **MRMDIR** (DM1000.T, MR2000)



●通知ビット格納(D :完了通知、D +1:エラー通知)

項　目		内　容
D	完了通知	関数実行開始時にOFF、フォルダ削除完了時にONします。 関数が正常終了、異常終了のどちらでもONします。
D +1	エラー通知	関数が正常終了でOFF、異常終了でONします。

ポイント

- フォルダ削除には、数スキャンを要する場合があります。
- 通知ビットデータは関数実行完了後、END処理で結果が書き込まれます。
- 通知ビットデータに間接指定、インデックス修飾を指定した時の格納先は、関数実行の立ち上がり時点での格納先になります。
- ストレージ命令実行中(CR3214がONしている)は、MRMDIR関数は実行されません。
- 次のいずれかの条件が成立した場合、関数は実行されません。
  - S で指定した文字列に終端コード(NUL(00H))が無い場合。
  - S で指定した文字列が247文字(終端コード(NUL(00H))含む)を超えた場合。
  - S に間接指定でタイマ・カウンタを指定した場合。
  - S にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。
- 割り込みプログラムおよび初期化モジュール内では使用できません。

MDEL ストレージ(メモリカード/CPUメモリ)ファイル削除

戻り値※1 = **MDEL** (削除ファイル名、削除結果格納対象)
ストレージファイル削除

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 削除ファイル名※3※5	削除するファイル名の格納先、または削除ファイル名を指定します。(1～247文字(終端コードNUL含む)の範囲の文字列を指定します)	—	—	—	—	—	—	—	.T	※2 ○	○	○
D 削除結果格納対象※4	削除結果を通知する対象を指定します。	—	—	—	—	—	—	.B	—	—	○	—
R 戻り値	関数の起動に成功したらONし、失敗のときOFFになります。	—	—	—	—	—	—	.B	—	—	○	—

※1 戻り値の省略は可能です。省略した場合、演算結果はワークリレーに格納されます。
※2 定数で文字列を指定する場合は、" "で囲ってください。
※3 ビットデータを指定した場合、チャンネルの先頭を指定してください。
※4 ビットデータを指定した場合、連続する2ビットを占有します。ワードデータを指定した場合、連続する2ワードのそれぞれ最下位ビットを通知ビットデータとして使用します。最下位ビット以外は変化しません。
※5 T, C, CR, CTC, CTH, Zは使用できません。

ポイント **KV-N14□□/N24□□/N40□□/N60□□では使用できません。(《KV STUDIO》では変換エラーにならず、警告のみ表示されます。)**

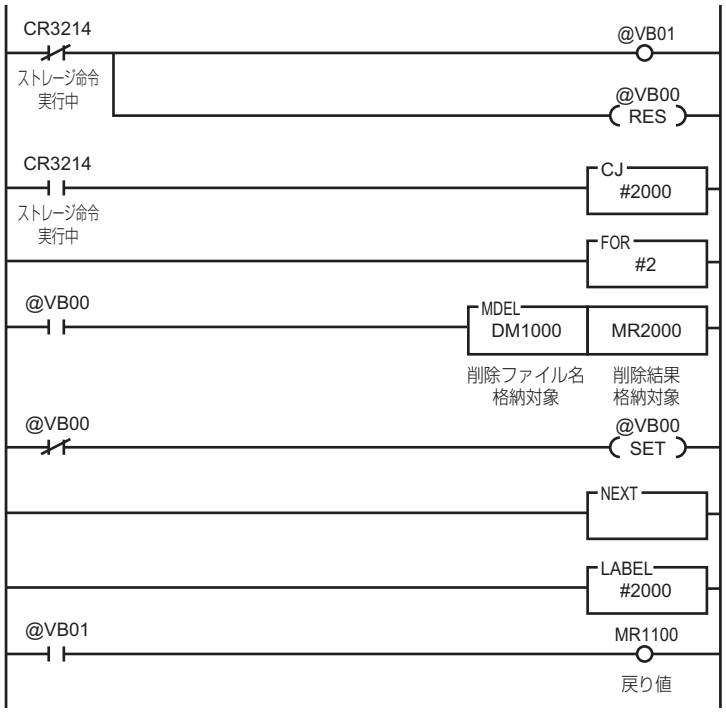
動作説明

S で指定されたファイル名(文字列)のファイルをメモリカードまたはCPUメモリから削除します。削除結果が D に格納されます。

- **S で指定したファイルがメモリカードに存在しない場合**
  D +1がONし、CM2390に3(ファイルもしくはフォルダが存在しない)が格納されます。
  「演算フラグについて(CR2009～CR2012)」(1-18ページ)
  詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ストレージ命令」を参照してください。

●書式例

MR1100 = **MDEL** (DM1000.T, MR2000)



●通知ビット格納(D :完了通知、D +1:エラー通知)

項　目		内　容
D	完了通知	関数実行開始時にOFF、ファイル削除完了時にONします。 関数が正常終了、異常終了のどちらでもONします。
D +1	エラー通知	関数が正常終了でOFF、異常終了でONします。

ポイント

- ファイル削除には、数スキャンを要する場合があります。
- 通知ビットデータは関数実行完了後、END処理で結果が書き込まれます。
- 通知ビットデータに間接指定、インデックス修飾を指定した時の格納先は、関数実行の立ち上がり時点での格納先になります。
- ストレージ命令実行中(CR3214がONしている)は、MDEL関数は実行されません。
- 次のいずれかの条件が成立した場合、関数は実行されません。
  - S で指定した文字列に終端コード(NUL(00H))が無い場合。
  - S で指定した文字列が247文字(終端コード(NUL(00H))含む)を超えた場合。
  - S に間接指定でタイマ・カウンタを指定した場合。
  - S にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。
- 割り込みプログラムおよび初期化モジュール内では使用できません。

MPRINT ストレージ(メモリカード/CPUメモリ)のファイルにテキスト書き込み

ストレージ文字列書き込み
戻り値※1 = **MPRINT** (保存ファイル名, 書き込み文字列, 書き込み結果格納対象)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 保存ファイル名※2	データを保存するファイル名の格納先、または保存ファイル名を指定します。(1～247文字(終端コードNUL含む)の範囲の文字列を指定します)	.U	.S	.D	.L	.F	.DF	.B	.T	○	○	—
S2 書き込み文字列※2	書き込む文字列を格納している先頭または、書き込み文字列を指定します。	.U	.S	.D	.L	.F	.DF	.B	.T	○	○	—
D 書き込み結果格納対象	書き込み結果を通知するビットデータを指定します。	—	—	—	—	—	—	.B	—	—	○	—
R 戻り値	関数の起動に成功したらONし、失敗のときOFFになります。	.U	.S	.D	.L	.F	.DF	—	—	—	○	—

※1 戻り値は省略できます。省略した場合、演算結果はワークデバイス/ワークリレーに格納されます。
※2 書き込み結果格納対象にビットデータを指定した場合、連続する2ビットが通知ビットとして使用されます。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 ビットデータを使用する場合はチャンネルの先頭を指定してください。
※5 T, C, CTC, CTH, Zは使用できません。

ポイント
- **MPRINT関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000 シリーズ CPU ユニット、CPU 機能バージョン 2.0 以降の KV-5500/5000/3000シリーズCPUユニット、KV-NC32Tでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)
- **KV-N14□□/N24□□/N40□□/N60□□では使用できません。(《KV STUDIO》では変換エラーにならず、警告のみ表示されます。)**

動作説明

S1 で指定されたファイル名(文字列)の末尾に、S2 で指定された文字列をメモリカードに書き込みます。
S1 名が存在しない場合、ファイルを新規作成します。

D に書き込み完了通知と異常終了のエラー通知が格納されます。
MPRINT関数が正常に起動できた場合、R をONし、失敗した場合はOFFします。

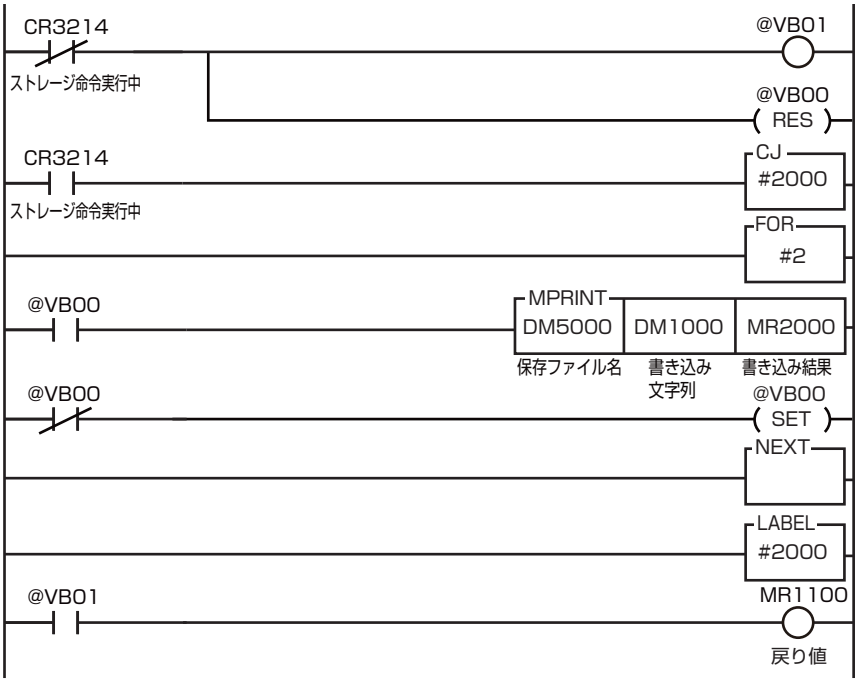
参考 D の通知ビットは次のような意味を持ちます。
(例) MR2000を指定した場合

通知ビット		ON(1, TRUE)	OFF(0, FALSE)
MR2000	完了通知	完了	未完了
MR2001	エラー通知	異常終了	正常終了

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「ストレージ命令」を参照してください。

●書式例

MR1100 = **MPRINT** (DM5000.T, DM1000.T, MR2000)



MWRIT
MREAD
MFREE
MFREEK
MMKDIR
MRMDIR
MDEL
MPRINT
MREADL
MCOPY
MMOV
MREN
MSTAT

MREADL ストレージ(メモリカード/CPUメモリ)のファイルから1行読み出し

戻り値※1 = **MREADL** (読み出しファイル名, 読み出し文字列格納先,
読み出しパラメータ, 読み出し結果格納対象)
ストレージ1行読み出し

引数/戻り値	説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
S 読み出しファイル名※6	データを読み出すファイル名の格納先、または読み出しファイル名を指定します。(1～247文字(終端コードNUL、フォルダ名を含む)の範囲の文字列を指定します)	—	—	—	—	—	—	—	.T	※4 ○	○	○
D1 読み出し文字列格納先※5	読み出した文字列を格納する先頭を指定します。	—	—	—	—	—	—	—	.T	—	○	—
D2 読み出しパラメータ※2	読み出しパラメータを指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
D3 読み出し結果格納対象※3	読み出し結果を通知するビットデータを指定します。	—	—	—	—	—	—	.B	—	—	○	—
R 戻り値	関数の起動に成功したらONし、失敗のときOFFになります。	—	—	—	—	—	—	.B	—	—	○	—

※1　戻り値は省略できます。省略した場合、演算結果はワークデバイス/ワークリレーに格納されます。
※2　読み出しパラメータは、連続する4ワードを占有します。
※3　読み出し結果格納対象は、連続する4ビットが通知ビットとして使用されます。
※4　定数で文字列を指定する場合は、" "で囲ってください。
※5　ビットデータを使用する場合はチャンネルの先頭を指定してください。
※6　T, C, CTC, CTH, Zは使用できません。

ポイント
- **MREADL関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV-NC32Tのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)
- **KV-N14□□/N24□□/N40□□/N60□□では使用できません。(《KV STUDIO》では変換エラーにならず、警告のみ表示されます。)**

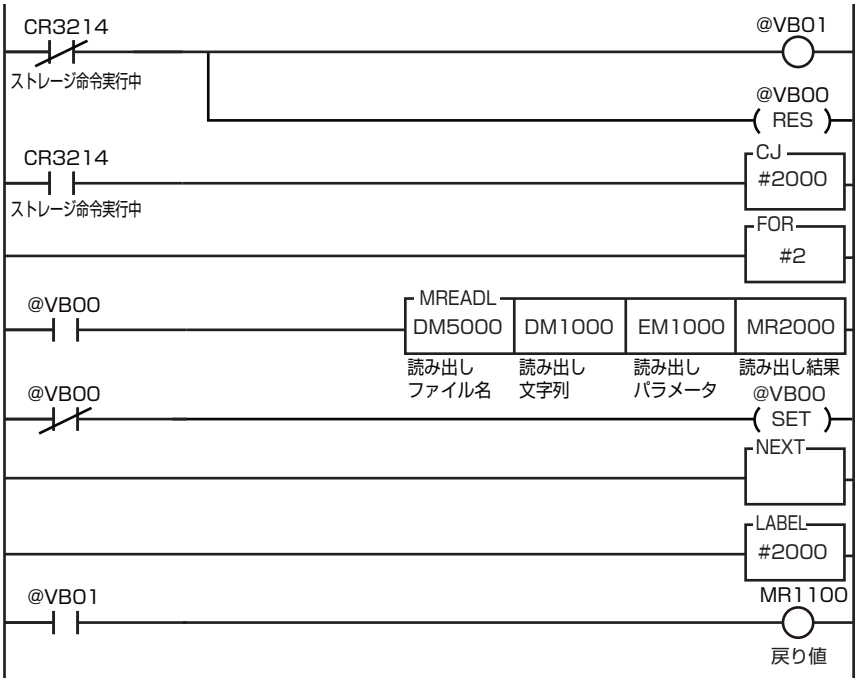
動作説明

S で指定されたファイル名(文字列)から1行読み出して D1 に読出します。
(行を表す文字コードは、0DH(CR)、0AH(LF)、0D0AH(CR+LF)です)
読み出す文字の最大バイトサイズは[D2 +2](読み出し上限バイト数)で設定します。
ファイル内の1行のテキストデータ長が[D2 +2]で指定した長さを超えている場合、
[D3 +3](改行文字なし通知)がONし、[D2 +2]バイトまでテキストデータを D1 に格納されます。

D3 に読み出し完了通知と異常終了アラーム通知が格納されます。
MREADL関数が正常に起動できた場合、 R をONし、失敗した場合はOFFします。
「演算フラグについて(CR2009～CR2012)」(1-18ページ)
D2 および D3 パラメータの詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「MREADL命令」を参照してください。

●書式例

MR1100 = **MREADL** (DM5000.T, DM1000.T, EM1000, MR2000)



MWRIT
MREAD
MFREE
MFREEK
MMKDIR
MRMDIR
MDEL
MPRINT
MREADL
MCOPY
MMOV
MREN
MSTAT

MCOPY ストレージ(メモリカード/CPUメモリ)のファイルをコピー

戻り値※1 = **MCOPY** (コピー元ファイル名･フォルダ名, コピー先ファイル名･フォルダ名, コピー形式パラメータ, コピー結果格納対象)
(ストレージファイルコピー)

引数/戻り値		説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
S1	コピー元ファイル名･フォルダ名※5※6	コピー元のファイル名･フォルダ名の格納先、またはファイル名･フォルダ名を指定します。(1～247文字(終端コードNUL、フォルダ名を含む)の範囲の文字列を指定します)	—	—	—	—	—	—	—	.T	※4 ○	○	○
S2	コピー先ファイル名･フォルダ名※5※6	コピー先のファイル･フォルダ名の格納先、またはファイル･フォルダ名を指定します。(1～247文字(終端コードNUL、フォルダ名を含む)の範囲の文字列を指定します)	—	—	—	—	—	—	—	.T	※4 ○	○	—
S3	コピー形式パラメータ※2	コピー形式のパラメータを指定します。	.U	.U	.U	.U	—	—	—	—	○	○	—
D	コピー結果格納対象※3	コピー結果を通知するビットデータを指定します。	—	—	—	—	—	—	.B	—	—	○	—
R	戻り値	関数の起動に成功したらONし、失敗のときOFFになります。	—	—	—	—	—	—	.B	—	—	○	—

※1 戻り値は省略できます。省略した場合、演算結果はワークデバイス/ワークリレーに格納されます。
※2 コピー形式パラメータは、1ワードを占有します。
※3 コピー結果格納デバイは、連続する2ビットが通知ビットとして使用されます。
※4 定数で文字列を指定する場合は、" "で囲ってください。
※5 ビットデータを使用する場合はチャンネルの先頭を指定してください。
※6 T, C, CTC, CTH, Zは使用できません。

ポイント
- **MCOPY関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV-NC32Tのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)
- **KV-N14□□/N24□□/N40□□/N60□□では使用できません。(《KV STUDIO》では変換エラーにならず、警告のみ表示されます。)**

動作説明

S1 で指定されたファイル名(文字列)を S2 で指定したファイルにコピーします。
コピー先に同名のファイル･フォルダが存在する場合、S3 に従って上書きするかエラーとするかを判断します。

コピー形式パラメータ値＝0の場合: 同名ファイル存在時、同名ファイルを上書きする
コピー形式パラメータ値＝1の場合: 同名ファイル存在時、上書きしないでエラー終了する

D にコピー完了通知と異常終了アラーム通知が格納されます。
MCOPY関数が正常に起動できた場合、R をONし、失敗した場合はOFFします。

S1 、S2 に指定できるファイル･フォルダ名の組み合わせ表

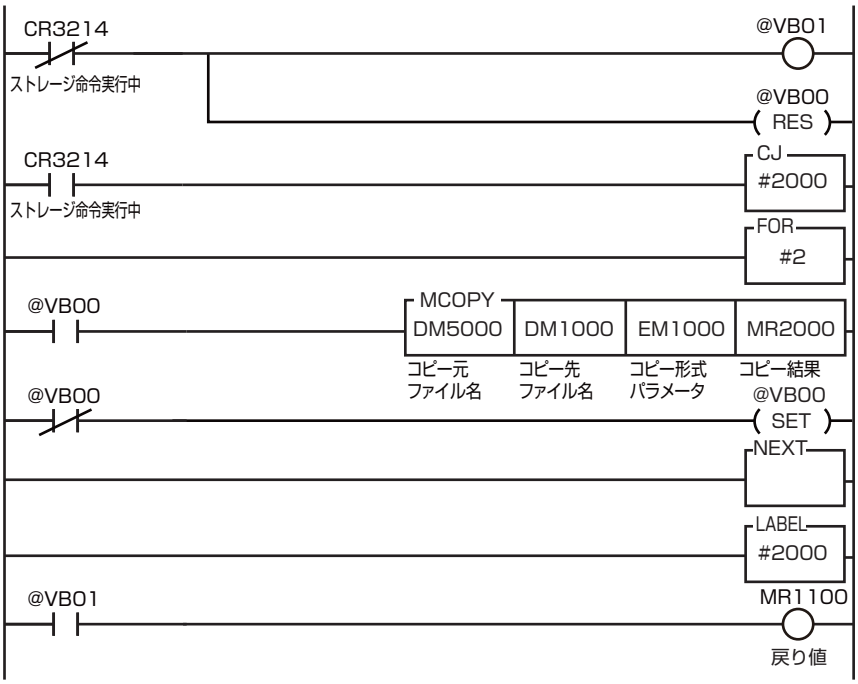
		S2	
		ファイル名	フォルダ名
S1	ファイル名	正常動作	正常動作
	フォルダ名	エラー発生	正常動作

「演算フラグについて（CR2009～CR2012）」（1-18ページ）

S3 パラメータおよび D コピー結果の詳細については、『KV-8000/7000/5000/3000/1000 シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「MCOPY命令」を参照してください。

●書式例

MR1100 = **MCOPY** (DM5000.T, DM1000.T, EM1000, MR2000)



MWRIT
MREAD
MFREE
MFREEK
MMKDIR
MRMDIR
MDEL
MPRINT
MREADL
MCOPY
MMOV
MREN
MSTAT

MMOV ストレージ(メモリカード/CPUメモリ)のファイルを移動

戻り値※1 = **MMOV** (移動元ファイル名･フォルダ名, 移動先ファイル名･フォルダ名, 移動形式パラメータ, 移動結果格納対象)
ストレージファイル移動

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	移動元ファイル名･フォルダ名※5※6	移動元のファイル名･フォルダ名の格納先、またはファイル名･フォルダ名を指定します。(1～247文字(終端コードNUL、フォルダ名を含む)の範囲の文字列を指定します)	—	—	—	—	—	—	—	.T	※4 ○	○	○
S2	移動先ファイル名･フォルダ名※5※6	移動先のファイル･フォルダ名の格納先、またはファイル･フォルダ名を指定します。(1～247文字(終端コードNUL、フォルダ名を含む)の範囲の文字列を指定します)	—	—	—	—	—	—	—	.T	※4 ○	○	—
S3	移動形式パラメータ※2	移動形式のパラメータを指定します。	.U	.U	.U	.U	—	—	—	—	○	○	—
D	移動結果格納対象※3	移動結果を通知するビットデータを指定します。	—	—	—	—	—	—	.B	—	—	○	—
R	戻り値	関数の起動に成功したらONし、失敗のときOFFになります。	—	—	—	—	—	—	.B	—	—	○	—

※1 戻り値は省略できます。省略した場合、演算結果はワークデバイス/ワークリレーに格納されます。
※2 移動形式パラメータは、1ワードを占有します。
※3 移動結果格納デバイスは、連続する2ビットが通知ビットとして使用されます。
※4 定数で文字列を指定する場合は、" "で囲ってください。
※5 ビットデータを使用する場合はチャンネルの先頭を指定してください。
※6 T, C, CTC, CTH, Zは使用できません。

ポイント
- **MMOV関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV-NC32Tでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)
- **KV-N14□□/N24□□/N40□□/N60□□では使用できません。(《KV STUDIO》では変換エラーにならず、警告のみ表示されます。)**

動作説明

S1 で指定されたファイル名(文字列)を S2 で指定したファイルに移動します。
移動先に同名のファイル･フォルダが存在する場合、S3 に従って上書きするかエラーとするかを判断します。

移動形式パラメータ値＝0の場合: 同名ファイル存在時、同名ファイルを上書きする
移動形式パラメータ値＝1の場合: 同名ファイル存在時、上書きしないでエラー終了する

D に移動完了通知と異常終了アラーム通知が格納されます。
MMOV関数が正常に起動できた場合、R をONし、失敗した場合はOFFします。

S1 、S2 に指定できるファイル･フォルダ名の組み合わせ表

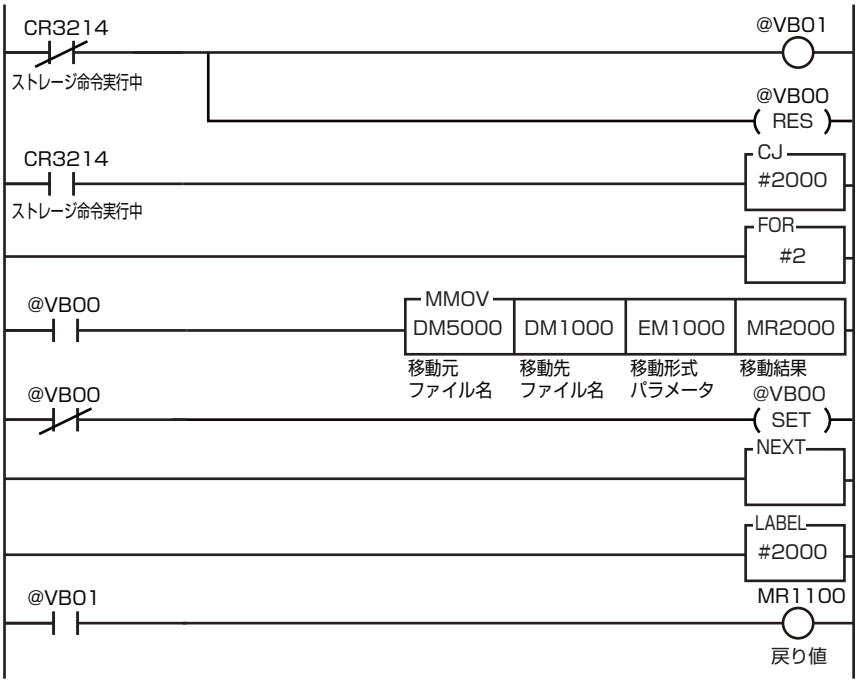
		S2	
		ファイル名	フォルダ名
S1	ファイル名	正常動作	正常動作
	フォルダ名	エラー発生	正常動作

「演算フラグについて（CR2009～CR2012）」（1-18ページ）

S3 パラメータおよび D 移動結果の詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「MMOV命令」を参照してください。

●書式例

MR1100 = **MMOV** (DM5000.T, DM1000.T, EM1000, MR2000)



MWRIT
MREAD
MFREE
MFREEK
MMKDIR
MRMDIR
MDEL
MPRINT
MREADL
MCOPY
MMOV
MREN
MSTAT

MREN ストレージ(メモリカード/CPUメモリ)のファイル名を変更

戻り値※1 = MREN (変更するファイル名･フォルダ名, 変更後のファイル名･フォルダ名, ファイル名変更結果格納対象)
ストレージファイル名変更

引数/戻り値	説明	認識する型 .U	.S	.D	.L	.F	.DF	.B	.T	定数 #/$	デバイス/変数	式
S1 変更するファイル名･フォルダ名※4※5	変更するフのファイル名･フォルダ名の格納先、またはファイル名･フォルダ名を指定します。(1～247文字(終端コードNUL、フォルダ名を含む)の範囲の文字列を指定します)	—	—	—	—	—	—	—	.T	※3 ○	○	○
S2 変更後のファイル名･フォルダ名※4※5	変更後のファイル･フォルダ名の格納先、またはファイル･フォルダ名を指定します。(1～247文字(終端コードNUL、フォルダ名を含む)の範囲の文字列を指定します)	—	—	—	—	—	—	—	.T	※3 ○	○	○
D ファイル名変更結果格納対象※2	ファイル名変更結果を通知するビットデータを指定します。	—	—	—	—	—	—	.B	—	—	○	—
R 戻り値	関数の起動に成功したらONし、失敗のときOFFになります。	—	—	—	—	—	—	.B	—	—	○	—

※1 戻り値は省略できます。省略した場合、演算結果はワークデバイス/ワークリレーに格納されます。
※2 移動結果格納デバイスは、連続する2ビットが通知ビットとして使用されます。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 ビットデータを使用する場合はチャンネルの先頭を指定してください。
※5 T, C, CTC, CTH, Zは使用できません。

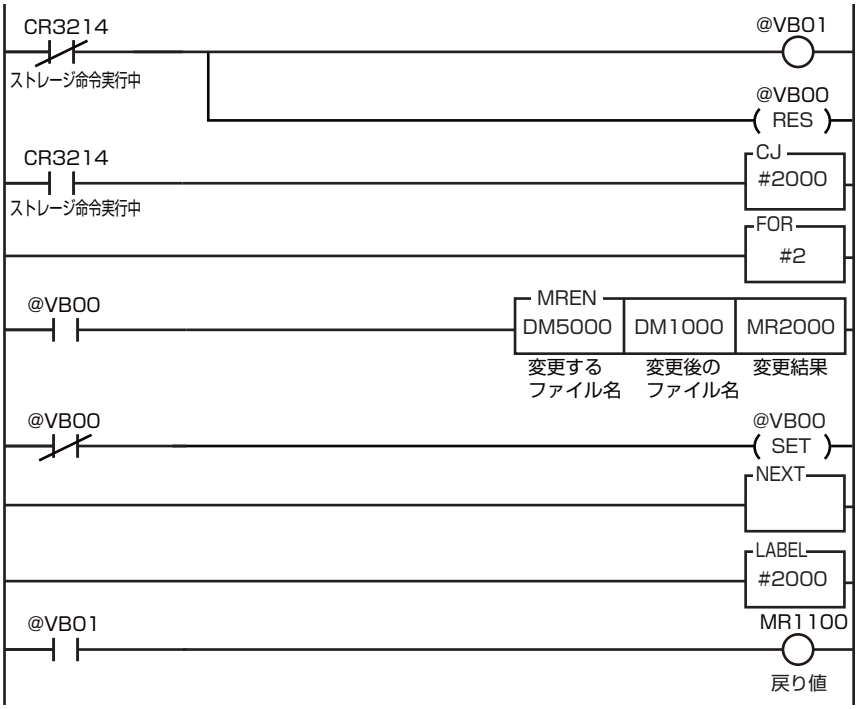
ポイント
- **MREN関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000 シリーズ CPU ユニット、CPU 機能バージョン 2.0 以降の KV-5500/5000/3000シリーズCPUユニット、KV-NC32Tでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)
- **KV-N14□□/N24□□/N40□□/N60□□では使用できません。(《KV STUDIO》では変換エラーにならず、警告のみ表示されます。)**

動作説明

S1 で指定されたファイル名･フォルダ名(文字列)を S2 で指定したファイル名･フォルダ名に変更します。
D に変更完了通知と異常終了アラーム通知が格納されます。
MREN関数が正常に起動できた場合、R をONし、失敗した場合はOFFします。
「演算フラグについて(CR2009～CR2012)」(1-18ページ)
動作の詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「MREN命令」を参照してください。

●書式例

MR1100 = **MREN** (DM5000.T, DM1000.T, MR2000)



MWRIT
MREAD
MFREE
MFREEK
MMKDIR
MRMDIR
MDEL
MPRINT
MREADL
MCOPY
MMOV
MREN
MSTAT

MSTAT ストレージ(メモリカード/CPUメモリ)のファイル状態取得

戻り値※1 = **MSTAT** (ファイル名, ファイル状態格納先, 取得結果格納対象)
ストレージファイル状態取得

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 ファイル名※5	状態を取得するファイル名の格納先、またはファイル名を指定します。(1～247文字(終端コードNUL、フォルダ名を含む)の範囲の文字列を指定します)	—	—	—	—	—	—	—	.T	※3 ○	○	○
D1 ファイル状態格納先※4	ファイル状態を格納する対象を指定します。	.U	.U	.U	.U	—	—	—	—	—	○	—
D2 取得結果格納対象※2	取得結果を通知するビットデータを指定します。	—	—	—	—	—	—	.B	—	—	○	—
R 戻り値	SEG関数は、.Uの型を持ちます。	—	—	—	—	—	—	.B	—	—	○	—

※1 戻り値は省略できます。省略した場合、演算結果はワークデバイス/ワークリレーに格納されます。
※2 読み出し結果格納対象は、連続する3ビットが通知ビットとして使用されます。
※3 定数で文字列を指定する場合は、" "で囲ってください。
※4 ファイル状態格納対象は、連続する10ワードを占有します。
※5 T, C, CTC, CTH, Zは使用できません。

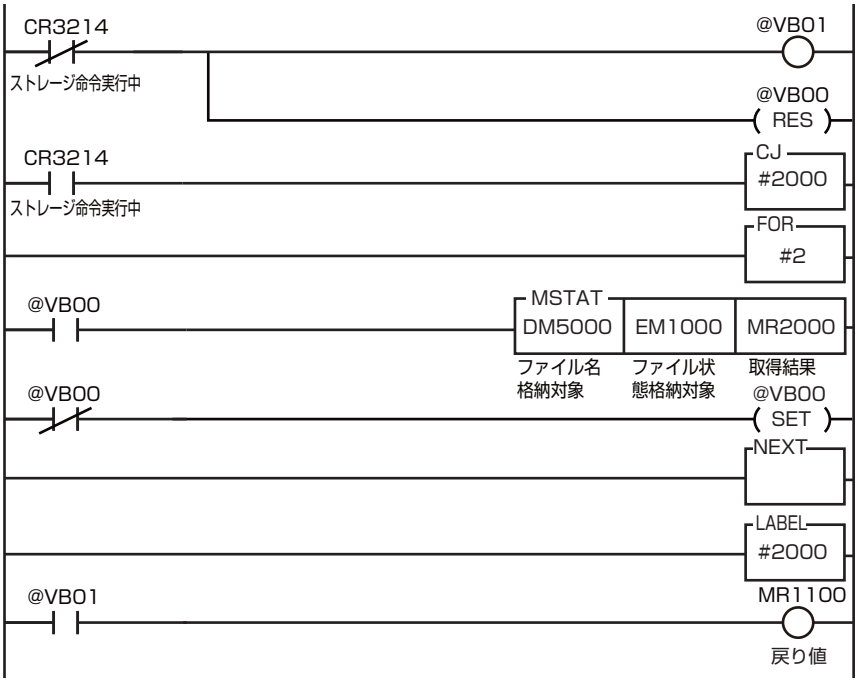
ポイント
- **MSTAT関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、KV-NC32Tでのみ指定できます。**
  「KV-5500/5000/3000 CPU機能バージョンについて」(4ページ)
- **KV-N14□□/N24□□/N40□□/N60□□では使用できません。(《KV STUDIO》では変換エラーにならず、警告のみ表示されます。)**

動作説明

S で指定されたファイルの状態を取得し、D1 に格納します。
D2 に取得完了通知と異常終了アラーム通知およびファイルの存在有無が格納されます。
MSTAT関数が正常に起動できた場合、R をONし、失敗した場合はOFFします。
「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ・KV Nanoシリーズ命令語リファレンスマニュアル』「MSTAT命令」を参照してください。

●書式例

MR1100 = **MSTAT** (DM5000.T, EM1000, MR2000)



MWRIT
MREAD
MFREE
MFREEK
MMKDIR
MRMDIR
MDEL
MPRINT
MREADL
MCOPY
MMOV
MREN
MSTAT

AWNUM アクセスウィンドウメッセージ1表示

ユーザメッセージ1表示
AWNUM (表示切換, AW書込データ)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	表示切換	アクセスウィンドウの表示切換を指定します。	—	—	—	—	—	—	.B	—	—	○	○
S2	AW 書込データ※1	アクセスウィンドウにユーザメッセージ1として表示する数値を格納した対象、またはユーザメッセージ1データを指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 ビットデータを指定した場合、連続する 16 ビットを占有します。チャンネルの先頭以外 (R002、R1012など)を指定した場合、次のチャンネルにまたがって16ビットを扱います。

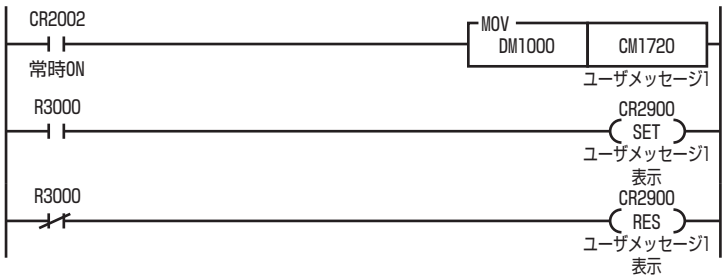
ポイント **KV-NC32Tにはアクセスウィンドウがありません。KV-NC32T、およびKV Nanoシリーズで増設アクセスウィンドウカセットを接続していない場合、エラーは発生せず、実行されません。(《KV STUDIO》では変換エラーになりません)。**

動作説明

S1 がONのとき、アクセスウィンドウにユーザメッセージ1として、S2 で指定された値を表示します。
S1 がOFFのとき、ユーザメッセージ1の表示は解除されます。

●書式例

AWNUM (R3000, DM1000)



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「アクセスウィンドウ命令」を参照してください。

参考 ユーザメッセージ1とユーザメッセージ2を同時に表示した場合、ユーザメッセージ1が優先して表示されます。

ユーザメッセージ表示の詳細については、『KV-5500/5000/3000シリーズ ユーザーズマニュアル』を参照してください。

ポイント

- 表示切換がONのときCR2900がセットされ、表示切換がOFFのときCR2900がリセットされます。また、S2で指定した値は表示切換ON時は常にCM01720にコピーされ、結果が即座にアクセスウィンドウに反映されます。
- 表示切換が ON 時にS2の値を変更すると、ユーザメッセージ 1 の表示内容も変更されます。
- 表示切換がONでも、アクセスウィンドウを操作するとメッセージ表示は解除されます(このときCR2900もリセットします)。再度AWNUM関数を呼び出せば、CR2900がONされます。
- AWNUM関数がプロジェクト中に複数存在していた場合、アクセスウィンドウの状態は最後に実行したAWNUM関数に従います。
- 割り込みプログラムおよび初期化モジュール内では使用できません。

AWMSG アクセスウィンドウメッセージ2表示

ユーザメッセージ2表示
AWMSG (表示切換, AW書込データ)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	表示切換	アクセスウィンドウの表示切換を指定します。	—	—	—	—	—	—	.B	—	—	○	○
S2	AW 書込データ※1※2	アクセスウィンドウにユーザメッセージ2として表示する文字列データを格納した対象、またはユーザメッセージ2データを指定します。	—	—	—	—	—	—	—	.T	○	○	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 アクセスウィンドウに表示できる文字列は24文字までです。
- ダブルコーテーション(“ ”)で囲んだ文字列定数を指定できます。
- ビットデータを使用する場合は、チャンネルの先頭を指定する必要があります。

※2 T, C, CR, CTC, CTH, Zは使用できません。

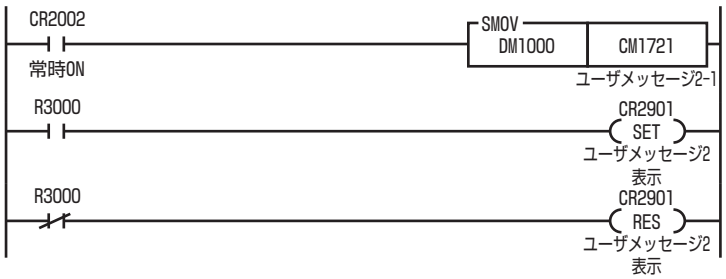
ポイント **KV-NC32Tにはアクセスウィンドウがありません。KV-NC32T、およびKV Nanoシリーズで増設アクセスウィンドウカセットを接続していない場合、エラーは発生せず、実行されません。(《KV STUDIO》では変換エラーになりません。)**

動作説明

S1 がONのとき、アクセスウィンドウにユーザメッセージ2として、S2 で指定された値を表示します。
S1 がOFFのとき、ユーザメッセージ2の表示は解除されます。

●書式例

AWMSG (R3000, DM1000.T)



「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「アクセスウィンドウ命令」を参照してください。

参考 ユーザメッセージ1とユーザメッセージ2を同時に表示した場合、ユーザメッセージ1が優先して表示されます。

ユーザメッセージ表示の詳細については、『KV-5500/5000/3000シリーズ ユーザーズマニュアル』を参照してください。

ポイント

- KV-5500/5000/3000/1000、KV Nanoシリーズのアクセスウィンドウには、全角文字は表示できません。
- 表示切換がONのときCR2901がセットされ、表示切換がOFFのときCR2901がリセットされます。また、S2を先頭に格納されている文字列は表示切換ON時は常にCM01721～CM01737にコピーされ、結果が即座にアクセスウィンドウに反映されます。
- 表示切換が ON 時にS2を先頭に格納されている文字列を変更すると、ユーザメッセージ2の表示内容も変更されます。
- 表示切換がONでも、アクセスウィンドウを操作するとメッセージ表示は解除されます(このときCR2901もリセットします)。再度AWMSG関数を呼び出せば、CR2900がONされます。
- AWMSG 関数がプロジェクト中に複数存在していた場合、アクセスウィンドウの状態は最後に実行したAWMSG関数に従います。
- アクセスウィンドウに表示できる文字列は、S2を先頭に格納されている文字列の先頭から24文字までの半角文字(アスキーコード 20h～7Eh、A1h～DFh)のみです。漢字などの2バイト文字は表示できません。
- 割り込みプログラムおよび初期化モジュール内では使用できません。
- 次のいずれかの条件が成立した場合、関数は実行されません。
  - S2にビットデータを指定したときに、チャンネルの先頭以外を指定した場合。
  - S2に間接指定でタイマ・カウンタを指定した場合。

AWSHOW ユーザメッセージ表示

ユーザメッセージ表示
AWSHOW (実行条件、表示するユーザメッセージID)

引数/戻り値		説明	認識する型								定数	デバイス/変数	式
			.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1	実行条件	実行条件がOFF→ONすると、アクセスウィンドウにユーザメッセージを表示します。	—	—	—	—	—	—	.B	—	—	○	○
S2	表示するユーザメッセージID	アクセスウィンドウに表示するユーザメッセージのIDを指定します。(1～63)	.U	.U	.U	.U	—	—	—	—	○	○	○
R	戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

動作説明

S1 がOFF→ONすると、S2 で指定したユーザメッセージを表示します。

参 考　ユーザメッセージを表示解除するには、アクセスウィンドウを操作するかAWHIDE命令を実行します。

「演算フラグについて(CR2009～CR2012)」(1-18ページ)
詳細については、『KV-8000/7000/5000/3000/1000 シリーズ･KV Nano シリーズ命令語リファレンスマニュアル』「アクセスウィンドウ命令」または使用しているCPUユニットのユーザーズマニュアルの「ユーザメッセージ」を参照してください。

ポイント
- **実行条件がON時に S2 の値を変更しても変化しません。**
- **実行条件の立ち上がりで指定された ID のユーザメッセージに対応する CR がセットされます。**
- **AWSHOW関数とAWHIDE関数が同時に実行された場合、後から実行された関数の動作に従います。**
- **AWSHOW関数がプロジェクト中に複数存在していた場合、アクセスウィンドウの状態は最初に実行したAWSHOW関数に従います。**
- **割り込みプログラムおよび初期化モジュール内では使用できません。**

AWHIDE ユーザメッセージ表示解除

ユーザメッセージ表示解除

AWHIDE (実行条件、表示解除するユーザメッセージID)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S1 実行条件	実行条件がOFF→ONすると、アクセスウィンドウにユーザメッセージを表示解除します。	—	—	—	—	—	—	.B	—	—	○	○
S2 表示解除するユーザメッセージID	表示解除するユーザメッセージのIDを指定します。(1～63)	.U	.U	.U	.U	—	—	—	—	○	○	○
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

動作説明

S1 がOFF→ONすると、S2 で指定したユーザメッセージを表示解除します。

「演算フラグについて(CR2009～CR2012)」(1-18ページ)

詳細については、『KV-8000/7000/5000/3000/1000 シリーズ･KV Nano シリーズ命令語リファレンスマニュアル』「アクセスウィンドウ命令」または使用しているCPUユニットのユーザーズマニュアルの「ユーザメッセージ」を参照してください。

ポイント

- **実行条件がON時に S2 の値を変更しても変化しません。**
- **実行条件の立ち上がりで指定された ID のユーザメッセージに対応する CR がリセットされます。**
- **AWSHOW関数とAWHIDE関数が同時に実行された場合、後から実行された関数の動作に従います。**
- **割り込みプログラムおよび初期化モジュール内では使用できません。**

RFSFRC フリーランカウンタリフレッシュ

フリーランカウンタリフレッシュ
RFSFRC ([実行条件,]※1)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 [ ]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)

動作説明

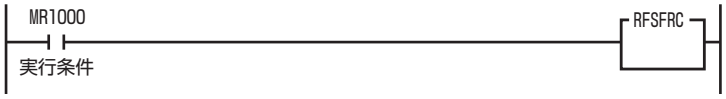
実行条件が成り立つ(TRUE)とき、フリーランカウンタを最新の値にリフレッシュします。
更新されるフリーランカウンタは以下のとおりです。

KV-8000/7500/7300	CM1030 ～CM1031：1msカウンタ
	CM1032 ～CM1033：10usカウンタ
	CM1034 ～CM1035：1usカウンタ
	CM1036 ～CM1037：0.1usカウンタ
KV-5500/5000/3000,KV Nano	CM708～CM709：1msカウンタ

詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「フリーランカウンタ命令」を参照してください

●書式例

RFSFRC (MR1000)



MEMO

SPRD センサパラメータ読み出し

SPRD (実行条件,ユニット番号,ノードアドレス,スロット番号,
パラメータ番号,完了コード)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 実行条件	実行条件に使用するビットデータを指定します。	—	—	—	—	—	—	.B	—	—	○	
n1 ユニット番号※1	ユニット番号(0～48、KV-8000/KV-7500/KV-5500本体は“0”)を指定します。	—	—	—	—	—	—	—	—	○	—	○
n2 ノードアドレス※3	ノードアドレス(1～256)、または格納先を指定します	.U	.U	.U	.U	—	—	—	—	○	○	○
n3 スロット番号※3	スロット番号、またはそれを格納した対象を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
n4 パラメータ番号※3	パラメータ番号、またはそれを格納した対象を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
D 完了コード※2※4※5	完了コードやパラメータの読出結果を格納するデバイスの対象の先頭を指定します。	.U	.U	.U	.U	.F	.DF	.B	.T	—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 $(16進数指定)は使用できません。KV Nanoシリーズのユニット番号は、0～15の範囲になります。
※2 CTC、CTH、Zは指定できません。
※3 ビットデータを指定した場合、連続した16ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16ビットを扱います。
※4 ビットデータを指定する場合は、チャンネルの先頭を指定してください。
※5 実行結果により格納されるワード数(4ワード以上)が変化します。

ポイント
- **SPRD関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、CPU機能バージョン2.0以降のKV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)
「KV Nanoシリーズ　CPU機能バージョンについて」(4ページ)

動作説明

S がONのとき、n1 番目のユニットに接続されたノードアドレス n2 、スロット番号 n3 のセンサの n4 で指定したパラメータを読み出し、完了時に D の0ビット目に完了ビット、D の1ビット目に実行失敗ビット、D ＋1に完了コード、D ＋2に詳細完了コード、D ＋3に配列要素数(配列でない場合は1)、D ＋4以降に読み出されたパラメータの内容が格納されます。

D ～：結果格納先

結果格納先			内容
D	0ビット	完了ビット	命令実行開始時にOFF、パラメータの読み出し完了時にONします。命令が正常完了、異常完了のどちらでもONします。
	1ビット	実行失敗ビット	命令の実行に失敗した時に完了ビットと同時にONします。命令実行開始時にOFFします。
	2～15ビット	システム予約	OFF(固定値)
D ＋1	完了コード		パラメータの読み出し完了時に完了コードが格納されます。(正常時は、0)
D ＋2	詳細完了コード		パラメータの読み出し完了時に詳細完了コードが格納されます。
D ＋3	配列要素数		読み出したパラメータの配列の要素数が格納されます。配列以外の場合は1、命令が異常完了した場合は、0が格納されます。
D ＋4～	読出結果		パラメータの値が格納されます。読み出すパラメータによって占有されるデータ数が変化します。

ポイント **「EtherNet/IP機器予約」されているセンサを指定した場合、実行失敗します。**

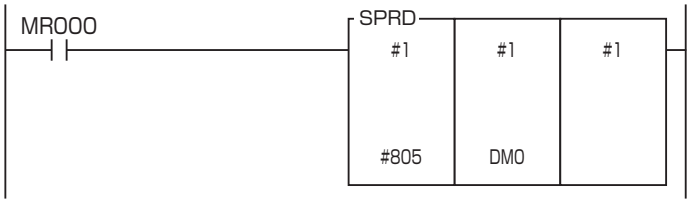
センサパラメータ読出命令は、実行条件の立ち上がりで命令が実行されると、次のスキャンで入力条件がOFFであっても処理が継続されます。
結果格納先の完了ビット、失敗ビットは、完了時にONし、実行開始時にOFFになります。
命令が実行されるとセンサ設定命令実行数に1が加算され、実行完了時に減算されます。
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「センサ設定命令」を参照してください。

●書式例

スクリプト記述 **SPRD** (MR0,1,1,1,805,DM0)

動作説明 MR000がONすると、1台目に接続されているユニットに接続されているノードアドレス1、スロット番号1のセンサのパラメータ番号805の内容を読み出して、DM0以降に格納します。

ラダー変換



SPWR　センサパラメータ書き込み

SPWR (実行条件,ユニット番号,ノードアドレス,スロット番号,
パラメータ番号,完了コード)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 実行条件	実行条件に使用するビットデータを指定します。	—	—	—	—	—	—	.B	—	—	○	
n1 ユニット番号※1	ユニット番号(0～48、KV-8000/KV-7500/KV-5500本体は“0”)を指定します。	—	—	—	—	—	—	—	—	○	—	○
n2 ノードアドレス※3	ノードアドレス(1～256)、または格納先を指定します	.U	.U	.U	.U	—	—	—	—	○	○	○
n3 スロット番号※3	スロット番号、またはそれを格納した対象を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
n4 パラメータ番号※3	パラメータ番号、またはそれを格納した対象を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
D 完了コード※2※4※5	完了コードやパラメータの読出結果を格納するデバイスの対象の先頭を指定します。	.U	.U	.U	.U	.F	.DF	.B	.T	—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1　$(16進数指定)は使用できません。KV Nanoシリーズのユニット番号は、0～15の範囲になります。
※2　CTC、CTH、Zは指定できません。
※3　ビットデータを指定した場合、連続した16ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16ビットを扱います。
※4　ビットデータを指定する場合は、チャンネルの先頭を指定してください。
※5　実行結果により格納されるワード数(4ワード以上)が変化します。

ポイント
- **SPWR関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、CPU機能バージョン2.0以降のKV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)
「KV Nanoシリーズ　CPU機能バージョンについて」(4ページ)

動作説明

S がONのとき、n1 番目のユニットに接続されたノードアレス n2 、スロット番号 n3 のセンサの n4 で指定したパラメータに、D ＋4以降で指定した設定値を書き込み、完了時に D の0ビット目に完了ビット、D の1ビット目に実行失敗ビット、D ＋1に完了コード、D ＋2に詳細完了コードが格納されます。

D ～：結果格納先

結果格納先			内容
D	0ビット	完了ビット	命令実行開始時にOFF、パラメータ書き込み完了時にONします。命令が正常完了、異常完了のどちらでもONします。
	1ビット	実行失敗ビット	命令の実行に失敗した時に完了ビットと同時にONします。命令実行開始時にOFFします。
	2～15ビット	システム予約	OFF(固定値)
D ＋1	完了コード		パラメータの書き込み完了時に完了コードが格納されます。(正常時は、0)
D ＋2	詳細完了コード		パラメータの書き込み完了時に詳細完了コードが格納されます。
D ＋3	システム予約		0(固定値)
D ＋4～	設定値		設定値を格納します。書き込むパラメータによって占有するデータ数が変化します。

ポイント **「EtherNet/IP機器予約」されているセンサ(アダプタ)を指定した場合、実行失敗します。**

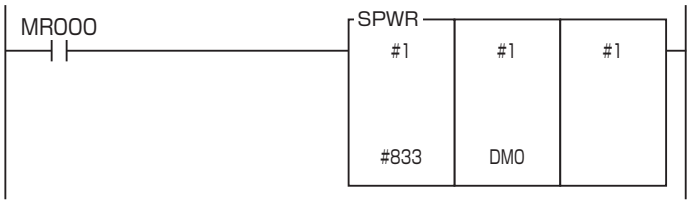
センサパラメータ書込命令は、実行条件の立ち上がりで命令が実行され、次のスキャンで入力条件がOFFであっても処理が継続されます。
結果格納先の完了ビット、失敗ビットは、完了時にONし、実行開始時にOFFになります。
命令が実行されるとセンサ設定命令実行数に1が加算され、実行完了時に減算されます。
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「センサ設定命令」を参照してください。

●書式例

スクリプト記述 **SPWR** (MR0,1,1,1,833,DM0)

動作説明 MR000がONすると、ユニット番号1のKV-EP21Vに接続されているノードアドレス1、スロット番号1のセンサのパラメータ番号833(設定値)へDM4以降で指定した設定値を書き込み、結果をDM0以降に格納します。

ラダー変換



SSVC センササービス実行

SSVC (実行条件,ユニット番号,ノードアドレス,スロット番号,
パラメータ番号,完了コード)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 実行条件	実行条件に使用するビットデータを指定します。	—	—	—	—	—	—	.B	—	—	○	
n1 ユニット番号※1	ユニット番号(0～48、KV-8000/KV-7500/KV-5500本体は“0”)を指定します。	—	—	—	—	—	—	—	—	○	—	○
n2 ノードアドレス※3	ノードアドレス(1～256)、または格納先を指定します	.U	.U	.U	.U	—	—	—	—	○	○	○
n3 スロット番号※3	スロット番号、またはそれを格納した対象を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
n4 パラメータ番号※3	パラメータ番号、またはそれを格納した対象を指定します。	.U	.U	.U	.U	—	—	—	—	○	○	○
D 完了コード※2※4※5	完了コードやパラメータの読出結果を格納するデバイスの対象の先頭を指定します。	.U	.U	.U	.U	.F	.DF	.B	.T	—	○	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

※1 $(16進数指定)は使用できません。
※2 CTC、CTH、Zは指定できません。
※3 ビットデータを指定した場合、連続した16ビットを扱います。チャンネルの先頭以外(R002, R1012など)を指定した場合、次のチャンネルにまたがって16ビットを扱います。
※4 ビットデータを指定する場合は、チャンネルの先頭を指定してください。
※5 指定するサービスにより最大6ワードを使用します。

ポイント
- **SSVC関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、CPU機能バージョン2.0以降のKV Nanoシリーズ基本ユニットでのみ指定できます。**

「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)
「KV Nanoシリーズ　CPU機能バージョンについて」(4ページ)

動作説明

S がONのとき、n1 番目のユニットに接続されたノードアレス n2 、スロット番号 n3 のセンサの n4 で指定したサービスを実行し、完了時に D の0ビット目に完了ビット、D の1ビット目に実行失敗ビット、D +1に完了コード、D +2に詳細完了コードが格納されます。

D ～：結果格納先

結果格納先			内容
D	0ビット	完了ビット	命令実行開始時にOFF、サービスの実行完了時にONします。 命令が正常完了、異常完了のどちらでもONします。
	1ビット	実行失敗ビット	命令の実行に失敗した時に完了ビットと同時にONします。 命令実行開始時にOFFします。
	2～15ビット	システム予約	OFF(固定値)
D ＋1	完了コード		サービスの実行完了時に完了コードが格納されます。(正常時は、0)
D ＋2	詳細完了コード		サービスの実行完了時に詳細完了コードが格納されます。
D ＋3	システム予約		0(固定値)
D ＋4 D ＋5	送信データ		実行するサービスに合わせて送信データ(最大2ワード)を指定します。

ポイント **「EtherNet/IP機器予約」されているセンサを指定した場合、実行失敗します。**

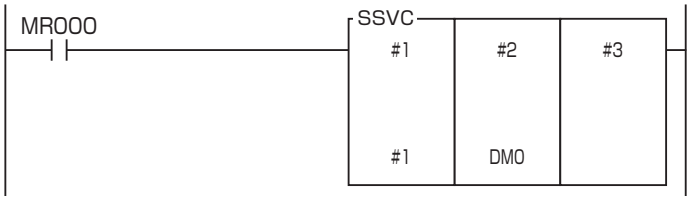
センササービス実行命令は、実行条件の立ち上がりで命令が実行され、次のスキャンで入力条件がOFFであっても処理が継続されます。
結果格納先の完了ビット、失敗ビットは、完了時にONし、実行開始時にOFFになります。
命令が実行されるとセンサ設定命令実行数に1が加算され、実行完了時に減算されます。
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「センサ設定命令」を参照してください。

●書式例

スクリプト記述　SSVC (MR0,1,2,3,1,DM0)

動作説明　MR000がONすると、ユニット番号1のKV-EP21Vに接続されているノードアドレス2、スロット番号3のセンサのサービス番号1のサービスを実行し、結果をDM0以降に格納します。

ラダー変換



RFSCI　サイクリック通信入力リフレッシュ

RFSCI ([実行条件,]※1 先頭, データ数)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 先頭※2	サイクリック通信入力データのリフレッシュをおこなう先頭を指定します。	.U	.S	.D	.L	.F	—	—	—	—	○	—
n データ数	リフレッシュをおこなうデータ数を指定します。	.U	.U	.U	.U	—	—	—	—	○	—	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

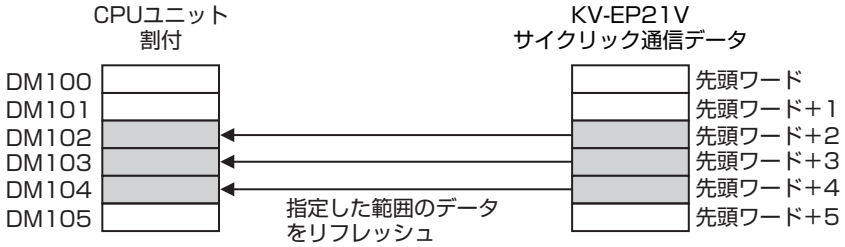
※1　[　]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2　ビットデータを指定する場合は、チャンネルの先頭を指定してください。

ポイント
- **RFSCI関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、CPU機能バージョン2.0以降のKV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)
  「KV Nanoシリーズ　CPU機能バージョンについて」(4ページ)

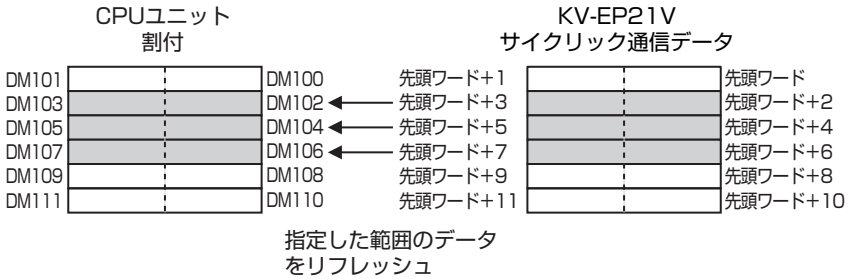
動作説明

実行条件がONのとき、S に割り付けられているデータから、n で指定したデータ数のサイクリック通信入力データをリフレッシュします。(先頭が、32ビットデータのとき、リフレッシュされるワード数は、データ数×2になります。)

- **S :先頭が16ビット(.U/.S)の場合**
  例)先頭をDM102(.U)、データ数を3で指定した場合



- **S :先頭が32ビット(.D/.L/.F)の場合**
  例)先頭をDM102.D、データ数を3で指定した場合



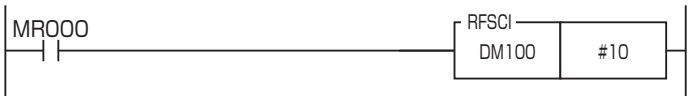
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「サイクリック通信リフレッシュ命令」を参照してください。

●書式例

スクリプト記述　**RFSCI** (MR0,DM100.U,10)

動作説明　MR000がONすると、データメモリDM100以降に割り付けられている10ワード分のサイクリック通信入力データをリフレッシュします。

ラダー変換



RFSCO　サイクリック通信出力リフレッシュ

RFSCO ([実行条件,]※1 先頭, データ数)

引数/戻り値	説明	認識する型								定数	デバイス/変数	式
		.U	.S	.D	.L	.F	.DF	.B	.T	#/$		
S 先頭※2	サイクリック通信出力データのリフレッシュをおこなう先頭を指定します。	.U	.S	.D	.L	.F	—	—	—	—	○	—
n データ数	リフレッシュをおこなうデータ数を指定します。	.U	.U	.U	.U	—	—	—	—	○	—	—
R 戻り値	なし	—	—	—	—	—	—	—	—	—	—	—

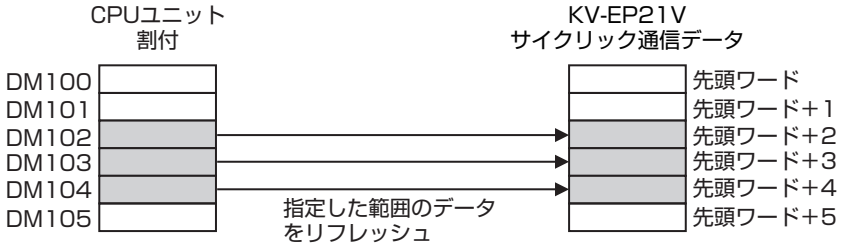
※1　[　]は省略可能です。(実行条件を省略した場合、常時(毎スキャン)実行します)
※2　ビットデータを指定する場合は、チャンネルの先頭を指定してください。

ポイント
- **RFSCO関数は、KV STUDIO Ver.6.00以降のバージョンでのみ使用できます。**
- **KV-8000/7000シリーズCPUユニット、CPU機能バージョン2.0以降のKV-5500/5000/3000シリーズCPUユニット、CPU機能バージョン2.0以降のKV Nanoシリーズ基本ユニットでのみ指定できます。**
  「KV-5500/5000/3000　CPU機能バージョンについて」(4ページ)
  「KV Nanoシリーズ　CPU機能バージョンについて」(4ページ)

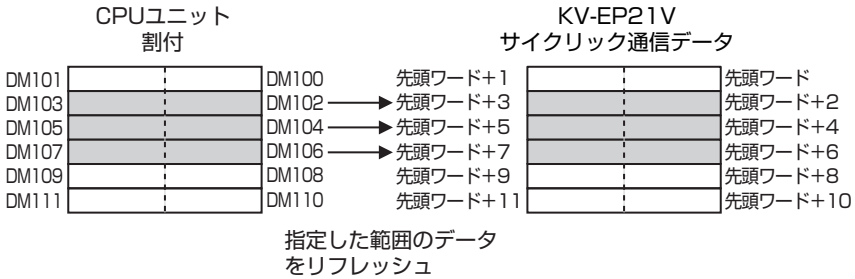
動作説明

実行条件がONのとき、D に割り付けられているデータから、n で指定したデータ数のサイクリック通信出力データをリフレッシュします。(先頭が、32ビットデータのとき、リフレッシュされるワード数は、データ数×2になります。)

- **D :先頭が16ビット(.U/.S)の場合**
  例)先頭をDM102(.U)、データ数を3で指定した場合



- **D :先頭が32ビット(.D/.L/.F)の場合**
  例)先頭をDM102.D、データ数を3で指定した場合



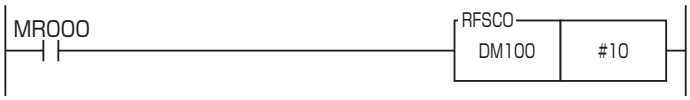
詳細については、『KV-8000/7000/5000/3000/1000シリーズ･KV Nanoシリーズ命令語リファレンスマニュアル』「サイクリック通信リフレッシュ命令」を参照してください。

●書式例

スクリプト記述　**RFSCO** (MR0,DM100.U,10)

動作説明　MR000がONの時、データメモリDM100以降に割り付けられている10ワード分のサイクリック通信出力データをリフレッシュします。

ラダー変換



MEMO

付　録

CR/CM一覧 付-2
ASCIIコード表 付-53
使用できない文字一覧 付-54
使用できない関数一覧 付-58
索　引 付-59
関数索引 付-64

CR/CM一覧

コントロールリレー CR KV-8000の場合

コントロールリレー（CR0000～CR7915）は、CPUユニットの動作状態の確認、各内蔵機能の設定やモニタなどができます。
システム予約されているコントロールリレーは使用できません。

CR	属性 R：読出専用 空欄：読書可能	機　能	
CR0000		ロギング・トレース機能（ID0）	ロギング・トレース 実行（ON：許可、OFF：禁止）
CR0001	R		ロギング・トレース 動作中
CR0002	R		ロギング・トレース ファイル書き込み終了
CR0003	R		ロギング バッファ50%未満警告
CR0004	R		ロギング バッファオーバーフロー
CR0005	R		ロギング 実行中RUN中書き込みあり
CR0006～CR0007	－		システム予約
CR0008	R		ロギング・トレース エラー
CR0009	R		ロギング・トレース メモリ容量なしエラー
CR0010	R		ロギング・トレース デバイス・トリガ設定異常
CR0011	R		トレース ファイル保存トリガモニタ
CR0012	R		トレース データ取得完了
CR0013	R		ロギング・トレース 設定書き込みあり
CR0014～CR0015	－		システム予約
CR0100～CR0915		ロギング・トレース機能（ID1～ID9）	
CR1000～CR1015		リアルタイムチャートモニタ（ID10）	
CR1100～CR2001	－	システム予約	
CR2002	R	常時ON	
CR2003	R	常時OFF	
CR2004	R	10msクロックパルス（duty50%）	
CR2005	R	100msクロックパルス（duty50%）	
CR2006	R	1sクロックパルス（duty50%）	
CR2007	R	運転開始時１スキャンのみOFF	
CR2008	R	運転開始時１スキャンのみON	
CR2009		演算結果が負またはオーバーフローのときON	
CR2010	R	演算結果がゼロのときON	
CR2011	R	演算結果が正のときON	
CR2012	R	演算実行エラーのときON	
CR2013～CR2302	－	システム予約	
CR2303		一定スキャンタイム運転（ON：有効、OFF：無効）	
CR2304	R	スキャンタイム設定（（一定スキャンタイム運転、END処理時間設定）オーバー時1スキャンのみON）	
CR2305	－	システム予約	
CR2306	R	バックアップ電池異常（ON:異常、OFF：正常）	
CR2307		スキャンタイム最小値最大値のクリア	
CR2308		END処理時間設定 （ON：有効、OFF:無効）（CR2303と同時にONした場合は、CR2303が優先）	
CR2309～CR2310	－	システム予約	
CR2311	R	RTC電圧低下エラー（ON：エラー発生中、OFF：正常）	
CR2312	－	システム予約	
CR2313		VTセンサアプリケーション画面強制切換（切換する：ON、切換しない：OFF）	
CR2314	R	プロジェクトパスワード認証禁止状態（ON：認証禁止、OFF:認証可能）	
CR2315～CR2813	－	システム予約	
CR2814		DASC、FASC命令（全サフィックス）実行時にゼロサプレスを（ON:する、OFF：しない）	
CR2815		DASC.S、DASC.L、FASC実行時に＋の符号を省略（ON:する、OFF：しない）	
CR2900～CR3005	－	システム予約	
CR3100	R	ユニット間同期リフレッシュ時間超過	
CR3101～CR3115	－	システム予約	
CR3200		CPUメモリ　保存処理要求	
CR3201	R	CPUメモリ　保存処理中	

CR	属性 R:読出専用 空欄:読書可能	機　能
CR3202	R	CPUメモリ　保存処理完了
CR3203	R	CPUメモリ　保存処理エラー
CR3204～CR3206	－	システム予約
CR3207		命令語 RAMモード動作 （CPUメモリに対するファイルレジスタ命令･ストレージ命令のRAMモード有効化 （ON:有効、OFF:無効））
CR3208～CR3209	－	システム予約
CR3210	R	任意ロギング･トレース動作中
CR3211～CR3213	－	システム予約
CR3214	R	ストレージ命令実行中
CR3215	－	システム予約
CR3300～CR3415		アラームリレー
CR3500	R	アラーム動作リレー（ON : CR3300～CR3415のいずれかがON）
CR3501		アラームリレーがON→OFFのときに履歴を記録（ON : 残す、OFF : 残さない）
CR3502		OFF→ON時にアラームリレー履歴クリア
CR3503～CR3512	－	システム予約
CR3513	R	ロギング内部バッファオーバフロー
CR3514		HKEY命令　複数キー禁止
CR3515	R	HKEY命令　スキャン終了
CR3600～CR3615	R	HKEYの情報格納エリア
CR3700～CR3901	－	システム予約
CR3902	R	重度エラー履歴が最大件数
CR3903	R	軽度エラー履歴が最大件数
CR3904	R	RUN/PRG切換SW状態（ON:RUN側/OFF:PRG側）
CR3905	R	重度エラー発生中
CR3906	R	軽度エラー発生中
CR3907～CR3908	－	システム予約
CR3909		現在のエラークリア（ラダープログラム中でSETしてください）
CR3910		OFF→ON時にすべてのエラー、イベント履歴をクリア
CR3911～CR4000	－	システム予約
CR4001～CR4300	R	拡張ユニット有無（拡張ユニット1～48）
CR4301～CR4315	－	システム予約
CR4400	R	AW　上キー押下中
CR4401	R	AW　下キー押下中
CR4402	R	AW　左キー押下中
CR4403	R	AW　右キー押下中
CR4404	R	AW　MENUキー押下中
CR4405	R	AW　ENTキー押下中
CR4406～CR4414	－	システム予約
CR4415	R	AW　ユーザメッセージ表示中
CR4500		AW　1行目ブリンク
CR4501		AW　2行目ブリンク
CR4502		AW　3行目ブリンク
CR4503		AW　4行目ブリンク
CR4504～CR4600	－	システム予約
CR4601～CR4915		AW ユーザメッセージ1～63
CR5000～CR5315	－	システム予約
CR5400		メモリカードマウント要求
CR5401		メモリカードアンマウント要求
CR5402	R	メモリカード使用中
CR5403	R	メモリカード認識完了
CR5404	R	メモリカードあり
CR5405	R	メモリカードライトプロテクト
CR5406～CR5407	－	システム予約
CR5408		CPUメモリマウント要求
CR5409		CPUメモリアンマウント要求
CR5410	R	CPUメモリ使用中

CR	属性 R:読出専用 空欄:読書可能	機　能	
CR5411	R	CPUメモリ認識完了	
CR5412～CR4915	―	システム予約	
CR5500		プロジェクトロード実行要求	
CR5501	―	システム予約	
CR5502		プロジェクトセーブ実行要求	
CR5503	―	システム予約	
CR5504	R	プロジェクトロード実行完了	
CR5505	R	プロジェクトロード実行失敗	
CR5506	R	プロジェクトセーブ実行完了	
CR5507	R	プロジェクトセーブ実行失敗	
R5508		レシピロード実行要求	
R5509	―	システム予約	
R5510	R	レシピロード実行完了	
R5511	R	レシピロード実行失敗	
R5512		レシピセーブ実行要求	
R5513	―	システム予約	
R5514	R	レシピセーブ実行完了	
R5515	R	レシピセーブ実行失敗	
CR5600		運転記録機能(ID0)	運転記録0 保存要求
CR5601	R		運転記録0 動作中
CR5602	R		運転記録0 保存トリガモニタ
CR5603	R		運転記録0 保存中
CR5604	R		運転記録0 収集中
CR5605	R		運転記録0 設定異常
CR5606	―		システム予約
DR5607	R		運転記録0 保存完了
CR5608	R		運転記録0 保存エラー
CR5609	R		デバイス記録0 バッファ不足
CR5610～ CR5613	―		システム予約
CR5614			運転記録0 バッファ領域全保存モード
CR5615	―		システム予約
CR5700～ CR5919		運転記録機能(ID1～ID3)	
CR6000～ CR7915	―	システム予約	

コントロールメモリ CM KV-8000の場合

コントロールメモリ（CM0000～CM7599）は、CPUユニットの動作状態の確認、各内蔵機能の設定やモニタなどができます。
システム予約されているコントロールメモリは使用できません。

CM	属性 R：読出専用 空欄：読書可能	機　能	保持 ○：保持
CM0000～CM0699	－	システム予約	
CM0700	R	RTCからの読み出し（年）	
CM0701	R	RTCからの読み出し（月）	
CM0702	R	RTCからの読み出し（日）	
CM0703	R	RTCからの読み出し（時）	
CM0704	R	RTCからの読み出し（分）	
CM0705	R	RTCからの読み出し（秒）	
CM0706	R	RTCからの読み出し（曜日）	
CM0707～CM0709	－	システム予約	
CM0710	R	ファイルレジスタのカレントバンク番号	
CM0711～CM0899	－	システム予約	
CM900	R	CPU機能バージョン	
CM901～CM906	－	システム予約	
CM907	R	プロジェクトバージョン	
CM908～CM999	－	システム予約	
CM1000～CM1001	R	スキャンタイム測定値（1μs）（32ビット）	
CM1002～CM1003	R	END処理時間測定値（1μs単位）（32ビット）	
CM1004～CM1005	R	スキャンタイム最小値（1μs単位）（32ビット）	
CM1006～CM1007	R	スキャンタイム最大値（1μs単位）（32ビット）	
CM1008～CM1009		一定スキャンタイム運転時設定値（1μs単位）（32ビット）	
CM1010～CM1011		END処理時間設定値（1μs単位）（32ビット）	
CM1012～CM1013	R	設定超過時スキャンタイム（1μs単位）（32ビット）	
CM1014～CM1015	R	END処理時間超過値（1μs単位）（32ビット）	
CM1016～CM1029	－	システム予約	
CM1030～CM1031	R	32ビット1msフリーランカウンタ	
CM1032～CM1033	R	32ビット10μsフリーランカウンタ	
CM1034～CM1035	R	32ビット1μsフリーランカウンタ	
CM1036～CM1037	R	32ビット0.1μsフリーランカウンタ	
CM1038～CM1499	－	システム予約	
CM1500～CM1509	R	ロギング･トレース（ID0～ID9）ファイル保存カウンタ	
CM1510～CM1629	－	システム予約	
CM1630～CM1639	R	ロギング･トレース リングバッファの空き容量最小 （ID0～ID9）（KByte単位）	
CM1640～CM1649	－	システム予約	
CM1650	R	センサ設定命令実行数	
CM1651～CM1657	－	システム予約	
CM1658	R	プロジェクトパスワード連続認証失敗回数	○
CM1659	R	プロジェクトパスワード累積認証失敗回数	○
CM1660～CM1669	R	ロギング･トレース リングバッファの空き容量（ID0~ID9）（KByte単位）	
CM1670～CM1679	－	システム予約	
CM1680～CM1689	R	ロギング･トレース ファイル保存カウンタ （ID0～ID9）（メモリカード用）	○
CM1690～CM1699	－	システム予約	
CM1700～CM1701		デジタルトリマ0上限値（32ビット）	
CM1702～CM1703		デジタルトリマ1上限値（32ビット）	
CM1704～CM1705		デジタルトリマ2上限値（32ビット）	
CM1706～CM1707		デジタルトリマ3上限値（32ビット）	
CM1708～CM1709		デジタルトリマ4上限値（32ビット）	
CM1710～CM1711		デジタルトリマ5上限値（32ビット）	
CM1712～CM1713		デジタルトリマ6上限値（32ビット）	

CM	属性 R:読出専用 空欄:読書可能	機　能		保持 ○:保持
CM1714～CM1715		デジタルトリマ7上限値(32ビット)		
CM1716～CM1719	－	システム予約		
CM1720	R	AW　ユーザメッセージ表示番号		
CM1721～CM1735	－	システム予約		
CM1736		AW　初期表示言語設定		
CM1737		AW　初期表示コメント番号		
CM1738		AW　操作禁止設定		
CM1739	－	システム予約		
CM1740～CM1772	R	ラダープロジェクト名(最大32文字＋終端コード)		
CM1773～CM1779	－	システム予約		
CM1780		AW　初期画面設定 有効		
CM1781		AW　初期画面設定 デバイス種別		
CM1782		AW　初期画面設定 画面ID		
CM1783		AW　初期画面設定 画面ID		
CM1784		AW　初期画面設定 デバイス番号		
CM1785		AW　初期画面設定 デバイス番号		
CM1786		AW　初期画面設定 表示形式		
CM1787		AW　初期画面設定 デバイス名表記		
CM1788		AW　初期画面設定 キーロック状態		
CM1789	－	システム予約		
CM1790		オートロード　フォルダ番号(要求)		
CM1791	R	オートロード　完了コード		
CM1792	R	オートロード　フォルダ番号(完了)		
CM1793	－	システム予約		
CM1794		ランロード　フォルダ番号(要求)		
CM1795	R	ランロード　完了コード		
CM1796	R	ランロード フォルダ番号(完了)		
CM1797～CM1868	－	システム予約		
CM1869	R	拡張ユニット接続台数		
CM1870		レシピロード　ドライブ番号		○
CM1871		レシピロード　フォルダ番号		○
CM1872	－	システム予約		
CM1873	R	レシピロード　完了コード		○
CM1874	－	システム予約		
CM1875		レシピセーブ　ドライブ番号		○
CM1876		レシピセーブ　参照フォルダ番号		○
CM1877		レシピセーブ　出力フォルダ番号		○
CM1878	R	レシピセーブ　完了コード		○
CM1879	－	システム予約		
CM1880		プロジェクトロード時ドライブ番号(要求)		
CM1881		プロジェクトロード時フォルダ番号(要求)		
CM1882	R	プロジェクトロード時ドライブ番号(完了)		
CM1883	R	プロジェクトロード時フォルダ番号(完了)		
CM1884	R	プロジェクトロード時完了コード		
CM1875～CM1877	－	システム予約		
CM1888		プロジェクトセーブ時ドライブ番号(要求)		
CM1889		プロジェクトセーブ時フォルダ番号(要求)		
CM1890	R	プロジェクトセーブ時ドライブ番号(完了)		
CM1891	R	プロジェクトセーブ時フォルダ番号(完了)		
CM1892	R	プロジェクトセーブ時完了コード		
CM1893～CM1999	－	システム予約		
CM2000～CM2001	R	運転記録機能(ID0)	デバイス記録0 平均サンプリング周期	
CM2002～CM2003	R		デバイス記録0 平均変化デバイス数	
CM2004～CM2005	R		デバイス記録0 最大収集期間	
CM2006～CM2009	－		システム予約	
CM2010～CM2039		運転記録機能(ID1～ID3)		
CM2040～CM2389	－	システム予約		

CM	属性 R：読出専用 空欄：読書可能	機　能		保持 ○：保持
CM2390	R	ストレージ命令用エラーコード		
CM2391		ストレージアクセス中電源OFF （発生時に0以外の値になる。プログラムから0にしてください。）		
CM2392		メモリカードリトライ発生回数（電源断時も保持）		
CM2393		メモリカードベリファイエラー発生回数（電源断時も保持）		
CM2394		メモリカードタイムアウト発生回数（電源断時も保持）		
CM2395		メモリカードエラー時リセット遅延時間		
CM2396～CM2399	－	システム予約		
CM2400～CM2401	R	ユニット間同期リフレッシュ開始時間(0.1μs)		
CM2402～CM2403	R	ユニット間同期リフレッシュ終了時間(0.1μs)		
CM2404～CM2405		ユニット間同期リフレッシュ開始時間最大(0.1μs) (リセットするにはプログラムから0にしてください。)		○
CM2406～CM2407		ユニット間同期リフレッシュ終了時間最大(0.1μs) (リセットするにはプログラムから0にしてください。)		○
CM2408～CM2409	R	ユニット間同期リフレッシュ実行時間(0.1μs)		
CM2410～CM5099	－	システム予約		
CM5100	R	最新重度エラー詳細	年月(10進数で下二桁が月、百の桁から上が年。例：1210→2012年10月)	○
CM5101	R		日	○
CM5102	R		時	○
CM5103	R		分	○
CM5104	R		秒	○
CM5105	R		エラー番号	○
CM5106	R		有効データ数	○
CM5107	R		詳細情報01	○
CM5108	R		詳細情報02	○
:	R			○
CM5126	R		詳細情報20	○
CM5127～CM5149	－	システム予約		
CM5150	R	最新軽度エラー詳細	年月(10進数で下二桁が月、百の桁から上が年。例：1210→2012年10月)	○
CM5151	R		日	○
CM5152	R		時	○
CM5153	R		分	○
CM5154	R		秒	○
CM5155	R		エラー番号	○
CM5156	R		有効データ数	○
CM5157	R		詳細情報01	○
CM5158	R		詳細情報02	○
:	R			○
CM5176	R		詳細情報20	○
CM5177～CM5871	－	システム予約		
CM5872～CM5873	R	CPUメモリ使用量（32ビット、Byte単位）		○
CM5874～CM5999	－	システム予約		
CM6000		運転記録機能(ID0)	運転記録0 保存完了コード	
CM6001	－		システム予約	
CM6002			運転記録0 保存カウンタ	
CM6003	－		システム予約	
CM6004	R		運転記録0 保存進捗詳細1	
CM6005	R		運転記録0 保存進捗詳細2	
CM6006	R		運転記録0 保存進捗	
CM6007～CM6009	－		システム予約	
CM6010～CM6171	R		運転記録0 保存パス	
CM6172～CM6199	－		システム予約	
CM6200～CM6799		運転記録機能(ID1～ID3)		
CM6800～CM7599	－	システム予約		

■エラーの詳細情報－KV-8000の場合

- No.128（演算エラー）の場合

詳細情報n	説　明
詳細情報1 （CM0712/CM5157）	**エラーの詳細内容** 0　（未使用） 1　間接指定が不適切 2　間接指定先がT/C 3　簡易間接指定が不適切 4　オペランド範囲外 5　浮動小数点型実数オーバーフロー 6　デバイス指定が不適切 7　ROOT命令の引数が負 8　ENCO命令でONしているビットがない 9　浮動小数点型実数フォーマット異常 10　三角関数命令の引数が不適切 11　変換できないデータ 12　テーブル指定が不適切 13　時刻比較命令の引数が不適切 14　カムスイッチオペランドが不適切 15　周波数カウンタ命令の引数が不適切 16　0除算 17　ロギングID不正 18　ダイレクト処理エラー 19　ダイレクト処理できないユニット 20　ジャンプ先サブルーチンが存在しない 21　RUN中書き込み中なので命令が実行不可 22　拡張ユニット専用命令対象エラー（ユニット番号） 23　システム予約 24　PIDAT 動作設定範囲外 25　PIDAT サンプリング周期範囲外 26　PIDAT 比例定数範囲外 27　PIDAT 積分定数範囲外 28　PIDAT 微分定数範囲外 29　PIDAT 比例定数2自由度化係数範囲外 30　PIDAT 積分定数2自由度化係数範囲外 31　PIDAT 微分定数2自由度化係数範囲外 32　PIDAT 操作量上限値･下限値範囲外 33　PIDAT 操作量変化幅制限値範囲外 34　PIDAT 測定値変化幅範囲外 35　PIDAT AT状態遷移異常 36　PIDAT 動作制御フラグ範囲外 37　演算周期オーバー 38　PIDAT AT調整パラメータ範囲外 39　PIDAT ATヒステリシス範囲外 40　PIDAT ATタイムアウト時間範囲外 41　PIDAT AT算出制御方式設定範囲外 42　PIDAT ATタイムアウト 43　PIDAT AT比例定数オーバー 44　PIDAT AT積分定数オーバー 45　PIDAT AT微分定数オーバー 46　負の数の累乗根は演算できない 47　ABS命令の引数が不適切 48　演算オーバーフロー 49　UR/UM/UV ユニット未接続 50　ユニットデバイス（R）未割付 51　ユニットデバイス（DM）未割付 52　ユニット種別不一致 53　運転記録ID不正 54　スタックオーバーフロー
詳細情報2 （CM5158）	00H固定
詳細情報3 （CM5159）	エラーが発生しているステップ番号上位
詳細情報4 （CM5160）	エラーが発生しているステップ番号下位

詳細情報n	説　明
詳細情報5～20 （CM5161～CM5176）	モジュール名をプロジェクト言語設定に従って格納します。 （モジュール名以外の部分は00H）

※ CR2012がON時にCM0712には即時、CM5150~CM5176には数スキャン後値が格納されます。

- No.50,53,55,56,58,59,63（ユニット番号に関するエラー）の場合

詳細情報n	説　明
詳細情報1 （CM5107）	ユニット番号（エラーの発生状況によってユニット番号が特定できない場合には、格納されません。）
詳細情報2～20 （CM5108～ CM5126）	00H固定

- No.54（ユニットの台数に関するエラー）の場合

詳細情報n	説　明
詳細情報1 （CM5107）	ユニット設定情報でのユニットの台数
詳細情報2 （CM5108）	実際に接続されているユニットの台数
詳細情報3～20 （CM5109～ CM5126）	00H固定

- No.127（オートローディング失敗エラー）の場合

詳細情報n	説　明
詳細情報1 （CM5157）	エラーの詳細内容
詳細情報2～20 （CM5158～ CM5176）	00H固定

- No.129（ユニットエラー）の場合

拡張情報n	解　説
拡張情報1 （CM5157）	ユニット番号
拡張情報2 （CM5158）	ユニット毎のエラー番号
拡張情報3～20 （CM5159～ CM5176）	00H固定

- 上記以外のエラー番号の場合

拡張情報n	解　説
拡張情報1～20	00H固定

※1　Bluetooth通信に対応したCPUユニットのみで有効です。

コントロールリレー CR KV-7500/7300の場合

コントロールリレー（CR0000～CR7915）は、CPUユニットの動作状態の確認、各内蔵機能の設定やモニタなどができます。
システム予約されているコントロールリレーは使用できません。

CR	属性 R：読出専用 空欄：読書可能	機　能	
CR0000		ロギング・トレース機能（ID0）	ロギング・トレース 実行（ON：許可、OFF：禁止）
CR0001	R		ロギング・トレース 動作中
CR0002	R		ロギング・トレース ファイル書き込み終了
CR0003	R		ロギング バッファ50%未満警告
CR0004	R		ロギング バッファオーバーフロー
CR0005	R		ロギング 実行中RUN中書き込みあり
CR0006～CR0007	－		システム予約
CR0008	R		ロギング・トレース エラー
CR0009	R		ロギング・トレース メモリ容量なしエラー
CR0010	R		ロギング・トレース デバイス・トリガ設定異常
CR0011	R		トレース ファイル保存トリガモニタ
CR0012	R		トレース データ取得完了
CR0013	R		ロギング・トレース 設定書き込みあり
CR0014～CR0015	－		システム予約
CR0100～CR0915		ロギング・トレース機能（ID1～ID9）	
CR1000～CR1015		リアルタイムチャートモニタ（ID10）	
CR1100～CR2001	－	システム予約	
CR2002	R	常時ON	
CR2003	R	常時OFF	
CR2004	R	10msクロックパルス（duty50%）	
CR2005	R	100msクロックパルス（duty50%）	
CR2006	R	1sクロックパルス（duty50%）	
CR2007	R	運転開始時１スキャンのみOFF	
CR2008	R	運転開始時１スキャンのみON	
CR2009		演算結果が負またはオーバーフローのときON	
CR2010	R	演算結果がゼロのときON	
CR2011	R	演算結果が正のときON	
CR2012	R	演算実行エラーのときON	
CR2013～CR2302	－	システム予約	
CR2303		一定スキャンタイム運転（ON：有効、OFF：無効）	
CR2304	R	スキャンタイム設定 （（一定スキャンタイム運転、END処理時間設定）オーバー時1スキャンのみON）	
CR2305	－	システム予約	
CR2306	R	バックアップ電池異常（ON:異常、OFF：正常）	
CR2307		スキャンタイム最小値最大値のクリア	
CR2308		END処理時間設定 （ON：有効、OFF:無効）（CR2303と同時にONした場合は、CR2303が優先）	
CR2309～CR2310	－	システム予約	
CR2311	R	RTC電圧低下エラー（ON：エラー発生中、OFF：正常）	
CR2312	－	システム予約	
CR2313		VTセンサアプリケーション画面強制切換（切換する：ON、切換しない：OFF）	
CR2314	R	プロジェクトパスワード認証禁止状態（ON：認証禁止、OFF:認証可能）	
CR2315～CR2813	－	システム予約	

CR	属性 R:読出専用 空欄:読書可能	機　能
CR2814		DASC、FASC命令(全サフィックス)実行時にゼロサプレスを(ON:する、OFF : しない)
CR2815		DASC.S、DASC.L、FASC実行時に＋の符号を省略(ON:する、OFF : しない)
CR2900～CR3115	－	システム予約
CR3200		CPUメモリ　保存処理要求
CR3201	R	CPUメモリ　保存処理中
CR3202	R	CPUメモリ　保存処理完了
CR3203	R	CPUメモリ　保存処理エラー
CR3204～CR3206	－	システム予約
CR3207		命令語 RAMモード動作 (CPUメモリに対するファイルレジスタ命令･ストレージ命令のRAMモード有効化(ON:有効、OFF:無効))
CR3208～CR3209	－	システム予約
CR3210	R	任意ロギング･トレース動作中
CR3211～CR3213	－	システム予約
CR3214	R	ストレージ命令実行中
CR3215	－	システム予約
CR3300～CR3415		アラームリレー
CR3500	R	アラーム動作リレー(ON : CR3300～CR3415のいずれかがON)
CR3501		アラームリレーがON→OFFのときに履歴を記録(ON : 残す、OFF : 残さない)
CR3502		OFF→ON時にアラームリレー履歴クリア
CR3503～CR3512	－	システム予約
CR3513	R	ロギング内部バッファオーバフロー
CR3514		HKEY命令　複数キー禁止
CR3515	R	HKEY命令　スキャン終了
CR3600～CR3615	R	HKEYの情報格納エリア
CR3700～CR3815	－	システム予約
CR3900		OFF→ON時に重度エラー履歴クリア
CR3901		OFF→ON時に軽度エラー履歴クリア
CR3902	R	重度エラー履歴が最大件数
CR3903	R	軽度エラー履歴が最大件数
CR3904	R	RUN/PRG切換SW状態(ON:RUN側/OFF:PRG側)
CR3905	R	重度エラー発生中
CR3906	R	軽度エラー発生中
CR3907		OFF→ON時に電源ON履歴クリア
CR3908		OFF→ON時に電源OFF履歴クリア
CR3909		現在のエラークリア(ラダープログラム中でSETしてください)
CR3910～CR4000	－	システム予約
CR4001～CR4300	R	拡張ユニット有無(拡張ユニット1～48)
CR4301～CR4315	－	システム予約
CR4400	R	AW　上キー押下中
CR4401	R	AW　下キー押下中
CR4402	R	AW　左キー押下中
CR4403	R	AW　右キー押下中
CR4404	R	AW　MENUキー押下中
CR4405	R	AW　ENTキー押下中
CR4406～CR4414	－	システム予約
CR4415	R	AW　ユーザメッセージ表示中
CR4500		AW　1行目ブリンク

CR	属性 R:読出専用 空欄:読書可能	機　能
CR4501		AW　2行目ブリンク
CR4502		AW　3行目ブリンク
CR4503		AW　4行目ブリンク
CR4504～CR4600	－	システム予約
CR4601～CR4915		AW ユーザメッセージ1～63
CR5000～CR5315	－	システム予約
CR5400		メモリカードマウント要求
CR5401		メモリカードアンマウント要求
CR5402	R	メモリカード使用中
CR5403	R	メモリカード認識完了
CR5404	R	メモリカードあり
CR5405	R	メモリカードライトプロテクト
CR5406～CR5407	－	システム予約
CR5408		CPUメモリマウント要求
CR5409		CPUメモリアンマウント要求
CR5410	R	CPUメモリ使用中
CR5411	R	CPUメモリ認識完了
CR5412～CR7915	－	システム予約

コントロールメモリ CM KV-7500/7300の場合

コントロールメモリ（CM0000～CM5999）は、CPUユニットの動作状態の確認、各内蔵機能の設定やモニタなどができます。
システム予約されているコントロールメモリは使用できません。

CM	属性 R：読出専用 空欄：読書可能	機　能	保持 ○：保持
CM0000～CM0699	－	システム予約	
CM0700	R	RTCからの読み出し（年）	
CM0701	R	RTCからの読み出し（月）	
CM0702	R	RTCからの読み出し（日）	
CM0703	R	RTCからの読み出し（時）	
CM0704	R	RTCからの読み出し（分）	
CM0705	R	RTCからの読み出し（秒）	
CM0706	R	RTCからの読み出し（曜日）	
CM0707～CM0709	－	システム予約	
CM0710	R	ファイルレジスタのカレントバンク番号	
CM0711～CM0899	－	システム予約	
CM900	R	CPU機能バージョン	
CM901～CM906	－	システム予約	
CM907	R	プロジェクトバージョン	
CM908～CM999	－	システム予約	
CM1000～CM1001	R	スキャンタイム測定値（1μs）（32ビット）	
CM1002～CM1003	R	END処理時間測定値（1μs単位）（32ビット）	
CM1004～CM1005	R	スキャンタイム最小値（1μs単位）（32ビット）	
CM1006～CM1007	R	スキャンタイム最大値（1μs単位）（32ビット）	
CM1008～CM1009		一定スキャンタイム運転時設定値（1μs単位）（32ビット）	
CM1010～CM1011		END処理時間設定値（1μs単位）（32ビット）	
CM1012～CM1013	R	設定超過時スキャンタイム（1μs単位）（32ビット）	
CM1014～CM1015	R	END処理時間超過値（1μs単位）（32ビット）	
CM1016～CM1029	－	システム予約	
CM1030～CM1031	R	32ビット1msフリーランカウンタ	
CM1032～CM1033	R	32ビット10μsフリーランカウンタ	
CM1034～CM1035	R	32ビット1μsフリーランカウンタ	
CM1036～CM1037	R	32ビット0.1μsフリーランカウンタ	
CM1038～CM1499	－	システム予約	
CM1500～CM1509	R	ロギング･トレース（ID0～ID9）ファイル保存カウンタ	
CM1510～CM1629	－	システム予約	
CM1630～CM1639	R	ロギング･トレース リングバッファの空き容量最小 （ID0～ID9）（KByte単位）	
CM1640～CM1649	－	システム予約	
CM1650	R	センサ設定命令実行数	
CM1651～CM1657	－	システム予約	
CM1658	R	プロジェクトパスワード連続認証失敗回数	○
CM1659	R	プロジェクトパスワード累積認証失敗回数	○
CM1660～CM1669	R	ロギング･トレース リングバッファの空き容量（ID0~ID9）（KByte単位）	
CM1670～CM1679	－	システム予約	
CM1680～CM1689	R	ロギング･トレース ファイル保存カウンタ （ID0～ID9）（メモリカード用）	○
CM1690～CM1699	－	システム予約	

CM	属性 R:読出専用 空欄:読書可能	機　能	保持 ○:保持
CM1700～CM1701		デジタルトリマ0上限値(32ビット)	
CM1702～CM1703		デジタルトリマ1上限値(32ビット)	
CM1704～CM1705		デジタルトリマ2上限値(32ビット)	
CM1706～CM1707		デジタルトリマ3上限値(32ビット)	
CM1708～CM1709		デジタルトリマ4上限値(32ビット)	
CM1710～CM1711		デジタルトリマ5上限値(32ビット)	
CM1712～CM1713		デジタルトリマ6上限値(32ビット)	
CM1714～CM1715		デジタルトリマ7上限値(32ビット)	
CM1716～CM1719	－	システム予約	
CM1720	R	AW　ユーザメッセージ表示番号	
CM1721～CM1735	－	システム予約	
CM1736		AW　初期表示言語設定	
CM1737		AW　初期コメント番号設定	
CM1738		AW　操作禁止設定	
CM1739	－	システム予約	
CM1740～CM1772	R	ラダープロジェクト名(最大32文字＋終端コード)	
CM1773～CM1779	－	システム予約	
CM1780		AW　初期画面設定 有効	
CM1781		AW　初期画面設定 デバイス種別	
CM1782		AW　初期画面設定 画面ID	
CM1783		AW　初期画面設定 画面ID	
CM1784		AW　初期画面設定 デバイス番号	
CM1785		AW　初期画面設定 デバイス番号	
CM1786		AW　初期画面設定 表示形式	
CM1787		AW　初期画面設定 デバイス名表記	
CM1788		AW　初期画面設定 キーロック状態	
CM1789	－	システム予約	
CM1790		オートロード　フォルダ番号(要求)	
CM1791	R	オートロード　完了コード	
CM1792	R	オートロード　フォルダ番号(完了)	
CM1793	－	システム予約	
CM1794		ランロード　フォルダ番号(要求)	
CM1795	R	ランロード　完了コード	
CM1796	R	ランロード フォルダ番号(完了)	
CM1797～CM1868	－	システム予約	
CM1869	R	拡張ユニット接続台数	
CM1870～CM2389	－	システム予約	
CM2390	R	ストレージ命令用エラーコード	
CM2391		ストレージアクセス中電源OFF (発生時に0以外の値になる。プログラムから0にしてください。)	
CM2392		メモリカードリトライ発生回数(電源断時も保持)	
CM2393		メモリカードベリファイエラー発生回数(電源断時も保持)	
CM2394		メモリカードタイムアウト発生回数(電源断時も保持)	
CM2395		メモリカードエラー時リセット遅延時間	
CM2396～CM5099	－	システム予約	

CM	属性 R：読出専用 空欄：読書可能	機　能		保持 ○：保持
CM5100	R	最新重度エラー詳細	年月	○
CM5101	R		日	○
CM5102	R		時	○
CM5103	R		分	○
CM5104	R		秒	○
CM5105	R		番号	○
CM5106	R		有効データ数	○
CM5107	R		詳細情報01※	○
CM5108	R		詳細情報02※	○
:	R			○
CM5126	R		詳細情報20※	○
CM5127～CM5149	－	システム予約		
CM5150	R	最新軽度エラー詳細	年月	○
CM5151	R		日	○
CM5152	R		時	○
CM5153	R		分	○
CM5154	R		秒	○
CM5155	R		番号	○
CM5156	R		有効データ数	○
CM5157	R		詳細情報01※	○
CM5158	R		詳細情報02※	○
:	R			○
CM5176	R		詳細情報20※	○
CM5177～CM5871	－	システム予約		
CM5872～CM5873	R	CPU機能バージョンメモリ使用量（32ビット、Byte単位）		○
CM5874～CM5999	－	システム予約		

※ 詳細情報は発生するエラー番号によって次頁のようになります。

- No.128（演算エラー）の場合

詳細情報n	説　明
詳細情報1 （CM5157）	**エラーの詳細内容** 0　（未使用） 1　間接指定が不適切 2　間接指定先がT/C 3　簡易間接指定が不適切 4　オペランド範囲外 5　浮動小数点型実数オーバーフロー 6　デバイス指定が不適切 7　ROOT命令の引数が負 8　ENCO命令でONしているビットがない 9　浮動小数点型実数フォーマット異常 10　三角関数命令の引数が不適切 11　変換できないデータ 12　テーブル指定が不適切 13　時刻比較命令の引数が不適切 14　カムスイッチオペランドが不適切 15　周波数カウンタ命令の引数が不適切 16　0除算 17　ロギングID不正 18　ダイレクト処理エラー 19　ダイレクト処理できないユニット 20　ジャンプ先サブルーチンが存在しない 21　RUN中書き込み中なので命令が実行不可 22　拡張ユニット専用命令対象エラー（ユニット番号） 23　システム予約 24　PIDAT 動作設定範囲外 25　PIDAT サンプリング周期範囲外 26　PIDAT 比例定数範囲外 27　PIDAT 積分定数範囲外 28　PIDAT 微分定数範囲外 29　PIDAT 比例定数2自由度化係数範囲外 30　PIDAT 積分定数2自由度化係数範囲外 31　PIDAT 微分定数2自由度化係数範囲外 32　PIDAT 操作量上限値･下限値範囲外 33　PIDAT 操作量変化幅制限値範囲外 34　PIDAT 測定値変化幅範囲外 35　PIDAT AT状態遷移異常 36　PIDAT 動作制御フラグ範囲外 37　演算周期オーバー 38　PIDAT AT調整パラメータ範囲外 39　PIDAT ATヒステリシス範囲外 40　PIDAT ATタイムアウト時間範囲外 41　PIDAT AT算出制御方式設定範囲外 42　PIDAT ATタイムアウト 43　PIDAT AT比例定数オーバー 44　PIDAT AT積分定数オーバー 45　PIDAT AT微分定数オーバー 46　負の数の累乗根は演算できない 47　ABS命令の引数が不適切 48　演算オーバーフロー 49　UR/UM/UV ユニット未接続 50　ユニットデバイス（R）未割付 51　ユニットデバイス（DM）未割付
詳細情報2 （CM5158）	00H固定
詳細情報3 （CM5159）	エラーが発生しているステップ番号上位
詳細情報4 （CM5160）	エラーが発生しているステップ番号下位
詳細情報5～20 （CM5161～CM5176）	モジュール名をプロジェクト言語設定に従って格納します。 （モジュール名以外の部分は00H）

- No.50,53,55,56,58,59,63（ユニット番号に関するエラー）の場合

詳細情報n	説　明
詳細情報1 （CM5107）	ユニット番号（エラーの発生状況によってユニット番号が特定できない場合には、格納されません。）
詳細情報2～20 （CM5108～CM5126）	00H固定

- No.54（ユニットの台数に関するエラー）の場合

詳細情報n	説　明
詳細情報1 （CM5107）	ユニット設定情報でのユニットの台数
詳細情報2 （CM5108）	実際に接続されているユニットの台数
詳細情報3～20 （CM5109～CM5126）	00H固定

- No.127（オートローディング失敗エラー）の場合

詳細情報n	説　明
詳細情報1 （CM5157）	エラーの詳細内容
詳細情報2～20 （CM5158～CM5176）	00H固定

- No.129（ユニットエラー）の場合

拡張情報n	解　説
拡張情報1 （CM5157）	ユニット番号
拡張情報2 （CM5158）	ユニット毎のエラー番号
拡張情報3～20 （CM5159～CM5176）	00H固定

- 上記以外のエラー番号の場合

拡張情報n	解　説
拡張情報1～20	00H固定

コントロールリレー CR KV-5500/5000/3000の場合

コントロールリレー(CR0000～CR3915)は、CPUユニットの動作状態の確認、各内蔵機能の設定やモニタなどができます。
システム予約されているコントロールリレーは使用できません。

デバイス番号	属性 R:読込み専用 空欄:読書き可能	機　能	
CR0000		ロギング･トレース機能(ID0)	ロギング･トレース 実行(ON:許可 OFF:禁止)
CR0001	R		ロギング･トレース 動作中
CR0002	R		ロギング･トレース ファイル書き込み終了
CR0003	R		ロギング バッファ50%未満警告
CR0004	R		ロギング バッファオーバーフロー
CR0005	R		ロギング 実行中RUN中書き込みあり
CR0006～CR0007	－		システム予約
CR0008	R		ロギング･トレース エラー
CR0009	R		ロギング･トレース メモリカード容量なしエラー
CR0010	R		ロギング･トレース デバイス･トリガ設定異常
CR0011	R		トレース ファイル保存トリガモニタ
CR0012	R		トレース データ取得完了
CR0013	R		ロギング･トレース 設定書き込みあり
CR0014～CR0015	－		システム予約
CR0100～CR0915		ロギング･トレース機能(ID1～ID9)	(CR0000～CR0015と同様)
CR1000～CR1015		リアルタイムチャートモニタ(ID10)	(CR0000～CR0015と同様)
CR1100～CR2001	－	システム予約	
CR2002	R	常時ON	
CR2003	R	常時OFF	
CR2004	R	10msクロックパルス(duty50%)	
CR2005	R	100msクロックパルス(duty50%)	
CR2006	R	1sクロックパルス(duty50%)	
CR2007	R	運転開始時1スキャンのみOFF	
CR2008	R	運転開始時1スキャンのみON	
CR2009		演算結果が負またはオーバーフローのときON	
CR2010	R	演算結果がゼロのときON	
CR2011	R	演算結果が正のときON	
CR2012	R	演算実行エラーのときON	
CR2013～CR2015	－	システム予約	
CR2100	R	CTH0専用内部クロック(1μs)	
CR2101	R	CTH0専用内部クロック(10μs)	
CR2102	R	CTH0専用内部クロック(100μs)	
CR2103		高速カウンタ コンパレータ CTC0のON時の動作	CTH0の自動リセット(ON:する OFF:しない)
CR2104			R500/R504へのコンパレーター致出力を禁止 (ON:する OFF:しない)
CR2105			R500/R504へのコンパレーター致出力をOFFに (ON:する OFF:しない)
CR2106			R500/R504へのコンパレーター致出力をONに (ON:する OFF:しない)
CR2107			ONするごとにR500/R504へのコンパレーター致出力のON/OFFを反転(ON:する OFF:しない)

<table>
<tr><th>デバイス番号</th><th>属性
R:読込み専用
空欄:読書き可能</th><th colspan="2">機　能</th></tr>
<tr><td>CR2108</td><td></td><td rowspan="4">高速カウンタ
コンパレータ
CTC1のON時の
動作</td><td>R500/R504 へのコンパレーター致出力を禁止
(ON:する OFF:しない)</td></tr>
<tr><td>CR2109</td><td></td><td>R500/R504 へのコンパレーター致出力をOFFに
(ON:する OFF:しない)</td></tr>
<tr><td>CR2110</td><td></td><td>R500/R504 へのコンパレーター致出力をONに
(ON:する OFF:しない)</td></tr>
<tr><td>CR2111</td><td></td><td>ONするごとに　R500/R504 へのコンパレーター致出力のON/OFFを反転(ON:する OFF:しない)</td></tr>
<tr><td>CR2112</td><td></td><td colspan="2">CTH0の自動リセット(ON:する OFF:しない)</td></tr>
<tr><td>CR2113～CR2115</td><td></td><td>高速カウンタ
CTH0カウント
入力方式選択</td><td>
<table>
<tr><th></th><th colspan="2">単相</th><th rowspan="2">1逓倍</th><th rowspan="2">2逓倍</th><th rowspan="2">4逓倍</th><th rowspan="2">2パルス</th></tr>
<tr><th>方向無し</th><th>方向有り</th></tr>
<tr><td>CR2113</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td></tr>
<tr><td>CR2114</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td><td>OFF</td><td>OFF</td></tr>
<tr><td>CR2115</td><td>OFF</td><td>OFF</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td></tr>
</table>
</td></tr>
<tr><td>CR2200</td><td>R</td><td colspan="2">CTH1専用内部クロック(1μs)</td></tr>
<tr><td>CR2201</td><td>R</td><td colspan="2">CTH1専用内部クロック(10μs)</td></tr>
<tr><td>CR2202</td><td>R</td><td colspan="2">CTH1専用内部クロック(100μs)</td></tr>
<tr><td>CR2203</td><td></td><td rowspan="5">高速カウンタ
コンパレータ
CTC2のON時の
動作</td><td>CTH1の自動リセット(ON:する OFF:しない)</td></tr>
<tr><td>CR2204</td><td></td><td>R501/R505 へのコンパレーター致出力を禁止
(ON:する OFF:しない)</td></tr>
<tr><td>CR2205</td><td></td><td>R501/R505 へのコンパレーター致出力をOFFに
(ON:する OFF:しない)</td></tr>
<tr><td>CR2206</td><td></td><td>R501/R505 へのコンパレーター致出力をONに
(ON:する OFF:しない)</td></tr>
<tr><td>CR2207</td><td></td><td>ONするごとに R501/R505 への出力のON/OFFを反転
(ON:する OFF:しない)</td></tr>
<tr><td>CR2208</td><td></td><td rowspan="5">高速カウンタ
コンパレータ
CTC3のON時の
動作</td><td>R501/R505 へのコンパレーター致出力を禁止
(ON:する OFF:しない)</td></tr>
<tr><td>CR2209</td><td></td><td>R501/R505 へのコンパレーター致出力をOFFに
(ON:する OFF:しない)</td></tr>
<tr><td>CR2210</td><td></td><td>R501/R505 へのコンパレーター致出力をONに
(ON:する OFF:しない)</td></tr>
<tr><td>CR2211</td><td></td><td>ONするごとに R501/R505 へのコンパレーター致出力のON/OFFを反転(ON:する OFF:しない)</td></tr>
<tr><td>CR2212</td><td></td><td colspan="2">CTH1の自動リセット(ON:する OFF:しない)</td></tr>
<tr><td>CR2213～CR2215</td><td></td><td>高速カウンタ
CTH1カウント
入力方式選択</td><td>
<table>
<tr><th></th><th colspan="2">単相</th><th rowspan="2">1逓倍</th><th rowspan="2">2逓倍</th><th rowspan="2">4逓倍</th><th rowspan="2">2パルス</th></tr>
<tr><th>方向無し</th><th>方向有り</th></tr>
<tr><td>CR2213</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td></tr>
<tr><td>CR2214</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td><td>OFF</td><td>OFF</td></tr>
<tr><td>CR2215</td><td>OFF</td><td>OFF</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td></tr>
</table>
</td></tr>
<tr><td>CR2300</td><td></td><td colspan="2">外部出力禁止(ON:する OFF:しない)</td></tr>
<tr><td>CR2301</td><td></td><td colspan="2">外部入力リフレッシュ禁止(ON:する OFF:しない)</td></tr>
<tr><td>CR2302</td><td></td><td colspan="2">ブレイク時外部出力禁止(ON:する OFF:しない)</td></tr>
<tr><td>CR2303</td><td></td><td colspan="2">一定スキャンタイム運転(ON:有効 OFF:無効)</td></tr>
<tr><td>CR2304</td><td>R</td><td colspan="2">スキャンタイム設定
(一定スキャンタイム運転･END処理時間の設定)オーバー時1スキャンのみON</td></tr>
<tr><td>CR2305</td><td></td><td colspan="2">CPUユニットの全入力の入力時定数を設定する
(ON:設定する→CM1620 OFF:設定しない[10ms])</td></tr>
<tr><td>CR2306</td><td>R</td><td colspan="2">バックアップ電池異常時にON</td></tr>
</table>

デバイス番号	属性 R:読込み専用 空欄:読書き可能	機　能
CR2307		スキャンタイム最大値/最小値のクリア
CR2308		END処理時間の設定(ON:有効 OFF:無効)
CR2309～CR2312	－	システム予約
CR2313		VTセンサアプリケーション画面強制切換(切替する:ON、切替しない:OFF)
CR2314	－	システム予約
CR2315	R	バッテリーレス運転中(通常運転時:OFF、バッテリーレス運転時:ON)
CR2400, CR2401		CTH0外部信号プリセット 未使用 / 立上がり / 立下がり / レベル CR2400: OFF / ON / OFF / ON CR2401: OFF / OFF / ON / ON
CR2402		CTH0カウント方式の選択(ON:リング OFF:リニア)
CR2403		CTH0オーバーフロー・アンダーフローを検出 (ON:検出した OFF:検出していない)
CR2404, CR2405		CTH1外部信号プリセット 未使用 / 立上がり / 立下がり / レベル CR2404: OFF / ON / OFF / ON CR2405: OFF / OFF / ON / ON
CR2406		CTH1カウント方式の選択(ON:リング OFF:リニア)
CR2407		CTH1オーバフロー・アンダーフローを検出(ON:検出した OFF:検出していない)
CR2408		CTH0プリセット禁止(ON:する　OFF:しない)
CR2409		CTH1プリセット禁止(ON:する　OFF:しない)
CR2410		CTH0内部イネーブルリレー(ON:カウント許可　OFF:カウント禁止)
CR2411		CTH1内部イネーブルリレー(ON:カウント許可　OFF:カウント禁止)
CR2412	R	CTH0変化方向(ON:減算側　OFF:加算側)
CR2413	R	CTH1変化方向(ON:減算側　OFF:加算側)
CR2414	R	CTH0専用内部クロック(50ns)
CR2415	R	CTH1専用内部クロック(50ns)
CR2500		周波数カウンタ(CTH0) 動作開始(ON:する　OFF:しない)
CR2501, CR2502		周波数カウンタ(CTH0) 入力ソース設定 単相方向無し / 単相方向あり / 2相1逓倍 / 2パルス CR2501: OFF / ON / OFF / ON CR2502: OFF / OFF / ON / ON
CR2503		周波数カウンタ(CTH0) Hz/rpm切り換え(ON:rpm　OFF:Hz)
CR2504		周波数カウンタ(CTH0) 1回転して回転速度を更新する機能を使用 (ON:使用する　OFF:使用しない)
CR2505		CTH0コンパレーター致出力リレー切り換え(ON:R504　OFF:R500)
CR2506		指定周波数パルス出力(CTH0)(ON:する　OFF:しない)
CR2507		指定周波数パルス出力(CTH0)エラー
CR2508		周波数カウンタ(CTH1) 動作開始(ON:する　OFF:しない)
CR2509, CR2510		周波数カウンタ(CTH1) 入力ソース設定 単相方向無し / 単相方向あり / 2相1逓倍 / 2パルス CR2509: OFF / ON / OFF / ON CR2510: OFF / OFF / ON / ON
CR2511		周波数カウンタ(CTH1) Hz/rpm切り換え(ON:rpm　OFF:Hz)
CR2512		周波数カウンタ(CTH1) 1回転して回転速度を更新する機能を使用 (ON:使用する　OFF:使用しない)
CR2513		CTH1コンパレーター致出力リレー切り換え(ON:R505　OFF:R501)
CR2514		指定周波数パルス出力(CTH1)(ON:する　OFF:しない)
CR2515		指定周波数パルス出力(CTH1)エラー

デバイス番号	属性 R：読込み専用 空欄：読書き可能	機　能	
CR2600,CR2601		INT R000割り込み極性	
CR2602,CR2603		INT R001割り込み極性（CR2600,CR2601と同様）	
CR2604,CR2605		INT R002割り込み極性（CR2600,CR2601と同様）	
CR2606,CR2607		INT R003割り込み極性（CR2600,CR2601と同様）	
CR2608,CR2609		INT R004割り込み極性（CR2600,CR2601と同様）	
CR2610,CR2611		INT R005割り込み極性（CR2600,CR2601と同様）	
CR2612,CR2613		INT R006割り込み極性（CR2600,CR2601と同様）	
CR2614,CR2615		INT R007割り込み極性（CR2600,CR2601と同様）	
CR2700,CR2701		INT R008割り込み極性（CR2600,CR2601と同様）	
CR2702,CR2703		INT R009割り込み極性（CR2600,CR2601と同様）	
CR2704～CR2715	－	システム予約	
CR2800		シリアル通信	ブレーク信号送出（ONの間送出）
CR2801	R		テキストデータ受信時に1スキャンのみON
CR2802	R		テキストデータ取り込みエラー
CR2803	R		テキストデータ受信エラー時に1スキャンのみON
CR2804			テキストデータ送信開始（送信中ON）
CR2805			テキストデータ受信をバイト単位で （ON：おこなう　OFF：おこなわない）
CR2806			テキストデータ送信をバイト単位で （ON：おこなう　OFF：おこなわない）
CR2807～CR2813	－	システム予約	
CR2814		DASC命令（全サフィックス）、FASC命令実行時にゼロサプレスを （ON：する　OFF：しない）	
CR2815		DASC.S、DASC.L、FASC命令実行時に＋の符号を省略（ON：する　OFF：しない）	
CR2900		アクセスウィンドウ	ユーザメッセージ表示1（ON：表示する　OFF：表示しない）
CR2901			ユーザメッセージ表示2（ON：表示する　OFF：表示しない）
CR2902～CR2915	－	システム予約	
CR3000		CPU位置決め機能X軸	リミットスイッチCW方向入力
CR3001			リミットスイッチCCW方向入力
CR3002			原点センサ入力
CR3003			停止センサ入力
CR3004	R		原点復帰動作中
CR3005	R		原点復帰完了
CR3006			位置決め完了リレー
CR3007			エラー/エラークリア
CR3008			ワーニング/ワーニングクリア
CR3009			パルス出力中/緊急停止
CR3010			ドライバ動作開始
CR3011			現在値変更要求(レベル検出)
CR3012			運転速度変更要求(レベル検出)
CR3013			コンパレータ2一致リレー
CR3014	－		システム予約
CR3015	－		

	CR2600	CR2601
	立ち上がり	立ち下がり	立ち上がり・下がり両方
CR2600	OFF	ON	OFF	ON
CR2601	OFF	OFF	ON	ON

デバイス番号	属性 R:読込み専用 空欄:読書き可能	機　能	
CR3100		CPU位置決め機能Y軸	リミットスイッチCW方向入力
CR3101			リミットスイッチCCW方向入力
CR3102			原点センサ入力
CR3103			停止センサ入力
CR3104	R		原点復帰動作中
CR3105	R		原点復帰完了
CR3106			位置決め完了リレー
CR3107			エラー/エラークリア
CR3108			ワーニング/ワーニングクリア
CR3109			パルス出力中/緊急停止
CR3110			ドライバ動作開始
CR3111			現在値変更要求(レベル検出)
CR3112			運転速度変更要求(レベル検出)
CR3113			コンパレータ2一致リレー
CR3114	－		システム予約
CR3115	－		
CR3200～CR3209	－	システム予約	
CR3210	R	任意ロギング･トレースメモリカード使用中	
CR3211	R	メモリカード使用中	
CR3212	R	メモリカード認識完了	
CR3213	R	メモリカードあり	
CR3214	R	メモリカード命令実行中	
CR3215	R	メモリカードライトプロテクト (ON:ライトプロテクトされている　OFF:ライトプロテクトされていない)	
CR3300～CR3415		アラームリレー	
CR3500	R	アラームリレー動作中(CR3300～CR3415のいずれかがONすればON)	
CR3501		アラームリレーがON→OFFのときに履歴を(ON:残す　OFF:残さない)	
CR3502		OFF→ON時にアラームリレー履歴クリア	
CR3503～CR3513	－	システム予約	
CR3514		HKEY命令	複数キー禁止
CR3515	R		スキャン終了
CR3600～CR3615	R	HKEY命令の情報格納エリア	

デバイス番号	属性 R：読込み専用 空欄：読書き可能	機　能	
CR3700		オペレータパネルKV-D30	F1 カスタマイズスイッチ1
CR3701			F2 カスタマイズスイッチ2
CR3702			F3 カスタマイズスイッチ3
CR3703			F4 カスタマイズスイッチ4
CR3704			LED1 カスタマイズランプ1
CR3705			LED2 カスタマイズランプ2
CR3706			LED3 カスタマイズランプ3
CR3707			LED4 カスタマイズランプ4
CR3708			緑バックライト（ON時点灯）
CR3709			赤バックライト（ON時点灯）
CR3710			システムメッセージ表示言語（ON：日本語　OFF：英語）
CR3711			ブザーON
CR3712			画面表示設定（ON：ポジ表示　OFF：ネガ表示）
CR3713			アラーム割り込み許可（ON：有効　OFF：無効）
CR3714			ページ切換（立ち上がりでCM0401の値のページに切り換え、強制OFFされる）
CR3715			「▲」「▼」キーでのページ切替禁止（ON：禁止　OFF：許可）
CR3800			ON時に外部出力0がON
CR3801			ON時に外部出力1がON
CR3802～CR3815	－	システム予約	
CR3900		OFF→ON時に重度エラー履歴クリア	
CR3901		OFF→ON時に軽度エラー履歴クリア	
CR3902	R	重度エラー履歴が最大件数	
CR3903	R	軽度エラー履歴が最大件数	
CR3904	R	RUN/PRG切換スイッチ状態（ON：RUN側　OFF：PROG側）	
CR3905	R	重度エラー発生中	
CR3906	R	軽度エラー発生中	
CR3907		OFF→ON時に電源ON履歴クリア	
CR3908		OFF→ON時に電源OFF履歴クリア	
CR3909		現在発生中のエラークリア（SET/RES命令のみ）	
CR3910～CR3915	－	システム予約	

コントロールメモリ CM KV-5500/5000/3000の場合

コントロールメモリ（CM0000～CM5999）は、CPUユニットの動作状態の確認、各内蔵機能の設定やモニタなどができます。
システム予約されているコントロールメモリは使用できません。

デバイス番号	属性 R：読込み専用 空欄：読書き可能	機　能	
CM0000, CM0001		オペレータパネル KV-D30 0ページ	1行目表示デバイス設定
CM0002, CM0003			2行目表示デバイス設定
CM0004, CM0005			3行目表示デバイス設定
CM0006, CM0007	－		4行目表示デバイス設定
CM0008, CM0009			システム予約
CM0010～CM0199		1～19ページ（CM0000～CM0009と同様）	
CM0200, CM0201		オペレータパネル KV-D30 0ページ	1行目表示属性設定
CM0202, CM0203			2行目表示属性設定
CM0204, CM0205			3行目表示属性設定
CM0206, CM0207			4行目表示属性設定
CM0208			テンプレート1設定
CM0209			テンプレート2設定
CM0210～CM0399		1～19ページ（CM0200～CM0209と同様）	
CM0400		オペレータパネル KV-D30	移行許可禁止設定用
CM0401			ページ切り換え設定用
CM0402			現在表示ページ
CM0403～CM0409			システム予約
CM0410～CM0415			ダイレクトアクセス機能
CM0416～CM0499	－		システム予約
CM0500～CM0699	－	システム予約	
CM0700	R	カレンダタイマからの読み出し	（年）
CM0701	R		（月）
CM0702	R		（日）
CM0703	R		（時）
CM0704	R		（分）
CM0705	R		（秒）
CM0706	R		（曜日）（0：（日）、1：（月）、2：（火）、3：（水）、4：（木）、5：（金）、6：（土））
CM0707	－	システム予約	
CM0708, CM0709	R	フリーランカウンタ（32ビット、1ms）	
CM0710	R	FMのカレントバンク番号	
CM0711	R	CPU機能バージョン2.0のとき「2」、それ以外の場合は「0」を格納	
CM0712	R	定周期モジュールの実行時間（10μs単位）	
CM0713～CM0715	－	システム予約	
CM0716		定周期モジュールの実行時間最大値（10μs単位）	
CM0717～CM0719	－	システム予約	
CM0720	R	スキャンタイム測定値（10μs単位）	
CM0721		一定スキャンタイム運転時設定値（10μs単位）	
CM0722	R	スキャンタイム設定（一定スキャンタイム運転・END処理時間の設定）オーバー時のスキャンタイム（10μs単位）	
CM0723		END処理時間測定値（10μs単位）	
CM0724～CM0725	－	システム予約	
CM0726	R	スキャンタイム最小値（10μs単位）	
CM0727	R	スキャンタイム最大値（10μs単位）	
CM0728		END処理時間の設定値（10μs単位）	
CM0729		スキャンタイム設定（一定スキャンタイム運転・END処理時間の設定）オーバー時のEND処理時間（10μs単位）	
CM0730～CM1479	－	システム予約	

デバイス番号	属性 R：読込み専用 空欄：読書き可能	機　能	
CM1480, CM1481	R	周波数カウンタ機能（CTH0）	周波数計測結果（Hzまたはrpm）
CM1482			1回転のパルス数
CM1483			スキャン回数
CM1484, CM1485		指定周波数パルス出力機能	設定周波数（Hz）
CM1486			Duty比（0～100%）
CM1487～CM1489	－	システム予約	
CM1490, CM1491	R	周波数カウンタ機能（CTH1）	周波数計測結果（Hzまたはrpm）
CM1492			1回転のパルス数
CM1493			スキャン回数
CM1494, CM1495		指定周波数パルス出力機能	設定周波数（Hz）
CM1496			Duty比（0～100%）
CM1497～CM1593	－	システム予約	
CM1594		CTH0変化方向検出時定数	
CM1595		CTH1変化方向検出時定数	
CM1596, CM1597		CTH0リングカウンタ下限値	
CM1598, CM1599		CTH1リングカウンタ下限値	
CM1600, CM1601		CTH0リングカウンタ上限値	
CM1602, CM1603		CTH1リングカウンタ上限値	
CM1604, CM1605		CTH0プリセット値	
CM1606, CM1607		CTH1プリセット値	
CM1608		CTH0イネーブル、リセット、極性設定	
CM1609		CTH1イネーブル、リセット、極性設定	
CM1610, CM1611	R	INT R00000発生時のインプットキャプチャ（下位、上位）	
CM1612, CM1613	R	INT R00001発生時のインプットキャプチャ（下位、上位）	
CM1614, CM1615	R	INT R00002発生時のインプットキャプチャ（下位、上位）	
CM1616, CM1617	R	INT R00003発生時のインプットキャプチャ（下位、上位）	
CM1618, CM1619	－	システム予約	
CM1620		入力時定数設定	
CM1621～CM1649		システム予約	
CM1650	R	センサ設定命令実行数	
CM1651～CM1659		システム予約	
CM1660～CM1669	R	ロギング･トレース リングバッファの空き容量（ID0～ID10）	
CM1670～CM1679	－	システム予約	
CM1680～CM1689	R	ロギング･トレース ファイル保存カウンタ（ID0～ID10）	
CM1690～CM1699	－	システム予約	
CM1700, CM1701		デジタルトリマ0上限値	
CM1702, CM1703		デジタルトリマ1上限値	
CM1704, CM1705		デジタルトリマ2上限値	
CM1706, CM1707		デジタルトリマ3上限値	
CM1708, CM1709		デジタルトリマ4上限値	
CM1710, CM1711		デジタルトリマ5上限値	
CM1712, CM1713		デジタルトリマ6上限値	
CM1714, CM1715		デジタルトリマ7上限値	
CM1716～CM1719		システム予約	
CM1720		アクセスウィンドウ	ユーザーメッセージ1
CM1721～CM1737			ユーザーメッセージ2（最大24文字＋終端コード（00H））
CM1738			操作禁止設定
CM1739	－		システム予約
CM1740～CM1756		ラダープロジェクト名（最大32文字＋終端コード（00H））	
CM1757～CM1763		アクセスウィンドウ表示用プロジェクト名（最大12文字＋終端コード（00H））	
CM1764～CM1774		Bluetoothデバイス識別名（最大20文字＋終端コード（00H））※1	
CM1775～CM1779		システム予約	
CM1780		AW初期画面設定 有効	

デバイス番号	属性 R:読込み専用 空欄:読書き可能	機　能	
CM1781		AW初期画面設定 デバイス種別	
CM1782,CM1783		AW初期画面設定 画面ID	
CM1784,CM1785		AW初期画面設定 デバイス番号	
CM1786		AW初期画面設定 表示形式	
CM1787		AW初期画面設定 デバイス名表記	
CM1788		AW初期画面設定 キーロック状態	
CM1789		AW初期画面設定 表示言語	
CM1790		電源投入時オートローディングフォルダ番号(要求)	
CM1791		電源投入時オートローディングフォルダ番号(完了)	
CM1792		PROG→RUN時オートローディングフォルダ番号(要求)	
CM1793		PROG→RUN時オートローディングフォルダ番号(完了)	
CM1794～CM1799		システム予約	
CM1800		重度エラー履歴 (最新)	年月(10進数で下二桁が月、百の桁から上が年。 例:706→2007年6月)
CM1801			日
CM1802			時(0～23)
CM1803			分
CM1804			エラー番号
CM1805～CM1899		重度エラー履歴(2～20件　CM1800～CM1804と同様)	
CM1900		電源ON履歴 (最新)	年月(10進数で下二桁が月、百の桁から上が年。 例:706→2007年6月)
CM1901			日
CM1902			時(0～23)
CM1903			分
CM1904			秒
CM1905～CM1999		電源ON履歴(2～20件　CM1900～CM1904と同様)	
CM2000		CPU 位置決め機能 X軸	入出力設定
CM2001			センサイネーブル
CM2002, CM2003			コンパレータ0
CM2004, CM2005			コンパレータ1
CM2006			原点復帰:起動速度
CM2007			原点復帰:加減速時間
CM2008, CM2009			原点復帰:運転速度
CM2010			JOG動作:起動速度
CM2011			JOG動作:加減速時間
CM2012, CM2013			JOG動作:運転速度
CM2014, CM2015			現在値変更設定値
CM2016, CM2017			運転速度変更設定値
CM2018			原点復帰詳細設定
CM2019			動作中ポイント番号(Mコード)
CM2020, CM2021	R		現在位置
CM2022, CM2023	R		現在速度
CM2024			エラーコード
CM2025			指定ポイント番号
CM2026, CM2027			ホームポジション座標設定値
CM2028, CM2029			コンパレータ2

デバイス番号	属性 R:読込み専用 空欄:読書き可能	機　能	
CM2030		CPU 位置決め機能 Y軸	入出力設定
CM2031			センサイネーブル
CM2032, CM2033			コンパレータ0
CM2034, CM2035			コンパレータ1
CM2036			原点復帰:起動速度
CM2037			原点復帰:加減速時間
CM2038, CM2039			原点復帰:運転速度
CM2040			JOG動作:起動速度
CM2041			JOG動作:加減速時間
CM2042, CM2043			JOG動作:運転速度
CM2044, CM2045			現在値変更設定値
CM2046, CM2047			運転速度変更設定値
CM2048			原点復帰詳細設定
CM2049			動作中ポイント番号(Mコード)
CM2050, CM2051	R		現在位置
CM2052, CM2053	R		現在速度
CM2054			エラーコード
CM2055			指定ポイント番号
CM2056, CM2057			ホームポジション座標設定値
CM2058, CM2059			コンパレータ2
CM2060, CM2061		CPU 位置決め機能 ポイント パラメータ 0	目標値/移動量
CM2062			起動速度
CM2063			加減速時間
CM2064, CM2065			運転速度
CM2066			動作モード
CM2067			停止センサ規程パルス数
CM2068, CM2069			システム予約
CM2070～CM2379		ポイントパラメータ　1～31(CM2060～CM2069と同様)	
CM2380～CM2389	－	システム予約	
CM2390	R	メモリカード命令用エラーコード	
CM2391		メモリカードアクセス中電源OFF(発生時:0以外の値)	
CM2392		メモリカードリトライ発生回数(電源断時も保持)	
CM2393		メモリカードベリファイエラー発生回数(電源断時も保持)	
CM2394		メモリカードタイムアウト発生回数(電源断時も保持)	
CM2395		メモリカードエラー時リセット遅延時間 (ms単位、上限は10000[10秒]、65535を設定するとリセットしなくなる)	
CM2396～CM2399		システム予約	
CM2400～CM2416		割り込み回数(INT0～INT3、CTC2、CTC3、CTC0、CTC1、位置決めX軸、位置決めY軸、INT4～INT9、定周期モジュール)	
CM2417～CM2999	－	システム予約	
CM3000		テキスト受信数	
CM3001～CM3999		テキスト受信データ(1998byte)	
CM4000		テキスト送信数	
CM4001～CM4999		テキスト送信データ(1998byte)	
CM5000		電源OFF履歴 (最新)	年月(10進数で下二桁が月、百の桁から上が年。 例:706→2007年6月)
CM5001			日
CM5002			時(0～23)
CM5003			分
CM5004			秒
CM5005～CM5099		電源OFF履歴(2～20件　CM5000～CM5004と同様)	

デバイス番号	属性 R：読込み専用 空欄：読書き可能	機　能	
CM5100		最新重度エラー詳細情報1件※	年月（10進数で下二桁が月、百の桁から上が年。 例：706→2007年6月）
CM5101			日
CM5102			時
CM5103			分
CM5104			秒
CM5105			エラー番号
CM5106			詳細情報有効データ数
CM5107			詳細情報1※
CM5108			詳細情報2※
： ：			： ：
CM5126			詳細情報20※
CM5150		最新軽度エラー詳細情報1件※	年月（10進数で下二桁が月、百の桁から上が年。 例：706→2007年6月）
CM5151			日
CM5152			時
CM5153			分
CM5154			秒
CM5155			エラー番号
CM5156			詳細情報有効データ数
CM5157			詳細情報1※
CM5158			詳細情報2※
： ：			： ：
CM5176			詳細情報20※
CM5177～CM5999	－	システム予約	

※ 詳細情報は発生するエラー番号によって次頁のようになります。

- No.128（演算エラー）の場合

詳細情報n	説　明
詳細情報1	**エラーの詳細内容** 0　（未使用） 1　間接指定が不適切 2　間接指定先がT/C 3　簡易間接指定が不適切 4　オペランド範囲外 5　浮動小数点型実数オーバーフロー 6　デバイス指定が不適切 7　ROOT命令の引数が負 8　ENCO命令でONしているビットがない 9　浮動小数点型実数フォーマット異常 10　三角関数命令の引数が不適切 11　変換できないデータ 12　テーブル指定が不適切 13　時刻比較命令の引数が不適切 14　カムスイッチオペランドが不適切 15　周波数カウンタ命令の引数が不適切 16　0除算 17　ロギングID不正 18　ダイレクト処理エラー 19　ダイレクト処理できないユニット 20　ジャンプ先サブルーチンが存在しない 21　RUN中書き込み中なので命令が実行不可 22　拡張ユニット専用命令対象エラー（ユニット番号） 23　システム予約 24　PIDAT 動作設定範囲外 25　PIDAT サンプリング周期範囲外 26　PIDAT 比例定数範囲外 27　PIDAT 積分定数範囲外 28　PIDAT 微分定数範囲外 29　PIDAT 比例定数2自由度化係数範囲外 30　PIDAT 積分定数2自由度化係数範囲外 31　PIDAT 微分定数2自由度化係数範囲外 32　PIDAT 操作量上限値･下限値範囲外 33　PIDAT 操作量変化幅制限値範囲外 34　PIDAT 測定値変化幅範囲外 35　PIDAT AT状態遷移異常 36　PIDAT 動作制御フラグ範囲外 37　演算周期オーバー 38　PIDAT AT調整パラメータ範囲外 39　PIDAT ATヒステリシス範囲外 40　PIDAT ATタイムアウト時間範囲外 41　PIDAT AT算出制御方式設定範囲外 42　PIDAT ATタイムアウト 43　PIDAT AT比例定数オーバー 44　PIDAT AT積分定数オーバー 45　PIDAT AT微分定数オーバー 46　負の数の累乗根は演算できない 47　ABS命令の引数が不適切 48　演算オーバーフロー

詳細情報n	説　明
詳細情報2	**00H固定**
詳細情報3	エラーが発生しているステップ番号上位
詳細情報4	エラーが発生しているステップ番号下位
詳細情報5～20	**モジュール名をSJISで格納します。(モジュール名以外の部分は00H)**

- No.50,53,55,56,58,59(ユニットIDに関するエラー)の場合

詳細情報n	説　明
詳細情報1	ユニット番号(エラーの発生状況によってユニット番号が特定できない場合には、格納されません。)
詳細情報2～20	00H固定

- No.54(ユニットの台数に関するエラー)の場合

詳細情報n	説　明
詳細情報1	ユニット設定情報でのユニットの台数
詳細情報2	実際に接続されているユニットの台数
詳細情報3～20	00H固定

- No.127(自動読み込みエラー) の場合

拡張情報n	解説
詳細情報1	エラーの詳細内容 詳細については、「ロード/セーブ実行時のエラーコード」を参照してください。
詳細情報2～20	00H固定

- No.129(ユニットエラー) の場合

拡張情報n	解説
拡張情報1	ユニット番号
拡張情報2	ユニット毎のエラー番号
拡張情報3～20	00H固定

- 上記以外のエラー番号の場合

拡張情報n	解説
拡張情報1～20	00H固定

コントロールリレー CR KV Nanoシリーズの場合

コントロールリレー(CR0000～CR8915)は、基本ユニットの動作状態の確認、各内蔵機能の設定やモニタなどができます。
システム予約されているコントロールリレーは使用できません。

CR	属性 R:読出専用 空欄:読書可能	機能	
CR0000		ロギング･トレース機能(ID0)	ロギング･トレース 実行(ON:許可、OFF:禁止)
CR0001	R		ロギング･トレース 動作中
CR0002	R		ロギング･トレース ファイル書き込み終了
CR0003	R		ロギング バッファ50%未満警告
CR0004	R		ロギング バッファオーバーフロー
CR0005	R		ロギング 実行中RUN中書き込みあり
CR0006～ CR0007	–		システム予約
CR0008	R		ロギング･トレース エラー
CR0009	R		ロギング･トレース メモリカード容量なしエラー
CR0010	R		ロギング･トレース デバイス･トリガ設定異常
CR0011	R		トレース ファイル保存トリガモニタ
CR0012	R		トレース データ取得完了
CR0013	R		ロギング･トレース 設定書き込みあり
CR0014～ CR0015	–		システム予約
CR0100～ CR0915		ロギング･トレース機能(ID1～ID9)	(CR0000～CR0015と同等)
CR1000～ CR1015		リアルタイムチャートモニタ(ID10)	(CR0000～CR0015と同等)
CR1100～ CR2001	–	システム予約	
CR2002	R	常時ON	
CR2003	R	常時OFF	
CR2004	R	10msクロックパルス(duty50%)	
CR2005	R	100msクロックパルス(duty50%)	
CR2006	R	1sクロックパルス(duty50%)	
CR2007	R	運転開始時１スキャンのみOFF	
CR2008	R	運転開始時１スキャンのみON	
CR2009		演算結果が負またはオーバーフローのときON	
CR2010	R	演算結果がゼロのときON	
CR2011	R	演算結果が正のときON	
CR2012	R	演算実行エラーのときON	
CR2013～ CR2100	–	システム予約	
CR2101	R	増設機器1有無(ON:接続、OFF:未接続)	
CR2102	R	増設機器2有無(ON:接続、OFF:未接続)	
CR2103～ CR2200	–	システム予約	
CR2201	R	拡張ユニット1有無(ON:接続、OFF:未接続)	
CR2202	R	拡張ユニット2有無(ON:接続、OFF:未接続)	
CR2203	R	拡張ユニット3有無(ON:接続、OFF:未接続)	
CR2204	R	拡張ユニット4有無(ON:接続、OFF:未接続)	
CR2205	R	拡張ユニット5有無(ON:接続、OFF:未接続)	
CR2206	R	拡張ユニット6有無(ON:接続、OFF:未接続)	
CR2207	R	拡張ユニット7有無(ON:接続、OFF:未接続)	
CR2208	R	拡張ユニット8有無(ON:接続、OFF:未接続)	

CR	属性 R:読出専用 空欄:読書可能	機能
CR2209～ CR2215	–	システム予約
CR2300		外部出力禁止(ON:する、OFF:しない)
CR2301		外部入力リフレッシュ禁止(ON:する、OFF:しない)
CR2302	–	システム予約
CR2303		一定スキャンタイム運転(ON:有効、OFF:無効)
CR2304	R	スキャンタイム設定(一定スキャンタイム運転、END処理時間設定)オーバー時１スキャンのみON
CR2305		基本ユニットの全入力の入力時定数を設定する (ON:設定する→CM1620、OFF:設定しない)
CR2306	–	システム予約
CR2307		スキャンタイム最小値最大値のクリア
CR2308		END処理時間設定(ON:有効、OFF:無効)(CR2303と同時にONした場合は、CR2303が優先)
CR2309		システム予約
CR2310	R	カレンダタイマあり(ON:あり、OFF:なし)
CR2311	R	時計データ消失エラー発生中にON(時計データを書き込むとOFF)
CR2312	–	システム予約
CR2313		VTセンサアプリケーション画面強制切換(切換する:ON、切換しない:OFF)
CR2314		プロジェクトパスワード認証禁止状態(ON:認証禁止、OFF:認証可能)
CR2315～ CR2515	–	システム予約
CR2600、 CR2601		INT0割り込み極性 立ち上がり / 立ち下がり / 立ち上がり・下がり両方 CR2600: OFF / ON / OFF / ON CR2601: OFF / OFF / ON / ON
CR2602、 CR2603		INT1割り込み極性 立ち上がり / 立ち下がり / 立ち上がり・下がり両方 CR2602: OFF / ON / OFF / ON CR2603: OFF / OFF / ON / ON
CR2604、 CR2605		INT2割り込み極性 立ち上がり / 立ち下がり / 立ち上がり・下がり両方 CR2604: OFF / ON / OFF / ON CR2605: OFF / OFF / ON / ON
CR2606、 CR2607		INT3割り込み極性 立ち上がり / 立ち下がり / 立ち上がり・下がり両方 CR2606: OFF / ON / OFF / ON CR2607: OFF / OFF / ON / ON
CR2608～ CR2813		システム予約
CR2814		DASC、FASC命令(全サフィックス)実行時にゼロサプレスを(ON:する、OFF:しない)
CR2815		DASC.S、DASC.L、FASC実行時に＋の符号を省略(ON:する、OFF:しない)
CR2900		アクセスウィンドウ ユーザメッセージ表示1 (ON:表示する、OFF:表示しない)
CR2901		アクセスウィンドウ ユーザメッセージ表示2 (ON:表示する、OFF:表示しない)
CR2902～ CR2914	–	システム予約
CR2915	R	AWカセットあり(ON:接続、OFF:未接続)
CR3000		プロジェクトロード実行要求
CR3001	–	システム予約
CR3002		プロジェクトセーブ実行要求
CR3003	–	システム予約
CR3004		プロジェクトロード実行完了

CR	属性 R:読出専用 空欄:読書可能	機能	
CR3005		プロジェクトロード実行失敗	
CR3006		プロジェクトセーブ実行完了	
CR3007		プロジェクトセーブ実行失敗	
CR3008～ CR3209	–	システム予約	
CR3210	R	任意ロギング･トレースメモリカード使用中	
CR3211	R	メモリカード使用中	
CR3212	R	メモリカード認識完了	
CR3213	R	メモリカードあり	
CR3214	R	メモリカード命令実行中	
CR3215	R	メモリカードライトプロテクト	
CR3300～ CR3415	–	アラームリレー	
CR3500	R	アラーム動作リレー(ON:CR3300～CR3415のいずれかがON)	
CR3501		アラームリレーがON→OFFのときに履歴を記録(ON:残す、OFF:残さない)	
CR3502		OFF→ON時にアラームリレー履歴クリア	
CR3503～ CR3513		システム予約	
CR3514		HKEY命令	複数キー禁止
CR3515	R		スキャン終了
CR3600～ CR3615	R	HKEYの情報格納エリア	
CR3700		オペレータパネルKV-D30	F1カスタマイズスイッチ1
CR3701			F2カスタマイズスイッチ2
CR3702			F3カスタマイズスイッチ3
CR3703			F4カスタマイズスイッチ4
CR3704			LED1カスタマイズランプ1
CR3705			LED2カスタマイズランプ2
CR3706			LED3カスタマイズランプ3
CR3707			LED4カスタマイズランプ4
CR3708			緑バックライト(ON時点灯)
CR3709			赤バックライト(ON時点灯)
CR3710			システムメッセージ表示言語(ON:日本語、OFF:英語)
CR3711			ブザーON
CR3712			画面表示設定(ON:ポジ表示、OFF:ネガ表示)
CR3713			アラーム割り込み許可(ON:有効、OFF:無効)
CR3714			ページ切換(立ち上がりでCM0401の値のページに切り換え、強制OFFされる)
CR3715			「▲」「▼」キーでのページ切替禁止 (ON:禁止　OFF:許可)
CR3800			ON時に外部出力0がON
CR3801			ON時に外部出力1がON
CR3802～ CR3815	–	システム予約	
CR3900		OFF→ON時に重度エラー履歴クリア	
CR3901		OFF→ON時に軽度エラー履歴クリア	
CR3902	R	重度エラー履歴が最大件数	
CR3903	R	軽度エラー履歴が最大件数	
CR3904	R	RUN/PROG切り換えSW状態(ON:RUN側/OFF:PROG側)	
CR3905	R	重度エラー発生中	
CR3906	R	軽度エラー発生中	
CR3907		OFF→ON時に電源ON履歴クリア	
CR3908		OFF→ON時に電源OFF履歴クリア	
CR3909		現在のエラークリア	

CR	属性 R:読出専用 空欄:読書可能	機能
CR3910～ CR4115	–	システム予約
CR4200	R	CTH0専用内部クロック(50ns)
CR4201	R	CTH0専用内部クロック(1μs)
CR4202	R	CTH0専用内部クロック(10μs)
CR4203	R	CTH0専用内部クロック(100μs)
CR4204	–	システム予約
CR4205	R	CTH0あり(ON:あり、OFF:なし)
CR4206		CTH0オーバーフロー･アンダーフローを検出(ON:検出した OFF:検出していない)
CR4207	R	CTH0変化方向(ON:減算側　OFF:加算側)
CR4208～ CR4209		CTH0外部信号プリセット 未使用 / 立ち上がり / 立ち下がり / レベル CR4208: OFF / ON / OFF / ON CR4209: OFF / OFF / ON / ON
CR4210		CTH0カウント方式の選択(ON:リング OFF:リニア)
CR4211		CTH0プリセット禁止(ON:する　OFF:しない)
CR4212		CTH0内部イネーブルリレー(ON:カウント許可　OFF:カウント禁止)
CR4213～ CR4302	–	システム予約
CR4303		高速カウンタコンパレータCTC0のON時の動作：CTH0の自動リセット(ON:する　OFF:しない)
CR4304		高速カウンタコンパレータCTC0のON時の動作：コンパレータCTC0のON時コンパレーター致出力を禁止(ON:する　OFF:しない)
CR4305		高速カウンタコンパレータCTC0のON時の動作：コンパレータCTC0のON時コンパレーター致出力をOFFに(ON:する　OFF:しない)
CR4306		高速カウンタコンパレータCTC0のON時の動作：コンパレータCTC0のON時コンパレーター致出力をONに(ON:する　OFF:しない)
CR4307		高速カウンタコンパレータCTC0のON時の動作：コンパレータCTC0がONするごとにコンパレーター致出力のON/OFFを反転(ON:する　OFF:しない)
CR4308		高速カウンタコンパレータCTC1のON時の動作：コンパレータCTC1のON時コンパレーター致出力を禁止(ON:する　OFF:しない)
CR4309		高速カウンタコンパレータCTC1のON時の動作：コンパレータCTC1のON時コンパレーター致出力をOFFに(ON:する　OFF:しない)
CR4310		高速カウンタコンパレータCTC1のON時の動作：コンパレータCTC1のON時コンパレーター致出力をONに(ON:する　OFF:しない)
CR4311		高速カウンタコンパレータCTC1のON時の動作：コンパレータCTC1がONするごとにコンパレーター致出力のON/OFFを反転(ON:する　OFF:しない)
CR4312		高速カウンタコンパレータCTC1のON時の動作：CTH0の自動リセット(ON:する　OFF:しない)
CR4313～ CR4315		高速カウンタCTH0カウント入力方式選択 単相(方向無し / 方向有り) / 1逓倍 / 2逓倍 / 4逓倍 / 2パルス CR4313: OFF / ON / OFF / ON / OFF / ON CR4314: OFF / OFF / ON / ON / OFF / OFF CR4315: OFF / OFF / OFF / OFF / ON / ON
CR4400	R	CTH1専用内部クロック(50ns)
CR4401	R	CTH1専用内部クロック(1μs)
CR4402	R	CTH1専用内部クロック(10μs)
CR4403	R	CTH1専用内部クロック(100μs)
CR4404	-	システム予約
CR4405	R	CTH1あり(ON:あり、OFF:なし)
CR4406		CTH1オーバーフロー･アンダーフローを検出(ON:検出した　OFF:検出していない)
CR4407	R	CTH1変化方向(ON:減算側　OFF:加算側)

<table>
<tr><th>CR</th><th>属性
R:読出専用
空欄:読書可能</th><th>機能</th></tr>
<tr><td>CR4408～
CR4409</td><td></td><td>CTH1外部信号プリセット
<table>
<tr><th></th><th>未使用</th><th>立ち上がり</th><th>立ち下がり</th><th>レベル</th></tr>
<tr><td>CR4408</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td></tr>
<tr><td>CR4409</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td></tr>
</table></td></tr>
<tr><td>CR4410</td><td></td><td>CTH1カウント方式の選択(ON:リング　OFF:リニア)</td></tr>
<tr><td>CR4411</td><td></td><td>CTH1プリセット禁止(ON:する　OFF:しない)</td></tr>
<tr><td>CR4412</td><td></td><td>CTH1内部イネーブルリレー(ON:カウント許可　OFF:カウント禁止)</td></tr>
<tr><td>CR4413～
CR4502</td><td>－</td><td>システム予約</td></tr>
<tr><td>CR4503</td><td></td><td>高速カウンタコンパレータCTC2のON時の動作</td><td>CTH1の自動リセット(ON:する　OFF:しない)</td></tr>
<tr><td>CR4504</td><td></td><td></td><td>コンパレータCTC2のON時コンパレーター致出力を禁止(ON:する　OFF:しない)</td></tr>
<tr><td>CR4505</td><td></td><td></td><td>コンパレータCTC2のON時コンパレーター致出力をOFFに(ON:する　OFF:しない)</td></tr>
<tr><td>CR4506</td><td></td><td></td><td>コンパレータCTC2のON時コンパレーター致出力をONに(ON:する　OFF:しない)</td></tr>
<tr><td>CR4507</td><td></td><td></td><td>コンパレータCTC2がONするごとにコンパレーター致出力のON/OFFを反転(ON:する OFF:しない)</td></tr>
<tr><td>CR4508</td><td></td><td>高速カウンタコンパレータCTC3のON時の動作</td><td>コンパレータCTC3のON時コンパレーター致出力を禁止(ON:する　OFF:しない)</td></tr>
<tr><td>CR4509</td><td></td><td></td><td>コンパレータCTC3のON時コンパレーター致出力をOFFに(ON:する　OFF:しない)</td></tr>
<tr><td>CR4510</td><td></td><td></td><td>コンパレータCTC3のON時コンパレーター致出力をONに(ON:する　OFF:しない)</td></tr>
<tr><td>CR4511</td><td></td><td></td><td>コンパレータCTC3がONするごとにコンパレーター致出力のON/OFFを反転(ON:する　OFF:しない)</td></tr>
<tr><td>CR4512</td><td></td><td></td><td>CTH1の自動リセット(ON:する OFF:しない)</td></tr>
<tr><td>CR4513～
CR4515</td><td></td><td>CTH1カウント入力方式選択
<table>
<tr><th></th><th colspan="2">単相</th><th rowspan="2">1逓倍</th><th rowspan="2">2逓倍</th><th rowspan="2">4逓倍</th><th rowspan="2">2パルス</th></tr>
<tr><th>方向無し</th><th>方向有り</th></tr>
<tr><td>CR4513</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td></tr>
<tr><td>CR4514</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td><td>OFF</td><td>OFF</td></tr>
<tr><td>CR4515</td><td>OFF</td><td>OFF</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td></tr>
</table></td></tr>
<tr><td>CR4600</td><td>R</td><td>CTH2専用内部クロック(50ns)</td></tr>
<tr><td>CR4601</td><td>R</td><td>CTH2専用内部クロック(1μs)</td></tr>
<tr><td>CR4602</td><td>R</td><td>CTH2専用内部クロック(10μs)</td></tr>
<tr><td>CR4603</td><td>R</td><td>CTH2専用内部クロック(100μs)</td></tr>
<tr><td>CR4604</td><td>－</td><td>システム予約</td></tr>
<tr><td>CR4605</td><td>R</td><td>CTH2あり(ON:あり、OFF:なし)</td></tr>
<tr><td>CR4606</td><td></td><td>CTH2オーバーフロー･アンダーフローを検出(ON:検出した　OFF:検出していない)</td></tr>
<tr><td>CR4607</td><td>R</td><td>CTH2変化方向(ON:減算側　OFF:加算側)</td></tr>
<tr><td>CR4608～
CR4609</td><td></td><td>CTH2外部信号プリセット
<table>
<tr><th></th><th>未使用</th><th>立ち上がり</th><th>立ち下がり</th><th>レベル</th></tr>
<tr><td>CR4608</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td></tr>
<tr><td>CR4609</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td></tr>
</table></td></tr>
<tr><td>CR4610</td><td></td><td>CTH2カウント方式の選択(ON:リング　OFF:リニア)</td></tr>
<tr><td>CR4611</td><td></td><td>CTH2プリセット禁止(ON:する　OFF:しない)</td></tr>
<tr><td>CR4612</td><td></td><td>CTH2内部イネーブルリレー(ON:カウント許可　OFF:カウント禁止)</td></tr>
<tr><td>CR4613～
CR4702</td><td>－</td><td>システム予約</td></tr>
</table>

CR	属性 R:読出専用 空欄:読書可能	機能	
CR4703		高速カウンタコンパレータCTC4のON時の動作	CTH2の自動リセット(ON:する OFF:しない)
CR4704			コンパレータCTC4のON時コンパレーター致出力を禁止(ON:する　OFF:しない)
CR4705			コンパレータCTC4のON時コンパレーター致出力をOFFに(ON:する　OFF:しない)
CR4706			コンパレータCTC4のON時コンパレーター致出力をONに(ON:する　OFF:しない)
CR4707			コンパレータCTC4がONするごとにコンパレーター致出力のON/OFFを反転(ON:する OFF:しない)
CR4708		高速カウンタコンパレータCTC5のON時の動作	コンパレータCTC5のON時コンパレーター致出力を禁止(ON:する　OFF:しない)
CR4709			コンパレータCTC5のON時コンパレーター致出力をOFFに(ON:する　OFF:しない)
CR4710			コンパレータCTC5のON時コンパレーター致出力をONに(ON:する　OFF:しない)
CR4711			コンパレータCTC5がONするごとにコンパレーター致出力のON/OFFを反転(ON:する　OFF:しない)
CR4712			CTH2の自動リセット(ON:する OFF:しない)
CR4713～ CR4715		CTH2カウント入力方式選択	

	単相		1逓倍	2逓倍	4逓倍	2パルス
	方向無し	方向有り				
CR4713	OFF	ON	OFF	ON	OFF	ON
CR4714	OFF	OFF	ON	ON	OFF	OFF
CR4715	OFF	OFF	OFF	OFF	ON	ON

CR4800	R	CTH3専用内部クロック(50ns)
CR4801	R	CTH3専用内部クロック(1μs)
CR4802	R	CTH3専用内部クロック(10μs)
CR4803	R	CTH3専用内部クロック(100μs)
CR4804	–	システム予約
CR4805	R	CTH3あり(ON:あり、OFF:なし)
CR4806		CTH3オーバーフロー･アンダーフローを検出(ON:検出した　OFF:検出していない)
CR4807	R	CTH3変化方向(ON:減算側　OFF:加算側)
CR4808～ CR4809		CTH3外部信号プリセット

	未使用	立ち上がり	立ち下がり	レベル
CR4808	OFF	ON	OFF	ON
CR4809	OFF	OFF	ON	ON

CR4810		CTH3カウント方式の選択(ON:リング　OFF:リニア)
CR4811		CTH3プリセット禁止(ON:する　OFF:しない)
CR4812		CTH3内部イネーブルリレー(ON:カウント許可　OFF:カウント禁止)
CR4813～ CR4902	–	システム予約

CR4903		高速カウンタコンパレータCTC6のON時の動作	CTH3の自動リセット(ON:する　OFF:しない)
CR4904			コンパレータCTC6のON時コンパレーター致出力を禁止(ON:する　OFF:しない)
CR4905			コンパレータCTC6のON時コンパレーター致出力をOFFに(ON:する　OFF:しない)
CR4906			コンパレータCTC6のON時コンパレーター致出力をONに(ON:する　OFF:しない)
CR4907			コンパレータCTC6がONするごとにコンパレーター致出力のON/OFFを反転(ON:する　OFF:しない)

<table>
<tr><th>CR</th><th>属性
R:読出専用
空欄:読書可能</th><th colspan="2">機能</th></tr>
<tr><td>CR4908</td><td></td><td rowspan="5">高速カウンタコンパレータCTC7のON時の動作</td><td>コンパレータCTC7のON時コンパレーター致出力を禁止(ON:する　OFF:しない)</td></tr>
<tr><td>CR4909</td><td></td><td>コンパレータCTC7のON時コンパレーター致出力をOFFに(ON:する　OFF:しない)</td></tr>
<tr><td>CR4910</td><td></td><td>コンパレータCTC7のON時コンパレーター致出力をONに(ON:する　OFF:しない)</td></tr>
<tr><td>CR4911</td><td></td><td>コンパレータCTC7がONするごとにコンパレーター致出力のON/OFFを反転(ON:する OFF:しない)</td></tr>
<tr><td>CR4912</td><td></td><td>CTH3の自動リセット(ON:する OFF:しない)</td></tr>
<tr><td>CR4913～
CR4915</td><td></td><td colspan="2">CTH3の逓倍モード選択
<table>
<tr><th></th><th colspan="2">単相</th><th rowspan="2">1逓倍</th><th rowspan="2">2逓倍</th><th rowspan="2">4逓倍</th><th rowspan="2">2パルス</th></tr>
<tr><th>方向無し</th><th>方向有り</th></tr>
<tr><td>CR4913</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td></tr>
<tr><td>CR4914</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td><td>OFF</td><td>OFF</td></tr>
<tr><td>CR4915</td><td>OFF</td><td>OFF</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td></tr>
</table></td></tr>
<tr><td>CR5000～
CR5315</td><td>－</td><td colspan="2">システム予約</td></tr>
<tr><td>CR5400</td><td></td><td rowspan="5">周波数カウンタ(CH0)</td><td>動作開始(ON:する　OFF:しない)</td></tr>
<tr><td>CR5401～
CR5402</td><td></td><td>CTH0入力ソース設定
<table>
<tr><th></th><th colspan="2">単相</th><th rowspan="2">2相
1逓倍</th><th rowspan="2">2パルス</th></tr>
<tr><th>方向無し</th><th>方向有り</th></tr>
<tr><td>CR5401</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td></tr>
<tr><td>CR5402</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td></tr>
</table></td></tr>
<tr><td>CR5403</td><td></td><td>Hz/rpm切り換え(ON:rpm　OFF:Hz)</td></tr>
<tr><td>CR5404</td><td></td><td>１回転して回転速度を更新する機能を ON:使用する、OFF:使用しない</td></tr>
<tr><td>CR5405</td><td></td><td>入力ソース切替機能を ON:使用する、OFF:使用しない</td></tr>
<tr><td>CR5406～
CR5407</td><td>－</td><td colspan="2">システム予約</td></tr>
<tr><td>CR5408</td><td></td><td colspan="2">指定周波数パルス出力(CTH0)(ON:する　OFF:しない)</td></tr>
<tr><td>CR5409</td><td></td><td colspan="2">指定周波数パルス出力(CTH0)エラー</td></tr>
<tr><td>CR5410～
CR5415</td><td>－</td><td colspan="2">システム予約</td></tr>
<tr><td>CR5500</td><td></td><td rowspan="5">周波数カウンタ(CH1)</td><td>動作開始(ON:する　OFF:しない)</td></tr>
<tr><td>CR5501～
CR5502</td><td></td><td>CTH1入力ソース設定
<table>
<tr><th></th><th colspan="2">単相</th><th rowspan="2">2相
1逓倍</th><th rowspan="2">2パルス</th></tr>
<tr><th>方向無し</th><th>方向有り</th></tr>
<tr><td>CR5501</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td></tr>
<tr><td>CR5502</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td></tr>
</table></td></tr>
<tr><td>CR5503</td><td></td><td>Hz/rpm切り換え(ON:rpm　OFF:Hz)</td></tr>
<tr><td>CR5504</td><td></td><td>１回転して回転速度を更新する機能を ON:使用する、OFF:使用しない</td></tr>
<tr><td>CR5505</td><td></td><td>入力ソース切替機能を ON:使用する、OFF:使用しない</td></tr>
<tr><td>CR5506～
CR5507</td><td>－</td><td colspan="2">システム予約</td></tr>
<tr><td>CR5508</td><td></td><td colspan="2">指定周波数パルス出力(CTH1)(ON:する　OFF:しない)</td></tr>
<tr><td>CR5509</td><td></td><td colspan="2">指定周波数パルス出力(CTH1)エラー</td></tr>
<tr><td>CR5510～
CR5515</td><td>－</td><td colspan="2">システム予約</td></tr>
</table>

<table>
<tr><th>CR</th><th>属性
R:読出専用
空欄:読書可能</th><th colspan="2">機能</th></tr>
<tr><td>CR5600</td><td></td><td rowspan="7">周波数カウンタ(CH2)</td><td>動作開始(ON:する　OFF:しない)</td></tr>
<tr><td>CR5601～
CR5602</td><td></td><td>CTH2入力ソース設定
<table>
<tr><th></th><th colspan="2">単相</th><th>2相</th><th rowspan="2">2パルス</th></tr>
<tr><th></th><th>方向無し</th><th>方向有り</th><th>1逓倍</th></tr>
<tr><td>CR5601</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td></tr>
<tr><td>CR5602</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td></tr>
</table></td></tr>
<tr><td>CR5603</td><td></td><td>Hz/rpm切り換え(ON:rpm　OFF:Hz)</td></tr>
<tr><td>CR5604</td><td></td><td>1回転して回転速度を更新する機能を ON:使用する、OFF:使用しない</td></tr>
<tr><td>CR5605</td><td></td><td>入力ソース切替機能を ON:使用する、OFF:使用しない</td></tr>
<tr><td>CR5606～
CR5607</td><td>－</td><td colspan="2">システム予約</td></tr>
<tr><td>CR5608</td><td></td><td colspan="2">指定周波数パルス出力(CTH2)(ON:する　OFF:しない)</td></tr>
<tr><td>CR5609</td><td></td><td colspan="2">指定周波数パルス出力(CTH2)エラー</td></tr>
<tr><td>CR5610～
CR5615</td><td>－</td><td colspan="2">システム予約</td></tr>
<tr><td>CR5700</td><td></td><td rowspan="7">周波数カウンタ(CH3)</td><td>動作開始(ON:する　OFF:しない)</td></tr>
<tr><td>CR5701～
CR5702</td><td></td><td>CTH3入力ソース設定
<table>
<tr><th></th><th colspan="2">単相方向</th><th>2相</th><th rowspan="2">2パルス</th></tr>
<tr><th></th><th>無し</th><th>有り</th><th>1逓倍</th></tr>
<tr><td>CR5701</td><td>OFF</td><td>ON</td><td>OFF</td><td>ON</td></tr>
<tr><td>CR5702</td><td>OFF</td><td>OFF</td><td>ON</td><td>ON</td></tr>
</table></td></tr>
<tr><td>CR5703</td><td></td><td>Hz/rpm切り換え(ON:rpm　OFF:Hz)</td></tr>
<tr><td>CR5704</td><td></td><td>1回転して回転速度を更新する機能を ON:使用する、OFF:使用しない</td></tr>
<tr><td>CR5705</td><td></td><td>入力ソース切替機能を ON:使用する、OFF:使用しない</td></tr>
<tr><td>CR5706～
CR5707</td><td>－</td><td colspan="2">システム予約</td></tr>
<tr><td>CR5708</td><td></td><td colspan="2">指定周波数パルス出力(CTH3)(ON:する　OFF:しない)</td></tr>
<tr><td>CR5709</td><td></td><td colspan="2">指定周波数パルス出力(CTH3)エラー</td></tr>
<tr><td>CR5710～
CR5715</td><td>－</td><td colspan="2">システム予約</td></tr>
<tr><td>CR5800～
CR5915</td><td>－</td><td colspan="2">システム予約</td></tr>
<tr><td>CR6000～
CR6515</td><td></td><td colspan="2">内蔵シリアル割付用</td></tr>
<tr><td>CR6600～
CR7115</td><td></td><td colspan="2">増設機能割付用1</td></tr>
<tr><td>CR7200～
CR7715</td><td></td><td colspan="2">増設機能割付用2</td></tr>
<tr><td>CR7800～
CR7915</td><td>－</td><td colspan="2">システム予約</td></tr>
</table>

CR	属性 R:読出専用 空欄:読書可能	機能	
CR8000		CPU位置決め機能　軸1	強制停止
CR8001			減速停止
CR8002			エラークリア
CR8003			ワーニングクリア
CR8004			現在座標変更要求(レベル検出)
CR8005			運転速度変更要求(レベル検出)
CR8006			目標座標変更要求(レベル検出)
CR8007～ CR8011	–		システム予約
CR8012			リミットスイッチCW方向入力
CR8013			リミットスイッチCCW方向入力
CR8014			原点センサ入力
CR8015			停止センサ入力
CR8100		CPU位置決め機能　軸2	強制停止
CR8101			減速停止
CR8102			エラークリア
CR8103			ワーニングクリア
CR8104			現在座標変更要求(レベル検出)
CR8105			運転速度変更要求(レベル検出)
CR8106			目標座標変更要求(レベル検出)
CR8107～ CR8111	–		システム予約
CR8112			リミットスイッチCW方向入力
CR8113			リミットスイッチCCW方向入力
CR8114			原点センサ入力
CR8115			停止センサ入力
CR8200		CPU位置決め機能　軸3	強制停止
CR8201			減速停止
CR8202			エラークリア
CR8203			ワーニングクリア
CR8204			現在座標変更要求(レベル検出)
CR8205			運転速度変更要求(レベル検出)
CR8206			目標座標変更要求(レベル検出)
CR8207～ CR8211			システム予約
CR8212			リミットスイッチCW方向入力
CR8213			リミットスイッチCCW方向入力
CR8214			原点センサ入力
CR8215			停止センサ入力
CR8300		CPU位置決め機能　軸4	強制停止
CR8301			減速停止
CR8302			エラークリア
CR8303			ワーニングクリア
CR8304			現在座標変更要求(レベル検出)
CR8305			運転速度変更要求(レベル検出)
CR8306			目標座標変更要求(レベル検出)
CR8307～ CR8311			システム予約
CR8312			リミットスイッチCW方向入力
CR8313			リミットスイッチCCW方向入力
CR8314			原点センサ入力
CR8315			停止センサ入力

CR	属性 R:読出専用 空欄:読書可能	機能	
CR8400	R	CPU位置決め機能　軸1	パルス出力中ON
CR8401	R		位置決め完了リレー
CR8402	R		エラー
CR8403	R		ワーニング
CR8404	R		原点復帰動作中
CR8405	R		原点復帰完了
CR8406～ CR8413	–		システム予約
CR8414	R		軸1 あり(ON:あり、OFF:なし)
CR8415	R		コンパレータ2一致リレー
CR8500	R	CPU位置決め機能　軸2	パルス出力中ON
CR8501	R		位置決め完了リレー
CR8502	R		エラー
CR8503	R		ワーニング
CR8504	R		原点復帰動作中
CR8505	R		原点復帰完了
CR8506～ CR8513	–		システム予約
CR8514	R		軸2 あり(ON:あり、OFF:なし)
CR8515	R		コンパレータ2一致リレー
CR8600	R	CPU位置決め機能　軸3	パルス出力中ON
CR8601	R		位置決め完了リレー
CR8602	R		エラー
CR8603	R		ワーニング
CR8604	R		原点復帰動作中
CR8605	R		原点復帰完了
CR8606～ CR8613	–		システム予約
CR8614	R		軸3 あり(ON:あり、OFF:なし)
CR8615	R		コンパレータ2一致リレー
CR8700	R	CPU位置決め機能　軸4	パルス出力中ON
CR8701	R		位置決め完了リレー
CR8702	R		エラー
CR8703	R		ワーニング
CR8704	R		原点復帰動作中
CR8705	R		原点復帰完了
CR8706～ CR8713	R		システム予約
CR8714	R		軸4 あり(ON:あり、OFF:なし)
CR8715	R		コンパレータ2一致リレー
CR8800～ CR8915	–	システム予約	

コントロールメモリ CM KV Nanoシリーズの場合

コントロールメモリ(CM0000～CM8999)は、基本ユニットの動作状態の確認、各内蔵機能の設定やモニタなどができます。
システム予約されているコントロールメモリは使用できません。

CM	属性 R:読出専用 空欄:読書可能	機能		保持 ○:保持 設定:保持設定可能
CM0000～CM0001		オペレータパネルKV-D300ページ	1行目表示デバイス設定	設定
CM0002～CM0003			2行目表示デバイス設定	設定
CM0004～CM0005			3行目表示デバイス設定	設定
CM0006～CM0007			4行目表示デバイス設定	設定
CM0008～CM0009			システム予約	設定
CM0010～CM0199		1～19ページ(CM0000～CM0009と同等)		設定
CM0200～CM0201		オペレータパネルKV-D300ページ	1行目表示属性設定	設定
CM0202～CM0203			2行目表示属性設定	設定
CM0204～CM0205			3行目表示属性設定	設定
CM0206～CM0207			4行目表示属性設定	設定
CM0208			テンプレート1設定	設定
CM0209			テンプレート2設定	設定
CM0210～CM0399		1～19ページ(CM0200～CM0209と同等)		設定
CM0400		オペレータパネルKV-D30	移行許可禁止設定用	設定
CM0401			ページ切り換え設定用	設定
CM0402			現在表示ページ	設定
CM0403～CM409			システム予約	設定
CM0410～CM415			ダイレクトアクセス機能	設定
CM0416～CM0499			システム予約	設定
CM0500～CM699	–	システム予約		
CM0700	R	RTCからの読み出し(年)		
CM0701	R	RTCからの読み出し(月)		
CM0702	R	RTCからの読み出し(日)		
CM0703	R	RTCからの読み出し(時)		
CM0704	R	RTCからの読み出し(分)		
CM0705	R	RTCからの読み出し(秒)		
CM0706	R	RTCからの読み出し(曜日)		
CM0707	–	システム予約		
CM0708～CM709	R	32ビット1msフリーランカウンタ		
CM0710		システム予約		
CM0711	R	CPU機能バージョン		

CM	属性 R:読出専用 空欄:読書可能	機能	保持 ○:保持 設定:保持設定可能
CM0712～ CM719	–	システム予約	
CM0720	R	スキャンタイム測定値(10μs単位)	
CM0721		一定スキャンタイム運転時設定値(10μs単位)	
CM0722	R	スキャンタイム設定(一定スキャンタイム運転･END処理時間の設定)オーバー時のスキャンタイム(10μs単位)	
CM0723	R	END処理時間測定値(10μs単位)	
CM0724～ CM725	–	システム予約	
CM0726	R	スキャンタイム最小値(10μs単位)	
CM0727	R	スキャンタイム最大値(10μs単位)	
CM0728		END処理時間設定値(10μs単位)	
CM0729	R	スキャンタイム設定(一定スキャンタイム運転･END処理時間の設定)オーバー時のEND処理時間(10μs単位)	
CM0730～ CM1599	–	システム予約	
CM1600～ CM1601	R	INT0発生時のインプットキャプチャ	
CM1602～ CM1603	R	INT1発生時のインプットキャプチャ	
CM1604～ CM1605	R	INT2発生時のインプットキャプチャ	
CM1606～ CM1607	R	INT3発生時のインプットキャプチャ	
CM1608～ CM1619	–	システム予約	
CM1620		入力時定数設定	○
CM1621～ CM1629	–	システム予約	
CM1630～ CM1631	R	アナログボリューム0	
CM1632～ CM1633	R	アナログボリューム1	
CM1634～ CM1649	–	システム予約	
CM1650	R	センサ設定命令実行数	
CM1651～ CM1657	–	システム予約	
CM1658	R	プロジェクトパスワード連続認証失敗回数	○
CM1659	R	プロジェクトパスワード累積認証失敗回数	○
CM1660～ CM1669	R	ロギング･トレース リングバッファの空き容量(K Byte単位) (ID0～ID9)	
CM1670～ CM1679	–	システム予約	
CM1680	R	ロギング･トレース0 ファイル保存カウンタ	○
CM1681	R	ロギング･トレース1 ファイル保存カウンタ	○
CM1682	R	ロギング･トレース2 ファイル保存カウンタ	○
CM1683	R	ロギング･トレース3 ファイル保存カウンタ	○
CM1684	R	ロギング･トレース4 ファイル保存カウンタ	○
CM1685	R	ロギング･トレース5 ファイル保存カウンタ	○
CM1686	R	ロギング･トレース6 ファイル保存カウンタ	○
CM1687	R	ロギング･トレース7 ファイル保存カウンタ	○
CM1688	R	ロギング･トレース8 ファイル保存カウンタ	○
CM1689	R	ロギング･トレース9 ファイル保存カウンタ	○

CM	属性 R:読出専用 空欄:読書可能	機能		保持 ○:保持 設定:保持設定可能
CM1690～ CM1699		システム予約		
CM1700	R	AutoLoadフォルダ番号(要求)		○
CM1701	R	RunLoadフォルダ番号(要求)		○
CM1702	R	プロジェクトロード時フォルダ番号(要求)		○
CM1703	R	プロジェクトセーブ時フォルダ番号(要求)		○
CM1704～ CM1709		システム予約		
CM1710	R	電源ON時オートローディング完了コード		
CM1711	R	電源ON時オートローディングフォルダ番号(完了)		
CM1712	R	PROG→RUN時オートローディング完了コード		○
CM1713	R	PROG→RUN時オートローディングフォルダ番号(完了)		○
CM1714	R	プロジェクトロード時完了コード		○
CM1715	R	プロジェクトロード時フォルダ番号(完了)		○
CM1716	R	プロジェクトセーブ時完了コード		○
CM1717	R	プロジェクトセーブ時フォルダ番号(完了)		○
CM1718～ CM1719	–	システム予約		
CM1720		アクセスウィンドウ	ユーザーメッセージ1	○
CM1721～ CM1737			ユーザーメッセージ2 (最大24文字＋終端コード(00H))	○
CM1738			AW 操作禁止設定	○
CM1739			システム予約	
CM1740～ CM1756			ラダープロジェクト名 (最大32文字＋終端コード(00H))	
CM1757～ CM1763			AW表示用プロジェクト名 (最大12文字＋終端コード(00H))	
CM1764～ CM1779			システム予約	
CM1780			AW初期画面設定 有効	○
CM1781			AW初期画面設定 デバイス種別	○
CM1782			AW初期画面設定 画面ID	○
CM1783			AW初期画面設定 画面ID	○
CM1784			AW初期画面設定 デバイス番号	○
CM1785			AW初期画面設定 デバイス番号	○
CM1786			AW初期画面設定 表示形式	○
CM1787			AW初期画面設定 デバイス名表記	○
CM1788			AW初期画面設定 キーロック状態	○
CM1789			AW初期画面設定 表示言語	○
CM1790～ CM2199	–		システム予約	
CM2200		最新重度エラー詳細情報1件	年月	○
CM2201			日	○
CM2202			時	○
CM2203			分	○
CM2204			秒	○
CM2205			番号	○
CM2206			詳細情報有効データ数	○
CM2207			詳細情報01	○
CM2208			詳細情報02	○
:				○
CM2226			詳細情報20	○
CM2227～ CM2249			システム予約	

CM	属性 R:読出専用 空欄:読書可能	機能		保持 ○:保持 設定:保持設定可能
CM2250		最新軽度エラー詳細情報1件※1	年月	○
CM2251			日	○
CM2252			時	○
CM2253			分	○
CM2254			秒	○
CM2255			番号	○
CM2256			詳細情報有効データ数	○
CM2257			詳細情報01	○
CM2258			詳細情報02	○
:				○
CM2276			詳細情報20	○
CM2277～CM2388	–	システム予約		
CM2389	R	拡張ユニット接続台数		
CM2390	R	メモリカード命令用エラーコード		
CM2391		メモリカードアクセス中電源OFF		○
CM2392		メモリカードリトライ発生回数		○
CM2393		メモリカードベリファイエラー発生回数		○
CM2394		メモリカードタイムアウト発生回数		○
CM2395	–	メモリカードエラー時リセット遅延時間		○
CM2396～CM2399	–	システム予約		
CM2400～CM2415		割り込み回数(INT0～INT3、CTC0～CTC7、位置決め軸1～軸4)		
CM2416～CM4799	–	システム予約		
CM4800		高速カウンタ機能(CTH0)	回転方向検出時定数	○
CM4801			イネーブル、クリア、符号設定	○
CM4802～CM4803			リングカウンタ下限値	○
CM4804～CM4805			リングカウンタ上限値	○
CM4806～CM4807			プリセット入力	○
CM4808～CM4809	–		システム予約	
CM4810		高速カウンタ機能(CTH1)	回転方向検出時定数	○
CM4811			イネーブル、クリア、符号設定	○
CM4812～CM4813			リングカウンタ下限値	○
CM4814～CM4815			リングカウンタ上限値	○
CM4816～CM4817			プリセット入力	○
CM4818～CM4819	–		システム予約	
CM4820		高速カウンタ機能(CTH2)	回転方向検出時定数	○
CM4821			イネーブル、クリア、符号設定	○
CM4822～CM4823			リングカウンタ下限値	○
CM4824～CM4825			リングカウンタ上限値	○
CM4826～CM4827			プリセット入力	○

CM	属性 R:読出専用 空欄:読書可能	機能		保持 ○:保持 設定:保持設定可能
CM4828～ CM4829	–	高速カウンタ機能(CTH3)	システム予約	
CM4830			回転方向検出時定数	○
CM4831			イネーブル、クリア、符号設定	○
CM4832～ CM4833			リングカウンタ下限値	○
CM4834～ CM4835			リングカウンタ上限値	○
CM4836～ CM4837			プリセット入力	○
CM4838～ CM4839	–		システム予約	○
CM4840～ CM4899	–	システム予約		
CM4900～ CM4901	R	周波数カウンタ機能 (CTH0)	周波数計測結果(Hz or rpm)	
CM4902			1回転のパルス数	○
CM4903			スキャン回数	○
CM4904			外部入力ソース	○
CM4905	–		システム予約	
CM4906		指定周波数パルス出力機能 (CH0)	設定周波数(Hz)	○
CM4908			Duty比(0～100%)	○
CM4909	–		システム予約	
CM4910～ CM4911	R	周波数カウンタ機能 (CTH1)	周波数計測結果(Hz or rpm)	
CM4912			1回転のパルス数	○
CM4913			スキャン回数	○
CM4914			外部入力ソース	○
CM4915	–		システム予約	
CM4916		指定周波数パルス出力機能 (CH1)	設定周波数(Hz)	○
CM4918			Duty比(0～100%)	○
CM4919	–		システム予約	
CM4920～ CM4921	R	周波数カウンタ機能 (CTH2)	周波数計測結果(Hz or rpm)	
CM4922			1回転のパルス数	○
CM4923			スキャン回数	○
CM4924			外部入力ソース	○
CM4925	–		システム予約	
CM4926		指定周波数パルス出力機能 (CH2)	設定周波数(Hz)	○
CM4928			Duty比(0～100%)	○
CM4929	–		システム予約	
CM4930～ CM4931	R	周波数カウンタ機能 (CTH3)	周波数計測結果(Hz or rpm)	
CM4932			1回転のパルス数	○
CM4933			スキャン回数	○
CM4934			外部入力ソース	○
CM4935	–		システム予約	
CM4936		指定周波数パルス出力機能 (CH3)	設定周波数(Hz)	○
CM4938			Duty比(0～100%)	○
CM4939			システム予約	
CM4940～ CM4999	–	システム予約		
CM5000～ CM5999		内蔵シリアル割付用		

CM	属性 R:読出専用 空欄:読書可能	機能		保持 ○:保持 設定:保持設定可能
CM6000～ CM6999		増設機能割付用1		
CM7000～ CM7999		増設機能割付用2		
CM8000～ CM8001		CPU位置決め機能軸1 ポイントパラメータ1	目標値/移動量	設定
CM8002			加速レート	設定
CM8003			減速レート	設定
CM8004～ CM8005			運転周波数	設定
CM8006			動作モード	設定
CM8007			システム予約	設定
CM8008			停止センサ入力後移動量	設定
CM8009	–		システム予約	
CM8010～ CM8199		CPU位置決め機能軸1	ポイントパラメータ2～20 (CM8000～CM8009と同等)	設定
CM8200～ CM8201		CPU位置決め機能軸2 ポイントパラメータ1	目標値/移動量	設定
CM8202			加速レート	設定
CM8203			減速レート	設定
CM8204～ CM8205			運転周波数	設定
CM8206			動作モード	設定
CM8207			システム予約	設定
CM8208			停止センサ入力後移動量	設定
CM8209	–		システム予約	
CM8210～ CM8399		CPU位置決め機能軸2	ポイントパラメータ2～20 (CM8200～CM8209と同等)	設定
CM8400～ CM8401		CPU位置決め機能軸3 ポイントパラメータ1	目標値/移動量	設定
CM8402			加速レート	設定
CM8403			減速レート	設定
CM8404～ CM8405			運転周波数	設定
CM8406			動作モード	設定
CM8407			システム予約	設定
CM8408			停止センサ入力後移動量	設定
CM8409	–		システム予約	
CM8410～ CM8599		CPU位置決め機能軸3	ポイントパラメータ2～20 (CM8400～CM8409と同等)	設定
CM8600～ CM8601		CPU位置決め機能軸4 ポイントパラメータ1	目標値/移動量	設定
CM8602			加速レート	設定
CM8603			減速レート	設定
CM8604～ CM8605			運転周波数	設定
CM8606			動作モード	設定
CM8607			システム予約	設定
CM8608			停止センサ入力後移動量	設定
CM8609	–		システム予約	
CM8610～ CM8799		CPU位置決め機能軸4	ポイントパラメータ2～20 (CM8600～CM8609と同等)	設定

CM	属性 R:読出専用 空欄:読書可能	機能		保持 ○:保持 設定:保持設定可能
CM8800		CPU位置決め機能軸1	入出力設定	
CM8801			センサイネーブル	
CM8802～ CM8803			現在値コンパレータ0	
CM8804～ CM8805			現在値コンパレータ1	
CM8806～ CM8807			位置決めコンパレータ2	
CM8808			起動速度	
CM8809			原点復帰:起動周波数	
CM8810			原点復帰:加速レート	
CM8811			原点復帰:減速レート	
CM8812～ CM8813			原点復帰:運転周波数	
CM8814			原点復帰詳細設定	
CM8815			JOG:起動周波数	
CM8816			JOG:加速レート	
CM8817			JOG:減速レート	
CM8818～ CM8819			JOG:運転周波数	
CM8820～ CM8821			ホームポジション	
CM8822～ CM8823	–		システム予約	
CM8824～ CM8825			現在座標変更設定値	
CM8826～ CM8827			速度制御モード時の運転速度変更設定値	
CM8828～ CM8829			目標座標変更設定値	
CM8830～ CM8831	R		出力パルス現在座標	
CM8832～ CM8833	R		出力周波数現在値	
CM8834	R		エラーコード	
CM8835	R		実行ポイント番号	
CM8836～ CM8839	–		システム予約	

CM	属性 R:読出専用 空欄:読書可能	機能		保持 ○:保持 設定:保持設定可能
CM8840		CPU位置決め機能軸2	入出力設定	
CM8841			センサイネーブル	
CM8842～ CM8843			現在値コンパレータ0	
CM8844～ CM8845			現在値コンパレータ1	
CM8846～ CM8847			位置決めコンパレータ2	
CM8848			起動速度	
CM8849			原点復帰:起動周波数	
CM8850			原点復帰:加速レート	
CM8851			原点復帰:減速レート	
CM8852～ CM8853			原点復帰:運転周波数	
CM8854			原点復帰詳細設定	
CM8855			JOG:起動周波数	
CM8856			JOG:加速レート	
CM8857			JOG:減速レート	
CM8858～ CM8859			JOG:運転周波数	
CM8860～ CM8861			ホームポジション	
CM8862～ CM8863	–		システム予約	
CM8864～ CM8865			現在座標変更設定値	
CM8866～ CM8867			速度制御モード時の運転速度変更設定値	
CM8868～ CM8869			目標座標変更設定値	
CM8870～ CM8871	R		出力パルス現在座標	
CM8872～ CM8873	R		出力周波数現在値	
CM8874	R		エラーコード	
CM8875	R		実行ポイント番号	
CM8876～ CM8879	–		システム予約	

CM	属性 R:読出専用 空欄:読書可能	機能		保持 ○:保持 設定:保持設定可能
CM8880		CPU位置決め機能軸3	入出力設定	
CM8881			センサイネーブル	
CM8882～ CM8883			現在値コンパレータ0	
CM8884～ CM8885			現在値コンパレータ1	
CM8886～ CM8887			位置決めコンパレータ2	
CM8888			起動速度	
CM8889			原点復帰:起動周波数	
CM8890			原点復帰:加速レート	
CM8891			原点復帰:減速レート	
CM8892～ CM8893			原点復帰:運転周波数	
CM8894			原点復帰詳細設定	
CM8895			JOG:起動周波数	
CM8896			JOG:加速レート	
CM8897			JOG:減速レート	
CM8898～ CM8899			JOG:運転周波数	
CM8900～ CM8901			ホームポジション	
CM8902～ CM8903			システム予約	
CM8904～ CM8905			現在座標変更設定値	
CM8906～ CM8907			速度制御モード時の運転速度変更設定値	
CM8908～ CM8909			目標座標変更設定値	
CM8910～ CM8911	R		出力パルス現在座標	
CM8912～ CM8913	R		出力周波数現在値	
CM8914	R		エラーコード	
CM8915	R		実行ポイント番号	
CM8916～ CM8919			システム予約	

CM	属性 R:読出専用 空欄:読書可能	機能		保持 ○:保持 設定:保持設定可能
CM8920		CPU位置決め機能軸4	入出力設定	
CM8921			センサイネーブル	
CM8922～ CM8923			現在値コンパレータ0	
CM8924～ CM8925			現在値コンパレータ1	
CM8926～ CM8927			位置決めコンパレータ2	
CM8928			起動速度	
CM8929			原点復帰:起動周波数	
CM8930			原点復帰:加速レート	
CM8931			原点復帰:減速レート	
CM8932～ CM8933			原点復帰:運転周波数	
CM8934			原点復帰詳細設定	
CM8935			JOG:起動周波数	
CM8936			JOG:加速レート	
CM8937			JOG:減速レート	
CM8938～ CM8939			JOG:運転周波数	
CM8940～ CM8941			ホームポジション	
CM8942～ CM8943			システム予約	
CM8944～ CM8945			現在座標変更設定値	
CM8946～ CM8947			速度制御モード時の運転速度変更設定値	
CM8948～ CM8949			目標座標変更設定値	
CM8950～ CM8951	R		出力パルス現在座標	
CM8952～ CM8953	R		出力周波数現在値	
CM8954	R		エラーコード	
CM8955	R		実行ポイント番号	
CM8956～ CM8959			システム予約	

- No.128(演算エラー)の場合

詳細情報n	説　明
詳細情報1	**エラーの詳細内容** 0　(未使用) 1　間接指定が不適切 2　間接指定先がT/C 3　簡易間接指定が不適切 4　オペランド範囲外 5　浮動小数点型実数オーバーフロー 6　デバイス指定が不適切 7　ROOT命令の引数が負 8　ENCO命令でONしているビットがない 9　浮動小数点型実数フォーマット異常 10　三角関数命令の引数が不適切 11　変換できないデータ 12　テーブル指定が不適切 13　時刻比較命令の引数が不適切 14　カムスイッチオペランドが不適切 15　周波数カウンタ命令の引数が不適切 16　0除算 17　ロギングID不正 18　ダイレクト処理エラー 19　ダイレクト処理できないユニット 20　ジャンプ先サブルーチンが存在しない 21　RUN中書き込み中なので命令が実行不可 22　拡張ユニット専用命令対象エラー(ユニット番号) 23　システム予約 24　PIDAT 動作設定範囲外 25　PIDAT サンプリング周期範囲外 26　PIDAT 比例定数範囲外 27　PIDAT 積分定数範囲外 28　PIDAT 微分定数範囲外 29　PIDAT 比例定数2自由度化係数範囲外 30　PIDAT 積分定数2自由度化係数範囲外 31　PIDAT 微分定数2自由度化係数範囲外 32　PIDAT 操作量上限値･下限値範囲外 33　PIDAT 操作量変化幅制限値範囲外 34　PIDAT 測定値変化幅範囲外 35　PIDAT AT状態遷移異常 36　PIDAT 動作制御フラグ範囲外 37　演算周期オーバー 38　PIDAT AT調整パラメータ範囲外 39　PIDAT ATヒステリシス範囲外 40　PIDAT ATタイムアウト時間範囲外 41　PIDAT AT算出制御方式設定範囲外 42　PIDAT ATタイムアウト 43　PIDAT AT比例定数オーバー 44　PIDAT AT積分定数オーバー 45　PIDAT AT微分定数オーバー 46　負の数の累乗根は演算できない 47　ABS命令の引数が不適切 48　演算オーバーフロー 49　UR/UM/UV ユニット未接続 50　ユニットデバイス(R) 未割付 51　ユニットデバイス(DM) 未割付

詳細情報n	説　明
詳細情報2	00H固定
詳細情報3	エラーが発生しているステップ番号上位
詳細情報4	エラーが発生しているステップ番号下位
詳細情報5～20	モジュール名をSJISで格納します。(モジュール名以外の部分は00H)

- No.50,53,55,56,58,59(ユニット番号に関するエラー)の場合

詳細情報n	説　明
詳細情報1	ユニット番号(エラーの発生状況によってユニット番号が特定できない場合には、格納されません。)
詳細情報2～20	00H固定

- No.54(ユニットの台数に関するエラー)の場合

詳細情報n	説　明
詳細情報1	ユニット設定情報でのユニットの台数
詳細情報2	実際に接続されているユニットの台数
詳細情報3～20	00H固定

- No.60(拡張I/O点数に関するエラー)の場合

詳細情報n	説　明
詳細情報1	基本ユニットの拡張I/O点数上限
詳細情報2	実際に接続されている拡張I/O点数
詳細情報3～20	00H固定

- No.61(割付デバイス重複に関するエラー)の場合

詳細情報n	説　明
詳細情報1	割付が重複している拡張ユニットのユニット番号
詳細情報2	重複しているデバイス種別(R(リレー)の場合:1、:DMの場合:2)
詳細情報3	重複しているデバイス番号 (デバイス種別がR(リレー)の場合はチャンネル番号(R1000の場合:10)
詳細情報4～20	00H固定

- No.96、97, 98, 99, 101(増設アダプタに関するエラー)の場合

詳細情報n	説　明
詳細情報1	増設アダプタ番号(左の場合:1、右の場合:2) (特定できない場合には0を格納)
詳細情報2	(システム予約)
詳細情報3～20	00H固定

- No.127(オートローディング失敗エラー)の場合

詳細情報n	説　明
詳細情報1	エラーの詳細内容 詳細については、「ロード/セーブ実行時のエラーコード」を参照してください。
詳細情報2～20	00H固定

- No.129(ユニットエラー)の場合

拡張情報n	解　説
拡張情報1	ユニット番号
拡張情報2	ユニット毎のエラー番号
拡張情報3～20	00H固定

- 上記以外のエラー番号の場合

拡張情報n	解　説
拡張情報1～20	00H固定

ASCIIコード表

ASCIIコードの一覧表です。

	上位4ビット															
下位4ビット	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	DLE	SP	0	@	P	`	p				ー	タ	ミ		
1	SOH	DC1	!	1	A	Q	a	q			。	ア	チ	ム		
2	STX	DC2	”	2	B	R	b	r			「	イ	ツ	メ		
3	ETX	DC3	#	3	C	S	c	s			」	ウ	テ	モ		
4	EOT	DC4	$	4	D	T	d	t			、	エ	ト	ヤ		
5	ENQ	NAK	%	5	E	U	e	u			・	オ	ナ	ユ		
6	ACK	SYN	&	6	F	V	f	v			ヲ	カ	ニ	ヨ		
7	BEL	ETB	’	7	G	W	g	w			ァ	キ	ヌ	ラ		
8	BS	CAN	(	8	H	X	h	x			ィ	ク	ネ	リ		
9	HT	EM	)	9	I	Y	i	y			ゥ	ケ	ノ	ル		
A	LF	SUB	*	:	J	Z	j	z			ェ	コ	ハ	レ		
B	VT	ESC	+	;	K	[	k	{			ォ	サ	ヒ	ロ		
C	FF	→	,	<	L	\	l	|			ャ	シ	フ	ワ		
D	CR	←	—	=	M	]	m	}			ュ	ス	ヘ	ン		
E	SO	↑	.	>	N	^	n	~			ョ	セ	ホ	゛		
F	SI	↓	／	?	O	＿	o	DEL			ッ	ソ	マ	゜		

使用できない文字一覧

プロジェクト名、モジュール名、マクロ名、ラベル、デバイスコメントで使用できない文字を説明します。

● プロジェクト名に使用できない文字

対応機種にKV-8000/7500/7300/5500/5000/3000、KV Nanoシリーズを選択した場合

使用できない文字	! ″ # $ % & ‘ * , . / : ; < = > ?
	@ ¥ ^ ` | ~ (スペース)

対応機種にKV-1000を選択した場合

使用できない文字	! ″ # $ % & ‘ ( ) * , . / : ; < = > ?
	@ [¥] ^ ` { | } ~ (スペース)
先頭に使用できない文字	+ − 0 1 2 3 4 5 6 7 8 9 _
末尾に使用できない文字または文字列	_ _0 _1 _2 _3 _4 _5 _6 _7 _8 _9
使用できない文字列	ｸﾞﾛｰﾊﾞﾙ グローバル

対応機種にKV-700/700＋M、KV-P16、KV-10(16)、KV-24(40)を選択した場合

使用できない文字	¥ / : * ? ″ < > |

● プログラム名(モジュール名、ファンクションブロック名、マクロ名)に使用できない文字

対応機種にKV-8000/7500/7300/5500/5000/3000/1000、KV Nanoシリーズを選択した場合

使用できない文字	! ″ # $ % & ’ ( ) * , . / : ; < = > ?
	@ [¥] ^ ` { | } ~ (スペース) (__)※
先頭に使用できない文字	+ − 0 1 2 3 4 5 6 7 8 9 _
末尾に使用できない文字または文字列	_ _0 _1 _2 _3 _4 _5 _6 _7 _8 _9
使用できない文字列	ｸﾞﾛｰﾊﾞﾙ グローバル

※ KV Nanoシリーズを選択した場合のみ、使用できます。

対応機種にKV-700/700＋M、KV-P16、KV-10(16)、KV-24(40)を選択した場合

使用できない文字	¥ / : * ? ″ < > |

● 変数/ラベル名に使用できない文字

使用できない文字	! ″ # $ % & ’ ( ) * + , − . / : ; < =
	> ? @ [¥] ^ ` { | } ~ (スペース)
先頭に使用できない文字※1	0 1 2 3 4 5 6 7 8 9 _ U _
使用できない文字列	ｸﾞﾛｰﾊﾞﾙ※1 グローバル※1
	(命令語と同じ名前)
	(デバイス名)
	• アルファベット1文字＋ R または M ＋数字 • アルファベット1文字＋数字 • X ＋(A ～ F) • Y ＋(A ～ F)

使用できない文字列	• KVスクリプトで規定された以下のキーワード !!, #PRAGMA, ', (,), *, *=, +, +=, -, -=, .B, .D, .DF, .F, .L, .S, .T, .U, /, /=, <, <=, <>, =, >, >=, ABS, ACNT※2, ACOS, ADRADD, ADRDEC, ADRINC, ADRSET, ADRSUB, AJST, AND, ANDA, APR, AS, ASC, ASIN, ASLA, ASRA, ATAN, ATAN2, AVG, AWHIDE, AWSHOW, B, BANDC, BCMP, BCMPI, BCNT, BLD, BLDB, BMOV, BOUB, BOUT, BREAK, BRES, BSET, BSL, BSR, BSUM, BSWAP, BYBMOV, BYLMOV, C, CALL, CASE, CHR, CM, CNT, COM, COS, CPMGET, CPMSET, CPSASC, CR, CRC, CTC, CTH, DASC, DCNT, DEC, DECO, DEG, DELETE, DFLOAT, DI, DIC, DIM, DISB, DISF, DISN, DISS, DM, DMX, DO, DOUBLE, DR, DSER, ECALL, EI, ELSE, ELSE IF, EM, ENCO, END, END IF, END SELECT, END TYPE, END WHILE, ENRA, EORA, EXP, FALSE, FASC, FB, FBCALL※2, FBSTRT※2, FDEL, FEND, FIFOR, FIFOW, FIND, FINS, FLOAT, FM, FMOV, FOR, FRLDM, FRSET, FRSTM, FUN, FWRIT, GRY, HASC, IEDGE, IF, INC, INSERT, INSTR, INT, INTG, IS, IS_7500, IS_8000, IS_AD40, IS_AD40G, IS_AD40V, IS_AM40V, IS_B16R, IS_B16T, IS_B16X, IS_B8RC, IS_B8XTD, IS_C16XTD, IS_C32T, IS_C32X, IS_C32XTD, IS_C64T, IS_C64X, IS_CA02, IS_CL20, IS_DA40, IS_DA40V, IS_DN20, IS_EP21V, IS_FL20V, IS_H20G, IS_H20S, IS_H40S, IS_L21V, IS_LE21V, IS_LH20V, IS_LM21V, IS_MC20V, IS_MC40V, IS_ML16V, IS_MVL40, IS_N20V, IS_SAD04, IS_SC20, IS_SDA04, IS_SFP24XT, IS_SFP32XT, IS_SH04PL, IS_SIR32XT, IS_SSC02, IS_TF40, IS_TP40, IS_XCA02, IS_XCM02※2, IS_XH04ML, IS_XH16EC, IS_XH16ML, IS_XL202, IS_XL402, IS_XLE02, LDF, LDFB, LDP, LDPB, LEFT, LEN, LIFOR, LIFOW, LIMIT, LLFLT, LOG, LOG10, LOGD, LOGE, LR, LTRIM, MAX, MC, MCALL, MCOPY, MCR, MDEL, MDSTOP, MDSTRT, MEND, MFREE, MFREEK, MID, MIN, MMKDIR, MMOV, MOD, MPRINT, MPX, MR, MREAD, MREADL, MREN, MRMDIR, MSTAT, MSTRT, MWRIT, NEG, NEXT, NOT, OFF, ON, OR, ORA, P, PIDAT, PMOV, R, RAD, RAMP, RASC, RCPSASC, RDASC, REM, REPLACE, RES, RFASC, RFSCI, RFSCO, RFSFRC, RFSX, RFSY, RGRY, RHASC, RIGHT, RLA, RLNCA, RND, ROOT, RRA, RRNCA, RSEC, RTRIM, SDEL, SEC, SEG, SELECT, SELECT CASE, SER, SET, SFIND, SFINDN, SIN, SINS, SLA, SLEFT, SMID, SORT, SORTN, SPLIT, SPRD, SPWR, SQRT, SRA, SRGHT, SRPLC, SSVC, STEP, STR, STRIM, STRWORK, SWAP, T, TAN, TBCD, TBIN, THEN, TM, TMH, TMR, TMS, TMU, TO, TOD, TODF, TOF, TOL, TOS, TOU, TPOUT, TRGD, TRGR, TRIM, TRUE, TYPE, UFILL, UFRSM, UFSUS, UM, UMALLOC, UNIB, UNIF, UNIN, UNIS, UNTIL, UPSTOP, UPSTRT, UR, UREAD, UV, UWRIT, U_CCTRCV, U_CCTSND, U_CCTST, U_CCTTO, U_CERR, U_CREG, U_CSTOP, U_CSTRT, U_DISOCP, U_DISOVP, U_ELCYC, U_ELSTAT, U_FCSTAT, U_MLATTA, U_MLSTAT, U_MLSUB, U_MLTEXT, U_MLTO, U_MLTOGR, U_MSGRCV, U_MSGSND, U_MSGST, U_MSGTO, U_NDSTAT, U_NDTO, U_PLCYC, U_PLSTAT, U_PRDBUF, U_RDAD, U_RDADB, U_RDAERC, U_RDBF, U_RDCNT, U_RDCPOS, U_RDCPT0, U_RDCPT1, U_RDCVEL, U_RDFPOS, U_RDFTRQ, U_RDFVEL, U_RDHZ, U_RDMD, U_RDMERC, U_RDPMC, U_RDPPB, U_RDPPNT, U_RDPSB, U_RDRPM, U_RDSERC, U_RDSPB, U_RDSVPB, U_RDSYSB, U_SBMON, U_SBNAME, U_SELSP, U_SLREG, U_SOPEN, U_SRCVNP, U_SRDBUF, U_SSELPT, U_SSTAT, U_STMON, U_SUDPTO, U_SWRBUF, U_WRAVG, U_WRBNK, U_WRCNT,

	• KVスクリプトで規定された以下のキーワード U_WRCPOS, U_WRCTC0, U_WRCTC1, U_WRDA, U_WRFBNO, U_WRFNO, U_WRFRNO, U_WRFVR, U_WRLMT, U_WRMD, U_WROFST, U_WRPOR, U_WRPPB, U_WRPPNT, U_WRPPS, U_WRPPSS, U_WRPSB, U_WRPST, U_WRPVEL, U_WRSCAM, U_WRSCL, U_WRSCOF, U_WRSCON, U_WRSPB, U_WRSVPB, U_WRSYSB, U_WRTTRQ, U_WRTVL, U_WRVTL, U_WRVVEL, V, VAL, W, WHILE, WSIZE※2, WSL, WSR, WSUM, WTIME, XOR, Z, ZF, ZONE, ZPOP, ZPUSH, ZRES, [, \\, \\A, \\B, \\F, \\N, \\R, \\T, \\V,]
使用できない文字列	• IEC61131-3で規定された以下のキーワード TRUE, FALSE TIME, DATE,TIME_OF_DAY, TOD, DATE_AND_TIME, DT BOOL, SINT, INT, DINT, LINT, USINT, UINT, UDINT, ULINT, REAL, LREAL, STRING, WSTRING, BYTE, WORD, DWORD, LWORD ANY, ANY_DERIVED, ANY_ELEMENTARY, ANY_MAGNITUDE, ANY_NUM, ANY_REAL, ANY_INT, ANY_BIT, ANY_STRING, ANY_DATE ACTION, END_ACTION, ARRAY, OF, AT, CASE, END_CASE, CONFIGURATION, END_CONFIGURATION, CONSTANT, EN, ENO, EXIT, F_EDGE, FOR, TO, BY, DO, END_FOR, FUNCTION, END_FUNCTION, FUNCTION_BLOCK, END_FUNCTION_BLOCK, IF, THEN, ELSIF, ELSE, END_IF, INITIAL_STEP, END_STEP, NOT, MOD, AND, XOR, OR, PROGRAM, WITH, END_PROGRAM, R_EDGE, READ_ONLY, READ_WRITE, REPEAT, UNTIL, END_REPEAT, RESOURCE, END_RESOURCE, RETAIN, NON_RETAIN, RETURN, STEP, STRUCT, END_STRUCT, TASK, TRANSITION, FROM, END_TRANSITION, TYPE, END_TYPE, VAR,VAR_INPUT, VAR_OUTPUT, VAR_IN_OUT, VAR_TEMP, VAR_EXTERNAL, VAR_ACCESS, VAR_CONFIG,VAR_GLOBAL, END_VAR, WHILE, DO, END_WHILE • その他(IEC61131-3の仕様) • ラベル名の中に、2つ以上連続した"_"が現れてはならない。 "a_b_c"はできますが、"a__bc"は使用できません。

※1　対応機種に「KV-1000、KV-700/700 ＋ M、KV-10(16)、KV-24(40)、KV-P16」を選択した場合、一部の文字列は使用できます。

※2　CPU機能バージョン2.0以降のKV-8000でのみ使用できない文字列です。

● 構造体のメンバ名に使用できない文字

使用できない文字	( ) * / + - < > = & ! ¥ ″ # $ % % & | ' ~ ` @ [ ] { } : ; , . ? ^ (半角スペース) (全角スペース) (__)
先頭に使用できない文字	0 1 2 3 4 5 6 7 8 9 _
末尾に使用できない文字または文字列	_ _0 _1 _2 _3 _4 _5 _6 _7 _8 _9
使用できない文字列	ｸﾞﾛｰﾊﾞﾙ グローバル Global 全局 全域 글로벌 • 数値と区別できない文字列 　• K 　• K + 数値 　• K + 数値 + サフィックス(U/S/D/L/F/DF) 　• K + 数値 + E 　• K + 数値 + E + 数値 　• K + 数値 + E + 数値 + サフィックス(F/DF) 　• H + 16進数値(10進数値、もしくはABCDEFの文字列) • KVスクリプトで規定されたのキーワード （キーワードの内容は変数/ラベル名に使用できない文字　を参照してください） • IEC61131-3で規定されたキーワード （キーワードの内容は変数/ラベル名に使用できない文字　を参照してください）

● デバイスコメントに使用できない文字

使用できない文字	,

使用できない関数一覧

初期化モジュール、ファンクションブロック、マクロ、割り込みプログラム内で使用できない関数を説明します。

●初期化モジュール、割り込みプログラム

関数の種類	関数	
接点関数	LDP/LDF	ロードパルス/ロードパルフ
	LDPB/LDFB	ロードパルスバー/ロードパルフバー
タイマ･カウンタ関数	TMR/TMH/TMS/TMU	タイマ/高速タイマ/高速1msタイマ/高速10μsタイマ
	C	カウンタ
ファイルレジスタ関数	FRSTM/FRLDM	ファイルレジスター括保存/ファイレジスター括読出
ストレージ関数	MWRIT/MREAD/MFREE/MMKDIR	ストレージライト/ストレージリード/ストレージ容量取得/ストレージフォルダ作成
	MRMDIR/MDEL	ストレージフォルダ削除/ストレージファイル削除
	MPRINT/MREADL/MCOPY/MMOV/MREN/MFREEK/MSTAT	ストレージテキスト書き込み/ストレージ1行読み出し/ストレージファイルコピー/ストレージファイル移動/ストレージファイルリネーム/ストレージ容量取得/ストレージ状態取得
アクセスウィンドウ関数	AWNUM/AWMSG AWSHOW/AWHIDE	ユーザメッセージ1/ユーザメッセージ2 ユーザメッセージ表示/ユーザメッセージ表示解除

●ファンクションブロック

関数の種類	関数	
マクロ関数	MCALL/MSTRT	マクロコール/マクロスタート

※FEND（ファンクションブロックエンド）関数は、ファンクションブロック内に記述します。ファンクションやモジュールなどほかの種類のプログラムでは使用できません。

●ファンクション

関数の種類	関数	
マクロ関数	MCALL/MSTRT	マクロコール/マクロスタート
ファンクションブロック関数	FB/FBSTRT/FBCALL	ファンクションブロック

●マクロ

関数の種類	関数	
マクロ関数	MCALL/MSTRT	マクロコール/マクロスタート
ファンクションブロック関数	FB/FBSTRT/FBCALL/FUN	ファンクションブロック/ファンクション

※MEND（マクロエンド）関数は、自己保持型マクロ内に記述します。サブルーチン型マクロやモジュールなどほかの種類のプログラムでは使用できません。

索　引

A

AND 4-7
ASCII コード表 ..... 付 -53

B

BREAK 処理 4-31

C

CR ..... 付 -18

D

DO ～ UNTIL 4-29
　WHILE 文と DO 文の違いについて 4-30

F

FOR ～ NEXT（指定回数の繰り返し制御） 4-24
　FOR ～ NEXT 文を強制終了させる場合 4-26
　FOR ～ NEXT 文を使用するときの注意 4-26
　STEP で指定できる範囲 4-26
　STEP で単精度浮動小数点型定数を
　　指定する場合 4-26

H

HELP 機能 2-66

I

IF ～ 4-12
IF ～ ELSE 4-14
IF ～ ELSE IF 4-16

M

MC ～ MCR 4-22
MC ～ MCR にタイマ / カウンタ関数を
　使用する場合 4-45
MOD 4-6

N

NOT 4-7

O

OR 4-7

P

PRAGMA 指示命令 3-72

S

SELECT CASE 4-18

T

TYPE 3-69

W

WHILE ～ 4-27
WHILE 文を強制終了させる場合 4-28

X

XOR 4-7
XYM 表記 2-77

あ

アウトプットウィンドウの使いかた 2-56
アウトプットウィンドウのエラー表示 2-55

い

インデックス修飾 3-55
　インデックス定数による修飾 3-56
　インデックスレジスタによる修飾 3-55
　サフィックスの付いたデバイスの修飾 3-56
　デバイスや式による修飾 3-56
　不正なインデックス修飾 3-56
インデックス修飾と間接指定 3-54
インデックス修飾と間接指定を併用する 3-61
インデックスレジスタ　Z 3-44
インラインニモニック 2-72

う

ウィンドウ入力 2-3
ウォッチウィンドウ 2-63

え

エラーメッセージ 2-54
　アウトプットウィンドウのエラー表示 2-55
　関数引数間違い 2-54
　スクリプト変換時のエラー 2-54
　タイプミスをしている 2-54
　不正な型指定 2-54
　文法を間違えている 2-54
　変換エラーが発生したときは 2-55
エリアスクリプト 1-9, 2-16
エリアスクリプトの IF 文の中で
　タイマ関数を使用する 4-42
エリアスクリプトの特長 2-16
エリアスクリプトはこんな時に使う 2-16
エリアスクリプトをコピーするには 2-24
エリアスクリプトを削除するには 2-23
エリアスクリプトを作成する 2-18
エリアスクリプトを新規作成する 2-19
エリアスクリプトを編集するには 2-22
演算関数を使用する場合 1-4
演算子 1-11, 4-6
演算子一覧 1-11, 4-6
　算術演算子 4-6
　乗除算･加減算の優先順位 4-8
　代入文と比較演算子の違い 4-9
　比較演算子 4-6
　比較演算子の優先順位 4-8
　文字列連結演算子 4-6
　優先順位の入れ換え 4-8
　論理演算子 4-7
　論理演算子の優先順位 4-8
演算式で扱えるデータの範囲 3-70
演算式の型について 3-70
演算子の種類 4-6
演算子の優先順位 4-7
演算処理を KV スクリプトで記述する 1-3
演算子を使用する場合 1-3
演算代入文　+=,-=,*=,/= 4-3
演算フラグ 1-18

か

カウンタ(現在値)　C 3-42
カウンタ (接点)　C 3-33
カウンタの設定値･現在値を扱う方法 3-33
カウンタデバイスを使う時の注意
　タイマ / カウンタ現在値の読み出し 4-40
　タイマ / カウンタ接点 4-41
カウンタを使う時の注意 4-40
カスタマイズ 2-73
　スクリプトウィンドウのカスタマイズ 2-73
型(データの型(サフィックス)) 3-62
　型宣言(TYPE)について 3-69
　最大値･最小値を超える時の数値の変化 3-68
　デバイスの型宣言について 1-16
　不正な型指定 2-54
型変換によるデータの欠落 2-52
簡易間接指定　#TM 3-61
関数 1-17
　スクリプト関数一覧 2-64
　スクリプト関数書式ヒント 2-65
　スクリプト関数について 1-17
関数一覧の呼び出し 2-64
関数引数間違い 2-54
間接指定(＊)とは 3-57
間接指定デバイスの型 3-59
間接指定の記述方法 3-57
間接指定の仕組み 3-57
間接指定の使用方法 3-58
間接指定の使用例 3-60
間接指定やインデックス修飾時の注意事項 1-18
　不正なアドレスを指定したときの
　　各関数の動作一覧 1-18

き

機能バージョン差分 5
　CPU 機能バージョンによる CPU 機能差分 5
　CPU 機能バージョンによる
　　イーサネット機能差分 6
　拡張入出力ユニット使用時の制限事項 6
機能バージョンについて(KV Nano シリーズ) 4
機能バージョンについて(KV-5500/5000/3000) ...4
機能バージョンについて(KV-7500/7300) 3
機能バージョンについて(KV-8000) 2

く

繰り返し制御文 1-13, 4-11
繰返し制御文中のタイマ動作について 4-37
繰返し制御文と処理時間 4-35
グローバルラベル 2-38
グローバルラベル配列 2-47
グローバルラベルのビット 2-45
グローバルラベルを使った
　プログラミングの流れ 2-38
グローバルラベルを登録する 2-38

け

警告メッセージ 2-52
　警告レベルについて 2-53
　警告レベルの設定方法 2-53
　スクリプト警告レベルの設定 2-53

こ

高速カウンタ(現在値)　CTH 3-42
高速カウンタコンパレータ (設定値)　CTC ...3-43
高速カウンタコンパレータ (接点)　CTC 3-34
コメント機能について 2-69
コメントの書き方 2-69
コントロールメモリ CM 3-44
コントロールリレー .....付 -18
コントロールリレー CR 3-34

さ

最下位ビットでビットデバイスとして扱う 3-46
サフィックス 3-62
　.U と .S について 3-63
　.D と .L について 3-64
　.DF について 3-66
　.B について 3-67
　.T について 3-67
　.F について 3-65
サフィックスについて 1-14
サフィックスの種類 1-14
サフィックスの省略 1-15
サフィックスを使ったデバイスの記述例 1-15

し

条件分岐制御文 1-12
使用できない文字一覧 ..... 付 -54
　デバイスコメントに使用できない文字 ..... 付 -55
　プログラム名
　　（モジュール名、ファンクションブロック名、
　　マクロ名）に使用できない文字 ..... 付 -54
　プロジェクト名に使用できない文字 ..... 付 -54
　変数 / ラベル名に使用できない文字 ..... 付 -54
処理時間について 4-35

す

スクリプト関数一覧 2-64
スクリプト関数書式ヒント 2-65
スクリプト関数について 1-17
スクリプト記述時の注意事項 2-73
スクリプト警告レベルの設定 2-53
スクリプトとは 1-2
スクリプトの記述方法 2-2
スクリプトの機能 1-10
スクリプトの種類 1-8
スクリプトの特長 1-2
スクリプトの入力方法 2-3
　ウィンドウ入力 2-3
　直接入力 2-3
スクリプトの文法と注意 4-1
スクリプトの利用方法 1-3
スクリプト変換時のエラー 2-54
スクリプト変換時のメッセージ 2-52
スクリプト用ワークデバイスの割付設定 2-74
スクリプトを記述する時の注意 4-38
スクリプトを記述する時の注意事項 2-51
スクリプトを作成する 2-1

せ

制御文一覧 4-10
制御文について 1-12
　繰り返し制御文 1-13
　条件分岐制御文 1-12
制御文のネスティング（入れ子） 4-32
制御文を KV スクリプトで記述する 1-7
　複数の分岐がある制御をする場合 1-7
制御文を使う時の注意 4-32
　繰返し制御文中のタイマ動作について 4-37
　繰返し制御文と処理時間 4-35

た

代入文について 1-10
　数値代入 1-10
　ビット代入 1-10
　文字列代入 1-10
代入文の使用方法 4-2
　右辺と左辺で型が異なる場合 4-4
　演算代入文 4-3
　左辺に代入できない場合 4-4
　数値代入文 4-2
　ビット代入文 4-2
　文字列代入文 4-2
代入文を使用するときの注意 4-4
タイマ（現在値）　T 3-41
タイマ (接点)　T 3-31
タイマ関数使用時の注意 3-32, 4-41
　エリアスクリプトの IF 文の中で
　　タイマ関数を使用する 4-42
　ボックススクリプトで
　　タイマ関数を指定する 4-43
タイマ関数を使用する場合 2-14
タイマデバイスを使うときの注意
　タイマ / カウンタ現在値の読み出し 4-40
　タイマ / カウンタ接点 4-41
タイマの誤差 3-32
タイマの設定値･現在値を扱う方法 3-32
タイマを使うときの注意 4-40
単精度浮動小数点型定数 3-52

ち

直接入力 2-3

て

データの型（サフィックス） 3-62
データの取り扱い 3-1
データメモリ / 拡張データメモリ
　DM/EM/FM 3-35
定数 3-51
　単精度浮動小数点型定数 3-52
　倍精度浮動小数点型定数 3-53
　固定文字列（文字列型定数） 3-53
　$16 進定数 3-52
　(#)10 進定数 3-51
デバイス 3-2
デバイス番号一覧 ..... 付 -2
デバイス一覧（KV Nano シリーズの場合） 3-17
　KV Nano シリーズのビットデバイス 3-17
　KV Nano シリーズのワードデバイス 3-18
　定数 3-19
　内部レジスタ 3-19
　マクロ引数デバイス 3-19
　ローカルデバイス一覧 3-20
デバイス一覧
　（KV-5500/5000/3000/1000 の場合） 3-10
　KV-1000 のビットデバイス 3-12
　KV-1000 のワードデバイス 3-13
　KV-5500/5000/3000 のビットデバイス 3-10
　KV-5500/5000/3000 のワードデバイス 3-11
　定数 3-14
　内部レジスタ 3-14
　マクロ引数デバイス 3-14
　ローカルデバイス一覧（KV-1000 の場合） 3-16
　ローカルデバイス一覧
　　（KV-5500/5000/3000 の場合） 3-15
デバイス一覧（KV-7000 シリーズの場合）
　定数 3-8
　内部レジスタ 3-8
　KV-7000 シリーズのビットデバイス 3-6
　ファンクションブロック引数 3-8
　マクロ引数デバイス 3-8
　ローカルデバイス一覧 3-9
　KV-7000 シリーズのワードデバイス 3-7
デバイス一覧（KV-7000 シリーズの場合） 3-6
デバイス一覧（KV-8000 シリーズの場合）
　定数 3-4
　内部レジスタ 3-4
　KV-8000 シリーズのビットデバイス 3-2
　ファンクションブロック引数 3-4
　マクロ引数デバイス 3-4
　ローカルデバイス一覧 3-5
　KV-8000 シリーズのワードデバイス 3-3
 3-2
テンポラリデータメモリ　TM 3-41

と

登録モニタ 2-62

な

内部補助リレー　R･MR･LR 3-27

に

入力 / 出力リレー　R 3-24
入力補助機能について 2-64
入力（スクリプトの入力方法） 2-3
　ウィンドウ入力 2-3
　直接入力 2-3
任意のビット位置で
　ビットデバイスとして扱う 3-46

ね

ネスティングの実動作 4-34

は

倍精度浮動小数点型定数 3-53
配列 2-46

ひ

比較演算子 4-6
比較演算子の優先順位 4-8
ビット代入 1-10
ビット代入文 4-2
ビットデバイス 3-24
　B 3-28
　LR（ラッチリレー） 3-30
　MR 3-29
　R 3-27
　ビットの状態 3-24
　制御したビットデータの状態保持 4-38
ビットデバイスのワード単位扱い 3-45
微分実行型関数を使用する場合 2-13

ふ

ブール値 3-24
ファイルレジスタ　ZF/FM 3-37
ファンクションブロック 2-58
ファンクションブロック関数 6-10
ファンクションブロックの呼び出し 2-58
　ファンクションの呼び出し 2-58
ファンクションブロックを
　スクリプトで記述する 2-59
フロー関数 6-2, 6-4, 6-18

へ

HELP 機能 2-66
変換エラーが発生したときは 2-55
変数 3-49
変数のサフィックス 1-15

ほ

ボックススクリプト 1-8, 2-4
ボックススクリプトで
　タイマ関数を指定する 4-43
ボックススクリプトの特長 2-4
ボックススクリプトの並列接続について 2-51
ボックススクリプトはこんな時に使う 2-4
ボックススクリプトや IF 文中に
　タイマ関数を使用するときの注意 4-42
ボックススクリプトをコピーするには 2-12
ボックススクリプトを削除する 2-11
ボックススクリプトを作成する 2-6
ボックススクリプトを使用するときの注意 2-13
　タイマ関数を使用する場合 2-14
ボックススクリプトを新規作成する 2-7
ボックススクリプトを編集するには 2-11

ま

マクロ 2-60
マクロ関数 6-6
マクロの呼び出し 2-60
　サブルーチン型マクロの呼び出し 2-60
　自己保持型マクロの呼び出し 2-60
マクロをスクリプトで記述する 2-61

み

見やすいスクリプトを書く 2-70

む

無限ループについて 4-37

も

文字列演算子を使用する場合 1-5
文字列型定数 3-49, 3-51
　固定文字列（文字列型定数） 3-53
文字列処理関数を使用する場合 1-6
文字列処理を KV スクリプトで記述する 1-5
文字列代入 1-10
文字列代入文 4-2
文字列連結演算子 4-6
モニタ機能 2-62

ゆ

優先順位の入れ換え 4-8

よ

予約語 2-66

ら

ラベル使用時の注意事項 2-44
ラベル配列 2-46
ラベルの型 2-44
ラベルのサフィックス 1-16, 2-44
ラベルプログラミング 2-37

り

リレー 3-27
　内部補助リレー　R·MR·LR 3-27
　入力 / 出力リレー　R 3-24
リンクリレー　B 3-26
リンクレジスタ　W 3-39

ろ

ローカルラベル 2-41
ローカルラベル配列 2-49
ローカルラベルの一括登録 2-43
ローカルラベルを登録する 2-41

わ

ワークデバイス 2-74
ワークデバイスに不足が生じた場合 2-75
ワークデバイスの使用領域 2-75
ワードデバイス 3-35
ワードデバイスのビットデバイス扱い 3-46

関数索引

A

ABS 7-36
ACNT 6-60
ACOS 7-148
ADRADD 6-32
ADRDEC 6-30
ADRINC 6-28
ADRSET 6-26
ADRSUB 6-34
AJST 8-76
ANDA 7-22
APR 7-68
ASC 7-154
ASIN 7-146
ASLA 7-44
ASRA 7-42
ATAN 7-150
ATAN2 7-152
AVG 8-22
AWHIDE 8-123
AWMSG 8-120
AWNUM 8-118
AWSHOW 8-122

B

BANDC 7-64
BCMP 8-36
BCMPI 8-38
BCNT 8-6
BLD 5-8
BLDB 5-10
BMOV 7-2
BOUB 5-16
BOUT 5-14
BRES 5-20
BSET 5-18
BSL 7-60
BSR 7-58
BSUM 8-30
BSWAP 7-102
BYBMOV 7-10
BYLMOV 7-12

C

CALL 6-2
CHR 7-232
CNT 5-32
COM 7-30
COS 7-142
CPMGET 7-112
CPMSET 7-110
CPSASC 7-234
CRC 8-32

D

DASC 7-158
DCNT 8-8
DEC 7-16
DECO 7-104
DEG 7-138
DELETE 7-206
DFLOAT 7-118
DI 8-78
DIC 8-80
DISB 7-96
DISF 7-126
DISN 7-92
DISS 7-228
DMX 7-86
DSER 8-12

E

ECALL 6-4
EI 8-78
ENCO 7-106
ENRA 7-28
EORA 7-26
EXP 7-130

F

FASC 7-174
FB 6-10
FBCALL 6-11
FBSTRT 6-13
FDEL 8-68
FEND 6-15
FIFOR 8-52
FIFOW 8-48
FIND 7-226
FINS 8-66
FLOAT 7-114
FMOV 7-4
FRLDM 6-40
FRSET 6-36
FRSTM 6-38
FUN 6-16
FWRIT 8-64

G

GRY 7-88

H

HASC 7-168

I

IEDGE 8-82
INC 7-14
INSERT 7-202
INSTR 7-222
INT 7-124
INTG 7-122
IS_ □□□ 6-58

L

LDF 5-4
LDFB 5-7
LDP 5-2
LDPB 5-6
LEFT 7-190
LEN 7-182
LIFOR 8-60
LIFOW 8-56
LIMIT 7-62
LLFLT 7-78
LOG 7-132
LOG10 7-134
LOGD 8-86
LOGE 8-86
LTRIM 7-210

M

MAX 8-14
MCALL 6-6
MCOPY 8-110
MDEL 8-104
MDSTOP 6-19
MDSTRT 6-18
MEND 6-8
MFREE 8-94
MFREEK 8-96
MID 7-194
MIN 8-18
MMKDIR 8-98
MMOV 8-112
MPRINT 8-106
MPX 7-84
MREAD 8-92
MREADL 8-108
MREN 8-114
MRMDIR 8-102
MSTAT 8-116
MSTRT 6-7
MWRIT 8-90

N

NEG 7-32

O

ORA 7-24

P

PIDAT 8-84
PMOV 7-6

R

RAD 7-136
RAMP 7-74
RASC 7-156
RCPSASC 7-236
RDASC 7-166
REPLACE 7-199
RES 5-13
RFASC 7-178
RFSCI 8-132
RFSCO 8-134
RFSCTH 8-79
RFSFRC 8-124
RFSX 5-36
RFSY 5-37
RGRY 7-90
RHASC 7-172
RIGHT 7-186
RLA 7-48
RLNCA 7-52
RND 8-40
ROOT 7-18
RRA 7-46
RRNCA 7-50
RSEC 8-74
RTRIM 7-212

S

SDEL 7-204
SEC 8-72
SEG 8-2
SER 8-10
SET 5-12
SFIND 7-218
SFINDN 7-220
SIN 7-140
SINS 7-200
SLA 7-40
SLEFT 7-188
SMID 7-192
SORT 8-42
SORTN 8-44
SPLIT 7-216
SPRD 8-126
SPWR 8-128
SQRT 7-20
SRA 7-38
SRGHT 7-184
SRPLC 7-196
SSVC 8-130
STR 7-162
STRIM 7-214
SWAP 7-100

T

TAN 7-144
TBCD 7-80
TBIN 7-82
TMH 5-26
TMR 5-22
TMS 5-28
TMU 5-30
TOD 7-108
TODF 7-120
TOF 7-116
TOL 7-108
TOS 7-108
TOU 7-108
TPOUT 7-76
TRGD 8-87
TRGR 8-88
TRIM 7-208

U

U_ □□□□□ (ユニット専用関数)
KV-XLE02/EP21V 用
U_CSTOP ..... ※1
U_CSTRT ..... ※1
U_CREG ..... ※1
U_CERR ..... ※1
U_MSGTO ..... ※1
U_MSGSND ..... ※1
U_MSGRCV ..... ※1
U_MSGST ..... ※1
U_NDTO ..... ※1
U_NDSTAT ..... ※1
U_SBMON ..... ※1
U_SBNAME ..... ※1
U_STMON ..... ※1
U_SLREG ..... ※1
KV-XLE02/EP21V/LE21V/LE20V 用
U_ELCYC(KV-EP21V/LE21V) ..... ※1
U_ELSTAT(KV-EP21V/LE21V) ..... ※1
U_FCSTAT(KV-XLE02/EP21V/LE21V) ..... ※1
U_MLATTA ..... ※1
U_MLSTAT ..... ※1
U_MLSUB ..... ※1
U_MLTEXT ..... ※1
U_MLTO ..... ※1
U_MLTOGR ..... ※1
U_SOPEN ..... ※1
U_SRCVNP ..... ※1
U_SRDBUF ..... ※1
U_SSTAT ..... ※1
U_SUDPTO ..... ※1
U_SWRBUF ..... ※1
KV-XLE02/XL202/XL402 用
U_PLSTAT ..... ※1
U_PLCYC ..... ※1
U_PRDBUF ..... ※1
KV-XLE02 用
U_CCTTO ..... ※1
U_CCTSND ..... ※1
U_CCTRCV ..... ※1
U_CCTST ..... ※1
U_SSELPT ..... ※1
KV-SC20V/SSC02 用
U_RDCNT ..... ※1
U_RDCPT0 ..... ※1
U_RDCPT1 ..... ※1
U_RDHZ ..... ※1
U_RDRPM ..... ※1
U_WRCNT ..... ※1
U_WRCTC0 ..... ※1
U_WRCTC1 ..... ※1
U_WRPST ..... ※1
KV-SSC02 用
U_DISOCP ..... ※1
U_RDBF ..... ※1
KV-AD40V/KV-SAD04 用
U_RDAD ..... ※1
U_RDADB ..... ※1
U_WRAVG ..... ※1
U_WRLMT ..... ※1
U_WROFST ..... ※1
U_WRSCL ..... ※1
KV-SAD04 用
U_RDBF ..... ※1
KV-DA40V/KV-SDA04 用
U_WRDA ..... ※1
U_WRLMT ..... ※1
U_WROFST ..... ※1
U_WRSCL ..... ※1
KV-ML16V/MC40V/MC20V 用
U_WRPPNT ..... ※1
U_RDPPNT ..... ※1
U_RDPMC ..... ※1
U_RDFPOS ..... ※1
U_RDFVEL ..... ※1
U_RDFTRQ(KV-ML16V) ..... ※1
U_RDCPOS ..... ※1
U_RDCVEL ..... ※1
U_RDCNT.D ..... ※1
U_WRCPOS ..... ※1
U_WRPPS ..... ※1
U_WRPVEL ..... ※1
U_WRPOR ..... ※1
U_RDMD ..... ※1
U_WRMD(KV-ML16V) ..... ※1
U_WRVVEL(KV-ML16V) ..... ※1
U_WRVTL(KV-ML16V) ..... ※1
U_WRTTRQ(KV-ML16V) ..... ※1
U_WRTVL(KV-ML16V) ..... ※1
U_WRFRNO ..... ※1
U_WRFBNO ..... ※1
U_WRFNO ..... ※1
U_WRFVR ..... ※1
U_RDMERC ..... ※1
U_RDSERC ..... ※1
U_RDAERC ..... ※1
U_WRBNK ..... ※1
U_WRPPB ..... ※1
U_RDPPB ..... ※1
U_WRSVPB(KV-ML16V) ..... ※1
U_RDSVPB(KV-ML16V) ..... ※1
U_WRSYSB ..... ※1
U_RDSYSB ..... ※1
U_WRPSB ..... ※1
U_RDPSB ..... ※1
U_WRSPB ..... ※1
U_RDSPB ..... ※1
U_WRSCON ..... ※1
U_WRSCOF ..... ※1

U_WRSCAM ..... ※1
KV-SIR32XT 用
U_DISOCP ..... ※1
U_DISOVP ..... ※1
U_RDBF ..... ※1
KV-XH16ML/XH04ML/SH04PL 用
U_RDAERC ..... ※1
U_RDFPOS ..... ※1
U_RDFVEL ..... ※1
U_RDMD ..... ※1
U_RDPMC ..... ※1
U_RDPPNT ..... ※1
U_SELSP ..... ※1
U_WRCPOS ..... ※1
U_WRPOR ..... ※1
U_WRPPNT ..... ※1
U_WRPPS ..... ※1
U_WRPPSS ..... ※1
U_WRPVEL ..... ※1
KV-XH16ML/XH04ML 用
U_RDCPOS ..... ※1
U_RDCVEL ..... ※1
U_RDFTRQ ..... ※1
U_WRMD ..... ※1
U_WRTTRQ ..... ※1
U_WRTVL ..... ※1
U_WRVTL ..... ※1
U_WRVVEL ..... ※1
KV-SH04PL 用
U_DISOCP ..... ※1
UFILL 6-56
UFRSM 6-48
UFSUS 6-46
UMALLOC 6-50
UNIB 7-98
UNIF 7-128
UNIN 7-94
UNIS 7-230
UPSTOP 6-44
UPSTRT 6-42
UREAD 6-52
UWRIT 6-54

V

VAL 7-181

W

WSIZE 6-62
WSL 7-56
WSR 7-54
WSUM 8-26
WTIME 8-70

Z

ZONE 7-66
ZPOP 6-24
ZPUSH 6-20
ZRES 8-34

※1　各ユニットのユーザーズマニュアルを参照してください。
※2　各基本ユニットのユーザーズマニュアル参照してください。

改訂履歴

印刷年月日	版　　数	改　訂　内　容
2015年　2月	初　版	
2015年　4月	2　版	
2015年　12月	3　版	
2016年　6月	4　版	
2017年　6月	5　版	
2017年　9月	6　版	
2019年　4月	改訂1版	
2020年　4月	2訂1版	
2021年　7月	3訂1版	
2021年　9月	4訂1版	

保証について

1　対象製品

以下に規定する保証は、本マニュアルに記載している製品(以下「対象製品」という)に適用します。
なお、対象製品に内蔵されているリレーや電池などの消耗品は対象外とさせていただきます。

2　保証期間

対象製品の保証期間は、貴社のご指定場所に納入後3年間とします。ただし、次に該当する場合の保証期間は1年間とさせていただきます。

① 対象製品を海外でご購入いただいた場合。
② 海外で修理をご依頼いただいた場合。

3　保証範囲

(1) 上記保証期間内に当社の責任による故障が発生した場合は、無償での代替品との交換または修理をさせていただきます。ただし、保証期間内であっても、次に該当する故障の場合は保証対象外とさせていただきます。なお、代替品との交換または修理をおこなった場合でも保証期間の起算日は対象製品の当初ご納入日とさせていただきます。

① 取扱説明書、ユーザーズマニュアル、別途取り交わした仕様書などに記載された以外の不適当な条件･環境･取り扱い･使用方法に起因した故障。
② お客様の装置または、ソフトウエアの設計内容など、対象製品以外に起因した故障。
③ 当社以外による改造、修理に起因した故障。
④ 取扱説明書、ユーザーズマニュアルなどに記載している消耗部品が正しく保守、交換されていれば、防止できたと確認できる故障。
⑤ 当社出荷時の科学･技術水準では、予見が不可能だった事由による故障。
⑥ その他、火災、地震、水害などの災害及び電圧異常など当社の責任ではない外部要因による故障。

(2) 保証範囲は上記(1)を限度とし、対象製品の故障に起因するお客様での二次損害(装置の損傷、機会損失、逸失利益等)及びいかなる損害も保証の対象外とさせていただきます。

4　適用用途

当社製品は、一般工業向けの汎用品として設計･製造されております。
従いまして、下記のような用途での使用は意図しておりませんので適用外とさせていただきます。ただし、事前に当社までご相談いただき、お客様の責任において製品の仕様をご確認のうえ、定格･性能に対してご了承いただき、必要な安全対策を講じていただく場合は適用可能とさせていただきます。なお、この場合においても保証範囲は上記と同様といたします。

① 原子力発電、航空、鉄道、船舶、車両、医療機器等の人命や財産に多大な影響が予想される設備
② 電気、ガス、水道等の公共設備
③ 屋外での使用および、それに準ずる取扱説明書などで規定していない条件･環境での使用
④ 上記①及び②に準じる安全に関して高度な配慮と注意が要求される用途

全商品、送料無料で
当日出荷

必要な時に、必要な量だけ
在庫不要でトータルコストを削減

調べたいことを、いつでも、すぐに調べられる!

▼ 本社技術スタッフ直営 ▼ FAQ(よくあるご質問)

www.keyence.co.jp/seigyofaq ← CLICK!

マニュアルと合わせてご活用ください。

■ 制御システムに関するお問合せ 06-6379-1164

月～金曜(祝日を除く)8:30～20:00

■ ユーザー登録 www.keyence.co.jp/kvuser

ユーザー登録で以下のサービスが受けられます。

KV STANDARD SERVICE

- **無償バージョンアップ**
  専用ホームページから最新版ソフトをダウンロードできます。
- **サイトライセンス**
  ソフトひとつ分の費用で、事務所内のPCなら何台でも使用可能。

■ 輸出書類サービス www.keyence.co.jp/yushutsu

輸出に必要な書類をその場でダウンロードできます。

株式会社 キーエンス
本社・研究所／制御システム事業部
〒533-8555 大阪市東淀川区東中島1-3-14

仕様は改良のため予告なく変更することがあります。
記載されている会社名、製品名等は、それぞれ各社の商標または登録商標です。

制御10-2041

Copyright© 2021 KEYENCE CORPORATION.
All rights reserved.

2091-1 084659