読み取り/書き込みアクセス

この章には下記に関する情報が記載されています：

PEEK: メモリアドレスの読み取り

説明

「メモリアドレスの読み取り」命令は、メモリ領域からのデータタイプを指定しないメモリアドレスの読み取りに使用します。

注記
この命令は、「標準」メモリ領域へのアクセスにのみ使用可能です。

構文

「メモリアドレスの読み取り」命令には、以下の構文を使用します。

```
CALL PEEK
??? ???
AREA := <operand>
DBNUMBER := <operand>
BYTEOFFSET := <operand>
RET_VAL := <operand>
```

パラメータ

次の表に、「メモリアドレスの読み取り」命令のパラメータを示します。

パラメータ	宣言	データタイプ	メモリ領域	説明
AREA	Input	BYTE	I、Q、M、D、L	以下の領域を選択できます。 • 16#81: 入力 • 16#82: 出力 • 16#83: ビットメモリ • 16#84: DB • 16#1: I/O 入力
DBNUMBER	Input	DINT, DB_ANY	I、Q、M、D、L	AREA = DB ならばデータブロックの番号、それ以外は「0」
BYTEOFF-SET	Input	DINT	I、Q、M、D、L	読み取り元のアドレス 16 個の最下位ビットのみが使用されます。
RET_VAL	Output	ビット列	I、Q、M、D、L	命令の結果

有効なデータタイプの追加情報については、「関連項目」を参照してください。

命令のデータタイプを[???]ドロップダウンリストから選択できます。

注記

入力、出力またはビットメモリ領域からメモリアドレスを読み取る場合、DBNUMBER パラメータに値「 0」を割り当てる必要があります。割り当てないと、命令が無効になります。

例

次の例で、命令がどのように動作するかを示します。

STL	説明
CALL PEEK	// 命令が呼び出されます。
value_type := WORD number_type := DINT	// 命令のデータタイプ
AREA := "Tag_Area"	// データブロック領域が選択されました
DBNUMBER := "Tag_DBNumber"	// データブロックの番号
BYTEOFFSET := "Tag_Byte"	// 読み取り元のアドレス
RET_VAL := "Tag_Result"	// 結果

次の表に、命令が特定のオペランド値を使用してどのように動作するかを示します。

パラメータ	オペランド	値
AREA	Tag_Area	16#84
DBNUMBER	Tag_DBNumber	5
BYTEOFFSET	Tag_Byte	20
RET_VAL	Tag_Result	バイト値「20」

この命令は、データブロック「5」の「Tag_Byte」オペランドからアドレス「20」の値を読み取り、結果を「Tag_Result」オペランドに返します。

PEEK_BOOL: メモリビットの読み取り

説明

「メモリビットの読み取り」命令は、メモリビットからのデータタイプを指定しないメモリアドレスの読み取りに使用します。

注記
この命令は、「標準」メモリ領域へのアクセスにのみ使用可能です。

構文

「メモリビットの読み取り」命令には、以下の構文を使用します。

```
CALL PEEK_BOOL
???
AREA := <operand>
DBNUMBER := <operand>
BYTEOFFSET := <operand>
BITOFFSET := <operand>
RET_VAL := <operand>
```

パラメータ

次の表に、「メモリビットの読み取り」命令のパラメータを示します。

パラメータ	宣言	データタイプ	メモリ領域	説明
AREA	Input	BYTE	I、Q、M、D、L	以下の領域を選択できます。 • 16#81: 入力 • 16#82: 出力 • 16#83: ビットメモリ • 16#84: DB • 16#1: I/O 入力
DBNUMBER	Input	DINT, DB_ANY	I、Q、M、D、L	AREA = DB ならばデータブロックの番号、それ以外は「0」
BYTEOFF-SET	Input	DINT	I、Q、M、D、L	読み取り元のアドレス 16 個の最下位ビットのみが使用されます。
BITOFFSET	Input	INT	I、Q、M、D、L	読み取り元のビット
RET_VAL	Output	BOOL	I、Q、M、D、L	命令の結果

有効なデータタイプの追加情報については、「関連項目」を参照してください。

命令のデータタイプを[???]ドロップダウンリストから選択できます。

注記
入力、出力またはビットメモリ領域からメモリビットを読み取る場合、DBNUMBER パラメータに値「 0」を割り当てる必要があります。割り当てないと、命令が無効になります。

例

次の例で、命令がどのように動作するかを示します。

STL	説明
CALL PEEK_BOOL	// 命令が呼び出されます。
number_type := DINT	// DBNUMBER パラメータのデータタイプ
AREA := "Tag_Area"	// データブロック領域が選択されました
DBNUMBER := "Tag_DBNumber"	// データブロックの番号
BYTEOFFSET := "Tag_Byte"	// 読み取り元のアドレス
BITOFFSET := "Tag_Bit"	// 読み取り元のビット
RET_VAL := "Tag_Result"	// 結果

次の表に、命令が特定のオペランド値を使用してどのように動作するかを示します。

パラメータ	オペランド	値
AREA	Tag_Area	16#84
DBNUMBER	Tag_DBNumber	5
BYTEOFFSET	Tag_Byte	20
BITOFFSET	Tag_Bit	3
RET_VAL	Tag_Result	3

この命令は、データブロック「5」のバイト「20」で「Tag_Bit」オペランドからメモリビット「3」の値を読み取り、結果を「Tag_Result」オペランドに返します。

POKE: メモリアドレスの書き込み

説明

「メモリアドレスの書き込み」命令は、メモリ領域へのデータタイプを指定しないメモリアドレスの書き込みに使用します。

注記
この命令は、「標準」メモリ領域へのアクセスにのみ使用可能です。

構文

「メモリアドレスの書き込み」命令には、以下の構文を使用します。

```
CALL POKE
??? ???
AREA := <operand>
DBNUMBER := <operand>
BYTEOFFSET := <operand>
VALUE := <operand>
```

パラメータ

次の表に、「メモリアドレスの書き込み」命令のパラメータを示します。

パラメータ	宣言	データタイプ	メモリ領域	説明
AREA	Input	BYTE	I、Q、M、D、L	以下の領域を選択できます。 • 16#81: 入力 • 16#82: 出力 • 16#83: ビットメモリ • 16#84: DB • 16#2: I/O 出力
DBNUMBER	Input	DINT, DB_ANY	I、Q、M、D、L	AREA = DB ならばデータブロックの番号、それ以外は「0」
BYTEOFF-SET	Input	DINT	I、Q、M、D、L	書き込まれるアドレス 16 個の最下位ビットのみが使用されます。
VALUE	Input	ビット列	I、Q、M、D、L	書き込まれる値

有効なデータタイプの追加情報については、「関連項目」を参照してください。

命令のデータタイプを[???]ドロップダウンリストから選択できます。

注記

入力、出力またはビットメモリ領域にメモリアドレスを書き込む場合、DBNUMBER パラメータに値「 0」を割り当てる必要があります。割り当てないと、命令が無効になります。

例

次の例で、命令がどのように動作するかを示します。

STL	説明
CALL POKE	// 命令が呼び出されます。
value_type := WORD number_type := DINT	// 命令のデータタイプ
AREA := "Tag_Area"	// データブロック領域が選択されました
DBNUMBER := "Tag_DBNumber"	// データブロックの番号
BYTEOFFSET := "Tag_Byte"	// 読み取り元のアドレス
VALUE := "Tag_Value"	// 書き込まれる値

次の表に、命令が特定のオペランド値を使用してどのように動作するかを示します。

パラメータ	オペランド	値
AREA	Tag_Area	16#84
DBNUMBER	Tag_DBNumber	5
BYTEOFFSET	Tag_Byte	20
VALUE	Tag_Value	16#11

この命令は、データブロック「5」内のメモリ領域「20」を値「16#11」で上書きします。

POKE_BOOL: メモリビットの書き込み

説明

「メモリビットの書き込み」命令は、メモリビットへのデータタイプを指定しないメモリアドレスの書き込みに使用します。

注記
この命令は、「標準」メモリ領域へのアクセスにのみ使用可能です。

構文

「メモリビットの書き込み」命令には、以下の構文を使用します。

```
CALL POKE_BOOL
???
AREA := <operand>
DBNUMBER := <operand>
BYTEOFFSET := <operand>
BITOFFSET := <operand>
VALUE := <operand>
```

パラメータ

次の表に、「メモリビットの書き込み」命令のパラメータを示します。

パラメータ	宣言	データタイプ	メモリ領域	説明
AREA	Input	BYTE	I、Q、M、D、L	以下の領域を選択できます。 • 16#81: 入力 • 16#82: 出力 • 16#83: ビットメモリ • 16#84: DB • 16#2: I/O 出力
DBNUMBER	Input	DINT, DB_ANY	I、Q、M、D、L	AREA = DB ならばデータブロックの番号、それ以外は「0」
BYTEOFF-SET	Input	DINT	I、Q、M、D、L	書き込まれるアドレス 16 個の最下位ビットのみが使用されます。
BITOFFSET	Input	INT	I、Q、M、D、L	書き込まれるビット
VALUE	Input	BOOL	I、Q、M、D、L	書き込まれる値

有効なデータタイプの追加情報については、「関連項目」を参照してください。

命令のデータタイプを[???]ドロップダウンリストから選択できます。

注記
入力、出力またはビットメモリ領域にメモリビットを書き込む場合、DBNUMBER パラメータに値「 0」を割り当てる必要があります。割り当てないと、命令が無効になります。

例

次の例で、命令がどのように動作するかを示します。

STL	説明
CALL POKE_BOOL	// 命令が呼び出されます。
number_type := DINT	// DBNUMBER パラメータのデータタイプ
AREA := "Tag_Area"	// データブロック領域が選択されました
DBNUMBER := "Tag_DBNumber"	// データブロックの番号
BYTEOFFSET := "Tag_Byte"	// 読み取り元のアドレス
BITOFFSET := "Tag_Bit"	// 読み取り元のビット
VALUE := "Tag_Value"	// 書き込まれる値

次の表に、命令が特定のオペランド値を使用してどのように動作するかを示します。

パラメータ	オペランド	値
AREA	Tag_Area	16#84
DBNUMBER	Tag_DBNumber	5
BYTEOFFSET	Tag_Byte	20
BITOFFSET	Tag_Bit	3
VALUE	Tag_Value	M0.0

この命令は、バイト「20」内のデータブロック「5」のメモリビット「3」を値「M0.0」で上書きします。

POKE_BLK: メモリ領域の書き込み

説明

「メモリ領域の書き込み」命令は、メモリ領域の内容をデータタイプを指定せずに他のメモリ領域にコピーします。

注記
この命令は、「標準」メモリ領域へのアクセスにのみ使用可能です。

構文

「メモリ領域の書き込み」命令には、以下の構文を使用します。

```
CALL POKE_BLK
???
AREA_SRC := <operand>
DBNUMBER_SRC := <operand>
BYTEOFFSET_SRC := <operand>
AREA_DEST := <operand>
DBNUMBER_DEST := <operand>
BYTEOFFSET_DEST := <operand>
COUNT := <operand>
```

命令の構文は以下の部分で構成されます。

パラメータ	宣言	データタイプ	メモリ領域	説明
AREA_SRC	Input	BYTE	I、Q、M、D、L	ソースのメモリ領域で、以下の領域を選択できます。 • 16#81: 入力 • 16#82: 出力 • 16#83: ビットメモリ • 16#84: DB
DBNUM-BER_SRC	Input	DINT, DB_ANY	I、Q、M、D、L	AREA = DB ならばソースのメモリ領域内のデータブロックの番号、それ以外は「0」
BYTEOFF-SET_SRC	Input	DINT	I、Q、M、D、L	書き込まれるソースのメモリ領域のアドレス 16 個の最下位ビットのみが使用されます。
AREA_DEST	Input	BYTE	I、Q、M、D、L	宛先のメモリ領域で、以下の領域を選択できます。 • 16#81: 入力

				• 16#82: 出力 • 16#83: ビットメモリ • 16#84: DB
DBNUM-BER_DEST	Input	DINT, DB_ANY	I、Q、M、D、L	AREA = DB ならば宛先のメモリ領域内のデータブロックの番号、それ以外は「0」 16 個の最下位ビットのみが使用されます。
BYTEOFF-SET_DEST	Input	DINT	I、Q、M、D、L	書き込まれる宛先のメモリ領域のアドレス 16 個の最下位ビットのみが使用されます。
COUNT	Input	DINT	I、Q、M、D、L	移動されるバイト数

有効なデータタイプの追加情報については、「関連項目」を参照してください。

命令のデータタイプを[???]ドロップダウンリストから選択できます。

注記
入力、出力またはビットメモリ領域にメモリアドレスを書き込む場合、DBNUMBER パラメータに値「 0」を割り当てる必要があります。割り当てないと、命令が無効になります。

例

次の例で、命令がどのように動作するかを示します。

STL	説明
CALL POKE_BLK	// 命令が呼び出されます。
number_type := DINT	// DBNUMBER パラメータのデータタイプ
AREA_SRC := "Tag_Source_Area"	// ソースメモリ領域でデータブロック領域が選択されました
DBNUMBER_SRC := "Tag_Source_DBNumber"	// ソースメモリ領域のデータブロックの番号
BYTEOFFSET_SRC := "Tag_Source_Byte"	// ソースメモリ領域の読み取り元のアドレス
AREA_DEST := "Tag_Source_Area"	// 宛先メモリ領域でデータブロック領域が選択されました
DBNUMBER_DEST := "Tag_Source_DBNumber"	// 宛先メモリ領域のデータブロックの番号
BYTEOFFSET_DEST := "Tag_Source_Byte"	// 宛先メモリ領域の読み取り元のアドレス
COUNT := "Tag_Count"	// 移動されるバイト数

次の表に、命令が特定のオペランド値を使用してどのように動作するかを示します。

パラメータ	オペランド	値
AREA_SRC	Tag_Source_Area	16#84
DBNUMBER_SRC	Tag_Source_DBNumber	5
BYTEOFFSET_SRC	Tag_Source_Byte	20

AREA_DEST	Tag_Destination_Area	16#83
DBNUMBER_DEST	Tag_Destination_DBNumber	0
BYTEOFFSET_DEST	Tag_Destination_Byte	30
COUNT	Tag_Count	100

この命令は、アドレス「30」から始まるビットメモリのメモリ領域内で、アドレス「20」から始まるデータブロック「5」から 100 バイトを書き込みます。