ＰＬＣラダーソフト以外との言語比較検討

Programmable Logic Controller

蜂谷　2022.4.6

目次

注）タッチパネルはＰＬＣではないが、ＰＬＣと合わせて使用されることが多いので取り上げた。
タッチパネルにおいても、ＳＴ言語のような高級言語が使用できる。

１．はじめに

ＰＬＣについて；

元々は、ＰＣ（プログラマブルコントローラ）と呼んでいましたが、ＰＣ（パーソナルコンピュータ）の登場と普及によって、混合を避けるためＰＬＣ（Programmable Logic Controller プログラマブルロジックコントローラ）と呼ぶようになりました。

世界ではＰＬＣは、ゼネラルモーターズ（アメリカ）の要求で、1969 年に Modicon（現シュナイダーエレクトリック）が最初に商品化しました。

日本でのＰＬＣについては、三菱電機が、1978 年頃から MELSEC シリーズを発売し、MELSEC-K, A, QnA, Q, iQ と現在に至ります。三菱電機は、ＰＬＣを商品名でシーケンサと名付け、未だにシーケンサと言えばＰＬＣの代名詞と呼ばれるほどです。

オムロンは、1989 年に SCY-P5R を 1999 年には C20 とそれから CQM1, C200H, C1000H, CJ1 等と続いています。

現在当たり前のように使用しているＵＳＢメモリ、ＣＤやＤＶＤなどの記憶媒体は、1980 年頃には無くて、少し前まで良く使われていたフロッピーディスクさえもありませんでした。プログラムの記憶はカセットテープで行っていました。

プログラムを書き込むのは、プログラムコンソールで、命令を直にニーモニック又はコーディングリストとして書き込んでいました。ここでの検討書では、インストラクションリストと呼んでいます。命令もほとんど無くて、ハードで組むシーケンス回路の延長のような AND, OR, NOT, OUT, タイマーやカウンタが主で、演算は四則演算くらいしかありませんでした。

しかし、現代では技術の発展と共に、高機能なＰＬＣが段々と造られ、比較演算、算術演算、データ変換、データ転送、プログラム分岐、論理演算、ロテーション、シフト、ビット処理、データ処理、サブルーチン、文字列処理、特殊関数、時計用命令、ネットワーク命令等、数百を超える命令を持つＰＬＣとなっています。それに加えて、ここで検討するラダー以外の言語を有するＰＬＣも出てきました。

弊社では、ＰＬＣのソフト（プログラミング）は、ほとんどがラダー言語で行ってきましたが、最近のＰＬＣはラダー言語以外の言語も扱えるようになってきました。

ラダー言語が不得意な処理を、ラダー以外での言語で処理することによって、簡単に、分かり易くて、ソフト作成時の時間も短縮できるのではないかと考え、それぞれの言語についての紹介と実際の例をあげながらのプログラミングを記載し、利点や欠点についても考慮し、どの言語が採用できるか等を検討・記載することとしました。

今後のソフト作成に役立てることを切望します。

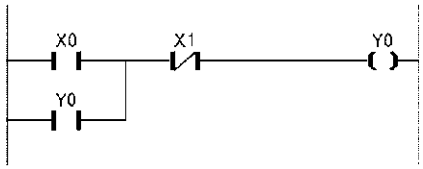
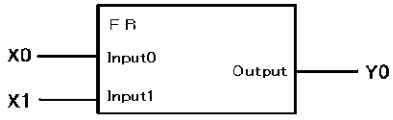
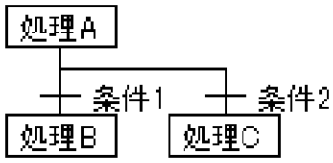
また、弊社においては、ＰＬＣの使用メーカーは三菱電機、オムロンが多いので、このメーカーを中心に記載します。

因みに、世界シャアでは、シーメンス、Ｒｏｃｋｗｅｌｌ(Allen-Bradley)、三菱電機。

日本国内では三菱電機、オムロン、富士、キーエンスだそうです。

２．ＰＬＣのプログラミング言語とは

ＰＬＣのプログラミング言語は、ＩＥＣ６１１３１－３（国際規格）で定められた世界共通の標準規格で、５種類定義されています。
以下、簡単に説明します。

言語体系	名称	説明	イメージ(例)
グラフィック言語	ＬＤ ラダーダイアグラム (Ladder Diagram)	日本で最も普及しているプログラム言語 リレーシーケンスのスタイル	
	ＦＢＤ ファンクションブロック ダイアグラム (Function Block Diagram)	様々な機能を持った電子部品(ﾌｧﾝｸｼｮﾝと呼ばれる箱BLOCK)と接続してプログラミングする	
	ＳＦＣ シーケンシャル ファンクションチャート (Sequential Function Chart)	制御システムの流れを記述。フローチャートのようなもの。条件分岐が記述し易く順次動作する処理に向いている	
テキスト言語	ＩＬ インストラクションリスト (Instruction List)	コンピュータのアセンブリ言語によく似ている	LD M11 MOV K100 D0 - D0 K10 D1
	ＳＴ ストラクチャードテキスト (Structured Text)	BASICやＣなど高級言語に似た形で記述。複雑な演算や文字列処理が得意	Ave:=(Val1+Val2+Val3)/3; IF M11 = ON THEN D11 := D1 - D2; END_IF;

ロボット制御やＦＡ機器の制御には、無くてはならない存在となったＰＬＣですが、その機能については、以前のＰＬＣと比べると、最近では格段に優れたものになっています。

日本の生産現場のほとんどが、ラダー言語が使用されてきたのですが、昨今の現場においてはＦＡ機器等との通信・制御と様々なコントロールを行うことが増えてきたため、ラダー言語だけでは不得意な処理が多くなり、非常に手間暇がかかるのが実態です。

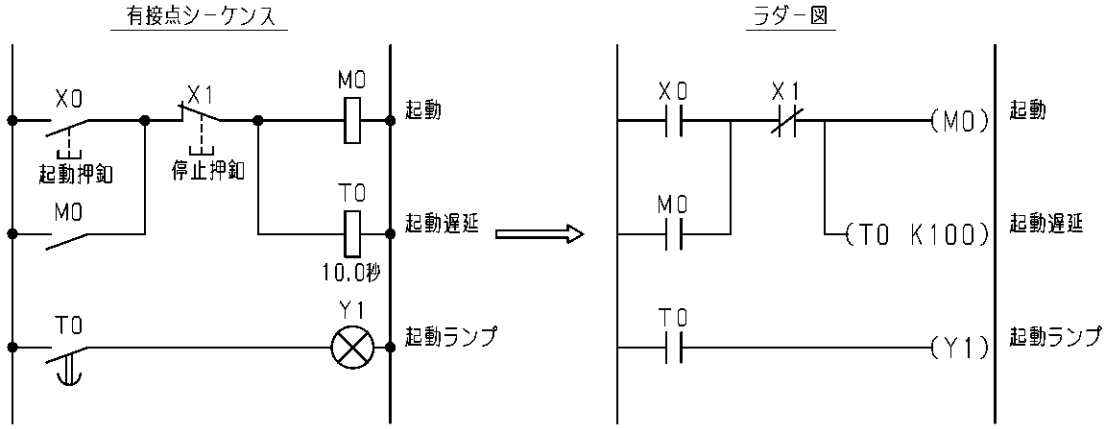
３．言語の紹介・説明

ＰＬＣのプログラミング言語の５種類を、前表よりもう少し詳しく説明します。
ラダー言語に慣れた人も、ラダー言語以外に、どういうプログラムがあるのか、どういう時に利用できそうかを考えて欲しいと思います。

３－１．ＬＤ(ラダーダイアグラム)

ラダーというものはリレー回路を記号化したもので、「ラダー図」という梯子のような図形で表す。　リレーシーケンス（有接点シーケンス）で回路設計していた電気技術者は、シーケンス図を書いたり見たりする感覚で仕事ができます。
縦に両端の線が母線で、母線の間に横書きでプログラムを書きます。接点、コイル、タイマー等を上から順に、リレーシーケンス回路図のように書いていきます。

以下は、有接点リレーシーケンスとＰＬＣラダー図の比較です。



基本は自己保持回路で、順番にＯＮしていくイメージです。
弊社でも、最もよく使うラダーですので、プログラムの組み方や注意について、簡単に説明しておきます。

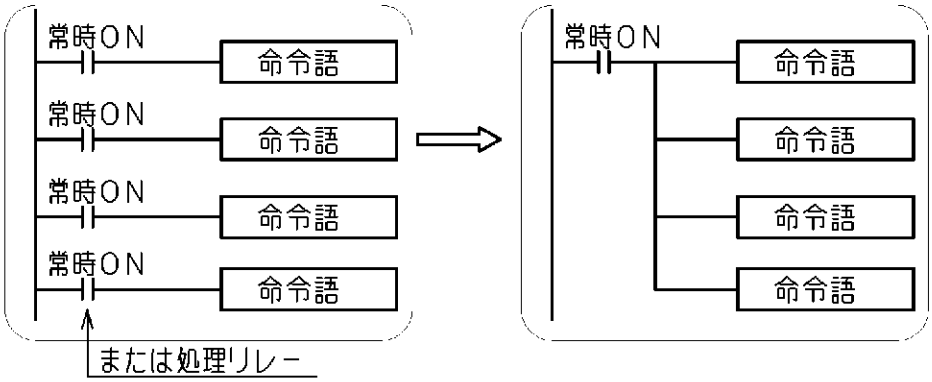
ラダーの組み方と注意

ＰＬＣのプログラムを組む前に、全体の動作のフローチャートやタイムチャートを記載し、また、プログラムで使用する機器デバイスやメモリアドレスもある程度決めた上でプログラム作成に臨んだほうがスムースにいきます。

プログラム作成においては、ＰＬＣのソフト（ラダー）に限らず、プログラムは同じ処理を書くのにも様々な記述が可能であるため、作成者によって全く違う形になることも珍しくありません。
また、作成者本人であっても、数年後に見ると内容を忘れてしまって、苦労することもあるかも知れません。
そのような状況を想定して、「見やすい・分かり易いラダーはバグも少なくメンテナンス性にも優れている」ラダーを作成することを心掛けることが必要です。
以下、いくつかの注意点を紹介します。

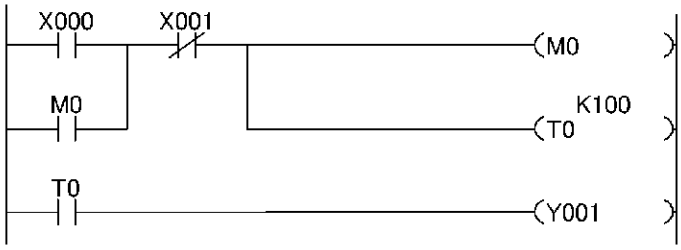
① 常時ＯＮリレー（または処理リレー）でまとめる

ラダーは比較的単純な命令の組合せのため、１つのデータを計算するだけで何行にもなることが良くあります。そんな時、先頭に「常時ＯＮ」のリレーを書いて回路をまとめると、処理の区切りが分かりやすくなります。



② コメント（Ｉ／Ｏコメント・行コメント）を入れる

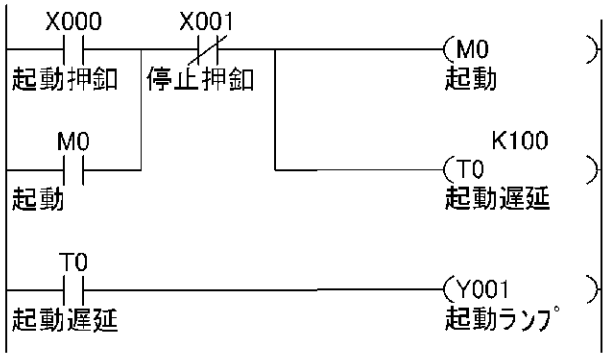
コメントないラダー図



コメントのないラダーは文字や数値だけで、さっぱり分からない。

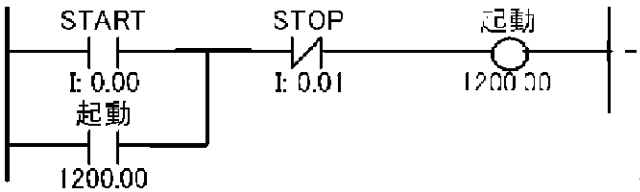
行コメントとＩ／Ｏコメントを入れたラダー図（三菱電機では行コメントをステートメントと呼んでいる）

＊起動回路＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝



注意としては、いい加減なコメントを入れない。
必ずコメントを入れ、的確な内容にしてミスを防ぐ。

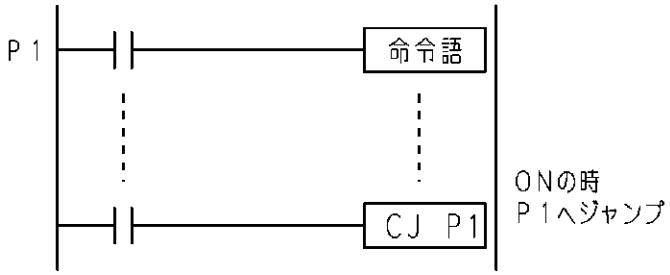
また、最近のＰＬＣは、アドレスの番号をラベル(変数名)として使う事も出来る。



←ＰＬＣがオムロンの例

③　ジャンプ命令は使わない

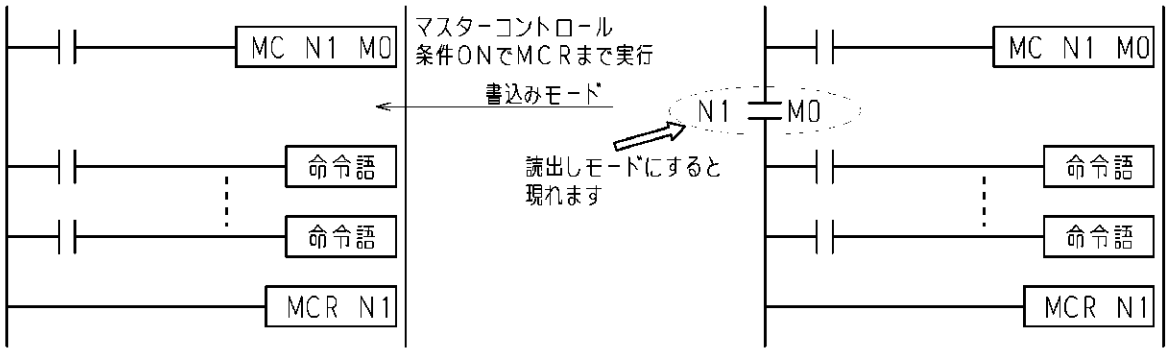
三菱電機は、無条件ジャンプ命令［ＪＭＰ］（ＦＸシリーズにはない）
　　条件付ジャンプ命令［ＣＪ］（Ｑ，ＦＸシリーズ共あり）
オムロンは、無条件ジャンプ命令［ＪＭＰ］、条件付ジャンプ命令［ＣＪＰ］など



条件が整ったとき指定のポイントやＪＭＥの位置に飛んで（ＪＵＭＰ）処理を続行するのが「ＪＭＰ（ＣＪ）命令」です。
この命令より上にジャンプ先を指定すると、通常は上から下に順に進んでいき、また初めに戻ってサイクル処理するという、ラダーの基本的なシーケンスの流れが変わってしまうため、処理を追いかけるのがとても大変になってしまいます。
特別な場合以外は、使用しないようにしましょう。

④　マスターコントロール（インターロック）は出来るだけ使用しない

ジャンプ命令に似た命令として、条件のＯＮ／ＯＦＦで複数の回路行の制御ができる命令があります。
三菱電機は、マスターコントロール［ＭＣ］（［ＭＣＲ］までの命令を ON/OFF する）
オムロンは、インターロック［ＩＬ］　　（［ＩＬＣ］までの命令を ON/OFF する）

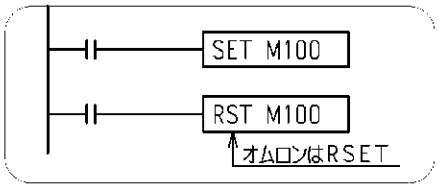


（これは三菱の例。オムロンは［ＩＬ］－［ＩＬＣ］で母線に接点は現れません）

これを使用することで効率の良い切換シーケンスが出来ますが、複雑な回路にするとＭＣとＭＣＲ（オムロンはＩＬとＩＬＣ）間の命令が、とても読みづらくなります。
マスターコントロール（インターロック）のある回路やマスターコントロールのない回路でのコイルや接点の受け渡しがある場合なんかは、どうしてこのコイルがＯＮしないのかと現地で悩んだこともあります。
極力使用しないのを原則として、どうしても使用する場合には十分注意して使用するようにしましょう。

⑤　ＳＥＴ／ＲＳＴ命令の使用は最低限に

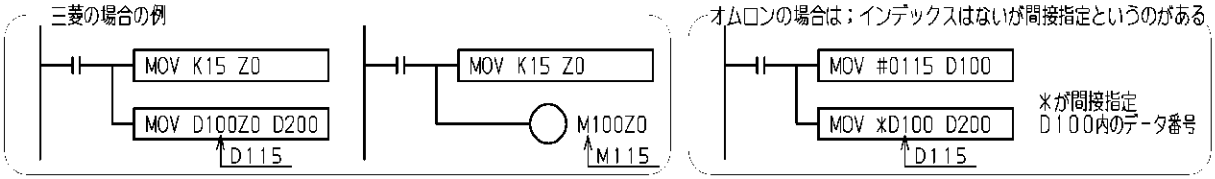
ＳＥＴ命令はビットをＯＮする命令です。ＳＥＴ命令の実行条件が不成立になっても、ＳＥＴ命令によってＯＮしたビットはそのままＯＮの状態を保ちます。
ＲＳＴ命令により、ビットをＯＦＦすることができます。



手軽にビットＯＮ／ＯＦＦできますが、
多用しすぎると最終的に一体どのＳＥＴ／ＲＳＴ命令でビットがＯＮ／ＯＦＦしているのかとても分かりにくくなってしまいます。
ビットをＯＮ／ＯＦＦするには、なるべく通常のコイル出力（ＯＵＴ命令）を使うことが望ましいですが、やむを得ずＳＥＴ／ＲＳＴ命令を使う場合は、対象のビットに対してＳＥＴ／ＲＳＴを一組とするように気をつけます

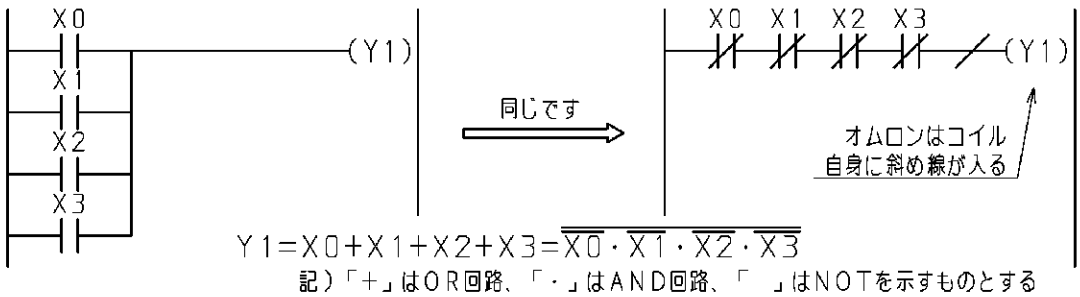
⑥　コイル出力に対してのインデックス処理は最低限に

インデックス処理とは、ＰＬＣのメーカー／機種によって呼び方は異なりますが、デバイスメモリに対してオフセットする処理です。
例えば、三菱電機ＰＬＣの場合、D100 のデバイスにインデックス値「15」をつけると、D115(D100+15)のデバイスにアクセスします。
インデックス処理は、プログラム量の圧縮（製作コストの削減）や、条件によって結果の出力先のデバイスを切り替える場合など、とても便利な機能です。
しかし、コイル出力にインデックスを使ってしまうと、デバイス検索で見つからず､回路を追いかけ難くなります。　使用の際は、行コメントなどに注釈を記入して、分かるようにしておく必要があります。



⑦　ＯＲ回路の表現変更

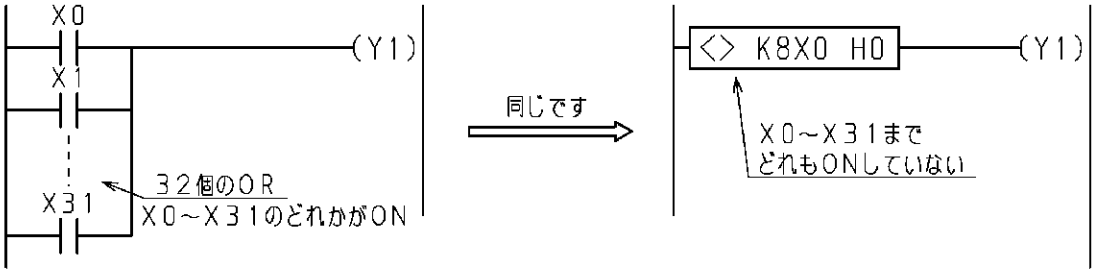
ＯＲ（並列）接点回路が複数ある場合、次のような回路にすることも出来ます。



論理演算で、ド・モルガンの定理などでも記載されていると思いますが、複数の

入力に対しての論理和（ＯＲ）から論理積（ＡＮＤ）への変換式の一例です。
接点数が多くなると、画面に収まらなくなり見づらいプログラムとなりますので、
ひと目で確認できれば、デバッグ時などにはとても助かります。

また、入力のデバイスを順に並べること（プログラムの工夫）によって、ワードで扱う事とすれば、以下のようにすることも出来ます。

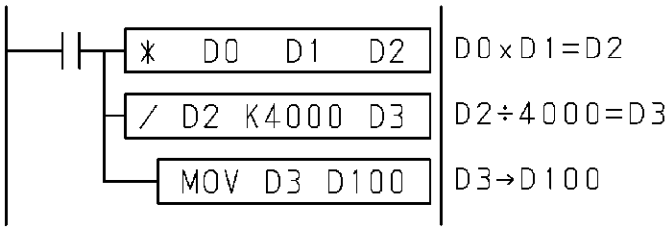


⑧　演算値がおかしい

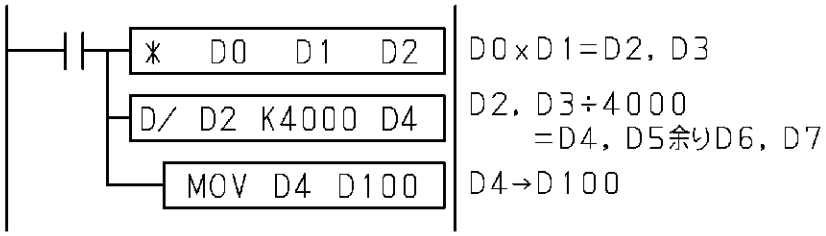
「四則演算式は間違いないのに、演算結果が違っている」と、このような場合は、以下の事が考えられます。　例を示します。

例）データ〔Ｄ０〕x〔Ｄ１〕÷〔４０００〕を〔Ｄ１００〕に入れる

（Ｄ０，Ｄ１，Ｄ１００　は１６ビット-32768 ～ 32767 とする）



四則演算などの命令によっては、演算結果が２ワードや４ワードになるものがあり、
使用していないと思って使ってしまう。
データが上書きされています。
次のようにすれば解決されます。



掛け算（＊）は演算結果が２ワード（Ｄ２，Ｄ３）となります。
従って、２ワードの割り算命令（Ｄ／）を使用します。
割り算の結果として、商がＤ４，Ｄ５に余りがＤ６，７に入ります。
このように、命令によって使用するワードが違ってきます。
データメモリの使用リスト等を作って、重複を避けるようにすることが必要です。

⑨　スケール変換の誤差が大きい！

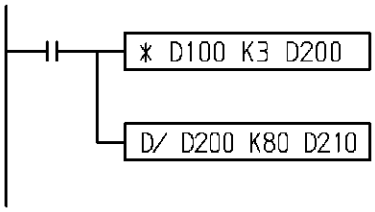
Ａ／Ｄ変換ユニットからのデータ（0～4000 など）を工業スケール値（0～15.0kg など）に変換する場合、四則演算を使ってスケール変換を行うことがよくあります。ＰＬＣ内のデータは通常、整数で扱われることが多いため、演算結果に多少なりとも誤差が生じてしまいます。誤差を少なくするために、まず「乗算」から行うように注意します。

上記の例では、入力 D100(0～4000)で、出力 D210(0～150；データは小数点以下は扱えないので１０倍とします)とすると、

$$D210(\text{工業スケール値}) = D100(\text{入力値}) \times \frac{\overset{3}{\cancel{150}}}{\underset{80}{\cancel{4000}}}$$

となり、

乗算を先に計算した場合

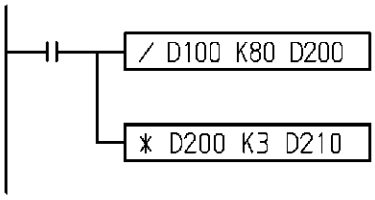


(D100)×3=(D200, D201)

(D200, D201)÷80
=(D210, D211)‥余り(D212, D213)

三菱の場合の例　　オムロンは「K」が「＋」又は「&」に変わる

乗算を後に計算した場合



(D100)÷80=(D200)‥余り(D201)

(D200)×3=(D210, D211)

「乗算が先」と「乗算が後」での値を比較してみると

D100(アナログ入力値) 0～4000の内の一例	D210(工業スケール値出力)演算値		実際の値 0～150
	乗算が先	乗算が後	
850	31	30	31.87
1268	47	45	47.55
2110	79	78	79.12
3025	113	111	113.43

演算結果は小数点以下切り捨てとなる
乗算が後のほうが誤差が大きい

乗算を後にすると、誤差が広がりますので、乗算は先にした方が良いです。
特に小数点以下をより詳しく表示させたい時などには誤差を少なくできます。

しかし、最近では、浮動小数点の演算命令を持つＰＬＣが多くなってきましたので、浮動小数点の演算を使用すれば、乗算や除算の順番を気にしなくて済むようになってきました。

⑩　プログラムは処理回路ごとに分ける

単純なプログラムでは必要ありませんが、複雑なプログラムになった場合は、プログラムをブロック（又はセクション）ごとに細分化して、読みやすいプログラムにする必要があります。　例えば、「初期設定回路」、「演算回路」、「自動回路」、「手動回路」、「出力回路」と、言うイメージです。
また、複数の作業者でプログラムを作成する時にも、分割して作業が出来るメリットがあります。
ただし、余り細分化し過ぎると反って見づらくなったりしますので、程ほどにする必要があります。
また、細分化するとＰＬＣのメモリを使いますので、プログラムのメモリ容量を大きくしないといけないかも知れませんので、注意下さい。

⑪　プログラムの置換時の注意

ＰＬＣが古くなると更新が必要となりますが、更新しようとした時には、もう当時のＰＬＣタイプは製造中止になっていることが多々あります。
各メーカーからＰＬＣ更新について、「ＰＬＣ置き換えリプレースのおすすめ」などのお知らせも多く出ていますが、特に注意したい点を記載します。

・新旧の命令が違う

新旧のＰＬＣでは命令が異なるものも多数ありますが、メーカーのサポートツールでＰＬＣ機種変換がほぼ出来ます。

・メモリ番号、サイズが違う

新旧で補助リレー、特殊リレー、データレジスタ等の番号が異なれば、変更する必要があります。

・リフレッシュ方式とダイレクト方式

ＰＬＣのプログラムのスキャンＥＮＤで出力状態が切り替わるのが「リフレッシュ方式」、プログラムの途中の命令実行時に出力状態が切り替わるのが「ダイレクト方式」です。　各メーカーによってもどちらの方式を採用しているのか確認する必要があります。　尚、一般的には入力関係デバイスはプログラムのスキャン頭で、出力関係はスキャンＥＮＤ時に行う「リフレッシュ方式」で、入出力以外のデバイスは命令時即実行の「ダイレクト方式」です。

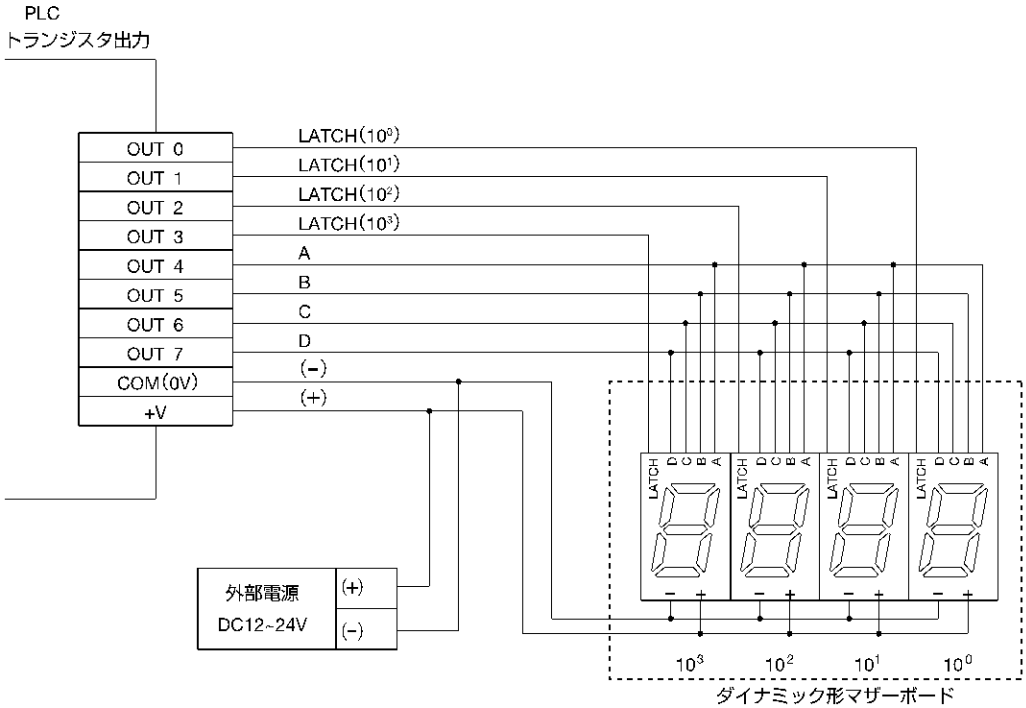
ＰＬＣ置換時には、この方式が変わることがありますので、場合によってはプログラムを変更する必要があります。

例えば、三菱電機Ａシリーズから Ｑシリーズに切り替える場合、「タイマー、カウンタの処理」はＡシリーズではＥＮＤで処理しますが、Ｑシリーズの場合は命令実行時に処理されます。　従って、タイマー、カウンタの下側に記載のプログラムは直ぐに実行されず、次のスキャンに実行されることになります。　このことで問題が

発生すれば、プログラムを見直す必要があります。　筆者もこれで悩んだことがあります。

・セグメント表示器の表示がおかしい

ダイナミック方式のセグメント表示器（下図例参照）において、ラッチを掛けて１桁ごとに出力して表示する場合には、出力するタイミング時間が狂ってしまうと、表示が、ばらついたり、固まったりすることがあります。
これは、ＰＬＣのスキャンタイムが早くなったためと思われます。
以前のＰＬＣのスキャンタイムが２０ｍｓとすると、新しいＰＬＣでは１０分の１の２ｍｓ程になっています。　この時間が早すぎて表示がうまく切り替わっていないのです。　こういう場合は、ＰＬＣの内部タイマーを伸ばしてやるかＰＬＣのスキャンタイムを伸ばしてやれば正常表示となります。

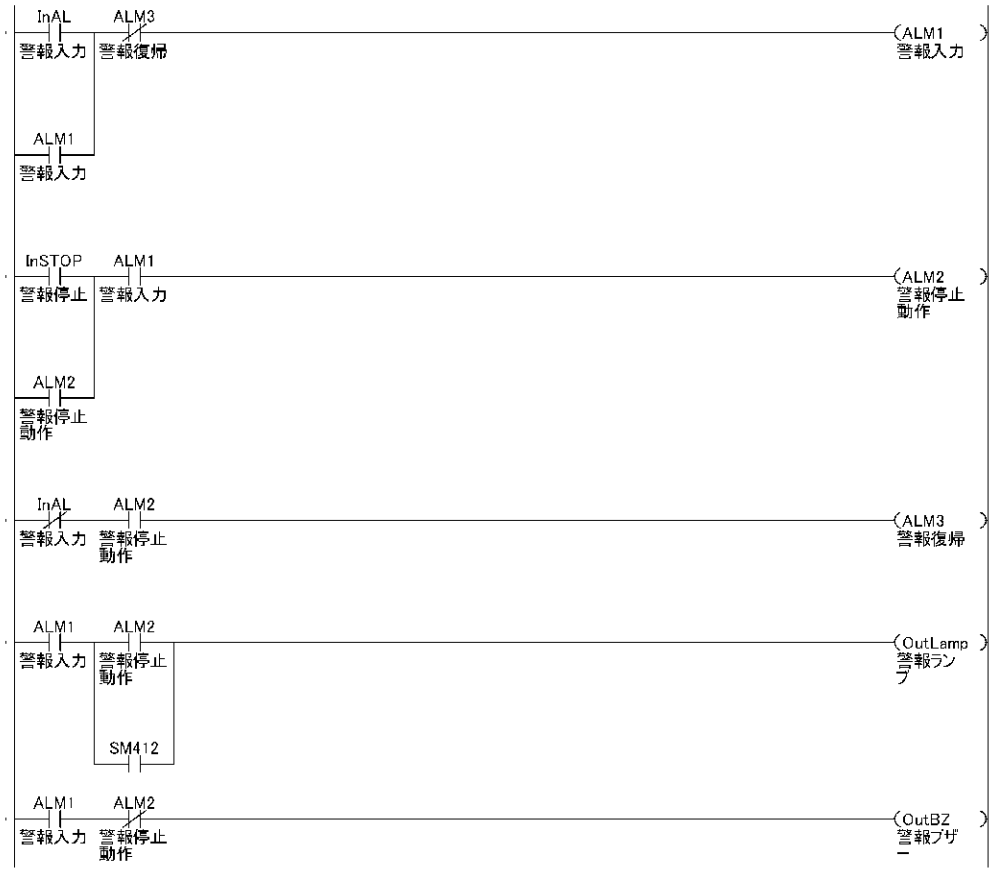


また、複数のデジタルスイッチを取り込む（入力）時にも、タイミング時間ごとに順次桁を切り替えながら取り込んでやる方法の場合は、上記と同様にＰＬＣの内部タイマーを伸ばすかＰＬＣのスキャンタイム設定を伸ばすようにすれば正常な取込ができるようになります。

他の機器との通信などがあれば、ＰＬＣのスキャンタイムが早くなっているため通信速度に合わせた設定時間にする必要があります。

３－２．ＦＢＤ（ファンクションブロックダイアグラム）

ＦＢＤとは、繰り返し使用する回路ブロックをシーケンスプログラムで流用するために、部品化したものです。
以下のような警報回路を考えてみます。まず、ファンクションブロックを作成します。
<u>ファンクションブロック</u>　データ名：ＦＢＴｅｓｔ

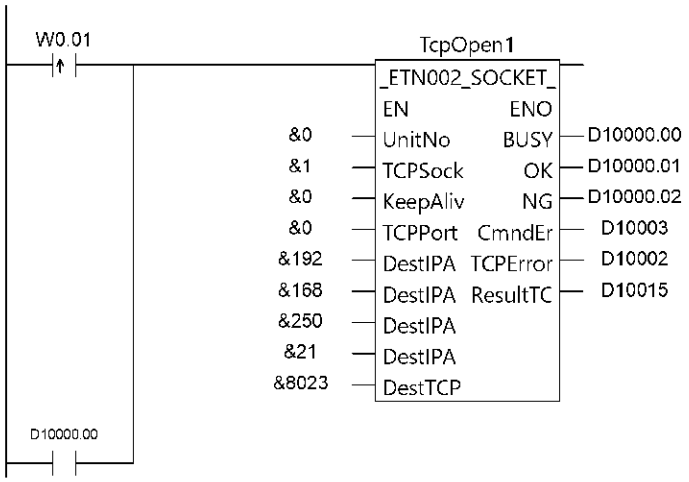


同様の回路を繰り返し、以下のようにメインプログラムに記載します。

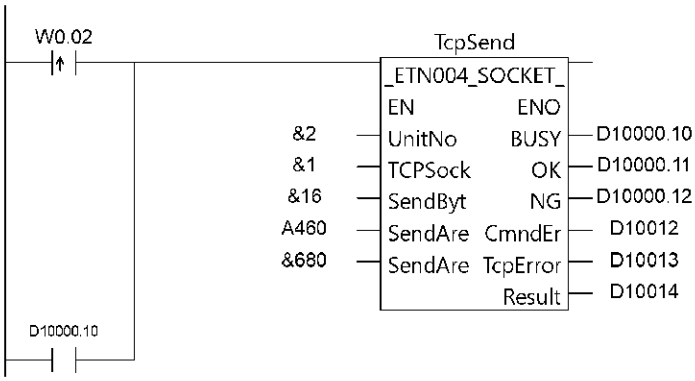
本図はＦＢＤ（ﾌｧﾝｸｼｮﾝﾌﾞﾛｯｸ）をラダー言語で記載したものですが、ＳＴ言語で記載も選択できます。（上記は三菱電機の例）
以上は三菱製ＰＬＣのＦＢＤの例ですが、以下にオムロン製も一例として記載しておき

ます。　弊社（筆者）もタイ向け負荷完成検査装置で使用した実績あり。

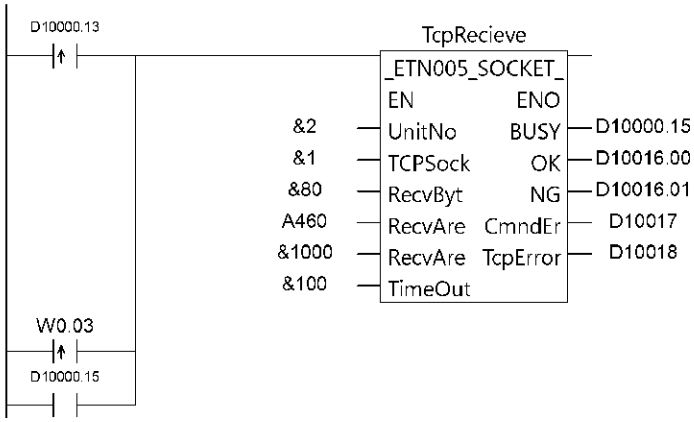
・ＥｔｈｅｒＮｅｔのＯＰＥＮ



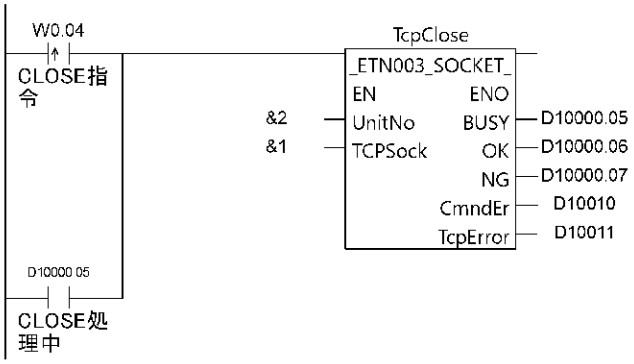
・ＥｔｈｅｒＮｅｔの送信



・ＥｔｈｅｒＮｅｔの受信



・ＥｔｈｅｒＮｅｔのＣＬＯＳＥ



このＦＢＤ（ﾌｧﾝｸｼｮﾝﾌﾞﾛｯｸ）については、ＰＬＣメーカー各社において、ライブラリーを用意していますので、便利な機能があります。
以下、ライブラリーについて、少し紹介しておきます。

三菱電機ＰＬＣの場合

・MELSEC ユニット用ＦＢ　‥‥ＣＰＵ，アナログ入出力ユニット、カウンタユニット
位置決めユニット　等
・パートナー製品用ＦＢ　‥‥情報／ネットワーク、ビジョンセンサー、ＲＦＩＤ
バーコードリーダ、レーザ変位センサー
温度調節計　等

オムロンＰＬＣの場合

・便利ＦＢ　‥‥ＣＰＵ、カレンダ、データ比較変換　等
・ドライバＦＢ‥‥メモリカード、通信関係設定、ＥｔｈｅｒＮｅｔ、
フィールドバス、インバータ、サーボ、ＲＦＩＤ、視覚センサー
コードリーダ、スマートセンサー、変位センサー、DeviceNet
温度調節器　等

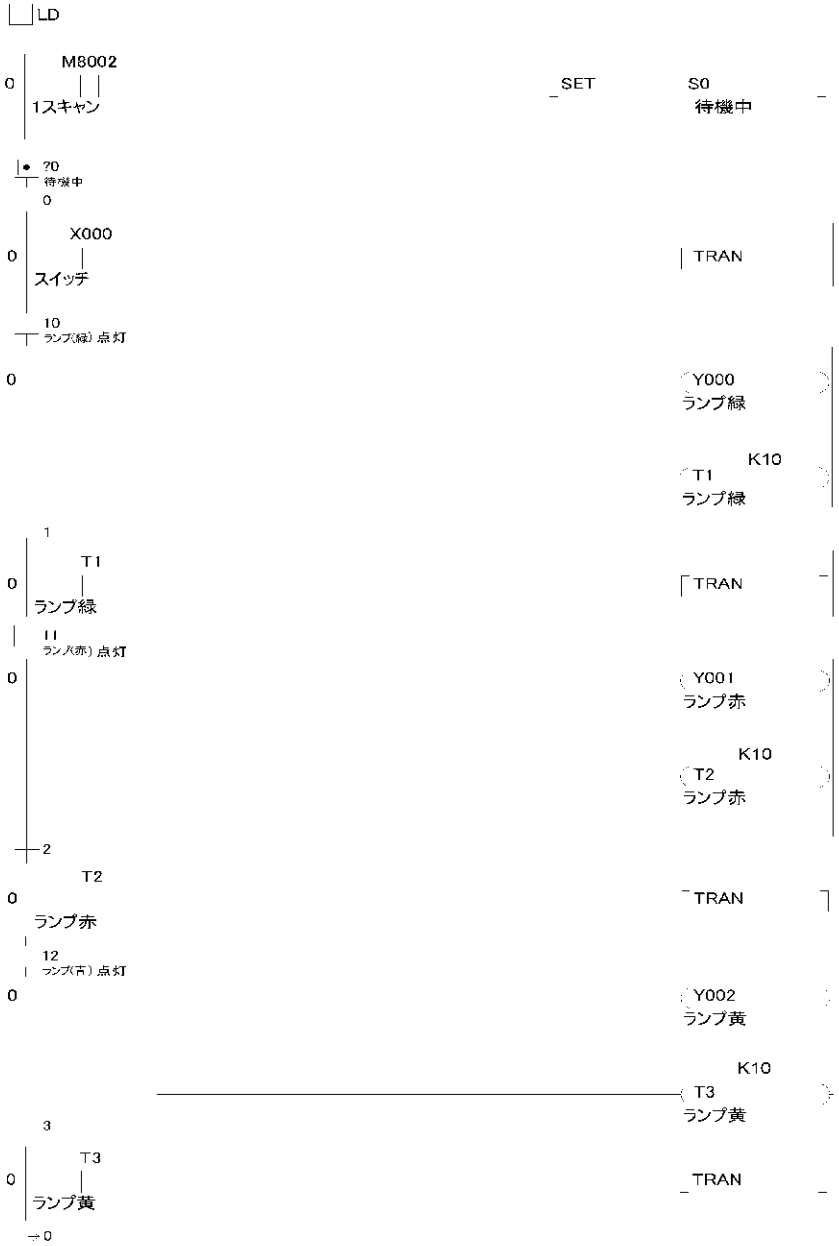
その他のＰＬＣメーカーも用意していますので､使用の際は一度調べてみてはと思います。
複雑なプログラムは断然、メーカー既存のライブラリーを使う方が楽です。

３－３．ＳＦＣ（シーケンシャルファンクションチャート）

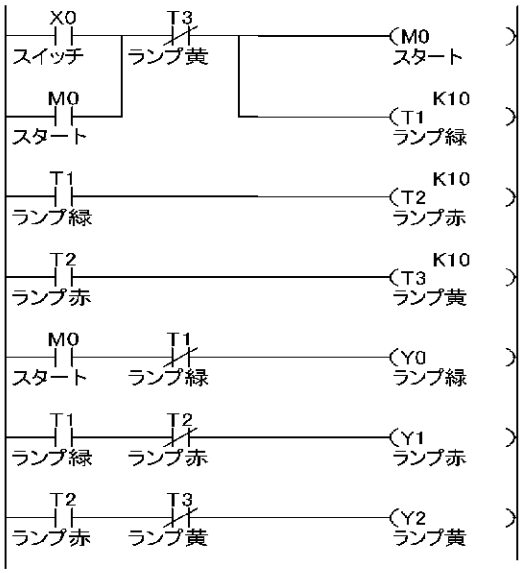
ＳＦＣ言語は工程歩進（ステップ）のようにプログラムする言語です。
フローチャートのように記述して、システムの処理順序や状態等が理解しやすいというのが利点ですが、演算機能や入出力機能がないので他の言語（ラダーやＳＴ等）で中身を記述しないといけません。

以下の例を考えてみます。

待機中、ランプ(緑)が１秒周期で点滅する。(ON：0.5 秒、OFF：0.5 秒)
スイッチを押すと、
ランプ(緑)点灯 ⇒ ランプ(赤)点灯 ⇒ ランプ(黄)点灯 ⇒ 戻る（全て消灯）
各々の点灯時間は１秒間とする。
〔　スイッチ：X0、ランプ(緑)：Y0、ランプ(赤)：Y1、ランプ(黄)：Y2　〕



これをＬＤ(ラダー)で現わせば、以下となる。



ラダーを主としている人にとっては、このＳＦＣは馴染みが薄いかも知れません。

３－４．ＩＬ（インストラクションリスト）

ＩＬ（インストラクションリスト）については、ニーモニックリストやコーディング等と呼ばれていて、ＰＬＣの初期には、このリストを命令として書き込んでいました。しかし、機械に指令するには良いリストなのですが、人間には分かり辛いため、ラダーを記載して変換すれば、このリストに自動で変わることが出来るようになり、今では、このリストを書き込むことも出来なくなったツールもあります。

以下は三菱電機ＰＬＣの例です（プログラム照合をした時にお目にかかる画面です）

結果一覧　詳細結果[1]

行	ステップ	照合元				ステップ	照合先			
1	0	問題 アナログを扱う				0	問題 アナログを扱う			
2	0	LD	X040			0	LD	X040		
3	1	OR	M0			1	OR	M0		
4	2	ANI	X041			2	ANI	X041		
5	3	ANI	T0			3	ANI	T0		
6	4	OUT	M0			4	OUT	M0		
7	5	MUL	D8030	K10	D10	5	MUL	D8030	K10	D10
8	12	OUT	T0	D10		12	OUT	T0	D10	
9	15	LD	M0			15	LD	M0		
10	16	OUT	Y040			16	OUT	Y040		
11						17	LD<=	D101	D200	
12						22	AND<=	D200	D100	
13						27	OUT	Y041		
14	17	LD	M0			28	LD	M0		
15	18	MUL	K4000	D8031	D2	29	MUL	K4000	D8031	D2
16	25	DDIV	D2	K255	D4	36	DDIV	D2	K255	D4
17	38	MOV	D4	D0		49	MOV	D4	D0	
18	43	LDI	M0			54	LDI	M0		
19	44	MOV	K0	D0		55	MOV	K0	D0	
20						60	LD	M0		
21						61	DIV	D0	K4	D200
22	49	END				68	END			

次に示すのはオムロンＰＬＣの例です。
オムロンでは、ラダーとニーモニックの表示が切り換えできるようになっています。

102	545	LD	51.12
	546	LD	71.04
	547	AND	71.08
	548	ORLD	
	549	MOV(021)	D190
			D354
	550	BIN(023)	D195
			D358
103	551	LD	71.00
	552	ANDNOT	71.07
	553	MOV(021)	D191
			D354
	554	BIN(023)	D196
			D358
	555	OUT	H2.00
104	556	LDNOT	71.04
	557	AND	71.08
	558	MOV(021)	D192
			D354
	559	BIN(023)	D195
			D358
105	560	LD	P_Off
	561	MOV(021)	D193
			D354
	562	BIN(023)	D197
			D358
106	563	LD	51.12
	564	LD	72.04
	565	AND	72.08
	566	ORLD	
	567	MOV(021)	D190
			D355
	568	BIN(023)	D195
			D359

３－５．ＳＴ（ストラクチャードテキスト）

ＳＴ言語の“ＳＴ”は“Structured Text（ストラクチャードテキスト）”の略で、日本語では「構造化テキスト」とも呼ばれます。
プログラミング言語のＰａｓｃａｌベースの高級言語で、ＩＥＣの標準に基づいているため、異なるメーカーのＰＬＣでのソフト互換があります。
データ処理や演算等が得意で、ラダーが苦手としている処理には優れています。
ただし、各メーカー独自で追加した関数（命令）もあるので、注意が必要です。

ＳＴ言語の例を以下説明します。

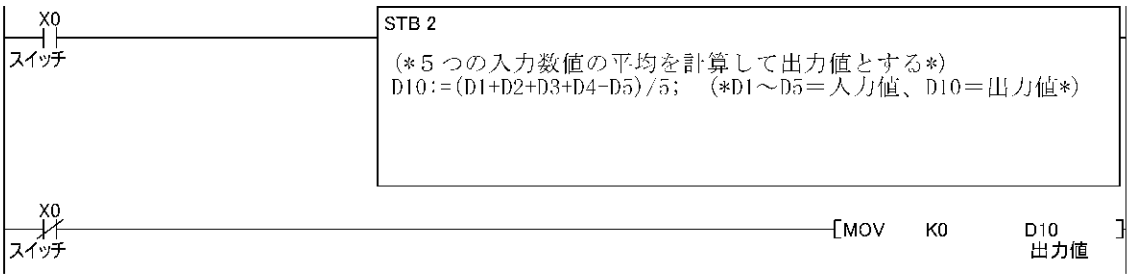
スイッチ(X0)をＯＮにすると、５つの入力数値の平均を計算して出力値とする。 入力値データ D1～D5、出力値データ D10 とします。

ＳＴ言語のみのプログラムの場合

```
(*スイッチ(X0)をＯＮにすると、５つの入力数値の平均を計算して出力値とする*)
IF X0 THEN
D10:=(D1+D2+D3+D4+D5)/5;  (*D1～D5＝入力値、D10＝出力値*)
ELSE D10:=0;
END_IF;
```

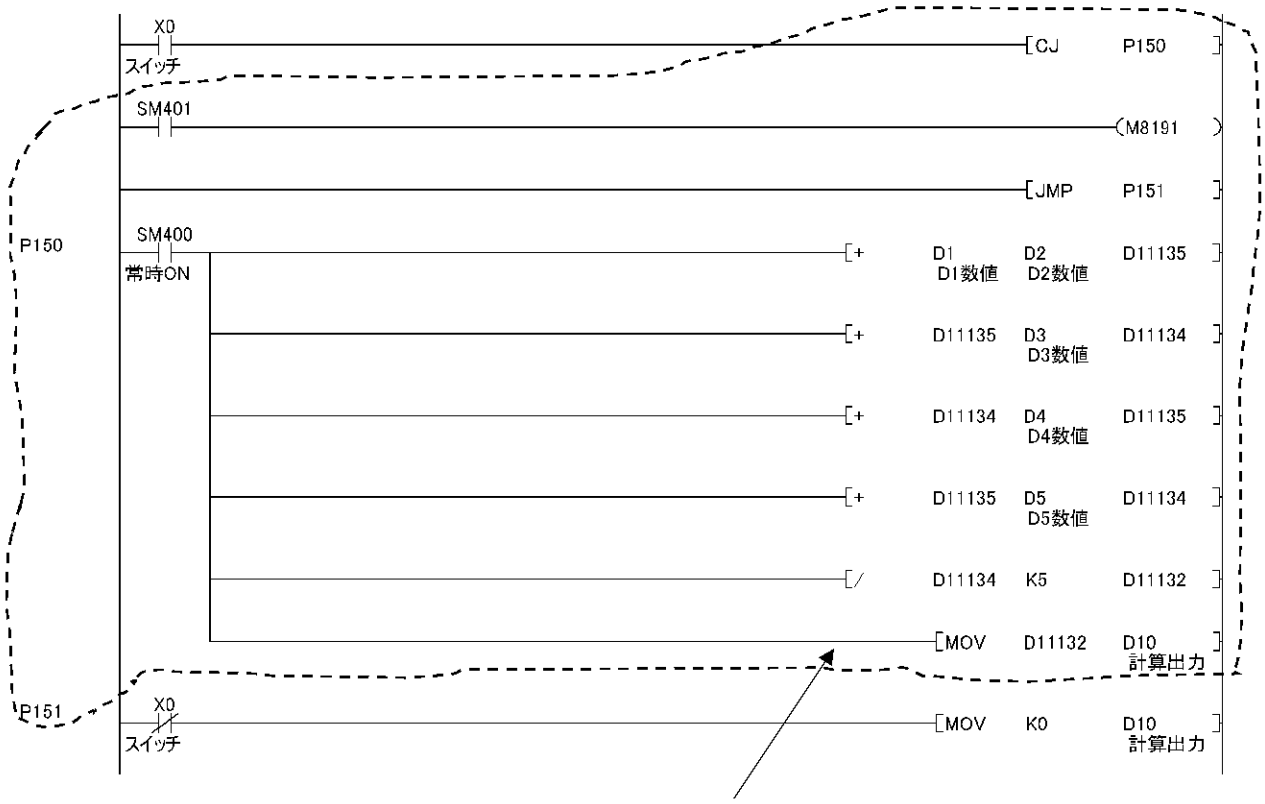
上記は三菱電機のＰＬＣの場合で、内部メモリ D1 等はそのまま使用しています。
ラベルにしなくても使用できます。(他メーカーではラベルしか使用できないものもあります)

ラダー言語のプログラム中に「インラインＳＴ」として組み込む場合



「インラインＳＴ」の場合、ラダーとＳＴを２つの画面で開かなくて良いし、一括でモニターも出来る利点があります。

尚、上記は「ラベル表示で印刷する」で印刷したものですが、「デバイス表示で印刷する」にした場合は、次のように、ラダーに変換されて印刷されます。



「デバイス表示で印刷する」を選択した場合（内部のメモリアドレスは自動で割付けられている）

「ＳＴ言語とは、どういう言語なのか」を少し説明しておきます。

ＳＴ言語のルール

・代入文は　【:=】　　（イコールの前にコロン）
・文の終わり【;】　　（セミコロン）
・四則演算は、【*】乗、【/】除、【+】加、【-】減　で表す
・比較は、　　【<】、【>】、【<=】、【>=】、【=】、【<>】
・論理演算　　【AND】【&】、【OR】、【NOT】、【XOR】
・コメント　【(*・・・・*)】　(*コメント*)　（または【//】オムロンはダメ）

例)
```
A:=B+C; (*B と C の和を A に入れる*)
```

ＳＴ言語の制御文

・条件式　IF　⇒　IF, THEN, ELSE, END_IF

例)
```
IF A>0 THEN X:=10;
       ELSE X:=0;
END_IF;
```

A>0 の時 X=10 を
それ以外の時 X=0 を代入

・整数式の値によって選択　CASE　⇒　CASE, ELSE, END_CASE

例)
```
CASE A OF
  1: X:=1;
  2: X:=2;
  ELSE X:=0;
END_CASE;
```

A=1 の時 X=1 を
A=2 の時 X=2 を
それ以外の時 X=0 を代入

・繰り返し文　**FOR**　⇒　FOR, TO, (BY), DO, END_FOR

例)
```
FOR i:=0 TO 100 BY 10 DO
  DATA[i]:=0;
END_FOR;
```

i=0～100 まで 10 飛ばしで
DATA[i]に 0 を代入

・条件内で繰り返し　**WHILE**　⇒　WHILE, DO, END_WHILE

例)
```
A:=0;
WHILE A<1000 DO
  A:=A+7;
END_WHILE;
```

７の倍数で 1000 を超える数を
算出して A に代入する

など、ほんの一部です。
Ｃ言語やＢＡＳＩＣなど経験のある方は、どこかで見たようだと思われたでしょう。

以上がＰＬＣの言語説明ですが、タッチパネルスクリプトについて以下説明します。

３－６．タッチパネルスクリプト

タッチパネルはＰＬＣではありませんが、ＰＬＣと合わせて使用されることが多いので取り上げました。
タッチパネルにも独自のプログラムがあり、独自の制御ができる機能があります。
三菱電機のタッチパネルＧＯＴの機能名を「スクリプト」(簡易なプログラムをスクリプトと呼んでいます）と呼んでいますが、オムロンＰＴや他メーカーでは「マクロ」と呼んだりしています。本検討書では、スクリプトと記しています。

装置には、操作と表示が必ず必要です。
以前は、操作スイッチと表示灯を機能ごとに別々に装備し、ＰＬＣとスイッチや表　示灯間を配線していました。
タッチパネルの普及で、価格も安くなってきましたし、何よりスイッチや表示灯が不要となりＰＬＣとの配線が不要となり、操作と表示が一体となり、装置も小型化が期待できることもあって、タッチパネルを装備する装置が多くなってきました。

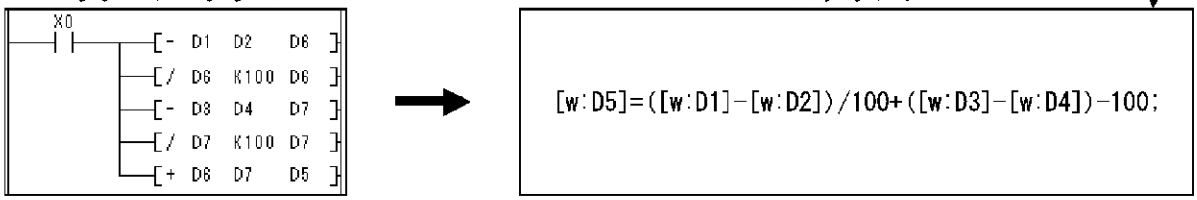
装置全体で考えれば、ＰＬＣであろうがタッチパネルであろうが、プログラムの所在は適切な制御が出来ているのであれば、問題ありません。
ただし、処理スピードに問題があったり、プログラムソフトが面倒だったり、デバッグし難かったり、何か不利益があるのであれば、このタッチパネルスクリプトを無理にでも使用することは勧めません。
使い方によっては、タッチパネルでのスクリプトの使用によって、ＰＬＣ側での表示に関するプログラムの負荷を大幅に軽減できるという利点もあります。

このスクリプトのプログラムは、Ｃ言語に似た言語型プログラムで、前述のＳＴ言語に似た機能を有しています。

タッチパネルスクリプトの例を示します。　　三菱電機ＧＯＴのスクリプトの例

ラダープログラムでは表現しにくかった多項式演算を1行でシンプルに表現する。

ラダープログラム　　**スクリプト**



ただし、このタッチパネルのスクリプト（マクロ）は、ＳＴ言語の様にＩＥＣによる標準化がなされていないので、メーカーによって命令の違いが多少あり、注意が必要です。

タッチパネル「スクリプト」とは、どういう言語なのか」を説明しておきます。
ＳＴ言語に近いので、ＳＴ言語と同じ項目で内容を説明します。
また、三菱電機の「スクリプト」とオムロンの「マクロ」も同時に説明して、その違いも記載します。

	三菱電機 GOT スクリプト	オムロン　マクロ
デバイス (ｱﾄﾞﾚｽ)	ビット[b:GB200]　ワード[w:GD200] など　[デバイス]の形式 PLC のデバイス X（入力）,Y（出力） M（内部）,L（ラッチ） D（データ）,W(リンク) R（ファイル）など GOT のデバイス GB(ビット),GD(ワード) も使用可能 ラベル名でも扱えます [<$:グループ名:ラベル名>]	ビット$B0　ワード$W0　など ホストのアドレス、タグは［ ］で囲って記述します。 通信用の関数(READCMEM,WRITECMEM)で使用します。 例：READCMEM ($W100,[SerialA:DM0],100); ’ SerialA:DM0～SerialA:DM9 を$W100～ $W199 に読み出し
単文の終り	【;】　(セミコロン)	【;】　(セミコロン)
四則演算	【*】乗、【/】除、【+】加、【-】減	【*】乗、【/】除、【+】加、【-】減
比較	【<】、【>】、【<=】、【>=】、【==】、【!=】	【<】、【>】、【<=】、【>=】、【==】、【!=】 【<>】も OK
論理演算	【&&】論理積、【||】論理和、【!】否定	【&&】【AND】論理積 【||】【OR】論理和、【<>】【!=】否定
コメント	【//】　(スラッシュ２本)	【’】　(クオーテーション)
加算	[w:D0]=[w:D1]+[w:D2]; //D1 と D2 の和を D0 に入れる	$W0=$W1+$W2; ’W1 と W2 の和を W0 に入れる
乗除	[w:D0]=([w:D1]*[w:D2])/[w:D3]; //D1 と D2 の積を D3 で割り D0 に	$W0=($W1*$W2)/$W3; ’W1 と W2 の積を W3 で割り W0 に入れる

スクリプトの制御文

制御文	三菱電機 GOT スクリプト	オムロン　マクロ

条件式
if
	if() {}else{}　なぜか if 小文字です
	
	if ([w:D0]>0) {[w:D10]=10;}
	else {[w:D10]=0;}
	
	//D0>0 なら D10=10 を
	それ以外は D10=0 を代入
		IF　ELSEIF　ELSE　ENDIF
		必ず最後は ENDIF　大／小文字 OK
		IF ($W0>0) $W10=10;
		ELSE $W10=0;
		ENDIF
		’ W0>0 なら W10=10 を
		それ以外は W10=0 を代入

整数式の値によって選択
case
	switch() {case: break; default:}
	
	switch([w:D0])
	{
	case 1: [w:D10]=1; break;
	//D0=1 の時 D10=1 を
	case 2: [w:D10]=2; break;
	//D0=2 の時 D10=2 を
	default :[w:D10]=0; //
	それ以外の時 D10=0
	}
		なし
		IF 文で代用できる

繰り返し文
for
	なし	FOR (BREAK) (CONTINUE) NEXT
		
		FOR(5)　‘5 回繰返し
		$W100=$W100+1;
		NEXT

条件内繰り返し
while
	while() {}
	
	[w:GD1]=0;
	while ([w:GD1]<1000)
	{[w:GD1]=[w:GD1]+7;}
	//７の倍数で 1000 を超える数を
	算出して GD1 に代入する
		なし

	三菱電機 GOT スクリプト	オムロン　マクロ
その他	プロジェクト全体のスクリプトと 対象画面ごとのスクリプトおよび 各部品（オブジェクト）のスクリプトが有る 応用演算関数（BCD,BIN や sin,cos 等）有 ファイル、文字列の関数有り <u>ビット操作演算子として</u>； set([b:Y10])；　ビットのセット [b:Y10]=1；又は[b:Y10]=ON;とも記載可能 rst([b:Y10])；　ビットのリセット [b:Y10]=0；又は[b:Y10]=OFF;とも記載可能 alt([b:Y10])；　ビットの反転 <u>連続操作演算子として</u>； bmov([w:D10],[w:D20],10)； ブロック転送；ラダーの BMOV と同じ fmov([w:D10],[w:D20],10)； 同一データブロック転送 ラダーの FMOV と同じ <u>GOT で周辺機器を使用する機器</u>として メーカーマニュアルに紹介有り（以下） GOTでバーコードリーダを使用する(バーコード機能) GOTでRFIDを使用する(RFID機能) GOTからパソコンを操作する(パソコンリモート操作機能(Ethernet)) GOTからパソコンを操作する(パソコンリモート操作機能(シリアル)) パソコンからGOTを閲覧する(VNCサーバ機能) ビデオカメラなどからの映像をGOTに表示する(ビデオ表示機能) パソコンなどからの映像をGOTに表示する(RGB表示機能) ビデオカメラなどからの映像をGOTで録画，再生する(マルチメディア機能) GOTで外部入出力機器を使用する(外部入出力機能，操作パネル機能) GOTの画面を外部ディスプレイに出力する(RGB出力機能) 収集したデータやアラームをプリンタに出力する(レポート機能) GOTで音声を出力する(音声出力機能) GOTがモニタしている接続機器のデバイスを他のパソコン，GOTからモニタする(サーバ，クライアント機能) GOTからメールを送信する(メール送信機能) GOT(FTPサーバ)と周辺機器(FTPクライアント)間でファイルを転送する(FTPサーバ機能) GOT(FTPクライアント)と周辺機器(FTPサーバ)間でファイルを転送する(ファイル転送機能(FTP転送)) GOTのドライブ間でファイルを転送する(ファイル転送機能(GOT内部転送)) タブレットなどから接続機器をモニタする(GOT Mobile機能)	プロジェクト全体のマクロと （画面のマクロ設定） 機能部品(ON-OFFﾎﾞﾀﾝ等)ごとﾏｸﾛ設定 応用演算関数(BCD,BIN や SIN,COS 等)有 ファイル、文字列の関数有り <u>ビット操作演算子として</u>； BITSET（D, c, n）　有 [D=PTﾒﾓﾘ(B/HB),c=0/1,n=数] BITSET($B0,1,1)；　ビットのセット BITSET($B0,0,1)；　ビットのリセット PLCﾒﾓﾘ(ﾎｽﾄｱﾄﾞﾚｽ)の場合は WRITEHOSTB（ﾎｽﾄｱﾄﾞﾚｽ書込）等有 <u>連続操作演算子として</u>； MEMCOPY($W10,$W20,10)； データコピー MEMSET($W10,$W20,10)； データ一括セット

４．言語比較例

実際どのようなプログラムかを、例を示して言語比較を行う事にします。

比較する言語として、ＬＤ（ラダー）、ＳＴ言語、タッチパネルスクリプト（GOT）とします。

記）タッチパネルはＰＬＣではないが、ＰＬＣと合わせて使用されることが多いので取り上げた。
タッチパネルにおいても、ＳＴ言語のような高級言語が使用できる。

比較例として、以下の参考例をあげてみます。

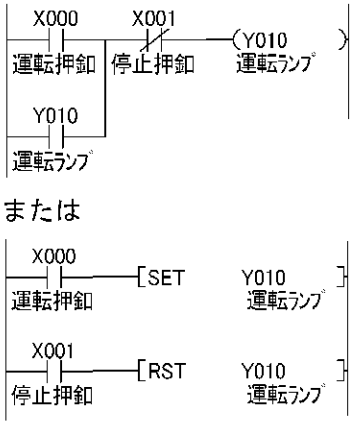
４－１．参考例１

例内容　（ビットのＯＮ／ＯＦＦ）

運転押釦(X0)を押すと(X0:OFF→ON)、ランプ(Y10)が点灯(Y10:ON)し保持する。

停止押釦(X1)を押すと(X1:OFF→ON)、ランプ(Y10)が消灯（Y10:OFF）する。

ＬＤ（ラダー）	ＳＴ言語	スクリプト(GOT)

	Y10:=(LDP(TRUE,X0) OR Y10) AND NOT X1;
	(*論理回路　ﾗﾀﾞｰと同じ回路*)
	
	または
	IF X0 THEN Y10:=TRUE;　END_IF;
	IF X1 THEN Y10:=FALSE; END_IF;
	(*条件文回路*)
	
	記；(*コメント*)
		if ([b:X0]==1) {set([b:Y10]);}
		//X0がONでY10をON
		if ([b:X1]==1) {rst([b:Y10]);}
		//X1がONでY10をOFF
		
		記；// 以降はコメント

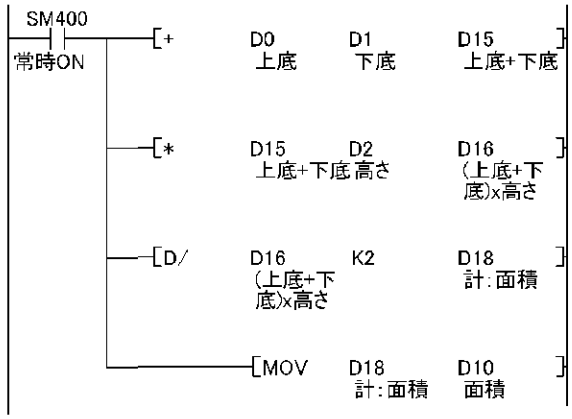
この例では、各言語とも余り差はありません。　　全て動作確認しています。

４－２．参考例２

例内容　（台形の面積）

上底(D0)、下底(D1)、高さ(D2)の台形の面積(D10)を求めます。

【(上底＋下底) x 高さ÷２＝面積】

ＬＤ（ラダー）	ＳＴ言語	スクリプト(GOT)
	(*台形の面積*) D10:=(D0+D1)*D2/2; 記；(*コメント*)	//台形の面積 [w:D10]=([w:D0]+[w:D1])*[w:D2]/2; 記；// 以降はコメント

この例では、ラダー言語の可読性が劣るように見えます。　　全て動作確認しています。

４－３．参考例３

例内容　パワーメータに送るコマンドの文字列をセットする。

コマンド CONF:VOLT:RANG 300V ←　300V レンジに設定するコマンド D100～

MEAS:NORM:VAL?　←　測定値を返すコマンド　D150～

ＬＤ（ラダー）	ＳＴ言語	スクリプト(GOT)
SM400 常時ON [DMOV H464E4F43 D100] [DMOV H4C4F563A D102] [DMOV H41523A54 D104] [DMOV H3320474E D106] [DMOV H563030 D108] [DMOV H5341454D D150] [DMOV H524F4E3A D152] [DMOV H4156204D D154] [DMOV H3F4C D156]	(*パワーメータのコマンド*) CONF='CONF:VOLT:RANG 300V'; MEAS='MEAS:NORM:VAL?';	[u64:D100]=0x464E4F43LL; [u64:D102]=0x4C4F563ALL; [u64:D104]=0x41523A54LL; [u64:D106]=0x3320474ELL; [u64:D108]=0x00563030LL; [u64:D150]=0x5341454DLL; [u64:D152]=0x524F4E3ALL; [u64:D154]=0x4156204DLL; [u64:D156]=0x00003F4CLL;
上記は ASCII コードを入力するやり方です。 最近は次の命令で楽になっています しかし、この命令は三菱Ｑシリーズのもの ＦＸには無し。オムロンにも無し	記；(*コメント*) CONF,MEAS はラベルとして登録	スクリプトには文字入力に関するコマンドが見当たりませんでした。 ラダー同様に ASCII コードを入力するやり方としました。
SM400 常時ON [$MOV "CONF:VOLT:RANG 300V" D100] [$MOV "MEAS:NORM VAL?" D150]		

この例では、ＳＴ言語が優れているようです。　　全て動作確認しています。

４－４．参考例４

例内容　（パワーメータより送信で読み取ったデータをＢＣＤにする）

読み取ったデータは、ASCII の浮動小数点表記で、Ｖ（電圧），Ａ（電流），Ｗ（電力）が区切り（デリミタ）として「カンマ」が入っています。

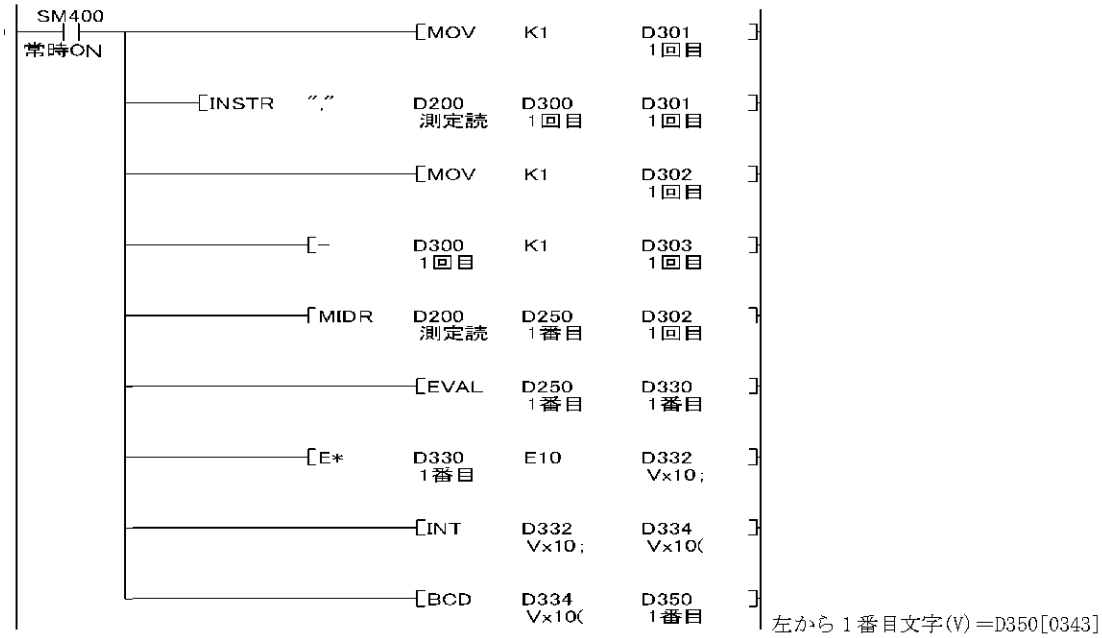
これを１０倍した値をＢＣＤに変換して、Ｖ，Ａ，Ｗ値とします。

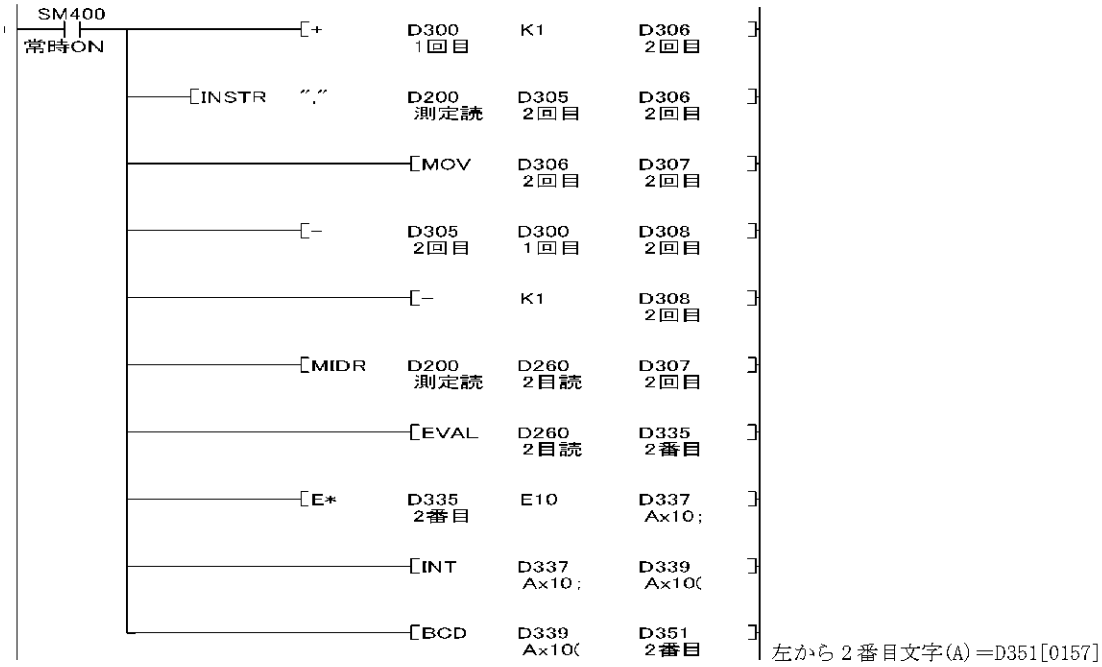
読込データは、例として【34.3459E+00,15.6789E+00,85.43210E+00】の ASCII 文字データを読んで、×１０して、ＢＣＤ値でＶ＝【0343】、Ａ＝【0156】、Ｗ＝【0854】に変換します。

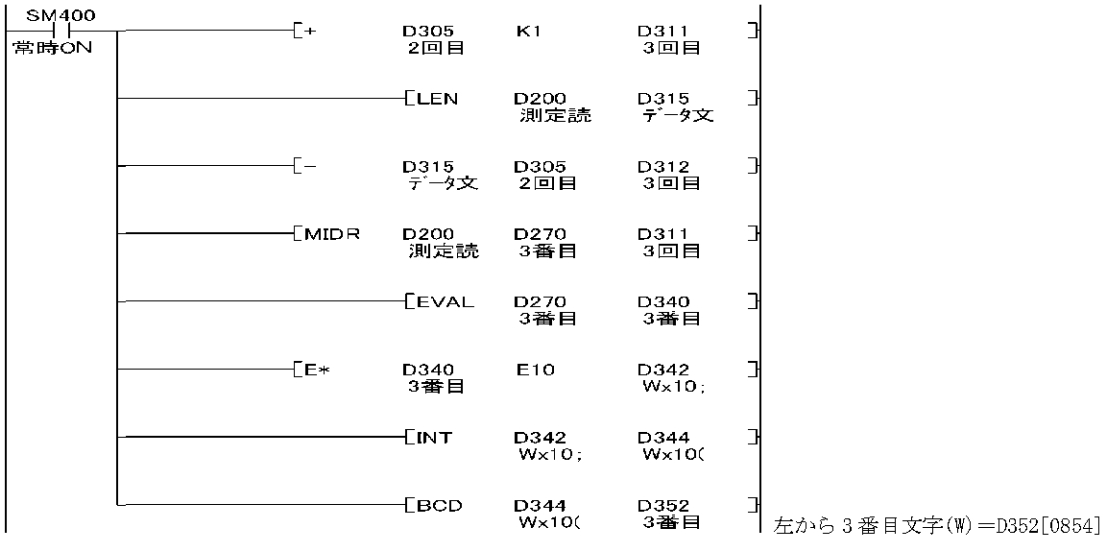
読み込んだ後の処理のみです。尚、文字列読込データアドレスは D200～とします。

ＢＣＤ値のＶを D350、Ａを D351、Ｗを D352 に入れます。

ＬＤ（ラダー）


SM400
常時ON
MOV K1 D301 1回目
INSTR "." D200 測定読 D300 1回目 D301 1回目
MOV K1 D302 1回目
- D300 1回目 K1 D303 1回目
MIDR D200 測定読 D250 1番目 D302 1回目
EVAL D250 1番目 D330 1番目
E* D330 1番目 E10 D332 Vx10;
INT D332 Vx10; D334 Vx10(
BCD D334 Vx10(D350 1番目
左から１番目文字(V)＝D350[0343]


SM400
常時ON
+ D300 1回目 K1 D306 2回目
INSTR "." D200 測定読 D305 2回目 D306 2回目
MOV D306 2回目 D307 2回目
- D305 2回目 D300 1回目 D308 2回目
- K1 D308 2回目
MIDR D200 測定読 D260 2目読 D307 2回目
EVAL D260 2目読 D335 2番目
E* D335 2番目 E10 D337 Ax10;
INT D337 Ax10; D339 Ax10(
BCD D339 Ax10(D351 2番目
左から２番目文字(A)＝D351[0157]


SM400
常時ON
+ D305 2回目 K1 D311 3回目
LEN D200 測定読 D315 ﾃﾞｰﾀ文
- D315 ﾃﾞｰﾀ文 D305 2回目 D312 3回目
MIDR D200 測定読 D270 3番目 D311 3回目
EVAL D270 3番目 D340 3番目
E* D340 3番目 E10 D342 Wx10;
INT D342 Wx10; D344 Wx10(
BCD D344 Wx10(D352 3番目
左から３番目文字(W)＝D352[0854]

ＳＴ言語

```
(*PM:='34.3459E+00,15.6789E+00,85.43210E+00';ﾃｽﾄﾃﾞｰﾀ*)
(*ﾊﾟﾜｰﾒｰﾀから読み取ったASCII文字からV,A,Wの文字に変換*)
IF  PMTRG THEN ;                    (*読込ﾄﾘｶﾞ*)

CT1:= FIND(PM,',') ;                (*左からの","の位置*)
Va:=MID(PM,CT1-1,1);                (*読取値の左から","までの文字*)
PM:=REPLACE(PM,' ',1,CT1);  (*読んだら元ﾃﾞｰﾀの","を" "としておく*)

CT2:= FIND(PM,',');                 (*左からの","の位置*)
Aa:=MID(PM,CT2-CT1-1,CT1+1);    (*読取値のその次から","までの文字*)
PM:=REPLACE(PM,' ',1,CT2);  (*読んだら元ﾃﾞｰﾀの","を" "としておく*)

Wa:=RIGHT(PM,LEN(PM)-CT2); (*最右の読込値*)

END_IF;
```

名称	ﾃﾞｰﾀ型	アドレス	コメント
Wa	STRING[20]	D5840	W読込文字
Va	STRING[20]	D5800	V読込文字
PMTRG	BOOL	W93.00	読込ﾀｲﾐﾝｸﾞ
PM	STRING[100]	D5700	ﾊﾟﾜｰﾒｰﾀ読込文字列
CT2	INT		
CT1	INT		
Aa	STRING[20]	D5820	A読込文字

上記は、オムロンのＣＪ２Ｍのものである。

三菱のＱシリーズには文字列抽出等命令無し（Ｌシリーズはある）

スクリプト（GOT）

```
[s16:GD200] = str_strlen([w:D200],100);

[s16:GD201] = str_scanf([w:D200],[s16:GD200],0,"%4s %4s
%4s",[w:D350],[w:D360],[w:D370]);
```

上記は、三菱 GOT でのスクリプト例です。

まだ完成していませんが、C 言語の知識が必要のため途中までとします。

この例では、文字列に関するプログラムなので、ラダーは不得意です。
しかし、最近のラダーは命令語が豊富となり文字列関係も充実しています。
タッチパネル（GOT）では各メーカーのスクリプト（マクロ）関数の保有機能によって、若干、不利・不向きと思われます。

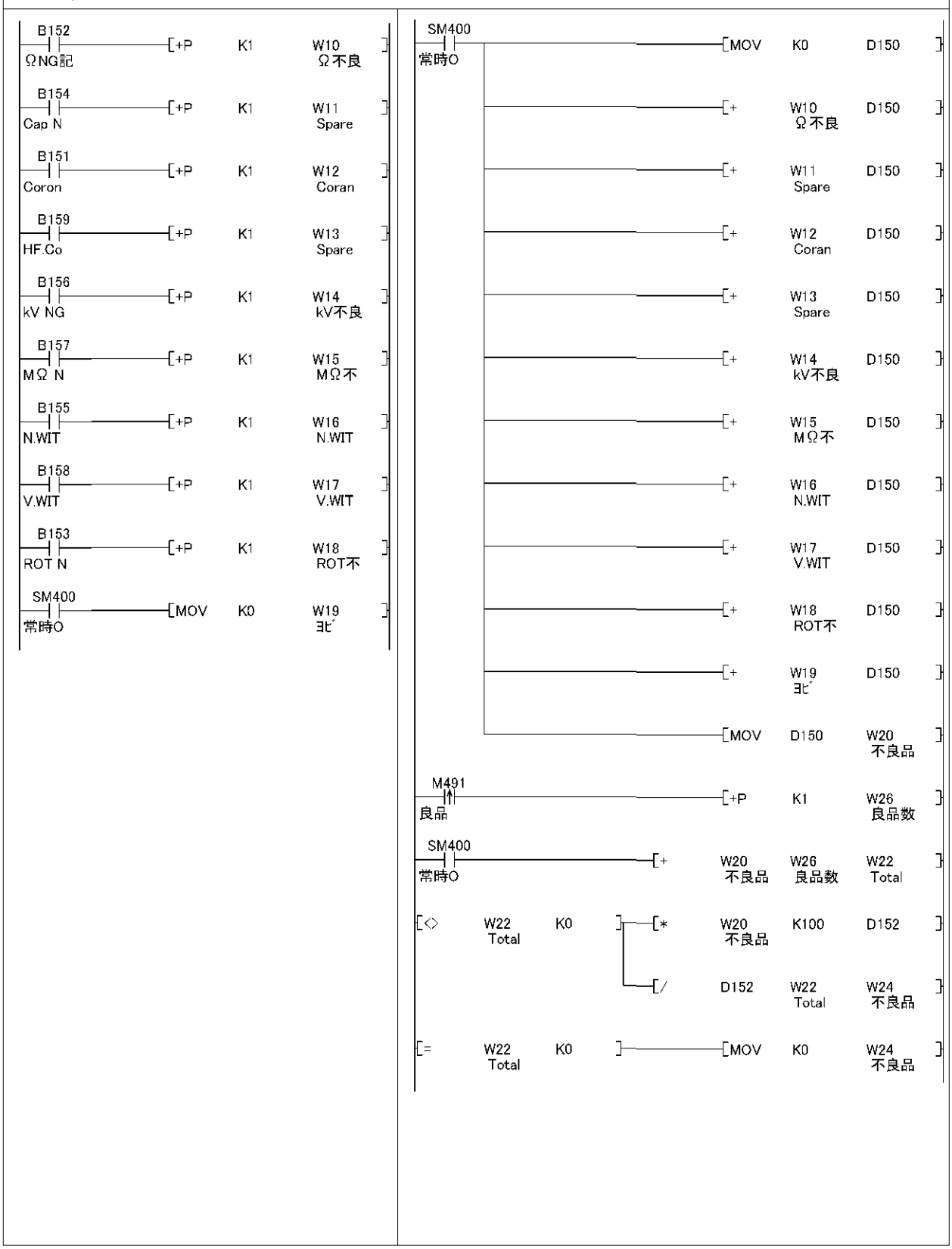
４－５．参考例５

例内容　（良品数・不良品数の演算表示）

検査した結果として、GOT 画面に判定した良品数、不良品数、不良率、総数を演算して画面に表示するプログラムを書いてみます。

これは、実際の巻線検査装置４ｓｔのプログラムを例としています。

ＬＤ（ラダー）



ＳＴ言語

```
(*検査結果の良不良数をカウントする*)

INCP(B152,W10); (*B152のONパルスでNG0(W10)が+1*)
INCP(B154,W11); (*B154のONパルスでNG1(W11)が+1*)
INCP(B151,W12); (*B151のONパルスでNG2(W12)が+1*)
INCP(B159,W13); (*B159のONパルスでNG3(W13)が+1*)
INCP(B156,W14); (*B156のONパルスでNG4(W14)が+1*)
INCP(B157,W15); (*B157のONパルスでNG5(W15)が+1*)
INCP(B155,W16); (*B155のONパルスでMG6(W16)が+1*)
INCP(B158,W17); (*B158のONパルスでNG7(W17)が+1*)
INCP(B153,W18); (*B153のONパルスでNG8(W18)が+1*)
W19:=0; (*常時NG9(W19)=0*)
W20:=W10+W11+W12+W13+W14+W15+W16+W17+W18+W19; (*不良品計*)

INCP(M491,W26); (*M491のONパルスでOK(W26)が+1     良品数 *)

W22:=W20+W26 ; (*合計数*)

IF W22<>0 THEN W24:=100*W20/W22; (*不良品率%  分母が0ならｴﾗｰになる*)
          ELSE W24:=0;
END_IF;
```

スクリプト (GOT)

トリガ種別

```
立上り(B00152) ──→ [w:W10]=[w:W10]+1; //W10=Ω不良数
立上り(B00154) ──→ [w:W11]=[w:W11]+1; //W11=予備不良数
立上り(B00151) ──→ [w:W12]=[w:W12]+1; //W12=COR不良数
立上り(B00159) ──→ [w:W13]=[w:W13]+1; //W13=予備不良数
立上り(B00156) ──→ [w:W14]=[w:W14]+1; //W14=kV不良数
立上り(B00157) ──→ [w:W15]=[w:W15]+1; //W15=MΩ不良数
立上り(B00155) ──→ [w:W16]=[w:W16]+1; //W16=N.WIT不良数
立上り(B00158) ──→ [w:W17]=[w:W17]+1; //W17=V.WIT不良数
立上り(B00153) ──→ [w:W18]=[w:W18]+1; //W18=ROT不良数
立上り(M491)   ──→ [w:W26]=[w:W26]+1; //W26=良品数

常時 ──────→ [w:W19]=0; //W19=予備不良数

              //W20=NG合計数
[w:W20]=[w:W10]+[w:W11]+[w:W12]+[w:W13]+[w:W14]+[w:W15]+[w:W16]+[w:W17]+[w:W18]+[w:W19];

[w:W22]=[w:W20]+[w:W26]; //良不良品合計

if ([w:W22] != 0 ) {[w:W24]=100*[w:W20]/[w:W22];}  //W22=不良品率  分母が0ならｴﾗｰ
else {[w:W24]=0;}
```

この例では、演算して画面に表示するだけなので、ＳＴ言語やスクリプト(GOT)の方が得意と思われます。

４－６．参考例６

例内容　（アナログ信号を扱う）

真空圧計のアナログ信号をＰＬＣに入力して、タッチパネル画面に真空値（Torr 単位とｋPa 単位）を表示させて、外部のアナログ電圧計にはkPa 単位の電圧値を出力するプログラムを考えてみます。詳細のスケールは以下の通りです。

ＰＬＣの入力アドレス　　；D500　0～760→(0～4000)

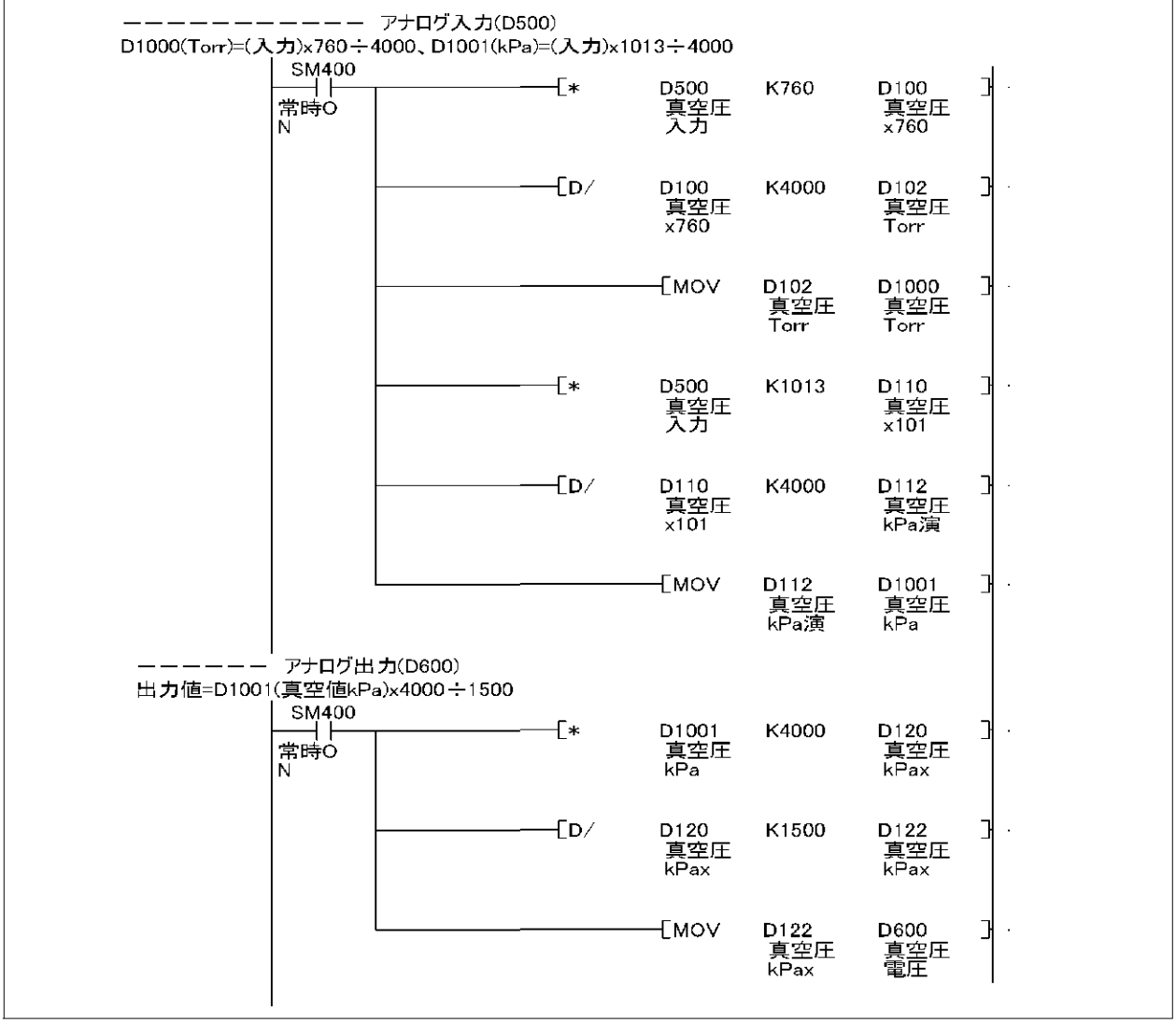
タッチパネルの Torr 表示アドレス；D1000　(0～760)

タッチパネルの kPa　表示アドレス；D1001　(0～1013)小数点以下は GOT 内の設定とする

電圧計への出力アドレス；D600　(0～4000→0～150)　　とし，

換算値は、1 atm＝ 101.3 kPa＝ 760 Torr とします。

ＬＤ（ラダー）



ＳＴ言語

```
(*アナログ入力  INP=D500*)
TOR:=INT_TO_DINT(INP)*760/4000 ; (* 真空圧TOR =D102   2ﾜｰﾄﾞ *)
Torr:=DINT_TO_INT(TOR) ;          (* 真空圧Torr =D1000   1ﾜｰﾄﾞ*)

KP:=INT_TO_DINT(INP)*1013/4000 ; (*真空圧KP=D112   2ﾜｰﾄﾞ*)
KPa:=DINT_TO_INT(KP) ;            (*真空圧KPa =D1001   1ﾜｰﾄﾞ*)

(*アナログ出力  OUT=D600*)
AOUT:=KP*4000/1500    ;(*電圧出力 *)
```

データ型が一致しないとエラーになるので、データ型を合わせるのが面倒です
以下が今回のデータ型の設定です。

	クラス	ラベル名	データ型		定数値	デバイス
1	VAR_GLOBAL	INP	ワード[符号付き]	...		D500
2	VAR_GLOBAL	TOR	ダブルワード[符号付き]	...		D102
3	VAR_GLOBAL	Torr	ワード[符号付き]	...		D1000
4	VAR_GLOBAL	KP	ダブルワード[符号付き]	...		D112
5	VAR_GLOBAL	KPa	ワード[符号付き]	...		D1001
6	VAR_GLOBAL	AOUT	ダブルワード[符号付き]	...		D600
7				...		

スクリプト(GOT)

```
//(入力)=D500、   真空圧Torr=D1000、   真空圧kPa=D1001
[w:D1000]=[w:D500] * 760 / 4000 ; //真空圧Torr=(入力)x760/4000
[w:D1001]=[w:D500] *1013 / 4000; //真空圧kPa=(入力)x1013/4000

//(出力)=D600
[w:D600]=[w:D1001] * 4000 / 1500 ; //出力値=真空圧kPax4000/1500
```

コメントを除けば、３行プログラムです。
データ型とかあまり気にしなくて良いみたい

この例でも、演算関係はＳＴ言語やスクリプト(GOT)の方が得意と思われます。

５．結論

結論としては、プログラムする人の得意な言語を使用するのが、ベストと考えます。
しかし、処理する内容によっては、別の言語によって随分簡素化した内容となり時間短縮が出来たり、あるいは、可読性が優れれば将来のプログラム変更時には短時間でバグなく変更が行えるので、やはり適材適所で使い分け出来ればと考えます。

以下は、JEMA（社団法人日本電機工業会）によって作成された５言語の比較表です。

プログラミング言語の得意・不得意

主な使用状況	LD言語	IL言語	ST言語	FBD言語	SFC言語
単純なリレーシーケンス処理	◎	×	△	△	×
数式演算処理	△	×	◎	○	×
状態せん移に基づく順序制御 (ステップシーケンス処理)	△	×	○	×	◎
連続的なアナログ信号処理	△	×	○	◎	×
複雑な情報処理	△	×	◎	△	×
プログラムメモリ制約の厳しい場合	○	◎	△	△	×
最も高速に性能を求められる場合	○	◎	○	○	×
運転方案と対応がとりやすい表現	×	×	○	○	◎
動作を視覚的に確認したい場合	○	×	×	◎	○
注記　記号の意味は，次による。 ◎：最も適している，○：適している，△：困難な場合もある，×：適さない					

弊社（筆者）で考えてみると（JAMA の比較と若干違います）
５言語に加えて「タッチパネルのスクリプト」も比較に入れています。

処理内容	ＬＤ言語	ＩＬ言語	ＳＴ言語	ＦＢＤ言語	ＳＦＣ言語	ｽｸﾘﾌﾟﾄ(GOT)
単純な リレーシーケンス	◎	×	△	△	×	×
数式演算	○ 可読性は良くないが最近の PLC は命令が豊富	×	◎ 高級言語の演算式で可読性良い	△	×	○ ＳＴ言語に近い GOT 画面内演算表示には良い
ステップシーケンス	◎ ラダーでもステップ式で書けば十分ＯＫ	×	○	×	◎ ＳＦＣに慣れていれば	×
繰り返し回路	○	×	○	◎ ブロック作成して使用	×	△

アナログ信号	○ パラメータ(I/Oﾃｰﾌﾞﾙ)設定でオフセット、ゲイン設定すれば演算は少なくてすむ	×	○ アナログにはスケール変換(演算)が付き物	○ データ変換等のFBDがPLCメーカで準備されていることが多い	×	○ GOT画面のみの使用ならＳＴ言語と同様に可能
情報（通信制御等）	△ 文字列が関係してくるので不得意	×	◎ 文字列が関係してくるので得意	○ メーカで準備されているFBDがあるなら得意	×	× 機能あるが、GOTではしない方が良い。PLCのソフトが混じって分かり難くなる
タッチパネルへの画面表示・演算	◎ PLCで処理したメモリ、データをタッチパネルに割付けて表示	×	△ 複雑な演算を画面表示させる時など	×	×	◎ GOT画面内だけで使用する時画面に無関係の処理はしないように。分かり難くなる。

注記　記号の意味　◎：最も適している、○：適している、△：困難な場合もある、×：適さない

総まとめとして、弊社ではＬＤ言語（ラダー）を以前からずっと使用してきました。
従って、ＬＤ（ラダー）を主としてプログラミングを行い、補助としてＳＴ言語を使用する。　得意とする演算・情報のみＳＴ言語として併用するのが良いと思います。
同じ回路が複数ある（繰り返し回路）場合には、ＦＢＤを用いるのも良いと思います。
ＬＤ（ラダー）とＳＴ言語、ＬＤ（ラダー）とＦＢＤとの組み合わせ使用も有効です。

タッチパネル画面だけの表示・演算があるなら、スクリプトを併用するのも良いです。

また、ＦＡ（パソコン）と制御を分け合っているような装置（制御器）においては、数値演算や情報関係はＦＡで行い、制御関係はＰＬＣで行って処理を分け合う方法となっています。　逆に言うと、ＦＡ処理の少ないような装置（制御器）では、ＦＡを止めてＰＬＣのＳＴ言語等を使用することで、ＦＡのコストを削減することも考えられます。

一度５言語を使用してみた上で、どれが最適言語かを理解して、今後のプログラム作成に役立ててもらえば幸いです。

以上

参考資料等

ＰＬＣ関係

三菱電機　シーケンサ

・ GX Works2 構造化プロジェクト編　SW1DNC-GXW2-J　108 頁
・ GX Works2 Version 1 オペレーティングマニュアル（ｼﾝﾌﾟﾙﾌﾟﾛｼﾞｪｸﾄ編）　SW1DNC-GXW2-J　358 頁
・ GX Works2 Version 1 オペレーティングマニュアル（構造化ﾌﾟﾛｼﾞｪｸﾄ編）　SW1DNC-GXW2-J　246 頁
・ GX Works2 Version 1 ｵﾍﾟﾚｰﾃｨﾝｸﾞﾏﾆｭｱﾙ（ｼﾝﾌﾟﾙﾌﾟﾛｼﾞｪｸﾄ･ﾌｧﾝｸｼｮﾝﾌﾞﾛｯｸ編）SW1DNC-GXW2-J　110 頁
・ GX Works3 オペレーティングマニュアル　SW1DNC-GXW3-J　946 頁
・ MELSOFT　GX-Works2　FB クイックスタートガイド

ｏｍｒｏｎ　プログラマブルコントローラ

・ CX-Programmer　Ver.9.□　CXONE-AL□□D-V4
　オペレーションマニュアル　ファンクションブロック編／ストラクチャードテキスト編　266 頁

ＳＩＥＭＥＮＳ

・ＳＩＭＯＴＩＯＮ　ＳＴ構造化テキスト　ﾌﾟﾛｸﾞﾗﾐﾝｸﾞ操作ﾏﾆｭｱﾙ　03/2007　346 頁

タッチパネル関係

三菱電機　ＧＯＴ

・グラフィックｵﾍﾟﾚｰｼｮﾝﾀｰﾐﾅﾙ　GT-Designer3(GOT2000)画面設計ﾏﾆｭｱﾙ　SW1DND-GTWK3-J　2790 頁

ｏｍｒｏｎ　ＰＴ

・プログラマブルターミナル　ＮＳシリーズ　マクロリファレンス　70 頁

その他インターネット検索

ホームページ　格式会社　ソフテック様　他