ラダー図開発ツール ＬＤ Ｃｖ2!

取扱説明書(第8版)

■はじめに

LD Cv2!は、シーケンス制御を記述するためのプログラミング言語であるラダー図(Ladder Diagram)の開発ツールです。

これを使って開発したプログラムは最終的にROMに書き込み、I/Oポートのあるマイクロコンピュータ上で実行させることを想定しています。

LD Cv2!の目的は、マイクロコンピュータのI/O制御を仮想PLCのラダー図で記述することです。

LD Cv2!の主な機能には、ラダー図を編集するエディタ機能と、作成したラダー図をC言語ソースに変換する変換機能があります。（おまけ的にシミュレーション機能もあります。）

扱えるラダー図容量は、横7シンボル×縦1024行です。 Ver2.0からワードデータが扱えます。

LD Cv2!で開発する場合、制御ロジック以外の部分(電源投入時の初期設定や、I/OとC関数とのデータの受け渡しのプログラムはプログラマがターゲットマシンに合わせてプログラムを作る必要があります。LD Cv2!が生成した制御ロジック部のプログラムと、プログラマが開発した制御ロジック以外のプログラムをCコンパイラにかけることでターゲットマシン上で実行可能なプログラムとなります。

変換前言語	変換後言語	実際のターゲットマシン	LD Cv2!以外に必用なツール
ラダー図	C言語(ソース)	C開発環境のある マイコンシステム	C開発環境 (コンパイラ、リンカ、ROM書込ツール)

なお、LD Cv2!が生成するプログラムには、自身の異常(マイクロコンピュータの暴走、メモリ変化など)を検出、処理するコードは含まれていません。適用分野によっては、一般のマイクロコンピュータ応用製品と同様、フェイルセーフの仕組みを追加する必要があります。

■動作環境

日本語版Windows10で動作します。

.NETアプリケーションであるためWindows10以外でも動作するはずですが、確認はしていません。

■インストールと削除

適当なZIP解凍ツールで解凍し、解凍したファイル全体を1つのフォルダに入れます。フォルダの中のldcv2.exeを実行することでLD Cv2!が起動します。削除する場合は、フォルダごと削除してください。

■使用条件、保証範囲

LD Cv2!は「フリーソフト」です。無償・無保証で提供されるソフトウェアです。複製・再配布は、自由に行ってかまいません。

■その他

・仮想PLCの仕様については「LD Cv!/LD Cv2!仮想PLC仕様書」(配布ファイルに同梱)をご覧ください。

・この取扱説明書(第8版)は、LD Cv2!(Ver3.x)に対応しています。

・本文書で使用する製品名は、各社、各組織の登録商標、または商標です。

第１章 操作方法

■簡単な操作例

ここではLD Cv2!の簡単な操作例をしましょう。

(1)LD Cv2!を起動する

LD Cv2!の実行ファイルldcv2.exeを実行するとラダー図ウィンドウが開きます。起動直後のラダー図ウィンドウは、回路が何もないラダー図を読み出している状態になっています。このときの画面左上の０は行番号を意味し、行カーソルがあるため背景が水色になっています。タイトルバーはファイル名とモードを表示し、起動直後はファイル名が「未定義.lad」、モードが「読出モード」になっています。(図1.1)

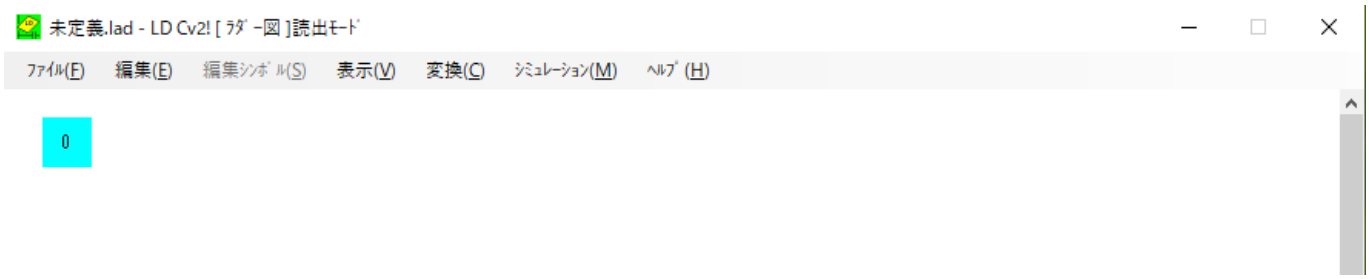

図1.1 起動直後(読出モード)

(2)[編集:挿入モード]メニューを実行

回路を作成するため、まず[編集:挿入モード]メニューを実行します。これは、現在のカーソル行(０行目)手前に挿入するラダー図１回路分をこれから作成することを意味します。(図1.2)

メニューの実行方法には、通常のマウス操作以外に、①マウス操作(行カーソル(水色)をクリックし、挿入ﾓｰﾄﾞを選択)、②キーボード操作([ALT+E][↓][↓][Enter])、③キーボード操作([Enter][↓][↓][Enter])、④キーボード操作 (Esc][→][↓][↓][Enter])があります。

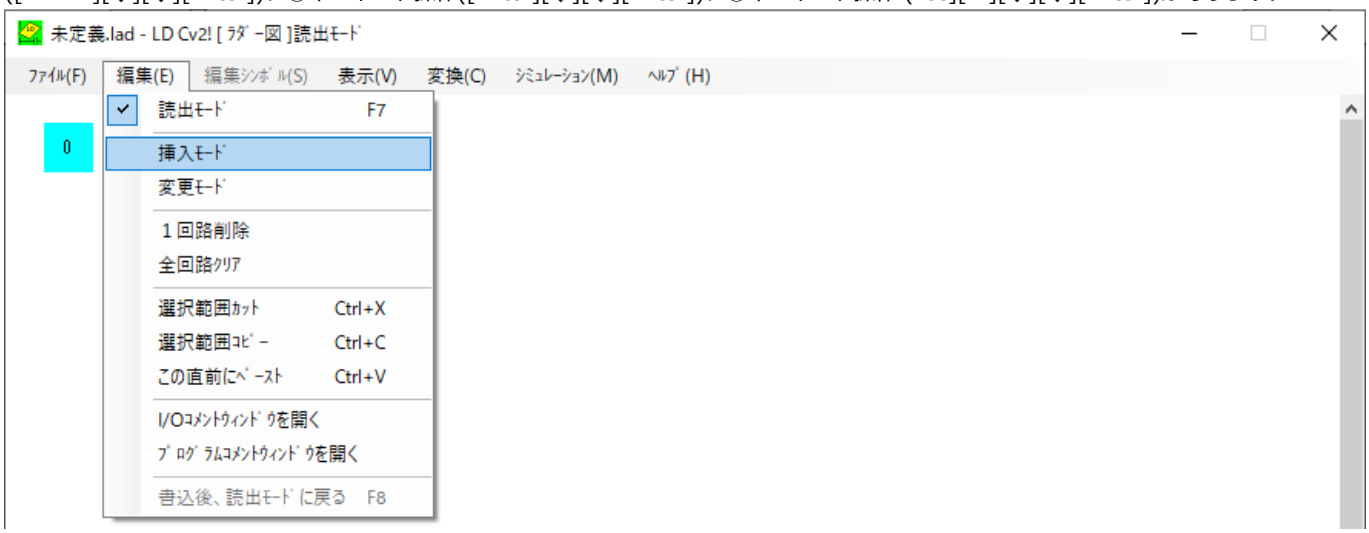

図1.2 挿入モードを選択

[編集:挿入モード]メニューを実行すると、ウィンドウは７行分の編集画面に切り替わり、行番号は＋０から＋６を水色の背景で表示、画面上にはセルカーソル(灰色の正方形)が表示されます。(図1.3) このセルカーソルは、シンボルやI/Oアドレスをどの位置に入力するかという編集対象のセルを示し、上下左右の矢印キーの押下やマウスのクリックで移動できます。

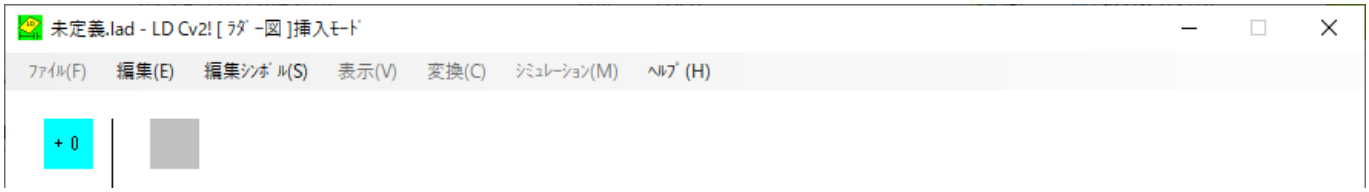

図1.3 挿入モードに移行、セルカーソルを表示

(3)[編集シンボル:A接点]メニューを実行

セルカーソル位置にA接点を入力するため、[編集シンボル:A接点]メニューを実行します。(図1.4)

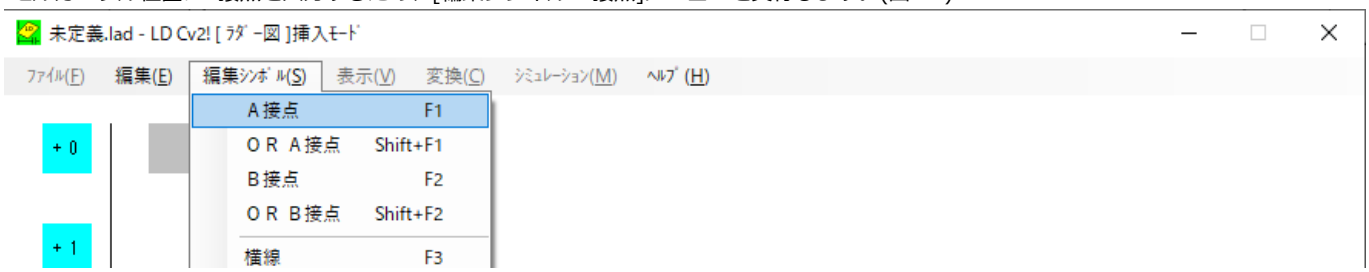

図1.4 A接点を選択

すると、セルカーソル上にA接点を表示した後、I/O番号ボックスが表示されます。I/O番号を入力し(図1.5)、OKをクリックするとセルカーソルの位置にI/O番号が表示され、セルカーソルは右へ１セル移動します。

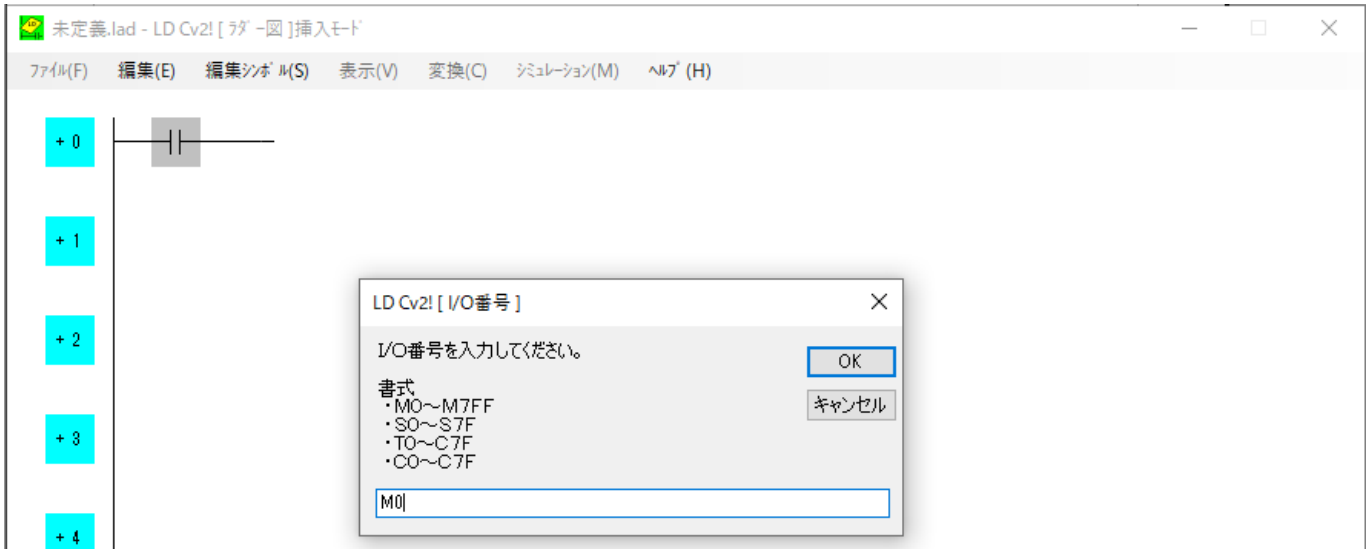

図1.5 I/O番号(M0)を入力

(4)回路を完成させる

このようにして、接点、コイル、横線、縦線、I/Oアドレスなどを入力して、１回路を完成させます。(図1.6)

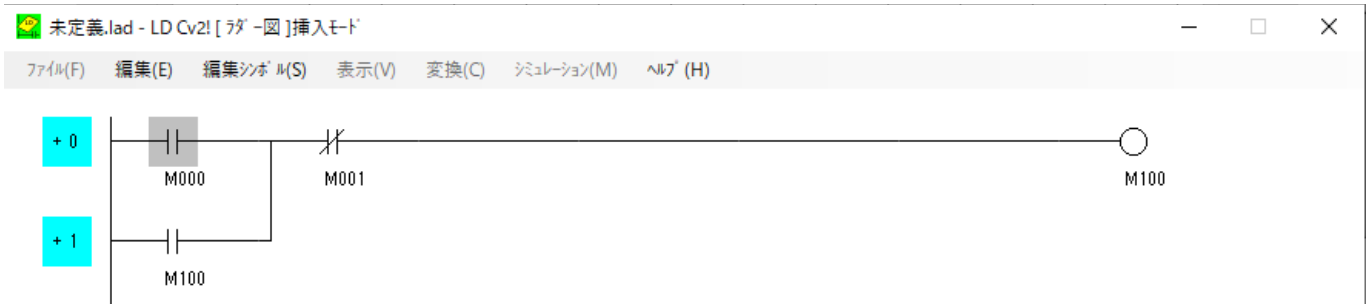

図1.6 １回路を完成させた直後

(5)[編集:書込後、読出モードに戻る]メニューを実行

1回路を完成させたら、[編集:書込後、読出モードに戻る]メニューを実行します。(図1.7)

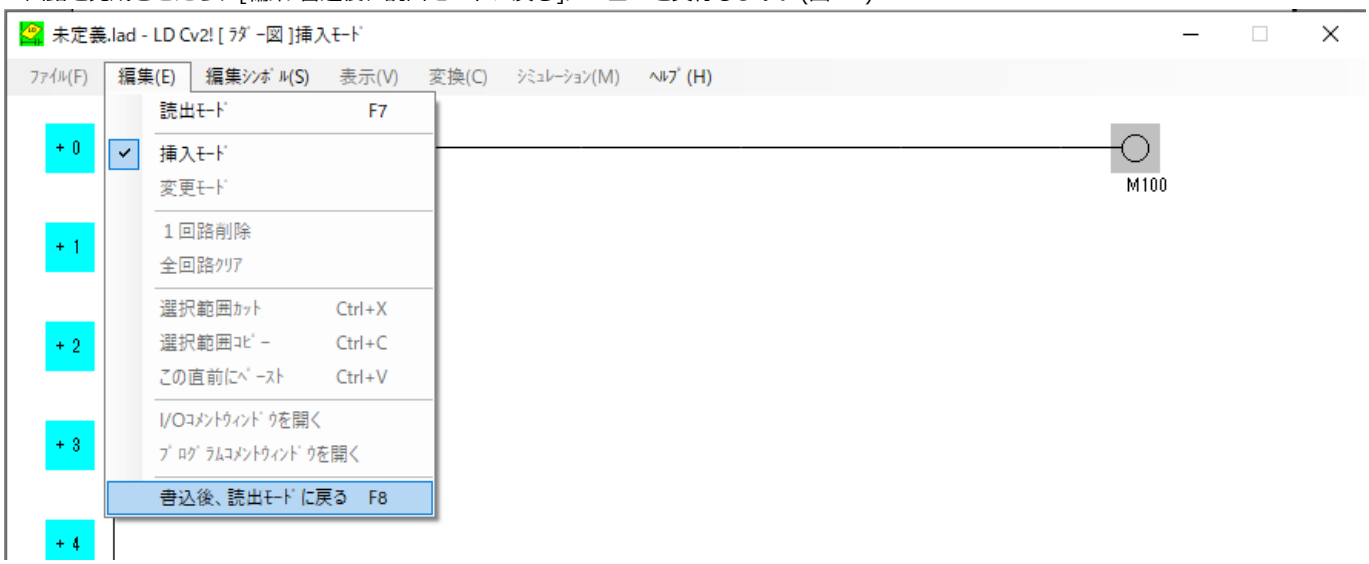

図1.7 書込後、読出モードに戻る

回路は整形後、開いているラダー図に挿入して読出モードに戻ります。回路の挿入によって、以降の行番号は自動的に更新されます。続けて回路を入力する場合は、再び[編集:挿入モード]メニューを実行します。

(6)回路に誤りがあった場合

[編集:書込後、読出モードに戻る]メニューを実行したとき回路に誤りがあった場合は、エラーメッセージを表示します。回路の誤りを修正し、再び[編集:書込後、読出モードに戻る]メニューを実行してください。

(7)編集を中止して読出モードに戻る場合

回路編集途中で編集を中止して読出モードに戻るには、[編集:読出モード]メニューを実行します。この場合、編集中ののラダー図は変化せずに、読出モードに戻ります。

(8)回路を変更する場合

すでに入力してある回路を変更する場合は、行カーソル(水色)を目的の回路の行番号まで移動させた後、[編集:変更モード]メニューを実行します。以降は、挿入モードと同様です。

(9)変換条件を設定する

ラダー図が完成したら、変換実行を行う前に、どのような条件で変換するかを設定しておきます。この設定は[変換:変換条件ウィンドウを開く]メニューを実行し、開いた変換条件ウィンドウ上で行います。(図1.8)

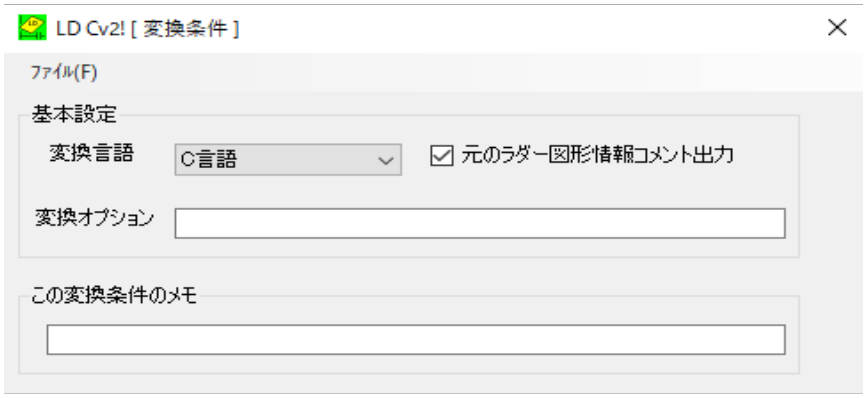

図1.8 変換条件ウィンドウ

ここでは、デフォルトの設定をそのまま使うので、確認後、変換条件ウィンドウを閉じて、ラダー図ウィンドウに戻ります。

(10)ラダー図をC言語に変換し、ファイルに保存する

[変換:変換実行]メニューを実行すると、変換を開始します。変換が終了するとボックスが開き保存場所を聞いてきますので、適当な場所にファイル名を指定し、保存ボタンをクリックします。変換実行時には自動的にラダー図チェックが行われます。このチェックでエラーを検出した場合、エラーメッセージを表示し変換は中止されます。

(11)変換結果を見る

変換が正常に終了すれば、変換結果がファイルとなって保存されます。このファイルはテキストファイルなのでエディタで開いてみることができます。次のリストは変換結果の一部です。(図1.9) なお、等幅フォント(MS ゴシックなど)で表示しないとラダー図形が崩れて表示されますのでご注意ください。

```
        /* User Program */

/*      |                                                                                  */
/*    0+----| |-----+----|/|-------------------------------------------------------------( )    */
/*      |M000        |M001                                                          M100      */
/*      |            |                                                                        */
/*      |            |                                                                        */
/*    1+----| |-----+                                                                        */
/*      |M100                                                                              */
/*      |                                                                                  */

        /* LD M000 */

        stk = stk << 1;
        stk = stk | (rg & 0x01);
        rg = m[0x000];


        /* OR M100 */

        rg = rg | m[0x100];
```

図1.9 変換結果の一部

以上が簡単な操作例となります。

■操作機能

次に、LD Cv2!の機能について説明します。

機能はメニューから選択、実行しますが、メニューの選択方法としては

①マウスでメニューを選択、左クリック

②Alt+英文字またはEscでプルダウンメニューを有効にして、[↑][↓][←][→]でメニューを選択、[Enter]キーを押す

があります。

表1.1に「ラダー図ウィンドウ」上のメニュー機能を説明します。

また、表1.1のメニュー機能以外に、セルカーソル位置のカット(Ctrl+X)、コピー(Ctrl+C)、ペースト(Ctrl+V)機能が使えます。

表1.1 メニュー機能

メニュー		機能説明
ファイル	新規作成 開く 上書き保存/名前を付けて保存	プログラムファイルのロード・セーブに関するコマンド群です。 LD Cv2!は、ラダー図、I/Oコメント、プログラムコメントが一体になったものをプログラムとして扱います。
	ラダー図をテキスト出力	ラダー図を疑似的に英数記号で表現して、テキストファイル出力します。テキストエディタでこのファイルを開き、等幅フォント(MS ゴシックなど)で印刷することで印刷機能の代用としてください。 等幅フォントを使用してもラダー図の縦線が横にずれて表示される場合は、(*1)をご覧下さい。
	終了	LD Cv2!を終了します。

編集	読出モード 挿入モード 変更モード 書込後、読出モードに戻る	LD Cv2!起動直後は、読出モードです。 ラダー図を編集するには、挿入モードまたは変更モードに移行する必要があります。挿入モードは、新規に１回路作成し行カーソル位置の回路の直前に挿入します。変更モードは、行カーソル位置の１回路を変更します。 編集結果は「書込後、読出モードに戻る」を実行することで、プログラムに反映されます。
	１回路削除	行カーソル(水色)位置の１回路をラダー図から削除します。
	全回路クリア	プログラムからラダー図をすべて削除します。I/Oコメント、プログラムコメントは残っています。
	選択範囲カット 選択範囲コピー この直前にペースト	回路単位でのカット、コピー、ペーストを行うコマンド群です。選択範囲の始点は行カーソル(水色)です。終点の指定方法は、シフトキーを押しながら上下矢印キーを押すか、シフトキーを押しながらマウスで終点の行番号をクリックします。これで選択範囲の行番号の背景が黄色に変化します。カット(コピー)した回路は、ファイル(temp.clp)に記憶されていて、ペーストを行うときに使用されます。 複数のLD Cv2!ウィンドウ間での回路のコピー&ペーストが可能です。
	I/Oコメントウィンドウを開く	I/Oコメントウィンドウを開き、I/O番号とI/Oコメント(I/O番号に紐づいた文字列)の対応を表示します。I/Oコメントは半角12文字(全角6文字)の範囲で指定してください。これを超える文字は末尾がカットされます。 I/.Oコメントウィンドウ上では、特殊メモリのコメント追加、CSV形式のコメントファイルのインポート、エクスポートができます。
	プログラムコメントウィンドウを開く	プログラムコメントウィンドウを開き、プログラムコメント(プログラムに紐づいた文字列)を表示します。プログラムコメントウィンドウ上では、プログラムコメントの編集ができます。
編集シンボル	(メニュー省略)	挿入モード、変更モードのとき、ラダー図を編集するコマンド群です。 タイマ番号1Fで設定値1234を入力する場合は「T1F 1234」のように、タイマ番号と設定値の間に半角スペースを空けて入力します。カウンタも同様です。 特殊メモリSの入力書式は「仮想PLC仕様書」をご覧ください。 挿入モード、変更モード中は、Enterキーを押すと編集シンボルメニューが表示され、そのまま編集シンボルが選択できます。
表示	前回路／次回路／先頭回路 最終回路の次／指定行に移動 カーソル位置に移動	読出モードのとき、ラダー図中のどの部分を画面に表示するかを指定するコマンド群です。
	I/O検索／コイル検索	[I/O検索]では指定されたI/O番号(M,S,T,C)の接点とコイルの両方が検索対象です。 [コイル検索]では指定されたI/O番号のコイルのみが検索対象です。 Wの検索はサポートしていません。
	I/Oコメント表示切替:表示 I/Oコメント表示切替:非表示	I/Oコメントの表示／非表示を切替えます。
	命令語表示ウィンドウを開く	命令語表示ウィンドウを開きます。 命令語表示ウィンドウは、行カーソルのある１回路分の命令語表現を表示し、行カーソルの移動に追従します。
変換	ラダー図チェック	現在のラダー図の構文をチェックします。エラーがあった場合、エラー番号とエラー内容を表示します。ラダー図チェックは仮想PLCの標準モデルを前提にしているため、SmallモデルやLargeモデル固有のエラーは検出しません。

	変換条件ウィンドウを開く	変換条件ウィンドウを開きます。 このウィンドウでは、ラダー図をどのような条件で変換するかを設定します。変換条件はファイル保存でき、また保存した変換条件ファイルの読込ができます。変換条件のデフォルトの拡張子は”cdt”です。 変換言語で「C言語(Smallモデル)」を選択した場合、MはM0～M7Fの128点しか使用できませんが、ターゲット(マイコン)のRAM容量が少なくて済みます。 また、変換言語で「C言語(Largeモデル)」を選択した場合、ワードメモリW0～W7FFの2048ワードが使用でき、ワード(2バイト)データの演算が行えます。 変換オプションで「-NH」を記述した場合、変換結果の1行目のヘッダ情報は表示しません。
	変換実行	現在開いているプログラム(ラダー図)を、現在の変換条件に従って変換し、変換結果をファイル名を付けて保存します。 変換途中でエラーが検出された場合、エラーメッセージを表示し、変換を中止します。
	変換結果をメモ帳で開く	変換結果をWindowsのメモ帳で開きます。 等幅フォントを使用してもラダー図の縦線が横にずれて表示される場合は、欄外(*1)をご覧下さい。
シミュレーション(*2)	シミュレーション開始	読出画面上で、仮想PLCの実行(スキャン)をシミュ―レーションします。 シミュ―レーション中は、接点とコイルのON/OFF状態、タイマ・カウンタの経過値を表示します。接点とコイルのシンボルはONのとき緑色になりますが、 MCS、MCRコイルはその状態にかかわらず常に白色に表示します。 また、タイマ、カウンタの経過値は画面右に括弧付きで表示、スキャン回数(下位3桁)は画面右下に背景が水色で表示します。 ワードメモリ(W)はモニタウィンドウ上でのみ表示します。
	シミュレーション終了	シミュ―レーションを終了し、読出モードに戻ります。
	モニタ	モニタウィンドウを開きます。 このウィンドウは、シミュ―レーション中の最大32点(ﾜｰﾄﾞ)のI/O番号の状態を表示し、ONのとき緑色になり、タイマ、カウンタは経過値を表示します。 ワードメモリは、10進数と16進数の両方を表示します。 モニタI/O番号の指定には、メニューから指定、I/O番号の空いているエリア(灰色の長方形)をクリック、の方法があります。
	Mセット・リセット タイマ経過値変更 カウンタ経過値変更 Wデータ変更	ウィンドウを開き、その中でシミュ―レーション中の指定したI/O(M)のON/OFF、タイマ・カウンタの経過値変更、ワードメモリ(W)のデータ変更ができます。
ヘルプ	バージョン表示	LD Cv2!のバージョン情報などを表示します。 なお、ヘルプメニュ―から取扱説明書を開くことはできません。

(*1)Ver1.0にI/Oコメント文字幅処理の誤りがあり、ラダー図の縦線がずれて表示されます。Ver1.0で作成したラダー図は、Ver1.1以上のLD Cv2!で開き、I/Oコメントウィンドウ上でコメントをエクスポート後、再インポートすることで、以後は縦線が正しく表示されるようになります。

(*2)シミュレーションは仮想PLCの「Largeモデル」として動作し、レジスタ(GRA、GRB)、ワードメモリ(W)の記憶サイズは16ビットと扱います。

第2章 C言語によるターゲットへの組み込み

■ターゲットへの組み込み方法

ここでは、C言語によるターゲット(マイコン)への組み込み方法について説明します。

(1)LD Cv2!が生成するプログラム

変換言語に「C言語」、「C言語(Smallモデル)」または「C言語(Largeモデル)」を指定した場合、変換実行を行うとラダー図はC言語で記述された関数plcに変換されます。

この関数plcには、ラダー図の制御ロジック以外に、I/O(M,S,T,C,W)のメモリ領域の確保、この関数に与えるコマンドの処理がすべて含まれています。

この関数plcの内部は関数外部から直接操作することはできず、関数plcの引数にコマンドを渡すことで操作します。

(2)プログラマが作成するプログラム

プログラマは、必要最低限のプログラム(フェーズ1～フェーズ5)をC言語で作成する必要があります。 (図2.1)

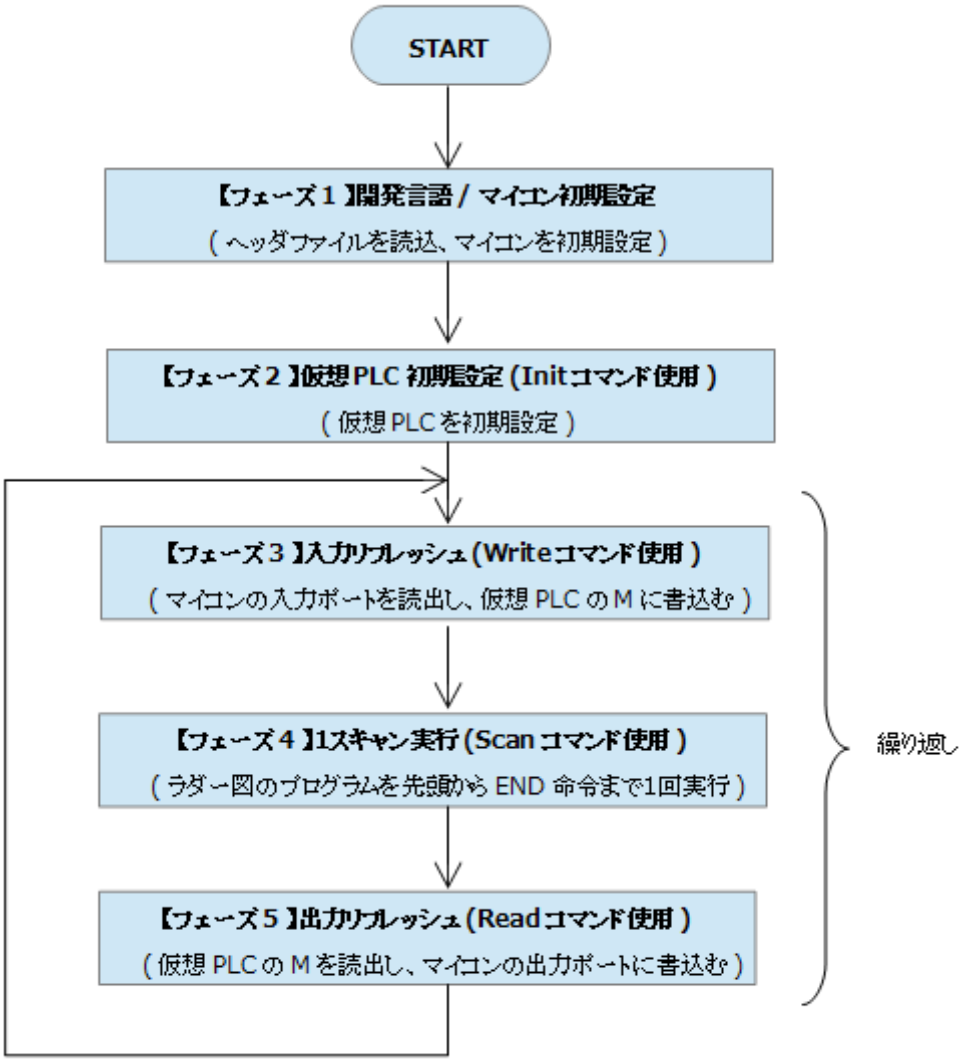

図2.1 必要最低限のプログラム

このうち、フェーズ2～フェーズ5でLD Cv2!が生成した関数plcのコマンドを使用します。フェーズ1 はアセンブラで記述する場合もあります。Largeモデルの場合、フェーズ3でワードデータを入力リフレッシュするには「ワードWriteコマンド」を、フェーズ5でワードデータを入力リフレッシュを使用するには 「ワードReadコマンド」を使用します。

作成ツールとしては、ターゲットのC開発環境のエディタか、メモ帳などのテキストエディタを使用します。

■関数plcのコマンド

表2.1にLD Cv2!が生成する関数plcのコマンド機能について説明します。

関数の型、引数の型は、変換言語に「C言語」または「C言語(Smallモデル)」を指定した場合は

unsigned char plc(unsigned char cmd,unsigned int prm1,unsigned int prm2)

「C言語(Largeモデル)」を指定した場合は

unsigned int plc(unsigned char cmd,unsigned int prm1,unsigned int prm2)

です。

表2. 1 関数plcのコマンド機能 (1/2)

Writeコマンド	Readコマンド
■書式 plc('W',prm1,prm2) ■パラメータ prm1 = MのI/O番号(0x000～0x7FF) prm2 = Mへの書込値(0または1) ■動作 パラメータprm1 で指定した仮想PLCのMエリアのI/O番号の1点に、パラメータprm2 で指定したデータを書き込みます。 ■戻り値：0 ■使用例 仮想PLCのM0に値１を、 M7FFに値0を書き込みます。 plc('W',0x0,1) plc('W',0x7ff,0)	■書式 plc('R',prm1,0) ■パラメータ prm1 = MのI/O番号(0x000～0x7FF) ■動作 パラメータprm1 で指定した仮想PLCのMエリアのI/O番号の1点からデータを読み出し、その値(0または1)を返します。 ■戻り値：Mの読出値 ■使用例 仮想PLCのM100とM1FFから値を読み出し、a、 b に代入します。 a=plc('R',0x100,0) a=plc('R',0x1ff,0)
Scanコマンド	**Initコマンド**
■書式 plc('S',0,0) ■動作 ラダー図のプログラムを先頭からEND命令まで１回実行します。 ■戻り値：0 ■使用例 ラダー図のプログラムを１スキャン実行します。 plc('S',0,0)	■書式 plc('I',0,0) ■動作 仮想PLCの内部メモリを初期化します。 このコマンドは、他のコマンドを使用する前に１回実行してください。 ■戻り値：0 ■使用例 仮想PLCの内部メモリを初期化します。 plc('I',0,0)

表2.1 関数plcのコマンド機能(Largeモデルでのみ有効) (2/2)

ワードWriteコマンド	ワードReadコマンド
■書式 plc('w',prm1,prm2)　　(注) 小文字の w ■パラメータ prm1 = WのI/O番号(0x00～0x7FF) prm2 = Wへの書込値(0x0000～0xFFFF) ■動作 パラメータprm1 で指定した仮想PLCのWエリアのI/O番号の1 ワードに、パラメータprm2 で指定したデータを書き込みます。 ■戻り値：0 ■使用例 仮想PLCのW0に値0xFFFFを、 W7FFに値1234を書き込みます。 plc('w',0x0,0xffff) plc('w',0x7ff,1234)	■書式 plc('r',prm1,0)　　(注) 小文字の r ■パラメータ prm1 = WのI/O番号(0x00～0x7FF) ■動作 パラメータprm1 で指定した仮想PLCのWエリアのI/O番号の1 ワードからデータを読み出し、その値(0x0000～0xFFFF)を返します。 ■戻り値：Wの読出値 ■使用例 仮想PLCのW10とW1FFから値を読み出し、a、 b に代入します。 a=plc('r',0x10,0) b=plc('r',0x1ff,0)

■開発の実例

簡単な例題でLD Cv2!による開発を行ってみます。

(1)開発対象

ここでは開発対象マイコンとして、入手が容易で、簡単にLD Cv2!が適用可能なマイコンボードArduinoを使用します。

Arduinoの開発言語はC言語とほぼ同一であるため、LD Cv2!の生成するコードは問題なくコンパイルできますし、電源投入直後の初期設定(図2.1のフェーズ１)の大部分をプログラマが書く必要が無いため手軽に開発が可能です。

(2)例題の仕様

マイコンボードにスイッチとLEDを接続し、スイッチの状態によりLEDの点滅速度が切り替わる制御を行います。(表2.2)

表2.2 例題の仕様

マイコン	Arduino Duemilanoveマイコンボード (Arduino UNOでも可)
接続I/O	ポートP12にスイッチを接続 (スイッチを押すと、入力データが１になる) ポートP13にLEDを接続 (出力データが１で、LEDが点灯する)
制御仕様	スイッチの状態によりLEDの点滅速度が切り替わる 点滅タイミングには、PLCの特殊メモリS13(１０００スキャン反転)、S14(１００００スキャン反転)を使用
C開発環境	Arduino開発環境を使用

(3)開発手順

表２.2の仕様で開発をおこなってみます。

【手順1】ラダー図の作成

LD Cv2!を使ってラダー図を作成します。

LD Cv2!の仮想PLCには、１０００スキャンまたは１００００スキャン実行毎にON/OFFが反転する特殊なメモリS13とS14が用意されています。

ラダー図上でポート12のスイッチ信号のA接点に接点S14を、B接点に接点S13を直列接続してやれば、必ずどちらか一方が点滅信号源になり、これらを並列接続してポートP13のLEDに出力(コイルを接続)してやれば目的の動作を行います。

ただ、 LD Cv2!の仮想PLCはI/Oポートを直接扱えないため、とりあえずここでは「メモリM0、M100をこれらのI/Oポートとし、後でP12→M0、M100→P13のデータ転送プログラムを追加する」ことにします。

では、LD Cv2!でラダー図を書きます。(図2.2)

１回路目は説明した通りです。プログラム末には、必ずラダー図の終わりを示すENDコイルを書きます。

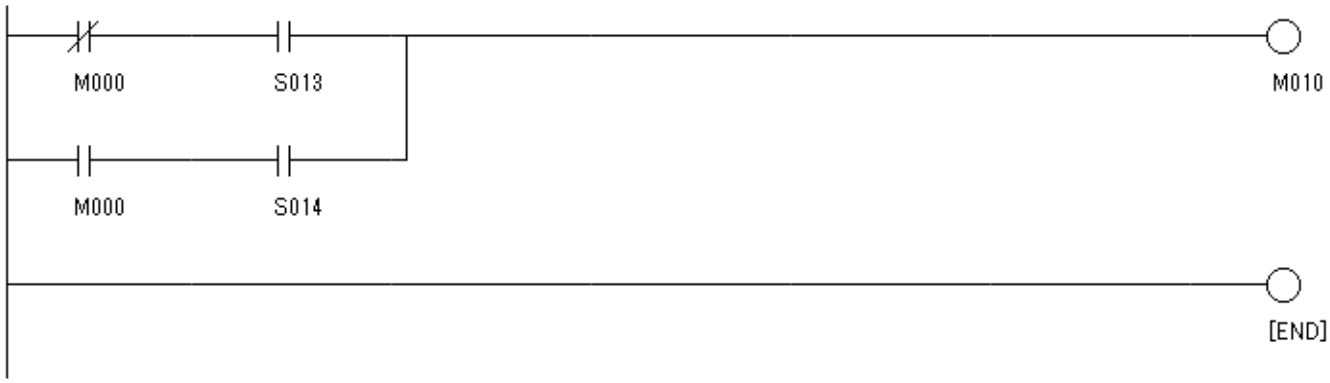

図2.2 作成したラダー図 (pg11a.ladと同じ)

このプログラムはサンプルプログラム(pg11a.lad)と同じなので、これを読み込んでもかまいません。

【手順2】C言語への変換

変換条件はデフォルトがC言語ですが、ArduinoはRAM容量が小さいので「C言語(Smallモデル)」に変更しておきます。

次に[変換：変換実行]メニューを実行し、変換結果をファイル名 LED.cとして保存します。

これで、例題の制御仕様のＣ言語プログラム(関数plc)が生成されました。

【手順3】マイコン開発環境上での作業

以後は、Arduinoの開発環境を使っての作業となります。

Arduino IDEを立ち上げます。リスト２.1で、黒字がArduino IDEが自動的に生成したフレームで、ここにプログラム（青字、赤字）を入力します。フェーズ１～２のプログラムはsetup() 関数の中(またはプログラム先頭)に記述し、フェーズ３～５のプログラムはloop() 関数の中に記述します。

Arduino ではloop() 関数は繰り返し実行されます。

プログラムの入力が終わったらArduino IDE上でプログラム(スケッチ)を保存し、LD Cv2!生成したLED.c を同じスケッチフォルダに入れます。

```
//【フェーズ１(1)】開発言語/マイコン初期設定
//ピンに名前を付与
#define p12 12
#define p13 13

//LD Cv2!が生成したプログラム(関数plc)をインクルード
#include "LED.c"

void setup() {
  //【フェーズ１(2)】開発言語/マイコン初期設定(ピンの入出力を設定)
  pinMode(p12, INPUT);
  pinMode(p13, OUTPUT);

  //【フェーズ２】仮想plc初期設定
  plc('I', 0, 0);
}

void loop() {
  //【フェーズ３】入力リフレッシュ(マイコンの入力ポートP12のON/OFF情報を、関数plc内部のM0に与える)
  plc('W', 0x00, digitalRead(p12));
```

```
  //【フェーズ４】１スキャン実行(ラダー図のプログラムを、先頭からEND命令まで１回実行)
  plc('S', 0, 0);

  //【フェーズ５】出力リフレッシュ(関数plc内部のM10のON/OFF情報を、マイコンの出力ポートP13に与える)
  digitalWrite(p13, plc('R', 0x10, 0));
}
```

リスト2.1 作成したArduinoプログラム

【手順4】マイコンボードに転送、実行

[スケッチ:マイコンボードに書き込む]で、ＲＯＭ書込が行われます。

エラーが無ければ、この時点でスイッチを押すとＬＥＤが点滅し、スイッチの操作で点滅速度が切り替わります。

制御プログラムは、マイコンボードにROMに書き込まれているのでパソコンと切り離しても電源さえ供給していれば動作します。

以上

来歴(ラダー図開発ツール LD Cv2! 取扱説明書)

2019/10/22 新規作成、第１版とする

2020/01/03 記載内容をVersion1.0に対応、第２版とする

2020/03/15 記載内容をVersion1.1に対応、図2.1を差替え、他、第３版とする

2020/05/16 記載内容をVersion1.2に対応、第４版とする

2020/11/01 記載内容をVersion2.0に対応、第５版とする

2020/11/03 表2.1(ｺﾏﾝﾄﾞ)、その他を訂正、第６版とする

2020/11/29 記載内容をVersion2.1に対応、表1.1(ﾜｰﾄﾞ数)、表2.1(Wの範囲)を訂正、他、第７版とする

2021/01/10 記載内容をVersion3.0に対応、シミュレーション機能に関する記述を変更、第８版とする