<u>サーボコントローラ</u>

<u>SC2000シリーズ用サポートツール</u>

<u>日本語版ソフト簡易説明書</u> ScTool v2.0.9.7



株式会社 ワコー技研

表題

ページ番号

1.	はじめに	1
2.	インストール	2
З.	アンインストール	5
4.	ハードウェア	6
5.	メニュー	7
	ファイルメニュー	7
	表示メニュー	8
	特殊制御メニュー	9
	ウィンドウメニュー	10
	ヘルプメニュー	10
6	画面イメージ	11
7.	参考付録 Helpファイル内容	14

1. はじめに

このたび、汎用コントローラSC2000シリーズサポートツールをお買い上げいただきまして、 有難うございます。

ご使用の前に、この取扱説明書を良くお読みいただき、本システムを正しくご利用ください。

*このセットの構成は下記の通りです。

SC2000サポートツールインストールCD 1枚 インストール説明書(本書)1冊 メニュー概略説明書 1冊

*本ソフトウェアが動作するハードウェア構成は、下記の通りです。

IBM-PC/AT互換ハードウェア (INTEL又はAMD製CPU搭載機種)

RS232C通信ポート

(RS232C通信ポートが無いPCの場合、USB-SERIAL変換器が御利用になれます)

800x600 ピクセル表示が可能なVGAカード(オンボード可)及び表示装置 (推奨:1024x768ピクセル)

CD-ROMドライブ (ソフトインストール時のみ使用します)

ハードディスクに 10MBytes以上の空き容量

*本ソフトウェアは下記OSで動作致します。

Windows98(制限有り(*1)) Windows98SE(制限有り(*1)) Windows2000 Professional WindowsXP Pro/Home

他のOSでは正しくご利用になれませんので、御注意下さい。 (エミュレータ上では上記動作保証OSでも、動作保証外とさせて頂きます)

(*1)最大プログラムサイズが 32KBytesで制限されます。 Windows2000/XPに、この制限は有りません。 2. インストール

SC2000シリーズサポートツール(ScTool)をインストールするには以下の操作を行います。 (注意!! 既にインストールされているシステムに新しいバージョンをインストールする場合は、 先にアンインストールを行ってください)

・SC2000シリーズサポートツールのCD-ROMをCD-ROMドライブに挿入します。 インストーラが自動的に起動し、下記画面が表示されます。

👘 ScTool	
ScTool セットアップ ウィザードへようこそ	WACOGIĶEN
インストーラは ScTool をコンピュータ上にインストールするために必要な	手順を示します。
維続するためには「次へ」をクリックしてください。	
警告:このコンピュータ プログラムは、日本国著作権法および国際条約に ます。このプログラムの全部または一部を無断で複製したり、無断で複製 作権の侵害となりますのでご注意ください。	より保護されてい 物を頒布すると著
<u>キャンセル(Q)</u> 戻る(P)	<u>冼へ他</u>

もし表示されない場合は、CD-ROMのルートディレクトリにあるSETUP. EXEをエクスプローラ等からダブルクリックして起動が可能です。

お使いのシステムに「WindowsInstaller」がインストールされていない場合、上記画面と違う場合があります。 この場合は画面の指示に従ってください。

「次へ(N)」を押すと次の画面に進みます。

😸 ScTool		
使用許諾契約		WACOGIKEN
使用許諾契約をお読みください。以 してから「次へ」 をクリックしてくださ い。 -	下の条件に同意される場合は「同意 さい。その他の場合は「キャンセル」	記します」 をクリック をクリックしてくださ
使用許諾契約書 重要一以下のライセンス契約書 (以下「本契約書」といいます)に (以下「本ソフトウェア製品」とい ずれであるかを問いません)とい いいます)との間に締結される注 ビュータンフトウェアを含み、「オ 体、ならびに印刷物(マニュアル ア製品またはアップデート(以下	を注意してお読みください。本使用 は、上記に示されたワコー技研ッフ います)に関してお客様(個人また 400GIKEN Corporation(以下「ワ 50な契約書です。本ッフトウェア シライン」または電子文書、それに などの文書)を含むこともあります に定義)をインストール、複製、ま	■ 井許諾契約書 가ウェア製品 は法人のい コー技研」と 製品は、コン に関連した媒 で。本ソフトウェ たは使用する、▼
● 同意しません(D)	○ 同意します(<u>A</u>)	
	キャンセル(C) 戻る(P)	次へ(型)

使用許諾契約画面が表示されます。良くお読みになって、同意される場合は「同意します(A)」を クリックしてください。同意されない場合には、インストールを継続することは出来ませんので、 「キャンセル」を押してインストールを中止してください。

「同意します(A)」をクリックすると「次へ(N)」のボタンが押せるようになりますので、押して次の 画面に移動してください。 2. インストール (続き)

🛃 ScTool		
インストール フォルダの選択		WACOGIĶEN
インストーラは以下のフォルダへ ScTool をインスト このフォルダヘインストールためには「次へ」をクリ ールするためには、以下に入力するかまたは「参照	ールします。 ックしてください。(3」をクリックしてくが	世のフォルダヘインスト ださい。
フォルダ(E): C:¥Program Files¥ScTool¥		参照(<u>B</u>)
ソフトウェアを以下のドライブにインストールできます 	+ \ ∕):	
ボリューム	ディスク容量	空き容量
	15GB	8559MB
D:	2047MB	1680MB
E:	1482MB	851 MB
	デ	イスク所要量(D)
キャンセル(空) <u>戻る(P</u>)	

インストール先フォルダ選択画面が表示されます。お使いのシステムに搭載されているハードディスクの容量 及び空き容量が表示されます。

(ハードディスクの表示容量はシステムに依存する為、上記画面と異なります)

特に問題が無い限り、ディレクトリの変更は行わない事を推奨します。

「次へ(N)」を押すと次の画面に進みます。

🐻 ScTool	
インストールの確認	WACOCIVEN
	WALUGINEN
ScTool をコンピュータにインストールする準備ができました。	
インストールを開始するためには「次へ」をクリックしてください。	
キャンセル(<u>C</u>) 戻る(E) <u>xan</u>

インストールの準備が出来ました。

「次へ(N)」を押すとインストールを開始します。

🙀 ScTool	
ScTool 情報	WACOGIKEN
本OS(オペレーティングシステム)が Windows98/98SEの場合、再起動が必要です。 Windows2000/XPの場合、ログオフが必要です。	
継続するためには「次へ」をクリックしてください。	
キャンセル©	戻る(P) (次へ(N))

インストール途中で、注意点が表示されます。

初回のインストール時に限り上記注意書きに従う必要があります。

(動作しているOSがWindows98系の場合は再起動、Windows2000系の場合はログオフが必要になります)

「次へ(N)」を押すと次の画面を表示します。

WACOCIKEN
MACOUNTIN
戻る(P) 閉じる(O)

上図画面が表示されましたらインストールは終了しました。「閉じる(C)」を押してインストーラを終了させて ください。

終了後、初回のインストールに限り、前ページの注意書きに従って下さい。 (2回目以降は必要ありません)

3. アンインストール

本ScToolが不要となった場合に、アンインストールする方法です。

SC2000シリーズサポートツールのCD-ROMをCD-ROMドライブに挿入します。 CD-ROMの自動起動機能が有効であれば、数秒で下記画面が表示されます。

表示されない場合は、CD-ROMのルートディレクトリにあるSETUP. EXEを エクスプローラ等からダブルクリックして起動して下さい。



「ScTool-Japaneseの削除(M)」にチェックをつけて、「完了(F)」を押せばアンインストールが開始されます。

コントロールパネルの「アプリケーションの追加と削除」から「ScTool-Japanese」を選択しても同じ事が行えます。

- 4. ハードウェア
- ・最小構成での接続図



最小構成の接続に必要なハードウェアは、下記の通りです。

- ・SC2000本体
- ・SC2K-RSMTケーブル
- ・DC24Vの安定化電源

PCにシリアルポートが無い場合、USB-SERIAL変換アダプタ(*1)を使用する事により、上記と同等な 接続が可能です。(PC本体にUSBポートが必要となります)

(*1)USBポートを1つ使い、シリアルポートとして使用出来る変換アダプタです。 ATEN社(*2)等から発売されています。

(*2)http://www.aten.com/

5. メニュー

本ツールを起動すると、下記のメニューが表示されます。 各メニューの詳細は、ツールのオンラインヘルプ機能を参照願います。 本説明書では概略のみ説明致します。

・ファイルメニュー

🗐 ScTool								
ファイル(E) 表示	〒(⊻) 特殊制御(<u>C</u>)	うィントウ ₩)	^ルフ°(<u>H</u>)					
新規作成(<u>N</u>)	Ctrl+	N 🕨						
ワークス^°ースの ワークス^°ースの	作成₩) 変更/設定(<u>C</u>)							
フリンタの設定	(<u>R</u>)							
アフリケーションの)終了⊗)							

このモードでは、下記の操作が行えます。 ・レジスタ・プログラム等のウィンドウを開いて編集(新規作成)

> SC2000の内部レジスタ(又はプログラム)を編集する為に、目的のウィンドウを 開く事が出来ます。

・ワークスペース(作業ディレクトリ)の作成/変更/設定

バックアップ・リストア等に使用されるディレクトリを指定します。 ファイルの保存/読込にも使用されます。

・プリンタの設定/印刷(レジスタ・プログラム類のウィンドウを開いている場合)

レジスタ/プログラムを印刷する為のメニューです。

・ソフトの終了

本ソフトウェアを終了します。

5. メニュー (続き)

・表示メニュー

ScTool							J	- 🗆 🗵
771H(E)	表示⊙	特殊制御(〇)	ታለንՒንን₩)	Λ/μフ°(<u>H</u>)				
XB	◆ ツール パ	ΥD						
	▼ ステータス ▼ Info ハ ⁵	-(P)						
	✓ アウトフ [®] ッ	N (B)						
	システム話							

このモードでは、下記の操作が行えます。

・各種ツールバー(又は情報ウィンドウ)の表示/非表示設定

各ツールバー(又は情報ウィンドウ)の表示/非表示を設定します。 閉じてしまったウィンドウは、このメニューから表示させる事が可能です。

・システム設定

本ソフトウェアの動作設定が行えます。

5. メニュー (続き)

・特殊制御メニュー

ScTool		
ファイル(E) 表示(<u>V</u>)	特殊制御(<u>C</u>) ウィント	ウ(W) ヘルフ°(H)
X B C 4	回線接続(Q) 回線切断(Q)	
	回線設定(S)	
	パッグアップ(<u>B</u>) リストア(<u>P</u>)	
	75ッシュROM(<u>F</u>) 🕨	
	特殊機能 ▶	

このモードでは、下記の操作が行えます。

・SC2000との回線接続/回線切断/回線設定

御使いのPCとSC2000の通信を開始/終了します。 レジスタの編集やプログラムの書き込み時には、回線接続が必須となります。 通信手段(RS232C/その他)の設定もこのメニューから行えます。

・SC2000のデータをバックアップ/SC2000ヘデータをリストア

接続されているSC2000から全てのデータ(*1)をバックアップ、又は バックアップされているデータをSC2000にリストアする為のメニュー です。 (*1)ファームウェア以外

・フラッシュROM操作(保存/呼び出し)

SC2000のデータは起動時にRAM(揮発性メモリー)に転送され、これに 対しての編集が行えます。 しかし電源を切ると編集内容は全て破棄されてしまうので、 搭載しているフラッシュROM(不揮発性メモリー)に対して書き込む事により、 レジスタ、プログラムの保存を行えます。 逆に、起動中のSC2000のレジスタ、又はプログラムを電源投入時に戻したい場合は、 呼び出しも行えます。

・特殊機能(消去/再起動/モーション動作状態閲覧)

特定のレジスタ、プログラムの消去が行えます。(消去) SC2000の電源を再投入した状態に戻す事が可能です。(再起動) 現在動作しているモーションプログラムの状況が閲覧出来ます(モーション動作状態閲覧) 5. メニュー (続き)

・ウィンドウメニュー

🗐 ScTool							
ファイル(圧)	表示⊙	特殊制御(<u>C</u>)	<u> ሳሪኑን(₩</u>)	ΔW	7°(<u>H</u>)		
X ₽	C #	?	画面状態		復元 保存	ł	
		- X			1411		

このモードでは、下記の操作が行えます。

・ウィンドウ状態の復元/保存

ウィンドウを毎回大量に開いていたのでは、編集の作業性効率が落ちる場合が有ります。 その様な場合に、「良く使うウィンドウ配列」を記憶しておく事が可能です。 (最大4パターン) 記憶された状態は、呼び出す事により「保存時のウィンドウ状態」に戻ります。 (戻るのはウィンドウ状態のみで、データが戻る訳では有りません)

・ヘルプメニュー

ScTool						_	
ファイル(<u>E</u>)	表示♡	特殊制御(<u>C</u>)	ሳለንՒንን₩)	^/レフ°(<u>H</u>)			
X Ba	6 5	?		^/⊮7°(<u>H</u> ScTool) のバージョン情報(<u>A</u>)		

このモードでは、下記の操作が行えます。

・オンラインヘルプ閲覧

ScToolのヘルプは基本的にオンラインマニュアルのみとなっております。 プログラム作成時に必要な関数リファレンス、メニューの構成、その他情報等から 構成されています。

・本ソフトのバージョン情報表示

本ScToolのバージョン番号が閲覧出来ます。

・SC2000本体ファームウェアのアップデート

SC2000本体のファームウェアをアップデート可能です。 ファームウェア自体は、当社から御送りするファイルか、インターネット経由での取得が 行えます。(インターネットからの取得には、その環境から当社ホームページが閲覧出来る 必要があります)

6. 画面イメージ

・レジスタ編集中

🚺 回線接続中	ー システムレジス	刻								
771N(E)	編集(E) 表示	1	持殊制	御ⓒ	ウむ	小ウ@) NK	7°(<u>H</u>)		_ 뭔 ×
X 🖻 🖬	la									
Name	Data(DEC)	28	24	20	16	12	08	04	00	Index 🔺
SO	2									ビット別タイミング信号
S1	4									現在実装されている軸数(軸ボード>
S2	1835									1秒カウンタ表示
S3	0									全ビットOFFの固定値表示
S4	-1									全ビットONの固定値表示
85	0									エラー表示
S6	0									ラダーで実行したDIV命令(割り算)0
87	0									各レジスタのメモリ書き込み実行 💻
S8	0									
S9	0									
S10	0									各メモリの読み書き時に発生したエ
S11	0									ラダーのレジスタ番号が範囲外であ
S12	0									モーションのレジスタ番号が範囲外
S13	0									
S14	64856									
S15	3582									制御周期(4mS)の内のアイドル時間 🚽
1 C	0									
K Ready. 通信回線の接続に失敗しました 通信回線を接続しました 接続先: Ver-1.10										
レディ							接続	先:Ve	r-1.10	//

・プログラム(モーション)編集中

🗐 ScTool - (1	E-9491 🔄								. 8 ×
□ ファイル(E)	編集(<u>E</u>)	表示⊙	特殊制御(①)	実行(R)	<u>ሳሪኑን(</u>)	^/レフ°(<u>H</u>)		_	- B ×
አ 🖻 🛍	8								
prg O, moni									
;動作モニタ	2								
L10:									
R10.0 = F2	2.0								
R10.1 = F2	4.0								
R10.2 = F2	3.0								
F25 = A0.2									
F26 = A1.2									
F27 = A2.2									
F28 = A3.2									
F28.0 = S1	6.U 0 00								
F28.1 - 31	5.2U 3 30								
F23.2 - 32	2.20 9 10								
F20.0 - 32	2.10								
rate 110									
SOLO LIO									
× ** ScTool t	for SC2xxx	x series	**						
Copyright(C)2000-200	04 WACOGI	KEN Co.,Itd.						
Ready.	ersion : A	2.0.3.7							
Ш									
レディ									
スタート	🥭 🧊	🗹 🛛 😤]window_mon.br	np - ペイン	E ScToo	ol - [₹∽シ	ı/]	∢ €_A	16:48

6. 画面イメージ(続き)

・マルチウィンドウでの編集中画面1

🔍 🚺 汎用レ	バスタ					_		回位置い	ンスタ		
(B) Name	Data(DEC)	Data(HEX)	Index					Name	Data(DEC)	Data(HEX)	Index
R0	0	0	I/Oボードの入	(力信号32	2ビット(書き)	負え不可)	_	P0.0	0	0	位置番号0のデータ(0軸対象
R1	0	0	I /Oボードの出	J力信号32	2ビット			P0.1	0	0	位置番号0のデータ(1軸対象
R2	0	0	I/Oボードの入	(力信号32	2ビット(2枚)	目のボード実装	時)	P0.2	0	0	位置番号0のデータ(2軸対象
R3	0	0	I /Oボードの出	力信号32	2ビット(2枚)	目のボード実装	時)	P0.3	0	0	位置番号0のデータ(3軸対象
R4	0	0	汎用レジスタ					P0.4	0	0	位置番号0のデータ(4軸対象
R5	0	0	汎用レジスタ					P0.5	0	0	位置番号0のデータ(5軸対象
R6	0	0	汎用レジスタ					P0.6	0	0	位置番号0のデータ(6軸対象
R7	0	0	汎用レジスタ					P0.7	0	0	位置番号0のデータ(7軸対
R8	0	0	汎用レジスタ					P0.8	0	0	位置番号0のデータ(8軸対
' R9	U	U	汎用レジスター					P0.9	0	0	位置番号0のデータ(9軸対象
R10	U	U	汎用レジスター					P0.10	0	0	位置番号0のデータ(10軸)
R11	U	U	川用レジスター 初日 日間	- /11 \ }-1				II PO 11	n		位置番号0のデータ(11軸)
R12	0	U	川田レジスタ	ファイルレジスタ	*				_		位置番号0のデータ(12軸)
R13	U	U	「元用レジスタ Na	une	Data(DEC)	Data(HEX)	Inde:	<			位置番号0のデータ(13軸)
			F0		0	0					位置番号0のデータ(14軸)
🛛 システム			F1		0	0					位置番号0のデータ(15軸)
Name	Data(DEC)	Data(HEX)	Index F2		0	0					10直番号0のデータ(16軸)
S0	34	22	ビット別タイ F3		0	0					(17軸)
S1	4	4	現在実装され F4		0	0					
\$2	29667	73e3	1秒カウンタ F5		0	0			•	deres des des des des	
\$3	0	0	全ビットOF F6		0	0			.** ?????	*************************************	いキッ印刷フェット
S4	-1	fffffff	全ビットON F7		0	0	_		*	nnusaan⊥ 21. erfi.42 200	レコン-(499ユニンコ 14/04/16
S5	0	0	エラー表示 F8		0	0			***********	******	*****
S6	0	0	ラダーで実行 F9		0	0					
\$7	0	0	各レジスタ0. F1	U	U	U	_		prg 0 ;[S16.0]	電源投入時	
S8	0	0	F1	1	0	0	_		if K0.31 = 1		
\$9	232	e8	F1	2	U	U			Y0 = 0		
S10	0	0	各メモリの部 F1	3	U	U			eset:		
S11	0	0	ラダーのレシ F1	4	U	U			if Y0 < 100		
S12	0	0	モーションの F1	5	U	U			E(200 + Y0)	= K(400 + Y0)	
S13	250	fa	FI	5	0	U			Y0 = Y0 + 1		
S14	262116	3ffe4	FI	1	0	0			goto eset		
S15	3925	f55	制御周期(4) F1	8	0	U O			endit		
S16	0	0	モーションロー	9	0	U			weit 2000		
\$17	0	0	モーション3 52	U 1	0	U			endif		
S18	0	0	モーションも	9	0	0			if K0.28 = 1		
\$19	0	0	モーション5	ć	U	U			S19.0 = 1		
S20	0	0	モーション1号						stop		
S21	0	0	モーション160	0~1					endif		
\$22	0	0	モーション192	2~2					gosub BatteryC	heck	
\$23	0	0	モーション224	4~2					R64 = K1		
\$24	0	0							R65 = K2		
\$25	0	0							L8.0 = 1		
S26	0	0							S16.1 = 1		
S27	0	0							stop		
S28	0	0	SCLink使用	目時の:					1		
\$29	0	0	通信異常								
series ##											
WACOGIKEN Co	o.,Itd.										

6. 画面イメージ(続き)

・マルチウィンドウでの編集中画面2

	<1
● 30000 ● Natk (DEC) 22 24 20 16 12 06 12 06 100 100 Asis 1/37,41/2/3050 3 4 1/3<	
15%41種俗地立- 非常体体-人力(515:1) 1	
	-
A/D CH1 CH2 CH3 CH4 ボード1 0	
jp-7/6/1,0)	

7.参考付録 Helpファイル内容

次ページ以降は、ソフトウェアに付属のオンラインHelpの内容を印刷した物です。複数ページの同時参照な どにお使い下さい。

概要

本ツールの概要 本ツールは「SC2xxxシリーズ」のサポート用ツールです。 (現時点では、SC2000のみが対象) 本ツールを使用する事により、「SC2xxxシリーズ」上で動作するプログラムの開発等を行う事が 可能になります。 ハードウェア

項目	内容
パソコン種類	PC/AT互換機
CPU	PentiumII-300MHz相当以上の物 (推奨:PentiumIII-700MHz相当以上の物)
メモリー	128MBytes以上 (推奨:OS起動後に空きメモリーが128M以上)
ビデオ	800×600ピクセル以上 (推奨:1024×768ピクセル以上)
CDROM	2倍速以上(インストール時のみ使用)
通信ポート	RS232Cポートを1つ使用 USB-RS232C変換でも動作可
ハードディスク	10MBytesの空き領域

ソフトウェア

項目	内容
オペレ ーテ ィングシステム (OS)	Windows98(非推奨) Windows98SE(非推奨) Windows Me(非推奨) Windows2000Pro(推奨環境) Windows2000Server(動作可) Windows2000AdvancedServer(動作可) WindowsXP home-edition(推奨環境) WindowsXP Professional-edition(推奨環境) 上記OS の何れかが正常稼動している環境 (エミュレータ(<u>※1</u>)上での動作保証は致しかねます) 注意!Windows95では動作しません

(※1) VMware/VirtualPC2004等のPC/ATエミュレータ

VMware is a trademark of VMware, Inc

VirtualPC2004 is a trademark of Microsoft, Inc

オプションモジュール(拡張機能)

型番	名称	内容			
(提供中)	拡張JOG運転DLL	複数軸(最大4軸)同時運転が可能なJOG 運転用拡張モジュール			
(作成中/BETA 版提供中)	モーションウィザード拡張 DLL	モーション作成時、駆動命令を簡単に入力 出来る拡張モジュール			
(要問い合わせ)	イーサネット通信DLL	SC2000とイーサネット経由で通信を行う為の拡張モジュール SC2000用イーサネット通信基板に付属します			
(作成中)	軸レジスタ制御拡張DLL	軸レジスタに対して、サーボ制御、各種簡易 運転を行う為の拡張モジュール			
(作成中)	簡易エミュレータ拡張DLL	SC2000実機が無い環境でのプログラム・レ ジスタ作成用の拡張モジュール(実機と完全 同一ではありません)			

オプションモジュール(拡張機能)のインストール方法

ScTool本体のインストール先ディレクトリ(規定値:C:¥Program files¥ScTool)の下に Plugin ディレクトリがあるので、

そこに拡張機能のDLLをコピーし、ScTool本体を起動しなおせば完了となります。

幾つかの拡張機能には設定項目が有り、ScToolの「表示(V)」→「システム設定」→「DLL」 タブ画面で設定する事が可能です (設定が無い拡張機能もあります)

又、導入する事によりメニューの項目が変化する場合も有りますので、各拡張機能のマニュ アルを参照願います。

起動方法



起動方法2 2. スタートメニューから起動



スタートメニューに登録されている「ScTool」を選択する事でも同様に起動が可能です。

ファイル(F)・新規作成(N)

ScTool		
_ファイル(E) 表示(V) 特殊制御(C) ウィ)	ሎን∰ ∿⊮ን°(<u>H</u>)	
新規作成(<u>N</u>) Ctrl+N ▶	レジスタウィントウ 🕨 🕨	
ワークスペースの作成(₩) ワークスペースの変更/設定(©)	ハッラメータ ワイントッワ モーション ウイントッウ ラダー ウイントッウ	
フツンタの設定(<u>R</u>)	ウォッチ ウィントゥウ	
アフリケーションの終了 😒	アラーム ウィンドウ	
	アカケ入出力	

このメニューには、下記7点の項目があります (下記項目をクリックすると、詳細説明に移動します)

レジスタ ウィンドウ パラメータ ウィンドウ モーション ウィンドウ ラダー ウィンドウ ウォッチ ウィンドウ アラーム ウィンドウ アナログ入出力

ファイル(F)・新規作成(N)・レジスタウィンドウ

 ScTool ファイル(E) 表示(V) 特殊制御(C) ウィン 	[▶] "ל <u>₩</u>) ∿וגז°(<u>H</u>)		
新規作成(N) Ctrl+N ▶ ワークスへ [®] ースの作成(W) ワークスへ [®] ースの変更/設定(C) フリンタの設定(B) アフリケーションの終了 ⊗	レジスタウィントウ ハペラメータウィントや モーション ウィントや ラダー ウィントや ウォッチ ウィントや アラーム ウィントや アナロゲ入出力	R(汎用)レジスタ A(軸)レジスタ S(システム)レジスタ P(位置)レジスタ V(速度)レジスタ O(オーダ)レジスタ K(キープ)レジスタ F(ファイル)レジスタ E(11年)レジスタ E(11年)レジスタ C(パラメータ)レジスタ Q(実数)レジスタ	

このメニューは、指定されたレジスタのウィンドウ(以下レジスタウィンドウ)を開きます。 開く事が出来るレジスタウィンドウは下記の通りです。

R(汎用)レジスタ

汎用レジスタ(入出カレジスタ及び汎用として使用出来るレジスタ)のウィンドウを開き ます

A(軸)レジスタ

軸レジスタ(各軸の制御・状態等のレジスタ)のウィンドウを開きます S(システム)レジスタ システムレジスタ(SC2xxxの制御・状態等のレジスタ)のウィンドウを開きます P(位置)レジスタ

位置レジスタ(各軸で使用する位置情報が格納されたレジスタ)のウィンドウを開きます

(このレジスタはフラッシュROMに保存が可能です)

V(速度)レジスタ

速度レジスタ(各軸で使用する速度情報が格納されたレジスタ)のウィンドウを開きます

(このレジスタはフラッシュROMに保存が可能です)

O(オーダ)レジスタ

オーダレジスタ(各軸で使用する駆動情報が格納されたレジスタ)のウィンドウを開きます

(このレジスタはフラッシュROMに保存が可能です)

K(キープ)レジスタ

キープレジスタ(不揮発性記憶(電源を切っても残ります)領域のレジスタ)のウィンド ウを開きます

F(ファイル)レジスタ

ファイルレジスタ(フラッシュメモリーに値を保持出来るレジスタ)のウィンドウを開きま す

L(リンク)レジスタ

リンクレジスタ(外部機器と通信を行う為のレジスタ)のウィンドウを開きます

E(拡張)レジスタ

拡張レジスタ(タッチパネル等と通信を行う為のレジスタ)のウィンドウを開きます C(パラメータ)レジスタ

パラメータレジスタ(軸単位及び共通の設定項目レジスタ)のウィンドウを開きます (パラメータウィンドウでは設定不可な項目も変更可能です)

Q(実数)レジスタ

実数レジスタ(モーションから利用可能な実数が扱えるレジスタ)を開きます

レジスタウィンドウの例(システムレジスタ)

🗐 回線接続中	- システムレジス	[夕]		_ 8 ×
🗐 771N(E)	編集(E) 表示	(⊻) 特殊制御(<u>ር</u>)	_ 8 ×
X 🖻 🖪	🖨 💡			
Name	Data(DEC)	Data(HEX)	Index	•
S0	2	2	ビット別タイミング信号	
S1	4	4	現在実装されている軸数(軸ボード×4軸)	
S2	137	89	1秒カウンタ表示	
S3	0	0	全ビットOFFの固定値表示	
S4	-1	fffffff	全ビットONの固定値表示	
S5	0	0	エラー表示	
S6	0	0	ラダーで実行したDIV命令(割り算)のあまり	
S7	0	0	各レジスタのメモリ書き込み実行	
S8	0	0		
S9	0	0		
S10	0	0	各メモリの読み書き時に発生したエラー	
S11	0	0	ラダーのレジスタ番号が範囲外であるかを表示	
S12	0	0	モーションのレジスタ番号が範囲外、スタックオーバー表示	
S13	0	0		
S14	64856	fd58		
S15	3582	dfe	制御周期(4mS)の内のアイドル時間表示	_
S16	Π	Π	モーション0~31番の記動状態	
× Ready. 通信回線を指 接続先:Ver-	8続しました 1.10			▲ ▼
לדיץ (גע ד יי		接続先:Ver-1.1	10 カーン	ημ(1,0)

ファイル(<u>F</u>)・新規作成(<u>N</u>)・パラメータウィンドウ

ScTool		
_ファイル(E) 表示(V) 特殊制御(C) ウィ)	ሎን∰ ∿⊮ን℃⊞)	
新規作成(<u>N</u>) Ctrl+N ▶	レジスタウイントウ 🕨	
ワークスペースの作成(₩) ワークスペースの変更/設定(©)		
フツンタの設定(<u>R</u>)	 	
アフリケーションの終了 😢	アラーム ウィントゥ	
	アカゲ入出力	

パラメータウィンドウを開きます

🚺 回線接続中 - 「パラメータレ	/ジスタ]				_ 8 ×
📃 ファイル(E) 編集(E) 表	示── 移動(M))特	殊制御(⊆) ウィントウ(₩) ヘルフ	*(Н)	_ 8 ×
X 🖻 🖬 🎒 💡					
Name	Data		Argument		•
[O車曲] M(系娄)	0				
[0軸]D係数	0				
10軸」原点復帰位置	0				
	0				
[U===]-ソノトリミット [0===1]	U 400000				
「しり戦」 歳入れ 私歴度	466666				
【19単1])迷皮です。	0	- 44	西よ無」 (0)		
[011]原点1度177万法	U	π	原魚無し(0) ↓古白山の反転 7相(2)		
			-方向 川 2 反転 7相(2)		
			+方向 H.S. 非反転 7相(4)		
			-方向 HIS 非反転 7相(5)		
			+方向 HIS 反転(6)		
			-方向 HLS 非反転(7)		
			+方向 Z相(8)		
			-方向 Z相(9)		
			+方向 PLS 反転 Z相(12)		
			-方向 MLS 反転 Z相(13)		
[0軸]入力論理	160	*	PLS/MLS A接(b0:0)		
			PLS/MLS B接(b0:1)		
		*	HLS A接(b1:0)		
			HLS B接(b1:1)		•
★ Ready. 通信回線を接続しました 接続先:Ver-1.10	·				▲ ▼
RESULT WINDOW	接続先:Ver-	-1.10		<u> カーソル</u>	/(1,0)

(パラメータのウィンドウを最大表示で開いています)

各個所の説明

「Name」

軸番号及びパラメータ名称が表示されます

「Data」

パラメータの現在の値が10進数で表示しています この場所に直接数値を入力する事も可能です

[Argument]

選択肢が表示されています。現在選択されている項目は、左側に「*」マークが付きます 項目を選択するには、選択肢をクリックするか、「*」の位置をクリックして下さい

共通パラメータと軸パラメータの切り替え方

(共通パラメータ表示中に「軸別パラメータ」に切り替えるには、下記メニューを選択します)

B) SC 1001							- 0 ×
ファイル(Ĕ) 編集(Ĕ)	表示♡	切り替え(<u>M</u>) 特	寺殊制御(C)	ሳብ≻ኮን₩	∿⊮7°(<u>H</u>)		
X 🖻 🖬 🚭	8	共通パラメータ					
		■ 曲房リハ 。 ラメータ	_				
		軸指定移動					
(軸別パラメータ	表示中に	「共通パラス	「タ」に切	り替えるに	こは、下記メニュ	ーを選択します)
🗐 ScTool							- 8 ×
ファイル(Ĕ) 編集(Ĕ)	表示⊙	切り替え(<u>M</u>) 特	寺殊制御(<u>C</u>)	ታለንՒንን₩)	∿⊮7°(<u>H</u>)		
X 🖻 🖻 🖊	?	共通ハ [®] ラメータ					
		単面がリバーブルーツ					
		軸指定移動					

ファイル(\underline{F})・新規作成(\underline{N})・モーションウィンドウ

ScTool		>
	ሎን∰ ∿⊮ን°(<u>H</u>)	
新規作成(<u>N</u>) Ctrl+N ▶	レジスタウィントウ 🕨 🕨	
ワークスペースの作成(W)	パラメータ ウィントウ	
ワーウスペースの変更/設定(⊆)	モニンヨノ ワイノト ワ ラダー ウィント・ウ	
フツンタの設定(<u>R</u>)	<u> </u>	
アフリケーションの終了 😒	ፖラーム ウィンドウ	
	アカケ入出力	

モーションを開きます

メニューを選択すると、下記ウィンドウが表示されます。

	×					
下記の項目から選択して 「決定」を押して下さい						
▶ 通信回線からプログラムを取得して開く						
□ 空のウィンドウを開く						
キャンセル 決定						

編集対象を選択すると、編集画面が表示されます

🗐 ScTool - [E는%)							_ 8 ×
□ 77イル(E)	編集(E)	表示⊙	特殊制御(C)	実行(<u>R</u>)	ሳለ⊁ን∰	^/↓フ°(<u>H</u>)		_ 8 ×
X 🖻 🛍	8							
prg O, moni								
;動作モニ:	9							
L10:								
R10.0 = F2	2.0							
R10.1 = F2	24.U							
R10.2 = F2	(3.U)							
F20 = A0.2 F28 = A1.2								
F27 = A2.2	- ,							
F28 = A3.2	2							
F29.0 = S1	6.0							
F29.1 = S1	6.20							
F29.2 = S2	2.28							
F29.3 = S2	2.18							
wait 4								
goto L10								
× ** ScTool	for SC2xx	x series	**					
Copyright(C)2000-20	04 WACOGI	KEN Co.,Itd.					
Ready.	ersion:	2.0.3./						
								*
ν 7 °r								
愛スタート	6 🗊	1 🗄	window_mon.br	np - ペイン	N ScTo	ol - [₹~ション]		€ E_A 16:48
(モーション	のウィン	ドウを最	大表示で開	いている	ます。又、	表示されている	るプログラム	はサンプルと

ファイル(F)・新規作成(N)・ラダーウィンドウ

ScTool		
_ファイル(E) 表示(V) 特殊制御(C) ウィ)	/ドウ(₩) ^ルプ(<u>H</u>)	
新規作成(<u>N</u>) Ctrl+N ▶	レジスタウィントウ 🕨 🕨	
ワークスペースの作成(₩) ワークスペースの変更/設定(©)	パラメータ ウィントッウ モーション ウィントッウ ラダニー ウィントッウ	
7%ンタの設定(R)	ウォッチ ウィントウ	
アフツケーションの終了 🕑	ፖラーム ウィントゥ	
	アカゲ入出力	

ラダーを開きます

メニューを選択すると、下記ウィンドウが表示されます。



編集対象を選択すると、編集画面が表示されます

📓 ScTool - [f	が-]							_ 8 ×
□ ファイル(E)	編集(E)	表示⊙	特殊制御(〇)	実行(R)	<u>ሳለኦንሙ</u>	^#7°(<u>H</u>)		_ 8 ×
X 🖻 🛍	3 ?							
NOP								
LD S0.1								
SETr S16.0								
NOP								
LD R10.1								
ANDN S16.2	0							
ANDN S22.2	8							
SEIr SZZ.1	8							
E01 24 A0	q							
ANDS	•							
SETr S22.1	9							
NOP								
NOP								
LD R10.2								
× ** ScTool t	for SC2xx	x series	** VEN Colled					A.
Software v	ersion :	2.0.9.7	NEM 60.,ICU.					
Ready.								
								$\overline{}$
レディ						カーソル(0,0)		
(ラダーのウ	ヮィンドウ	を最大著	表示で開いて	ています	。又、表示	されている	プログラムに	はサンプルとして
入力した物	です)							

ファイル(F)・新規作成(N)・ウォッチウィンドウ

ScTool		
_ ファイル(E) 表示(V) 特殊制御(C) ウィ)	ሎን∰ ∿⊮ን°⊞)	
新規作成(N) Ctrl+N ▶	レジスタウィントウ 🕨	
ワークスペースの作成(\ <u>\</u> /) ワークスペースの変更/設定(<u>C</u>)	ハッラメータ ワイントッワ モーション ウイントッウ ラダー ウイントッウ	
フツンタの設定(<u>R</u>)		
アフリケーションの終了 😢	アラーム ウィンドウ	
	アカロゲ入出力	

ウォッチウィンドウを開きます

🗐 ScTool -	[ウォッチ]			_ 8 ×
□ ファイル(E)	编集(E)表示	示(⊻) 特殊制御	♥(<u>C</u>) ウィントウ(₩) ヘルフ°(<u>H</u>)	_ 8 ×
X 🖻 🖡	1 🖨 🤶			
Name	Data(DEC)	Data(HEX)	Index	
S2	8456	2108	1秒カウンタ表示	
*	?	?		
A0.0	33	21	0軸の状態表示	
*	?	?		
R0	0	0	I/Oボードの入力信号32ビット(書き換え不可)	
R2	0	0	I/Oボードの入力信号32ビット(2枚目のボード実装時)	
*	?	?		
*	?	?		
*	?	?		
*	?	?		
*	?	?		
*	?	?		
*	?	?		
*	?	?		
*	?	?		-
•				
× ** ScTool Copyright Software Ready.	for SC2xxx se (C)2000-2004 W version : 2.0.	ries ** ACOGIKEN Co., 9.7	ltd.	*
レディ			カーソル(0,0)	
スタート) 🥭 🤤 🗹	Window_la	ad_text.bmp - 🖑 🗐 ScTool - [9477]	<u>A</u> 16:36

(ウォッチウィンドウを最大表示で開いています。又、入力済みの個所はサンプルとして入力した物です)

各個所の説明

「Name」

監視するレジスタ名が表示されます(入力欄)

「Data(DEC)」

監視するレジスタの現在の値を10進数で表示しています

「Data(HEX)」

監視するレジスタの現在の値を16進数で表示しています

「Index」

監視するレジスタに対する説明を表示しています

キー入力関係の説明

キー操作	举動
リターンキー	入力したデータをカーソル位置に書き込み
シフトキーを押しながらリターンキー	入力したデータをカーソル位置に挿入
枠内を消してリターンキー	カーソル位置を消去し、それより下にある項 目を上に移動する

ファイル(<u>F</u>)・新規作成(<u>N</u>)・アラームウィンドウ

ScTool		
_ファイル(E) 表示(V) 特殊制御(C) ウィ)	ሎን∰ ∿⊮ን°(<u>H</u>)	
新規作成(<u>N</u>) Ctrl+N ▶	レジスタウィントウ 🕨 🕨	
ワークスペペースの作成(\\) ワークスペペースの変更/設定(<u>C</u>)	パラメータ ウィンドウ モーション ウィンドウ ラダー ウィンドウ	
フツンタの設定(R)	ウォッチ ウィントゥ	
アフリケーションの終了 😢	፻∋−ፊ ዕለን⊦ን	
	アカケ入出力	

アラームウィンドウを開きます

アラームが発生していない状態でのアラームウィンドウ

■回線接続中 - [アラーム状態]	回線接続中 - [アラーム状態]					
 ファイル(E) 編集(E) 表示 	(型) 特殊制御(C)	ウィントゥ(W) ヘルフヾ(<u>H</u>)	_ & ×			
X 🖻 🖻 🚭 🦿						
Name D)ata Alai	m				
x						
Ready. 通信回線を接続しました						
接続先:Ver-1.10						
			•			
RESULT WINDOW	接続先:Ver-1.10					
(アラームのウィンドウを	最大表示で開	いています)				

アラームが発生している状態でのアラームウィンドウ

🗐 回線接続中 - 「アラーム状態	態]		6	X
📓 ファイル(E) 編集(E) 表	示(⊻) 特殊制	≇©) ሳለ⊁ኮነ₩) ∿ルフ°(<u>H</u>)	_ 5	×
X 🖻 🖬 🎒 🤶				
Name	Data	Alarm		
[共通]アラーム発生		アラーム発生(b15:1) システムエラー発生(b14:1)		
[システム]ウジー固有		ラダーレジスク番号範囲外(b0:1)		
× Ready. 通信回線を接続しました 接続先:Ver-1.10				•
レディ (アラー / のウィンドウ:	接続先:Ver- た是ナキー	1.10	/ ////2.12)
	で取八衣小			

各個所の説明

[Name]

アラーム分類が表示されます

「Alarm」

発生しているアラームの詳細を表示しています

ファイル(<u>F</u>)・開く(<u>O</u>)

🗐 ScTool					
ファイル(E) 編集(E) 表示(V)	特殊制御(C)	ሳብንኑንን∰	^ルフ°(<u>H</u>)		
新規作成(N)	Ctrl+N ▶				
開((<u>O</u>)	Ctrl+O	_	_	 	
閉じる(©) 上書き保存(<u>S</u>) 名前を付けて保存(<u>A</u>)	Ctrl+S				
ED刷(<u>P</u>) ED刷プレビュー(V) プソンタの設定(R)	Ctrl+P				
ワーりスペースの作成(₩) ワーりスペースの変更/設定(©)					
アフツケーションの終了 🖄					

開いているウィンドウのモードによって、読み込めるファイルに相違があります。 (詳細は下記)

ウィンドウのモ ー ド	読み込めるファイル
レジスタ(全レジスタ対応)	各モ ー ドのレジスタファイル テーブルファイル
モーション	モーションファイル
ラダー	ラダーファイル
パラメータ	パラメータファイル
ウォッチ	ウォッチファイル

レジスタファイル読み込み時の詳細

ウィンドウのモードがレジスタであれば、レジスタの種類には関係なく「レジスタファイル」を 読み込む事が可能です。 読み込んだファイルは、即座に接続中のSC2xxxに転送され、書き込まれます。 但し、フラッシュROMに保存するには、別途フラッシュROMに対する操作が必要となるの で注意が必要です。 (Kレジスタを除く)

モーション/ラダーファイル読み込み時の詳細

モーションのファイルを読み込んだ場合、編集領域にはファイルの内容が表示されますが、 SC2xxxには転送されていません。 「実行」メニューから操作して、初めて転送されます。 レジスタと同様にフラッシュROMへの操作が必要です。

ウォッチファイル読み込み時の詳細

読み込んだ時点から即監視が始まります。

テーブルファイル読み込み時の詳細

テーブルファイル(旧ツール類で作成したプロジェクト、又はバックアップで作成されたプロジェクト形式のデータに含まれるファイル)は、 読み込まれた時点で、「定義内容が空白でない場合は、現在の定義に上書き」されます。

ファイル(<u>F</u>)・閉じる(<u>C</u>)

🗐 ScTool					 IJŇ
ファイル(E) 編集(E) 表示(V)	特殊制御(2)	ታለንՒንን₩)	^ルフ°(<u>H</u>)		
新規作成(N) 開((<u>O</u>)	Ctrl+N + Ctrl+O			 	
<u>閉じる(C)</u> 上書き保存(S) 名前を付けて保存(<u>A</u>)…	Ctrl+S				
印刷(P) 印刷プレビュー(V) プリンタの設定(B)	Ctrl+P				
ワークスペースの作成(\\) ワークスペースの変更/設定(_)					
アフリケーションの終了 🗵					

ウィンドウを閉じます。

ウィンドウを閉じる事により、編集内容が破棄されてしまう場合には、編集内容を反映させる為の 選択肢が表示される場合もあります。

ファイル(<u>F</u>)・上書き保存(<u>S</u>)

ScTool							
ファイル(E) 編集(E) 表示(V)) 特殊制御(<u>C</u>)	ሳ⁄ንՒኻ₩)	^ルフ°(<u>H</u>)				
新規作成(N) 開((Q) 閉じる(<u>C</u>)	Ctrl+N Ctrl+O	-	-	-	-	-	
上書き保存(S) 名前を付けて保存(<u>A</u>)	Ctrl+S						
印刷(<u>P)</u> 印刷プレビュー(\/) プリンタの設定(<u>P</u>)	Ctrl+P						
ワークスペースの作成(₩) ワークスペースの変更/設定(<u>C</u>)							
アフツケーションの終了⊗							

編集中のデータをファイルに保存します 保存されるファイルは、開いているウィンドウのモードによって変わります(下記)

ウィンドウのモード	作成されるファイル
レジスタ(全レジスタ対応)	各モードに対応したレジスタファイル 作成されたファイルには、「定義」も一緒に格納され る。 (テキストファイル)
モーション	モーションプログラム(テキストファイル)
ラダー	ラダープログラム(テキストファイル)
ウォッチ	ウォッチファイル(テキストファイル)

ファイル(F)・名前を付けて保存(A)

ScTool							- D X
ファイル(E) 編集(E) 表示(V)	特殊制御(①)	<u>ሳ</u> ብ≻ን∰	^/レプ(<u>H</u>)				
新規作成(<u>N</u>) 開((<u>O</u>) 閉じる(<u>C</u>)	Ctrl+N ► Ctrl+O	_	-	_	-	_	
上書き保存(S)	Otrl+S						
名前を付けて保存(<u>A</u>)							
印刷(<u>P)</u> 印刷プレビュー(Ѵ) プツンタの設定(<u>B</u>)	Ctrl+P						
ワークスペースの作成(₩) ワークスペースの変更/設定(<u>C</u>)							
アフリケーションの終了 😢							

編集中のデータを任意のファイル名を指定して保存します 詳細は、「<u>上書き保存(S)</u>」を参照願います。

ファイル(<u>F</u>)・印刷(<u>P</u>)

🗐 ScTool						_ 🗆 ×
ファイル(E) 編集(E) 表示(V)	特殊制御(0)	ウィンドウ(₩)	^/レプ(<u>H</u>)			
新規作成(N) 開く(Q) 閉じる(<u>C</u>) 上書き保存(<u>S</u>) 名前を付けて保存(<u>A</u>)	Ctrl+N ► Ctrl+O Ctrl+S					
EP刷(<u>P)</u> EP刷プレビュー(<u>V</u>) プソンタの設定(<u>R</u>)	Ctrl+P					
ワークスペースの作成(\\) ワークスペースの変更/設定(_)						
アフリケーションの終了 🗵						

編集中のデータ又は接続対象のSC2xxxのデータを印刷します。 印刷される対象は、ウィンドウのモードによって変わります。(下記)

ウィンドウのモード	指定1	指定2	印刷詳細
レジスタ (全レジスタ)	開始レジスタ名	終了レジスタ名	接続対象のレジスタ
モーション	開始行番号	終了行番号	編集中のモーション
ラダー	開始行番号	終了行番号	編集中のラダー
ウォッチ	無し	無し	監視中のレジスタ名及び データ

ファイル(<u>F</u>)・印刷プレビュー(<u>V</u>)

🗐 ScTool						
ファイル(<u>F</u>) 編集(E) 表示(⊻)	特殊制御(C)	ታィントንን₩)	^/μフ°(<u>H</u>)			
新規作成(N) 開く(Q) 閉じる(Q)	Ctrl+N ► Ctrl+O					
上書き保存(S) 名前を付けて保存(A) 	Ctrl+S					
ED刷(<u>P</u>)	Ctrl+P					
印刷プレビュー(Ѵ)						
7判2%の設定(<u>R</u>)						
ワークスペースの作成(\\) ワークスペースの変更/設定(<u>C</u>)						
アフツケーションの終了 🗵						

印刷結果を画面上で確認出来ます。 詳細は「<u>印刷(P)</u>」を参照願います。

ファイル(<u>F</u>)・プリンタの設定(<u>R</u>)

ScTool			×
ファイル(E) 表示(⊻)	特殊制御(<u>C</u>) ウィン	ት"ኃ(₩) ∿ルフ°(<u>H</u>)	
新規作成(N)	Ctrl+N ►		
ワークスペースの作成(<u>M</u> ワークスペースの変更/言	୬ 設定(<u>C</u>)		
7判ンタの設定(<u>R</u>)			
アフリケーションの終了心	0		

印刷設定ウィンドウを開きます(下記はWindows2000での表示)

プリンタの設定					? X
「プリンターーーー					
プリンタ名(N):	RICOH IPSIO NX810			プロパティ(<u>P</u>)	
状態:	準備完了				
種類	RICOH IPSiO NX810				
場所:	IP_192.168.2.163				
אלאב	RPDL Driver (Windows 2000/XP)				
			一印刷の向	うき	
サイズ(Z):	A4 297 × 210 mm	•		◉ 縦(())	
給紙方法(<u>S</u>):	自動化化選択	•	Α	○ 横(<u>A</u>)	
ネットワーク(<u>W</u>)]		OK		il I

ファイル(<u>F</u>)・ワークスペースの作成(<u>W</u>)

ScTool		- D X
ファイル(<u>F</u>) 表示(⊻) 特殊制御() ウィント [™] ウ(W) ヘルフ [®] (H)	
新規作成(<u>N</u>) C	I+N •	
ワークスペースの変更/設定(<u>C</u>)		
フツンタの設定(<u>R</u>)		
アフツケーションの終了 🖄		

ワークスペース(作業ディレクトリ)の作成を行います。 起動すると下記ウィンドウが表示されます

		HO /	 			
<u> </u>	ースの作用	戎				×
7-	יג−°~גל	名				
ļ					_	
	/ + m					
	位置					
					<u></u>	照
			キャンセル		作成	
F ()	18.7	1 1-1-1	11.54	 		

「参照」ボタンを押して、基準となるディレクトリを選択する事が可能です。 ディレクトリは、「位置」+「ワークスペース名」で作られます。

ファイル(<u>F</u>)・ワークスペースの変更/設定(<u>C</u>)
🗐 ScTool											_	미지
771N(E)	表示⊙	特殊制御(<u>C</u>)	ウルド	უ₩	^⊮7°(<u>H</u>)							
新規作成	\$(<u>N</u>)	Ctrl+I	N ▶									
	-スの作成(<u>w</u>)										
- ም-ትአላጭ	-スの変更/	設定(<u>C</u>)										
フツンタの言	资定(<u>R</u>)											
アフツケーシ	いの終了の	Σ										

ワークスペース(作業ディレクトリ)の変更(又は指定)を行います。 起動すると下記ウィンドウが表示されます

ワークスペースの設定		×
位置	_	
		参照
	,	
	キャンセル	設定

「参照」ボタンを押して、ディレクトリを選択する事が可能です。

ファイル(<u>F</u>)・アプリケーションの終了(<u>X</u>)

ScTool	
ファイル(E) 表示(V) 特殊制御(C) ウル	ት"ኃ(<u>₩</u>)\₽ጋ°(<u>H</u>)
新規作成(N) Ctrl+N ▶	
ワーウスヘ⁰ースの作成(₩) ワーウスヘ⁰ースの変更/設定(©)	
7判ンタの設定(R)	
アフリケーションの終了 🕗	

本ソフトを終了します

保存されていないデータが存在する場合には、該当するウィンドウが警告を表示します。 (警告された時点で保存する事も可能です)

編集(<u>E</u>)メニュー

レジスタ類ウィンドウ選択時

🗐 ScTool				
7711U(E)	編集(<u>E</u>) 表示(⊻) 特殊	制御(<u>C</u>) ウル	ՒᲣ๎๎๎๎๎∭` ∿⊮プ(⊞)	
X 🖻	元(J戻す(U) Otrl+Z			
	切り取り(<u>T</u>) Otrl+X			
	北~(©) Ctrl+C 貼り付(ナ(P) Ctrl+V			
÷… 📄 Wo	検索(S) 置換(E)			

プログラム類ウィンドウ選択時

🗐 ScTool							
7711U(E)	編集(<u>E</u>) 表示(<u>∨</u>) 特殊制御(<u>(C</u>) 実行(<u>R</u>)	ሳብ≻ንት∰	∧ルフ°(<u>H</u>)		
X 🖻	元に戻す(凹)	Ctrl+Z					
	切り取り(I) コピー(C) 貼り付(ナ(P) すべて選択(<u>A</u>)	Ctrl+X Ctrl+C Ctrl+V Ctrl+A					
	検索(<u>S</u>) 置換(E)	Ctrl+F Ctrl+H					
	文書整形(R)						

このメニューには、下記8点の項目があります



編集(<u>E</u>)・元に戻す

Sc Tool								_ 🗆 ×
771N(E)	編集(E) 表示(⊻) 特殊制	御(<u>C</u>)	実行(R)	ウィントንን₩)	∿ルフ°(<u>H</u>)		
X 🖻	元に戻す(U)	Otrl+Z						
	切り取り(T) コピー(C) 貼り付け(P) すべて選択(<u>A</u>)	Otrl+X Otrl+G Otrl+V Otrl+A						
	検索(<u>S</u>) 置換(<u>E</u>)	Ctrl+F Ctrl+H						
	文書整形(<u>R</u>)							

直前に行った作業をキャンセルします。 現在殆どの機能で動作しません。御迷惑をお掛けします

編集(E)・切り取り(T)

ScTool							
771N(E)	編集(E) 表示(⊻) 特殊制	御(<u>C</u>)	実行(R)	ታለንՒንን₩)	∿ルフ°(<u>H</u>)	
🔏 🖻	元に戻す(U)	Ctrl+Z					
	切り取り(工)	Ctrl+X					
🗉 🕀 💼 SC	⊐Ľ°−©)	Ctrl+C					
🗄 🗄 💼 Infe	貼り付け(<u>P</u>)	Ctrl+V					
÷ 🧰 Wo	すべて選択(<u>A</u>)	Ctrl+A					
	検索(<u>S</u>)	Ctrl+F					
		Ctrl+H					
	文書整形(<u>R</u>)						

指定された領域を切り取ります

編集(<u>E</u>)・コピー(<u>C</u>)

ScTool						_ 🗆 ×
ファイル(E)	編集(<u>E</u>) 表示(⊻) 特殊制御	№) 実行(R)	ሳሪኑን 😡	^//フ°(<u>H</u>)	
🐰 🗈	元に戻す(世)	Ctrl+Z				
	切り取り(工)	Ctrl+X				
🕀 🖳 🔁 🔁	⊐Ľ°−©)	Ctrl+C				
🗄 🕀 🛄 Infe	貼り付け(P)	Ctrl+V				
Ė~ 🧰 ₩o	すべて選択(<u>A</u>)	Ctrl+A				
	検索(S)	Ctrl+F				
	置換(<u>E</u>)	Ctrl+H				
	文書整形(R)					

指定された領域をクリップボードにコピーします

編集(<u>E</u>)•検索

ScTool								_ 🗆 X
7711UE)	編集(E) 表示(⊻) 特殊制	御(<u>C</u>)	実行(R)	ታለንՒንን₩)	∿μフ°(<u>H</u>)		
X 🗈	元に戻す(世)	Ctrl+Z						
	切り取り(<u>T</u>)	Ctrl+X						
🕀 🕀 🔁 😳	⊐Ľ°−(C)	Ctrl+C						
🗄 💼 Infe	貼り付け(<u>P</u>)	Ctrl+V						
Ė∾🪞 Wo	すべて選択(<u>A</u>)	Ctrl+A						
	検索(S)	Otrl+F						
	置換(<u>E</u>)	Ctrl+H						
	文書整形(R)							

文字列を検索します。

選択すると、下記ウィンドウが表示されます。

検索		? ×
検索する文字列(<u>N</u>):		次を検索(E)
□ 単語単位で探す₩) □ 大文字と小文字を区別する(<u>C</u>)	検索する方向 ○ 上へ(山) ④ 下へ(D)	キャンセル

「検索する文字列(N)」

検索を行う文字列を入力します

「単語単位で探す(<u>W</u>)」

現在使用出来ません。御迷惑をお掛けします

「大文字と小文字を区別する(C)」

検索対象と比較する時に、入力された文字を厳密に比較します

「検索する方向」

現在の編集位置から、上方向、又は下方向のどちらに向かって検索を行うか設定します

検索対象となる個所

検索対象となる個所は、開いているウィンドウのモードによって変わります(下記)

ウィンドウのモ ー ド	検索対象
レジスタ(全レジスタ対応)	レジスタ名
モーション	モーションプログラム
ラダー	ラダープログラム
パラメータ	パラメータ名
ウォッチ	レジスタ名

編集(E)・貼り付け

ScTool						_ 🗆 🗵
7711UE)	編集(<u>E</u>) 表示(⊻) 特殊制御	即(<u>C</u>) 実行(R)	ሳለንՒንን₩)	∧μフ°(<u>H</u>)	
X 咱	元に戻す(世)	Ctrl+Z				
terrent for the second	切り取り(T) コピー(C) 貼り付(ナ(P) すべて選択(<u>A</u>)	Otrl+X Otrl+O Otrl+V Otrl+A				
	検索(<u>S</u>) 置換(<u>E</u>)	Ctrl+F Ctrl+H				
	文書整形(<u>R</u>)					

クリップボードに積まれたデータを貼り付けます

編集(<u>E</u>)•置換

ScTool							
7711U(E)	編集(<u>E</u>) 表示(⊻) 特殊制	御(<u>C</u>)	実行(R)	ウィンドウ๎๎₩	^/レフ°(<u>H</u>)	
🐰 🗈	元に戻す(世)	Ctrl+Z					
	切り取り(工)	Ctrl+X					
🖳 🕀 💼 SC	⊐Ľ°(<u>C</u>)	Ctrl+C					
🗄 🗄 💼 Infe	貼り付け(P)	Ctrl+V					
±…🪞 ₩o	すべて選択(<u>A</u>)	Ctrl+A					
	検索(S)	Ctrl+F					
	置換(<u>E</u>)	Ctrl+H					
	文書整形(R)						

文字列を置換します。

現在動作しません。御迷惑をお掛けします

編集(<u>E</u>)·文書整形

ScTool								
7711U(E)	編集(E) 表示(⊻) 特殊制	御(<u>C</u>)	実行(R)	ሳብ≻ኮን₩)	∿μフ°(<u>H</u>)		
🐰 🖻	元に戻す(世)	Ctrl+Z						
ternet for the second	切り取り(T) エピー(C) 貼り付(け(P) すべて選択(<u>A</u>)	Ctrl+X Ctrl+C Ctrl+V Ctrl+A						
	検索(<u>S</u>) 置換(<u>E</u>)	Ctrl+F Ctrl+H						
	文書整形(R)							

編集中プログラムを整形します。 モーションの場合、下記法則で整形されます

命令	法則
prg / sub	この行は左端から始まり、次の行から繰り下がる
ラベル	この行は左端から始まり、次の行は前の繰り下がり位置に戻 る
if	次の行から繰り下がる
elseif / else	この行は繰り上がるが、次の行は元の繰り下がり位置に戻る
endif	この行から繰り上がる
その他の命令	現在の位置を継続

ラダーの場合

下記法則で整形されます

命令	法則

NOP	この行は左端から始まる
その他の命令	左端から1つ繰り下がった位置

整形+	ナンプル
モ — シ	ンヨン
	$\begin{array}{l} \text{prg 1} \\ \text{if } r0.0 = 0 \\ \text{if } r0.2 = 0 \\ \text{f100 = 1} \\ \text{endif} \\ \text{f101=1} \\ \text{endif} \\ \text{f102=1} \\ \text{elseif } r0.3 = 0 \\ \text{f103=0} \\ \text{endif} \\ \text{stop} \end{array}$
ラダー	_
	LD R0.0 SETr S16.0 LDN R0.0 RESr A0.24.0 NOP LD R0.1 SETr S16.1 LD R0.2 SETr S16.2 LD S0.6 SETr S16.3 NOP

表示(V)メニュー

LD RO.3 SETr S16.4

レジスタ/プログラム類ウィンドウ未選択時



レジスタ/プログラム類ウィンドウ選択時

🗐 ScTool		
ファイル(Ё) 編集(Ё)	表示(⊻) 特殊制御(<u>C</u>) ウイ	ンドウ (W) ヘルフ*(H)
SC2000 Information Workspace	 ・ ツール パー(T) ・ ステータス パー(S) ・ Info パー(P) ・ アウトフット(B) 福準形式 拡張形式 ・ 横ライン 画面高速更新 Alt+F5 システム設定 	

このメニューには、最大下記9点の項目があります (項目名をクリックすると、詳細説明に移動します)

<u>ツール バー</u> ステータスバー Info バー(情報バー) アウトプット 標準形式 拡張形式 横ライン 画面高速更新 システム設定

表示(⊻)・ツールバー

🚺 ScTool					
7711UE)	表示(⊻) 特殊制御(C)	ታለንՒንን₩)	^/μフ°(<u>H</u>)		
X Ba	✓ ツール パー(T)				
	 ✓ ステータス バー(S) ▲ したし いき (D) 				
🕂 👘 🔁 SC	✓ Into //~(P) ✓ P0kp‰k (R)				
🗄 🚞 Info					
	システム設定				

ツールバーの表示/非表示を制御します 2度選択すると元に戻ります。

表示(⊻)・ステータスバー

ScTool					- D ×
7711(E)	表示(⊻)	特殊制御(〇)	ታለንՒንን₩)	^/μフ°(<u>H</u>)	
X Ba	◆ ツール バ	~D			
	🖌 रहेम्फ्रेर	<u></u>		_	
🖃 📄 SC	 Info ກ⁸ 	-(<u>P</u>)			
🕂 🛄 Infi	✓ アウトフ [®] ッ	K®			
⊕ · <mark>⊡</mark> Wo	システム語	定			

ステータスバーの表示/非表示を制御します 2度選択すると元に戻ります。

表示(<u>V</u>)・Infoバー(<u>P</u>)

ScTool								-OX
7711(E)	表示(⊻)	特殊制御(<u>C</u>)	ሳብንኑንን∰	^/レプ(<u>H</u>)				
X 🖻	✓ ツール ハ → フニーカフ							
	 ✓ Info N^S 	-(P)						
	✓ アウトフ [®]	ŀ(₽)						
÷ 🧰 Wo	システム語	定						

Infoバー(情報バー)の表示/非表示を制御します 2度選択すると元に戻ります。

表示(⊻)・アウトプット

🗐 ScTool						- U ×
ファイル(圧)	表示(⊻) 特殊	៖制御⊙)	ታィンՒንን₩)	^ルフ°(<u>H</u>)		
X Ba	✓ ツール パー(①)					
<u> </u>	→ ステータス バー(S) 🗕	_		 _	
m 🖨 se	✓ Info ハ ^v →(P)					
- 🕂 🛄 30	→ アウトプット (B)					
	システム設定					

アウトプット領域(リザルトウィンドウ)の表示/非表示を制御します 2度選択すると元に戻ります。

表示(⊻)・標準形式

🗐 ScTool		
ファイル(Ё) 編集(Ё)	表示(v) 特殊制御(C) ウィンドウ(W) ヘルフ°(H)	
X Image: Barrier Barrier Image: Barrier Barrier Image: Barrier Barrier Image: Barrier Barrier Image: Barrier Image: Barrier	 ツール バー(T) ステータス バー(S) Info バー(P) アウトフット(P) アウトフット(P) 構準形式 拡張形式 横ライン 	
	画面高速更新 Alt+F5	
	<u>システム設定</u>	

レジスタ表示時に、表示内容を標準形式で表示します。 (データの一部を16進数で表示します)

拡張表示時にしか選択出来ません

(表示例)レジスタ・ウィンドウ(システムレジスタ)

🗐 回線接続中	- システムレジス	[タ]		_ 8 ×
🗐 771N(E)	編集(E) 表示	(⊻) 特殊制御(©) ウィンドウ(W) ヘルフ°(H)	_ 8 ×
X 🖻 🖻	la 🗧			
Name	Data(DEC)	Data(HEX)	Index	▲
SO	2	2	ビット別タイミング信号	
S1	4	4	現在実装されている軸数(軸ボード×4軸)	
S2	137	89	1秒カウンタ表示	
S3	0	0	全ビットOFFの固定値表示	
S4	-1	fffffff	全ビットONの固定値表示	
85	0	0	エラー表示	
S6	0	0	ラダーで実行したDIV命令(割り算)のあまり	
87	0	0	各レジスタのメモリ書き込み実行	
S8	0	0		
S9	0	0		
S10	0	0	各メモリの読み書き時に発生したエラー	
S11	0	0	ラダーのレジスタ番号が範囲外であるかを表示	
S12	0	0	モーションのレジスタ番号が範囲外、スタックオーバー表示	
S13	0	0		
S14	64856	fd58		
S15	3582	dfe	制御周期(4mS)の内のアイドル時間表示	
S16	N	Π	モーションの~31番の記動状態	 ▶
× Ready. 通信回線を打 接続先:Ver-	_{鏡しました} 1.10			
b7%		接続先:Ver-1.1	10	/µ/(1,0)

表示(⊻)•拡張形式

🛐 ScTool		
ファイル(E) 編集(E)	表示(⊻) 特殊制御(C) ウ	//ʰウ(₩) _^ルフ°⊞)
X Image: Constraint of the second	 ツール パー(T) ステータス パー(S) Info パー(P) アウトフット (R) 標準形式 拡張形式 ・横ライン 画面高速更新 Alt+F5 システム設定 	

レジスタ表示時に、表示形式を拡張形式(ビット表示)で表示します。 ビット表示部は、マウスでクリックする事により、操作が可能です。 (ビット表示部でマウスをクリックした場合、該当する位置に有るビットは反転します)

(表示例)レジスタ・ウィンドウ(システムレジスタ)

回線接続中	ー システムレジス	(列)								
🛐 771N(E)	編集(E) 表示	: ⊘	持殊制	御(<u>C</u>)	ウル	トウ <u>₩</u>) ///	7°(<u>H</u>)		
X 🖻 🖬	la 🗧									
Name	Data(DEC)	28	24	20	16	12	08	04	00	Index 🔺
SO	34									ビット別タイミング信号
S1	4									現在実装されている軸数(軸ボード×
S2	154									1秒カウンタ表示
S3	0									全ビットOFFの固定値表示
S4	-1									全ビットONの固定値表示
S5	0									エラー表示
S6	0									ラダーで実行したDIV命令(割り算)の
S7	0									各レジスタのメモリ書き込み実行 📃
S8	0									
S9	0									
S10	0									各メモリの読み書き時に発生したエミ
S11	0									ラダーのレジスタ番号が範囲外である
S12	0									モーションのレジスタ番号が範囲外、
S13	0									
S14	64856									
S15	3582									制御周期(4mS)の内のアイドル時間表
S16	Π									<u> モーション0~31番の</u> 記動状態 ▼
× Ready. 通信回線を打 接続先:Ver-	接続しました 1.10									
レディ		接続	洗:Ver	-1.10				-		カーソル(1,0)

表示(V)・横ライン

🗐 ScTool		
ファイル(Ё) 編集(Ё)	表示(⊻) 特殊制御(⊆) ウイ	ンドウ (W) _ ヘルプ(H)
X Image: Constraint of the second	 ・ ツール パー(T) ・ ステータス パー(S) ・ Info パー(P) ・ アウトフット(P) ・ アウトフット(P) ・ アウトフット(P) ・ アウトフット (P) ・ マクトフット ・ マクトフット ・ マクトフット ・ マクトフット ・ マクトフット ・ アクトフット ・ アクトフット ・ アクトフット ・ マクトフット ・ マクトラット ・ マクト・シー ・ マクトラット ・ マクト	

ウィンドウの項目間にライン(縦・横)を表示します 2度選択すると元に戻ります。

(表示例)レジスタ・ウィンドウ(システムレジスタ・横ライン表示有り)

回線接続中	- システムレジス	[夕]		_ 8 ×
🗐 771N(E)	編集(E) 表示	(⊻) 特殊制御(©) ウィントや (\\/) ヘルフ°(H)	_ 8 ×
X 🖻 💼	la 🗧			
Name	Data(DEC)	Data(HEX)	Index	▲
SO	2	2	ビット別タイミング信号	
S1	4	4	現在実装されている軸数(軸ボード×4軸)	
S2	137	89	1秒カウンタ表示	
S3	0	0	全ビットOFFの固定値表示	
S4	-1	fffffff	全ビットONの固定値表示	
85	0	0	エラー表示	
S6	0	0	ラダーで実行したDIV命令(割り算)のあまり	
87	0	0	各レジスタのメモリ書き込み実行	
S8	0	0		
S9	0	0		
S10	0	0	各メモリの読み書き時に発生したエラー	
S11	0	0	ラダーのレジスタ番号が範囲外であるかを表示	
S12	0	0	モーションのレジスタ番号が範囲外、スタックオーバー表示	
S13	0	0		
S14	64856	fd58		
S15	3582	dfe	制御周期(4mS)の内のアイドル時間表示	
S16	N	Π	モーション0~31番の記動状態	
× III Decidio				
Heady. 通信回線を打 接続先:Ver-	接続しました 1.10			•
レディ		接続先:Ver-1.1	10 カーソ	%µ(1,0)

(表示例)レジスタ・ウィンドウ(横ライン表示無し)

回線接続中	- [入出力レジス	.刻		_ 8 ×
🗐 ファイル(E)	編集(E) 表示	(⊻) 特殊制御	(©) ウィンドウ(₩) ヘルプ(Ħ)	_ 8 ×
X 🖻 💼	la 🗧			
Name	Data(DEC)	Data(HEX)	Index	
R0	0	0	I/Oボードの入力信号32ビット(書き換え不可)	
R1	0	0	I /Oボードの出力信号32ビット	
R2	0	0	I /Oボードの入力信号32ビット(2枚目のボード実装時)	
R3	0	0	I /Oボードの出力信号32ビット(2枚目のボード実装時)	
R4	0	0	汎用レジスター	
R5	0	0	汎用レジスター	
R6	0	0	汎用レジスタ	
R7	0	0	汎用レジスター	
R8	0	0	汎用レジスタ	
R9	0	0	汎用レジスタ	
R10	0	0	汎用レジスター	
R11	0	0	汎用レジスター	
R12	0	0	汎用レジスター	
R13	0	0	汎用レジスター	
R14	0	0	汎用レジスター	
R15	0	0	汎用レジスター	_
R16	N	Û	汎用レジス々	
× Ready. 通信回線を打 接続先:Ver-	8 続しました 1.10			▲ ▼
ĺ⁄₹°r			接続先:Ver-1.10	

表示(⊻)•画面高速更新

🛐 ScTool			- U X
ファイル(Ĕ) 編集(Ĕ)	表示(V) 特殊制御(C) ウル	/ドウ \W/ _ ヘルプ(<u>H</u>)	
X Image: Constraint of the second	 ツール パー(T) ステータス パー(S) Info パー(P) アウトフット (R) 標準形式 拡張形式 横ライン 画面高速更新 Alt+F5 システム設定 		

表示が自動更新されるモードの時、更新速度を変更します メニュー左側のチェック状態(選択状態)で、更新速度が変わります(下記) 但しこの更新速度は、御使用のPC・SC2xxxにより多少変化する場合があります。

初期状態は「チェック無し(1秒に1回更新)」です

チェック状態	画面更新間隔
1	

表示(<u>∨</u>)・システム設定

ScTool							J	
77イル(<u>F</u>)	表示♡	特殊制御(C)	ታለንՒንን₩)	^ルフ°(<u>H</u>)				
X Ba	> ツール バ	~D. [
	✓ ステータス ・ Turket with	$\mathcal{N} = (\underline{S})$						
🕂 🧰 SC	 ✓ Into /\`` ✓ アウトラ% 	-(P) k (B)						
🗄 🚊 Infe	- / // / /							
	- ジステム語	ŘΞ.						

本ソフトの動作設定を行います 設定後、「確定」を押す事により確定されます

システム設定			×
基本設定 DLL			1
パラメータ表示カラム数	1列 🗨	使用通信回線	(0)RS232C通信 💌
オフセット領域制限	保護	レジス外別期表示	通常表示 💽
起動時ウィントウ	最大表示	起動時復元	復元しない ・
文書整形(インテント)	スペース使用 💌	<u>\$7</u> *ቻ	0
情報ウィントや補助	ウィントウが有れば利 - ウ・	心やり復元時エラー	継続・
ワークスペース設定	標準 💽		参照
表示用フォント名	MS ゴシック ホペ	P 40	参照
印刷用フォント名	MS ゴシック ホッ	9 10	参照
印刷マーシン	上 D 下 D	左□	右□
文書色分け	🥅 使わない		
数值編集色	文字 色 背景	色	
	文字 色 背景	色 予約	 色設定 初期化
71774編集巴	特殊 色 駆動	色 計算	<u>è</u>
		(+ +7)t	

「基本設定タブ」	
項目名	内容
パラメータ表示カラム数	パラメータ表示時の横項目数を設定します (デフォルト: 1列)
使用通信回線	SC2xxxと接続する通信回線種別(RS232C等)を設定します。 通信ポートの設定は、「通信設定」メニューか、別タブ画面(DLL)で 設定する必要があります

	┃(デフォルト: <mark>RS232C</mark>)
オフセット領域	アナログボード搭載時、アナログボードのオフセット値が格納され るレジスタ領域があります。 その領域に対して、「ファイル読み込み」「リストア」類で本来の値が 書き換わる事を防ぐ為の設定です。 保護を無効にする為には、「無制限」に設定する必要があります (デフォルト:保護)
レジスタ初期表示	レジスタウィンドウを開いた場合に、初期表示形態を通常/拡張に 設定します。 (デフォルト:通常)
起動時ウィンドウ	起動時にトップレベルのウィンドウを最大表示する為の設定です。 (デフォルト: <mark>最大表示</mark>)
起動時復元	起動時に任意のウィンドウ状態を復元する為の設定です。 対象となる設定は、「ウィンドウ」メニューで保存が可能な設定1~ 設定4となります。 起動時に自動でウィンドウ設定を戻さない場合には、「復元しない」 にチェックを入れる必要があります (デフォルト:復元しない)
文書整形(インデント)	文書(モーション・ラダー)整形時にスペース・タブのどちらを使用す るか設定します。 (デフォルト: <mark>スペース使用</mark>)
タブサイズ	文書(モーション・ラダー)入力・整形時に使用されるタブサイズを 設定します。(0を設定すると4になります) (デフォルト:0)
情報ウィンドウ補助	Infoバーに表示される「レジスタ/プログラムへのショートカット」を ダブルクリックした時の挙動を設定します。 「ウィンドウが有る場合に、そのウィンドウに移動する」又は「常にウ ィンドウを開く」設定が行えます。 (デフォルト: ウィンドウが有れば利用)
ウィンドウ復元時エラ ー	ウィンドウの復元時にエラーが発生した場合、その後の挙動を決 定します。 (デフォルト: <mark>継続</mark>)
ワークスペース設定	「バックアップ/リストア」等を行う時にデータが格納されるフォルダ を指定します。 (「標準」を選択した状態では、プログラムをインストールしたディレ クトリに作成されます。) 「参照」ボタンを押す事により、任意のフォルダを指定する事が可 能です。 (デフォルト:標準)
表示用フォント名・ポイント	各ウィンドウの表示に使用するフォント及びフォントサイズを指定します。 「参照」ボタンを押す事により、フォントダイアログから選択する事が可能です
印刷フォント名・ポイント	印刷に使用されるフォント及びフォントサイズを指定します。 「参照」ボタンを押す事により、フォントダイアログから選択する事 が可能です。
	印刷時の上下左右マージン(余白)をミリ単位で設定します。

文書色分け	文書(モーション・ラダー)編集時に、予約語に関して色をつけるか どうか設定します。 (デフォルト:使う(チェック無し))
数値編集色	レジスタ編集時、数値入力時に表示される文字色/背景色を設定 します。 「参照」ボタンを押すことにより、カラーダイアログから選択する事 が可能です。
プログラム編集色	文書(モーション・ラダー)編集時、特定の言葉に関して文字色/背 景色を設定します。 「参照」ボタンを押すことにより、カラーダイアログから選択する事 が可能です。

ステム設定 基本設定 DLL	<
Name	File
RS232C)通信	C:¥Program Files¥ScTool¥Plugin¥ScCOM232.dll
	著作権情報 設定

「DLLタブ」

項目名	内容
著作権情報	選択された拡張機能の著作権情報を表示します。
設定	選択された拡張機能(プラグイン)の設定画面を表示します。 通信回線種別に関する機能の場合、通信に使用するポートの設定 等が有ります。 他の拡張機能に関しても設定項目が有る場合があります。 詳細については、各拡張機能のマニュアルを参照願います

特殊制御(C)メニュー

ScTool		
ファイル(E) 表示(⊻)	特殊制御(<u>C</u>) ウィント	ウ(₩) ヘルフ°(H)
X b c 3	回線接続(Q) 回線切断(Q)	
	回線設定(S)	
	パックアップ(<u>B</u>) リストア(<u>R</u>)	
	7ラッシュROM(<u>F</u>) 🕨	
	特殊機能 ▶	

レジスタ/プログラム類ウィンドウ選択時

ScTool		
ファイル(E) 編集(E) 表示(⊻)	特殊制御(<u>C</u>) ウルトウ(₩) ^µフ°(<u>H</u>)
X & C 😂 💡	回線接続(Q) 回線切断(<u>C</u>)	
	回線設定(S)	
	パッウアッフ [*] (<u>B</u>) リストア(<u>R</u>)	
	アラームリセット(<u>R</u>) フラッシュROM(<u>F</u>) ・	
	特殊機能 ▶	

このメニューには、下記7点の項目があります (項目名をクリックすると、詳細説明に移動します)

<u>回線接続</u> <u>回線切断</u> <u>回線設定</u> <u>バックアップ</u> <u>リストア</u> <u>アラームリセット</u> <u>フラッシュROM</u> 特殊機能

特殊制御(<u>C</u>)·回線接続(<u>O</u>)



通信回線を使用して、SC2xxxと接続します。 (通常、SC2xxxと接続が必要となった場合には、自動的に接続されます) 使用する回線の設定は、「<u>システム設定</u>」の接続先と、「<u>回線設定</u>」の設定が使用されますので、 事前に合わせる必要が有ります。

特殊制御(\underline{C})·回線切断(\underline{C})



SC2xxxとの「通信回線での接続」を切断します

通信回線を明示的に切断します。以降、回線を接続(自動接続含む)するまで、SC2xxxとの通信 は行えません。

特殊制御(<u>C</u>)・バックアップ(<u>B</u>)



SC2xxxから全データを取得し、ファイルとして保存します 起動すると下記ウィンドウが表示されます

ハ [、] ックアッフ ^{°(} PC <- SC2K)	×
形式	
℃ Ver2 形式(一括形式	式)
✓ Ver1 形式(プロジェク	下形式》
ファイル名指定	
● 日付(yyyymmdd_hh	immss)
○ 任意	参照
言羊糸田	
作成日	
最終保存日	
会社名	
担当者	
バージョン	
□ バックアップ前に定義マ	を初期化
	キャンセル 実行

形式

ver1形式

データ種類別のファイル(複数ファイル)

ver2形式

全てのデータを包括する単一ファイル(現在使用不可)

ファイル名指定

日付(yymmdd_hhmmss)

バックアップ時点の時間を元にしたファイルネームで保存します 同名のフォルダの中に作成されます。(推奨)

任意

ユーザーで任意のファイル名を付けます

詳細

作成日

最初にデータが作成された年月日を入力します(空欄可)

最終保存日

最後に保存した年月日が入力されます(自動)

会社名

作成した会社名を入力します(空欄可)

担当者

作成した担当者を入力します(空欄可)

バージョン

バージョン番号を入力します(空欄可)

特殊制御(C)・リストア(R)

🗐 ScTool		
ファイル(E) 編集(E) 表示(⊻)	特殊制御(C) ウイントウ(W)	^⊮7°(<u>H</u>)
X 🖻 🖻 🎒 🤗	回線接続(Q) 回線切断(C)	
	 ハ [*] ックアッフ [*] (<u>B</u>)	
	УХХ7(<u>R</u>)	
	アラームリセット(<u>R</u>) フラッシュROM(<u>F</u>) ト	
	特殊機能 ▶	

起動すると下記ウィンドウが表示されます

「参照」を押して、バックアップしたファイルがあるディレクトリの「*.PSC」ファイルを指定して下さい

リストア(PC → SC2K)			×
形式			
C Ver2 形式(-	・括形式)		
● Ver1 形式(7	"ロシ゛ェクト用	(元3	
ファイル名指	定		
C 日付(yyyym	ndd_hhm	mss)	
⊙ 任意			参照
言羊糸田			
作成日			
最終保存日			
会社名			
担当者			
バージョン			
		キャンセル	実行

形式

ver1形式

データ種類別のファイル(複数ファイル)

ver2形式

全てのデータを包括する単一ファイル(現在使用不可)

ファイル名指定

バックアップされたデータの中に「.PSC」という拡張子のファイルがありますので、それを指定して下さい

詳細

作成日

最初にデータが作成された年月日が表示されます

最終保存日

最後に保存した年月日が表示されます

会社名

作成した会社名が表示されます

担当者 作成した担当者が表示されます バージョン バージョン番号が表示されます

特殊制御(<u>C</u>)・アラームリセット(<u>R</u>)



SC2xxxで発生しているアラームを解除します このメニューは、通信回線が接続されている状態のみ有効となります

特殊制御(<u>C</u>)・フラッシュROM(<u>F</u>)



SC2xxxに搭載されているフラッシュROMを制御します <u>保存</u> <u>読出</u>

特殊制御(<u>C</u>)・フラッシュROM(<u>F</u>)・読出(<u>R</u>)

ScTool	×
ワァイル(E) 表示(V) 特殊制御(C) ウィンドウ(W) ヘルプ(H)	
※ 略 配 合	_
<u>パックアップ(B)</u> リストア(B)	
フラッシュROM(E) ▶ 保存(S)	
特殊機能	
** ScTool for SC2xxx series ** Copyright(C)2000-2004 WACOGIKEN Co.,ltd. Software version : 2.0.9.7 Ready.	
	-
C2xxxの7ラッシュROMからデータを戻します/フラッシュ読出	

SC2xxxの不揮発性記憶領域(フラッシュROM)に格納されているデータを、揮発性記憶領域(RAM)に呼び出します

フラッシュROM 読出領域	選択
🗖 እንተንጉሥን አል (P \	√0 レジスタが該当)
🔲 ファイルレジスタ(F レ	ジスタが該当)
🔲 パラメータ (C レジスタ	が該当)
🗖 モーション	
□ ラダー	
□ テキストレジスタ(T レ)	ジス対談当)
全選択	キャンセル 確定
ドライブレシ	ジスタ

3種類のレジスタが対応します P(位置)レジスタ V(速度)レジスタ O(オーダ)レジスタ ファイルレジスタ 1種類のレジスタが対応します F(ファイル)レジスタ パラメータ 1種類のレジスタが対応します C(パラメータ)レジスタ

モーション

1種類のプログラムが対応します モーションプログラム

ラダー

3種類のプログラムが対応します

ラダープログラム:ラダーのプログラム ラダーの定義:ラダー内にある定義文(#define) ラダーのコメント:ラダー内にあるコメント(//で始まる行)

テキスト

1種類のレジスタが対応します T(テキスト)レジスタ

特殊制御(<u>C</u>)・フラッシュROM(<u>F</u>)・保存(<u>S</u>)

ScTool	X
ファイル(E) 表示(V) 特殊制御(C) ウィントウ(W) ヘルフ°(H)	
	_
回線設定(S)	
ハ [*] ックアッフ [*] (<u>B</u>) リストア(<u>B</u>)	
フラッシュROM(E) ▶ 保存(S)	
特殊機能	
× ** ScTool for SC2xxx series ** Copyright(C)2000-2004 WACOGIKEN Co.,Itd. Software version : 2.0.9.7 Ready.	
ーー SC2xxxのフラッショROMに保存します/フラッシュ保存	

SC2xxxの揮発性記憶領域(RAM)に格納されているデータを、不揮発性記憶領域(フラッシュR OM)に保存します



T(テキスト)レジスタ

特殊制御(C)•特殊機能

🗐 ScTool	
ファイル(E) 編集(E) 表示(V	特殊制御(C) ウィントウ(W) ヘルフ°(H)
) X 🖻 🖻 🎒 🤗	回線接続(Q) 回線切断(C)
E	回線設定(S)
⊞ <mark></mark> Workspace	<u>パックアップ(B)</u> リストア(R)
	アラームリセット(<u>R)</u> フラッシュROM(<u>F</u>) ・
	特殊機能 消去 再起動 モーション動作状況

SC2xxxの特殊機能を使用します。 (SC2xxxのファームウェアバージョンにより、使用出来る機能は異なります) <u>消去</u> <u>再起動</u> <u>モーション動作状況</u>

特殊制御(<u>C</u>)•特殊機能•消去

🛐 ScTool		
ファイル(Ё) 表示(⊻)	特殊制御(<u>C</u>) ウィント	^ነ ታ \\\// ^\\L_?\\L
X h C 4	回線接続(0) 回線切断(<u>C</u>)	
	回線設定(S)	
	ハ [、] ックアッフ [。] (<u>B</u>) リストア(<u>R</u>)	
	7ラッシュROM(<u>F</u>) ▶	
	特殊機能 ▶	· 消去···································
		再起動 モーション動作状況

SC2xxxの不揮発性記憶領域に格納されているデータの内、指定する領域を初期化します。 メニューを選択すると、下記ウィンドウが表示されます



ドライブレジスタ 3種類のレジスタが対応します P(位置)レジスタ V(速度)レジスタ O(オーダ)レジスタ ファイルレジスタ 1種類のレジスタが対応します F(ファイル)レジスタ パラメータ 1種類のレジスタが対応します C(パラメータ)レジスタ モーションプログラムが対応します モーションプログラム

> 3種類のプログラムが対応します ラダーのプログラム ラダー内にある定義文(#define) ラダー内にあるコメント(//で始まる行)

テキスト

1種類のレジスタが対応します T(テキスト)レジスタ

ラダーの定義

1種類のプログラムが対応します

ラダー内にある定義文(#define)

(ファームウェアのバージョンによって使えない場合があります)

ラダーのコメント

1種類のプログラムが対応します

ラダー内にあるコメント(//で始まる行)

(ファームウェアのバージョンによって使えない場合があります)

チェックを決定後、「確定」を押す事により、初期化を行います。

特殊制御(C)•特殊機能•再起動

🗐 ScTool			
ファイル(E) 表示(⊻)	特殊制御(<u>C</u>) ウルト	ֻלַיַשׂ ∧µס°(<u>H</u>)	
X 🖻 🖻 🕭	回線接続(Q) 回線切断(<u>C</u>)		
	回線設定(S)		
	ハ [、] ックアッフ [。] (<u>B</u>) リストア(<u>R</u>)		
	7ラッシュROM(<u>F</u>) ・		
	特殊機能 ▶	消去	
		再起動	
		モーション動作状況	

現在接続しているSC2xxxを再起動(電源再投入と同じ)を行います。

実行すると下記の状態になります

変更されているレジスタは、フラッシュROMに記録されている値に戻ります 変更した事でついた「変更済み表示」は、「未変更」時の状態に戻ります ネットワークボード搭載時、IP設定が「DHCPサーバからの取得」の場合は再取得さ れます

特殊制御(C)・特殊機能・モーション動作状況



SC2xxxで動作中のモーション状態を取得します。

メニューを選択すると、モーションを読み込んだ後、下記ウィンドウ(表示内容は一例)が表示され ます

(表示迄に時間が掛かる場合があります)

🔟 ScTool - [モーション実行状態	£]									
📃 ファイル(E) 編集(E) 表示	₹(⊻) 特殊制御	(©) ウィンド	ウ₩) ∿ルフ	и°(Н)	_ 8 ×					
X 🖻 🖬 🎒 🤗 📔										
×	プログラム番号	実行状態	行番号	7°ロク*ラムコート*	<u>▲</u>					
〇 軸レジ'スタ(A)	0	Running	3	F0 = F0 + 1						
	1	Running	9	F1 = F1 + 1						
□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	2	Running	15	F2 = F2 + 1						
	3	Running	21	F3 = F3 + 1						
	4	Stopped								
	5	Stopped								
	6	Stopped								
	7	Stopped								
······································	8	Stopped								
 〇 実数レジスタ(Q) 	9	Stopped								
- • モーション(MOT)	10	Stopped								
	11	Stopped								
🗄 💼 Information	12	Stopped								
🗄 💼 Workspace 🛛 🥃	13	Stopped			-					
▲	14	Stoppod								
× エラーはありません										
1										
					_					
ν 7 °r					//.					

各個所の説明

プログラム番号

モーションのプログラム番号に該当します

実行状態

モーションのプログラム状態を表示します。 「Running」...実行中

「Stopped」 停止中

行番号

モーションの実行中行番号を表示します

「プログラムコード」

実行中のプログラムコードを表示します

実行(R)メニュー



ラダーウィンドウ選択時

ScTool								
ファイル(<u>E</u>)	編集(E)	表示⊙	特殊制御(<u>C</u>)	実行(R)	ሳሪኦንን∰	^/μフ°(<u>H</u>)		
X Ba	6 4	?		- ラダー保	存⊗			

このメニューには、下記2点の項目があります

<u>モーション保存</u> <u>ラダー保存</u>

実行(X)・モーション保存

ScTool								
771H(E)	編集(E)	表示♡	特殊制御(<u>C</u>)	実行(R)	<u> </u>	^⊮フ°(<u>H</u>)		
X B	6 4	?		モーション	保存⊗			

編集中のモーションを、SC2xxxに転送します。 実行時にモーションのエラーチェックも行われ、結果は「<u>アウトプットウィンドウ</u>」(<u>リザルトウィンド</u> ウ)に表示されます。

実行(X)・ラダー保存(X)

ScTool		
ファイル(E) 編集(E) 表示	(⊻) 特殊制御(⊆)	実行(R) ウィントウ(W) ヘルフ*(H)
X 🖻 🖻 🎒 🤶		

編集中のラダーを、SC2xxxに転送します。

実行時にラダーのエラーチェックも行われ、結果は「<u>アウトプットウィンドウ</u>」(<u>リザルトウィンドウ</u>)に 表示されます。

ウィンドウ(W)メニュー

レジスタ/プログラム類ウィンドウ未選択時

🗐 ScTool					×
771N(E)	表示⊙	特殊制御(2)	<u> ሳለንኑን (₩</u>)	^₩7°(<u>H</u>)	
X B	6 4	?	画面状態	•	

レジスタ/プログラム類ウィンドウ選択時

🗐 ScTool					
ファイル(E) 編集	(E) 表示(⊻)	特殊制御(<u>C</u>)	ウィンドウ(₩)	^ルフ°(<u>H</u>)	
X 🖻 🖬 (∰ ?		重ねて表示 並べて表示	⊼(<u>©</u>) ₹ ►	
			画面状態	•	

このメニューには、下記3点の項目があります (項目名をクリックすると、詳細説明に移動します)

<u>重ねて表示</u> <u>並べて表示</u> <u>画面状態</u>

ウィンドウ(<u>W</u>)・重ねて表示(<u>C</u>)

🛐 ScTool			
ファイル(E) 編集(E) 表示(V) 特殊制	御心) ウィントウ(W)	Λ//7°(<u>H</u>)	
	<u>重ねて表示</u> 並べて表示	⊼(<u>C)</u> ⊼ •	
	画面状態	•	

開いているウィンドウを重ねて表示します。

ウィンドウ(W)・並べて表示



このメニューには、下記2点の項目があります

<u>縦方向</u> 横方向

ウィンドウ(W)・並べて表示・縦方向

ScTool							
771N(E)	編集(E)	表示♡	特殊制御(①)	ウィンドウ(₩)	^/μフ°(<u>H</u>)		
	-2000	? ──×	-	重ねて表示 並べて表示		縦方向 横方向	_
	2000			画面状態	t	1,C)TH	

開いているウィンドウを縦方向に並べます

(実行例)

ĺ	ScTool ファイル(F)	編集(E)	表示(V)	特殊制御(C)	<u>ሳ</u> ረንኑነት (₩)	ヘルフ [*] (H)	_ & ×
Ī	X 酉 (8 6	8				
	<mark>創</mark> 汎用レ	ジスタ					_ 🗆 🗵
l	Name	Da	ta(DEC)	Data(HEX)	Index		▲
l	RO	0		0	Ⅰ/Οボー	ドの入力信号32ビット(書き換え不可)	
I	R1	0		0	I∕Oボー	ドの出力信号32ビット	
I	R2	0		0	I/Oボー	「ドの入力信号32ビット(2枚目のボード実装時)	
I	R3	0		0	I/Oボー 2月11日 - 2月1日	「ドの出力信号32ピット(2枚目のボード実装時)	
Į	R4 ◀				N⊞15*ZX	~	
	🗖 સનગંગગ						- 🗆 🗵
l	prg O,	moni					
l	;動1作日 10・	529					
l	R10.0	= F22.0					
I	R10.1	= F24.0					
l	R10.2	= F23.0					
Ľ	F25 =	AU.2					
2	· エラーlよあ	りません					A
1]						∇
V	ディ					カーソル(2,11)	
•	👖 Հջ–ի	·) 🥭	😂 🗹 🛛	ScTool		Ywindow_tile_horz.bmp - ペ	<u>A</u> 14:10

ウィンドウ(W)・並べて表示・横方向

ScTool								_	
771N(E)	編集(E)	表示♡	特殊制御(<u>C</u>)	ウィンドウ₩)	^#7°(<u>H</u>)				
X 🖻	6 4	?		重ねて表示	R(<u>C</u>)	***	1		
		× [一业へて表力	۲ ۲	能力回			
	2000			画面状態). 	便方回	J		

開いているウィンドウを横方向に並べます

(実行例)

ScTool					_ 8 ×
ファイル(E)	編集(E) 表示(V)	特殊制御(<u>C</u>)	ታለንՒንን₩)	^ルフ <u>゚(H</u>)	
X 🖻	648				
圓 汎用I	バスタ		_ 🗆 ×	■モーション	- O ×
Name	Data(DEC)	Data(HEX)	Index 🔺	prg O, moni	_
RO	0	0	I/0#] ;動作モニタ	
R1	0	0	I∕O#	L10:	
R2	0	0	I∕Oħ	R10.0 = F22.0	
R3	0	0	I∕O‡	R10.1 = F24.0	
R4	0	0	汎用レジ	R10.2 = F23.0	
R5	0	0	汎用レジ	F25 = AU.2	
R6	0	0	汎用レジ	F20 - A1.2	
R7	0	0	汎用レジ	$F_{20} = A_{20} = A$	
R8	0	0	汎用レジ	$F_{20} = H_{0.2}$	
R9	0	0	汎用レジ	$F_{29,1} = S16_{20}$	
R10	0	0	汎用レジ	$F_{29,2} = S_{22,28}$	
R11	0	0	汎用レジ	F29.3 = S22.18	
R12	0	0	汎用レジー	wait 4	
				goto L10	
× エラーlよを	うりません			-	A.
					~
レディ				カーソル (2,11)	
🚮 スター	- [] 🥭 😂 🚺	ScTool			<mark>.∢</mark> € <u>.</u> A_ 14:08

ウィンドウ(W)・画面状態

ScTool							. D × D.
771N(E)	表示⊙	特殊制御(<u>C</u>)	<u> ሳሪኑን(₩</u>)	^//7°	(H)		
X Ba	6 6	?	画面状態	•	復元	ł	
		×			1本1士	<u> </u>	

このメニューには、下記2点の項目があります

<u>復元</u> <u>保存</u>

ウィンドウ(W)・画面状態・復元

🗐 ScTool			
ファイル(E) 表示(V) 特殊制御(C) ሳለኦንሙ ^ルプ(<u>H</u>)		
X Image: Constraint of the second seco	画面状態 → 復元 保存	 ▶ 設定1 ▶ 設定2 設定3 設定4 	

このメニューには、下記4点の項目があります

設定1

以前に「設定1」に保存されたウィンドウ位置設定を復元します 設定2

以前に「設定2」に保存されたウィンドウ位置設定を復元します

設定3

以前に「設定3」に保存されたウィンドウ位置設定を復元します

設定4

以前に「設定4」に保存されたウィンドウ位置設定を復元します

「復元についての詳細」

復元されるのは、下記の設定のみです。

ウィンドウのモード(新規作成で選択したモード) ウィンドウ位置

(復元されるウィンドウが、「通信」を必要としる場合には、自動で接続処理を行います。 但し、接続処理が失敗した場合は、そのウィンドウは復元されません。 又、「モーション」「ラダー」の復元がある場合に、ウィンドウを復元する途中で、編集するデ ータに関しての問い合わせが表示されます。 (次に進むには、その問い合わせに対して、答えを選択する必要があります)

ウィンドウ(W)・画面状態・保存



このメニューには、下記4点の項目があります

設定1

「設定1」に現在のウィンドウ位置設定を保存します 設定2

「設定2」に現在のウィンドウ位置設定を保存します 設定3 「設定3」に現在のウィンドウ位置設定を保存します 設定4

「設定4」に現在のウィンドウ位置設定を保存します 「保存についての詳細」

保存されるのは、下記の設定のみです。

ウィンドウのモード(新規作成で選択したモード) ウィンドウ位置 最小化・最大化状態

注意! ウィンドウ内のデータが保存されるわけではありません

ヘルプ(H)メニュー

ScTool						
ファイル(<u>E</u>)	表示⊙	特殊制御(C)	ታለንՒንን₩)	ヘルフ°(<u>H</u>)		
1 X 🖻	6 6	? ×		∧ルフ°(<u>H</u> ScTool) のバージョン情報(<u>A</u>)…	

このメニューには、下記2点の項目があります

<u>ヘルプ</u> ScToolのバージョン表示/バージョンアップ

ヘルプ(<u>H</u>)・ヘルプ(<u>H</u>)

ScTool						
771N(E)	表示⊙	特殊制御(<u>C</u>)	ታለንՒንን₩)	ヘルフ°(<u>H</u>)		
X 🖻 🔂 🔗 🧣				∧µフ°(<u>H</u>)	
		×		Sclool	のハージョン情報(<u>A</u>)	

オンラインヘルプを表示します

ヘルプ(<u>H</u>)・ScToolのバージョン情報(<u>A</u>)

ScTool						
771N(E)	表示♡	特殊制御(2)	ታィンՒኻ₩)	ヘルフ°(<u>H</u>)		
X B	C #	?		^//7°(<u>H</u>) - のいら	
-		x		301001	())) ニクヨノ旧ギII(<u>円</u> /	

本ツールのバージョン番号を表示します



「OK」を押すとバージョン表示を閉じます 「<u>Firmware</u>」を押すとSC2xxx搭載のファームウェアアップデートが行えます

目的別:通信接続まで



目的別:レジスタ編集

初期状態からレジスタ編集終了迄の手順




目的別:パラメータ編集

初期状態からパラメータ編集終了迄の手順





目的別:モーション編集





目的別:ラダー編集





リファレンス選択

モーションで使用できる関数・コマンド一覧 <u>アルファベット順一覧</u> 機能別一覧 ラダーで使用できる関数・コマンド一覧 <u>アルファベット順一覧</u> 機能別一覧

リファレンス選択

モーションで使用できる関数・コマンドー覧 アルファベット順一覧 機能別一覧

リファレンス一覧(モーション)

モーションで使用できる関数・コマンドー覧(アルファベット順)

アルファベット順			
<u>代入文</u>	<u>コメント文</u>	abs	acos
asin	atan	atan2	ceil
close	COS	else	elseif
endif	float	floor	getcom
getl	gosub	goto	if
int	movc	movci	movh
movm	<u>movmi</u>	movo	movs
movsi	movt	movti	open
padj	<u>PI</u>	<u>PI2</u>	pow
prg	<u>print</u>	prio	pset
putcom	putl	reset	<u>return</u>
scan	seplh	sin	sqrt
stklh	stop	sub	tan
wait			

リファレンス一覧(機能別一覧)

プログラム制御				
prg	sub	if	elseif	
else	<u>endif</u>	goto	gosub	
return	wait	stop	prio	
駆動/軸関係				
movh	movs	movsi	movm	
<u>movmi</u>	movc	movci	movo	
movt	<u>movti</u>	pset	padj	
エラー/その他	エラー/その他			
reset	<u>代入文</u>	<u>コメント文</u>		
関数				
sin	COS	tan	asin	
COS	tan	<u>tan2</u>	sqrt	
pow	ceil	<u>floor</u>	<u>float</u>	
int	<u>abs</u>	<u>PI</u>	<u>PI2</u>	
printf	scan	seplh	stklh	
通信関係				
open	close	getl	putl	
getcom	putcom			

モーションで使用できる関数・コマンド一覧(機能別)

代入文

書式

<reg>=<expr>

説明

右辺の計算式の結果を左辺のレジスタに代入します。 左辺、右辺ともワード指定、ビット指定が可能で、左辺がビット指定、右辺がワード指定の場 合右辺の最下位ビットが代入されます

コメント文

書式

;(セミコロン) <remarks>

説明

;(セミコロン)の後に任意の文字列を書くことが出来ます。

#define

書式

#define <label> <arg>

説明

```
<arg>を別名<label>で定義します。
定義した行以降 で、別名が使用可能となります。
SC2xxxから読み出す時に「逆変換」されている為、逆変換すると困る様な定義は御止めく
ださい
```

#include

書式

#include <filename>

説明

<filename>のファイルをその位置に展開します。 (この命令は、現在動作いたしません)

abs()

書式

```
abs(<expr>)
```

説明

```
<expr>の絶対値を計算します。<expr>が整数の場合結果は整数、実数の場合実数を返します。
```

asin(),acos(),atan(),atan2()

書式

```
asin(<expr>)
acos(<expr>)
atan(<expr>)
atan2(<expr:y>,<expr:x>)
```

説明

<expr>のasin,acos,atanの計算を行います。角度の表現はradです。 atan2は<expr:y>,<expr:x>の2つの因数から計算され、結果は±πで得られます。

asin(),acos(),atan(),atan2()

書式

```
asin(<expr>)
acos(<expr>)
atan(<expr>)
atan2(<expr:y>,<expr:x>)
```

説明

```
<expr>のasin,acos,atanの計算を行います。角度の表現はradです。
atan2は<expr:y>,<expr:x>の2つの因数から計算され、結果は±πで得られます。
結果は実数を返します。
```

asin(),acos(),atan(),atan2()

書式

```
asin(<expr>)
acos(<expr>)
atan(<expr>)
atan2(<expr:y>,<expr:x>)
```

説明

<expr>のasin,acos,atanの計算を行います。角度の表現はradです。 atan2は<expr:y>,<expr:x>の2つの因数から計算され、結果は±πで得られます。 結果は実数を返します。

asin(),acos(),atan(),atan2()

書式

```
asin(<expr>)
acos(<expr>)
atan(<expr>)
atan2(<expr:y>,<expr:x>)
```

説明

<expr>のasin,acos,atanの計算を行います。角度の表現はradです。 atan2は<expr:y>,<expr:x>の2つの因数から計算され、結果は±πで得られます。 結果は実数を返します。

ceil(),floor()

書式

```
ceil(<expr>)
floor(<expr>)
```

説明

<expr>の小数点以下の、ceilは切り捨て、floorは切り上げを行います。 結果は実数を返します。

close

書式

close <FileNo>

説明

open命令で開いたファイルを閉じます。 <FileNo>には、open命令を使用したときの返り値(変数RES)の値を指定します。

sin(),cos(),tan()

書式

```
sin(<expr>)
cos(<expr>)
tan(<expr>)
```

説明

<expr>のsin,cos,tanの計算を行います。角度の表現はradです。 結果は実数を返します。

if,elseif,else,endif

書式

```
if <expr>
elseif <expr>
else
endif
```

説明

ifから始まりendifで終了する条件命令です。 間にelseif,elseをはさむ事によって条件から1つを選択する事が出来ます。

例

```
if <expr1>

····· <expr1>が真なら実行します

elseif <expr2>

····· <expr2>が真なら実行します
```

```
elseif <expr3>
・・・・・ <expr3>が真なら実行します
else
・・・・・ <expr1><expr2><expr3>いずれも真で無いなら実行します
endif
```

if,elseif,else,endif

```
書式
   if <expr>
   elseif <expr>
   else
   endif
説明
   ifから始まりendifで終了する条件命令です。
   間にelseif,elseをはさむ事によって条件から1つを選択する事が出来ます。
例
   if <expr1>
   ・・・・・ <expr1>が真なら実行します
   elseif <expr2>
   ・・・・・ <expr2>が真なら実行します
   elseif <expr3>
   ・・・・・ <expr3>が真なら実行します
   else
   ・・・・・ <expr1><expr2><expr3>いずれも真で無いなら実行します
   endif
```

if,elseif,else,endif

```
書式
   if <expr>
   elseif <expr>
   else
   endif
説明
   ifから始まりendifで終了する条件命令です。
   間にelseif,elseをはさむ事によって条件から1つを選択する事が出来ます。
例
   if <expr1>
   ・・・・・ <expr1>が真なら実行します
   elseif <expr2>
   ・・・・・ <expr2>が真なら実行します
   elseif <expr3>
   ・・・・・ <expr3>が真なら実行します
   else
```

```
••••• <expr1><expr2><expr3>いずれも真で無いなら実行します
endif
```

float(),int()

書式

```
float(<expr>)
int(<expr>)
```

説明

floatは<expr>を実数に型変換します。 intは<expr>は整数に型変換します。

ceil(),floor()

書式

ceil(<expr>) floor(<expr>)

説明

<expr>の小数点以下の、ceilは切り捨て、floorは切り上げを行います。 結果は実数を返します。

getcom

書式

getcom <port>,<timeoutMS>

説明

<port>で指定された通信ポートから1バイト取得します。 <timeoutMS>で指定される時間(単位:ms)受信するデータが無かった場合、エラーで戻りま す。 変数RESに受信したデータが返ります。(0未満でエラー)

getl,putl

書式

getl <FileNo>,<Text_reg> putl <FileNo>,<Text_reg>

説明

getlはopen命令で開いたファイルから1行取得して、〈Text_reg〉で指定されるテキストレジス

タに格納します。 putlはopen命令で開いたファイルに、<Text_reg>で指定されるテキストレジスタの内容を格納 します。 <FileNo>には、open命令を使用したときの返り値(変数RES)の値を指定します。

gosub

書式

gosub <label>,<arg1>,<arg2>,<arg3>...

説明

prgまたはsubのラベルに分岐します。 分岐先でreturn命令を実行すると呼び出したgosub命令の次の行に戻ります。 スタックが一杯になるとs12.0がON し、エラーを知らせます。 arg1(1~)が指定されている場合、分岐先でY16~(実数はZ16~)で使用する事が可能に なります。

例

prg 0 gosub test, 1 , 2 , 3.0 , 4 , 5.2 stop

```
sub test
(この時点で、下記変数が自動的に設定されています)
|
Y16 = 1
Y17 = 2
Z18 = 3.0 (実数レジスタに入るので注意)</FONT >
Y19 = 4
Z20 = 5.2 (実数レジスタに入るので注意)</FONT >
|
return
```

goto

書式

goto <label>

説明

```
現在のプログラム中へのラベルへ分岐します。
ラベルを探す範囲はprg又はsubで始まる1つのプログラム内で、他のプログラム中に同一の
ラベルがあっても参照されません。
```

if,elseif,else,endif

書式

```
if <expr>
elseif <expr>
else
endif
```

説明

例

```
ifから始まりendifで終了する条件命令です。
間にelseif,elseをはさむ事によって条件から1つを選択する事が出来ます。
if <expr1>
・・・・・ <expr1>が真なら実行します
elseif <expr2>
・・・・・ <expr2>が真なら実行します
elseif <expr3>
・・・・・ <expr3>が真なら実行します
else
・・・・・ <expr1><expr2><expr3>いずれも真で無いなら実行します
endif
```

float(),int()

書式

```
float(<expr>)
int(<expr>)
```

説明

floatは<expr>を実数に型変換します。 intは<expr>は整数に型変換します。

movc

書式

movc <expr:ax>

説明

movcは絶対値、movciは相対値円弧補完運転の起動を行います。 円弧補完運転が終了するとプログラムは次のステップへ進みます。 命令の先駆けて位置(A口.17)、速度(A口.18)、駆動軸指定(A口.22)、円弧通過点の位置(A 口.26)、円弧補完を行う軸(A口.27)の指定を行います。 実行開始する為にmovc,movci命令はA口.16.6をONにしますが,既に命令実行中の場合、実 行終了を待ってから行います。 A口.16.6がONのままだと命令は無視されますので注意が必要です。 実行終了後、A口.16の全ビットはOFFされます。 詳細はSC2000取扱説明書の【4.1.5円弧運転】を参照して下さい。

movc

書式

movc <expr:ax>

説明

moveは絶対値、moveiは相対値円弧補完運転の起動を行います。 円弧補完運転が終了するとプログラムは次のステップへ進みます。 命令の先駆けて位置(A口.17)、速度(A口.18)、駆動軸指定(A口.22)、円弧通過点の位置(A 口.26)、円弧補完を行う軸(A口.27)の指定を行います。 実行開始する為にmove,movei命令はA口.16.6をONにしますが,既に命令実行中の場合、実 行終了を待ってから行います。 A口.16.6がONのままだと命令は無視されますので注意が必要です。 実行終了後、A口.16の全ビットはOFFされます。 詳細はSC2000取扱説明書の【4.1.5円弧運転】を参照して下さい。

movh

書式

movh <expr:ax>

説明

パラメータに指定された原点復帰方法で<expr:ax>で指定する1軸を原点復帰し、動作終了 後次のステップへ進みます。

命令に先駆けて速度(A口.18)の指定を行います。

実行開始する為にmovh命令はA口.16.7をONにしますが、既に命令実行中の場合、実行終 了を待ってから行います。

A口.16.7がONのままだと命令は無視されますので注意が必要です。

実行終了後、A口.16の全ビットはOFFされます。

詳細はSC2000取扱説明書の【4.1.2】原点復帰を参照して下さい。

movm,movmi

書式

movm <expr:ax> movmi <expr:ax>

説明

movmは絶対値、movmiは相対値多軸運転の起動を行います。 多軸運転が終了するとプログラムは次のステップへ進みます。 命令の先駆けて位置(A口.17)、速度(A口.18)の指定を行います。 実行開始する為にmovm,movmi命令はA口.16.3をONにしますが,既に命令実行中の場合、 実行終了を待ってから行います。 A口.16.3がONのままだと命令は無視されますので注意が必要です。 実行終了後、A口.16の全ビットはOFFされます。 詳細はSC2000取扱説明書の【4.1.4多軸運転】を参照して下さい。

movm,movmi

書式

movm <expr:ax> movmi <expr:ax>

説明

movmは絶対値、movmiは相対値多軸運転の起動を行います。 多軸運転が終了するとプログラムは次のステップへ進みます。 命令の先駆けて位置(A口.17)、速度(A口.18)の指定を行います。 実行開始する為にmovm,movmi命令はA口.16.3をONにしますが,既に命令実行中の場合、 実行終了を待ってから行います。 A口.16.3がONのままだと命令は無視されますので注意が必要です。 実行終了後、A口.16の全ビットはOFFされます。 詳細はSC2000取扱説明書の【4.1.4多軸運転】を参照して下さい。

movo

書式

movo <expr:ax>

説明

オーダー運転の起動を行います。 オーダー運転が終了するとプログラムは次のステップへ進みます。 命令の先駆けてステップ番号(A口.19)の指定を行います。 実行開始する為にA口.16.4をONにしますが,既に命令実行中の場合、実行終了を待ってか ら行います。 A口.16.4がONのままだと命令は無視されますので注意が必要です。 実行終了後、A口.16の全ビットはOFFされます。 詳細はSC2000取扱説明書の【4.1.4多軸運転】を参照して下さい。

movs,movsi

書式

movs <expr:ax> movsi <expr:ax>

説明

movsは絶対値、movsiは相対値で<expr:ax>で指定される1軸の単軸運転を行います。 単軸運転が終了するとプログラムは次のステップへ進みます。 命令の先駆けて位置(A口.17)、速度(A口.18)の指定を行います。 実行開始する為にmovs,movsi命令はA口.16.2をONにしますが,既に命令実行中の場合、実 行終了を待ってから行います。 A口.16.2がONのままだと命令は無視されますので注意が必要です。

movs,movsi

書式

movs <expr:ax> movsi <expr:ax>

説明

movsは絶対値、movsiは相対値で<expr:ax>で指定される1軸の単軸運転を行います。 単軸運転が終了するとプログラムは次のステップへ進みます。 命令の先駆けて位置(A口.17)、速度(A口.18)の指定を行います。 実行開始する為にmovs,movsi命令はA口.16.2をONにしますが,既に命令実行中の場合、実 行終了を待ってから行います。 A口.16.2がONのままだと命令は無視されますので注意が必要です。 実行終了後、A口.16の全ビットはOFFされます。 詳細はSC2000取扱説明書の【4.1.3単軸運転】を参照して下さい。

movt,movti

書式

movc <expr:ax>

説明

movtは絶対値、movtiは相対値単位時間運転の起動を行います。 単位時間運転が終了するとプログラムは次のステップへ進みます。 命令の先駆けて位置(A口.17)、時間指定(A口.20)の指定を行います。 実行開始する為にA口.16.5をONにしますが,既に命令実行中の場合、実行終了を待ってか ら行います。 A口.16.5がONのままだと命令は無視されますので注意が必要です。 実行終了後、A口.16の全ビットはOFFされます。 詳細はSC2000取扱説明書【4.1.7単位時間運転】を参照して下さい。

movt,movti

書式

movc <expr:ax>

説明

movtは絶対値、movtiは相対値単位時間運転の起動を行います。 単位時間運転が終了するとプログラムは次のステップへ進みます。 命令の先駆けて位置(A口.17)、時間指定(A口.20)の指定を行います。 実行開始する為にA口.16.5をONにしますが,既に命令実行中の場合、実行終了を待ってか ら行います。

open

書式

open "ファイルネーム",<text_mode> open <text_arg>,<text_mode>

説明

MMC(メモリーカード)に対してアクセスする為に開きます 最初の書式では、アクセスするファイル名を直接指定出来ます。 2番目の書式では、アクセスするファイル名をTレジスタで指定する事が出来ます。

<text_mode>には、"R","W","D"の何れかを指定 する必要が有ります。

"R"を指定した場合、読み込み用モードとしてファイルを開きます。

"W"を指定した場合、書き込み用モードとしてファイルを開きます。 (既存の同名ファイルがある場合、開く前に削除されます)

"D"を指定した場合、MMCのディレクトリー覧を取得するモードとして開きます。 この時、ファイルネームには、一覧を取得する「絶対パス」を指定する事が可能です。 (空文字列を指定した場合は、ルートディレクトリが対象になります)

変数RESに開いたファイル番号が返ります。

エラーの場合は、変数RESに0未満の値が返ります。

例

ファイル名("test.txt")を読み込みモードで開いて、T0~レジスタに格納する場合

sub readtest

Y0 = 0	;ループ用
open "test.txt","R"	;test.txtを読み込みモードで開きます
Y1 = RES	;Y1にファイル番号を格納します
loop:	;ループの開始位置
getl Y1,T(Y0)	;ファイルから1行読み込みます
if(RES < 0)	;読めなかったかどうか確認します
goto loopend	;読めなかったらループから抜けます(終了)
endif	;
Y0 = Y0 + 1	;次の格納場所に変数を設定します
goto loop	;ループの開始位置に戻る
loopend:	;ループの終了位置
close Y1	;開いたファイルを閉じます
return	

ファイル名("test2.txt")を書き込みモードで開いて、T0~レジスタからの値を保存する場合

sub writetest	
Y0 = 0	
open "test2.txt","W"	;test2.txtを書き込みモードで開きます
Y1 = RES	;Y1にファイル番号を格納します
loop:	
putl Y1,T(Y0)	;ファイルへ1行書き込みます
if(T(Y0).0 = 0)	;書き込んだデータが空かどうか確認します
goto loopend	;空なら終了と判断します
endif	
Y0 = Y0 + 1	;次の格納場所に変数を設定します
goto loop	
loopend:	
close Y1	;開いたファイルを閉じます
return	

ルートディレクトリの一覧をT0~レジスタに保存します

sub dirtest	
Y0 = 0	
open ‴,″D″	;ルートディレクトリを一覧モードで開きます
Y1 = RES	;Y1にファイル番号を格納します
loop:	
getl Y1,T(Y0)	;一覧から1行読み込みます
if(RES < 0)	;読めなかったかどうか確認します
goto loopend	;読めなかったら終了と判断します
endif	
Y0 = Y0 + 1	;次の格納場所に変数を設定します
goto loop	
loopend:	
close Y1	;開いたファイルを閉じます
return	

padj

書式

padj <expr:ax>

説明

padjは<expr>で指定される1軸の現在値調整を行います。 実行開始する為にpadj命令はA口.16.14をONにしますが,既に命令実行中の場合、実行終 了を待ってから行います。 A口.16.14がONのままだと命令は無視されますので注意が必要です。 実行終了後、A口.16の全ビットはOFFされます。

PI

書式

PI

説明

πの値を実数で返します。

PI2

書式

PI

説明

πの2乗を実数で返します。

pow()

書式

pow(<expr:x>,<expr:y>)

説明

<expr:x>の<expr:y>乗を計算します。結果は実数を返します。

prg

書式

prg <no>,<label>

説明

モーションプログラムの先頭に記述し、プログラム番号<no>とプログラム名<label>から構成されます。

<no>は0から255の範囲で指定します。 <label>はgosubから参照する場合に必要ですが、省略も可能です。 実行中にprg命令を見つけた場合、スタックが空でないなら呼び出したgosub命令の次に戻 り、スタックが空ならプログラムは終了します。

prgno

書式

prgno

説明

現在のプログラム番号(sub等の場合は分岐元プログラム番号)を戻します。

print

書式

print <Text_reg>,<Format>,<arg1>,....

説明

<arg>で指定されたレジスタ(又は数値)を指定の書式に変換した物を<Text_reg>で指定され るテキストレジスタに格納します。 <Format>と<arg>は、型および個数が一致しなくてはいけません。 もしもこれらが一致しないと、おかしな結果が得られることがあります。

prio

書式

prio <arg>

説明

<arg>で指定される行数(実際はこの値に+10された物)を元にモーションの一括動作行数を 指定します。 行数を上げると動作速度は上がりますが、他への影響が出る場合もあるので使用は慎重に 行って下さい。 (通常、使用する必要は有りません)

pset

書式

pset <expr:ax>

説明

psetは<expr>で指定される1軸の現在値設定を行います。 命令の先駆けて位置(A口.17)の指定を行います。 実行開始する為にpset命令はA口.16.12をONにしますが,既に命令実行中の場合、実行終 了を待ってから行います。 A口.16.12がONのままだと命令は無視されますので注意が必要です。 実行終了後、A口.16の全ビットはOFFされます。

putcom

書式

putcom <port>,<arg>

説明

<port>で指定された通信ポートへ<arg>で指定される1バイトのデータを送信します。 エラーが発生した場合、変数RESの値が0以外になります。

getl,putl

書式

getl <FileNo>,<Text_reg> putl <FileNo>,<Text_reg>

説明

getlはopen命令で開いたファイルから1行取得して、<Text_reg>で指定されるテキストレジス タに格納します。 putlはopen命令で開いたファイルに、<Text_reg>で指定されるテキストレジスタの内容を格納 します。 <FileNo>には、open命令を使用したときの返り値(変数RES)の値を指定します。

reset

書式

reset

説明

エラー状態から復帰する為に必要な処理を行い、エラーを解除します。

return

書式

return

説明

呼び出し元のgosub命令の次に戻ります。 return命令実行時にスタックが空の場合プログラムは終了します。 この時エラーにはなりません。

float(),int()

書式

```
float(<expr>)
int(<expr>)
```

説明

floatは<expr>を実数に型変換します。 intは<expr>は整数に型変換します。

scan

書式

scan <Text_reg>,<Format>,<arg1>,....

説明

<Text_reg>で指定されたテキストレジスタを指定の書式で変換した物を<arg>で指定されるレジスタに格納します。
<Format>と<arg>は、型および個数が一致しなくてはいけません。

もしもこれらが一致しないと、おかしな結果が得られることがあります。

seplh

書式

```
sepIh <val32>,<low16>,<high16>
```

説明

<val32>の値(32bitレジスタを指定)を16Bits x 2に分割して、<low>と<high>(両方とも16Bitレ ジスタを指定)に格納します。

例

```
F10 = $12345678
seplh(F10,E100,E101)
```

```
E100に$5678が入ります。
E101に$1234が入ります。
```

sin(),cos(),tan()

書式

```
sin(<expr>)
cos(<expr>)
tan(<expr>)
```

説明

```
<expr>のsin,cos,tanの計算を行います。角度の表現はradです。
結果は実数を返します。
```

sqrt()

書式

```
sqrt(<expr>)
```

説明

<expr>の平方根を計算します。結果は実数を返します。

stklh()

書式

stklh(<low16>,<high16>)

説明

<low>と<high>(両方とも16Bitレジスタを指定)を下位、上位として結合した値を32Bitで返します。

例

```
E100 = $5678
E101 = $1234
F0 = stklh(E100,E101)
```

F0に\$12345678が入ります。

stop

書式

stop

説明

プログラムを停止します

sub

書式

sub <label>

説明

サブルーチンの先頭に記述します。 <label>はgosub命令から参照されます。 サブルーチン中のreturn命令を実行すると、呼ばれたgosub命令の直後に復帰します。 実行中にsub命令を見つけた場合、スタックが空で無いなら呼び出したgosub命令の次に戻り、スタックが空ならプログラムは終了します。

sin(),cos(),tan()

書式

```
sin(<expr>)
cos(<expr>)
tan(<expr>)
```

説明

<expr>のsin,cos,tanの計算を行います。角度の表現はradです。 結果は実数を返します。

wait

書式

wait <expr>

説明

<expr>の値(1ms単位)プログラムを停止します。 実際の実行時間は4msの処理周期で判定される為、4msの整数倍となります。 またモーションの実行中のプログラム数が多い場合、実行時間が影響を受ける場合もあり ます。

リファレンス選択

ラダーで使用できる関数・コマンド一覧 アルファベット順一覧 機能別一覧

リファレンス一覧

ラダーで	で使用で	きる関数・	コマンドー	・覧(アルフ	ァベット順)
------	------	-------	-------	--------	--------

アルファベット順			
ADD	ADDI	ADDIr	ADDr
AND	ANDN	ANDS	CPL
DIV	DIVI	DIVIr	DIVr
DIX	EQ	EQI	FALL
GE	GEI	GT	GTI
IXR	LD	LDA	LDBCD

<u>LDBCDr</u>	LDN	LDS	LDSA
LE	LEI	LT	LTI
MC	MCR	MERGE	MERGEr
MOV	MOVB	MOVBL	MOVBLr
MOVBr	MOVBW	MOVBWr	MOVI
MOVIr	MOVr	MUL	MULI
MULIr	MULr	NE	NEG
<u>NEGr</u>	<u>NEI</u>	NOP	OR
ORN	ORS	<u>OUT</u>	OUTBCD
OUTBCDr	RAISE	RES	RESr
<u>SET</u>	<u>SETr</u>	<u>SFT</u>	<u>SFTr</u>
SIX	<u>SUB</u>	<u>SUBI</u>	SUBIr
<u>SUBr</u>	TMR	TMRI	TMRIr
TMRr	TMRRST	TMRRSTr	XOR
XORN	<u>XTRACT</u>	<u>XTRACTr</u>	

リファレンス一覧(機能別一覧)

ラダーで使用できる関数・コマンド一覧(機能別)

基本命令			
LD	LDN	LDA	OUT
AND	ANDN	ANDS	OR
ORN	ORS	LDSA	LDS
XOR	XORN	CPL	RAISE
FALL	MC	MCR	TMR
TMRr	TMRI	TMRIr	TMRRST
TMRRSTr	EQ	NE	GE
GT	<u>LE</u>	LT	EQI
NEI	GEI	GTI	LEI
LTI]		
応用命令			
<u>SET</u>	<u>SETr</u>	RES	<u>RESr</u>
MOV	MOVr	MOVI	MOVIr
ADD	ADDr	ADDI	ADDIr
SUB	<u>SUBr</u>	<u>SUBI</u>	SUBIr
MUL	MULr	MULI	MULIr
DIV	DIVr	DIVI	DIVIr

<u>NEG</u>	<u>NEGr</u>	<u>SFT</u>	<u>SFTr</u>
LDBCD	LDBCDr	OUTBCD	<u>OUTBCDr</u>
<u>XTRACT</u>	<u>XTRACTr</u>	MERGE	MERGEr
MOVB	MOVBr	MOVBW	MOVBWr
MOVBL	MOVBLr	SIX	DIX
IXR	NOP		

ADD, ADDr, ADDI, ADDIr

書式

ADD <reg:src>,<reg:dist> ADDr <reg:src>,<reg:dist> ADDI <imm>,<reg:dist> ADDIr <imm>,<reg:dist>

実行条件

ADD,ADDIはAccがONの時実行。 ADDr,ADDIrはAccがOFFからONになる時実行。

説明

ADD,ADDrは<reg:dist>と<reg:src>を加算し、<reg:dist>に格納します。<reg:src>の値は変化 しません。 ADDI,ADDIrは<reg:dist>と<imm>を加算し、<reg:dist>に格納します。

ADD, ADDr, ADDI, ADDIr

書式

ADD <reg:src>,<reg:dist> ADDr <reg:src>,<reg:dist> ADDI <imm>,<reg:dist> ADDIr <imm>,<reg:dist>

実行条件

ADD,ADDIはAccがONの時実行。

ADDr,ADDIrはAccがOFFからONになる時実行。

説明

ADD,ADDrは<reg:dist>と<reg:src>を加算し、<reg:dist>に格納します。<reg:src>の値は変化しません。

ADDI,ADDIrは<reg:dist>と<imm>を加算し、<reg:dist>に格納します。

ADD, ADDr, ADDI, ADDIr

書式

ADD <reg:src>,<reg:dist> ADDr <reg:src>,<reg:dist> ADDI <imm>,<reg:dist> ADDIr <imm>,<reg:dist>

実行条件

ADD,ADDIはAccがONの時実行。 ADDr,ADDIrはAccがOFFからONになる時実行。

説明

ADD,ADDrは<reg:dist>と<reg:src>を加算し、<reg:dist>に格納します。<reg:src>の値は変化しません。

ADDI,ADDIrは<reg:dist>と<imm>を加算し、<reg:dist>に格納します。

ADD, ADDr, ADDI, ADDIr

書式

ADD <reg:src>,<reg:dist> ADDr <reg:src>,<reg:dist> ADDI <imm>,<reg:dist> ADDIr <imm>,<reg:dist>

実行条件

ADD,ADDIはAccがONの時実行。 ADDr,ADDIrはAccがOFFからONになる時実行。

説明

ADD,ADDrは<reg:dist>と<reg:src>を加算し、<reg:dist>に格納します。<reg:src>の値は変化しません。

ADDI,ADDIrは<reg:dist>と<imm>を加算し、<reg:dist>に格納します。

LDA

書式

AND <bit> ANDN <bit>

実行条件

常時

説明

ANDはAccと<bit>の積をとり、Accに格納します。 ANDNはAccと<bit>の反転との積をとり、Accに格納します。

LDA

書式

AND <bit> ANDN <bit> 実行条件 常時

説明

ANDはAccと<bit>の積をとり、Accに格納します。 ANDNはAccと<bit>の反転との積をとり、Accに格納します。

ANDS

書式

ANDS

実行条件

常時

説明

Accの状態とスタックの状態の論理積を取りAccに格納します。 実行後スタックは1段ポップされます。

例

```
LD R5.2 (回路1)
OR R6.8
LD S16.2 (回路2)
OR R2.3
ANDS (結合)
```

CPL

書式

CPL

実行条件 常時

説明

Accの状態を反転させます。OFFならONに、ONならOFFになります。

DIV,DIVr,DIVI,DIVIr

書式

DIV <reg:src>,<reg:dist> DIVr <reg:src>,<reg:dist> DIVI <imm>,<reg:dist> DIVIr <imm>,<reg:dist>

実行条件

DIV,DIVIはAccがONの時実行。 DIVr,DIVIrはAccがOFFからONになる時実行。

説明

DIV,DIVrは<reg:dist>を<reg:src>で除算し、<reg:dist>に格納します。<reg:src>の値は変化し ません。 DIVI,DIVIrは<reg:dist>を<imm>で除算し、<reg:dist>に格納します。 除算の余りはS6レジスタに格納されます。

DIV,DIVr,DIVI,DIVIr

書式

DIV <reg:src>,<reg:dist> DIVr <reg:src>,<reg:dist> DIVI <imm>,<reg:dist> DIVIr <imm>,<reg:dist>

実行条件

DIV,DIVIはAccがONの時実行。 DIVr,DIVIrはAccがOFFからONになる時実行。

説明

DIV,DIVrは<reg:dist>を<reg:src>で除算し、<reg:dist>に格納します。<reg:src>の値は変化し ません。 DIVI,DIVIrは<reg:dist>を<imm>で除算し、<reg:dist>に格納します。 除算の余りはS6レジスタに格納されます。

除身の余りはS6レシスタに格納されます。

DIV,DIVr,DIVI,DIVIr

書式

DIV <reg:src>,<reg:dist> DIVr <reg:src>,<reg:dist> DIVI <imm>,<reg:dist> DIVIr <imm>,<reg:dist>

実行条件

DIV,DIVIはAccがONの時実行。 DIVr,DIVIrはAccがOFFからONになる時実行。

説明

DIV,DIVrは<reg:dist>を<reg:src>で除算し、<reg:dist>に格納します。<reg:src>の値は変化しません。 DIVI DIVIrは<reg:dist>を<imm>で除算し、<reg:dist>に格納します

DIVI,DIVIrは<reg:dist>を<imm>で除算し、<reg:dist>に格納します。 除算の余りはS6レジスタに格納されます。

DIV,DIVr,DIVI,DIVIr

書式

DIV <reg:src>,<reg:dist> DIVr <reg:src>,<reg:dist> DIVI <imm>,<reg:dist> DIVIr <imm>,<reg:dist>

実行条件

DIV,DIVIはAccがONの時実行。 DIVr,DIVIrはAccがOFFからONになる時実行。

説明

DIV,DIVrは<reg:dist>を<reg:src>で除算し、<reg:dist>に格納します。<reg:src>の値は変化しません。 DIVI,DIVIrは<reg:dist>を<imm>で除算し、<reg:dist>に格納します。

DIVI,DIVIrla<reg:dist>を<imm>で除昇し、<reg:dist>l=格納します。 除算の余りはS6レジスタに格納されます。

SIX,DIX

書式

SIX <reg> DIX <reg>

実行条件

常時

説明

SIXは<reg>の値をソースインデックスに設定します。 DIXは<reg>の値をディスティネーションインデックスに設定します。

EQ,NE,GE,GT,LE,LT,EQI,NEI,GEI,GTI,LEI,LTI

書式

EQ <reg1>,<reg2> NE <reg1>,<reg2> GE <reg1>,<reg2> GT <reg1>,<reg2> LE <reg1>,<reg2> LT <reg1>,<reg2> EQI <imm>,<reg2> NEI <imm>,<reg2> GEI <imm>,<reg2> GTI <imm>,<reg2> LEI <imm>,<reg2> LEI <imm>,<reg2> LEI <imm>,<reg2> LEI <imm>,<reg2> LEI <imm>,<reg2> 実行条件

常時

説明

比較を行い結果をAccに格納します。 実行時現在のAccの状態をスタックに格納してから行います。 比較前のAccと演算する場合はANDS,ORS命令を使用します。 比較の結果AccがONする条件は以下の通りです。

命令	結果が真になる条件
EQ FQI	$\langle \operatorname{reg1} \rangle = \langle \operatorname{reg2} \rangle$ $\langle \operatorname{imm} \rangle = \langle \operatorname{reg2} \rangle$
NE NEI	<pre></pre>
GE GEI	$\langle \operatorname{reg1} \rangle \langle = \langle \operatorname{reg2} \rangle$ $\langle \operatorname{imm} \rangle \langle = \langle \operatorname{reg2} \rangle$
GT GTI	<reg1> < <reg2> <imm> < <reg2></reg2></imm></reg2></reg1>
LE LEI	<reg1> >= <reg2> <imm> >= <reg2></reg2></imm></reg2></reg1>
LT LTI	$\begin{array}{c} \langle \mathrm{reg1} \rangle > \langle \mathrm{reg2} \rangle \\ \langle \mathrm{imm} \rangle > \langle \mathrm{reg2} \rangle \end{array}$

例

LD R5.2 EQ R10,F25

(動作説明) ANDS R5.2がONかつ、R10とF25が等しい時AccがON

EQ,NE,GE,GT,LE,LT,EQI,NEI,GEI,GTI,LEI,LTI

書式

EQ <reg1>,<reg2> NE <reg1>,<reg2> GE <reg1>,<reg2> GT <reg1>,<reg2> LE <reg1>,<reg2> LT <reg1>,<reg2> EQI <imm>,<reg2> NEI <imm>,<reg2> GEI <imm>,<reg2> GTI <imm>,<reg2> LEI <imm>,<reg2> 比較を行い結果をAccに格納します。 実行時現在のAccの状態をスタックに格納してから行います。 比較前のAccと演算する場合はANDS,ORS命令を使用します。 比較の結果AccがONする条件は以下の通りです。

命令	結果が真になる条件
EQ	$\langle \operatorname{reg1} \rangle = \langle \operatorname{reg2} \rangle$
FQI	$\langle \operatorname{imm} \rangle = \langle \operatorname{reg2} \rangle$
NE NEI	<pre></pre>
GE	<reg1> <= <reg2></reg2></reg1>
GEI	<imm> <= <reg2></reg2></imm>
GT	<reg1> < <reg2></reg2></reg1>
GTI	<imm> < <reg2></reg2></imm>
LE LEI	
LT	<reg1> > <reg2></reg2></reg1>
LTI	<imm> > <reg2></reg2></imm>

例

LD R5.2

EQ R10,F25

(動作説明) ANDS R5.2がONかつ、R10とF25が等しい時AccがON

FALL

書式

FALL

実行条件

常時

説明

AccのONからOFFになった1周期だけAccをONにします。 立ち下がりエッジを検出します。

EQ,NE,GE,GT,LE,LT,EQI,NEI,GEI,GTI,LEI,LTI

書式

EQ <reg1>,<reg2> NE <reg1>,<reg2> GE <reg1>,<reg2> GT <reg1>,<reg2> LE <reg1>,<reg2> LT <reg1>,<reg2> EQI <imm>,<reg2> NEI <imm>,<reg2> GEI <imm>,<reg2> GTI <imm>,<reg2> LEI <imm>,<reg2> LTI <imm>,<reg2>

実行条件

常時

説明

比較を行い結果をAccに格納します。 実行時現在のAccの状態をスタックに格納してから行います。 比較前のAccと演算する場合はANDS,ORS命令を使用します。 比較の結果AccがONする条件は以下の通りです。

命令	結果が真になる条件
EQ	$\langle reg1 \rangle = \langle reg2 \rangle$
EQI	<imm> = <reg2></reg2></imm>
NE	<reg1> <> <reg2></reg2></reg1>
NEI	$\langle imm \rangle < > \langle reg2 \rangle$
GE	$\langle reg1 \rangle <= \langle reg2 \rangle$
GEI	$\langle imm \rangle \langle reg2 \rangle$
GT	<reg1> < <reg2></reg2></reg1>
GTI	<imm> < <reg2></reg2></imm>
LE	<reg1> >= <reg2></reg2></reg1>
LEI	$\langle imm \rangle > = \langle reg2 \rangle$
LT	<reg1> > <reg2></reg2></reg1>
LTI	<imm> > <reg2></reg2></imm>

例

LD R5.2 EQ R10,F25

(動作説明) ANDS R5.2がONかつ、R10とF25が等しい時AccがON

EQ,NE,GE,GT,LE,LT,EQI,NEI,GEI,GTI,LEI,LTI

書式

EQ <reg1>,<reg2> NE <reg1>,<reg2> GE <reg1>,<reg2> GT <reg1>,<reg2> LE <reg1>,<reg2> LT <reg1>,<reg2> EQI <imm>,<reg2> NEI <imm>,<reg2> GEI <imm>,<reg2> GEI <imm>,<reg2> GTI <imm>,<reg2> LEI <imm>,<reg2>

LTI <imm>,<reg2>

実行条件

常時

説明

比較を行い結果をAccに格納します。 実行時現在のAccの状態をスタックに格納してから行います。 比較前のAccと演算する場合はANDS,ORS命令を使用します。 比較の結果AccがONする条件は以下の通りです。

命令	結果が真になる条件
EQ	$\langle reg1 \rangle = \langle reg2 \rangle$
EQI	$\langle imm \rangle = \langle reg2 \rangle$
NE	<reg1> <> <reg2></reg2></reg1>
NEI	$\langle imm \rangle \langle reg2 \rangle$
GE	<reg1> <= <reg2></reg2></reg1>
GEI	$\langle imm \rangle \langle reg2 \rangle$
GT	<reg1> < <reg2></reg2></reg1>
GTI	<imm> < <reg2></reg2></imm>
LE	$\langle reg1 \rangle > = \langle reg2 \rangle$
LEI	$\langle imm \rangle > = \langle reg2 \rangle$
LT	<reg1> > <reg2></reg2></reg1>
LTI	<imm> > <reg2></reg2></imm>

例

LD R5.2 EQ R10,F25

(動作説明) ANDS R5.2がONかつ、R10とF25が等しい時AccがON

EQ,NE,GE,GT,LE,LT,EQI,NEI,GEI,GTI,LEI,LTI

書式

EQ <reg1>,<reg2> NE <reg1>,<reg2> GE <reg1>,<reg2> GT <reg1>,<reg2> LE <reg1>,<reg2> LT <reg1>,<reg2> EQI <imm>,<reg2> NEI <imm>,<reg2> GEI <imm>,<reg2> LEI <imm>,<reg2>

説明

比較を行い結果をAccに格納します。 実行時現在のAccの状態をスタックに格納してから行います。 比較前のAccと演算する場合はANDS,ORS命令を使用します。 比較の結果AccがONする条件は以下の通りです。

命令	結果が真になる条件
EQ	$\langle \operatorname{reg1} \rangle = \langle \operatorname{reg2} \rangle$
EQI	$\langle \operatorname{imm} \rangle = \langle \operatorname{reg2} \rangle$
NE NEI	<pre></pre>
GE	$\langle \operatorname{reg1} \rangle \langle = \langle \operatorname{reg2} \rangle$
GEI	$\langle \operatorname{imm} \rangle \langle = \langle \operatorname{reg2} \rangle$
GT	<reg1> < <reg2></reg2></reg1>
GTI	<imm> < <reg2></reg2></imm>
LE LEI	
LT	<reg1> > <reg2></reg2></reg1>
LTI	<imm> > <reg2></reg2></imm>

例

LD R5.2

EQ R10,F25

(動作説明) ANDS R5.2がONかつ、R10とF25が等しい時AccがON

EQ,NE,GE,GT,LE,LT,EQI,NEI,GEI,GTI,LEI,LTI

書式	
	EQ <reg1>,<reg2></reg2></reg1>
	NE <reg1>,<reg2></reg2></reg1>
	GE <reg1>,<reg2></reg2></reg1>
	GT <reg1>,<reg2></reg2></reg1>
	LE <reg1>,<reg2></reg2></reg1>
	LT <reg1>,<reg2></reg2></reg1>
	EQI <imm>,<reg2></reg2></imm>
	NEI <imm>,<reg2></reg2></imm>
	GEI <imm>,<reg2></reg2></imm>
	GTI <imm>,<reg2></reg2></imm>
	LEI <imm>,<reg2></reg2></imm>
	LTI <imm>,<reg2></reg2></imm>
実行	条件
	常時
説明	
	比較を行い結果をAcc

比較を行い結果をAccに格納します。 実行時現在のAccの状態をスタックに格納してから行います。 比較前のAccと演算する場合はANDS,ORS命令を使用します。 比較の結果AccがONする条件は以下の通りです。

命令	結果が真になる条件
EQ	$\langle reg1 \rangle = \langle reg2 \rangle$
EQI	$\langle imm \rangle = \langle reg2 \rangle$
NE	<reg1> <> <reg2></reg2></reg1>
NEI	$\langle imm \rangle \langle reg2 \rangle$
GE	<reg1> <= <reg2></reg2></reg1>
GEI	$\langle imm \rangle \langle reg2 \rangle$
GT	<reg1> < <reg2></reg2></reg1>
GTI	<imm> < <reg2></reg2></imm>
LE	$\langle reg1 \rangle > = \langle reg2 \rangle$
LEI	$\langle imm \rangle > = \langle reg2 \rangle$
LT	<reg1> > <reg2></reg2></reg1>
LTI	<imm> > <reg2></reg2></imm>

例

LD R5.2 EQ R10,F25

(動作説明) ANDS R5.2がONかつ、R10とF25が等しい時AccがON

IXR

書式

IXR

実行条件 常時

説明

ソースインデックス及びディスティネーションインデックスの値を0クリアします。

LD,LDN

書式

LD <bit> LDN <bit> 実行条件 常時 説明 LDは<bit>の状態をAccにロードします。 LDNは<bit>の反転状態をAccにロードします。 直前のAccの状態はスタックにプッシュされ後で使用することが出来ます。
LDA

書式

LDA

実行条件 常時

説明

Accの状態をスタックにプッシュします。

この命令は現在のAccの状態を別の回路で使用するためにスタックに保持する為に使用します。

LDBCD,LDBCDr

書式

LDBCD <reg:src>,<reg:dist> LDBCDr <reg:src>,<reg:dist>

実行条件

LDBCDはAccがONの時実行。 LDBCDrはAccがOFFからONになる時実行。

説明

```
<reg:src>のBCDコードをバイナリーに変換して<reg:dist>に格納します。</reg:src>の値は変化しません。
```

LDBCD,LDBCDr

書式

LDBCD <reg:src>,<reg:dist> LDBCDr <reg:src>,<reg:dist>

実行条件

LDBCDはAccがONの時実行。 LDBCDrはAccがOFFからONになる時実行。

説明

```
<reg:src>のBCDコードをバイナリーに変換して<reg:dist>に格納します。</reg:src>の値は変化しません。
```

LD,LDN

書式

```
LD <bit>
LDN <bit>
実行条件
常時
説明
```

LDは<bit>の状態をAccにロードします。 LDNは<bit>の反転状態をAccにロードします。 直前のAccの状態はスタックにプッシュされ後で使用することが出来ます。

LDS

書式

LDS

実行条件

常時

説明

スタックの先頭をAccにロードします。 実行後スタックは1段ポップされます。

LDSA

書式

LDSA 実行条件 常時

説明

スタックの先頭をAcclこロードします。 実行後スタックはポップされず、以前のままの状態です。

EQ,NE,GE,GT,LE,LT,EQI,NEI,GEI,GTI,LEI,LTI

書式

EQ <reg1>,<reg2> NE <reg1>,<reg2> GE <reg1>,<reg2> GT <reg1>,<reg2> LE <reg1>,<reg2> LT <reg1>,<reg2> EQI <imm>,<reg2> NEI <imm>,<reg2> GEI <imm>,<reg2> GTI <imm>,<reg2> LEI <imm>,<reg2> LTI <imm>,<reg2>

実行条件

常時

説明

比較を行い結果をAccに格納します。 実行時現在のAccの状態をスタックに格納してから行います。 比較前のAccと演算する場合はANDS,ORS命令を使用します。 比較の結果AccがONする条件は以下の通りです。

命令	結果が真になる条件
EQ	$\langle \operatorname{reg1} \rangle = \langle \operatorname{reg2} \rangle$
EQI	$\langle \operatorname{imm} \rangle = \langle \operatorname{reg2} \rangle$
NE NEI	<pre></pre>
GE	<reg1> <= <reg2></reg2></reg1>
GEI	<imm> <= <reg2></reg2></imm>
GT	<reg1> < <reg2></reg2></reg1>
GTI	<imm> < <reg2></reg2></imm>
LE LEI	
LT	<reg1> > <reg2></reg2></reg1>
LTI	<imm> > <reg2></reg2></imm>

例

LD R5.2 EQ R10,F25

(動作説明) ANDS R5.2がONかつ、R10とF25が等しい時AccがON

EQ,NE,GE,GT,LE,LT,EQI,NEI,GEI,GTI,LEI,LTI

書式

EQ <reg1>,<reg2> NE <reg1>,<reg2> GE <reg1>,<reg2> GT <reg1>,<reg2> LE <reg1>,<reg2> LT <reg1>,<reg2> EQI <imm>,<reg2> NEI <imm>,<reg2> GEI <imm>,<reg2> GTI <imm>,<reg2> LEI <imm>,<reg2> LEI <imm>,<reg2> LEI <imm>,<reg2> LEI <imm>,<reg2> LEI <imm>,<reg2> LEI <imm>,<reg2> 実行条件

常時

説明

比較を行い結果をAccに格納します。 実行時現在のAccの状態をスタックに格納してから行います。 比較前のAccと演算する場合はANDS,ORS命令を使用します。 比較の結果AccがONする条件は以下の通りです。

命令	結果が真になる条件
EQ FQI	$\langle \operatorname{reg1} \rangle = \langle \operatorname{reg2} \rangle$ $\langle \operatorname{imm} \rangle = \langle \operatorname{reg2} \rangle$
NE NEI	<pre></pre>
GE GEI	$\langle \operatorname{reg1} \rangle \langle = \langle \operatorname{reg2} \rangle$ $\langle \operatorname{imm} \rangle \langle = \langle \operatorname{reg2} \rangle$
GT GTI	<reg1> < <reg2> <imm> < <reg2></reg2></imm></reg2></reg1>
LE LEI	<reg1> >= <reg2> <imm> >= <reg2></reg2></imm></reg2></reg1>
LT LTI	$\begin{array}{c} \langle \mathrm{reg1} \rangle > \langle \mathrm{reg2} \rangle \\ \langle \mathrm{imm} \rangle > \langle \mathrm{reg2} \rangle \end{array}$

例

LD R5.2 EQ R10,F25

(動作説明) ANDS R5.2がONかつ、R10とF25が等しい時AccがON

EQ,NE,GE,GT,LE,LT,EQI,NEI,GEI,GTI,LEI,LTI

書式

EQ <reg1>,<reg2> NE <reg1>,<reg2> GE <reg1>,<reg2> GT <reg1>,<reg2> LE <reg1>,<reg2> LT <reg1>,<reg2> EQI <imm>,<reg2> NEI <imm>,<reg2> GEI <imm>,<reg2> GTI <imm>,<reg2> LEI <imm>,<reg2> 比較を行い結果をAccに格納します。 実行時現在のAccの状態をスタックに格納してから行います。 比較前のAccと演算する場合はANDS,ORS命令を使用します。 比較の結果AccがONする条件は以下の通りです。

命令	結果が真になる条件
EQ	$\langle \text{reg1} \rangle = \langle \text{reg2} \rangle$
NE	<reg1> <> <reg2></reg2></reg1>
NEI	$\langle imm \rangle \langle reg2 \rangle$
GE	<reg1> <= <reg2></reg2></reg1>
GEI	$\langle imm \rangle \langle reg2 \rangle$
GT	<reg1> < <reg2></reg2></reg1>
GTI	<imm> < <reg2></reg2></imm>
LE	<reg1> >= <reg2></reg2></reg1>
LEI	$\langle imm \rangle > = \langle reg2 \rangle$
LT	<reg1> > <reg2></reg2></reg1>
LTI	<imm> > <reg2></reg2></imm>

例

LD R5.2 EQ R10,F25

(動作説明) ANDS R5.2がONかつ、R10とF25が等しい時AccがON

EQ,NE,GE,GT,LE,LT,EQI,NEI,GEI,GTI,LEI,LTI

+	÷
書	エ

EQ <reg1>,<reg2> NE <reg1>,<reg2> GE <reg1>,<reg2> GT <reg1>,<reg2> LE <reg1>,<reg2> LT <reg1>,<reg2> EQI <imm>,<reg2> NEI <imm>,<reg2> GEI <imm>,<reg2> GEI <imm>,<reg2> LEI <imm>,<reg2> LEI <imm>,<reg2> LEI <imm>,<reg2> LEI <imm>,<reg2> LEI <imm>,<reg2> LEI <imm>,<reg2>

常時

説明

比較を行い結果をAccに格納します。 実行時現在のAccの状態をスタックに格納してから行います。 比較前のAccと演算する場合はANDS,ORS命令を使用します。 比較の結果AccがONする条件は以下の通りです。

命令	結果が真になる条件	
EQ	$\langle \operatorname{reg1} \rangle = \langle \operatorname{reg2} \rangle$	
EQI	<imm> = <reg2></reg2></imm>	
NE	<reg1> <> <reg2></reg2></reg1>	
NEI	<imm> <> <reg2></reg2></imm>	
GE	<reg1> <= <reg2></reg2></reg1>	
GEI	$\langle imm \rangle \langle reg2 \rangle$	
GT	<reg1> < <reg2></reg2></reg1>	
GTI	<imm> < <reg2></reg2></imm>	
LE	<reg1> >= <reg2></reg2></reg1>	
LEI	$\langle imm \rangle > = \langle reg2 \rangle$	
LT	<reg1> > <reg2></reg2></reg1>	
LTI	<imm> > <reg2></reg2></imm>	

例

LD R5.2 EQ R10,F25

(動作説明) ANDS R5.2がONかつ、R10とF25が等しい時AccがON

MC,MCR

書式

MC

MCR 実行条件

コネロ常時

説明

MCとMCR命令に囲まれたプログラムの実行を制御することが出来ます。 制御する命令は LD,LDN,EQ,EQI,NE,NEI,GE,GEI,LE,LEI でこれらの命令実行後のAccをOFF とします。 MCとMCR命令のペアは入れ子で使用することが可能です。 つまりMCとMCR命令間のプログラムにさらにMCとMCR命令を記述する事が出来、複雑な 回路を整理することが可能です。 入れ子は31回まで可能です。 MCR命令はAccの状態にかかわらず常に実行を行います。

MC,MCR

書式

MC MCR 実行条件 常時

説明

MCとMCR命令に囲まれたプログラムの実行を制御することが出来ます。 制御する命令は LD,LDN,EQ,EQI,NE,NEI,GE,GEI,LE,LEI でこれらの命令実行後のAccをOFF とします。 MCとMCR命令のペアは入れ子で使用することが可能です。 つまりMCとMCR命令間のプログラムにさらにMCとMCR命令を記述する事が出来、複雑な 回路を整理することが可能です。 入れ子は31回まで可能です。 MCR命令はAccの状態にかかわらず常に実行を行います。

MERGE, MERGEr

書式

MERGE <reg:src>,<imm:nbit>,<reg:dist>,<imm:bpos> MERGEr <reg:src>,<imm:nbit>,<reg:dist>,<imm:bpos>

実行条件

MERGEはAccがONの時実行。 MERGErはAccがOFFからONになる時実行。

説明

<reg:src>の0ビット目から<imm:nbit>の示すビット数だけ<reg:dist>の<imm:bpos>の示すビット 番号にコピーします。 <reg:src>は変化しません。 <imm:nbit>から<reg:dist>ビットだけが変化します。

例

MERGE F0,4,F1,6

転送元		転送先
F1.0 - F1.5	\rightarrow	F1.0 - F1.5
F0.0	\rightarrow	F1.6
F0.1	\rightarrow	F1.7
F0.2	\rightarrow	F1.8
F0.3	\rightarrow	F1.9
F1.10 - F1.31	\rightarrow	F1.10 - F1.31

MERGE, MERGEr

書式

MERGE <reg:src>,<imm:nbit>,<reg:dist>,<imm:bpos> MERGEr <reg:src>,<imm:nbit>,<reg:dist>,<imm:bpos>

実行条件

MERGEはAccがONの時実行。

MERGErはAccがOFFからONになる時実行。

説明

<reg:src>の0ビット目から<imm:nbit>の示すビット数だけ<reg:dist>の<imm:bpos>の示すビット 番号にコピーします。 <reg:src>は変化しません。 <imm:nbit>から<reg:dist>ビットだけが変化します。

例

MERGE F0,4,F1,6

転送元		転送先
F1.0 - F1.5	\rightarrow	F1.0 - F1.5
F0.0	\rightarrow	F1.6
F0.1	\rightarrow	F1.7
F0.2	\rightarrow	F1.8
F0.3	\rightarrow	F1.9
F1.10 - F1.31	\rightarrow	F1.10 - F1.31

MOV,MOVr,MOVI,MOVIr

書式

MOV <reg:src>,<reg:dist> MOVr <reg:src>,<reg:dist> MOVI <imm>,<reg:dist> MOVIr <imm>,<reg:dist>

実行条件

MOV,MOVIはAccがONの時実行。 MOVr,MOVIrはAccがOFFからONになる時実行。

説明

MOV,MOVrは<reg:src>を<reg:dist>にコピーします。 MOVI,MOVIrは<imm>を<reg:dist>にコピーします。

MOVB,MOVBr

書式

MOVB <reg:src>,<reg:dist>,<imm:n> MOVBr <reg:src>,<reg:dist>,<imm:n>

実行条件

MOVBはAccがONの時実行。 MOVBrはAccがOFFからONになる時実行。

説明

<reg:src>から<reg:dist>へ<imm:n>の示すワード数だけコピーします。

ビット長の異なるレジスタを指定した場合、下位ビットのみコピーされます。 <reg:src>の値は変化しません。 <imm:n>の範囲は1から63です。 SIX,DIX命令によるインデックス修飾が行えます。

例

MOVB F0,F10,4

転送元		転送先
F0.0 - F0.31	\rightarrow	F10.0 - F10.31
F1.0 - F1.31	\rightarrow	F11.0 - F11.31
F2.0 - F2.31	\rightarrow	F12.0 - F12.31
F3.0 - F3.31	\rightarrow	F13.0 - F13.31

MOVBL, MOVBLr

書式

MOVBL <reg:src32>,<reg:dist16>,<imm:n> MOVBLr <reg:src32>,<reg:dist16>,<imm:n>

実行条件

MOVBLはAccがONの時実行。 MOVBLrはAccがOFFからONになる時実行。

説明

<reg:src32>から<reg:dist16>へ<imm:n>の示すワード数だけコピーします。 この時32ビットレジスタ1個を16ビットレジスタ2個にコピーします。 コピーするワード数は32ビットレジスタの数で指定します。 <reg:src16>の値は変化しません。 <imm:n>の範囲は1から63です。 SIX,DIX命令によるインデックス修飾が行えます。

例

MOVBL F0,E10,3

転送元		転送先
F10.0 - F10.15	\rightarrow	E0.0 - E0.15
F10.16 - F10.31	\rightarrow	E1.0 - E1.15
F11.0 - F11.15	\rightarrow	E2.0 - E2.15
F11.16 - F11.31	\rightarrow	E3.0 - E3.15
F12.0 - F12.15	\rightarrow	E4.0 - E4.15
F12.16 - F12.31	\rightarrow	E5.0 - E5.15

MOVB,MOVBr

書式

```
MOVB <reg:src>,<reg:dist>,<imm:n>
MOVBr <reg:src>,<reg:dist>,<imm:n>
```

実行条件

MOVBはAccがONの時実行。 MOVBrはAccがOFFからONになる時実行。

説明

<reg:src>から<reg:dist>へ<imm:n>の示すワード数だけコピーします。 ビット長の異なるレジスタを指定した場合、下位ビットのみコピーされます。 <reg:src>の値は変化しません。 <imm:n>の範囲は1から63です。 SIX,DIX命令によるインデックス修飾が行えます。

例

MOVB F0,F10,4

転送元		転送先
F0.0 - F0.31	\rightarrow	F10.0 - F10.31
F1.0 - F1.31	\rightarrow	F11.0 - F11.31
F2.0 – F2.31	\rightarrow	F12.0 - F12.31
F3.0 - F3.31	\rightarrow	F13.0 - F13.31

MOVBW,MOVBWr

書式

MOVBW <reg:src16>,<reg:dist32>,<imm:n> MOVBWr <reg:src16>,<reg:dist32>,<imm:n>

実行条件

MOVBWはAccがONの時実行。 MOVBWrはAccがOFFからONになる時実行。

説明

<reg:src16>から<reg:dist32>へ<imm:n>の示すワード数だけコピーします。 この時16ビットレジスタ2個を32ビットレジスタ1個にコピーします。 コピーするワード数は32ビットレジスタの数で指定します。 <reg:src16>の値は変化しません。<imm:n>の範囲は1から63です。 SIX,DIX命令によるインデックス修飾が行えます。

例

MOVBW E0,F10,3

転送元		転送先
E0.0 - E0.15	\rightarrow	F10.0 - F10.15
E1.0 - E1.15	\rightarrow	F10.16 - F16.31
E2.0 – E2.0	\rightarrow	F11.0 - F11.15
E3.0 - E3.15	\rightarrow	F11.16 - F11.31

E4.0 – E4.15	\rightarrow	F12.0 - F12.15
E5.0 - E5.15	\rightarrow	F12.16 - F12.31

MOVBW,MOVBWr

書式

MOVBW <reg:src16>,<reg:dist32>,<imm:n> MOVBWr <reg:src16>,<reg:dist32>,<imm:n>

実行条件

MOVBWはAccがONの時実行。 MOVBWrはAccがOFFからONになる時実行。

説明

<reg:src16>から<reg:dist32>へ<imm:n>の示すワード数だけコピーします。 この時16ビットレジスタ2個を32ビットレジスタ1個にコピーします。 コピーするワード数は32ビットレジスタの数で指定します。 <reg:src16>の値は変化しません。<imm:n>の範囲は1から63です。 SIX,DIX命令によるインデックス修飾が行えます。

例

MOVBW E0,F10,3

転送元		転送先
E0.0 - E0.15	\rightarrow	F10.0 - F10.15
E1.0 - E1.15	\rightarrow	F10.16 - F16.31
E2.0 - E2.0	\rightarrow	F11.0 - F11.15
E3.0 - E3.15	\rightarrow	F11.16 - F11.31
E4.0 - E4.15	\rightarrow	F12.0 - F12.15
E5.0 - E5.15	\rightarrow	F12.16 - F12.31

MOV,MOVr,MOVI,MOVIr

書式

MOV <reg:src>,<reg:dist> MOVr <reg:src>,<reg:dist> MOVI <imm>,<reg:dist> MOVIr <imm>,<reg:dist>

実行条件

MOV,MOVIはAccがONの時実行。 MOVr,MOVIrはAccがOFFからONになる時実行。

説明

MOV,MOVrは<reg:src>を<reg:dist>にコピーします。 MOVI,MOVIrは<imm>を<reg:dist>にコピーします。

MOV,MOVr,MOVI,MOVIr

書式

MOV <reg:src>,<reg:dist> MOVr <reg:src>,<reg:dist> MOVI <imm>,<reg:dist> MOVIr <imm>,<reg:dist>

実行条件

MOV,MOVIはAccがONの時実行。 MOVr,MOVIrはAccがOFFからONになる時実行。

説明

MOV,MOVrは<reg:src>を<reg:dist>にコピーします。 MOVI,MOVIrは<imm>を<reg:dist>にコピーします。

MOV,MOVr,MOVI,MOVIr

書式

MOV <reg:src>,<reg:dist> MOVr <reg:src>,<reg:dist> MOVI <imm>,<reg:dist> MOVIr <imm>,<reg:dist>

実行条件

MOV,MOVIはAccがONの時実行。 MOVr,MOVIrはAccがOFFからONになる時実行。

説明

MOV,MOVrは<reg:src>を<reg:dist>にコピーします。 MOVI,MOVIrは<imm>を<reg:dist>にコピーします。

MUL, MULr, MULI, MULIr

書式

MUL <reg:src>,<reg:dist> MULr <reg:src>,<reg:dist> MULI <imm>,<reg:dist> MULIr <imm>,<reg:dist> g行条件 MUL,MULIはAccがONの時実行。 MULr,MULIrはAccがOFFからONになる時実行。

説明

MUL,MULrは<reg:dist>と<reg:src>を乗算し、<reg:dist>に格納します。<reg:src>の値は変化

EQ,NE,GE,GT,LE,LT,EQI,NEI,GEI,GTI,LEI,LTI

書式

EQ <reg1>,<reg2> NE <reg1>,<reg2> GE <reg1>,<reg2> GT <reg1>,<reg2> LE <reg1>,<reg2> LT <reg1>,<reg2> EQI <imm>,<reg2> REI <imm>,<reg2> GEI <imm>,<reg2> GEI <imm>,<reg2> LTI <imm>,<reg2> GTI <imm>,<reg2> LTI <imm>,<reg2> LTI <imm>,<reg2> LTI <imm>,<reg2>

常時

説明

比較を行い結果をAccに格納します。 実行時現在のAccの状態をスタックに格納してから行います。 比較前のAccと演算する場合はANDS,ORS命令を使用します。 比較の結果AccがONする条件は以下の通りです。

命令	結果が真になる条件
EQ	$\langle reg1 \rangle = \langle reg2 \rangle$
EQI	$\langle imm \rangle = \langle reg2 \rangle$
NE	<reg1> <> <reg2></reg2></reg1>
NEI	$\langle imm \rangle \langle reg2 \rangle$
GE	<reg1> <= <reg2></reg2></reg1>
GEI	$\langle imm \rangle \langle reg2 \rangle$
GT	<reg1> < <reg2></reg2></reg1>
GTI	<imm> < <reg2></reg2></imm>
LE	<reg1> >= <reg2></reg2></reg1>
LEI	$\langle imm \rangle > = \langle reg2 \rangle$
LT	<reg1> > <reg2></reg2></reg1>
LTI	<imm> > <reg2></reg2></imm>

例

LD R5.2 EQ R10,F25

(動作説明) ANDS R5.2がONかつ、R10とF25が等しい時AccがON

NEG,NEGr

書式

NEG <reg:src>,<reg:dist> NEGr <reg:src>,<reg:dist>

実行条件

NEGはAccがONの時実行。 NEGrはAccがOFFからONになる時実行。

説明

<reg:src>を符号反転して<reg:dist>に格納します。 <reg:src>の値は変化しません。

NEG, NEGr

書式

NEG <reg:src>,<reg:dist> NEGr <reg:src>,<reg:dist>

実行条件

NEGはAccがONの時実行。 NEGrはAccがOFFからONになる時実行。

説明

<reg:src>を符号反転して<reg:dist>に格納します。 <reg:src>の値は変化しません。

EQ,NE,GE,GT,LE,LT,EQI,NEI,GEI,GTI,LEI,LTI

書式

EQ <reg1>,<reg2> NE <reg1>,<reg2> GE <reg1>,<reg2> GT <reg1>,<reg2> LE <reg1>,<reg2> LT <reg1>,<reg2> EQI <imm>,<reg2> REI <imm>,<reg2> GEI <imm>,<reg2> GTI <imm>,<reg2> LEI <imm>,<reg2> BTI <imm>,<reg2> LEI <imm>,<reg2> 比較を行い結果をAccに格納します。 実行時現在のAccの状態をスタックに格納してから行います。 比較前のAccと演算する場合はANDS,ORS命令を使用します。 比較の結果AccがONする条件は以下の通りです。

命令	結果が真になる条件		
EQ	$\langle \operatorname{reg1} \rangle = \langle \operatorname{reg2} \rangle$ $\langle \operatorname{imm} \rangle = \langle \operatorname{reg2} \rangle$		
NE	$\langle \operatorname{reg1} \rangle \langle \rangle \langle \operatorname{reg2} \rangle$ $\langle \operatorname{imm} \rangle \langle \rangle \langle \operatorname{reg2} \rangle$		
GE GEI	<reg1> <= <reg2> <imm> <= <reg2></reg2></imm></reg2></reg1>		
GT GTI	<reg1> < <reg2> <imm> < <reg2></reg2></imm></reg2></reg1>		
LE LEI			
LT LTI	<reg1> > <reg2> <imm> > <reg2></reg2></imm></reg2></reg1>		

例

LD R5.2

EQ R10,F25

(動作説明) ANDS R5.2がONかつ、R10とF25が等しい時AccがON

NOP

書式

NOP

実行条件

常時

説明

何もせずに次に進みます。

OR,ORN

書式

OR <bit> ORN <bit> 実行条件 常時

説明

ORはAccと<bit>の論理和を取りAccに格納します。

OR,ORN

書式

OR <bit> ORN <bit>

実行条件 常時

説明

ORはAccと<bit>の論理和を取りAccに格納します。 ORNはAccと<bit>の反転との論理和を取りAccに格納します。

ORS

書式

ORS 実行条件

常時

説明

Accの状態とスタックの状態の論理和をとりAccに格納します。 実行後スタックは1段ポップされます。

LDA

書式 OUT <bit>

実行条件

常時

説明

Accの状態を<bit>に出力します。

OUTBCD,OUTBCDr

書式

OUTBCD <reg:src>,<reg:dist> OUTBCDr <reg:src>,<reg:dist>

実行条件

OUTBCDはAccがONの時実行。 OUTBCDrはAccがOFFからONになる時実行。

説明

<reg:src>のバイナリーをBCDに変換して<reg:dist>に格納します。</reg:src>の値は変化しません。

OUTBCD,OUTBCDr

書式

OUTBCD <reg:src>,<reg:dist> OUTBCDr <reg:src>,<reg:dist>

実行条件

OUTBCDはAccがONの時実行。 OUTBCDrはAccがOFFからONになる時実行。

説明

<reg:src>のバイナリーをBCDに変換して<reg:dist>に格納します。 <reg:src>の値は変化しません。

RAISE

書式

RAISE

実行条件

常時

説明

AccのOFFからONになった1周期だけAccをONにします。 立ち上がりエッジを検出します。

RES,**RES**r

書式

RES <bit> RESr <bit>

実行条件

RESはAccがONの時実行。 RESrはAccがOFFからONになる時実行。

説明

<bit>をOFFします。 OUT命令とは異なり実行時だけ出力を行いますので、RES命令でOFFさせたビットは他の回

RES,RESr

書式

RES <bit> RESr <bit>

実行条件

RESはAccがONの時実行。 RESrはAccがOFFからONになる時実行。

説明

<bit>をOFFします。

OUT命令とは異なり実行時だけ出力を行いますので、RES命令でOFFさせたビットは他の回路でセットすることが出来ます。

SET,SETr

書式

SET <bit> SETr <bit>

実行条件

SETはAccがONの時実行。 SETrはAccがOFFからONになる時実行。

説明

<bit>をONします。
OUT命令とは異なり実行時だけ出力を行いますので、SET命令でONさせたビットは他の回路でリセットすることが出来ます。

SET,SETr

書式

SET <bit> SETr <bit>

実行条件

SETはAccがONの時実行。 SETrはAccがOFFからONになる時実行。

説明

<bit>をONします。
OUT命令とは異なり実行時だけ出力を行いますので、SET命令でONさせたビットは他の回路でリセットすることが出来ます。

SFT,SFTr

書式

SFT <reg:src>,<reg:dist>,<imm:nbit> SFTr <reg:src>,<reg:dist>,<imm:nbit>

実行条件

SFTはAccがONの時実行。 SFTrはAccがOFFからONになる時実行。

説明

<reg:src>を<imm:nbit>ビットシフトして<reg:dist>に格納します。 <imm:nbit>が正の場合左シフト、負の場合右シフトを行います。 <reg:src>の値は変化しません。

SFT,SFTr

書式

SFT <reg:src>,<reg:dist>,<imm:nbit> SFTr <reg:src>,<reg:dist>,<imm:nbit>

実行条件

SFTはAccがONの時実行。 SFTrはAccがOFFからONになる時実行。

説明

<reg:src>を<imm:nbit>ビットシフトして<reg:dist>に格納します。 <imm:nbit>が正の場合左シフト、負の場合右シフトを行います。 <reg:src>の値は変化しません。

SIX,DIX

書式

SIX <reg> DIX <reg>

実行条件

常時

説明

SIXは<reg>の値をソースインデックスに設定します。 DIXは<reg>の値をディスティネーションインデックスに設定します。

SUB,SUBr,SUBI,SUBIr

書式

SUB <reg:src>,<reg:dist> SUBr <reg:src>,<reg:dist> SUBI <imm>,<reg:dist> SUBIr <imm>,<reg:dist>

実行条件

SUB,SUBIはAccがONの時実行。 SUBr.SUBIrはAccがOFFからONになる時実行。

説明

SUB,SUBrは<reg:dist>から<reg:src>を減算し、<reg:dist>に格納します。<reg:src>の値は変 化しません。

SUBI,SUBIrは<reg:dist>から<imm>を減算し、<reg:dist>に格納します。

SUB, SUBr, SUBI, SUBIr

書式

SUB <reg:src>,<reg:dist> SUBr <reg:src>,<reg:dist> SUBI <imm>,<reg:dist> SUBIr <imm>,<reg:dist>

実行条件

SUB,SUBIはAccがONの時実行。 SUBr,SUBIrはAccがOFFからONになる時実行。

説明

SUB,SUBrは<reg:dist>から<reg:src>を減算し、<reg:dist>に格納します。<reg:src>の値は変 化しません。

SUBI,SUBIrは<reg:dist>から<imm>を減算し、<reg:dist>に格納します。

SUB, SUBr, SUBI, SUBIr

書式

SUB <reg:src>,<reg:dist> SUBr <reg:src>,<reg:dist> SUBI <imm>,<reg:dist> SUBIr <imm>,<reg:dist>

実行条件

SUB,SUBIはAccがONの時実行。 SUBr,SUBIrはAccがOFFからONになる時実行。

説明

SUB,SUBrは<reg:dist>から<reg:src>を減算し、<reg:dist>に格納します。<reg:src>の値は変 化しません。 SUBI,SUBIrは<reg:dist>から<imm>を減算し、<reg:dist>に格納します。

SUB,SUBr,SUBI,SUBIr

書式

SUB <reg:src>,<reg:dist> SUBr <reg:src>,<reg:dist> SUBI <imm>,<reg:dist> SUBIr <imm>,<reg:dist>

実行条件

SUB,SUBIはAccがONの時実行。 SUBr,SUBIrはAccがOFFからONになる時実行。

説明

SUB,SUBrは<reg:dist>から<reg:src>を減算し、<reg:dist>に格納します。<reg:src>の値は変化しません。 SUBI,SUBIrは<reg:dist>から<imm>を減算し、<reg:dist>に格納します。

TMR, TMRr, TMRI, TMRIr

書式

TMR <reg> TMRr <reg> TMRI <imm> TMRIr <imm>

実行条件

TMR,TMRIはAccがONの時実行。 TMRr,TMRIrはAccがOFFからONになる時実行。

説明

TMR,TMRrは<reg>の時間(1ms単位)だけAccをONにします。 TMRI,TMRIrは<imm>の時間(1ms単位)だけAccをONにします。 TMR,TMRIは入力がONの時出力もON、ONからOFFになるとそこから設定時間後に出力を OFFします。 途中で出力をOFFするには、TMRRST命令を使用します、

TMR, TMRr, TMRI, TMRIr

書式

TMR <reg> TMRr <reg> TMRI <imm> TMRIr <imm>

実行条件

TMR,TMRIはAccがONの時実行。 TMRr,TMRIrはAccがOFFからONになる時実行。

説明

TMR,TMRrは<reg>の時間(1ms単位)だけAccをONにします。 TMRI,TMRIrは<imm>の時間(1ms単位)だけAccをONにします。 TMR,TMRIは入力がONの時出力もON、ONからOFFになるとそこから設定時間後に出力を OFFします。 途中で出力をOFFするには、TMRRST命令を使用します、

TMR, TMRr, TMRI, TMRIr

書式

TMR <reg> TMRr <reg> TMRI <imm> TMRIr <imm>

実行条件

TMR,TMRIはAccがONの時実行。 TMRr,TMRIrはAccがOFFからONになる時実行。

説明

TMR,TMRrは<reg>の時間(1ms単位)だけAccをONにします。 TMRI,TMRIrは<imm>の時間(1ms単位)だけAccをONにします。 TMR,TMRIは入力がONの時出力もON、ONからOFFになるとそこから設定時間後に出力を OFFします。 途中で出力をOFFするには、TMRRST命令を使用します、

TMR, TMRr, TMRI, TMRI

書式

TMR <reg> TMRr <reg> TMRI <imm> TMRIr <imm>

実行条件

TMR,TMRIはAccがONの時実行。

TMRr,TMRIrはAccがOFFからONになる時実行。

説明

TMR,TMRrは<reg>の時間(1ms単位)だけAccをONにします。 TMRI,TMRIrは<imm>の時間(1ms単位)だけAccをONにします。 TMR,TMRIは入力がONの時出力もON、ONからOFFになるとそこから設定時間後に出力を OFFします。 途中で出力をOFFするには、TMRRST命令を使用します、

TMRRST, TMRRSTr

書式

TMRRST TMRRSTr

実行条件

TMRRSTはAccがONの時実行。

TMRRSTrはAccがOFFからONになる時実行。

説明

この命令直後のTMR,TMRr,TMRI,TMRIr命令の出力をOFFします。

直後のタイマー命令1つに対して有効です。

TMRRST, TMRRSTr

書式

TMRRST TMRRSTr

実行条件

TMRRSTはAccがONの時実行。

TMRRSTrはAccがOFFからONになる時実行。

説明

この命令直後のTMR,TMRr,TMRI,TMRIr命令の出力をOFFします。 直後のタイマー命令1つに対して有効です。

XOR,XORN

書式

XOR <bit> XORN <bit>

実行条件

常時

説明

XORはAccと<bit>との排他的論理和を取りAccに格納します。 XORNはAccと<bit>の反転との排他的論理和を取りAccに格納します。

XOR,XORN

書式

XOR <bit> XORN <bit> 実行条件

常時

説明

XORはAccと<bit>との排他的論理和を取りAccに格納します。 XORNはAccと<bit>の反転との排他的論理和を取りAccに格納します。

XTRACT, XTRACTr

書式

XTRACT <reg:src>,<imm:bpos>,<reg:dist>,<imm:nbit> XTRACTr <reg:src>,<imm:bpos>,<reg:dist>,<imm:nbit>

実行条件

XTRACTはAccがONの時実行。 XTRACTrはAccがOFFからONになる時実行。

説明

<reg:src>の<imm:bpos>の示すビット位置から<imm:nbit>の示すビット数だけ<reg:dist>の0ビット目にコピーします。<reg:src>は変化しません。<reg:dist>は下位ビットから<imm:nbit>だけ変化します。

例

XTRACT F0,4,F1,6

転送元		転送先
F0.4	\rightarrow	F1.0
F0.5	\rightarrow	F1.1
F0.6	\rightarrow	F1.2
F0.7	\rightarrow	F1.3
F1.4 - F1.31	\rightarrow	F1.4 - F1.31

XTRACT, XTRACTr

書式

XTRACT <reg:src>,<imm:bpos>,<reg:dist>,<imm:nbit> XTRACTr <reg:src>,<imm:bpos>,<reg:dist>,<imm:nbit>

実行条件

XTRACTはAccがONの時実行。 XTRACTrはAccがOFFからONになる時実行。

説明

<reg:src>の<imm:bpos>の示すビット位置から<imm:nbit>の示すビット数だけ<reg:dist>の0ビット目にコピーします。 <reg:src>は変化しません。 <reg:dist>は下位ビットから<imm:nbit>だけ変化します。

XTRACT F0,4,F1,6

転送元		転送先
F0.4	\rightarrow	F1.0
F0.5	\rightarrow	F1.1
F0.6	\rightarrow	F1.2
F0.7	\rightarrow	F1.3
F1.4 - F1.31	\rightarrow	F1.4 - F1.31



株式会社 ワコー技研 本社 〒230-0045 横浜市鶴見区末広町1-1-50 EL:045-502-4441 FAX:045-502-8624 大阪営業所 〒577-0843 大阪府東大阪市荒川3-26-10-101 EL:06-6728-1172 FAX:06-6782-1173 名古屋出張所 〒482-0011 愛知県岩倉市昭和町2-62-1-302 EL:0587-38-4033 FAX:0587-38-4033 当社HP http://www.wacogiken.co.jp/