オープンＣＮＣマシニングセンタの遠隔モニタリングと遠隔操作の研究

平成１３年2月２８日提出

指導教官　　長尾高明　教授

高知工科大学　知能機械システム工学科

１０１０１５０　更谷　謙仁

目次

目次

第一章　序論

第一章　　序論

１．１　研究の背景

ここ５，６年の間にインターネットという言葉か盛んに聞こえ出した。それにともないパソコンのハード、ソフトが急激に発展したことにより、ネットワークの重要さが改めて認識されている。ネットワークは人と人とを結びつける時間と距離を無くしてしまったといえる。工業界においても加工の現場と操作者を切り離し、ネットワークによって２つの場所を結びつけるというテレイグジスタンス技術が注目されてきている。

加工の現場（以下本論文ではマシンサイトと呼ぶ）と操作室（以下ではオペレーションルームと呼ぶ）を別にすることは、操作の環境を快適なものにできるという利点があるが、利点はそれだけではない。たとえば設計者はデザインルームにいながらにして実験的な加工を行うことが出来るようになる。また、ネットワークによる情報の標準化は操作者に対してコンピュータの手助けを受け易くするだろう。つまりテレイグジスタンス技術は、これまで機械に近すぎた人にはもっと離れるチャンスを、機械に遠すぎた人にはもっと近づくチャンスを与えてくれる技術といえるだろう。

また、本社は日本、工場は海外というように土地代、人件費の安い海外に工場を建設しそこに日本から社員が出向き現地の人たちに機械操作方法を教えたり何かのトラブルの時のためにつねに何人かの日本人が駐在していなければならない。たとえ工場が日本にあったとしても操作方法が複雑化した最近の工作機械の操作を完全に覚えて扱えるようになるにはかなりの時間を要する。しかしこのテレイグジスタンス技術により、マシンサイトでは複雑な操作を覚えることなく、機械に直接触れる回数を減らすことで作業者の安全を守ることが出来る。オペレーションルームではリアルタイムに加工の状況を知ることができ、未然にトラブルを避けることができたり、製品に対する信頼性を持つことができる。新しいものを加工するときも、マシンサイトに行かなくてもオペレーションルームで加工プログラムを作成しそれをマシンサイトに送信するだけで加工が可能となり大幅な時間短縮になる。

しかしここでひとつ大きな問題が生じる。実際の加工は３次元の状態で行うけれど、遠隔操作は２次元のディスプレイを通じて操作を行っている。そのために思いもよらないミスが発生する場合がある。そのためにも使いやすいシステム、すなわち３次元座標系で直接操作を与え、かつ加工状態を手の感覚で感じることのできる操作システムを開発しなくてはならない。

１．２　本研究の目的

本研究では次世代の生産システムを目指して、あるべき機能と姿を考える。遠隔モニタリングにより加工中の切削力をリアルタイムに知ることにより、加工中の事故を防ぐことができ、また遠隔操作によって作業者が常に機械の前にいなくてはならないということがなくなり、安全性の向上にもつながる。長年工作機械に持たれていた危険、汚い、操作が難しいなどイメージを払拭することが可能になる。クライアントとサーバをネットワークで介して地理的に遠く離れた設計者、管理者、操作者等の分散オブジェクトとを接続して、協調作業ができるような開放型知能化加工システムを実現する。

第二章

本研究の遠隔操作の構成

２．１　ハードウェアの構成

ここで本研究に用いた操作システムのハードウェアの構成について述べる。

まず研究の中心となるマシニングセンタはパソコンとデータの送受信ができるものでなくてはならない。本研究ではマシニングセンタに大阪機工株式会社（ＯＫＫ）ＶＭ４Ⅱを使用し、データ送受信にパソコンに組み込むボードをＦＡＮＵＣ社のＨＳＳＢボードを使用する。

切削力を検出するためには力センサが必要である。本研究ではＫＩＳＴＬＥＲ社の３分力動力計を使用しする。力センサの出力はチャージ・アンプを介してアナログな電圧情報に変換され、Ａ／Ｄ変換ボードに入力されてデジタル信号となり、制御用のパソコンに取り込まれる。

２．１．１　オープンＣＮＣマシニングセンタ

オープンＣＮＣマシニングセンタとは外部のパソコンとデータの送受信とコンピュータの数値制御が可能なマシニングセンタのことである。本研究のオープンＣＮＣマシニングセンタは、大阪機工株式会社（ＯＫＫ）のＶＭ４Ⅱ（図2.1）を使用する。そのマシニングセンタにＩＢＭ　ＰＣ互換の市販のパソコンとＣＮＣの間を光ファイバケーブルで結合したシステムでノイズの影響を受けない高速・広範囲のデータ転送が可能なＦＡＮＵＣ社の高速シリアルバス（ＨＳＳＢ：High Speed Serial Bus）（図 2.2）を搭載する。これによりＨＳＳＢボードを組み込んだパソコンとデータの送受信が可能になる。

図 2.1　ＯＫＫ　ＶＭ４Ⅱ

第二章　　本研究の遠隔操作の構成

図 2.2　　FANUC 160i-M

２．１．２　力センサ

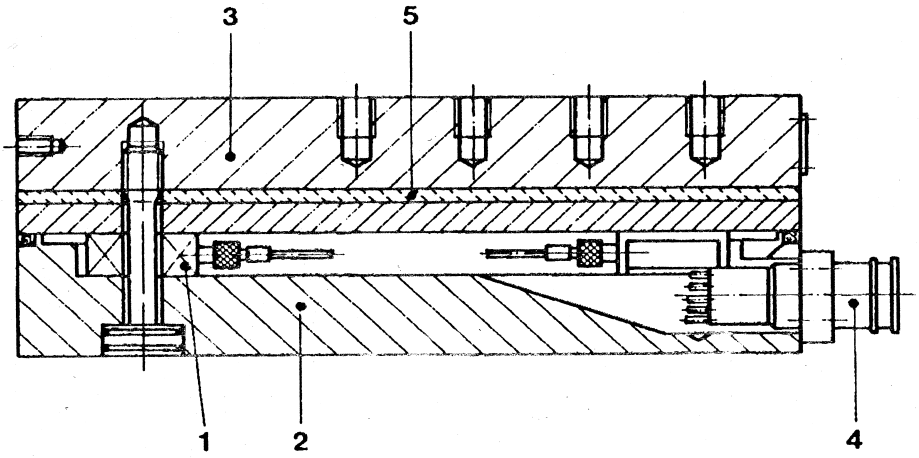
切削中の切削力を検出するために、3 軸力を検出するセンサを用いる。本研究では日本キスラー株式会社の 3 成分動力計（図 2.3）を使用する。

動力計の構造（図 2.4）は、ベース・プレートとトップ・プレートの間に高いプリロードをかけて挟み込まれた、4 個の 3 成分力変換器によって構成される。このプリロードは、摩擦力の伝達のために必要である。

４個の力変換器は、非接地の状態で組み込まれている。この結果グラウンド・ループ（接地による回り込み）の問題は、ほぼ解決している。

動力計には防錆材が使用されており、また水および冷却液の飛沫程度の進入に対しては、保護されている。

図 2.3　　3 成分動力計



1　　力変換器
2　　ベース・プレート
3　　トップ・プレート
4　　コネクタ
5　　断熱コーティング

図 2.4　動力計の構造

寸法		mm	140×170
重量		kg	7.3
許容過負荷	Fx,Fy,Fz	kN	－7.5/7.5
	Fz	kN	－7.5/15
接地絶縁		Ω	$>10^{8}$
静電容量		pF	220
感度	Fx,Fy	pC/N	－7.5
	Fz	pC/N	－3.5

表 2.1　3 成分動力計の仕様

２．１．２　チャージ・アンプとＡ／Ｄボード

チャージアンプとは水晶圧電式センサで発生した電荷をそれに比例した電圧に変換するもので本研究では日本キスラー株式会社のマルチチャンネルチャージ・アンプ 5019A（図 2.5）を使用する。使用したチャージ・アンプの仕様を表 2.2 に示す。

マルチチャンネルチャージ・アンプを通し、電圧に変換された力センサの出力を読み取るためにの株式会社インターフェイスのＡ／Ｄ変換ボード PCI‐3155（図 2.6）を使用する。使用したＡ／Ｄボードの使用を表 2.3 に示す。

図 2.5　チャージ・アンプ

チャンネル数		1～4
測定範囲(出力±10Vに対する入力電化量)	pC	±10~999,000
電源	VAC	115/230(切り替え可能)
	%	＋15/－22
	Hz	50/60(48～62)
消費電力	VA	20
寸法　幅×高×奥行	mm	396×187×280
重量	Kg	8
精度　±99.9pCFS まで	%	≦±3
±100pCFS 以上	%	≦±1
周波数特性　－3dB	kHz	0～200
出力　フルスケール出力電圧	V	±10
オーバーロード信号発生値	V	10.5
最大出力電流	mA	±5
出力インピーダンス	Ω	10
入力電圧(最大許容)	V	±125

表 2.2　チャージ・アンプの仕様

図 2.6　A／D変換ボード

入力チャンネル数	シングルエンド入力 16CH
	差動入力　8CH
入力形式	マルチプレクサ方式
入力レンジ	バイポーラ：±1V,±2.5V,±5V, ±10V
入力インピーダンス	10MΩ
入力保護	POWER ON 時　±35V
	POWER OFF 時　±20V
分解能	16bit
変換時間	10μsec(チャンネル固定時)
	10μsec(チャンネル切り替え高速時)
	20μsec(チャンネル切り替え通常時)
相対精度	±2LSBmax(at25℃)
誤差	±0.10%max(0～50℃): ±10V
	±0.15%max(0～50℃): ±1V, ±2.5V, ±5V
絶縁方式	非絶縁

表 2.3　A／Dボードの仕様

２．２　　プログラミング言語の構成

本研究では遠隔プログラム作成にあたって 2 つのプログラミング言語を用いた。Microsoft Visual J++6.0 と Microsoft Visual C++6.0 である。ウェブ上で公開できるJava 言語とプログラミング言語の主流であるC 言語とのそれぞれの特徴を生かした遠隔プログラムを作成する。

２．２．１　Ｖｉｓｕａｌ　Ｊ＋＋について

Java はもともと、Sun Microsystems 社により組込用の開発言語として研究されていたプログラミング言語で、1996 年 Microsoft 社が Java を開発言語として「Microsoft Visual J++1.0」を発表した。この Visual J++には Microsoft Windows 環境での資産を利用する仕組みを持たせていた開発ツールであった。本研究では Microsoft 社の「Microsoft Visual J++6.0」を使用する。

Java のメリットとしてマルチプラットフォームがある。１度プログラムを作れば、どんな環境でも動く。通常、Ｃ／Ｃ＋＋言語などのプログラムは、機械コードへコンパイルされる。これは、intelX86 や、PowerPC、DEC Alpha などのプロセッサに特化した形式にコンパイルされる。しかし、Java は機械コードへコンパイルしまい。Java 仮想マシンが理解する中間コードへとコンパイルされる。Java 仮想マシンは、実行時に必要な中間コードを動的にロードして実行する。その結果、オペレーティングシステムなどの環境の違いを Java 仮想マシンが吸収して、１度作成した Java プログラムがどのような環境でも動作するということが可能になる。

また、WWW(World Wide Web)でプログラムを発信できるということがある。この Java のメリットを生かせば世界中のどこからでも遠隔操作が可能というソフトを作成できる。

２．２．２　Ｖｉｓｕａｌ　Ｃ＋＋について

Visual C++は、Microsoft C の後継として登場した、Windows アプリケーション作成のための統合開発環境という位置付けにある言語。プログラム原型の自動作成やデバックなどの便利な開発支援ツールと、Windows の能力を最大限に引き出すためのライブラリを備えている。

Visual C++開発環境は、プログラミング支援のための強力なツールである２つの Wizard とクラスライブラリおよびエディタやコンパイラ、デバッガなどによって構成されている。

プログラム開発にあたって第一に利用するのが、AppWizard である。AppWizard は、Windows アプリケーションプログラムのもっとも基本的なソースコードを、プログラマに代わって自動作成してくれる便利なツールである。

Windows アプリケーションの動作はすべて Windows システムを経由して送られてくるメッセージという信号に対する応答によって行われる。メッセージに対して行う処理はプログラマがそのコードを書くことになるが、特にメッセージに応答するための特別の関数をメッセージハンドラと言い、そのメッセージハンドラを置くファイル、書き込む場所、また関数のプロトタイプ宣言や関数名までも用意してくれるのが、ClassWizard である。

Visual C++のプログラミングにあたって中核となるのが、MFC(Microsoft Foundation Class)ライブラリ。これらは独立したいろいろな関数の集合体ではなく、関連のある関数をグループ化してまとめたクラスとしてのライブラリである。

２．３　　遠隔操作の構成

本研究の構成は図 2.7 に示す。

MC に力センサを取り付け、力センサの出力をチャージアンプで電圧に変換し、パソコンに組み込んだ A/D 変換ボードでデータを読み取る。クライアントとサーバはネットワークでつながっており各種データや命令をリアルタイムに送受信する。またクライアントとサーバーは C 言語のソケット通信（図 2.8）でつながっており、クライアント側にサーバのパソコンの IP アドレスをいれ、1 対 1 のデータの送受信を繰り返す。サーバの HSSB ボードと MC の高速シリアルバス（HSSB）は光ケーブルでつながっておりこれもリアルタイムでデータの送受信が行える。

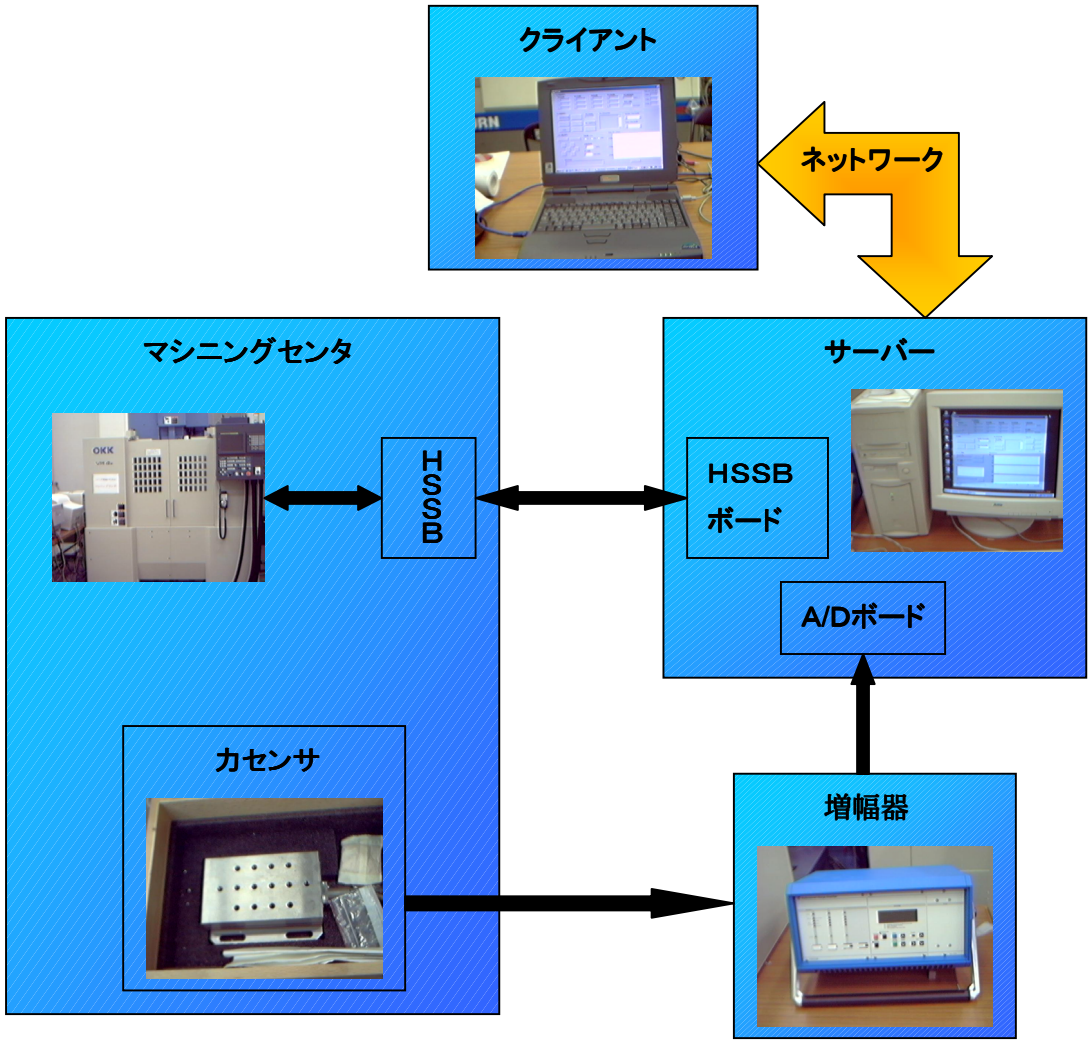


図 2.7　　本研究の構成図

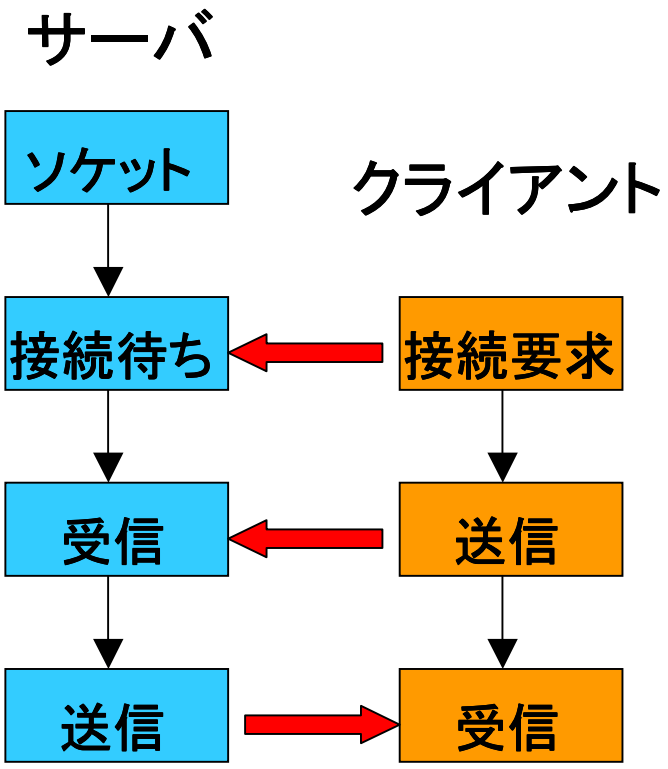


図 2.8　ソケット通信の構成図

第三章

Visual J ++による遠隔操作プログラムの作成

３．１　遠隔操作画面の構成

遠隔操作は誰もが簡単に行えるように単純で分かりやすいものにしなくてはならない。今回、Java 言語で作成する遠隔操作画面はそのことを一番に考えて作成する。図 3.1 のようにＸ、Ｙ、Ｚそれぞれの移動ボタン、移動量の選択ボタン、ＨＳＳＢ接続・切断ボタン、加工を開始するサイクルスタート、加工を一時停止するフィードホールド、そして異常停止ボタンを作成する。またＮＣプログラム検索と送信、ＮＣプログラム表示、絶対・機械座標、主軸回転数、送り速度の表示をするようにする。

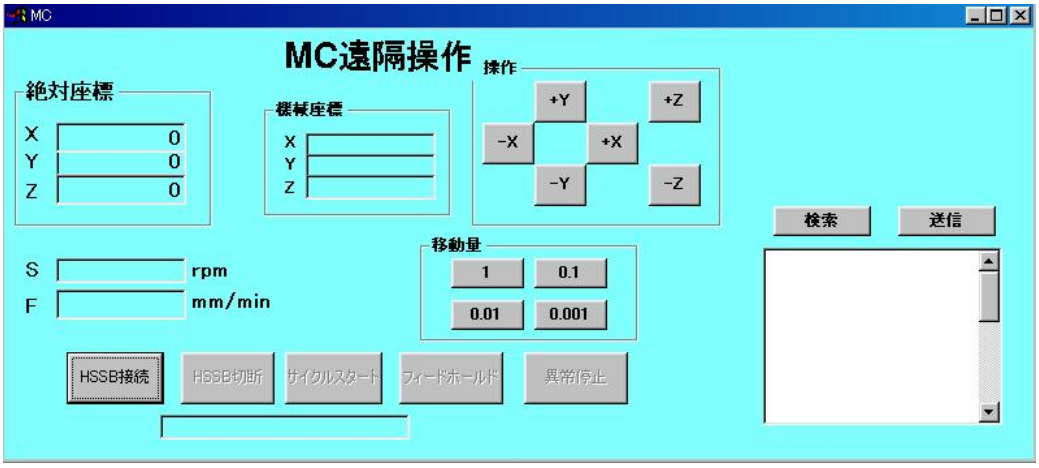


図 3.1　Visual J++で作成した遠隔操作画面

３．２　遠隔操作プログラムの説明

遠隔操作はクライアントとサーバの関係から成り立っており、クライアントとサーバはソケット通信でつながっている。今回はクライアントが Java 言語、サーバがＣ言語と異なるプログラミング言語だがソケット通信の構成は図 2.8 と同じである。

各種命令の送信は、ボタンを押したときにサーバに文字列を送信するようにし、サーバでその文字を識別しＭＣに命令を与えるようにする。たとえばＨＳＳＢ接続の命令の場合は“Ｏ”、ＨＳＳＢ切断には“Ｌ”を送信する。

また誤操作を防ぐために押せるボタンと押せないボタンを区別するように、button1.setEnabled(true)でボタンを押せるようにし、ボタンを押せなくするために button1.setEnabled(false)をプログラムに組み込む。

プログラムは付録参照。

３．３　実験結果

今回の Java 言語で作成した遠隔操作プログラムは、さまざまなボタンと各座標値の表示画面を作成したが、受信プログラムが作動せず送信のみとなってしまった。Java 言語のプログラムを作成して実行する前にはコンパイルという作業が必要で、このコンパイルでプログラムのエラーなどを見つけるのだが、今回作成したプログラムはエラーもなく、送受信できるプログラムなのだがなぜか受信のみできなかった。送信のほうは作成したボタンに対応する命令はきちんと送信され問題なくＭＣも稼動し、成功したといえる。

当初の目的はまずは簡単な送信と受信のプログラムを作成し、このプログラムが成功すればその他のボタンなどを作成し遠隔機能を向上させるつもりだったが受信という遠隔操作の大事な部分が Java 言語では実行できなかったのでこの遠隔実験は失敗に終わった。

第四章

Visual C++による遠隔操作プログラムの作成

４．１　遠隔操作画面の構成

Ｃ言語の場合も Java 言語と同様に遠隔操作画面を作成する。Ｃ言語の遠隔操作画面は Java 言語のものとは違いさまざまな機能を持たせた。

まず大きな違いとしてデータの受信がある。Java 言語の遠隔プログラムではできなかったデータの受信が C 言語の遠隔プログラムではリアルタイムで行える。機械座標、相対座標などの各座標値、主軸回転数、送り速度、そして力センサのデータなどの数値をリアルタイムに取得できるようにする。

また送信命令の数も大幅に増え、DNC 運転などのように新たな機能も持たせることにした。このことによりボタンの数も増え、一見複雑そうな操作画面になったが、研究の目的で述べたように誰にでも操作できる遠隔操作画面にしなければならないので、さまざまなガイド機能と誤操作防止機能を持たせることにした。

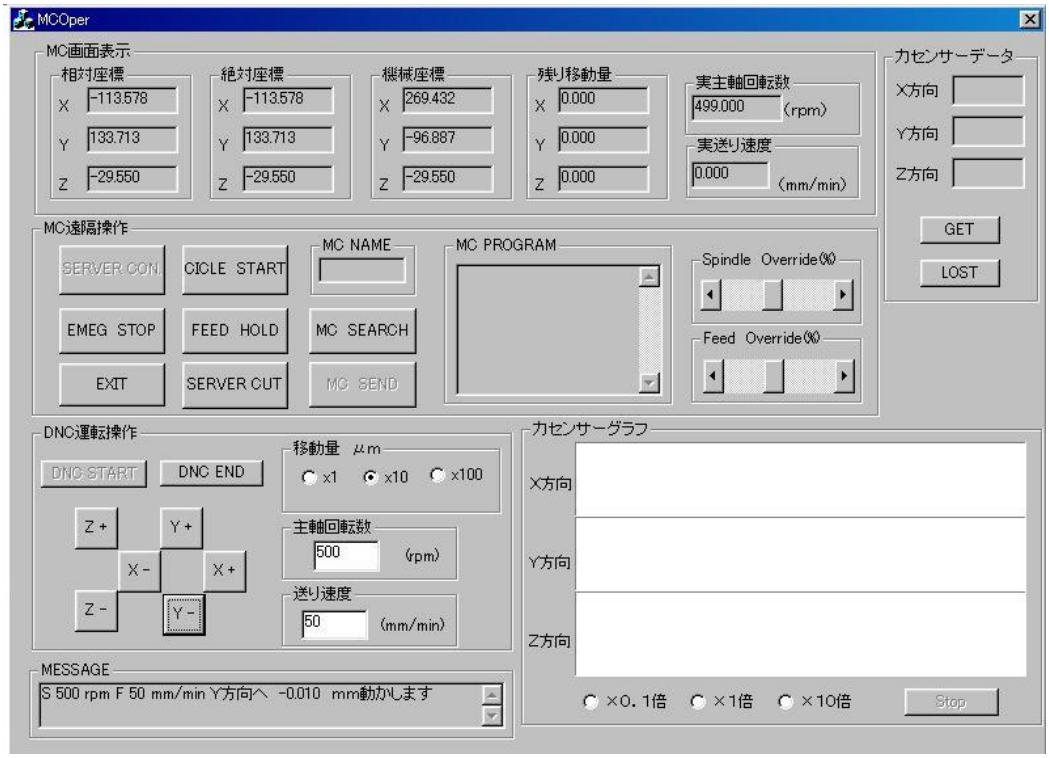


図 4.1　Visual C++で作成した遠隔操作画面

４．２　データの送受信

データの送信は前章の Java 言語の遠隔プログラム作成で述べたようにボタンを押したら文字列を送信するようにするという部分は同じだが、今回Ｃ言語の遠隔プログラムでは文字列のほかに複数の数字を同時に送信するので若干形が変わってくる。まず、受信データには rData、送信データには sData と名前を定義しておく。サイクルスタートなどの Java 言語の遠隔プログラムにあった命令は文字列のみの送信だが今回新たに加わったＤＮＣ運転は文字列と数列を同時に送信する。ＤＮＣ運転については 4.3 で詳しく説明する。

今回のデータに送受信には大きな条件がある。その条件がリアルタイムということである。遠隔操作をする際、各種命令の送信やデータの受信がリアルタイムに行えなければ、異常停止ボタンの命令が遅れたり、その異常事態に気づくことですら遅れてしまう。今回作成したプログラムでは加工中の映像などは見られないが、その映像が見えるようになっていればなおさら映像と取得する座標値データの違いに違和感を感じてしまうだろう。こういった問題を解決するためにもリアルタイムでのデータの送受信というのは必要不可欠なのである。

指定した時間間隔ごとにある処理を行うときには、タイマーを利用する。タイマー機能は、Windows に一定間隔でメッセージ（WM_TIMER）を送らせ、それをアプリケーション側に用意したメッセージハンドラで処理するという形で実現する。Visual C++6.0 には OnTimer というタイマーを利用するための関数がデフォルトで用意されているのでこれを利用する。

OnTimer 関数を追加しただけでは、タイマーは動作しない。タイマーを実際に使うときには、WindowsAPI 関数である、SetTimer 関数を使う。SetTimer 関数の構文は以下のようになる。

```
UINT SetTimer( UINT nIDEvent, UINT nElapse,
void (CALLBACK EXPORT*　lpfnTimer) (HWND,UINT,UINT,DWORD));
```

nIDEvent：タイマーの識別ＩＤを指定する。０では指定できない
nElapse　：　タイマーの間隔(WM_TIMER を発生させる間隔)をミリ秒単位で指定する
lpfnTimer：WM_TIMER メッセージを渡す関数を指定します。OnTimer を利用するときは NULL を指定する

今回の遠隔プログラムでは 100 ミリ秒（＝0.1 秒）ごとに OnTimer 関数を呼び出すようにした。リアルタイムでのデータ送受信の構成は図 4.2 に示す。

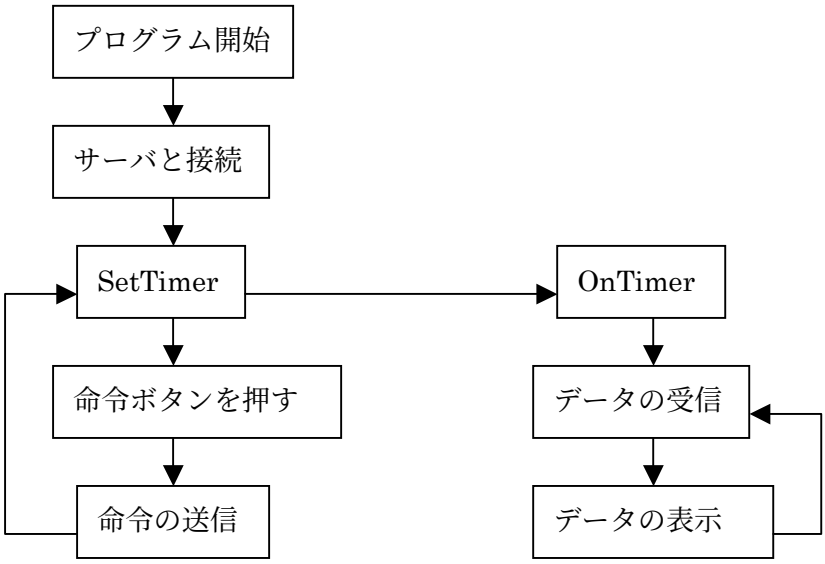


図 4.2　リアルタイムの送受信の構成

４．３　ＤＮＣ運転

前に述べたように今回Ｃ言語の遠隔操作画面にＤＮＣ運転という新しい機能を追加する。

DNC とは Direct Numerical Control の略で直接数値制御という意味である。ＭＣはＮＣプログラムという加工プログラムを入力して加工開始し製品を作成するのだが、ＤＮＣは加工や製品を作成するのではなくＸ軸を 0.1mmだけ動かしたいなどの加工の準備段階などで必要な機能である。ＭＣを直接操作する場合にはボタンや手動ハンドルなどで操作できる機能だが、遠隔操作の場合それと同じことをするには製品などを作成するときと同じようなＮＣプログラムが必要となる。しかしＤＮＣを遠隔操作するたびにこのようなＮＣプログラムを作成していたら遠隔操作の利点である時間短縮という点で問題になる。そのために本遠隔プログラムではＭＣを直接操作するようにボタンを押すだけでＤＮＣ運転ができるようにする。

ＤＮＣ運転画面（図 4.3）にはＸ、Ｙ、Ｚのそれぞれの＋、－方向ボタン、移動量、主軸回転数、送り速度を選択できるようにする。そして、選択したそれぞれのデータを同時に送信する。

図 4.3　ＤＮＣ運転操作画面

４．４　力センサのデータ取得とグラフ化

ＤＮＣ運転と同様に今回の遠隔プログラムでの新しい機能に力センサのデータ取得とグラフ化がある。

力センサのデータの取得は、各座標値の取得と同様にリアルタイムに行う。ただし力センサのデータは遠隔操作するたび必要なものではないので、データの取得の有無ができるようにボタンを作成する（図 4.4)。データの取得のボタンを押した状態でグラフを描くようにするのだが、力センサデータは加工の条件で数値が大きく変わるためグラフ自体に倍率選択ボタンを用意し過大、過小データでもグラフ化できるようにする（図 4.5)。このグラフもリアルタイムに描画できるように OnTimer 関数を使用する。

図 4.4　力センサデータ表示画面

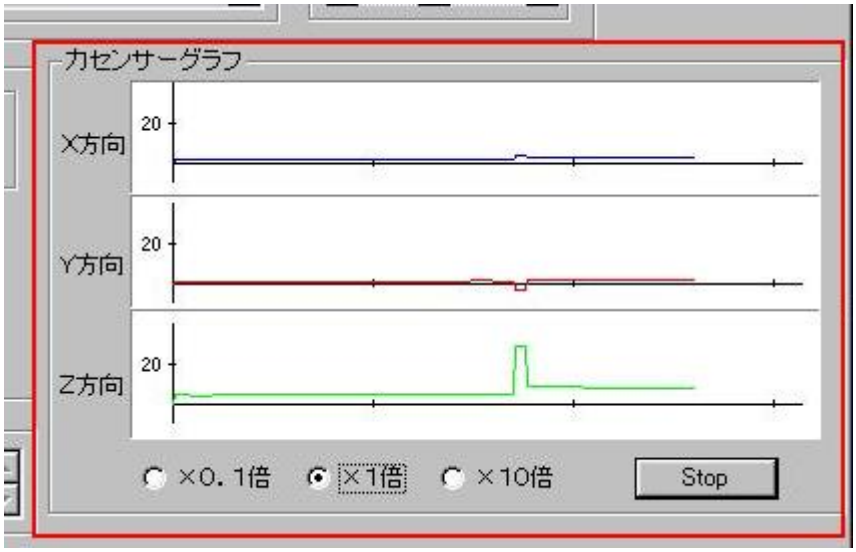


図 4.5　力センサグラフ

４．５　誤操作を防ぐ

遠隔操作で一番問題になるのが遠隔操作である。誤って押してはいけないボタンを押して命令が送信され、ＭＣが停止するなどといった誤操作を防ぐ処置を遠隔操作画面にも取り入れなくてはならない。また誰にでも操作でき、難しそうというイメージを払拭させるような遠隔操作画面にするのが目的なので、触ると壊しそうなどと思われないように今押せるボタンと押せないボタンを区別できるようにした（図 4.6)。これでうっかり違うボタンを押してしまうということもなくなり、押せるボタンしか押せないということは初心者でも分かるようなガイド機能も役割も果たせることになる。

ＤＮＣ運転では主軸回転数と送り速度を直接数値入力するようになっており、誤った数字を入力してしまうと、そのままＭＣに送信され機械の故障や事故にもつながるので一番注意しなければならない部分である。ＤＮＣ運転はＸ、Ｙ、Ｚそれぞれのボタンを押したときにデータを送信するようになっているので、まず移動量、主軸回転数、送り速度の３つをきちんと設定していなければ警告文を出すようにし、主軸回転数と送り速度は入力できる数値の範囲をそれぞれ設定し、範囲外の数値を入力したときにも警告文を出すようにする（図 4.7)。

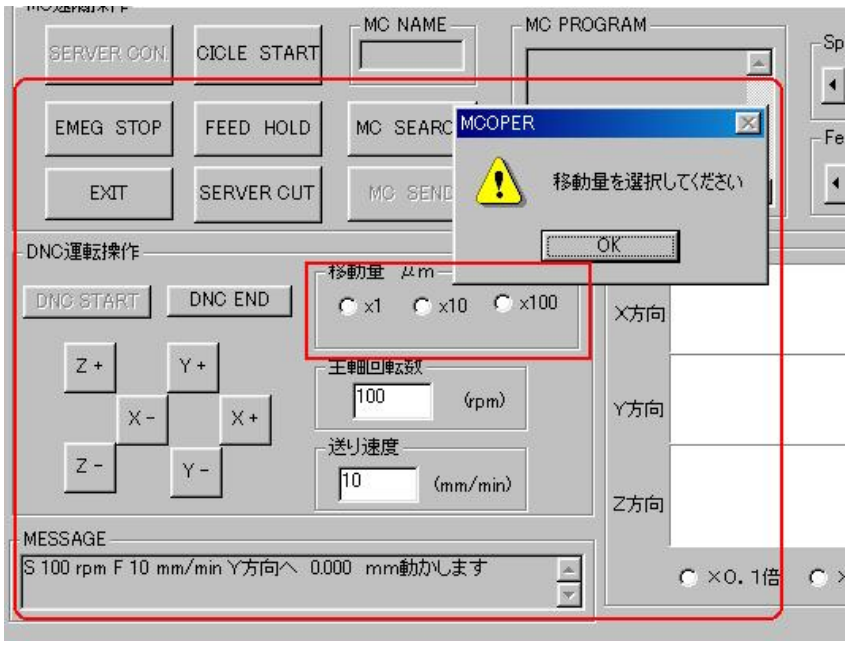


図 4.6　警告文１


MCOPER
送り速度0では動きません
OK
MCOPER
主軸回転数は0から6000の間
OK
MCOPER
移動量を選択してください
OK

図 4.7　警告文２

４．６　実験結果

遠隔操作プログラムは正常に作動し、遠隔画面にあるすべての命令ボタンはリアルタイムに送信できた。受信のほうも問題なく受信できたが、若干の時間のずれが生じリアルタイムとまではいかなかった。約２～３秒遅れて受信してしまい、力センサグラフも OnTimer 関数で 0.1 秒ごとに描画するようになっているが、受信データが遅れてくるために次のデータを受信するまでそれまでの値でグラフを描画するので、階段状のグラフになってしまう。

このように受信の時間のずれという問題がでてしまい、現時点では時間のずれの原因究明ができておらず、問題を解決できなかった。

第五章

結論

第五章　結論

今回､遠隔操作プログラムを Java 言語とＣ言語という 2 つのプログラミング言語で作成したのだが、両方ともすぐに使えるぐらいの実用性のあるものが完全な遠隔プログラムとすれば、問題だらけの不完全な遠隔プログラムだった。特に Java 言語で作成した遠隔プログラムは送信のみというものに終わってしまった。しかし、Java 言語はこのインターネット時代に対応したプログラミング言語であるので、最終的には Java 言語での遠隔プログラムが必要である。

またＣ言語で作成した遠隔プログラムは受信こそリアルタイムに行えなかったがそれ以外の機能は問題なく稼動し、遠隔操作自体もそれほど違和感なく行えることができた。しかし、遠隔操作実験を繰り返すうちにさまざまな問題点がでてきた。まずは遠隔操作の環境が限られてくるという点である。私の当初の目標がインターネットのホームページに遠隔操作画面を表示しどこからでも遠隔操作ができるというものであったが Java 言語の失敗で結局サーバとクライアントの１対１という遠隔操作の環境になってしまった。

つぎに、加工の様子や工具の位置などが映像などで見ることができないという点である。現時点では遠隔操作画面の座標値や力センサデータの数字のみで工具の位置や加工状態を認識しなければならなく、非常に分かりづらいという問題もある。

また力センサを用いて加工中の切削力を測定しているが、そのデータを利用していないという問題もある。今はただ加工が始まった、終わったという情報のみを力センサのデータで読み取っているだけである。

以上にように今回作成した遠隔操作プログラムにはさまざまな問題が残っている。しかし現段階でも危険、汚い、操作が難しいという問題は改善されつつあるのではないだろうか。遠隔操作実験にあたって直接機械を操作するのは機械の起動、ワークピースの取り付けのみで格段に機械に触れる回数が減ってきているし、実験中は機械からはなれていたので危険という感じもなく、また今回の実験にあたってはさまざまな人に協力してもらったのだが遠隔操作画面の説明を 1 度しただけで簡単に遠隔操作ができていたので操作が難しいということもなかったと考える。

結果としては遠隔操作は確実に今までの工作機械へのイメージを変えることのできるものであり、さまざまの問題を解決したとき、次世代の加工技術のひとつになるであると私は考える。

第六章
展望

第六章　展望

これからの展望としては、本研究で使用したＭＣは外部パソコンから送信されたパルス信号によって補正プログラムを加工中のＮＣプログラムに割り込ませる機能（外部ハンドルパルス信号割り込み機能）があるので、その機能を利用して切削力による加工誤差を修正する制御命令を送信するようにし、高精度加工を実現する。

また、加工の様子をグラフィックで表示し、誰にでも分かりやすい操作画面を作成すること。このグラフィック機能を利用してＮＣプログラムをチェックできるようにし、加工中の事故をなくすようにする。

次に、サーバにクライアントのプログラムを入れ、遠隔操作画面をホームページで公開し、そのホームページにアクセスすれば遠隔操作が出来るようにして、遠隔操作の環境を広げる。

私の研究室ではジョイスティックを使用した遠隔操作の研究をしているグループがあり、その研究はまだＭＣを遠隔操作するまでには至ってないので、私のＤＮＣ運転部分のプログラムにジョイスティックを使用できるプログラムを組み込み、味覚、嗅覚以外の五感を使う「感じる」遠隔操作プログラムを実現させる。

謝辞

謝辞
本研究は、長尾高明教授の御薫陶を受けて学び研究したものであります。研究全般にわたり、長尾教授には、貴重な御助言、御指導を頂きました。
指導教官である李軍揮助教授には、終始並々ならぬ御指導、そして暖かい励ましを頂きました。
４年の木崎さん、増吉さんには、研究室の同士として、いろいろな面でお世話になりました。
以上の方々に深く感謝し、厚くお礼を申し上げます。

参考文献

参考文献

[1]　佐々木整“はじめての Java”株式会社秀和システム、１９９６
[2]　本俊也“Visual J++6.0 入門”株式会社インプレス、１９９８
[3]　小橋一“ＭＦＣプログラミング” 株式会社秀和システム、１９９９
[4]　山地秀美“はじめての Visual C++6.0” 株式会社技術評論社、１９９９
[5]　吉田弘一郎“極める Visual C＋＋” 株式会社技術評論社、１９９８
[6]　河西朝雄“標準 Visual C＋＋プログラミングブック” 株式会社技術評論社、２０００
[7]　清水康晶“作ってわかる Visual C++6.0” 株式会社秀和システム、２０００
[8]　山岡祥“Visual C＋＋入門”ＣＱ出版株式会社、１９９６
[9]　林晴比古“新Ｃ＋＋入門”ソフトバンクパブリッシング株式会社、１９９８
[10]　長尾高明、畑村洋太郎、光石衛、中尾政之“知能化生産システム” 株式会社朝倉書店、２０００
[11]　東豊一郎“加工の臨場感通信システムにおける操作システムの研究”平成４年卒業論文
[12]　李軍揮“オープンアーキテクチャＣＮＣを用いたセンサ情報に基づく知能化加工システム研究”平成１０年学位論文

付録

付録

本研究で私が作成した Java 言語、C 言語の遠隔操作プログラムを付録とします。ただしプログラム無断使用を防ぐために数カ所プログラムを削除しています。ご了承ください。

Java 言語で作成した遠隔操作プログラム

```
import com.ms.wfc.app.*;
import com.ms.wfc.core.*;
import com.ms.wfc.ui.*;
import com.ms.wfc.html.*;
import java.net.*;
import java.io.*;
import java.applet.Applet;
import java.util.StringTokenizer;
import java.applet.AudioClip;

/**
 * このクラスはコマンド行から任意の個数の引数を受けとることができま
 * す。プログラムの実行は main() メソッドから始まります。このクラス
 * のコンストラクタは、 main() メソッドで 'Form1' 型のオブジェ
 * クトが生成された際に初めて呼び出されます。
 */
public class Form1 extends Form
{
                .
                .
                    // Visual J++ フォーム デザイナのために必要
        Thread  thread;
        String dummy;
        String rData;
        int       sleeptime;
        public Form1()
        {
// Visual J++ フォーム デザイナのために必要
        initForm();
        creatConnection();
// TODO: 構築用のコードを追加します。initForm の呼び出し前には追加しないでください。
        }
          public void start() {
        if (thread == null) {
```

```
            thread = new Thread();
            thread.start();
        }
    }
    public void stop()
    {
        if (thread != null) {
            thread.stop();
            thread = null;
        }
    }
        public void creatConnection()
                {
                        .
                        .
                        .
                }
// TODO: 構築用のコードを追加します。initForm の呼び出し前には追加しないでください。
        /**
         * Form1 は、コンポーネント リストを削除するために dispose メソッドを
         * オーバーライドします。
         */
        public void dispose()
        {
                        .
                        .
        }
        private void button1_click(Object source, Event e)
        {
                edit4.setText("HSSB 接続しました");
                button1.setEnabled(false);
                button2.setEnabled(true);
                button3.setEnabled(true);
                button4.setEnabled(true);
                button15.setEnabled(true);
                        .
```

```
                    .
}
private void button2_click(Object source, Event e)
{
        edit4.setText("HSSB 解除しました");
        button2.setEnabled(false);
        button1.setEnabled(true);
                    .
                    .
}
private void button3_click(Object source, Event e)
{
        edit4.setText("サイクルスタートしました");
        button3.setEnabled(false);
        button4.setEnabled(true);
                    .
                    .
}
private void button4_click(Object source, Event e)
{
        edit4.setText("フィードホールドしました");
        button4.setEnabled(false);
        button3.setEnabled(true);
                    .
                    .
}
double a=0;
private void button11_click(Object source, Event e)
{
        edit4.setText("移動量 1.0 にしました");
        a=1;
}
private void button12_click(Object source, Event e)
{
        edit4.setText("移動量 0.1 にしました");
        a=0.1;
```

```
	}
	private void button13_click(Object source, Event e)
	{
		edit4.setText("移動量 0.01 にしました");
		a=0.01;
	}
	private void button14_click(Object source, Event e)
	{
		edit4.setText("移動量 0.001 にしました");
		a=0.001;
	}
double m1,n1;
	private void button5_click(Object source, Event e)
	{
	n1=m1+a;
	edit2.setText(Double.toString(n1));
	m1=n1;
	}
	private void button6_click(Object source, Event e)
	{
		n1=m1-a;
	edit2.setText(Double.toString(n1));
	m1=n1;
	}
double o,p;
	private void button7_click(Object source, Event e)
	{
	p=o+a;
	edit1.setText(Double.toString(p));
	o=p;
	}
	private void button8_click(Object source, Event e)
	{
		p=o-a;
	edit1.setText(Double.toString(p));
	o=p;
```

```
        }
double m,n;
        private void button9_click(Object source, Event e)
        {
                n=m+a;
        edit3.setText(Double.toString(n));
        m=n;
        }
        private void button10_click(Object source, Event e)
        {
                n=m-a;
        edit3.setText(Double.toString(n));
        m=n;
        }
    private void button15_click(Object source, Event e)
        {
                edit4.setText("異常停止します");
        this.setBackColor(Color.RED);
        button1.setEnabled(false);
        button2.setEnabled(false);
        button3.setEnabled(false);
        button4.setEnabled(false);
        button15.setEnabled(false);
        button1.hide();
        button2.hide();
        button3.hide();
        button4.hide();
        button15.hide();
        button16.show()
                .
                .
        }
        private void button16_click(Object source, Event e)
        {
                edit4.setText("異常停止解除します");
                .
```

```
        button1.setEnabled(true);
        button2.setEnabled(true);
        button3.setEnabled(true);
        button4.setEnabled(true);
        button15.setEnabled(true);
        button16.hide();
        button1.show();
        button2.show();
        button3.show();
        button4.show();
        button15.show();
        }
public void run() {
 StringTokenizer st;
        while(true) {
                    /* スリープ */
            try {
                thread.sleep(100);
            } catch (InterruptedException e) {
                break;
            }
                try{
                                .
                                .
            }}}
        * 注意: 以下のコードは Visual J++ フォームデザイナのために
        * 必要です。フォーム デザイナで変更することができます。コード
        * エディタで変更しないようにしてください。
        */
Container components = new Container();
Button button1 = new Button();
Button button3 = new Button();
Button button4 = new Button();
Button button2 = new Button();
Label label1 = new Label();
Label label4 = new Label();
```

```
Label label3 = new Label();
Edit edit3 = new Edit();
Edit edit2 = new Edit();
Label label14 = new Label();
Edit edit1 = new Edit();
Edit edit4 = new Edit();
Button button15 = new Button();
Button button16 = new Button();
Label label6 = new Label();
Edit edit5 = new Edit();
Label label2 = new Label();
Label label7 = new Label();
Edit edit7 = new Edit();
Edit edit8 = new Edit();
Edit edit6 = new Edit();
Label label8 = new Label();
Label label10 = new Label();
Label label9 = new Label();
Label label13 = new Label();
Button button5 = new Button();
Button button6 = new Button();
Button button7 = new Button();
Button button8 = new Button();
Button button9 = new Button();
Button button10 = new Button();
GroupBox groupBox1 = new GroupBox();
Button button11 = new Button();
Button button12 = new Button();
Button button13 = new Button();
Button button14 = new Button();
GroupBox groupBox2 = new GroupBox();
ListBox listBox1 = new ListBox();
Button button18 = new Button();
Edit edit9 = new Edit();
Label label5 = new Label();
Label label11 = new Label();
```

```
Label label12 = new Label();
Edit edit12 = new Edit();
Edit edit11 = new Edit();
Edit edit10 = new Edit();
Label label15 = new Label();
Label label16 = new Label();
Label label17 = new Label();
Label label21 = new Label();
Label label20 = new Label();
Label label19 = new Label();
Label label18 = new Label();
Edit edit15 = new Edit();
Edit edit14 = new Edit();
Edit edit13 = new Edit()
private void initForm()
{
        this.setBackColor(new Color(128, 255, 255));
        this.setLocation(new Point(2, 3));
        this.setText("MC");
        this.setAutoScaleBaseSize(new Point(5, 12));
        this.setClientSize(new Point(792, 515));
        button1.setLocation(new Point(144, 312));
        button1.setSize(new Point(75, 40));
        button1.setTabIndex(0);
        button1.setText("HSSB 接続");
        button1.addOnClick(new EventHandler(this.button1_click));
        button3.setEnabled(false);
        button3.setLocation(new Point(312, 312));
        button3.setSize(new Point(75, 40));
        button3.setTabIndex(2);
        button3.setText("サイクルスタート");
        button3.addOnClick(new EventHandler(this.button3_click));
        button4.setEnabled(false);
        button4.setLocation(new Point(400, 312));
        button4.setSize(new Point(80, 40));
        button4.setTabIndex(3);
```

```
        button4.setText("フィードホールド");
        button4.addOnClick(new EventHandler(this.button4_click));
        button2.setEnabled(false);
        button2.setLocation(new Point(232, 312));
        button2.setSize(new Point(72, 40));
        button2.setTabIndex(1);
        button2.setText("HSSB 切断");
        button2.addOnClick(new EventHandler(this.button2_click));
        label1.setFont(new Font("ＨＧｺﾞｼｯｸ E-PRO", 20.0f, FontSize.POINTS,
FontWeight.NORMAL, false, false, false, CharacterSet.DEFAULT, 0));
        label1.setLocation(new Point(256, 0));
        label1.setSize(new Point(152, 32));
        label1.setTabIndex(4);
        label1.setTabStop(false);
        label1.setText("MC 遠隔操作");
        label4.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
        label4.setLocation(new Point(16, 100));
        label4.setSize(new Point(16, 23));
        label4.setTabIndex(30);
        label4.setTabStop(false);
        label4.setText("Y");
         label3.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
        label3.setLocation(new Point(16, 74));
        label3.setSize(new Point(16, 23));
        label3.setTabIndex(29);
        label3.setTabStop(false);
        label3.setText("X");
        edit3.setBackColor(new Color(128, 255, 255));
        edit3.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
        edit3.setLocation(new Point(40, 120));
        edit3.setSize(new Point(100, 23));
        edit3.setTabIndex(9);
        edit3.setText("0");
```

```
            edit3.setReadOnly(true);
            edit3.setTextAlign(HorizontalAlignment.RIGHT);
            edit2.setBackColor(new Color(128, 255, 255));
            edit2.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            edit2.setLocation(new Point(40, 96));
            edit2.setSize(new Point(100, 23));
            edit2.setTabIndex(8);
            edit2.setText("0");
            edit2.setReadOnly(true);
            edit2.setTextAlign(HorizontalAlignment.RIGHT);
            label14.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            label14.setLocation(new Point(280, 120));
            label14.setSize(new Point(16, 23));
            label14.setTabIndex(21);
            label14.setTabStop(false);
            label14.setText("Z");
            edit1.setBackColor(new Color(128, 255, 255));
            edit1.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            edit1.setLocation(new Point(40, 72));
            edit1.setSize(new Point(100, 23));
            edit1.setTabIndex(25);
            edit1.setText("0");
            edit1.setMaxLength(6);
            edit1.setReadOnly(true);
            edit1.setTextAlign(HorizontalAlignment.RIGHT);
            edit4.setBackColor(new Color(128, 255, 255));
            edit4.setLocation(new Point(200, 272));
            edit4.setSize(new Point(192, 19));
            edit4.setTabIndex(5);
            edit4.setText("");
            edit4.setReadOnly(true);
            button15.setEnabled(false);
            button15.setLocation(new Point(496, 312));
```

```
        button15.setSize(new Point(80, 40));
        button15.setTabIndex(6);
        button15.setText("異常停止");
        button15.addOnClick(new EventHandler(this.button15_click));
        button16.setFont(new Font("MS UI Gothic", 9.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
        button16.setLocation(new Point(144, 312));
        button16.setSize(new Point(432, 40));
        button16.setTabIndex(7);
        button16.setText("異常停止解除");
        button16.setVisible(false);
        button16.addOnClick(new EventHandler(this.button16_click));
        label6.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
        label6.setLocation(new Point(16, 125));
        label6.setSize(new Point(16, 23));
        label6.setTabIndex(31);
        label6.setTabStop(false);
        label6.setText("Z");
        edit5.setBackColor(new Color(128, 255, 255));
        edit5.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
        edit5.setLocation(new Point(40, 200));
        edit5.setSize(new Point(100, 23));
        edit5.setTabIndex(32);
        edit5.setText("");
        edit5.setReadOnly(true);
        label2.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
        label2.setLocation(new Point(16, 204));
        label2.setSize(new Point(16, 23));
        label2.setTabIndex(33);
        label2.setTabStop(false);
        label2.setText("F");
        label7.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
```

```
label7.setLocation(new Point(144, 176));
label7.setSize(new Point(32, 23));
label7.setTabIndex(34);
label7.setTabStop(false);
label7.setText("rpm");
edit7.setBackColor(new Color(128, 255, 255));
edit7.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
edit7.setLocation(new Point(168, 96));
edit7.setSize(new Point(100, 23));
edit7.setTabIndex(18);
edit7.setText("");
edit7.setReadOnly(true);
edit7.setTextAlign(HorizontalAlignment.RIGHT);
edit8.setBackColor(new Color(128, 255, 255));
edit8.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
edit8.setLocation(new Point(168, 120));
edit8.setSize(new Point(100, 23));
edit8.setTabIndex(12);
edit8.setText("");
edit8.setReadOnly(true);
edit8.setTextAlign(HorizontalAlignment.RIGHT);
edit6.setBackColor(new Color(128, 255, 255));
edit6.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
edit6.setLocation(new Point(168, 72));
edit6.setSize(new Point(100, 23));
edit6.setTabIndex(24);
edit6.setText("");
edit6.setReadOnly(true);
edit6.setTextAlign(HorizontalAlignment.RIGHT);
label8.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
label8.setLocation(new Point(152, 72));
label8.setSize(new Point(16, 23));
```

```
            label8.setTabIndex(26);
            label8.setTabStop(false);
            label8.setText("X");
            label10.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            label10.setLocation(new Point(152, 120));
            label10.setSize(new Point(16, 23));
            label10.setTabIndex(28);
            label10.setTabStop(false);
            label10.setText("Z");
            label9.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            label9.setLocation(new Point(152, 96));
            label9.setSize(new Point(16, 23));
            label9.setTabIndex(27);
            label9.setTabStop(false);
            label9.setText("Y");
            label13.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            label13.setLocation(new Point(280, 96));
            label13.setSize(new Point(16, 23));
            label13.setTabIndex(20);
            label13.setTabStop(false);
            label13.setText("Y");
            button5.setFont(new Font("MS UI Gothic", 9.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            button5.setLocation(new Point(96, 368));
            button5.setSize(new Point(40, 24));
            button5.setTabIndex(35);
            button5.setText("+Y");
            button5.addOnClick(new EventHandler(this.button5_click));
            button6.setFont(new Font("MS UI Gothic", 9.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            button6.setLocation(new Point(96, 432));
            button6.setSize(new Point(40, 24));
            button6.setTabIndex(36);
```

```
            button6.setText("-Y");
            button7.setFont(new Font("MS UI Gothic", 9.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            button7.setLocation(new Point(136, 400));
            button7.setSize(new Point(40, 24));
            button7.setTabIndex(37);
            button7.setText("+X");
            button7.addOnClick(new EventHandler(this.button7_click));
            button8.setFont(new Font("MS UI Gothic", 9.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            button8.setLocation(new Point(56, 400));
            button8.setSize(new Point(40, 24));
            button8.setTabIndex(38);
            button8.setText("-X");
            button9.setFont(new Font("MS UI Gothic", 9.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            button9.setLocation(new Point(184, 368));
            button9.setSize(new Point(40, 24));
            button9.setTabIndex(39);
            button9.setText("+Z");
            button9.addOnClick(new EventHandler(this.button9_click));
            button10.setFont(new Font("MS UI Gothic", 9.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            button10.setLocation(new Point(184, 432));
            button10.setSize(new Point(40, 24));
            button10.setTabIndex(40);
            button10.setText("-Z");
            button10.addOnClick(new EventHandler(this.button10_click));
            groupBox1.setFont(new Font("MS UI Gothic", 9.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            groupBox1.setLocation(new Point(48, 352));
            groupBox1.setSize(new Point(192, 120));
            groupBox1.setTabIndex(41);
            groupBox1.setTabStop(false);
            groupBox1.setText("操作");
            button11.setFont(new Font("MS UI Gothic", 9.0f, FontSize.POINTS,
```

```
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            button11.setLocation(new Point(312, 376));
            button11.setSize(new Point(56, 23));
            button11.setTabIndex(42);
            button11.setText("1");
            button11.addOnClick(new EventHandler(this.button11_click));
            button12.setFont(new Font("MS UI Gothic", 9.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            button12.setLocation(new Point(376, 376));
            button12.setSize(new Point(56, 23));
            button12.setTabIndex(43);
            button12.setText("0.1");
            button12.addOnClick(new EventHandler(this.button12_click));
            button13.setFont(new Font("MS UI Gothic", 9.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            button13.setLocation(new Point(312, 408));
            button13.setSize(new Point(56, 23));
            button13.setTabIndex(44);
            button13.setText("0.01");
            button13.addOnClick(new EventHandler(this.button13_click));
            button14.setFont(new Font("MS UI Gothic", 9.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            button14.setLocation(new Point(376, 408));
            button14.setSize(new Point(56, 23));
            button14.setTabIndex(45);
            button14.setText("0.001");
            groupBox2.setFont(new Font("MS UI Gothic", 9.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            groupBox2.setLocation(new Point(288, 360));
            groupBox2.setSize(new Point(168, 80));
            groupBox2.setTabIndex(46);
            groupBox2.setTabStop(false);
            groupBox2.setText("移動量");
            listBox1.setLocation(new Point(584, 168));
            listBox1.setSize(new Point(184, 136));
            listBox1.setTabIndex(47);
```

```
        listBox1.setText("listBox1");
        listBox1.setUseTabStops(true);
        button18.setFont(new Font("MS UI Gothic", 9.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
        button18.setLocation(new Point(688, 136));
        button18.setSize(new Point(75, 23));
        button18.setTabIndex(49);
        button18.setText("送信");
        edit9.setBackColor(new Color(128, 255, 255));
        edit9.setLocation(new Point(40, 176));
        edit9.setSize(new Point(100, 19));
        edit9.setTabIndex(50);
        edit9.setText("");
        label5.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
        label5.setLocation(new Point(16, 176));
        label5.setSize(new Point(100, 23));
        label5.setTabIndex(51);
        label5.setTabStop(false);
        label5.setText("S");
        label11.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
        label11.setLocation(new Point(144, 200));
        label11.setSize(new Point(64, 23));
        label11.setTabIndex(52);
        label11.setTabStop(false);
        label11.setText("mm/min");
        label12.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
        label12.setLocation(new Point(280, 72));
        label12.setSize(new Point(16, 23));
        label12.setTabIndex(19);
        label12.setTabStop(false);
        label12.setText("X");
        edit12.setBackColor(new Color(128, 255, 255));
        edit12.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
```

```
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
        edit12.setLocation(new Point(296, 120));
        edit12.setSize(new Point(100, 23));
        edit12.setTabIndex(11);
        edit12.setText("0");
        edit12.setMaxLength(6);
        edit12.setReadOnly(true);
        edit12.setTextAlign(HorizontalAlignment.RIGHT);
        edit11.setBackColor(new Color(128, 255, 255));
        edit11.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
        edit11.setLocation(new Point(296, 96));
        edit11.setSize(new Point(100, 23));
        edit11.setTabIndex(17);
        edit11.setText("0");
        edit11.setMaxLength(6);
        edit11.setReadOnly(true);
        edit11.setTextAlign(HorizontalAlignment.RIGHT);
        edit10.setBackColor(new Color(128, 255, 255));
        edit10.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
        edit10.setLocation(new Point(296, 72));
        edit10.setSize(new Point(100, 23));
        edit10.setTabIndex(23);
        edit10.setText("0");
        edit10.setMaxLength(6);
        edit10.setReadOnly(true);
        edit10.setTextAlign(HorizontalAlignment.RIGHT);
        label15.setFont(new Font("ＭＳ ゴシック", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
        label15.setLocation(new Point(56, 48));
        label15.setSize(new Point(100, 23));
        label15.setTabIndex(53);
        label15.setTabStop(false);
        label15.setText("機械座標");
        label16.setFont(new Font("ＭＳ ゴシック", 12.0f, FontSize.POINTS,
```

```
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            label16.setLocation(new Point(184, 48));
            label16.setSize(new Point(100, 23));
            label16.setTabIndex(54);
            label16.setTabStop(false);
            label16.setText("相対座標");
            label17.setFont(new Font("ＭＳ ゴシック", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            label17.setLocation(new Point(312, 48));
            label17.setSize(new Point(100, 23));
            label17.setTabIndex(55);
            label17.setTabStop(false);
            label17.setText("絶対座標");
            label21.setFont(new Font("ＭＳ ゴシック", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            label21.setLocation(new Point(424, 48));
            label21.setSize(new Point(120, 23));
            label21.setTabIndex(48);
            label21.setTabStop(false);
            label21.setText("残りの移動量");
            label20.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            label20.setLocation(new Point(408, 120));
            label20.setSize(new Point(16, 23));
            label20.setTabIndex(16);
            label20.setTabStop(false);
            label20.setText("Z");
            label19.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            label19.setLocation(new Point(408, 96));
            label19.setSize(new Point(16, 23));
            label19.setTabIndex(15);
            label19.setTabStop(false);
            label19.setText("Y");
            label18.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
```

```
            label18.setLocation(new Point(408, 72));
            label18.setSize(new Point(16, 23));
            label18.setTabIndex(14);
            label18.setTabStop(false);
            label18.setText("X");
            edit15.setBackColor(new Color(128, 255, 255));
            edit15.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            edit15.setLocation(new Point(424, 120));
            edit15.setSize(new Point(100, 23));
            edit15.setTabIndex(10);
            edit15.setText("0");
            edit15.setMaxLength(6);
            edit15.setReadOnly(true);
            edit15.setTextAlign(HorizontalAlignment.RIGHT);
            edit14.setBackColor(new Color(128, 255, 255));
            edit14.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            edit14.setLocation(new Point(424, 96));
            edit14.setSize(new Point(100, 23));
            edit14.setTabIndex(13);
            edit14.setText("0");
            edit14.setMaxLength(6);
            edit14.setReadOnly(true);
            edit14.setTextAlign(HorizontalAlignment.RIGHT);
            edit13.setBackColor(new Color(128, 255, 255));
            edit13.setFont(new Font("MS UI Gothic", 12.0f, FontSize.POINTS,
FontWeight.BOLD, false, false, false, CharacterSet.DEFAULT, 0));
            edit13.setLocation(new Point(424, 72));
            edit13.setSize(new Point(100, 23));
            edit13.setTabIndex(22);
            edit13.setText("0");
            edit13.setMaxLength(6);
            edit13.setReadOnly(true);
            edit13.setTextAlign(HorizontalAlignment.RIGHT);
            this.setNewControls(new Control[] {
```

label21,
label20,
label19,
label18,
edit15,
edit14,
edit13,
label17,
label16,
label15,
label14,
label13,
label12,
edit12,
edit11,
edit10,
label10,
label9,
label8,
edit8,
edit7,
edit6,
label11,
edit9,
button18,
listBox1,
button14,
button13,
button12,
button11,
button10,
button9,
button8,
button7,
button6,
button5,

```
                                                    label7,
                                                    label2,
                                                    edit5,
                                                    label6,
                                                    label4,
                                                    label3,
                                                    edit1,
                                                    edit3,
                                                    edit2,
                                                    edit4,
                                                    button4,
                                                    button3,
                                                    button2,
                                                    button1,
                                                    button15,
                                                    label1,
                                                    button16,
                                                    groupBox1,
                                                    groupBox2,
                                                    label5});
    }
public static void main (String args[]) {
        Application.run(new Form1());
        /**
         * アプリケーションのメイン エントリ ポイントです。
         *
         * 引数 args に、コマンドラインからアプリケーションに渡された引数が
         * 配列として設定されます。
         */
}
}
```

Ｃ言語で作成した遠隔操作プログラム

```
// MCOperDlg.cpp : インプリメンテーション ファイル
#include "stdafx.h"
#include "MCOper.h"
#include "MCOperDlg.h"
int l,Y=0;
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
double tu,xa,ya,yb,yc,y1,y2,y3,V;
#endif
#define READTIMER_ID 0
static S_DATA    sData,rData;
static HANDLE  g_hThread;
static CComm*   sock;
static double n;
static char                    dummy[256];
static bool StartFlag = false;
/////////////////////////////////////////////////////////////////////////////
// アプリケーションのバージョン情報で使われている CAboutDlg ダイアログ
class CAboutDlg : public CDialog
{
public:
CAboutDlg();
// ダイアログ データ
        enum { IDD = IDD_ABOUTBOX };
// ClassWizard は仮想関数のオーバーライドを生成します
        protected:
virtual void DoDataExchange(CDataExchange* pDX);      // DDX/DDV のサポート
// インプリメンテーション
protected:
        //{{AFX_MSG(CAboutDlg)
                DECLARE_MESSAGE_MAP()
};
```

```
CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
	//{{AFX_DATA_INIT(CAboutDlg)
	//}}AFX_DATA_INIT
}
void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
	//{{AFX_DATA_MAP(CAboutDlg)
	//}}AFX_DATA_MAP
}
BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
	//{{AFX_MSG_MAP(CAboutDlg)
		// メッセージ ハンドラがありません。
	//}}AFX_MSG_MAP
END_MESSAGE_MAP();
////////////////////////////////////////////////////////////////////////////
// CMCOperDlg ダイアログ
CMCOperDlg::CMCOperDlg(CWnd* pParent /*=NULL*/)
	: CDialog(CMCOperDlg::IDD, pParent)
{
	//{{AFX_DATA_INIT(CMCOperDlg)
	m_rely = _T("");
	m_relz = _T("");
	m_absx = _T("");
	m_absy = _T("");
	m_absz = _T("");
	m_spindle = _T("");
	m_feed = _T("");
	m_message = _T("");
	m_ncprog = _T("");
	NCNAME = _T("");
	m_macx = _T("");
	m_macy = _T("");
	m_macz = _T("");
	m_relx = _T("");
```

```
	m_remx = _T("");
	m_remy = _T("");
	m_remz = _T("");
	m_feedover = 100;
	m_spindleover = 100;
	m_tuyox = _T("");
	m_tuyoy = _T("");
	m_tuyoz = _T("");
	m_fm = 0;
	m_sm = 0;
	//}}AFX_DATA_INIT
	// メモ: LoadIcon は Win32 の DestroyIcon のサブシーケンスを要求しません。
	m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);
}
void CMCOperDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
	//{{AFX_DATA_MAP(CMCOperDlg)
	DDX_Control(pDX, IDC_LOST, m_lost);
	DDX_Control(pDX, IDC_GET, m_get);
	DDX_Control(pDX, IDC_STOP, m_stop);
	DDX_Control(pDX, IDC_DNCEND, m_dncend);
	DDX_Control(pDX, IDC_DNCSTART, m_dncstart);
	DDX_Control(pDX, IDC_PICT3, m_pict3);
	DDX_Control(pDX, IDC_PICT2, m_pict2);
	DDX_Control(pDX, IDC_PICT, m_pict);
	DDX_Control(pDX, IDC_FORCEBUTTON1, m_foce);
	DDX_Control(pDX, IDC_NCSEND5, m_xp);
	DDX_Control(pDX, IDC_NCSEND4, m_ym);
	DDX_Control(pDX, IDC_NCSEND8, m_yp);
	DDX_Control(pDX, IDC_NCSEND6, m_xm);
	DDX_Control(pDX, IDC_NCSEND7, m_zm);
	DDX_Control(pDX, IDC_NCSEND3, m_zp);
	DDX_Control(pDX, IDC_SCROLLBAR_SPINDLE, m_spindlescrol);
	DDX_Control(pDX, IDC_SCROLLBAR_FEED, m_feedscrol);
	DDX_Control(pDX, IDC_NCSEND, m_ncsend);
```

```
        DDX_Control(pDX, IDC_HSSBOPEN, m_hssbopen);
        DDX_Control(pDX, IDC_HSSBCOLSE, m_hssbcolse);
        DDX_Control(pDX, IDC_FEEDHOLD, m_feedhold);
        DDX_Control(pDX, IDC_EMEGSTOP, m_emegstop);
        DDX_Control(pDX, IDC_CICLESTART, m_ciclestart);
        DDX_Text(pDX, IDC_RELY, m_rely);
        DDX_Text(pDX, IDC_RELZ, m_relz);
        DDX_Text(pDX, IDC_ABSX, m_absx);
        DDX_Text(pDX, IDC_ABSY, m_absy);
        DDX_Text(pDX, IDC_ABSZ, m_absz);
        DDX_Text(pDX, IDC_EDIT_SPINDLE, m_spindle);
        DDX_Text(pDX, IDC_EDIT_FEED, m_feed);
        DDX_Text(pDX, IDC_EDIT_MESSA, m_message);
        DDX_Text(pDX, IDC_EDIT_NCPROG, m_ncprog);
        DDX_Text(pDX, IDC_NCNAME, NCNAME);
        DDX_Text(pDX, IDC_MACX, m_macx);
        DDX_Text(pDX, IDC_MACY, m_macy);
        DDX_Text(pDX, IDC_MACZ, m_macz);
        DDX_Text(pDX, IDC_RELX, m_relx);
        DDX_Text(pDX, IDC_REMX, m_remx);
        DDX_Text(pDX, IDC_REMY, m_remy);
        DDX_Text(pDX, IDC_REMZ, m_remz);
        DDX_Scroll(pDX, IDC_SCROLLBAR_FEED, m_feedover);
        DDX_Scroll(pDX, IDC_SCROLLBAR_SPINDLE, m_spindleover);
        DDX_Text(pDX, IDC_tuyoX, m_tuyox);
        DDX_Text(pDX, IDC_tuyoY, m_tuyoy);
        DDX_Text(pDX, IDC_tuyoZ, m_tuyoz);
        DDX_Text(pDX, IDC_EDIT_FEED2, m_fm);
        DDX_Text(pDX, IDC_EDIT_SPINDLE2, m_sm);
        //}}AFX_DATA_MAP
}
BEGIN_MESSAGE_MAP(CMCOperDlg, CDialog)
        //{{AFX_MSG_MAP(CMCOperDlg)
        ON_WM_SYSCOMMAND()
        ON_WM_PAINT()
        ON_WM_QUERYDRAGICON()
```

```
        ON_BN_CLICKED(IDC_HSSBOPEN, OnHssbopen)
        ON_BN_CLICKED(IDC_NCSELECT, OnNcselect)
        ON_BN_CLICKED(IDC_NCSEND, OnNcsend)
        ON_BN_CLICKED(IDC_CICLESTART, OnCiclestart)
        ON_BN_CLICKED(IDC_FEEDHOLD, OnFeedhold)
        ON_BN_CLICKED(IDC_EMEGSTOP, OnEmegstop)
        ON_BN_CLICKED(IDC_HSSBCOLSE, OnHssbcolse)
        ON_WM_TIMER()
        ON_WM_HSCROLL()
        ON_BN_CLICKED(IDC_EXIT, OnExit)
        ON_BN_CLICKED(IDC_NCSEND8, OnNcsend8)
        ON_BN_CLICKED(IDC_NCSEND4, OnNcsend4)
        ON_BN_CLICKED(IDC_NCSEND5, OnNcsend5)
        ON_BN_CLICKED(IDC_NCSEND6, OnNcsend6)
        ON_BN_CLICKED(IDC_NCSEND3, OnNcsend3)
        ON_BN_CLICKED(IDC_NCSEND7, OnNcsend7)
        ON_BN_CLICKED(IDC_FORCEBUTTON1, OnForcebutton1)
        ON_BN_CLICKED(IDC_ZEROONE, OnZeroone)
        ON_BN_CLICKED(IDC_ONE, OnOne)
        ON_BN_CLICKED(IDC_TEN, OnTen)
        ON_BN_CLICKED(IDC_GET, OnGet)
        ON_BN_CLICKED(IDC_LOST, OnLost)
        ON_EN_CHANGE(IDC_EDIT_FEED2, OnChangeEditFeed2)
        ON_EN_CHANGE(IDC_EDIT_SPINDLE2, OnChangeEditSpindle2)
        ON_BN_CLICKED(IDC_STOP, OnStop)
        ON_BN_CLICKED(IDC_X1, OnX1)
        ON_BN_CLICKED(IDC_X10, OnX10)
        ON_BN_CLICKED(IDC_X100, OnX100)
        ON_BN_CLICKED(IDC_DNCSTART, OnDncstart)
        ON_BN_CLICKED(IDC_DNCEND, OnDncend)
        //}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CMCOperDlg メッセージ ハンドラ
```

```
BOOL CMCOperDlg::OnInitDialog()
{
	CDialog::OnInitDialog();
	// "バージョン情報..." メニュー項目をシステム メニューへ追加します。
	// IDM_ABOUTBOX はコマンド メニューの範囲でなければなりません。
	ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
	ASSERT(IDM_ABOUTBOX < 0xF000);
	CMenu* pSysMenu = GetSystemMenu(FALSE);
	if (pSysMenu != NULL)
	{
		CString strAboutMenu;
		strAboutMenu.LoadString(IDS_ABOUTBOX);
		if (!strAboutMenu.IsEmpty())
		{
			pSysMenu->AppendMenu(MF_SEPARATOR);
			pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX,
strAboutMenu);
		}
	}
// このダイアログ用のアイコンを設定します。フレームワークはアプリケーションのメイン
	// ウィンドウがダイアログでない時は自動的に設定しません。
	SetIcon(m_hIcon, TRUE);			// 大きいアイコンを設定
	SetIcon(m_hIcon, FALSE);		// 小さいアイコンを設定
	// TODO: 特別な初期化を行う時はこの場所に追加してください。
	m_hsfeedover = (CScrollBar* )GetDlgItem(IDC_SCROLLBAR_FEED);
	m_hsfeedover ->SetScrollRange(20,200);
	m_hsfeedover ->SetScrollPos(100);

	m_hsspindleover = (CScrollBar* )GetDlgItem(IDC_SCROLLBAR_SPINDLE);
	m_hsspindleover ->SetScrollRange(20,200);
	m_hsspindleover ->SetScrollPos(100);
	m_hssbopen.EnableWindow(TRUE);
	m_hssbcolse.EnableWindow(FALSE);
	m_ciclestart.EnableWindow(FALSE);
	m_feedhold.EnableWindow(FALSE);
	m_ciclestart.EnableWindow(FALSE);
```

```
		m_ncsend.EnableWindow(FALSE);
		m_feedscrol.EnableWindow(FALSE);
		m_spindlescrol.EnableWindow(FALSE);
		m_emegstop.EnableWindow(FALSE);
		m_dncstart.EnableWindow(FALSE);
		m_dncend.EnableWindow(FALSE);
		m_foce.EnableWindow(FALSE);
		m_stop.EnableWindow(FALSE);
		m_xp.EnableWindow(FALSE);
		m_yp.EnableWindow(FALSE);
		m_zp.EnableWindow(FALSE);
		m_xm.EnableWindow(FALSE);
		m_ym.EnableWindow(FALSE);
		m_zm.EnableWindow(FALSE);
		m_get.EnableWindow(FALSE);
		m_lost.EnableWindow(FALSE);
		return TRUE;  // TRUE を返すとコントロールに設定したフォーカスは失われません。
}
void CMCOperDlg::OnSysCommand(UINT nID, LPARAM lParam)
{
		if ((nID & 0xFFF0) == IDM_ABOUTBOX)
		{
			CAboutDlg dlgAbout;
			dlgAbout.DoModal();
		}
		else
		{
			CDialog::OnSysCommand(nID, lParam);
		}
}
// もしダイアログボックスに最小化ボタンを追加するならば、アイコンを描画する
// コードを以下に記述する必要があります。MFC アプリケーションは document/view
// モデルを使っているので、この処理はフレームワークにより自動的に処理されます。
void CMCOperDlg::OnPaint()
{
		if (IsIconic())
```

```
		{
			CPaintDC dc(this); // 描画用のデバイス コンテキスト
			SendMessage(WM_ICONERASEBKGND, (WPARAM)
dc.GetSafeHdc(), 0);
			// クライアントの矩形領域内の中央
			int cxIcon = GetSystemMetrics(SM_CXICON);
			int cyIcon = GetSystemMetrics(SM_CYICON);
			CRect rect;
			GetClientRect(&rect);
			int x = (rect.Width() - cxIcon + 1) / 2;
			int y = (rect.Height() - cyIcon + 1) / 2;
// アイコンを描画します。
			dc.DrawIcon(x, y, m_hIcon);
		}
		else
		{
			CDialog::OnPaint();
		}
}
// システムは、ユーザーが最小化ウィンドウをドラッグしている間、
// カーソルを表示するためにここを呼び出します。
HCURSOR CMCOperDlg::OnQueryDragIcon()
{
		return (HCURSOR) m_hIcon;
}
volatile unsigned int counter ;
int pos_x=0;
static HANDLE hThread = NULL ;
static DWORD threadID = 0 ;
void CMCOperDlg::OnHssbopen() //HSSB OPEN
{
		// TODO: この位置にコントロール通知ハンドラ用のコードを追加してください
		sock = new CComm;
		if (sock->Initialize("210.163.149.97") == false) {
		sock->~CComm();
		AfxMessageBox("socket connect failed");
```

```
            }
// timer
            SetTimer(READTIMER_ID,100,NULL);
            m_hssbopen.EnableWindow(FALSE);
            m_hssbcolse.EnableWindow(TRUE);
            m_feedhold.EnableWindow(TRUE);
            m_ciclestart.EnableWindow(TRUE);
            m_feedscrol.EnableWindow(TRUE);
            m_spindlescrol.EnableWindow(TRUE);
            m_emegstop.EnableWindow(TRUE);
            m_dncstart.EnableWindow(TRUE);
            m_foce.EnableWindow(TRUE);
            m_get.EnableWindow(TRUE);
            m_message += "Connection Established.¥r¥n";
            UpdateData(false);
}
void CMCOperDlg::OnNcselect()
{
          .
          .
          .
}
void CMCOperDlg::OnNcsend()
{
                .
                .
            // NC プログラムを受信し、MC へ転送
                .
                .
                .
            // NC プログラム転送終了
                .
}
void CMCOperDlg::OnCiclestart()
{
}
```

```
void CMCOperDlg::OnFeedhold()
{
}
void CMCOperDlg::OnEmegstop()
{
// TODO: この位置にコントロール通知ハンドラ用のコードを追加してください
      sprintf(sData.command, "Emg. stop from client %d", counter);
      sData.fv[0] = double(counter);
      sData.fv[1] = double(counter+1);
      sData.fv[2] = double(counter+2);
      sock->SockWrite(sData);
      KillTimer(READTIMER_ID);
            m_hssbopen.EnableWindow(TRUE);
            m_hssbcolse.EnableWindow(FALSE);
            m_ciclestart.EnableWindow(FALSE);
            m_feedhold.EnableWindow(FALSE);
            m_ciclestart.EnableWindow(FALSE);
            m_ncsend.EnableWindow(FALSE);
            m_feedscrol.EnableWindow(FALSE);
      m_spindlescrol.EnableWindow(FALSE);
      m_emegstop.EnableWindow(FALSE);
      m_message ="EMEG. STOP";
      UpdateData(false);
}
void CMCOperDlg::OnHssbcolse()
{
// TODO: この位置にコントロール通知ハンドラ用のコードを追加してください
   counter = 0 ;
   KillTimer(READTIMER_ID);
   StartFlag = false;
      m_hssbopen.EnableWindow(TRUE);
      m_hssbcolse.EnableWindow(FALSE);
      m_ciclestart.EnableWindow(FALSE);
      m_feedhold.EnableWindow(FALSE);
      m_ciclestart.EnableWindow(FALSE);
      m_ncsend.EnableWindow(FALSE);
```

```
	m_feedscrol.EnableWindow(FALSE);
	m_spindlescrol.EnableWindow(FALSE);
	m_emegstop.EnableWindow(FALSE);
	m_message ="HSSB CLOSE";
	UpdateData(false);
}
void CMCOperDlg::OnHScroll(UINT nSBCode, UINT nPos, CScrollBar *pScrollBar)
{
		・
		・
}
void CMCOperDlg::OnTimer(UINT nIDEvent)
{
		・
		・
			//力センサグラフ描画
		・
		・
}
void CMCOperDlg::OnExit()
{
// TODO: この位置にコントロール通知ハンドラ用のコードを追加してください
	if(StartFlag == true) {
		KillTimer(READTIMER_ID);
	      }
		CDialog::OnClose();
	 PostMessage(WM_CLOSE, 0, 0);
}
void CMCOperDlg::OnNcsend8()
{
			・
			・
}
void CMCOperDlg::OnNcsend4()
{
}
```

```
void CMCOperDlg::OnNcsend5()
{
}
void CMCOperDlg::OnNcsend6()
{
}
void CMCOperDlg::OnNcsend7()
{
}
void CMCOperDlg::OnForcebutton1()
{
        m_foce.EnableWindow(FALSE);
        m_stop.EnableWindow(TRUE);
// TODO: この位置にコントロール通知ハンドラ用のコードを追加してください
    CWnd* h=GetDlgItem(IDC_PICT);
    CDC*pDC=h->GetDC();
    CFont newFont;
    CFont* pOldFont;
        newFont.CreateFont(10,
                0,
                0,
                0,
                FW_NORMAL,
                FALSE,
                FALSE,
                0,
                SHIFTJIS_CHARSET,
                OUT_DEFAULT_PRECIS,
                CLIP_DEFAULT_PRECIS,
                DEFAULT_QUALITY,
                DEFAULT_PITCH | FF_MODERN,
                "ＭＳ ゴシック");
        pDC->MoveTo(18,20);
        pDC->LineTo(22,20);
        pOldFont=(CFont*)pDC->SelectObject(&newFont);
        pDC->TextOut(5,15,dummy,strlen(dummy));
```

```
pDC->MoveTo(18,80);
pDC->LineTo(22,80);
pDC->TextOut(5,75,dummy,strlen(dummy));
pDC->MoveTo(18,140);
 pDC->LineTo(22,140);
 pDC->TextOut(5,135,dummy,strlen(dummy));
pDC->MoveTo(20,0);
pDC->LineTo(20,50);
pDC->MoveTo(20,60);
pDC->LineTo(20,110);
pDC->MoveTo(20,120);
pDC->LineTo(20,170);
pDC->MoveTo(20,40);
pDC->LineTo(335,40);
pDC->MoveTo(20,100);
pDC->LineTo(335,100);
pDC->MoveTo(20,160);
pDC->LineTo(335,160);
pDC->MoveTo(120,38);
pDC->LineTo(120,42);
pDC->MoveTo(220,38);
pDC->LineTo(220,42);
pDC->MoveTo(320,38);
pDC->LineTo(320,42);
pDC->MoveTo(120,98);
pDC->LineTo(120,102);
pDC->MoveTo(220,98);
pDC->LineTo(220,102);
pDC->MoveTo(320,98);
pDC->LineTo(320,102);
pDC->MoveTo(120,158);
pDC->LineTo(120,162);
pDC->MoveTo(220,158);
pDC->LineTo(220,162);
pDC->MoveTo(320,158);
pDC->LineTo(320,162);
```

```
            tu=20;
            xa=20;
            ya=40; yb=100; yc=160;
            l=1;
      UpdateData(false);
}
void CMCOperDlg::OnZeroone()
{
            // TODO: この位置にコントロール通知ハンドラ用のコードを追加してください
            V=0.1;
            Y=200;
            sprintf(dummy,"%d",Y);
            CDC*pDC=m_pict.GetDC();
            CRect myRECT;
            m_pict.GetClientRect(myRECT);
            pDC->FillSolidRect(myRECT,RGB(255,255,255));
//y
            CDC*pDC2=m_pict2.GetDC();
            CRect myRECT2;
            m_pict2.GetClientRect(myRECT2);
            pDC2->FillSolidRect(myRECT2,RGB(255,255,255));
//z
            CDC*pDC3=m_pict3.GetDC();
            CRect myRECT3;
            m_pict3.GetClientRect(myRECT3);
            pDC3->FillSolidRect(myRECT3,RGB(255,255,255));
            UpdateData(false);
            OnForcebutton1() ;
}
void CMCOperDlg::OnOne()
{
// TODO: この位置にコントロール通知ハンドラ用のコードを追加してください
            V=1;
            Y=20;
            sprintf(dummy,"%d",Y);
            CDC*pDC=m_pict.GetDC();
```

```
		CRect myRECT;
		m_pict.GetClientRect(myRECT);
		pDC->FillSolidRect(myRECT,RGB(255,255,255));
//y
		CDC*pDC2=m_pict2.GetDC();
		CRect myRECT2;
		m_pict2.GetClientRect(myRECT2);
		pDC2->FillSolidRect(myRECT2,RGB(255,255,255));
//z
		CDC*pDC3=m_pict3.GetDC();
		CRect myRECT3;
		m_pict3.GetClientRect(myRECT3);
		pDC3->FillSolidRect(myRECT3,RGB(255,255,255));
		UpdateData(false);
		OnForcebutton1() ;
}
void CMCOperDlg::OnTen()
{
		// TODO: この位置にコントロール通知ハンドラ用のコードを追加してください
		V=10;
		Y=2;
		sprintf(dummy,"%d",Y);
		CDC*pDC=m_pict.GetDC();
		CRect myRECT;
		m_pict.GetClientRect(myRECT);
		pDC->FillSolidRect(myRECT,RGB(255,255,255));
//y
		CDC*pDC2=m_pict2.GetDC();
		CRect myRECT2;
		m_pict2.GetClientRect(myRECT2);
		pDC2->FillSolidRect(myRECT2,RGB(255,255,255));
//z
		CDC*pDC3=m_pict3.GetDC();
		CRect myRECT3;
		m_pict3.GetClientRect(myRECT3);
		pDC3->FillSolidRect(myRECT3,RGB(255,255,255));
```

```
            UpdateData(false);
            OnForcebutton1() ;
}
void CMCOperDlg::OnGet()
{
// TODO: この位置にコントロール通知ハンドラ用のコードを追加してください
            m_get.EnableWindow(FALSE);
            m_lost.EnableWindow(TRUE);
            sprintf(sData.command, "P%d", counter);
            sock->SockWrite(sData);
            m_message ="力センサーのデータを取得します";
            SetTimer(1,100,NULL);
            UpdateData(false);
}
void CMCOperDlg::OnLost()
{
// TODO: この位置にコントロール通知ハンドラ用のコードを追加してください
            m_get.EnableWindow(TRUE);
            m_lost.EnableWindow(FALSE);
      sprintf(sData.command, "O%d", counter);
      sock->SockWrite(sData);
            m_message ="力センサーのデータを破棄します";
            KillTimer(1);
            UpdateData(false);
}
void CMCOperDlg::OnChangeEditFeed2()
{
// TODO: これが RICHEDIT コントロールの場合、コントロールは、 lParam マスク
// 内での論理和の ENM_CHANGE フラグ付きで CRichEditCrtl().SetEventMask()
// メッセージをコントロールへ送るために CDialog::OnInitDialog() 関数をオーバー
// ライドしない限りこの通知を送りません。
// TODO: この位置にコントロール通知ハンドラ用のコードを追加してください
            CEdit* str =(CEdit*)GetDlgItem(IDC_EDIT_FEED2);
            char data[10];
            int m;
            m=str->GetLine(0,data,10);
```

```
        data[m]=0;
        m_fm=atoi(data);
        UpdateData(false);
        if(m_fm==0){
        AfxMessageBox("0 では動きません");
        }
}
void CMCOperDlg::OnChangeEditSpindle2()
{
// TODO:  これが RICHEDIT コントロールの場合、コントロールは、 lParam マスク
// 内での論理和の ENM_CHANGE フラグ付きで CRichEditCrtl().SetEventMask()
// メッセージをコントロールへ送るために CDialog::OnInitDialog() 関数をオーバー
// ライドしない限りこの通知を送りません。
// TODO:  この位置にコントロール通知ハンドラ用のコードを追加してください
        CEdit* str =(CEdit*)GetDlgItem(IDC_EDIT_SPINDLE2);
                char count[10];
                int mm;
                mm=str->GetLine(0,count,10);
                count[mm]=0;
                m_sm=atoi(count);
                UpdateData(false);
                if (m_sm>6000){
                AfxMessageBox("主軸回転数は 0 から 6000 の間");
                }
}
void CMCOperDlg::OnStop()
{
// TODO:  この位置にコントロール通知ハンドラ用のコードを追加してください
        l=0;
        CDC*pDC=m_pict.GetDC();
        CRect myRECT;
        m_pict.GetClientRect(myRECT);
        pDC->FillSolidRect(myRECT,RGB(255,255,255));
//y
        CDC*pDC2=m_pict2.GetDC();
        CRect myRECT2;
```

```
            m_pict2.GetClientRect(myRECT2);
            pDC2->FillSolidRect(myRECT2,RGB(255,255,255));
//z
            CDC*pDC3=m_pict3.GetDC();
            CRect myRECT3;
            m_pict3.GetClientRect(myRECT3);
            pDC3->FillSolidRect(myRECT3,RGB(255,255,255));
            m_foce.EnableWindow(TRUE);
            m_stop.EnableWindow(FALSE);
            UpdateData(false);
}
void CMCOperDlg::OnX1()
{
// TODO: この位置にコントロール通知ハンドラ用のコードを追加してください
            n=0.001;
}
void CMCOperDlg::OnX10()
{
// TODO: この位置にコントロール通知ハンドラ用のコードを追加してください
            n=0.01;
}
void CMCOperDlg::OnX100()
{
// TODO: この位置にコントロール通知ハンドラ用のコードを追加してください
            n=0.1;
}
void CMCOperDlg::OnDncstart()
{
// TODO: この位置にコントロール通知ハンドラ用のコードを追加してください
            sprintf(sData.command, "A dnc start %d", counter);
            m_message="DNC 運転をスタートします";
            sock->SockWrite(sData);
            m_dncstart.EnableWindow(FALSE);
            m_dncend.EnableWindow(TRUE);
            m_yp.EnableWindow(TRUE);
            m_xp.EnableWindow(TRUE);
```

```
        m_zp.EnableWindow(TRUE);
        m_xm.EnableWindow(TRUE);
        m_ym.EnableWindow(TRUE);
        m_zm.EnableWindow(TRUE);
        UpdateData(false);
}
void CMCOperDlg::OnDncend()
{
// TODO: この位置にコントロール通知ハンドラ用のコードを追加してください
        sprintf(sData.command, "B dnc    end%d", counter);
        m_message="DNC 運転を終了します";
        sock->SockWrite(sData);
        m_dncstart.EnableWindow(TRUE);
        m_dncend.EnableWindow(FALSE);
        m_yp.EnableWindow(FALSE);
        m_xp.EnableWindow(FALSE);
        m_zp.EnableWindow(FALSE);
        m_xm.EnableWindow(FALSE);
        m_ym.EnableWindow(FALSE);
        m_zm.EnableWindow(FALSE);
        UpdateData(false);
        }
```