修士論文

熱変形能動補償型高精密マシニングセンタによる加工精度向上の研究

１～１１４ページ　完

平成１５年　1月３０日

指導教官　　長尾高明教授

知能機械コース　１０５５０４２

更谷　謙仁

目次

目次

目次

第 1 章

序論

1.1　　研究の背景

古代から我々人類は頭と手を使って人間の暮らしに役立つものを作ってきた。やがて、道具により道具を作るようになってきた。そして、機械化の進んだ現在、道具を作る道具も機械となった。それはマザーマシンと呼ばれ、母性原理の働く生産の場で、高精度の製品を生産するために高水準の生産性や信頼性が要求されている。

時代は、高度経済成長を終え、空前の低成長時代に突入して久しくなる。また、物質的な豊かさを手に入れた我々の中では、精神的な豊かさを求めて環境問題への興味・関心が非常に高まっている。このような時代背景の中、生産の現場にはより高効率な製品、より高精度な製品への要求がますます増加している。

しかし、高精度な生産を担ってきた熟練労働者は減少の一途をたどっている。その背景には、熟練者でなくては高い水準の加工ができなかった汎用工作機械から工作機械のＮＣ化によって、本やマニュアルを読んだだけで一定の加工ができるようになり、また 1 日 24 時間の生産が可能になったことも一因である。

しかし、熟練者が全く必要でなくなったのではなく、高い精度が要求される加工では、熟練者の経験は必要不可欠である。

加工の高精度化、高信頼性への要求が高まる中で、工作機械が熱や加工反力より変形し精度が低下する問題は未だ解決されておらず、熟練者の経験に依存しており、高生産性を実現できていない。

実際、工作機械の熱変形による精度の低下に対し、加工中に発生する熱を冷却することによって抑えてはいるが、発生する全ての熱を解消することは不可能である。熟練者が経験に基づいて補正を加えるか、あるいは機械を事前に数時間ほど空運転することによって熱変形が安定するのを待って使用するといった方法で対処しているのが現状である。

熱変形そのものを抑制する手段はいくつか講じられているが、熱伝播、熱膨張を全くなくすことが出来ない限り精度低下は起こってしまう。

そこで、発熱を抑制するだけでなく工作機械の熱変形を補償し、熱環境に依存しない工作機械をつくりだそうという研究があり、本研究はマシニングセンタにおいて、熱変形による精度低下を低減する試みである。

1.2　　従来の研究

工作機械の精度を下げる要因のうち、熱変形の占める割合がかなり大きいことは古くから知られていた。それ以来、熱変形の要因を調べる研究がなされてきた。現在では、有限要素法を利用して機械本体の熱変形挙動を解析する研究なども挙げられる。

一方、熱変形に対抗して加工精度の向上を目指す試みも数多くある。それらは大きく２つに分類される。１つには熱変形を抑制する研究。もう１つは熱変形による精度低下に対し補正を行う研究が試みられている。

前者には、コンクリートのように熱伝導率の小さい材質を構造部に用いて熱変形を抑えようとする研究などが挙げられる。後者の例としては、圧電素子を用いて熱変形を補正しようとした研究がある。

1.3　　加工の知能化の概念

加工の目的は所望の寸法をもつ製品をできるだけ正確に効率よく作り出すことであるが、「正確に」「効率よく」を実現するのは、実際には非常に困難な問題である。その原因として挙げられるのが図 1.1 に示すような加工中に生じる様々な物理現象である。

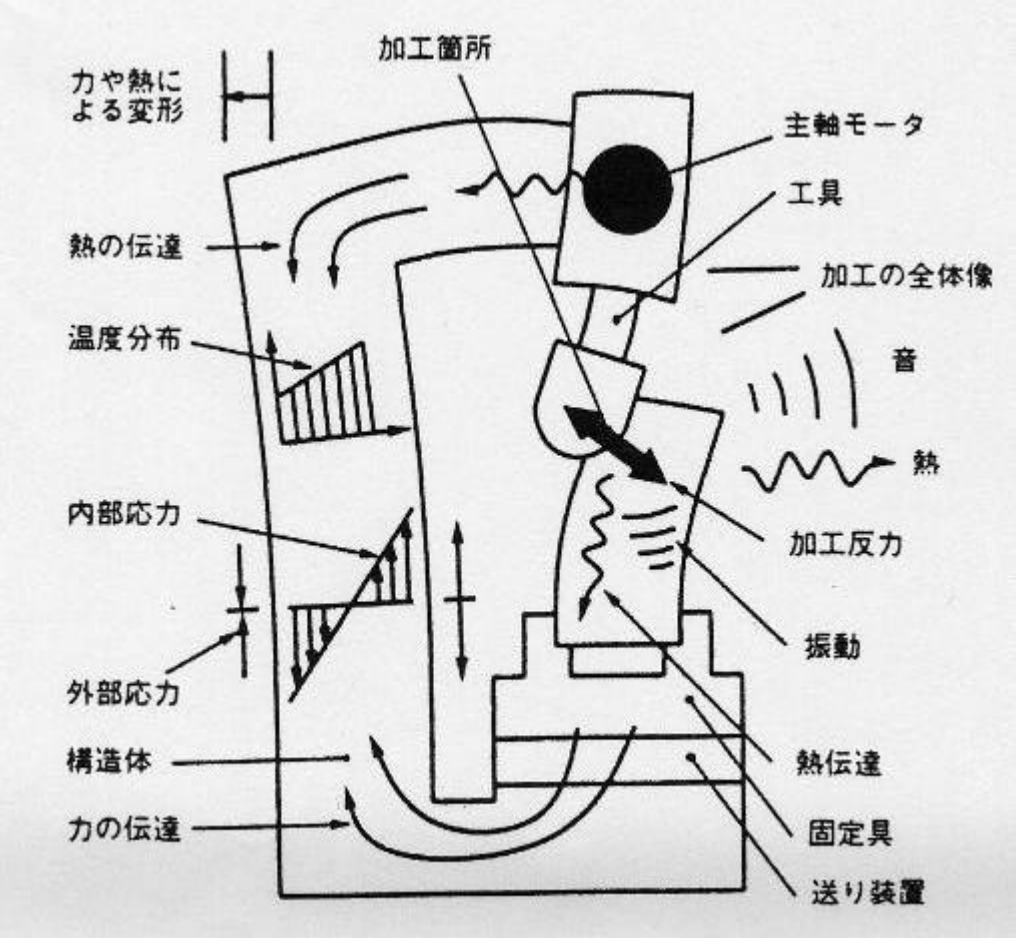


図 1.1　加工中に生じる物理現象

マシニングセンタを始めとする工作機械はこれらの物理現象によって生じる力と熱によって変形し、同時に歪、温度、音などの情報を発生する。そこで、これらの情報をセンサによって取り込み、設計に関する知識に基づいて処理することで目標値と実際のずれを予測し、機械に設置したアクチュエータによってそのずれを修正することができれば製品を正確に作り出すことが可能になる。また、センサ情報、アクチュエータの動作内容、加工の結果をデータベース化し、それに基づいて予測のための内部モデルを修正することで、システム自体が学習、自己進化することも実現可能である。このような考えを「加工の知能化」と呼びその基本構成は図 1.2 のダイヤグラムに示すようなものとなる。

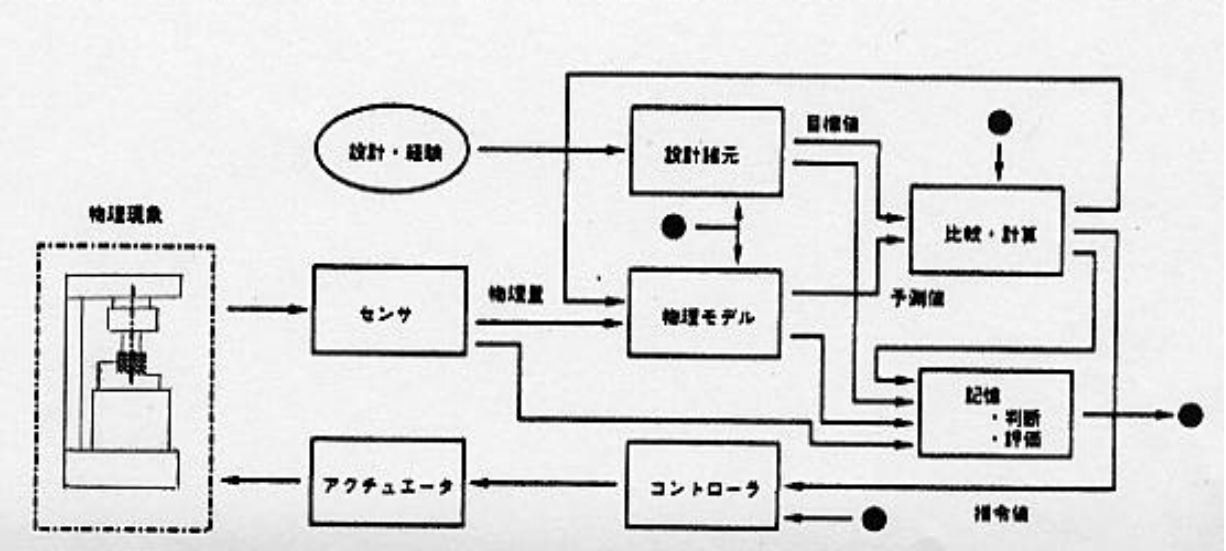


図 1.2 加工の知能化の基本構成

1.3.1　マシニングセンタにおける熱変形能動補償の意義

前途したように様々に物理現象が加工中に生じているわけだが、このうちマシニングセンタの加工精度を低下させる大きな要因として、加工反力による変形、熱による変形を挙げることができる。

このうち加工反力による変形に対処するために、従来は工作機械本体の剛性

を高めることによって変形を抑えるという方法がとられてきた。しかしこの方法では、材料の剛性に限界があることや、構造体の重量が重くなり過ぎることなどの制約があり、根本的な解決には至ってない。

一方、もう１つの要因として取り上げた熱変形については、従来は発熱する部分を冷やす、あるいは熱が他の部分に伝わるのを出来るだけ抑制する、といった対策を施すことで少しでも変形を軽減しようとしていたが、発生した熱を全て冷却することが出来ない以上、おのずから限界ができてしまう。事実、加工することなく単に主軸を回転させるだけでも、わずか 2 時間で 40～50μｍもの主軸変位が発生する。

実際には、主軸モータの発熱に加え、工具と披加工物との間に生じる加工発熱、周囲の環境の変化に伴う熱の移動など、さらに多くの熱変形の要因が存在しており、これらを全て取り除くことは不可能である。

また、マシニングセンタのように多くの部品と複雑な構造で構成されるものでは、加工反力による変形と熱変形を具体的に解析し、モデル化することは非常に困難である。よって、モデル化による熱変形補正は複雑な熱変形を十分に補正できない。

しかし、実際問題として加工の高精密化が必要されている現状では、これらの変形による加工精度の低下は解決されなければならない問題である。そこで本研究では、「加工の知能化」の概念に基づいて、マシニングセンタの熱によって生じる変形を能動的に補正することを試みていく。

1.3.2　高精密マシニングセンタの有すべき機能

以上のことを踏まえて、加工の知能化を実現するために、マシニングセンタが備えているべき機能を抽出すると以下のようになる。

1. 自ら変形情報を把握するためのセンサを有すること。
2. センサによって得られた情報に基づいて、加工中にリアルタイムに自己の変形状態を導き出すモデルを持っていること。
3. モデルによって得られた構造体変位を打ち消すためのアクチュエータを有すること。

第１章　　序論

本研究では 1.変形センサは自分たちで設計・開発、製作し、2.変形センサからのデータをリアルタイムに取得し熱アクチュエータを稼動させる。3.加熱板と冷却ファンを設置する。

1.4　　本研究の目的

1. 変形センサの設計開発及びデータ取得のソフト作成。
2. 主軸回転時の変位を測定し、変位量によって熱アクチュエータを使用し変位量を軽減させる。
3. 上記の目標を達成するために、変形センサの仕組みを学習し、また、マシニングセンタが熱変形を起こすメカニズムを解明する。

1.5　本論文の構成

本論文は全 7 章から構成される。

第 1 章では研究の背景および目的について述べる。その中で、本研究で使用する高精密マシニングセンタの開発並びにその研究の根本的思想となっている「加工の知能化」の概念について説明し、本研究の主題となるマシニングセンタの熱変形能動補償の意義と、そのために必要とされる機能について検討する。

第 2 章では、マシニングセンタによる高精密加工を実現するために変形センサの設計開発・製作、歪測定プログラムなどのハードウェアとソフトウェアの構成について述べる。

第 3 章では、高精密加工を実現するための姿勢制御の手段について述べる。

第 4 章では、高精密マシニングセンタの有する姿勢制御のためのハードウェアについて、性能評価を行う。

第 5 章では、熱アクチュエータによる姿勢制御実験を行う。

第 6 章では、総括的な考察と今後の展望を示す。

第 7 章では、本論文の結論を述べ本研究のまとめとする。

第 2 章

高精密マシニングセンタの構成

2.1　システムの概略

本研究に用いた高精密加工システムの概略を図 2.1 に示す。

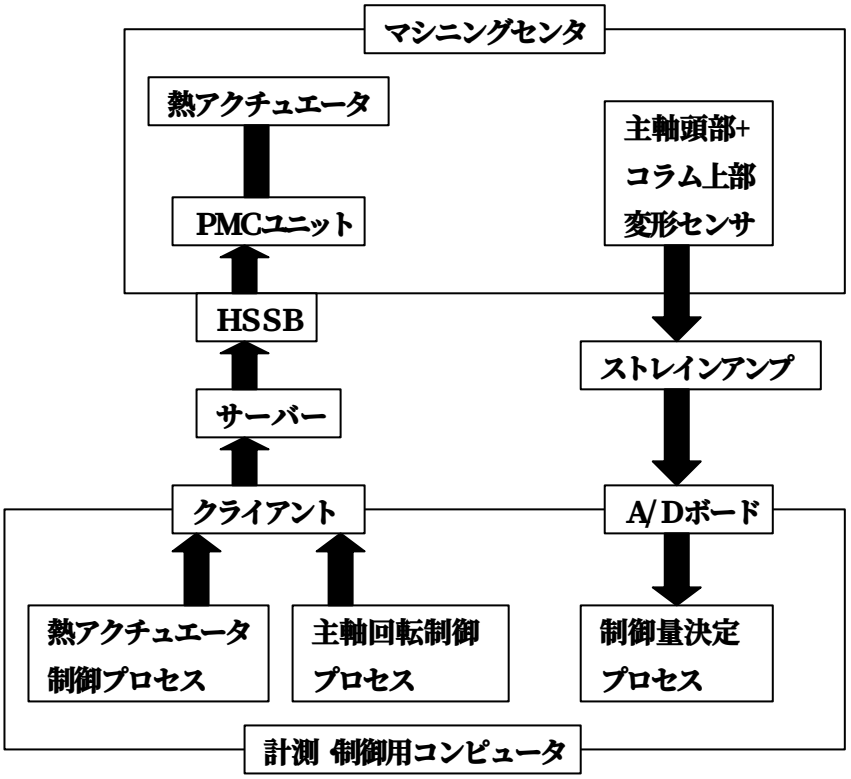


図 2.1　システムの概略

次節以降で、それぞれの機能について述べていく。

2.2　マシニングセンタ

図 2.2、表 2.1 に本研究に用いた高精密マシニングセンタの外観及びその仕様を示す。

図 2.2　高精密マシニングセンタ

このマシニングセンタは、大阪機工（株）製のＶＭ４－Ⅱのマシニングセンタに加工の知能化を実現するために特殊な機構を装備したものである。

その機構とは以下のようなものである。

・　コラム下部に熱アクチュエータ（加熱、冷却用各１４箇所）を装備しており、熱変形に対してアクティブな補正が可能である

・　マシニングセンタの各部に合計 5 本の変形センサが取り付けられており、マシニングセンタの変形状態や熱アクチュエータの作動状況をリアルタイムに監視できる。

テーブル作業面	mm	８００×４００
最大移動量 テーブル左右（X） サドル前後　（Y） 主軸頭上下　（Z）	mm	 ６３０ ４１０ ４６０
主軸回転速度	ｒｐｍ	２５～６０００
早送り速度	m/min	２４　（X，Y） １２　（Z）
切削送り速度	m/min	１～１００００
ジョグ送り速度	m/min	２０００

表 2.1　ＯＫＫマシニングセンタＶＭ４－Ⅱの仕様

・　主軸モータ‐主軸頭部間、主軸頭部‐コラム上部、コラム上部‐コラム下部間などに熱遮断体が設けられており、熱の伝播を極力小さくしている。
・　外部制御装置からマシニングセンタの各種機能が使用できる。
・　作業台にはX，Y軸フェイルセイフテーブルが設けられており、過大な荷重が生じた時マシニングセンタ本体を守る。
・　オープンＣＮＣマシニングセンタなのでインターネットに繋がっており遠隔操作が可能。

以上の機構を装備したマシニングセンタの構成の概略を図 2.3 に示す。

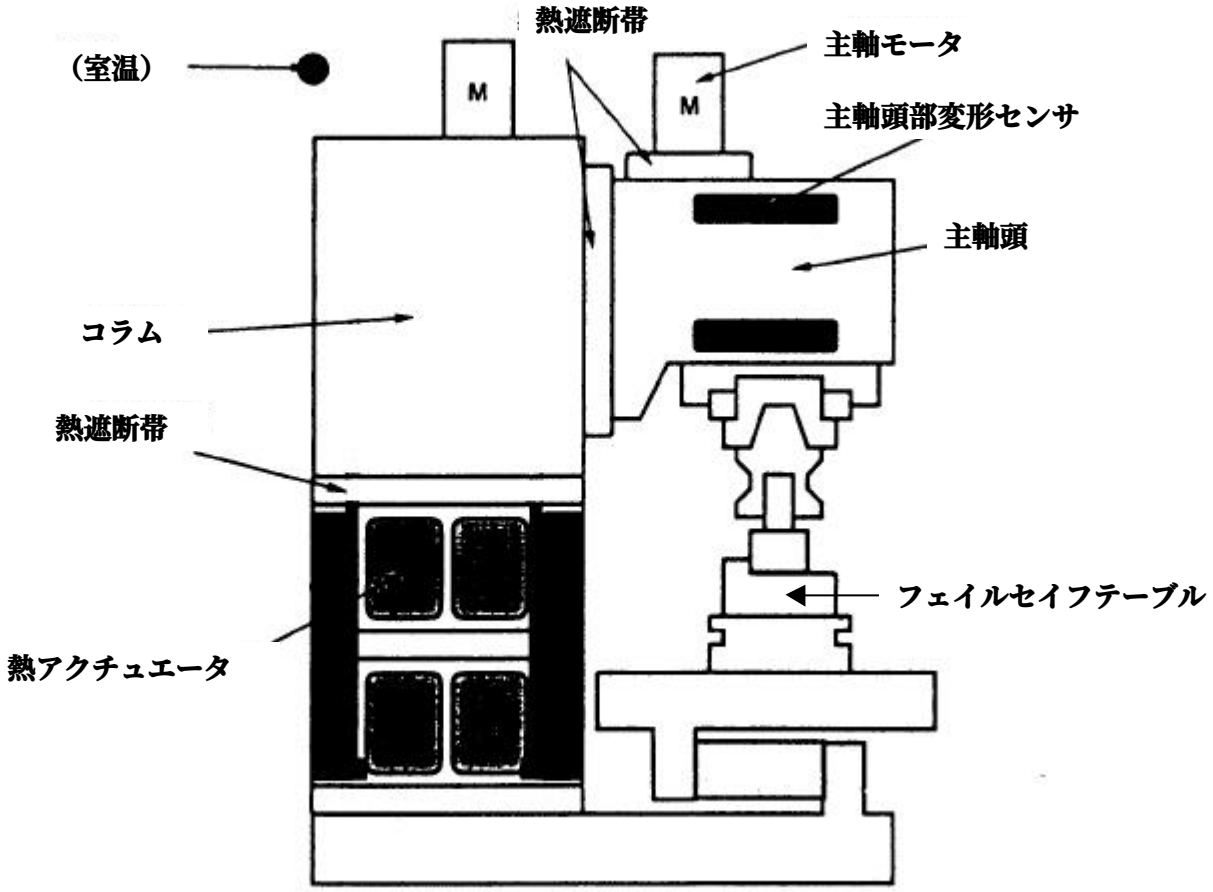


図 2.3　高精密マシニングセンタの構成の概略

2.3　変形検出機構（変形センサ）

2.3.1　システムの概略

図 2.4 に変形検出システムの概略を示す。

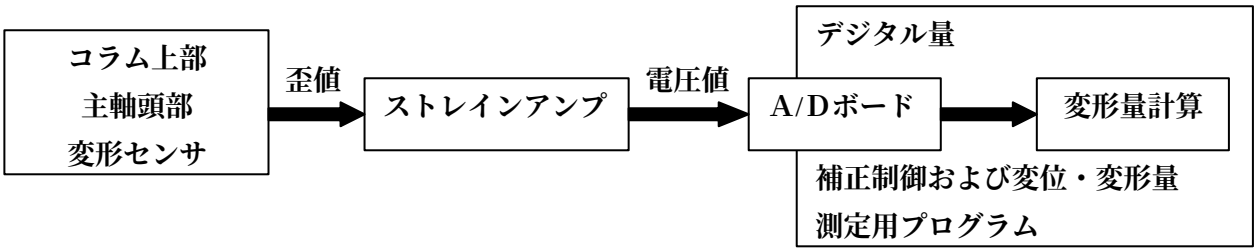


図 2.4　変形量検出システム

2.3.2　変形センサとは

3 次元形状測定機であるマシニングセンタでは、切削工具付近のスピンドルモータの発熱や切削液の噴霧冷却によって、構造体であるコラムがたわみ、工具位置が変化する。このとき、その構造体の熱変形と工具位置との因果関係を知っていれば、熱変形を測定することで工具位置が測定できる。そこで、我々は変形センサをマシニングセンタに設置し熱変形を測定することにした。変形センサとは構造体の 2 点間に固定された 2 個の弾性ブロックと、その 2 個の弾性ブロックを結ぶ剛体連結棒とからなる。この 2 個のブロックは、片方または両方に平行平板構造を配していることが特徴であり、この構造によって図 2.5 の連結棒の軸方向（x 方向）のみに柔に弾性変形できる。構造体が変形すると、連結棒は x 方向に剛であるから、図に示すようにブロックの平行平板だけが変形し、これを検出素子で測定する。

変形センサの主要機能は、構造体の 2 点間の微小変形をセンサ内の平行平板の変形に集中させ、任意の方向の変形をそのほかの方向のそれと干渉させずに、力によって生じた変形と熱によって生じた変形とを分離して、その変形を各種の検出素子で検出することである。

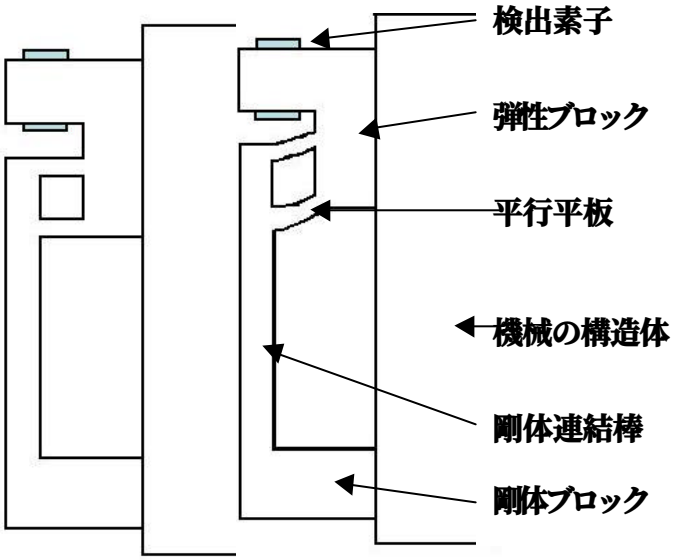


図 2.5　変形センサの変形検出の原理

2.3.3　変形センサの設計

今回の研究では変形センサの設計から自分たちで行うことにした。設計図を図 2.6、2.7、2.8 に示す。また設計図をもとに ANSYS を使用して引っ張りと圧縮のシミュレーションを行った。

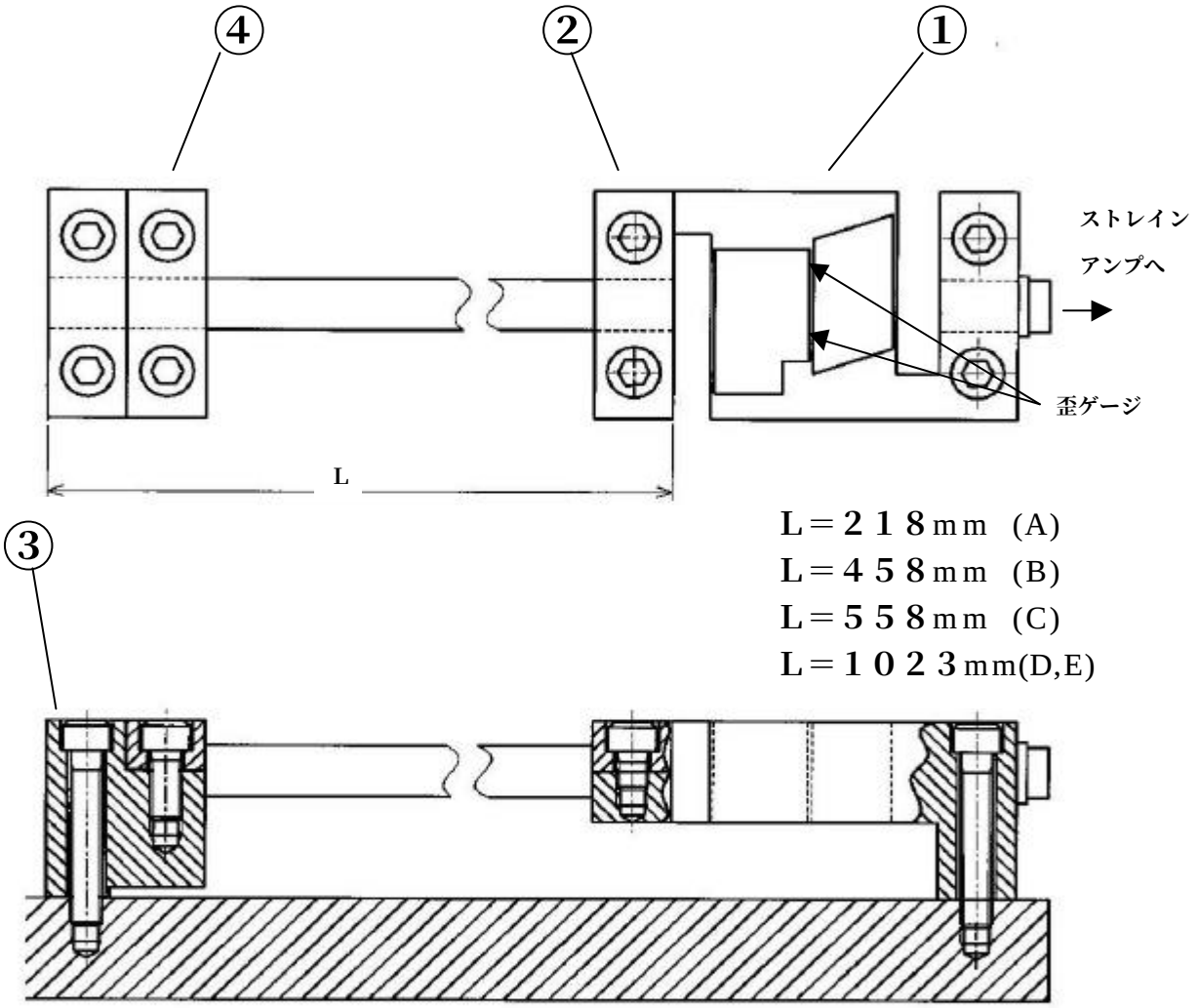


図 2.6　変形センサの構造

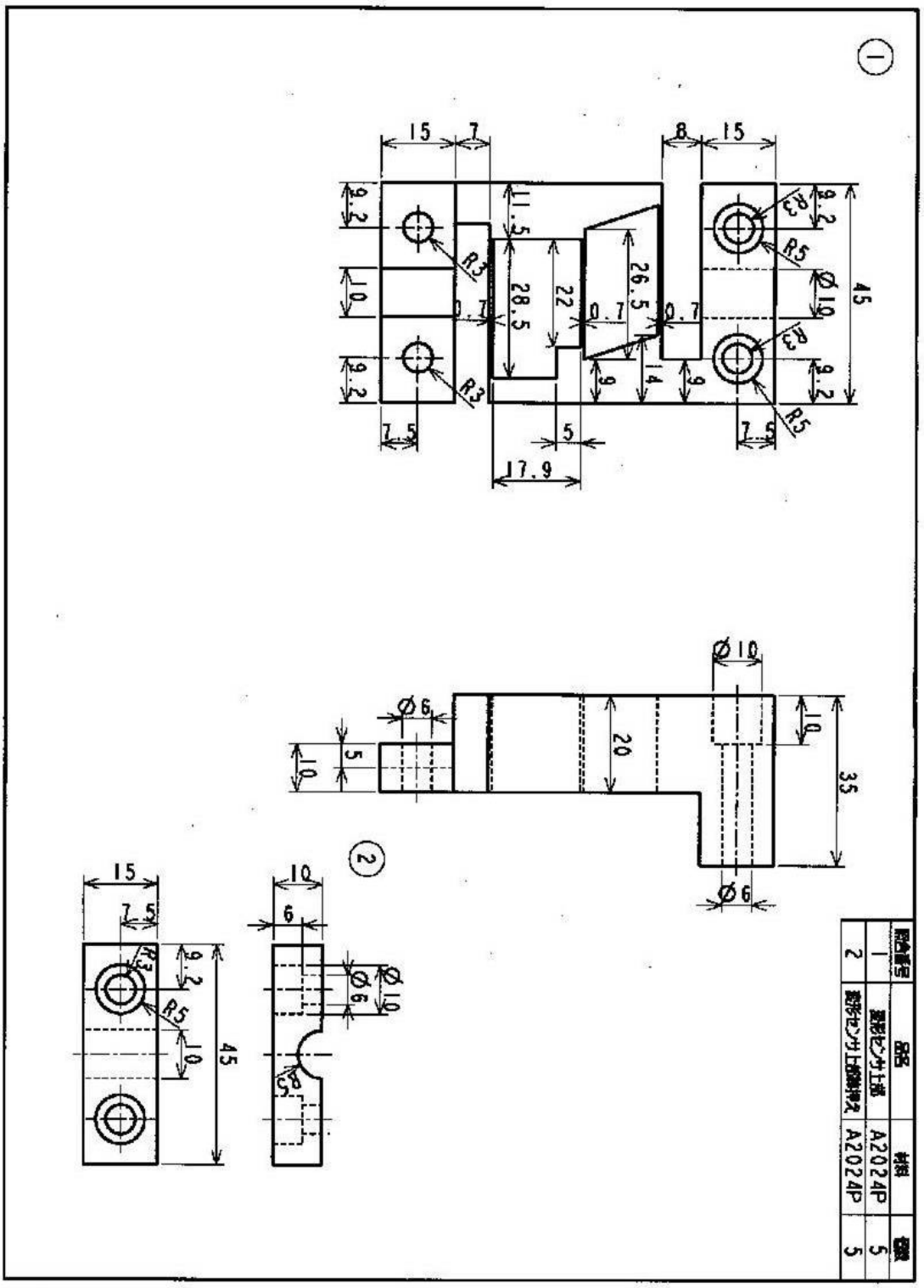


図 2.7　変形センサの設計図 1

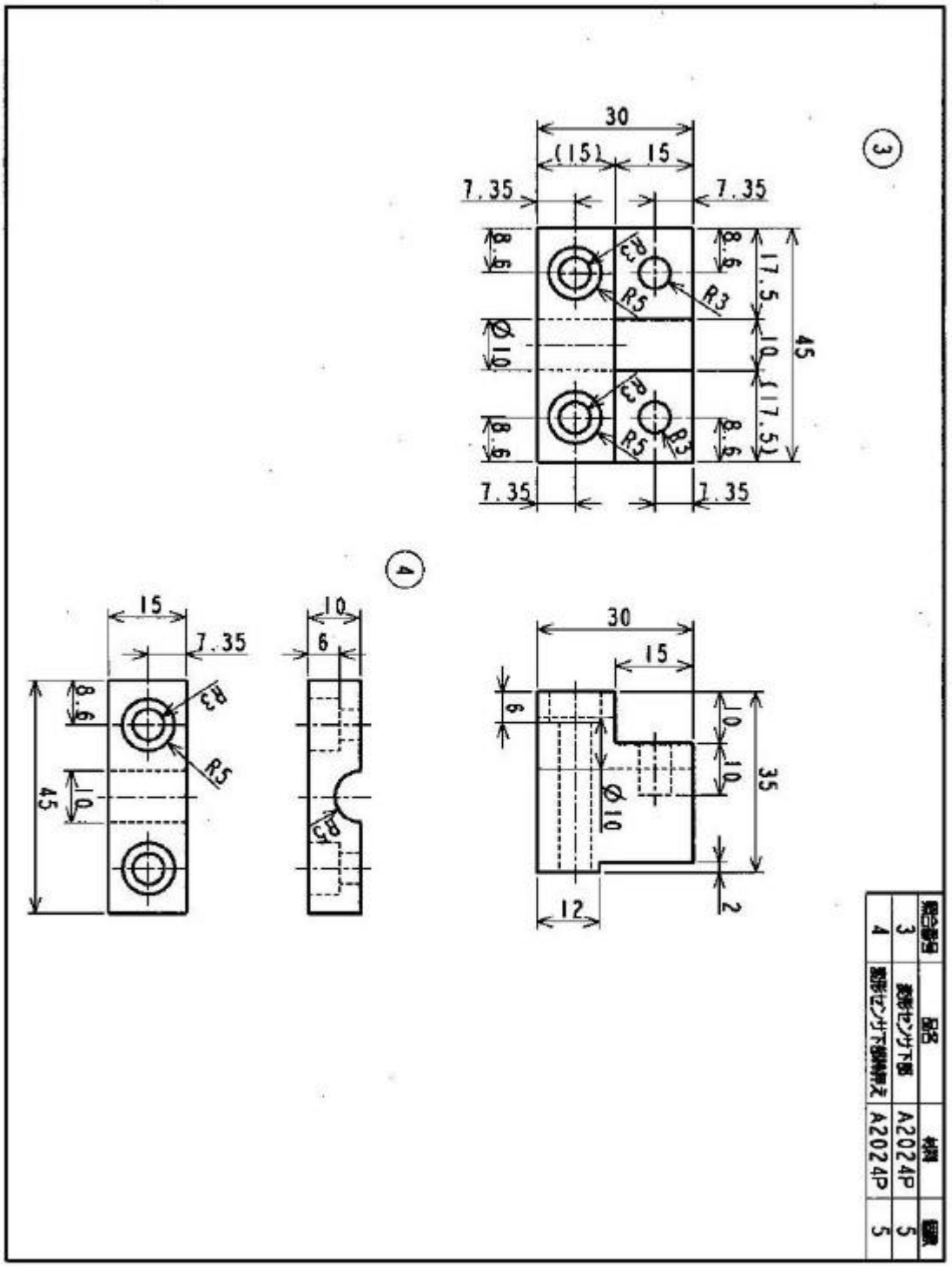


図 2.8　変形センサの設計図２

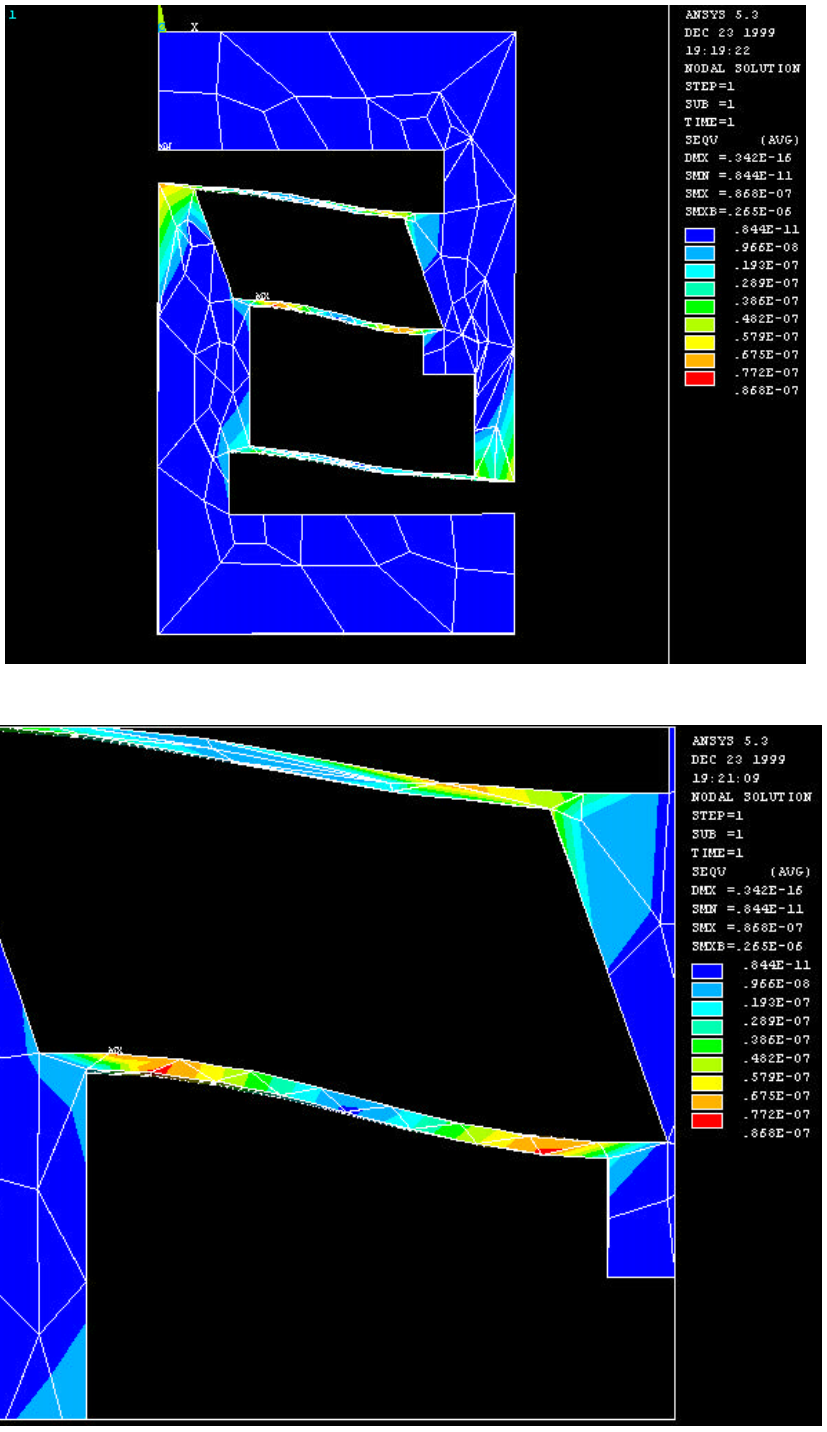


図 2.9　ANSYS による圧縮のシミュレーション

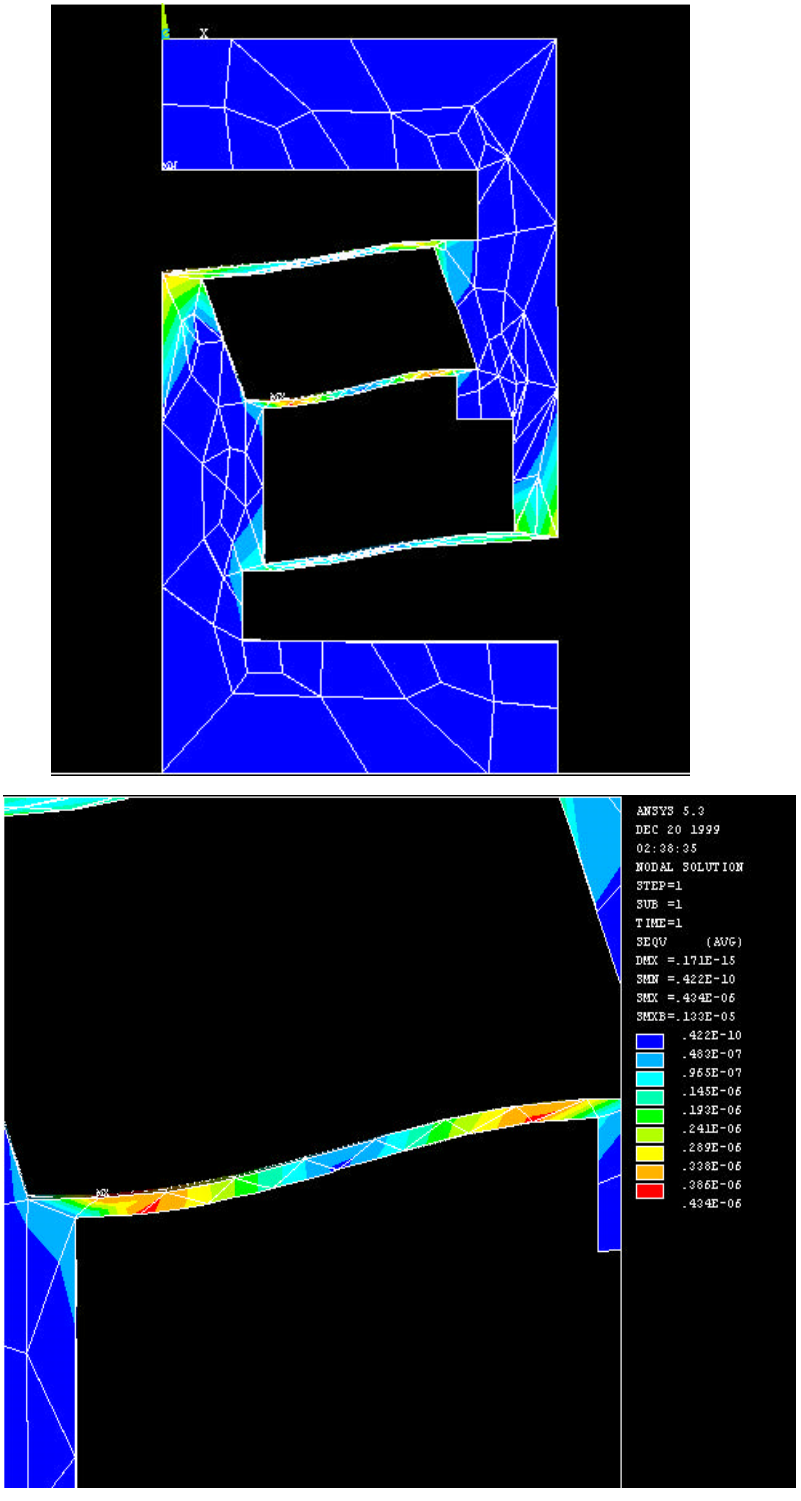


図 2.10　ANSYS による引っ張りのシミュレーション

2.3.4　変形センサの製作

今回の変形センサの加工は平行平板部分をワイヤーカット、それ以外の部分をマシニングセンタを使用して製作した。

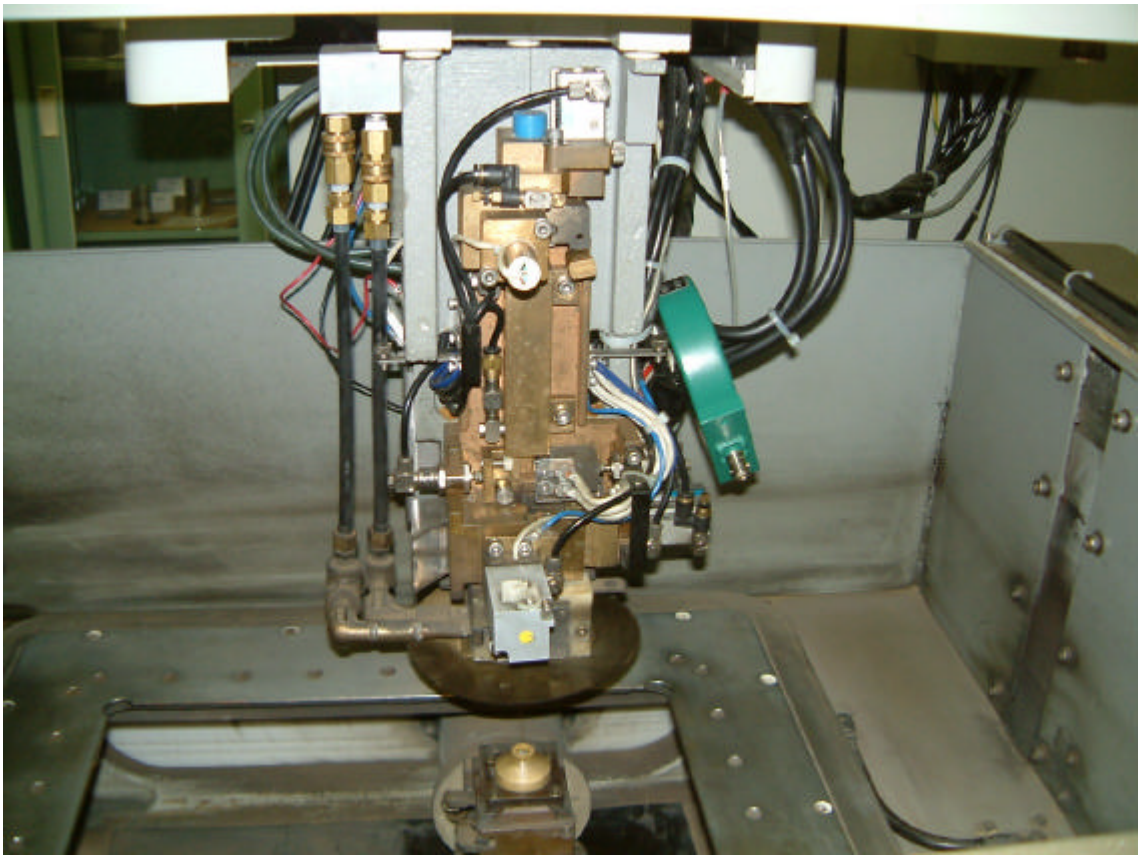


図 2.11　ワイヤーカット放電加工機

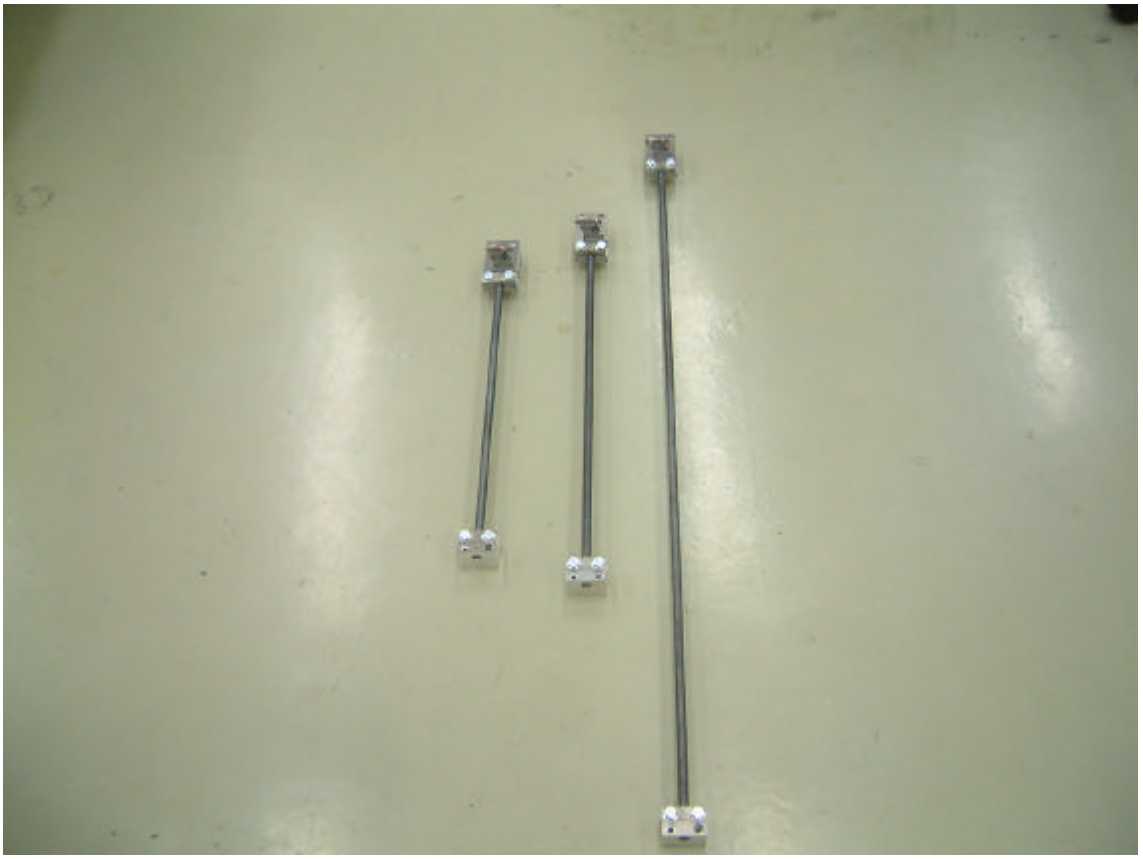


図 2.12　変形センサ

2.3.4.1　ストレインゲージ

今回使用したストレインゲージは(株)KYOWA のKFG-2-120-C1-23である。仕様は表 2.2 に示す。

TEMPERATURE COMPENSATION	ALUMINUM
GAGE LENGTH	2 mm
GAGE RESISTANCE(24℃,50%RH)	119.8 ±0.2 Ω
GAGE FACTOR(24℃,50%RH)	2.12 ±1.0 %
ADOPTABLE THERMAL WXPANSION	23.4 PPM/℃
TRANSVERSE SENSITIVITY(24℃,50%RH)	0.7 %
APPLICABLE GAGE CEMENT	CC-33A,PC-6

表 2.2　KYOWA KFG-2-120-C1-23 の仕様

ストレインゲージの張り方は曲げ歪の検出でき引っ張り、圧縮歪を消去でき温度補償のある 2 アクティブゲージ法を採用する。

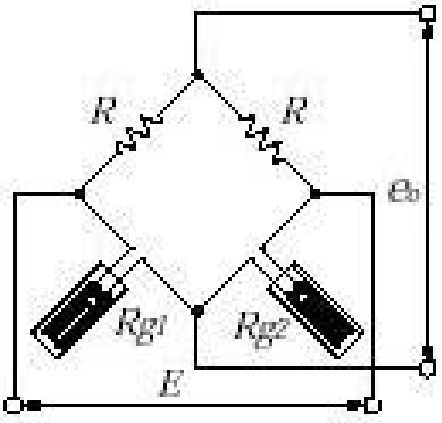


図 2.13　2 アクティブゲージ法の回路図

2.4 ストレインアンプ

図 2.12 に示すのは日本電気三栄（株）のストレインアンプである。今回使用したＡＨ1108 は収納ケースに 8 プラグインのユニットが収納可能である。プラグインユニットにはＡＣストレインアンプを 5 台、熱電対アンプを 3 台装備した。

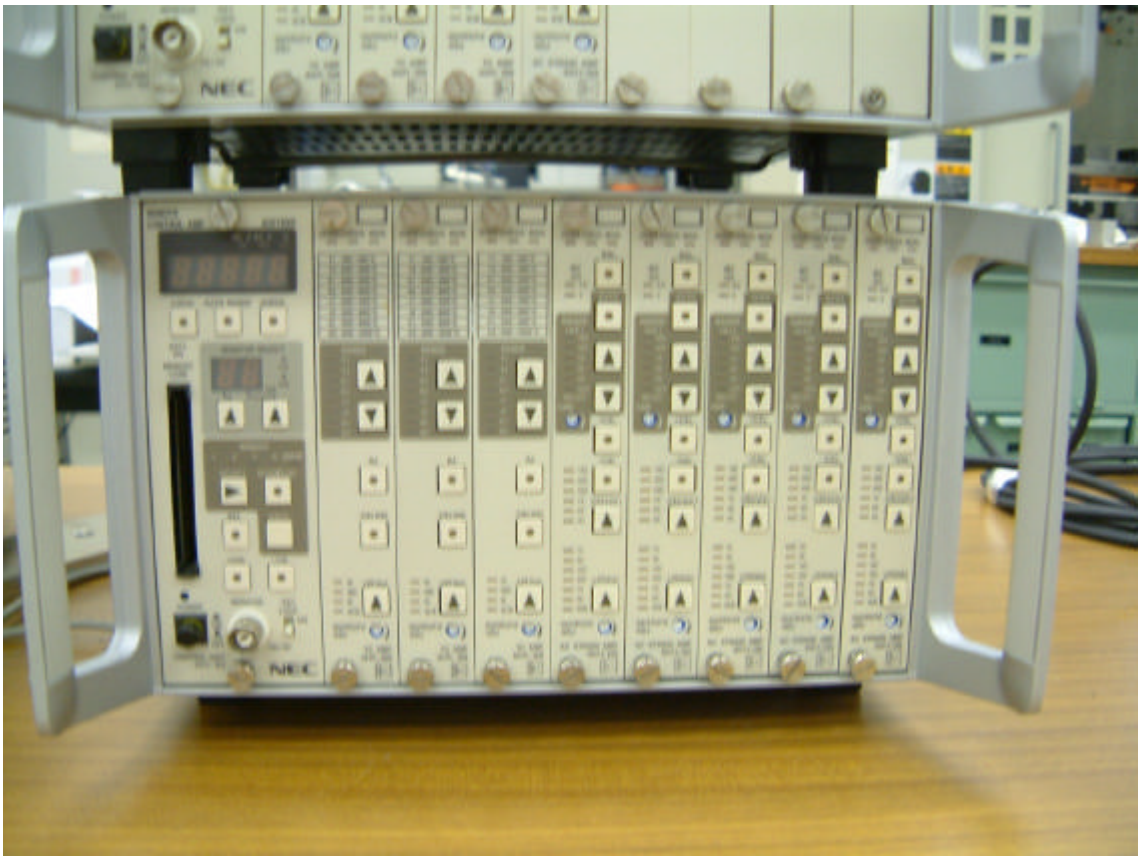


図 2.14　ストレインアンプ

適用ゲージ抵抗	120Ω～１ｋΩ
ブリッジ電源	2Ⅴ、0.5Vrms 内部切り替え 正弦波 25kHz
平衡調整方式	抵抗分自動バランス、バックアップ約１ヶ月、 容量分自動除去
平衡調整範囲	抵抗分±約 2%(±約 10000×10^{-6} ひずみ)
電圧感度	200×10^{-6} ひずみ入力にて 5V 以上(VAR 最大)
測定範囲	200、500、1k、2k、5k×10^{-6} ひずみ/FS、OFF (VAR 最大、BV=2V)
非直線性	±0.2%/FS 以内
周波数特性	DC～10kHz±10%

表 2.3　ＡＣストレインアンプユニットＡＨ11－104

仕様熱電対	Ｔ形(CC)、Ｅ形(CRC)、Ｊ形(IC)、Ｋ形(CA)
測定温度範囲	Ｔ形　T1:－150～150℃　T2:－200～300℃ Ｅ形　E1:－200400℃　E2:－200～800℃ Ｊ形　J1:－200～400℃　J2:－200～800℃ Ｋ形　K1:－200～600℃　K2:－200～1200℃
温度補償回路	誤差±2℃以内
リニアライザ回路	±0.5%/FS 以内(0℃以上)　±1.0%/FS 以内(0℃以上)
周波数特性	DC~10kHz +1、－3dB

表 2.4　熱電対アンプユニットＡＨ11－109

2.5　A／D変換ボード

ストレインアンプの出力はアナログ電圧値であるので、interface 社製 16 ビット A／D 変換ボードＰＣＩ－3155 により、コンピュータに取り込む。仕様を以下の表 2.5 に示す。

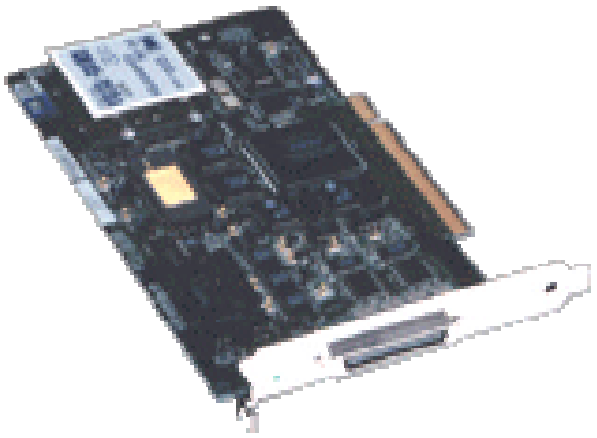


図 2.15　A／D変換ボード

入力チャンネル数	シングルエンド入力 16 チャンネル / 差動入力 8 チャンネル
入力制御方式	マルチプレクサ方式
入力アクセス方式	FIFO 方式
変換時間	10 μs (チャンネル固定時) 10 μs/チャンネル (チャンネル切替時)
入力分解能	16bit
入力レンジ	バイポーラ: ±1V,±2.5V,±5V,±10V
バス仕様	PCI ローカルバス(Rev. 2.1 以上), 32 bit, 33 MHz

表 2.5　A／D変換ボードＰＣＩ－3155 の仕様

2.6　熱アクチュエータ

本研究に用いるような 3 軸のマシニングセンタでは、構造体の変形による主軸位置の変位のうち並進方向の変位については座標系の原点をずらすことで 1μm刻みでの補正が可能であるが、主軸の傾きについてはＮＣ機能では補正できない。そこで、マシニングセンタのコラム下部に図 2.9 のように加熱板と冷却ファンをコラム前面、背面、側面、ATC 側面に設置する。

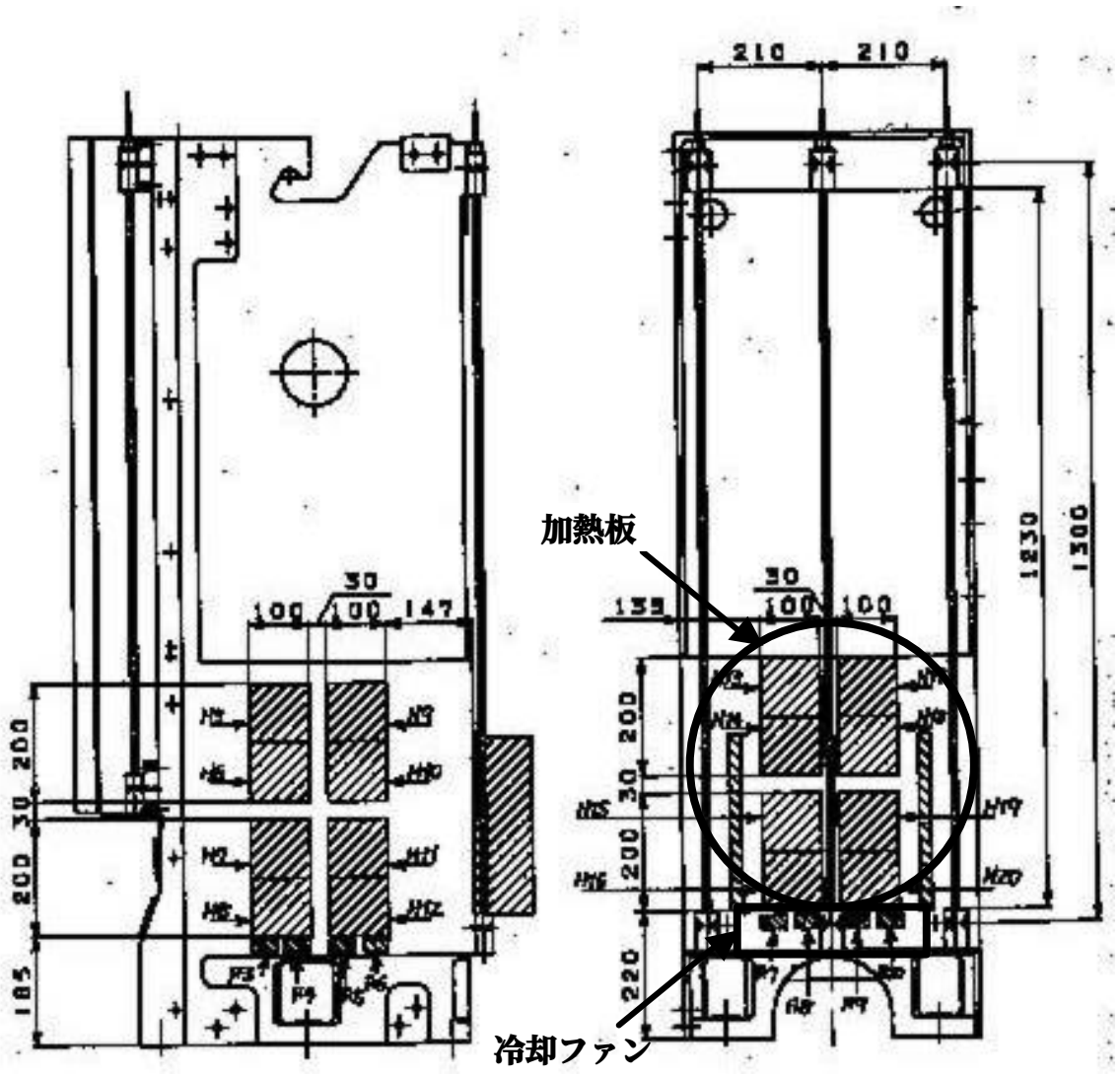


図 2.16　熱アクチュエータの設置図

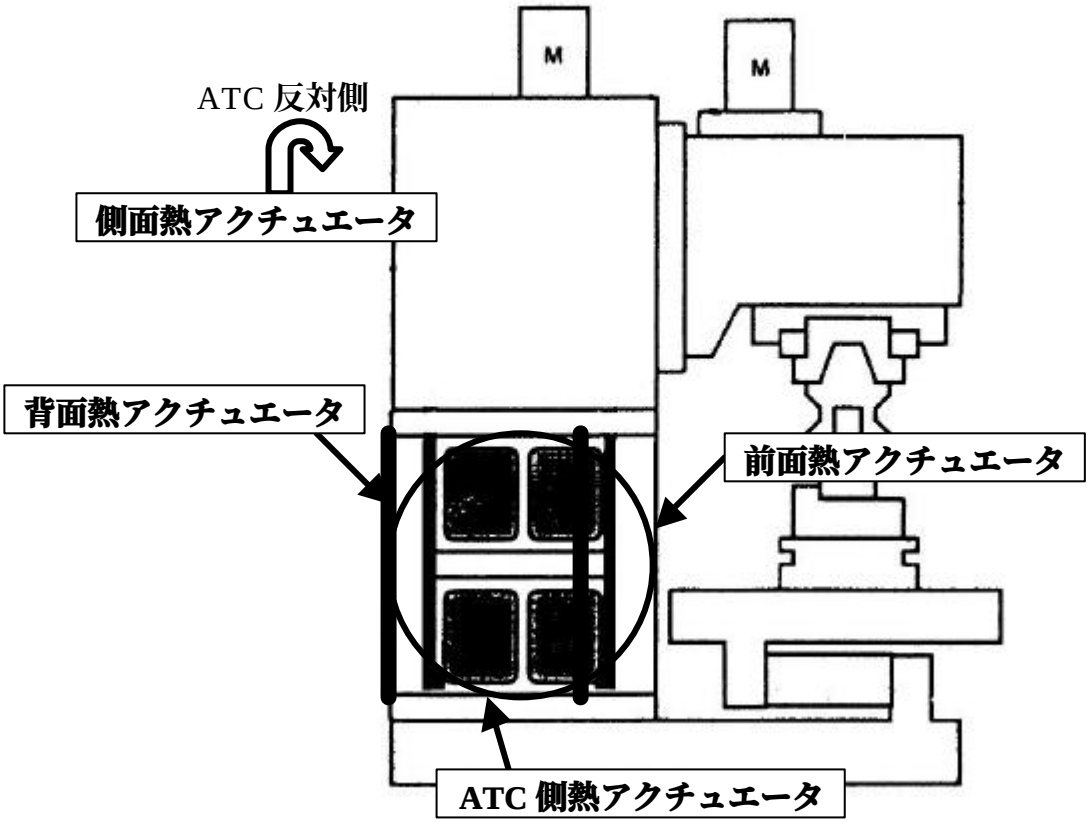


図 2.17　　熱アクチュエータ配置図

熱アクチュエータは加熱板と冷却ファンがそれぞれ 14 個で構成されている。

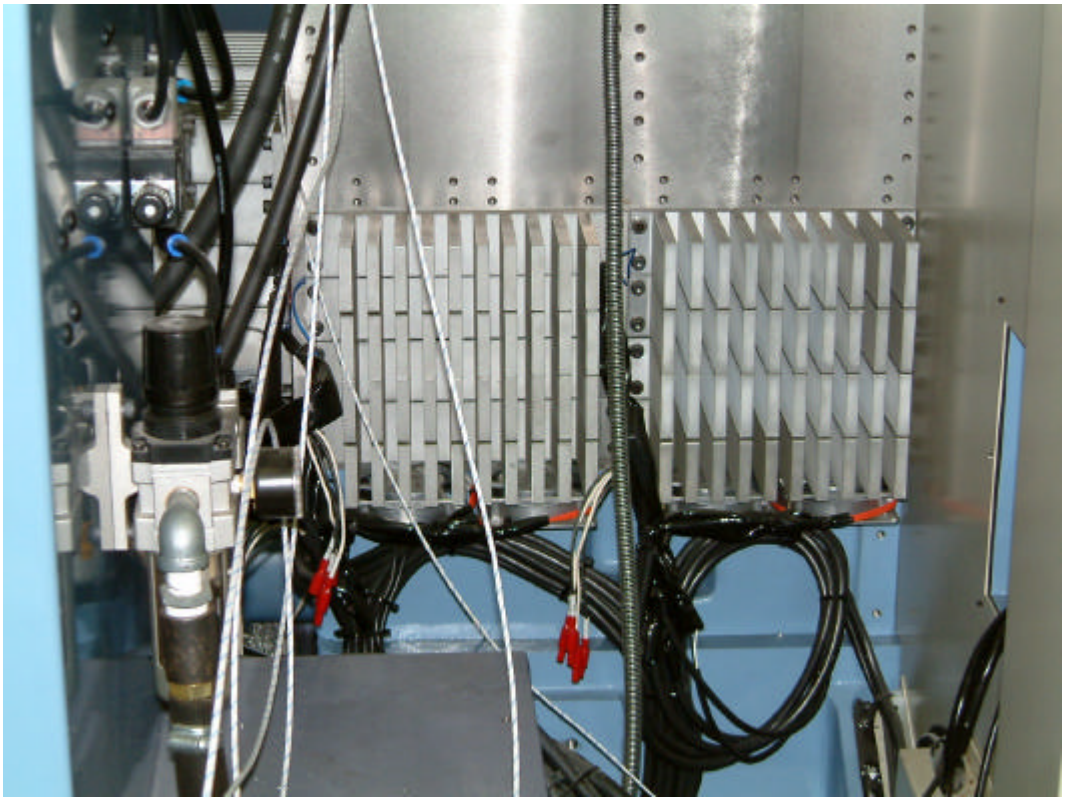


図 2.18　熱アクチュエータ

これらの熱アクチュエータを加熱、冷却することで図 2.11 に示すような種々の変形モードで構造体を変形させることが可能である。この変形を必要なモードで変化させることで、熱変形による主軸の傾きの変化を打ち消すことができる。

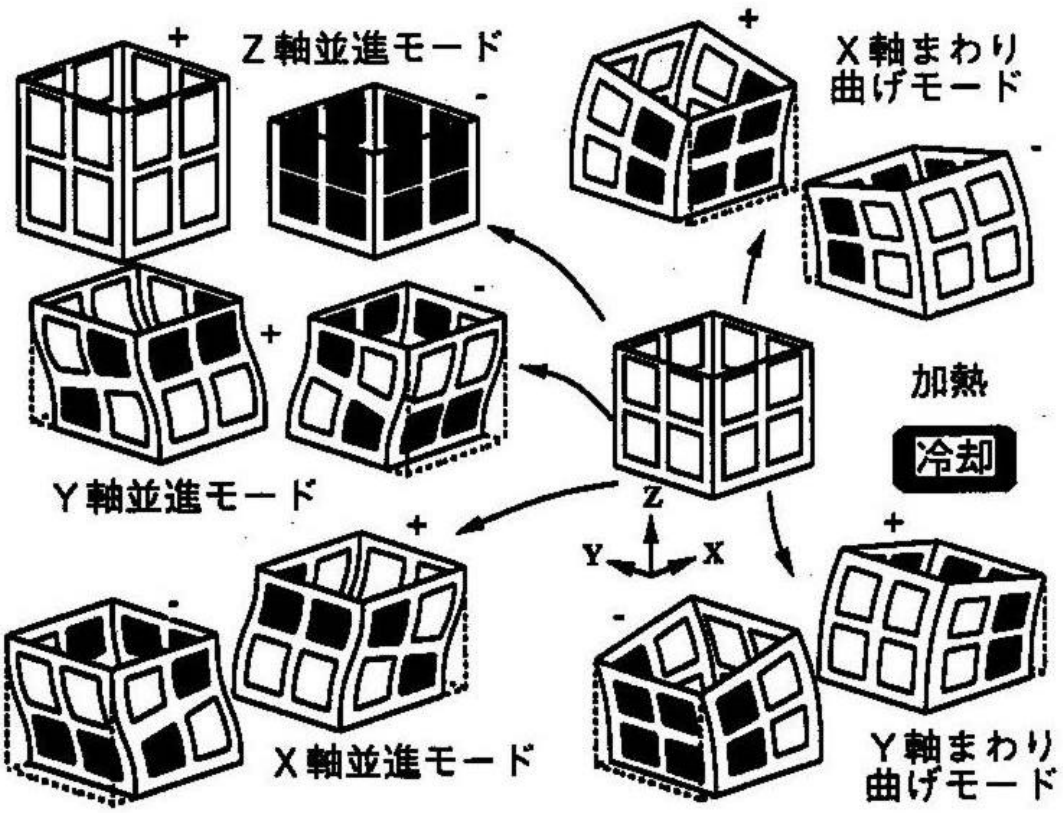


図 2.19　熱アクチュエータの変形モード

2.7　歪測定プログラム

今回歪を測定するために、VisualC++を使用してデータ測定プログラムを作成した。操作画面を図 2.14 に示す。測定条件を変えられるようにし、データをメモ帳に保存できるようにした。

A／D変換ボードから取得するデジタル値を歪に変えなければならない。

歪＝パネル表示校正値×デジタル値 / 分解能×1/2　　（2.1）

今回のゲージ法は 2 アクティブゲージなので、パネル表示校正値×1/2 で分解能が 65536 なので

歪＝1000×デジタル値 / 65536×1/2　　（2.2）

という式が成り立つ。

また今回は変形センサの性能テストの際に熱伝対を使用して温度変化もはかるのでデジタル値から温度を求める。今回は－150℃～150℃までを測定できるようにする。レンジは±5Vで、分解能が 65536 である。

温度＝30.0×デジタル値－32768 / 65536　　(2.3)

この計算式をもとに変形センサ 5 本分、5 チャンネルの測定を行うことのできるプログラムを作成。

2.8　遠隔制御プログラム

本研究で使用するマシニングセンタには外部パソコンと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）を搭載している。

このＨＳＳＢを通じて外部のパソコンから、以下のような制御を行うことができる。

- 主軸回転数の設定（Ｓコード）
- 主軸回転の制御及びクーラント、ワークライト等その他の装置の制御（Ｍコード）
- オートツールチェンジャの工具の選択（Ｔコード）
- 軸送り量の設定
- 原点シフト量の設定
- 熱アクチュエータの制御（Ｈコード）
- 自動運転のサイクルスタート、フィードホールド

遠隔制御の構成図を図 2.20 に遠隔操作の操作画面を図 2.21 に示す。

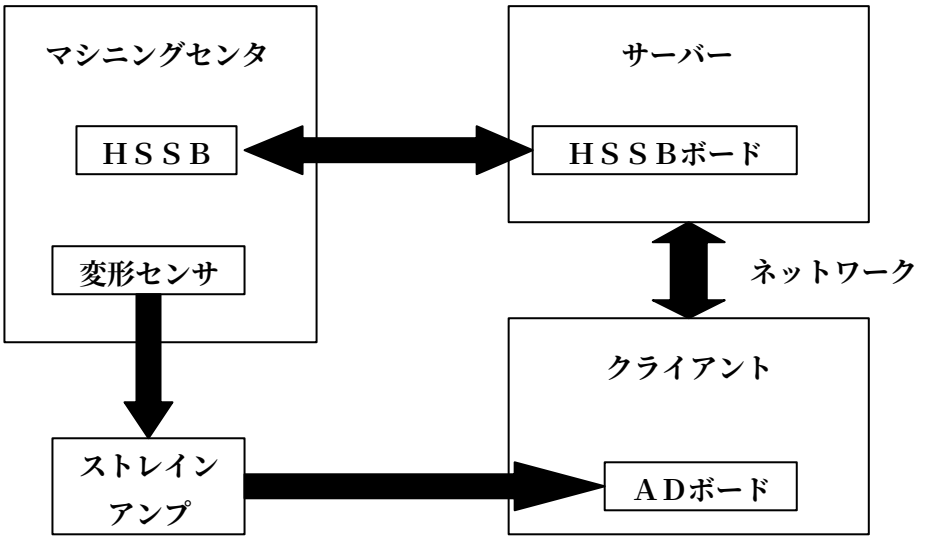


図 2.20　遠隔制御の構成図

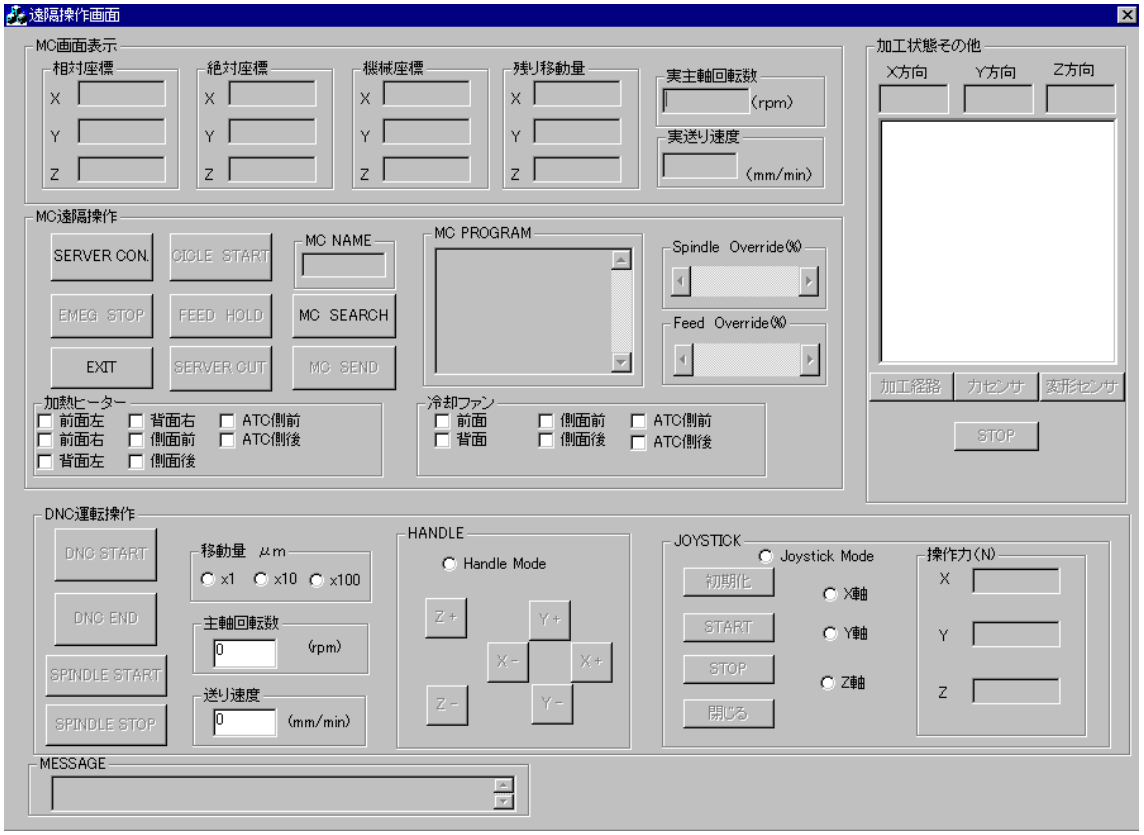

遠隔操作画面
MC画面表示
相対座標
絶対座標
機械座標
残り移動量
実主軸回転数
(rpm)
実送り速度
(mm/min)
MC遠隔操作
SERVER CON.
CICLE START
MC NAME
MC PROGRAM
Spindle Override(%)
EMEG STOP
FEED HOLD
MC SEARCH
Feed Override(%)
EXIT
SERVER CUT
MC SEND
加熱ヒーター
前面左
前面右
背面左
背面右
側面前
側面後
ATC側前
ATC側後
冷却ファン
前面
背面
側面前
側面後
ATC側前
ATC側後
加工状態その他
X方向
Y方向
Z方向
加工経路
力センサ
変形センサ
STOP
DNC運転操作
DNC START
DNC END
SPINDLE START
SPINDLE STOP
移動量 μm
x1
x10
x100
主軸回転数
(rpm)
送り速度
(mm/min)
HANDLE
Handle Mode
Z+
Y+
X-
X+
Z-
Y-
JOYSTICK
Joystick Mode
初期化
START
STOP
閉じる
X軸
Y軸
Z軸
操作力(N)
X
Y
Z
MESSAGE

図 2.21　遠隔操作画面

2.9　変形センサの設置

今回作成した変形センサ 5 本はすべて主軸ヘッド周辺に設置する。図 2.12 に示したセンサ A,B,C,D,E はそれぞれ図 2.22 に設置場所を示す。

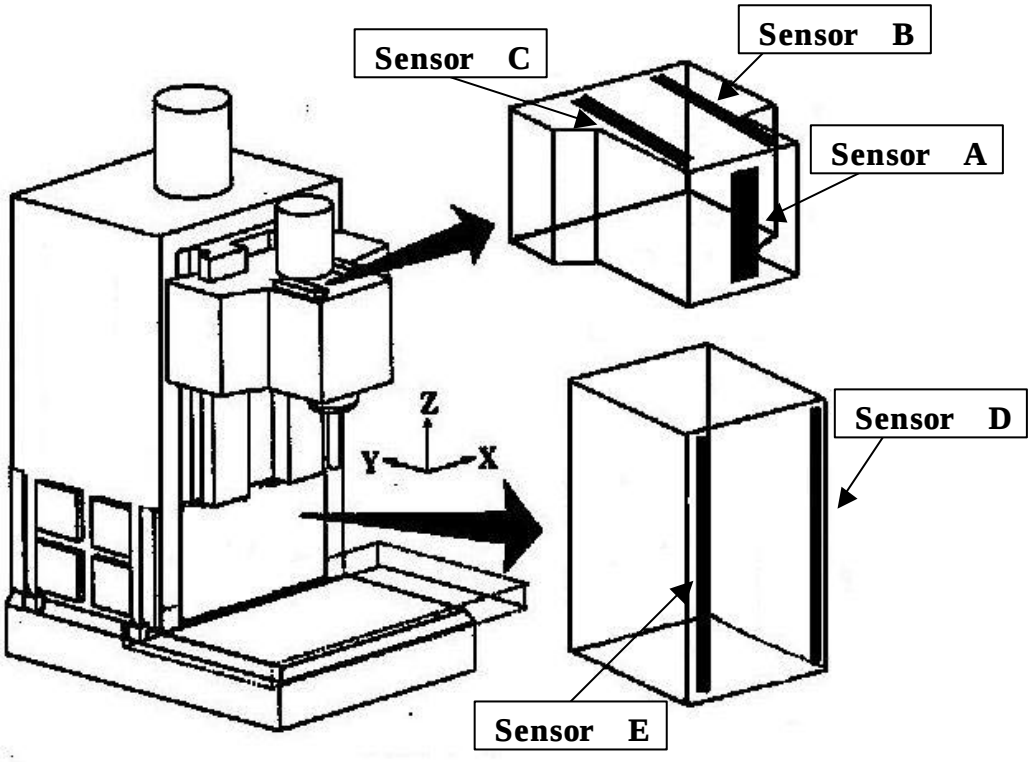


図 2.22　変形センサの設置図

第 3 章

姿勢制御の手法

3.1　基本変形モード

マシニングセンタの主軸位置は 3 軸方向の並進成分と、X 軸および Y 軸の回転成分に分割され計５次元の成分で記述できるものと考えられる。

構造体の変形による主軸変位を補正する手段としては、軸送りで対処する方法もあるが、これにより補正が可能なのは並進成分のみであり、回転成分は補正不可能である。並進成分に関しては、機械使用者が経験的に変位分を補正する、ないしは肉眼で確認するなどして補正できるが、回転成分に関しては、3 軸の送り機能しか持たない一般の工作機械ではどうにもならず、ドリルによる穴あけ加工やタッピング時の垂直度、フライス加工での加工面の平行度に悪影響を与える。

今回はこのうちのX軸およびY軸の回転成分に注目して研究を行っているが、この回転成分を熱アクチュエータで全て打ち消すことを考えた場合、かなり特殊な工夫をしなければならない。

そこで前章でも示したような熱アクチュエータ配置にするのだが、この配置で回転成分を打ち消すことが可能であると図 2.17 に示す。

3.2　マシニングセンタ姿勢制御法

今回変形センサからの取得する歪のデータを利用してマシニングセンタの姿勢制御をすることにする。2 ゲージアクティブ法では曲げのひずみを検出することができるのでマシニングセンタの主軸回転中の熱変形を取得する。そしてコラムに取り付けられた熱アクチュエータを稼動させそのときの熱変形を取得する。これらのデータによりどの部分がどれだけ変形するかがわかる。このデータをもとに加工中の熱変形を遠隔操作で熱アクチュエータを動かすことにより、できるだけ熱変形を 0 に持っていくような変形モードの組み合わせの遠隔プログラムを作成する。姿勢制御の流れを図 3.1 に示す。

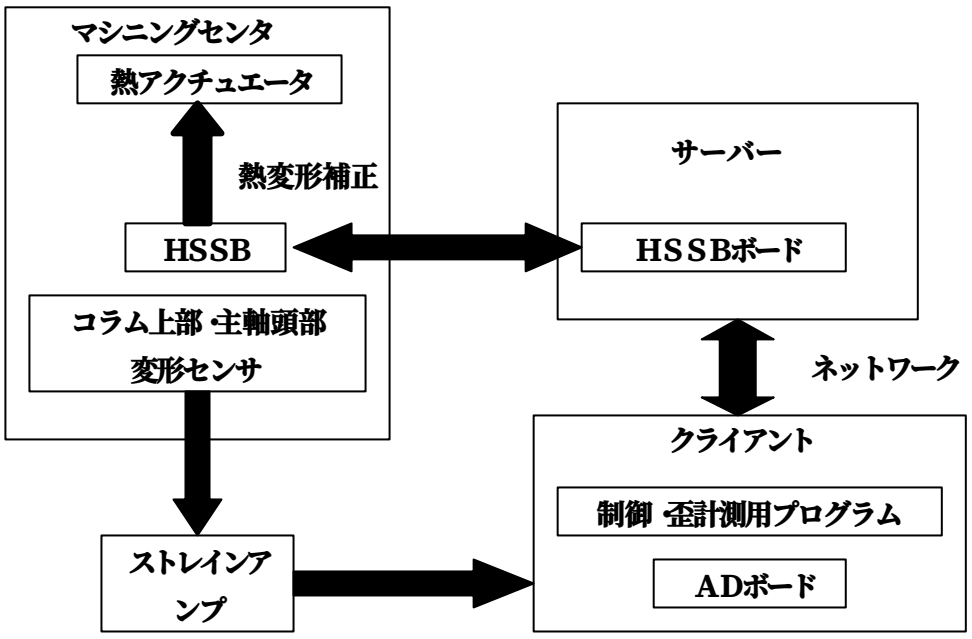


図 3.1　マシニングセンタの熱変形能動補償システム

第 4 章

変形センサの性能評価

4.1　ハードウェアの性能評価

本章では、高精密マシニングセンタに設置した熱アクチュエータ、変形センサ等の姿勢制御のためのハードウェアについて、その性能を評価し、高精密マシニングセンタの能力を確認する。

4.2　変形センサの性能評価

本研究では変形センサを自分たちで設計・開発、製作したので性能評価実験を何回も繰り返して正確なデータの測定できる変形センサのみをマシニングセンタに設置しなければならない。今回性能評価実験として、SS400 の 10×1300×100 にシリコンラバーヒーターを取り付け、変形センサを設置し、30 分加熱 30 分冷却し温度変化と歪を測定する。

幅×長さ	100×100
容量	60W
連続使用温度	200℃
最高使用温度	250℃
リード線の長さ	300mm

表 4.1　シリコンラバーヒーターの仕様

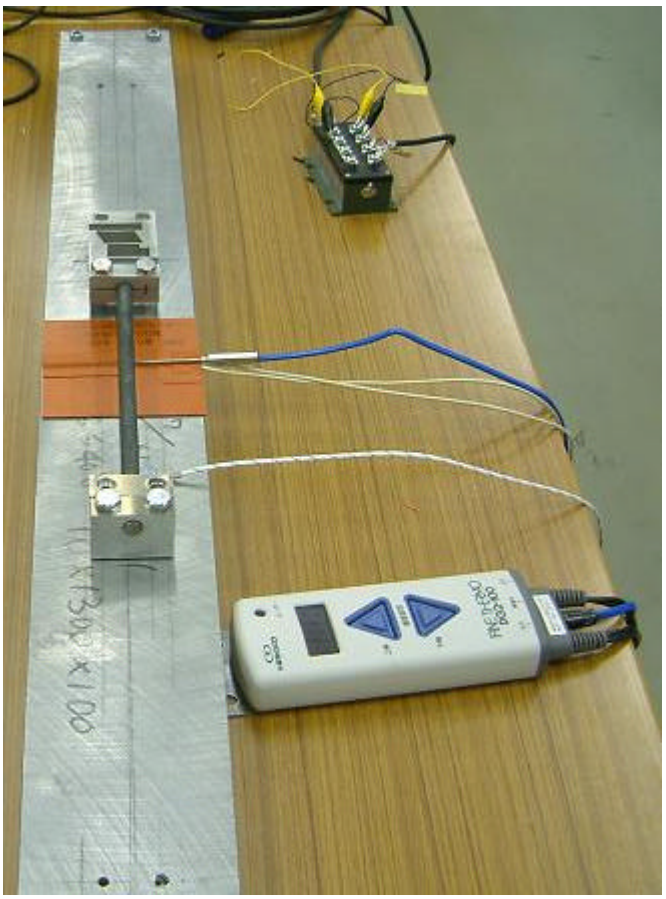


図 4.1　変形センサ性能評価実験

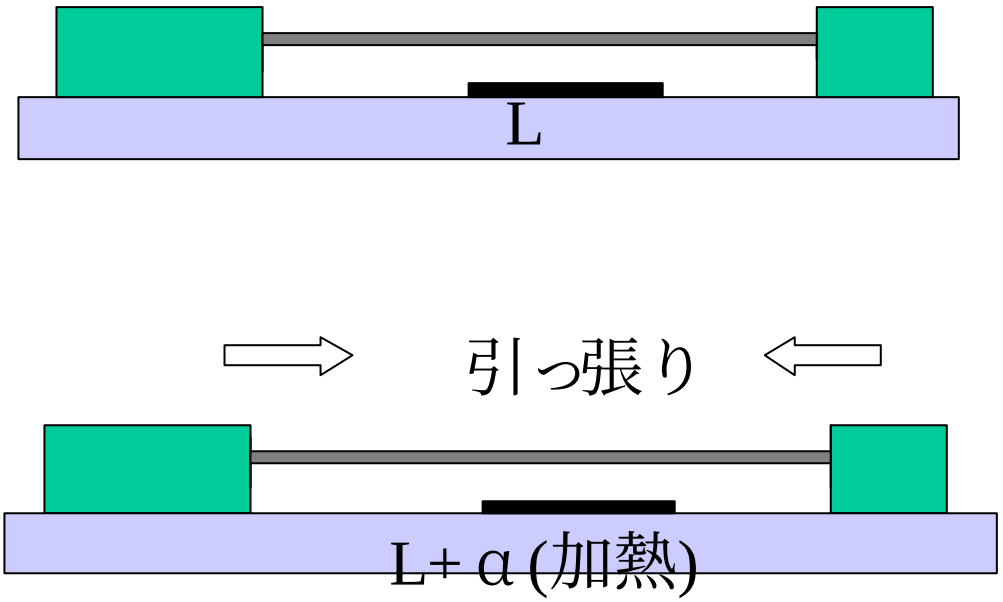


図 4.2　変形センサ性能評価実験外略図

このように加熱板で鉄板の中央のみを加熱するのでその部分が膨張し変形センサに引っ張りの力がかかり、変形センサの性能がよければマイナスの値が出るはずである。

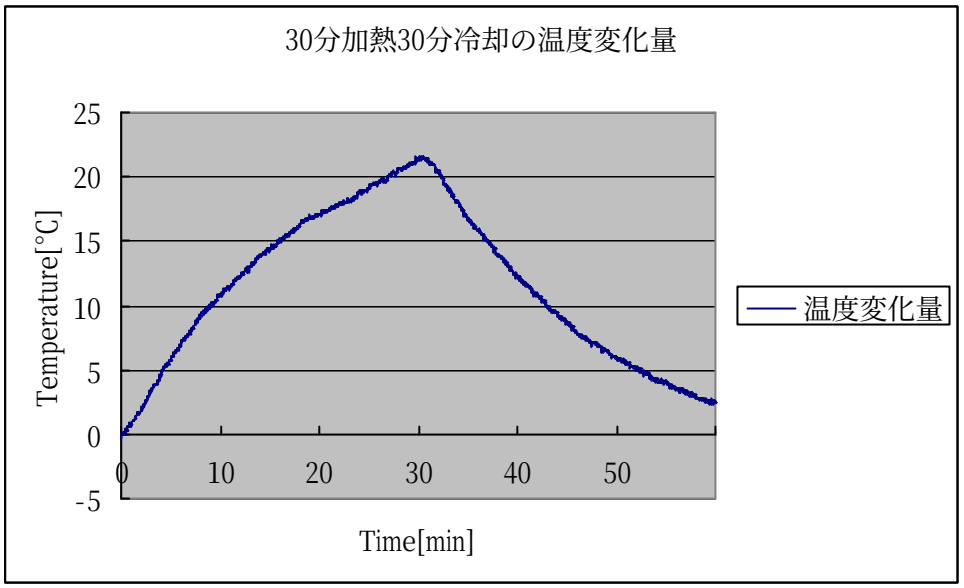


図 4.3　性能評価実験の温度変化

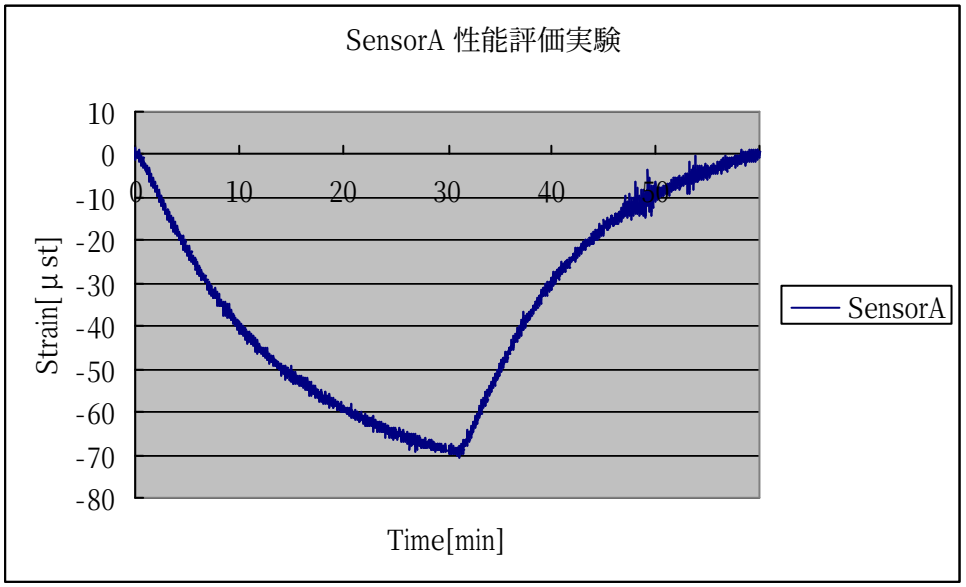


図 4.4　センサ A の性能評価実験

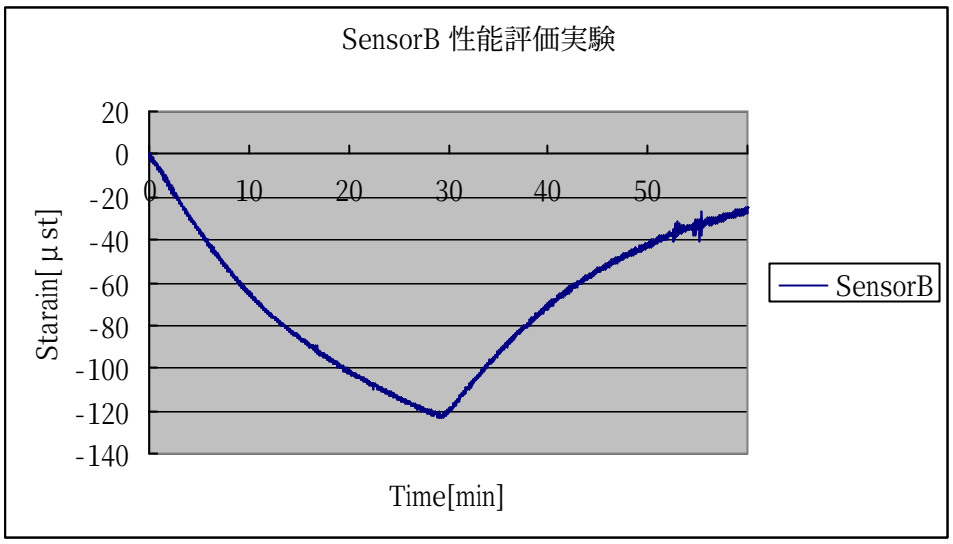


図 4.5　センサ B の性能評価実験

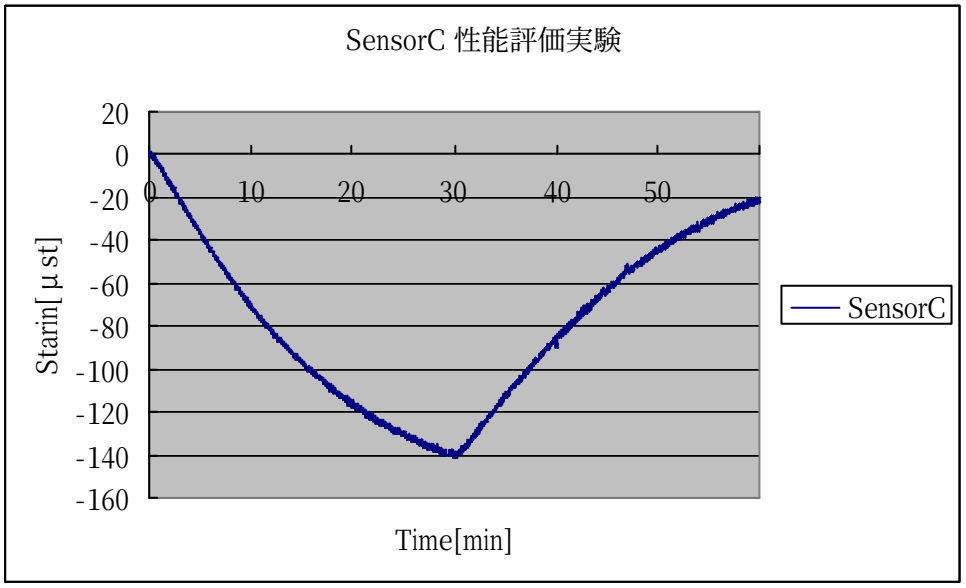


図 4.6　センサ C の性能評価実験

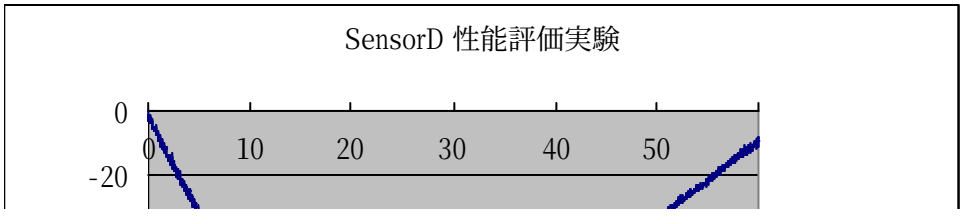


図 4.7　センサ D の性能評価実験

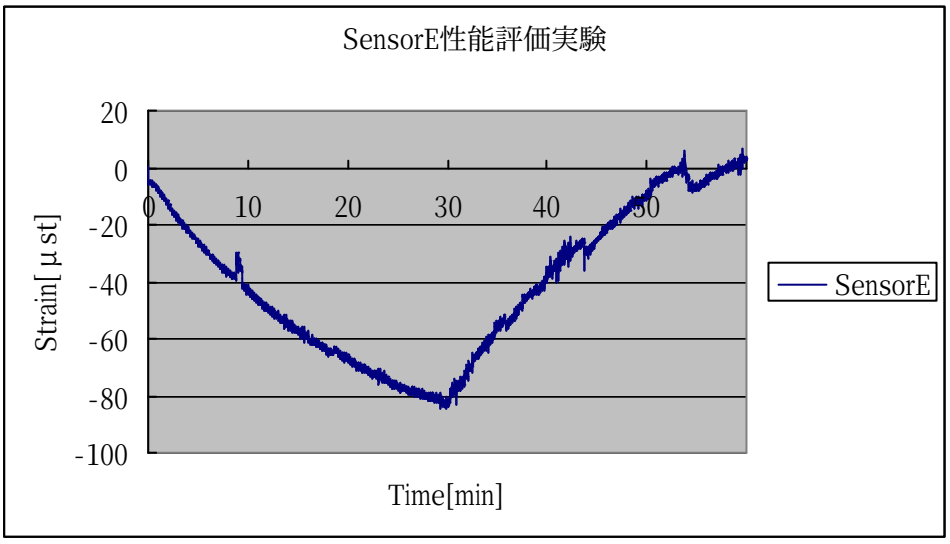


図 4.8　センサ E の性能評価実験

以上の結果より変形センサ 5 本は実験に必要な性能を十分持ち合わせているといえる。

4.3　熱アクチュエータの性能評価

マシニングセンタのコラム部下部に設置されてある熱アクチュエータについて、30 分加熱し、30 分冷却して熱アクチュエータの各部の性能評価をする。今回熱アクチュエータ部分には変形センサを設置してなしが、熱アクチュエータの加熱冷却で主軸部がどれだけ変形するかも同時に見てみる。

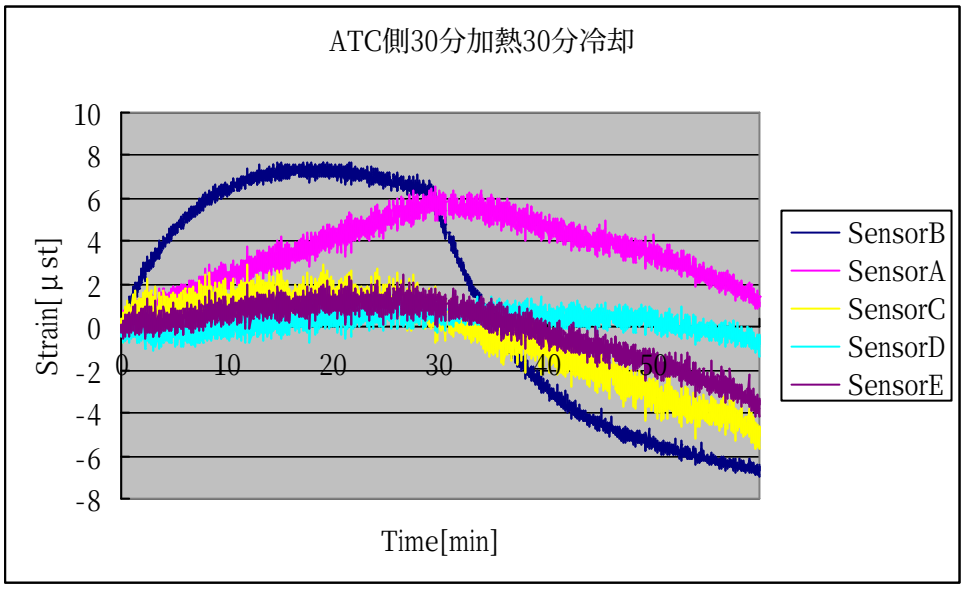


図 4.9　ATC 側熱アクチュエータ性能評価実験

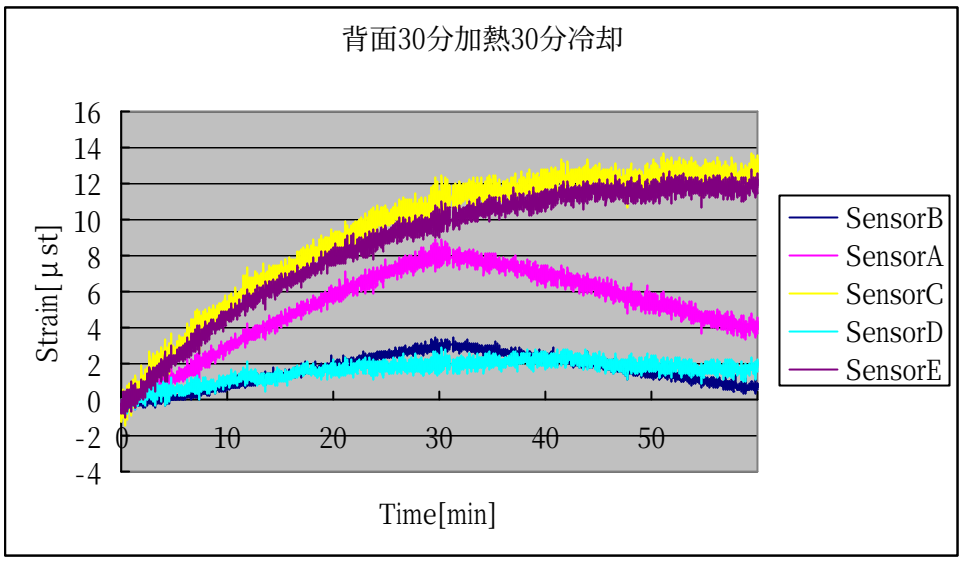


図 4.10　　コラム背面熱アクチュエータ性能評価実験

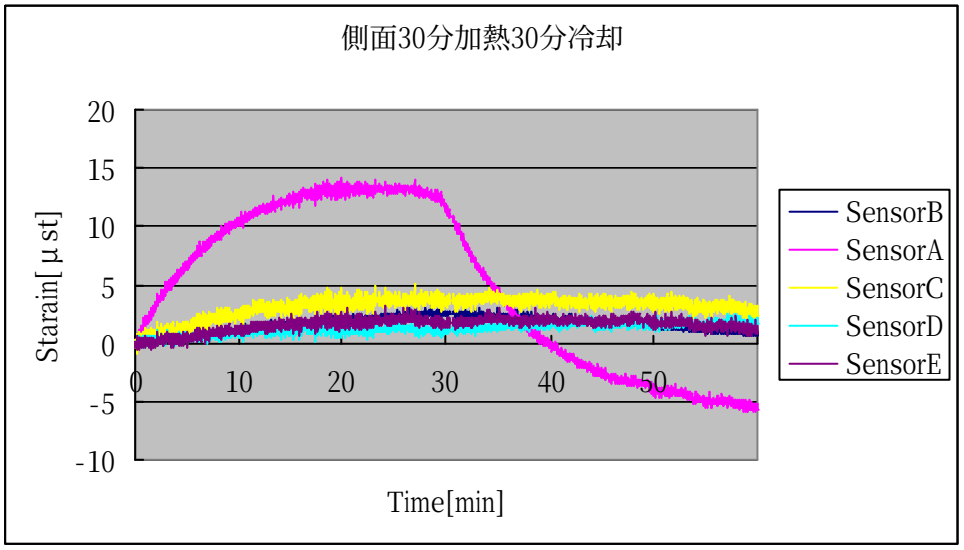


図 4.11　コラム側面熱アクチュエータ性能評価実験

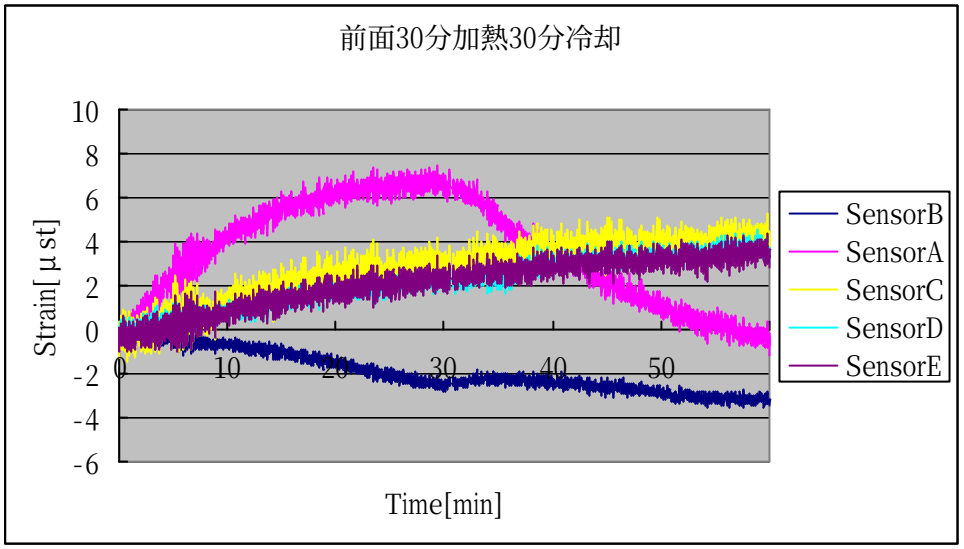


図 4.12　コラム前面熱アクチュエータ性能評価実験

4.4　結果及び考察

本研究では変形センサを自分たちで設計開発、製作したので歪がうまく測定できるか不安だったがうまく測定することができた。しかし、若干ノイズが気になるが今回の研究には支障がないレベルである。

しかし、変形センサの性能評価実験の時よりも出力値にばらつきが見られるのは変形センサの取り付け方に問題があると考えられる。変形センサを取り付けるときには、センサに過大な負荷がかかることのないように取り付けたのだが、結果を見る限りでは、最も出力が出るセンサと出ないセンサとの間に倍程度の差があり、均等化できてないようである。もちろん変形センサ自体にも個体差がある。このように取り付け方法、さらには取り扱い、メンテナンスが難しいセンサがシステムに組み込まれている場合には、このような機械が世の中に広がる可能性は非常に低いといわざる得ない。

熱アクチュエータ部分には変形センサを取り付けてないが、主軸部分の変形センサでもきちんと熱アクチュエータの 30 分加熱 30 分冷却を取得することができ、熱アクチュエータの加熱板と冷却ファンは十分な性能を持っているといえる。

第 5 章

熱アクチュエータによる姿勢制御

5.1　主軸回転時の熱変形測定

熱アクチュエータによる姿勢制御をするために主軸回転時の歪測定を行い、測定データから変形モードを導き出す。今回は加工時によく使用する回転数 1500rpm と 3000rpm、またマシニングセンタの最高回転数である 6000rpm で歪測定を行った。

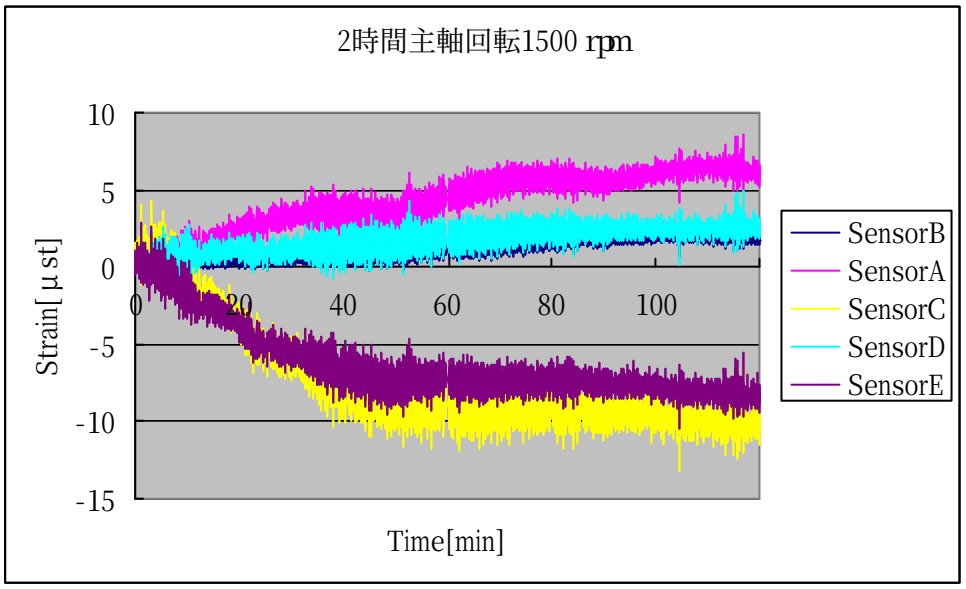


図 5.1　2 時間主軸回転 1500rpm での歪測定

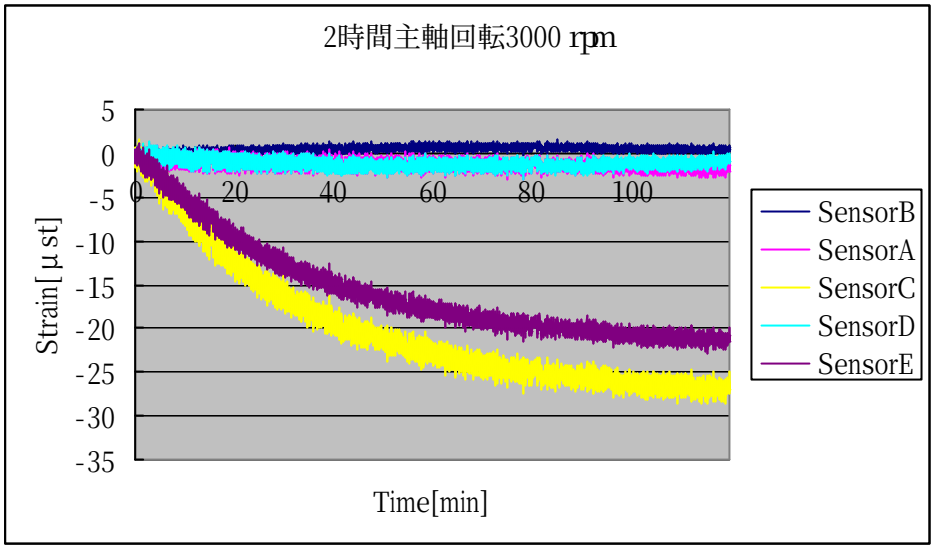


図 5.2　2 時間主軸回転 3000rpm での歪測定

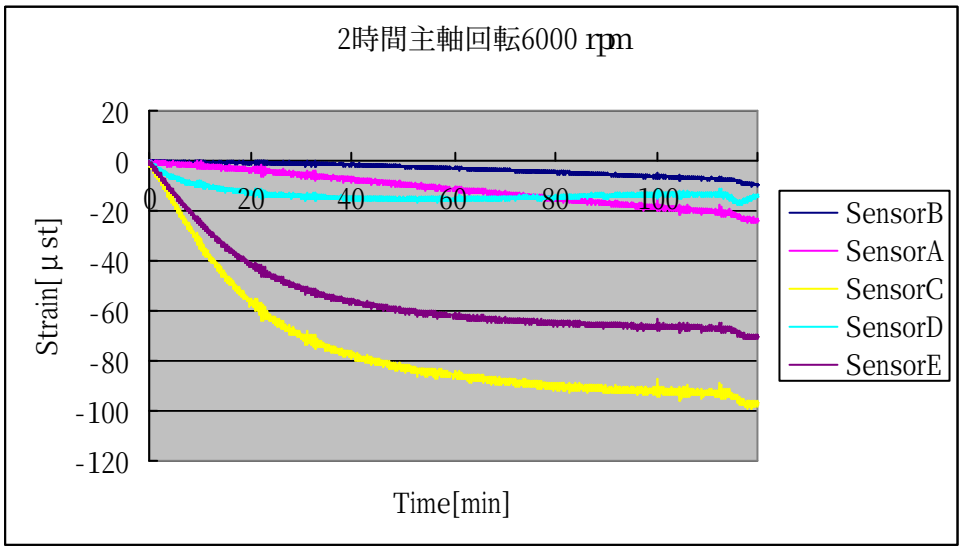


図 5.3　2 時間主軸回転 6000rpm での歪測定

主軸回転による熱変形ではセンサ A、B、D ではそれほど大きな熱変形はなかったが、センサ C、E が大きく変形した。この変形センサは圧縮が＋引張りが－となっており、図 2.19 からわかるようにマシニングセンタの左側に大きな熱変形が生じている。これはマシニングセンタ右側に電源などがあり主軸回転時に熱を持ちやすいと考えられる。また機械自体の冷却装置によってあまり加熱してない左側が冷却されることによって圧縮され、そのために左側のセンサ C、E が大きく変形したのではないだろうか。最大回転数の 6000 回転だとセンサ A、B、D も若干変形を生じた。しかし実際の加工で 6000 回転を使用することがなく、これからの制御には実際に使用する最大回転数 3000 回転を基準に制御していく。

5.2　熱アクチュエータでの熱変形測定

今回の研究では熱アクチュエータを使用して熱変形を制御するので、熱アクチュエータでどれぐらいの変形が起こるかを把握しなくてはならない。そこで、4 章でおこなった熱アクチュエータの性能評価実験と同じように、前面、背面、側面、ATC 側の各熱アクチュエータを 30 分加熱 30 分冷却し、それを数回繰り返しその平均をグラフ化し、熱アクチュエータが各センサにどれだけ変形を与えているかを実験する。

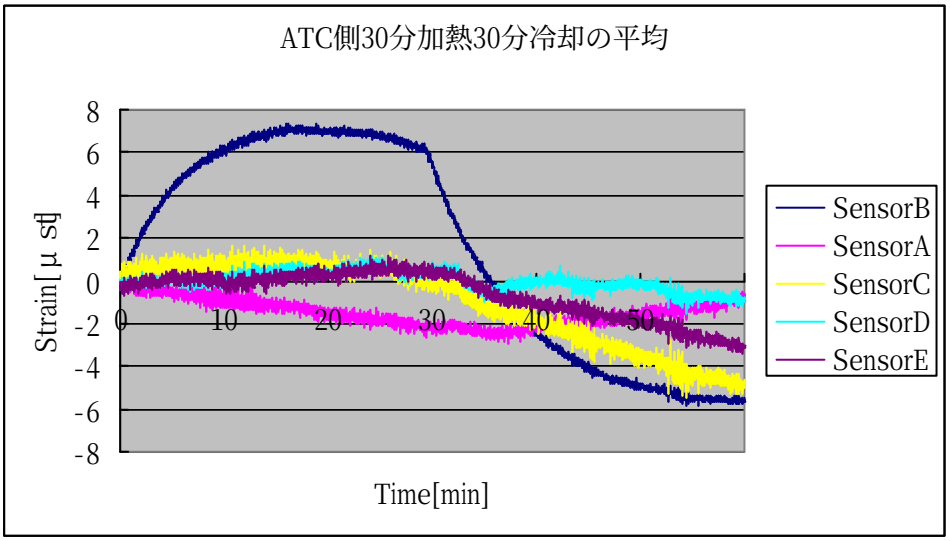


図 5.4　ATC 側 30 分加熱 30 分冷却の平均

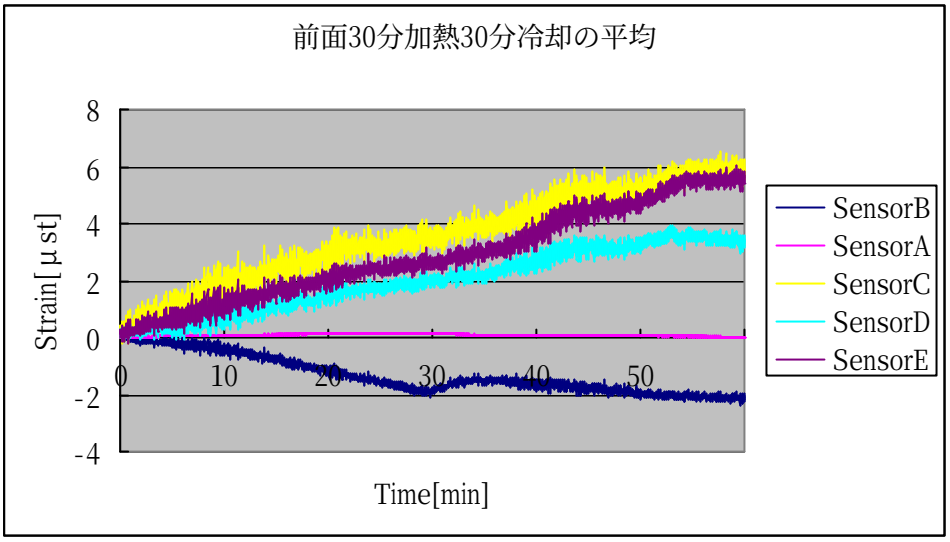


図 5.5　前面 30 分加熱 30 分冷却の平均

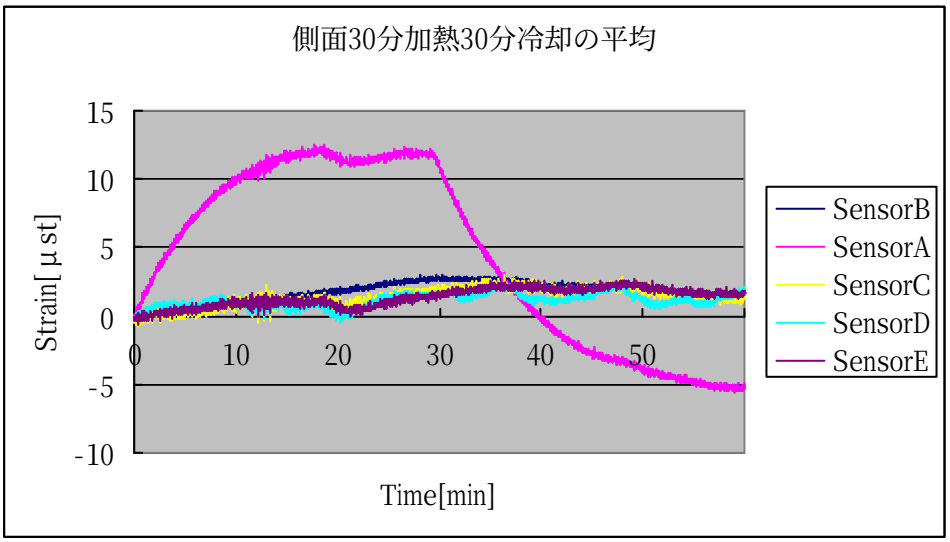


図 5.6　側面 30 分加熱 30 分冷却の平均

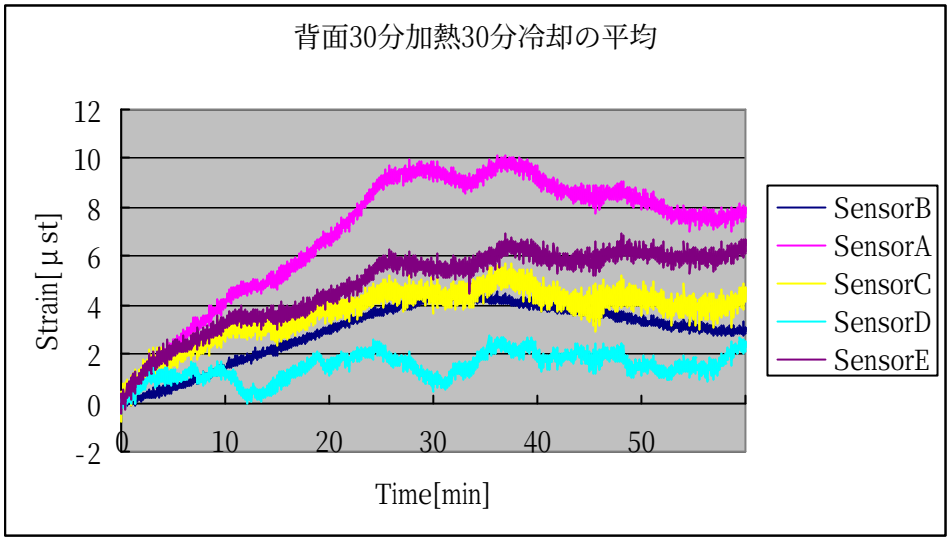


図 5.6　背面 30 分加熱 30 分冷却の平均の近似曲線

以上の結果から熱アクチュエータと歪の関係は表 5.1 のようになる

前面	加熱	冷却	背面	加熱	冷却
Sensor A	?	?	SensorA	?	?
B	?	?	B	?	?
C	?	?	C	?	?
D	?	?	D	―	―
E	?	?	E	?	―
ATC	加熱	冷却	側面	加熱	冷却
Sensor A	―	?	SensorA	?	?
B	?	?	B	―	―
C	―	?	C	―	―
D	―	?	D	―	―
E	―	?	E	―	―

表 5.1　熱アクチュエータと歪の関係

これの結果から主軸 3000 回転の時の熱変形を熱アクチュエータを使用して軽減させるように実験をしていき、最良の熱アクチュエータの組み合わせを見つけそれを利用して制御していく。

5.3　姿勢制御プログラムによる熱変形補償実験

実際に主軸を回転させた時に熱アクチュエータを稼動させ、マシニングセンタの熱変形を軽減させる実験を行う。今回は実際に加工時に使用する最大回転数 3000rpm で遠隔制御プログラムの元で実験を行う。この遠隔制御プログラムには常に歪±0.02 以下に抑えるように熱アクチュエータを遠隔で操作するようにプログラムを組んである。センサ C、E に影響の大きい前面と背面の熱アクチュエータを使用し、実験を行う。

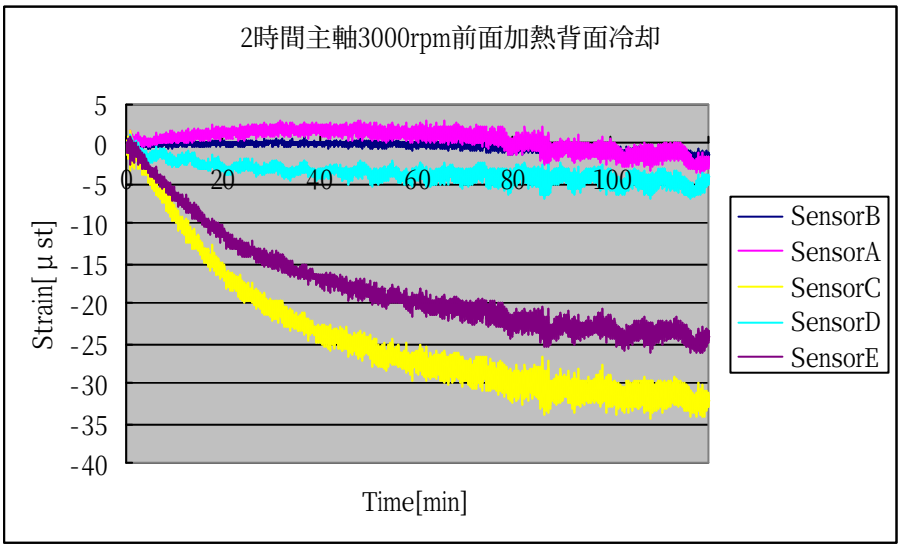


図 5.7　前面加熱背面冷却

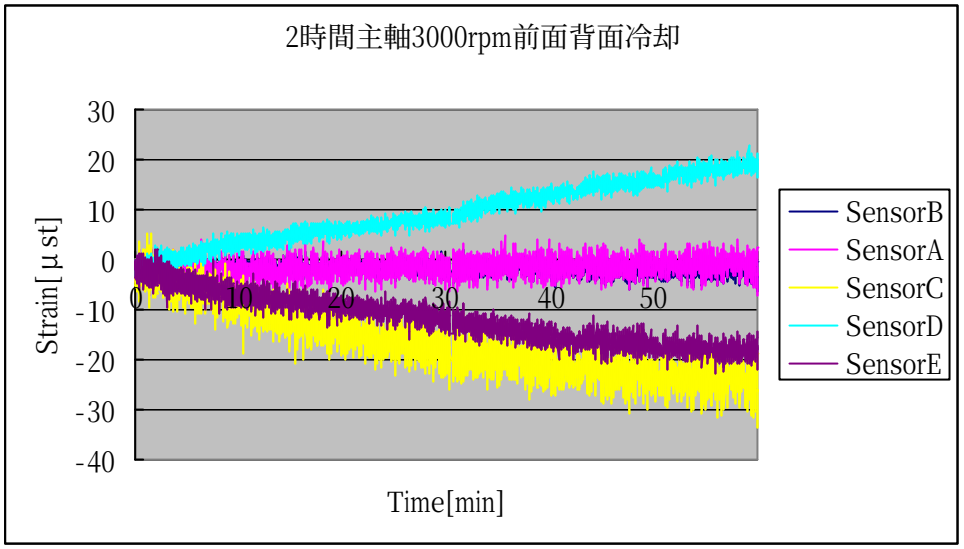


図 5.8　前面背面冷却

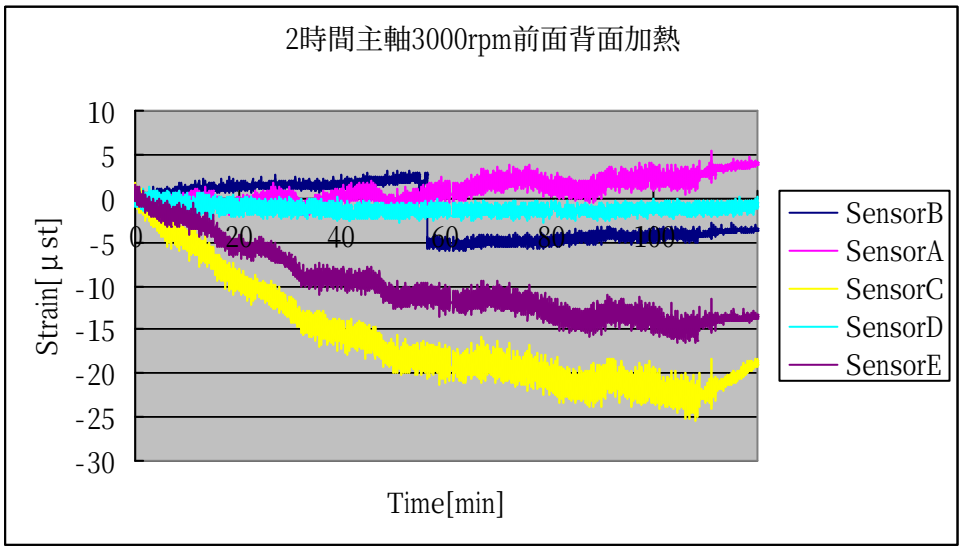


図 5.9　前面背面加熱

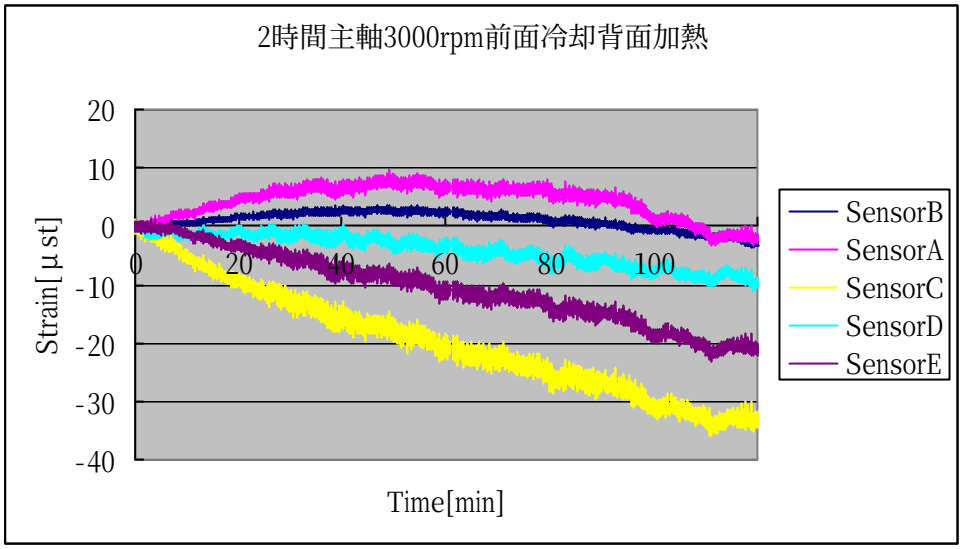


図 5.10　前面冷却背面加熱

今回の実験主軸 2 時間 3000rpm で熱変形を減少できたのは熱アクチュエータ前面背面加熱、前面背面冷却のみであった。しかし、減少したとはいえその減少幅は 10μst だけだった。前面背面の熱アクチュエータ以外の組み合わせ行っても若干の軽減もしくは変化なしであった。主軸回転時の歪はセンサ C、E すなわち熱アクチュエータ前面部が後ろに仰け反るような形となり引っ張られ、マイナスの歪が出たので、その前面を冷却し反対の背面を加熱することによって歪が軽減されると考えたが制御に至るまでの結果が出なかった。

5.4　結果、及び考察

熱アクチュエータによる姿勢制御は、センサ A、B、C、D、E において制御することができなかった。主軸回転時の熱変形に伴う歪の値が思っていたより大きく、今回使用した熱アクチュエータの数では制御することができなかった。しかし若干であるが軽減させることはできたのは事実である。

主軸頭部に設置した変形センサが主軸回転時に大きく変形することはだいたい想像はできたことだが、コラム下部に設置した熱アクチュエータによる影響があまり主軸部分に影響がないこと分かり、今回の熱アクチュエータによる熱変形の制御は失敗に終わった。

加工前に熱アクチュエータを稼動しておいてから主軸回転すればもう少し歪は軽減されると思うが、それでは加工時間の短縮ということにならないのでも少し熱アクチュエータの数と設置場所を考える必要がある。

しかし､変形を 0 にすることで主軸の傾きが０になるかどうかはわからなく、疑問が多少残るがこれは今後の課題として、後輩たちに任せることにする。

第 6 章

総括と展望

6.1　姿勢制御のためにハードウェアとして有している性能

・変形センサは、測定対象の熱変形を正確にとらえることができていることがわかった。
・高精密マシニングセンタは熱変形が正確に把握できた時、ハードウェアとしては熱変形を制御することができる可能性があるということが確かめられた。

以上のように、高精密マシニングセンタは加工の知能化の概念を実現し、ハードウェアとしては非常に高い性能を持っていることが明らかとなった。

6.2　展望

今回は熱アクチュエータがマシニングセンタにどのような熱変形をもたらすかということがわからず、最初からデータを測定しなければならなかった。また実際に遠隔で制御する際も、熱変形が起こってから熱アクチュエータを稼動させるというような仕組みであったから精度の要求されるモノの加工には不向きな制御方法である。また今回は主軸回転のみの研究で実際の加工中の歪測定は行っていない。

今回のデータをもとに熱変形を予測するシステムを作り、熱変形を 0 に持っていくような遠隔制御システムが必要である。

具体的には、有限要素法による構造シミュレーションにより、幾何学的な関係を明らかにすることや、それに、熱流束や熱変形を加味したモデルなどが考えられる。これは今後、ぜひ検討すべきである。

結果として制御するまで至らなかった今回の研究は成功とはいえないが、新たな問題点がいくつか見えてきた。まず変形センサの数である。今回の研究では 5 本という数だったが、これではマシニングセンタ全体の変形を把握することは難しい。また熱アクチュエータもコラム下部のみにしか搭載していない。

今後この研究を引き継いでもらう後輩にはぜひこういった問題点を解決し、熱変形を制御することのできる高精密マシニングセンタを開発してもらいたい。

第 7 章

結論

7.1　結論

・変形センサの検出部の板厚は 1 *mm* で十分な歪を検出できることができた。

・遠隔操作プログラムに歪測定プログラムを組み込むことによって遠隔姿勢制御プログラムを作ることができた。

・主軸回転時のマシニングセンタの熱変形による歪を測定することにより、加工中にどのような曲げが生じているかが明らかになった。

・熱変形能動補償型高精密マシニングセンタは、主軸回転数 3000rpm までの熱変形による歪を若干であるが軽減することができた。しかし制御するところまでには至らなかった。

・更なる精度向上には主軸の変位測定とその変位予測機能と、熱変形予測機能を検討する必要がある。

謝辞

本研究を進めるにあたり、多くの方々にご指導、ご協力をいただきました。本論文を締めくくるにあたり、高精密マシニングセンタの開発、研究に携わってこられた多くの方々に感謝の意を表します。

指導教官である長尾教授には日頃から親身なご指導をいただきました。おかげで大学院の 2 年間を大変有意義なものにすることができました。

同じ研究室の木崎君、上條君にも遠隔プログラム作成時や、変形センサ製作時にいろいろなアドバイスをしてもらい大変助かりました。

小林研究室の橋本君、小山君には変形センサ作成時のワイヤーカットでの加工の時には大変お世話になりました。

最後に今回の研究を一緒に行った学部 4 年の浅田君、高谷君には最初から最後まで実験を手伝ってもらい大変感謝しています。

本当にありがとうございました。

参考文献

[1]田口将、“ 変形情報に基づく姿勢制御によるマシニングセンタの高精密化の究 ”、1991 年度東京大学大学院工学系研究科産業機械工学専攻修士論文

[2]奥村努、“ 変形情報に基づく姿勢制御によるマシニングセンタの高精密化の究 ”、1994 年度東京大学大学院工学系研究科産業機械工学専攻修士論文

[3]花山良平、“ 熱変形能動補償型高精密マシニングセンタによる加工精度向上の研究 ”、1998 年東京大学工学部産業機械工学科卒業論文

[4]長尾高明、畑村洋太郎、衛光石衛、中尾政之、“ 知能化生産システム ”、朝倉書店、2000 年

[5]加藤賢一、“ マシニングセンタの知能化による高精密化の試み ”、1991 年度東京大学大学院工学系研究科産業機械工学専攻修士論文

[6]市川純理、“ ニューラルネットワークを用いたマシニングセンタ高精密化の試み ”、1992 年東京大学工学部産業機械工学科卒業論文

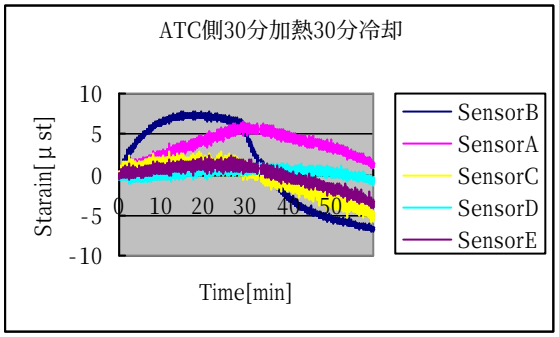
[7]堤和久、“ 熱変形能動補償型高精密マシニングセンタによる加工精度向上の研究 ”、1996 年東京大学工学部産業機械工学科卒業論文

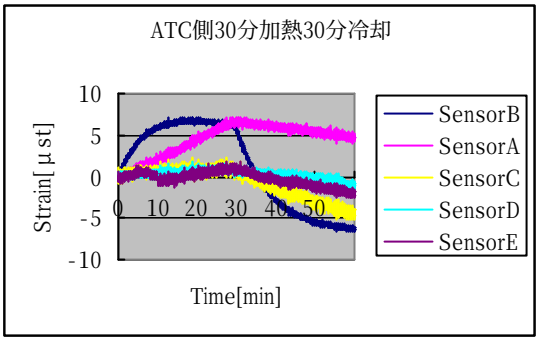
[8]佐藤善治、“ マシニングセンタ加工 ”、日刊工業新聞社、1996 年

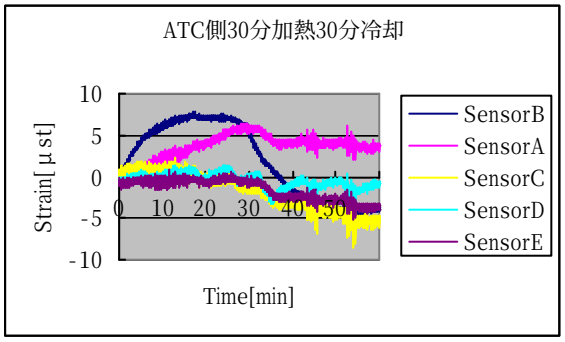
付録

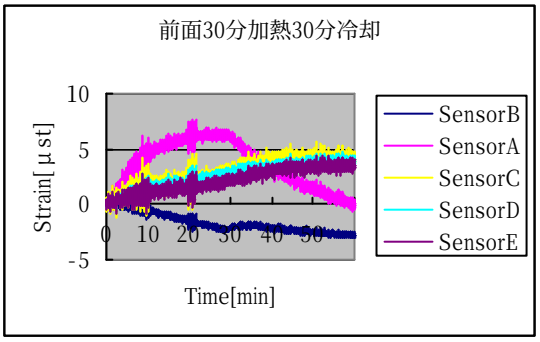
1.データ集

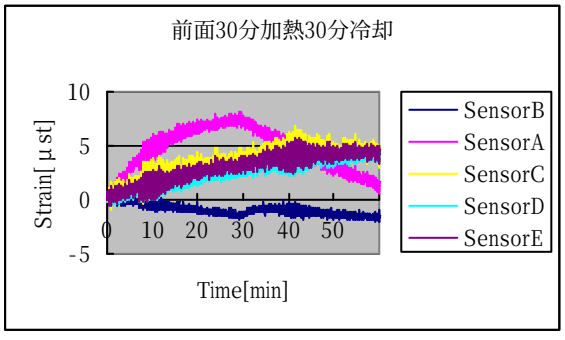
（a）熱アクチュエータの 30 分加熱 30 分冷却実験データ

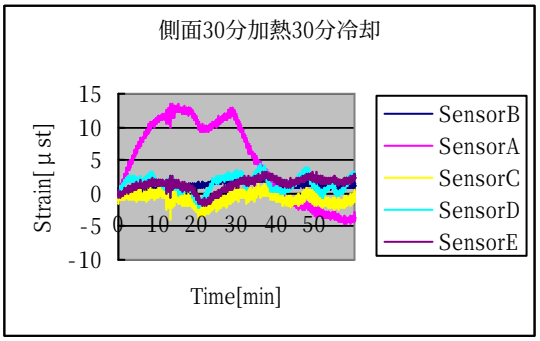












2.実験プログラム

歪測定プログラム

```
// 改良変形センサ Dlg.cpp : インプリメンテーション ファイル
//

#include "stdafx.h"
#include "改良変形センサ.h"
#include "改良変形センサ Dlg.h"
#include"FbiAd.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
HANDLE hDeviceHandle;
WORD wSmpData[30000][5];
int SamplingNum;
int i;
ULONG ulSmplNum;
ADSMPLREQ SmplConfig;
int
ch1,ch2,ch3,ch4,ch5,x1,x2,ynew1,ynew2,ynew3,ynew4,
ynew5,yold1,yold2,yold3,yold4,yold5,xa,SaveData;
int         Grafu;
CString A,B,C;
char dummy[256];
FILE *fp;
int SamplingPeriod=0;
int SamplingFreq=0;
#endif

/////////////////////////////////////////////////////////////////////////////
// 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
        //{{AFX_DATA(CAboutDlg)
        enum { IDD = IDD_ABOUTBOX };
        //}}AFX_DATA

        // ClassWizard は仮想関数のオーバーライドを生成します
        //{{AFX_VIRTUAL(CAboutDlg)
        protected:
        virtual void DoDataExchange(CDataExchange*  pDX);    // DDX/DDV のサポート
        //}}AFX_VIRTUAL

// インプリメンテーション
protected:
        //{{AFX_MSG(CAboutDlg)
        //}}AFX_MSG
        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()

/////////////////////////////////////////////////////////////////////////////
// CMyDlg ダイアログ

CMyDlg::CMyDlg(CWnd* pParent /*=NULL*/)
	: CDialog(CMyDlg::IDD, pParent)
{
	//{{AFX_DATA_INIT(CMyDlg)
	m_message = _T("");
	m_e1ch = _T("");
	m_e2ch = _T("");
	m_e3ch = _T("");
	m_e4ch = _T("");
	m_e5ch = _T("");
	m_ch1 = FALSE;
	m_ch10 = FALSE;
	m_ch11 = FALSE;
	m_ch12 = FALSE;
	m_ch13 = FALSE;
	m_ch14 = FALSE;
	m_ch2 = FALSE;
	m_ch3 = FALSE;
	m_ch4 = FALSE;
	m_ch5 = FALSE;
	m_ch6 = FALSE;
	m_ch7 = FALSE;
	m_ch8 = FALSE;
	m_ch9 = FALSE;
	//}}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 CMyDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
	//{{AFX_DATA_MAP(CMyDlg)
	DDX_Control(pDX, IDC_PICT, m_pict);
	DDX_Control(pDX, IDC_PICT5, m_5ch);
	DDX_Control(pDX, IDC_PICT4, m_4ch);
	DDX_Control(pDX, IDC_PICT3, m_3ch);
	DDX_Control(pDX, IDC_PICT2, m_2ch);
	DDX_Control(pDX, IDC_PICT1, m_1ch);
	DDX_Control(pDX, IDC_SAMPLING, m_sampling);
	DDX_Control(pDX, IDC_SMPNUM, m_smpnum);
	DDX_Control(pDX, IDC_SMPFREQ, m_smpfreq);
	DDX_Control(pDX, IDC_STOP, m_stop);
	DDX_Control(pDX, IDC_START, m_start);
	DDX_Control(pDX, IDC_SET, m_set);
	DDX_Control(pDX, IDC_ADCLOSE, m_adclose);
	DDX_Control(pDX, IDC_ADOPEN, m_adopen);
	DDX_Text(pDX, IDC_EDIT_MESSAGE, m_message);
	DDX_Text(pDX, IDC_CH1, m_e1ch);
	DDX_Text(pDX, IDC_CH2, m_e2ch);
	DDX_Text(pDX, IDC_CH3, m_e3ch);
	DDX_Text(pDX, IDC_CH4, m_e4ch);
	DDX_Text(pDX, IDC_CH5, m_e5ch);
	DDX_Check(pDX, IDC_CHECK1, m_ch1);
	DDX_Check(pDX, IDC_CHECK10, m_ch10);
	DDX_Check(pDX, IDC_CHECK11, m_ch11);
	DDX_Check(pDX, IDC_CHECK12, m_ch12);
	DDX_Check(pDX, IDC_CHECK13, m_ch13);
	DDX_Check(pDX, IDC_CHECK14, m_ch14);
	DDX_Check(pDX, IDC_CHECK2, m_ch2);
	DDX_Check(pDX, IDC_CHECK3, m_ch3);
	DDX_Check(pDX, IDC_CHECK4, m_ch4);
	DDX_Check(pDX, IDC_CHECK5, m_ch5);
	DDX_Check(pDX, IDC_CHECK6, m_ch6);
	DDX_Check(pDX, IDC_CHECK7, m_ch7);
	DDX_Check(pDX, IDC_CHECK8, m_ch8);
	DDX_Check(pDX, IDC_CHECK9, m_ch9);
	//}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CMyDlg, CDialog)
```

```
	//{{AFX_MSG_MAP(CMyDlg)
	ON_WM_SYSCOMMAND()
	ON_WM_PAINT()
	ON_WM_QUERYDRAGICON()
	ON_BN_CLICKED(IDC_ADOPEN, OnAdopen)
	ON_BN_CLICKED(IDC_ADCLOSE, OnAdclose)
	ON_BN_CLICKED(IDC_START, OnStart)
	ON_BN_CLICKED(IDC_SET, OnSet)
	ON_BN_CLICKED(IDC_RADIO1, OnRadio1)
	ON_BN_CLICKED(IDC_RADIO2, OnRadio2)
	ON_WM_TIMER()
	ON_BN_CLICKED(IDC_STOP, OnStop)
	ON_BN_CLICKED(IDC_SERVERCON, OnServercon)
	ON_BN_CLICKED(IDC_CHECK1, OnCheck1)
	ON_BN_CLICKED(IDC_CHECK2, OnCheck2)
	ON_BN_CLICKED(IDC_CHECK3, OnCheck3)
	ON_BN_CLICKED(IDC_CHECK4, OnCheck4)
	ON_BN_CLICKED(IDC_CHECK5, OnCheck5)
	ON_BN_CLICKED(IDC_CHECK6, OnCheck6)
	ON_BN_CLICKED(IDC_CHECK7, OnCheck7)
	ON_BN_CLICKED(IDC_CHECK8, OnCheck8)
	ON_BN_CLICKED(IDC_CHECK9, OnCheck9)
	ON_BN_CLICKED(IDC_CHECK10, OnCheck10)
	ON_BN_CLICKED(IDC_CHECK11, OnCheck11)
	ON_BN_CLICKED(IDC_CHECK12, OnCheck12)
	ON_BN_CLICKED(IDC_CHECK13, OnCheck13)
	ON_BN_CLICKED(IDC_CHECK14, OnCheck14)
	//}}AFX_MSG_MAP
END_MESSAGE_MAP()

/////////////////////////////////////////////////////////////////////////////
// CMyDlg メッセージ ハンドラ

BOOL CMyDlg::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_adclose.EnableWindow(FALSE);
	m_adopen.EnableWindow(TRUE);
		m_start.EnableWindow(FALSE);
	m_stop.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 CMyDlg::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 CMyDlg::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 CMyDlg::OnQueryDragIcon()
{
		return (HCURSOR) m_hIcon;
}

void CMyDlg::OnAdopen()
{
		// 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_adclose.EnableWindow(TRUE);
		m_adopen.EnableWindow(FALSE);
		m_message="AD ボードオープンしました";
		 hDeviceHandle = AdOpen("FBIAD1");
		if (hDeviceHandle == INVALID_HANDLE_VALUE){
			MessageBox("Device FBIAD1 は使用できません");
```

```
                exit(0);
        }
        AdGetSamplingConfig(hDeviceHandle,&S
mplConfig);
  SmplConfig.ulSmplNum = SamplingNum;
  SmplConfig.ulChCount =5;
        SmplConfig.ulSamplingMode          =
AD_IO_SAMPLING;
        SmplConfig.ulSingleDiff            =
AD_INPUT_SINGLE;
        SmplConfig.fSmplFreq= SamplingFreq;
        SmplConfig.SmplChReq[0].ulChNo = 1;
  SmplConfig.SmplChReq[0].ulRange = AD_5V;
  SmplConfig.SmplChReq[1].ulChNo = 2;
  SmplConfig.SmplChReq[1].ulRange = AD_5V;
  SmplConfig.SmplChReq[2].ulChNo = 3;
  SmplConfig.SmplChReq[2].ulRange = AD_5V;
        SmplConfig.SmplChReq[3].ulChNo = 4;
  SmplConfig.SmplChReq[3].ulRange = AD_5V;
        SmplConfig.SmplChReq[4].ulChNo = 5;
  SmplConfig.SmplChReq[4].ulRange = AD_5V;

        AdSetSamplingConfig(hDeviceHandle,&S
mplConfig);
        UpdateData(false);
}

void CMyDlg::OnAdclose()
{
        m_message="AD ボード閉じました";
        m_adclose.EnableWindow(FALSE);
        m_adopen.EnableWindow(TRUE);
        m_start.EnableWindow(FALSE);
  m_stop.EnableWindow(FALSE);
        m_set.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
         AdClose(hDeviceHandle);
         UpdateData(false);
}

void CMyDlg::OnStart()
{
        // 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_message="サンプリング開始します";
        m_start.EnableWindow(FALSE);
        m_stop.EnableWindow(TRUE);
        m_smpfreq.GetWindowText(B);
 SamplingFreq=atoi(B);
        m_smpnum.GetWindowText(C);
SamplingNum=atoi(C);
        if(Grafu==1){
SetTimer(1,45,NULL);

//1CH の色
 CBrush newBrush;
 CDC* pDC=m_1ch.GetDC();
 newBrush.CreateSolidBrush(RGB(255,0,0));
 pDC->SelectObject(&newBrush);
 pDC->Rectangle(0,0,10,10);
//2CH の色
        CDC* pDC2=m_2ch.GetDC();
        CBrush newBrush2;
newBrush2.CreateSolidBrush(RGB(0,0,255));
 pDC2->SelectObject(&newBrush2);
 pDC2->Rectangle(0,0,10,10);
        //3CH の色
        CDC* pDC3=m_3ch.GetDC();
        CBrush newBrush3;
newBrush3.CreateSolidBrush(RGB(0,255,0));
 pDC3->SelectObject(&newBrush3);
 pDC3->Rectangle(0,0,10,10);
        //4CH の色
        CDC* pDC4=m_4ch.GetDC();
        CBrush newBrush4;
newBrush4.CreateSolidBrush(RGB(0,255,255));
 pDC4->SelectObject(&newBrush4);
 pDC4->Rectangle(0,0,10,10);
        //5CH の色
        CDC* pDC5=m_5ch.GetDC();
        CBrush newBrush5;
newBrush5.CreateSolidBrush(RGB(255,0,255));
 pDC5->SelectObject(&newBrush5);
 pDC5->Rectangle(0,0,10,10);
        //座標軸
        CDC*pDC6=m_pict.GetDC();
        pDC6->MoveTo(0,200);
 pDC6->LineTo(410,200);
 pDC6->MoveTo(10,0);
```

```
    pDC6->LineTo(10,310);
    }
   if(SaveData==1){
            AdStartSampling(hDeviceHandle,
FLAG_SYNC);
            ulSmplNum=SamplingNum;
            AdGetSamplingData(hDeviceHandle,
&wSmpData[0][0], &ulSmplNum);
            fp = fopen("SampleData.txt","w+");
            if(fp==NULL){
                        fprintf(stderr,"¥n ¥t Cannot
Open SampleData.txt¥n¥t");
            }
            for (i=0; i<SamplingNum; i++)

            fprintf(fp,"%.2f %f %f
        %f %f %f
¥n",i*1/SmplConfig.fSmplFreq,30.0*(wSmpData[i][0]-
32768.0)/6553.6/*(10.0*wSmpData[i][0]/65536-5.0)*0.9
*/,
            (10.0*wSmpData[i][1]/65536-5.0)*0.9,(10.0
*wSmpData[i][2]/65536-5.0)*0.9,(10.0*wSmpData[i][3
]/65536-5.0)*0.9,(10.0*wSmpData[i][4]/65536-5.0)*0.9)
;
                        fclose(fp);
   m_message="サンプリング完了しました";
            AdStopSampling(hDeviceHandle);
}
            UpdateData(false);

}

void CMyDlg::OnSet()
{
            // 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_start.EnableWindow(TRUE);
  m_sampling.GetWindowText(A);
  SamplingPeriod=atoi(A);
  m_smpfreq.GetWindowText(B);
  SamplingFreq=atoi(B);

SamplingNum=SamplingPeriod*60*SamplingFreq;
  char dsp[100];
  sprintf(dsp,"%d",SamplingNum);
  m_smpnum.SetWindowText(dsp);
}

void CMyDlg::OnRadio1()
{
            // 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_sampling.EnableWindow(TRUE);
            m_smpnum.EnableWindow(TRUE);
            SaveData=1;
            Grafu=0;
}

void CMyDlg::OnRadio2()
{
            // 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_sampling.EnableWindow(FALSE);
            m_smpnum.EnableWindow(FALSE);
            SaveData=0;
            Grafu=1;

}

void CMyDlg::OnTimer(UINT nIDEvent)
{
            // TODO: この位置にメッセージ ハンドラ
用のコードを追加するかまたはデフォルトの処理を呼び
出してください

            AdStartSampling(hDeviceHandle,
FLAG_ASYNC);
            ulSmplNum=1000;
            AdGetSamplingData(hDeviceHandle,
&wSmpData[0][0], &ulSmplNum);
      sprintf(dummy,
"%f",(10.0*wSmpData[0][0]/65536-5.0)/2.0);
      m_e1ch=du mmy;
             sprintf(dummy,
"%f",(10.0*wSmpData[1][0]/65536-5.0)/2.0);
      m_e2ch=dummy;
             sprintf(dummy,
"%f",(10.0*wSmpData[2][0]/65536-5.0)/2.0);
      m_e3ch=dummy;
             sprintf(dummy,
```

```
"%f",(10.0*wSmpData[3][0]/65536-5.0)/2.0);
      m_e4ch=dummy;
            sprintf(dummy,
"%f",(10.0*wSmpData[4][0]/65536-5.0)/2.0);
      m_e5ch=dummy;
            //描画
            ch1=atoi(m_e1ch)*100.0;
   ch2=atoi(m_e2ch)*100.0;
            ch3=atoi(m_e3ch)*100.0;
            ch4=atoi(m_e4ch)*100.0;
            ch5=atoi(m_e5ch)*100.0;
            x1=x1+1;
            x2=x1;
            ynew1=((ch1/4)-200)*-1;
   ynew2=((ch2/4)-200)*-1;
            ynew3=((ch3/4)-200)*-1;
            ynew4=((ch4/4)-200)*-1;
            ynew5=((ch5/4)-200)*-1;
            CDC* pDC=m_pict.GetDC();
              CPen
BluePen,RedPen,GreenPen,BrackPen,CianPen,Mazen
daPen;

RedPen.CreatePen(PS_SOLID,1,RGB(255,0,0));
          pDC->SelectObject(&RedPen);
                pDC->MoveTo(xa,yold1);
                pDC->LineTo(x2,ynew1);
               yold1=ynew1;

BluePen.CreatePen(PS_SOLID,1,RGB(0,0,255));
          pDC->SelectObject(&BluePen);
                pDC->MoveTo(xa,yold2);
                pDC->LineTo(x2,ynew2);
               yold2=ynew2;

GreenPen.CreatePen(PS_SOLID,1,RGB(0,255,0));
          pDC->SelectObject(&GreenPen);
                pDC->MoveTo(xa,yold3);
                pDC->LineTo(x2,ynew3);
               yold3=ynew3;

CianPen.CreatePen(PS_SOLID,1,RGB(0,255,255));
          pDC->SelectObject(&CianPen);
                pDC->MoveTo(xa,yold4);
                pDC->LineTo(x2,ynew4);
               yold4=ynew4;

MazendaPen.CreatePen(PS_SOLID,1,RGB(255,0,255)
);
          pDC->SelectObject(&MazendaPen);
                pDC->MoveTo(xa,yold5);
                pDC->LineTo(x2,ynew5);
               yold5=ynew5;
                     xa=x2;
                     if(x1>410){
               CDC*pDC=m_pict.GetDC();
               CRect myRECT;
               m_pict.GetClientRect(myRECT);

pDC->FillSolidRect(myRECT,RGB(255,255,255));

pDC->SelectObject(&BrackPen);

            pDC->MoveTo(0,200);
            pDC->LineTo(410,200);
            pDC->MoveTo(10,0);
            pDC->LineTo(10,310);
          x1=9;
          xa=10;
        yold1=200;yold2=200;
yold3=200;yold4=200;yold5=200;
                          }
   UpdateData(false);
            CDialog::OnTimer(nIDEvent);
}

void CMyDlg::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
            m_message="測定を停止します";
             KillTimer(1);
             m_stop.EnableWindow(FALSE);
            m_start.EnableWindow(TRUE);
            UpdateData(false);
}

void CMyDlg::OnServercon()
{
/*          // 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");
            }*/
}

void CMyDlg::OnCheck1()
{
/*          // 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(m_Ch1 == TRUE){
                        m_Ch1 = FALSE;
              sData.fv[0] = double(0);
               m_message ="前面左ヒーター停止";
            }
            else if (m_Ch1 == FALSE){
                        m_Ch1 = TRUE;
              sData.fv[0] = double(1);
               m_message ="前面左ヒーター稼動";

            }
    sock->SockWrite(sData);
   UpdateData(false);
            sprintf(sData.command, "g"); */
}

void CMyDlg::OnCheck2()
{
            // 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(m_Ch2 == TRUE){
                        m_Ch2 = FALSE;
              sData.fv[0] = double(0);
               m_message ="前面右ヒーター停止";
            }
            else if (m_Ch2 == FALSE){
                        m_Ch2 = TRUE;
              sData.fv[0] = double(1);
               m_message ="前面右ヒーター稼動";

            }
    sock->SockWrite(sData);
    UpdateData(false);
            sprintf(sData.command, "h"); */
}

void CMyDlg::OnCheck3()
{
            // 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(m_Ch3 == TRUE){
                        m_Ch3 = FALSE;
              sData.fv[0] = double(0);
               m_message ="背面左ヒーター停止";
            }
            else if (m_Ch3 == FALSE){
                        m_Ch3 = TRUE;
              sData.fv[0] = double(1);
               m_message ="背面左ヒーター稼動";

            }
    sock->SockWrite(sData);
   UpdateData(false);
            sprintf(sData.command, "i"); */
}

void CMyDlg::OnCheck4()
{
            // 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(m_Ch4 == TRUE){
                        m_Ch4 = FALSE;
              sData.fv[0] = double(0);
               m_message ="背面右ヒーター停止";
            }
            else if (m_Ch4 == FALSE){
                        m_Ch4 = TRUE;
              sData.fv[0] = double(1);
               m_message ="背面右ヒーター稼動";

            }
    sock->SockWrite(sData);
   UpdateData(false);
            sprintf(sData.command, "j"); */
}
```

```
void CMyDlg::OnCheck5()
{
        // 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(m_Ch5 == TRUE){
                m_Ch5 = FALSE;
          sData.fv[0] = double(0);
          m_message ="側面前ヒーター停止";
        }
        else if (m_Ch5 == FALSE){
                m_Ch5 = TRUE;
          sData.fv[0] = double(1);
          m_message ="側面前ヒーター稼動";

        }
    sock->SockWrite(sData);
   UpdateData(false);
        sprintf(sData.command, "k"); */
}

void CMyDlg::OnCheck6()
{
        // 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(m_Ch6 == TRUE){
                m_Ch6 = FALSE;
          sData.fv[0] = double(0);
          m_message ="側面後ヒーター停止";
        }
        else if (m_Ch6 == FALSE){
                m_Ch6 = TRUE;
          sData.fv[0] = double(1);
          m_message ="側面後ヒーター稼動";

        }
    sock->SockWrite(sData);
   UpdateData(false);
        sprintf(sData.command, "l"); */
}

void CMyDlg::OnCheck7()
{
        // 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(m_Ch7 == TRUE){
                m_Ch7 = FALSE;
          sData.fv[0] = double(0);
          m_message ="ATC 側前ヒーター停止";
        }
        else if (m_Ch7 == FALSE){
                m_Ch7 = TRUE;
          sData.fv[0] = double(1);
          m_message ="ATC 側前ヒーター稼動";

        }
    sock->SockWrite(sData);
     UpdateData(false);
        sprintf(sData.command, "m"); */
}

void CMyDlg::OnCheck8()
{
        // 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(m_Ch8 == TRUE){
                m_Ch8 = FALSE;
          sData.fv[0] = double(0);
          m_message ="ATC 側後ヒーター停止";
        }
        else if (m_Ch8 == FALSE){
                m_Ch8 = TRUE;
          sData.fv[0] = double(1);
          m_message ="ATC 側後ヒーター稼動";

        }
    sock->SockWrite(sData);
   UpdateData(false);
        sprintf(sData.command, "n"); */
}

void CMyDlg::OnCheck9()
{
        // 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(m_ch9 == TRUE){
                ch9 = FALSE;
          sData.fv[0] = double(0);
          m_message ="前面冷却ファン停止";
        }
        else if (ch9 == FALSE){
```

```
                    ch9 = TRUE;
                sData.fv[0] = double(1);
                m_message ="前面冷却ファン稼動";

            }
    sock->SockWrite(sData);
   UpdateData(false);
            sprintf(sData.command, "a"); */
}

void CMyDlg::OnCheck10()
{
            // 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(m_ch10 == TRUE){
                    m_ch10 = FALSE;
                sData.fv[0] = double(0);
                m_message ="背面冷却ファン停止";
            }
            else if (m_ch10 == FALSE){
                    m_ch10 = TRUE;
                sData.fv[0] = double(1);
                m_message ="背面冷却ファン稼動";

            }
    sock->SockWrite(sData);
   UpdateData(false);
            sprintf(sData.command, "c"); */
}

void CMyDlg::OnCheck11()
{
            // 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(m_ch11 == TRUE){
                    m_ch11 = FALSE;
                sData.fv[0] = double(0);
                m_message ="側面前冷却ファン停止";
            }
            else if (m_ch11 == FALSE){
                    m_ch11 = TRUE;
                sData.fv[0] = double(1);
                m_message ="側面前冷却ファン稼動";

            }
    sock->SockWrite(sData);
   UpdateData(false);
            sprintf(sData.command, "b"); */
}

void CMyDlg::OnCheck12()
{
            // 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(m_ch12 == TRUE){
                    m_ch12 = FALSE;
                sData.fv[0] = double(0);
                m_message ="側面後冷却ファン停止";
            }
            else if (m_ch12 == FALSE){
                    m_ch12 = TRUE;
                sData.fv[0] = double(1);
                m_message ="側面後冷却ファン稼動";

            }
    sock->SockWrite(sData);
   UpdateData(false);
            sprintf(sData.command, "e"); */
}

void CMyDlg::OnCheck13()
{
            // 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(m_ch13 == TRUE){
                    m_ch13 = FALSE;
                sData.fv[0] = double(0);
                m_message ="ATC 側前冷却ファン停止";
            }
            else if (m_ch13 == FALSE){
                    m_ch13 = TRUE;
                sData.fv[0] = double(1);
                m_message ="ATC 側前冷却ファン稼動";

            }
    sock->SockWrite(sData);
   UpdateData(false);
            sprintf(sData.command, "f"); */
}
```

```
void CMyDlg::OnCheck14()
{
        // 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(m_ch14 == TRUE){
                    m_ch14 = FALSE;
          sData.fv[0] = double(0);
          m_message ="ATC 側後冷却ファン停止";
        }
        else if (m_ch14 == FALSE){
                    m_ch14 = TRUE;
          sData.fv[0] = double(1);
          m_message ="ATC 側後冷却ファン稼動";

        }
   sock->SockWrite(sData);
  UpdateData(false);
        sprintf(sData.command, "d"); */
}
```

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"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
int tu,xa,ya,yb,yc,y1,y2,y3,x1,xnew,xold,y4,ynew,yold;
int kakou,tikara,henkei,MappingModel;
#endif

#define READTIMER_ID        0
static S_DATA       sData,rData;
static HANDLE       g_hThread;
static CComm*       sock;
//static void session();

//static int counter;
static double n;
static char         dummy[256];
static bool StartFlag = false;

#include "pcpg46.h"
#include"windows.h"
#include"stdio.h"
#include"FbiAd.h"

#define AXIS_NUM 3

HANDLE hDeviceHandle;
ADSMPLCHREQ SmplChReq[3];
WORD wSmpData[3];

int adDataOrg[AXIS_NUM];
int adDataCal[AXIS_NUM];

double mstPerBit[AXIS_NUM];
double nonIntfMatrix[AXIS_NUM][AXIS_NUM];
double newtonPerMst[AXIS_NUM];
double
newtonPerMstMatrix[AXIS_NUM][AXIS_NUM];    /*
newtonPerMst * nonIntfMatrix */
double          sensorNon[AXIS_NUM][AXIS_NUM];
        /* newtonPerMstMatrix * newtonPerMst */
double meanForce[AXIS_NUM];
//int counter;
static int joystickStart = 0;
static int m_handlemode = -1;
pcpg46 pcpg;

        TCHAR
        szBuf[1024];
        BYTE
        bData[256];
        WORD
        wBsn, wAxis;
        PCPG46RESOURCE  ri;
        BYTE
        b;
        DWORD
        dwCount;
```

```
/////////////////////////////////////////////////////////////////////////////
// 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
	//{{AFX_DATA(CAboutDlg)
	enum { IDD = IDD_ABOUTBOX };
	//}}AFX_DATA

	// ClassWizard は仮想関数のオーバーライドを生成します
	//{{AFX_VIRTUAL(CAboutDlg)
	protected:
	virtual void DoDataExchange(CDataExchange* pDX);	// DDX/DDV のサポート
	//}}AFX_VIRTUAL

// インプリメンテーション
protected:
	//{{AFX_MSG(CAboutDlg)
	//}}AFX_MSG
	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_sm = 0;
	m_fm = 0;
	m_tuyox = _T("");
	m_tuyoy = _T("");
	m_tuyoz = _T("");
	m_fx = _T("");
	m_fy = _T("");
	m_fz = _T("");
	m_C1 = FALSE;
```

```
	m_C2 = FALSE;
	m_C3 = FALSE;
	m_C4 = FALSE;
	m_C5 = FALSE;
	m_C6 = FALSE;
	m_Ch1 = FALSE;
	m_Ch2 = FALSE;
	m_Ch3 = FALSE;
	m_Ch4 = FALSE;
	m_Ch5 = FALSE;
	m_Ch6 = FALSE;
	m_Ch7 = FALSE;
	m_Ch8 = FALSE;
	//}}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_BUTTON7, m_tikara);
	DDX_Control(pDX, IDC_BUTTON8, m_henkei);
	DDX_Control(pDX, IDC_BUTTON5, m_kakou);
	DDX_Control(pDX, IDC_BUTTON4, m_spinstart);
	DDX_Control(pDX, IDC_BUTTON6, m_spinstop);
	DDX_Control(pDX, IDC_BUTTON3, m_stop);
	DDX_Control(pDX, IDC_PICT, m_pict);
	DDX_Control(pDX, IDC_AXISX, m_axisx);
	DDX_Control(pDX, IDC_JOYSTICK_STOP, m_joystickStop);
	DDX_Control(pDX, IDC_JOYSTICK_START, m_joystickStart);
	DDX_Control(pDX, IDC_PCPGAD_CLOSE, m_pcpgadClose);
	DDX_Control(pDX, IDC_PCPGAD_INIT, m_pcpgadInit);
	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_BUTTON2, m_dncend);
	DDX_Control(pDX, IDC_BUTTON1, m_dncstart);
	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_EDIT1, m_sm);
        DDX_Text(pDX, IDC_EDIT2, m_fm);
        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_FORCEX, m_fx);
        DDX_Text(pDX, IDC_FORCEY, m_fy);
        DDX_Text(pDX, IDC_FORCEZ, m_fz);
        DDX_Check(pDX, IDC_CHECK10, m_C1);
        DDX_Check(pDX, IDC_CHECK11, m_C2);
        DDX_Check(pDX, IDC_CHECK12, m_C3);
        DDX_Check(pDX, IDC_CHECK13, m_C4);
        DDX_Check(pDX, IDC_CHECK14, m_C5);
        DDX_Check(pDX, IDC_CHECK15, m_C6);
        DDX_Check(pDX, IDC_CHECK1, m_Ch1);
        DDX_Check(pDX, IDC_CHECK2, m_Ch2);
        DDX_Check(pDX, IDC_CHECK3, m_Ch3);
        DDX_Check(pDX, IDC_CHECK4, m_Ch4);
        DDX_Check(pDX, IDC_CHECK5, m_Ch5);
        DDX_Check(pDX, IDC_CHECK6, m_Ch6);
        DDX_Check(pDX, IDC_CHECK7, m_Ch7);
        DDX_Check(pDX, IDC_CHECK8, m_Ch8);
        //}}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_RADIO1, OnRadio1)
        ON_BN_CLICKED(IDC_RADIO2, OnRadio2)
        ON_BN_CLICKED(IDC_RADIO3, OnRadio3)
        ON_BN_CLICKED(IDC_BUTTON1, OnButton1)
        ON_BN_CLICKED(IDC_BUTTON2, OnButton2)
        ON_EN_CHANGE(IDC_EDIT1, OnChangeEdit1)
        ON_EN_CHANGE(IDC_EDIT2,
```

```
OnChangeEdit2)
        ON_BN_CLICKED(IDC_FORCEBUTTON
1, OnForcebutton1)
        ON_BN_CLICKED(IDC_PCPGAD_INIT,
OnPcpgadInit)
        ON_BN_CLICKED(IDC_PCPGAD_CLOS
E, OnPcpgadClose)
        ON_BN_CLICKED(IDC_JOYSTICK_STO
P, OnJoystickStop)
        ON_BN_CLICKED(IDC_JOYSTICK_STA
RT, OnJoystickStart)
        ON_BN_CLICKED(IDC_HANDLEMODE,
OnHandlemode)
        ON_BN_CLICKED(IDC_BUTTON3,
OnButton3)
        ON_BN_CLICKED(IDC_JOYSTICKMOD
E, OnJoystickmode )
        ON_BN_CLICKED(IDC_BUTTON6,
OnSPINDLESTOP)
        ON_BN_CLICKED(IDC_BUTTON4,
OnSindleStart)
        ON_BN_CLICKED(IDC_CHECK10,
OnC1)
        ON_BN_CLICKED(IDC_CHECK11,
OnC2)
        ON_BN_CLICKED(IDC_CHECK12,
OnC3)
        ON_BN_CLICKED(IDC_CHECK13,
OnC4)
        ON_BN_CLICKED(IDC_CHECK14,
OnC5)
        ON_BN_CLICKED(IDC_CHECK15,
OnC6)
        ON_BN_CLICKED(IDC_CHECK1,
OnCh1)
        ON_BN_CLICKED(IDC_CHECK2,
OnCh2)
        ON_BN_CLICKED(IDC_CHECK3,
OnCh3)
        ON_BN_CLICKED(IDC_CHECK4,
OnCh4)
        ON_BN_CLICKED(IDC_CHECK5,
OnCh5)
        ON_BN_CLICKED(IDC_CHECK6,
OnCh6)
        ON_BN_CLICKED(IDC_CHECK7,
OnCh7)
        ON_BN_CLICKED(IDC_CHECK8,
OnCh8)
        ON_BN_CLICKED(IDC_BUTTON5,
Onkakou)
        ON_BN_CLICKED(IDC_BUTTON8,
Onhenkei)
        ON_BN_CLICKED(IDC_BUTTON7,
Ontikara)
        //}}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_ABOUTB
OX);
                if (!strAboutMenu.IsEmpty())
                {

        pSysMenu->AppendMenu(MF_SEPARATO
R);

        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_kakou.EnableWindow(FALSE);
    m_henkei.EnableWindow(FALSE);
        m_tikara.EnableWindow(FALSE);
        m_pcpgadInit.EnableWindow(FALSE);
        m_pcpgadClose.EnableWindow(FALSE);
        m_joystickStart.EnableWindow(FALSE);
        m_joystickStop.EnableWindow(FALSE);
        m_spinstart.EnableWindow(FALSE);
        m_spinstop.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 countThread( void )
{
   char pos[20];
            counter = 1 ;
            while( counter ) {

                  //m_relz=pos;
      sprintf(pos,"%5.3f", (double)counter++);
      // m_absx=pos;
       //  m_relx=pos;
     // dlg.UpdateData(false);
    //        SetDlgItemInt(NULL,IDC_RELX,
int(8),FALSE);
            //          CWnd * pEd = (CWnd *)
GetDlgItem(IDC_EDIT_MESSA);
            //
            SetDlgItemInt(NULL,IDC_RELX,
counter++,FALSE);
            //          SetDlgItemInt(          ghWnd,
IDD_COUNT, counter++, FALSE ) ;
                        // New function added by me
                        if((counter%100)==0)
                                    Beep(400,200);
                        Sleep( 100 ) ;


            }
            ExitThread( 0 ) ;
}
*/


void CMCOperDlg::OnHssbopen()
{
            // 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( hThread == NULL) {
                  threadID = 0 ;
                                                // create
a thread
            hThread = CreateThread( 0, 0,

            (LPTHREAD_START_ROUTINE)countTh
read,

            0, 0, &threadID ) ;
            }

            DWORD threadID;
```

```
*/
	sock = new CComm;
	if  (sock->Initialize("210.163.149.97")  == false) {
		sock->~CComm();
		AfxMessageBox("socket connect failed");
	}

	// timer
	SetTimer(READTIMER_ID,10,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_kakou.EnableWindow(TRUE);
   m_henkei.EnableWindow(TRUE);
	m_tikara.EnableWindow(TRUE);

	m_message += "Connection Established.¥r¥n";
	UpdateData(false);

}

void CMCOperDlg::OnNcselect()
{
	// 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

	char	buf[256];
	int		len,line_num=0;

	CFileDialog FileDLG(
		true,			//[ファイルを開く]ダイアログボックス
		"out",			//デフォルト拡張子
		"*.out",	//デフォルトファイル名
		OFN_FILEMUSTEXIST | OFN_HIDEREADONLY,	//ダイアログのカスタマイズ
		"NC プログラム(*.out)|*.out|すべて(*.*)|*.*||"	//フィルタ
		);

	// !! 初期ディレクトリ 要変更 !!
	FileDLG.m_ofn.lpstrInitialDir = "D:¥¥ljq¥¥teleokk¥¥MCOper";

	if(FileDLG.DoModal()==IDOK){
		CStdioFile fin(FileDLG.GetPathName(),CFile::modeRead);
		m_ncprog.Empty();

	while(fin.ReadString(buf,255)!=NULL){
			len = strlen(buf);
			buf[len-1] = '¥0';
			m_ncprog += buf;
			m_ncprog += "¥r¥n";
			line_num++;
		}
		NCNAME= FileDLG.GetFileName();
	}
   m_ncsend.EnableWindow(TRUE);
	UpdateData(false);

}

void CMCOperDlg::OnNcsend()
{
	// 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

	char buf[255];
	int  len, line_num;
	//int  Datlen, ProgNumOrg;

	// 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_message ="NC Program Send.";
	UpdateData(false);
```

```
            sprintf(sData.command, "NC program send %d", counter);
            sData.fv[0] = double(counter);
            sData.fv[1] = double(counter+1);
            sData.fv[2] = double(counter+2);
            sock->SockWrite(sData);

            // NC プログラムを受信し、MC へ転送
            int j = 0;

                        CStdioFile fin(NCNAME,CFile::modeRead | CFile::typeText);
                        m_ncprog.Empty();

            while(fin.ReadString( buf,255)!=NULL){
                                len = strlen(buf);
                strcpy(sData.command, buf);
                                buf[len-1] = '¥0';
                                m_ncprog += buf;
                                m_ncprog += "¥r¥n";
                                line_num++;

            if(sData.command[0] == 'O') {
              int i,num;
                                char NCNum[256];
                                for (i=0; i<len-2;i++){
                                    NCNum[i] = buf[i+1];
                                }
                                num = atoi(NCNum);
              sData.fv[0] = double(num);
              sprintf(NCNum, " NC prog num %d ",num);

            m_message = NCNum;
              UpdateData(false);
                                }else{
             sData.fv[0] = double(counter);
            sData.fv[1] = double(counter+1);
            sData.fv[2] = double(counter+2);
                                }
            sock->SockWrite(sData);
                UpdateData(false);
                }
                //File send over
                buf[0] = '%';
                strcpy(sData.command,buf);
                sock->SockWrite(sData);

            // NC プログラム転送終了
            m_message += "FileTrans Done.";
            m_message += "¥r¥n";
            UpdateData(false);
}

void CMCOperDlg::OnCiclestart()
{
            // 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, "Cycle start from client %d", counter);
   sData.fv[0] = double(counter);
   sData.fv[1] = double(counter+1);
   sData.fv[2] = double(counter+2);
    sock->SockWrite(sData);
   m_message ="CICLE START";
   UpdateData(false);
}

void CMCOperDlg::OnFeedhold()
{
            // 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, "Holding Feed from client . %d", counter);
   sData.fv[0] = double(counter);
   sData.fv[1] = double(counter+1);
   sData.fv[2] = double(counter+2);
    sock->SockWrite(sData);
    m_message ="FEED HOLD";
   UpdateData(false);
}

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 ;
/*                      if( hThread != NULL) {
                                // wait until the thread has finished

        while( WaitForSingleObject(hThread, 0) == WAIT_TIMEOUT)
                                //
        SetDlgItemText( hWnd, IDD_COUNT, "待つ." ) ;
                        //
        SetDlgItemText( hWnd, IDD_COUNT, "スレッド終わり." ) ;
                                hThread = NULL ;
                        }
                                */
 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)
{
        int pos, x, xmin, xmax;
        //char text[20];

        pScrollBar->GetScrollRange(&xmin, &xmax);
        pos = pScrollBar->GetScrollPos();
        x = pos;

        switch(nSBCode)
        {
           case SB_LINELEFT:
                        x-=2;
                break;
    case SB_LINERIGHT:
                        x+=2;
                break;
    case SB_PAGELEFT:
                        x-=10;
                break;
    case SB_PAGERIGHT:
                        x+=10;
```

```
                        break;
        case SB_THUMBTRACK:
                        x=nPos;
                break;
                        default:
                                return;
            }
            if(x<xmin) x=xmin;
            else if(x>xmax) x=xmax;
            if(x!=pos)
            {
                        pScrollBar->SetScrollPos(x);

                        if(pScrollBar == m_hsfeedover){
                        //              sprintf(text,    "%d",
x);
                                        m_feedover = x;
                                        m_message ="FEED
OVERRIDE";
                    sprintf(sData.command,      "Feed
override from client %d", x);
            sData.fv[0] = double(x);

                        }
                        else                    if(pScrollBar
==m_hsspindleover){
                //    sprintf(text, "%d", x);
                                        m_spindleover = x;
                                        m_message
="SPINDLE OVERRIDE";
                    sprintf(sData.command,   "Spindle
override from client %d", x);
            sData.fv[0] = double(x);
                        }
        sData.fv[1] = double(counter+1);
        sData.fv[2] = double(counter+2);
          sock->SockWrite(sData);

                        UpdateData(false);


            }
}


void CMCOperDlg::OnTimer(UINT nIDEvent)
{
            // TODO: この位置にメッセージ ハンドラ
用のコードを追加するかまたはデフォルトの処理を呼び
出してください
    int axis, index;
    double force[AXIS_NUM];


            if(nIDEvent == READTIMER_ID)
            {
      rData.command[0] = NULL;
                        sock->SockRead(&rData);
                        // Get Relative axis position.
                        if (rData.command[0] =='R'){
          sprintf(dummy,"%.3f",rData.fv[0]);
                m_relx = dummy;
        sprintf(dummy,"%.3f",rData.fv[1]);
                m_rely = dummy;
        sprintf(dummy,"%.3f",rData.fv[2]);
                m_relz = dummy;
        }else
              // Get machine axis position.
                        if (rData.command[0] =='M'){
        sprintf(dummy,"%.3f",rData.fv[0]);
                m_macx = dummy;
        sprintf(dummy,"%.3f",rData.fv[1]);
                m_macy = dummy;
        sprintf(dummy,"%.3f",rData.fv[2]);
                m_macz = dummy;
                        }else
                        // Get absolute axis position.
                        if (rData.command[0] =='A'){
        sprintf(dummy,"%.3f",rData.fv[0]);
                m_absx = dummy;
        sprintf(dummy,"%.3f",rData.fv[1]);
                m_absy = dummy;
        sprintf(dummy,"%.3f",rData.fv[2]);
                m_absz = dummy;
                        }else
              // Get remain moving  distance
              if (rData.command[0] =='N'){
        sprintf(dummy,"%.3f",rData.fv[0]);
                m_remx = dummy;
        sprintf(dummy,"%.3f",rData.fv[1]);
                m_remy = dummy;
         sprintf(dummy,"%.3f",rData.fv[2]);
```

```
                m_remz = dummy;
        }else
                if (rData.command[0] =='S'){
                        // Get feed rate and spindle
speed.
        sprintf(dummy,"%.3f",rData.fv[0]);
                m_feed = dummy;
        sprintf(dummy,"%.3f",rData.fv[1]);
           m_spindle = dummy;
                        }else
                                if
(rData.command[0] =='K'){ //追加
                        //力センサーデータ取得
        sprintf(dummy,"%.3f",rData.fv[0]);
                m_tuyox  = dummy;
        sprintf(dummy,"%.3f",rData.fv[1]);
         m_tuyoy  = dummy;

sprintf(dummy,"%.3f",rData.fv[2]);
         m_tuyoz  = dummy;
                        }
                        if(joystickStart == 1){
        AdInputAD(hDeviceHandle,          3,
AD_INPUT_SINGLE, &SmplChReq[0], &wSmpData);
                for (axis = 0; axis < AXIS_NUM;
axis++)
            force[axis] = 0.0;
            for (axis = 0; axis < AXIS_NUM; axis++) {
                    for   (index   =   0;   index   <
AXIS_NUM; index++) {
                        force[axis]                 +=
sensorNon[axis][index]*((int)wSmpData[index]-adDat
aOrg[index]);
                                        }
                        }
                        for   (axis   =   0;   axis   <
AXIS_NUM; axis++)
            meanForce[axis]+=force[axis];
                counter++;
                if(counter == 5){
                  for (axis = 0; axis < AXIS_NUM;
axis++)
            meanForce[axis]                     =
meanForce[axis]/10.0;
                    sprintf(dummy,
"%lf",meanForce[0]);
                m_fx=dummy;
                    sprintf(dummy,
"%lf",meanForce[1]);
                m_fy=dummy;
                    sprintf(dummy,
"%lf",meanForce[2]);
                m_fz=dummy;
                UpdateData(false);
               if (meanForce[0] >= 8.0){
                    if      (      FALSE      ==
pcpg.Pcpg46wDataFullWrite(
                                wBsn,      wAxis,
PCPG46_PLUS_PRESET_PULSE_DRIVE, 100 ) ) {

pcpg.ErrorMessage(wBsn);
                                return ;
                                }
                CButton*        radio1        =
(CButton*)GetDlgItem(IDC_AXISX);

CButton*                radio2                =
(CButton*)GetDlgItem(IDC_AXISY);

CButton*                radio3                =
(CButton*)GetDlgItem(IDC_AXISZ);

if(radio1->GetCheck()){

            sprintf(sData.command, "X%d", counter);

            sData.fv[0] = double(1*n);
                    sData.fv[1]             =
double(m_sm);
                            sData.fv[2]     =
double(m_fm);

sock->SockWrite(sData);
                sprintf(dummy,"S   %.0f   rpm
F  %.0f mm/min X 方向へ  %.3f  mm動かします
",sData.fv[1],sData.fv[2],sData.fv[0] );
                m_message = dummy;
                  UpdateData(false);

                                }
```

```
if(radio2->GetCheck()){

        sprintf(sData.command, "Y%d", counter);

        sData.fv[0] = double(1*n);
                sData.fv[1]     =
double(m_sm);
                        sData.fv[2]     =
double(m_fm);

sock->SockWrite(sData);
            sprintf(dummy,"S  %.0f  rpm
F %.0f mm/min Y 方向へ  %.3f  mm動かします
",sData.fv[1],sData.fv[2],sData.fv[0] );
            m_message = dummy;
            UpdateData(false);

                                    }

if(radio3->GetCheck()){

        sprintf(sData.command, "Z%d", counter);

        sData.fv[0] = double(1*n);
                sData.fv[1]     =
double(m_sm);
                        sData.fv[2]     =
double(m_fm);

sock->SockWrite(sData);
            sprintf(dummy,"S  %.0f  rpm
F %.0f mm/min Z 方向へ  %.3f  mm動かします
",sData.fv[1],sData.fv[2],sData.fv[0] );
            m_message = dummy;
            UpdateData(false);

                                    }
                                }else
        if (meanForce[0] <= -8.0){
                if      (       FALSE       ==
pcpg.Pcpg46wDataFullWrite(
                                wBsn,  wAxis,
PCPG46_MINUS_PRESET_PULSE_DRIVE, 100 ) ){
pcpg.ErrorMessage(wBsn);
                                return ;
                                }
            CButton*        radio1      =
(CButton*)GetDlgItem(IDC_AXISX);

CButton*            radio2          =
(CButton*)GetDlgItem(IDC_AXISY);

CButton*            radio3          =
(CButton*)GetDlgItem(IDC_AXISZ);

if(radio1->GetCheck()){

        sprintf(sData.command, "X%d", counter);

        sData.fv[0] = double(-1*n);
                sData.fv[1]     =
double(m_sm);
                        sData.fv[2]     =
double(m_fm);

sock->SockWrite(sData);
            sprintf(dummy,"S  %.0f  rpm
F %.0f mm/min X 方向へ  %.3f  mm動かします
",sData.fv[1],sData.fv[2],sData.fv[0] );
            m_message = dummy;
             UpdateData(false);

                                    }

if(radio2->GetCheck()){

        sprintf(sData.command, "Y%d", counter);

        sData.fv[0] = double(-1*n);
                sData.fv[1]     =
double(m_sm);
                        sData.fv[2]     =
double(m_fm);

sock->SockWrite(sData);
            sprintf(dummy,"S  %.0f  rpm
F %.0f mm/min Y 方向へ  %.3f  mm動かします
",sData.fv[1],sData.fv[2],sData.fv[0] );
```

```
                m_message = dummy;
                UpdateData(false);

                                        }

if(radio3->GetCheck()){

            sprintf(sData.command, "Z%d", counter);

            sData.fv[0] = double(-1*n);
                        sData.fv[1]          =
double(m_sm);
                              sData.fv[2]      =
double(m_fm);

sock->SockWrite(sData);
                  sprintf(dummy,"S   %.0f   rpm
F %.0f mm/min Z 方向へ  %.3f  mm動かします
",sData.fv[1],sData.fv[2],sData.fv[0] );
                m_message = dummy;
                UpdateData(false);

                                        }

                                    }
                counter = 0;
                for (axis = 0; axis < AXIS_NUM;
axis++)
          meanForce[axis] = 0.0;
                        }
                    }
                    //加工経路描画
if(kakou==1){

MappingModel=MM_HIMETRIC;
                        x1=atoi(m_macx);
                        y4=atoi(m_macy);
          xnew=(x1-272)*10;
                         ynew=(y4+347)*-1*10;

                        CDC* pDC=m_pict.GetDC();
                CPen RedPen;

RedPen.CreatePen(PS_SOLID,1,RGB(255,0,0));
pDC->SetMapMode(MappingModel);
            pDC->SelectObject(&RedPen);
                        pDC->LineTo(xold,yold);
                    pDC->MoveTo(xnew,ynew);
              xold=xnew;
                        yold=ynew;
                        }

            }
            //力センサー描画
            if(tikara==1){
                y1=atoi(m_tuyox);
                        y2=atoi(m_tuyoy);
                        y3=atoi(m_tuyoz);

                    tu=tu+1;

                int x1,yx,yy,yz;
                    x1=tu;
                    yx= y1+30;
                    yy= y2+90;
                    yz= y3+150;

          CDC* pDC=m_pict.GetDC();
                    CPen
BluePen,RedPen,GreenPen,BrackPen;

BluePen.CreatePen(PS_SOLID,1,RGB(0,0,255));

                    pDC->SelectObject(&BluePen);
                    pDC->MoveTo(ya,xa);
                    pDC->LineTo(yx,x1);

                    ya=yx;

          //CDC* pDC=m_pict.GetDC();

RedPen.CreatePen(PS_SOLID,1,RGB(255,0,0));
            pDC->SelectObject(&RedPen);
                    pDC->MoveTo(yb,xa);
                    pDC->LineTo(yy,x1);

                    yb=yy;

          //CDC* pDC=m_pict.GetDC();
```

```
GreenPen.CreatePen(PS_SOLID,1,RGB(0,255,0));
            pDC->SelectObject(&GreenPen);
                  pDC->MoveTo(yc,xa);
                  pDC->LineTo(yz,x1);

                  yc=yz;
                  xa=x1;
                  //x
                 if(tu>190){
                CDC*pDC=m_pict.GetDC();
                 CRect myRECT;
                m_pict.GetClientRect(myRECT);

pDC->FillSolidRect(myRECT,RGB(255,255,255));
                  //y
             /*
              CDC*pDC=m_pict.GetDC();
                   CRect myRECT;
                  m_pict.GetClientRect(myRECT);

pDC->FillSolidRect(myRECT,RGB(255,255,255));
                  //z

CDC*pDC3=m_pict3.GetDC();
                  CRect myRECT3;
                m_pict3.GetClientRect(myRECT3);

pDC3->FillSolidRect(myRECT3,RGB(255,255,255));*/

                  pDC->SelectObject(&BrackPen);
                  pDC->MoveTo(30,0);
           pDC->LineTo(30,190);
                        pDC->MoveTo(90,0);
                        pDC->LineTo(90,190);
                        pDC->MoveTo(150,0);
                        pDC->LineTo(150,190);
           x1=0;
           xa=0;
          ya=30; yb=90; yc=150;
                         }

                        //変形センサー描画
                        if(henkei==1){
                                 }
        UpdateData(false);
             }
             CDialog::OnTimer(nIDEvent);
}



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);


}

//Handle Mode Y+
void CMCOperDlg::OnNcsend8()
{
             // 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, "Y%d", counter);
       sData.fv[0] = double(1*n);
                        //sData.fv[1] = double(m_sm);
                        sData.fv[2] = double(m_fm);
                        sock->SockWrite(sData);
              sprintf(dummy,"F %.0f mm/min  Y方向へ  %.3f  mm動かします",sData.fv[2],sData.fv[0] );
       m_message = dummy;
       UpdateData(false);
}

//Handle Mode Y-
void CMCOperDlg::OnNcsend4()
{
             // 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, "Y%d", counter);
             sData.fv[0] = double(-1*n);
             //sData.fv[1] = double(m_sm);
```

```
            sData.fv[2] = double(m_fm);
            sock->SockWrite(sData);
    sprintf(dummy,"F %.0f mm/min Y方向へ %.3f
mm動かします",sData.fv[2],sData.fv[0] );
    m_message = dummy;
    UpdateData(false);
}

//Handle Mode X+
void CMCOperDlg::OnNcsend5()
{
            // 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, "X%d", counter);
            sData.fv[0] = double(1*n);
//          sData.fv[1] = double(m_sm);
            sData.fv[2] = double(m_fm);
            sock->SockWrite(sData);
    sprintf(dummy,"F %.0f mm/min X方向へ %.3f
mm動かします",sData.fv[2],sData.fv[0] );
    m_message = dummy;
    UpdateData(false);
}


//Handle Mode X-
void CMCOperDlg::OnNcsend6()
{
            // 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, "X%d", counter);
            sData.fv[0] = double(-1*n);
//          sData.fv[1] = double(m_sm);
            sData.fv[2] = double(m_fm);
            sock->SockWrite(sData);
    sprintf(dummy,"F %.0f mm/min X方向へ %.3f
mm動かします",sData.fv[2],sData.fv[0] );
    m_message = dummy;
    UpdateData(false);
}

//Handle Mode Z+
void CMCOperDlg::OnNcsend3()
{
            // 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, "Z%d", counter);
            sData.fv[0] = double(1*n);
//          sData.fv[1] = double(m_sm);
            sData.fv[2] = double(m_fm);
            sock->SockWrite(sData);
    sprintf(dummy,"F %.0f mm/min Z 方向へ %.3f
mm動かします",sData.fv[2],sData.fv[0] );
     m_message = dummy;
    UpdateData(false);
}

//Handle Mode Z-
void CMCOperDlg::OnNcsend7()
{
            // 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, "Z%d", counter);
            sData.fv[0] = double(-1*n);
//          sData.fv[1] = double(m_sm);
            sData.fv[2] = do uble(m_fm);
            sock->SockWrite(sData);
    sprintf(dummy,"F %.0f mm/min Z 方向へ %.3f
mm動かします",sData.fv[2],sData.fv[0] );
    m_message = dummy;
    UpdateData(false);
}

void CMCOperDlg::OnRadio1()
{
            // 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::OnRadio2()
{
            // 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::OnRadio3()
```

```
{
        // 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;

}

//DNC Start Button
void CMCOperDlg::OnButton1()
{
        // 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_spinstart.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);
                CButton* radio1 = (CButton*)GetDlgItem(IDC_RADIO1);
                radio1->SetCheck(1);
                CButton* radio2 = (CButton*)GetDlgItem(IDC_RADIO2);
                radio2->SetCheck(0);
                CButton* radio3 = (CButton*)GetDlgItem(IDC_RADIO3);
                radio3->SetCheck(0);
                n=0.001;
    m_sm = 500;
                m_fm = 100;
                UpdateData(false);
}

//DNC Stop Button
void CMCOperDlg::OnButton2()
{
        // 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);

m_spinstart.EnableWindow(FALSE);
          m_spinstop.EnableWindow(FALSE);
                UpdateData(false);
}


void CMCOperDlg::OnChangeEdit1()
{
        // 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_EDIT1);
                char count[10];
                int mm;
                mm=str->GetLine(0,count,10);
                count[mm]=0;
                m_sm=atoi(count);
```

```
                UpdateData(false);
}

void CMCOperDlg::OnChangeEdit2()
{
        // 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_EDIT2);
        char data[10];
        int m;
    m=str->GetLine(0,data,10);
        data[m]=0;
        m_fm=atoi(data);
                UpdateData(false);
}

void CMCOperDlg::OnForcebutton1()
{

        /*m_stop.EnableWindow(TRUE);
          m_foce.EnableWindow(FALSE);
                if(a1==3){
          CWnd* h=GetDlgItem(IDC_PICT);
      CDC*pDC=h->GetDC();
       pDC->MoveTo(30,0);
       pDC->LineTo(30,190);
       pDC->MoveTo(90,0);
       pDC->LineTo(90,190);
       pDC->MoveTo(150,0);
       pDC->LineTo(150,190);
                int y;
                for(y=60;y<=180;y=y+60){
                        pDC->MoveTo(27,y);
                pDC->LineTo(32,y);
                }
                for(y=60;y<=180;y=y+60){
                        pDC->MoveTo(87,y);
                pDC->LineTo(92,y);
                }
                for(y=60;y<=180;y=y+60){

          pDC->MoveTo(147,y);
                pDC->LineTo(152,y);
                }
                tu=0;
                xa=0;
                ya=30; yb=90; yc=150;
                u=1;}
          else if(a1==1){
                  x1=atoi(m_macx);
                  y4=atoi(m_macy);
                  xnew=x1-272;
                  ynew=y4+347;
                  CWnd*
h=GetDlgItem(IDC_PICT);
                CDC* pDC=m_pict.GetDC();
              CPen RedPen;

RedPen.CreatePen(PS_SOLID,1,RGB(255,0,0));
           pDC->SelectObject(&RedPen);
                pDC->LineTo(xnew,ynew);
                pDC->MoveTo(xold,yold);
          xold=xnew;
                  yold=ynew;
                  u=2;
                }
          else if(a1==2){
                        u=3;
                }*/
}


void CMCOperDlg::OnPcpgadInit()
{
        // 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

        //-------------------------------------------------------
-----
        // Pcpg46.Dll を開く
        //-------------------------------------------------------
```

```
-----
        if ( FALSE == pcpg.Pcpg46wDllOpen() ) {

        pcpg.ErrorMessage(PCPG46_BSN_AUTO);
                return ;                // DLL の
ロードまたはドライバのオープンに失敗しました
        }
        //-----------------------------------------------------
-----
        // ＤＬＬバージョン情報の表示
        //-----------------------------------------------------
-----
        if      (       FALSE   ==
pcpg.Pcpg46wGetLibVersion(bData)) {

        pcpg.ErrorMessage(PCPG46_BSN_AUTO);
                return ;
        }
        wsprintf( szBuf,
                _T("PCPG46¥n")
                _T("Dll Version %s"),
                bData );
        pcpg.ResultMessage( szBuf );
        //-----------------------------------------------------
-----
        // ドライババージョン情報の表示
        //-----------------------------------------------------
-----
        if      (       FALSE   ==
pcpg.Pcpg46wGetDrvVersion(bData)) {

        pcpg.ErrorMessage(PCPG46_BSN_AUTO);
                return ;
        }
        wsprintf( szBuf,
                _T("PCPG46¥n")
                _T("Drv Version %s"),
                bData );
        pcpg.ResultMessage( szBuf );
        //-----------------------------------------------------
-----
        // デバイスの使用を宣言する
        //-----------------------------------------------------
-----
        wBsn = PCPG46_BSN_AUTO;
        // 空きデバイス自動検索を指定
        if      (       FALSE   ==
pcpg.Pcpg46wCreate(&wBsn) ){

        pcpg.ErrorMessage(PCPG46_BSN_AUTO);
                return ;
        }
        wsprintf( szBuf,
                _T("PCPG46 Created¥n")
                _T("BSN Number %d"), wBsn );
        pcpg.ResultMessage( szBuf );
        //-----------------------------------------------------
-----
        // リソース情報を表示
        //-----------------------------------------------------
-----
        if      (       FALSE   ==
pcpg.Pcpg46wGetResource(wBsn,&ri)){
                pcpg.ErrorMessage(wBsn);
                return ;
        }
        wsprintf(szBuf, _T("Board  Name: %s¥n")

        _T("IO Address : %04Xh-%04Xh¥n"),

        szPcpg46,

        ri.dwIOPortBase[1],ri.dwIOPortBase[1]   +
ri.dwIOPortLength[1]- 1 );
        pcpg.ResultMessage(szBuf);


    //-----------------------------------------------------
        // 第１軸を選択
        //-----------------------------------------------------
-----
        wAxis = PCPG46_AXIS_1;
        //-----------------------------------------------------
-----
        // 制御軸の初期化
        //-----------------------------------------------------
-----
        if      (       FALSE   ==
pcpg.Pcpg46_InitAxis(wBsn,wAxis)){
                pcpg.ErrorMessage(wBsn);
```

```
                return ;
        }
        wsprintf( szBuf,
                _T("PCPG46¥n")
                _T("制御軸 %d を初期化しました"), wAxis );
        pcpg.ResultMessage( szBuf );
    hDeviceHandle = AdOpen("FBIAD1");
        if (hDeviceHandle == INVALID_HANDLE_VALUE){
                MessageBox("Device FBIAD1 は使用できません");
        }
        else{
                MessageBox("AD ボードオープンしました");
        }
        SetMatrix();
        GetOrgAD();
        GetCalAD();
        CalcMstPerBit();
    CalcSensorNon();
    m_pcpgadInit.EnableWindow(FALSE);
   // m_pcpgadClose.EnableWindow(TRUE);

m_joystickStart.EnableWindow(TRUE);
          //
m_joystickStop.EnableWindow(TRUE);

}

/* 非干渉化行列のセット*/
void CMCOperDlg::SetMatrix()
{
    int axis, index;

    nonIntfMatrix[0][0] = 1.03371;
    nonIntfMatrix[0][1] = -0.725237;
      nonIntfMatrix[0][2] = 0.186736;
    nonIntfMatrix[1][0] = -0.148806;
    nonIntfMatrix[1][1] = 0.722155;
    nonIntfMatrix[1][2] = 0.012234;
    nonIntfMatrix[2][0] = 1.10853;
    nonIntfMatrix[2][1] = -0.34993;
    nonIntfMatrix[2][2] = -1.28004;

      newtonPerMst[0] = 1.0;
    newtonPerMst[1] = 1.0;
    newtonPerMst[2] = 1.0;
    for (axis = 0; axis < AXIS_NUM; axis++) {
            for (index = 0; index < AXIS_NUM; index++) {
              newtonPerMstMatrix[axis][index] =
              newtonPerMst[axis] * nonIntfMatrix[axis][index];
            }
      }
} /* SetMatrix() */


/* 0mst. での A/D の出力を測定する,結果は adDataOrg に格納 */
void CMCOperDlg::GetOrgAD()
{
    static int meani = 10;          /* meani 回の平均を取る*/
    int  mi, axis;
    int   tmpDataOrg[AXIS_NUM];

    /* tmpDataOrg の初期化 */
     for (axis = 0; axis < AXIS_NUM; axis++) tmpDataOrg[axis] = 0;
            wsprintf( szBuf,
                    _T("PCPG46¥n")
                    _T("Balance Strain Amp. and Hit Return"));
            pcpg.ResultMessage( szBuf );

    /* Sample the data from each axis of AD converter */
    /* Calculate the sum of each axis */
    SmplChReq[0].ulChNo = 1;
    SmplChReq[0].ulRange = AD_5V;
    SmplChReq[1].ulChNo = 2;
    SmplChReq[1].ulRange = AD_5V;
    SmplChReq[2].ulChNo = 3;
    SmplChReq[2].ulRange = AD_5V;

            for (mi = 0; mi < meani; mi++) {
        AdInputAD(hDeviceHandle, 3,
```

```
AD_INPUT_SINGLE, &SmplChReq[0], &wSmpData);
                for (axis = 0; axis < AXIS_NUM;
axis++)
                tmpDataOrg[axis]                    =
tmpDataOrg[axis] +(int)wSmpData[axis];
                        Sleep(100);
    }
  /* Calculate the avarage of each axis and Print
them */
    for (axis = 0; axis < AXIS_NUM; axis++) {
              adDataOrg[axis]                       =
(int)(tmpDataOrg[axis] / meani);
    }
          sprintf(dummy, "%d",adDataOrg[0]);
  m_fx=dummy;
          sprintf(dummy, "%d",adDataOrg[1]);
  m_fy=dummy;
          sprintf(dummy, "%d",adDataOrg[2]);
  m_fz=dummy;
  UpdateData(false);
} /* GetOrgAD() */


/*  100mst.での A/D の出力を測定する，結果は
adDataCal に格納*/
void CMCOperDlg::GetCalAD()
{
  static int meani = 10;          /* meani 回の平均を
取る*/
  int  mi, axis;
  int  tmpDataCal[AXIS_NUM];

  /* tmpDataCal の初期化*/
    for (axis = 0; axis < AXIS_NUM; axis++)
tmpDataCal[axis] = 0;
          wsprintf( szBuf,
                  _T("PCPG46¥n")
                  _T("Turn on 100 micro strain
calibration switch and Hit Return"));
          pcpg.ResultMessage( szBuf );

          SmplChReq[0].ulChNo = 1;
  SmplChReq[0].ulRange = AD_5V;
   SmplChReq[1].ulChNo = 2;
  SmplChReq[1].ulRange = AD_5V;
  SmplChReq[2].ulChNo = 3;
  SmplChReq[2].ulRange = AD_5V;

  for (mi = 0; mi < meani; mi++) {
    AdInputAD(hDeviceHandle,                3,
AD_INPUT_SINGLE, &SmplChReq[0], &wSmpData);
          for (axis = 0; axis < AXIS_NUM; axis++)
            tmpDataCal[axis]                    =
tmpDataCal[axis] +(int)wSmpData[axis];
          Sleep(100);
    }

  for (axis = 0; axis < AXIS_NUM; axis++) {
          adDataCal[axis]                       =
(int)(tmpDataCal[axis] / meani);
    }
          sprintf(dummy, "%d",adDataCal[0]);
  m_fx=dummy;
          sprintf(dummy, "%d",adDataCal[1]);
  m_fy=dummy;
          sprintf(dummy, "%d",adDataCal[2]);
  m_fz=dummy;
  UpdateData(false);
          wsprintf( szBuf,
                  _T("PCPG46¥n")
                  _T("Turn off Cal and Hit
return¥n"));
          pcpg.ResultMessage( szBuf );

} /* GetCalAD() */


/* calibration の結果より１ビット当たりの歪の計算*/
void CMCOperDlg::CalcMstPerBit()
{
  int axis;

   /*  １ビット当たりの歪の計算(100ust の場合) */
  for (axis = 0; axis < AXIS_NUM; axis++) {
          mstPerBit[axis]
          =  (double)100.0  /  (adDataCal[axis]  -
adDataOrg[axis]);
    }
          sprintf(dummy, "%lf",mstPerBit[0]);
  m_fx=dummy;
```

```
            sprintf(dummy, "%lf",mstPerBit[1]);
    m_fy=dummy;
            sprintf(dummy, "%lf",mstPerBit[2]);
    m_fz=dummy;
    UpdateData(false);

} /* CalcMstPerBit() */

/* newtonPerMstMatrix × mstPerBit = sensorNon を
計算する*/
void CMCOperDlg::CalcSensorNon()
{
    int axis, index;

  /* 非干渉化行列×１ビット当たりの歪の計算*/
  for (axis = 0; axis < AXIS_NUM; axis++) {
    for (index = 0; index < AXIS_NUM; index++) {
      sensorNon[axis][index]
            =     newtonPerMstMatrix[axis][index]    *
mstPerBit[axis];
    }
  }
} /* CalcSensorNon() */



void CMCOperDlg::OnPcpgadClose()
{
            // 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

   AdClose(hDeviceHandle);

            //-------------------------------------------------------
-----
            // デバイスを解放する
            //-------------------------------------------------------
-----
            if ( FALSE == pcpg.Pcpg46wClose(wBsn)){
                        pcpg.ErrorMessage(wBsn);
                        return ;
            }
            wsprintf( szBuf,
                        _T("PCPG46 Close¥n")
                        _T("Bsn %d"), wBsn );
            pcpg.ResultMessage( szBuf );
            //-------------------------------------------------------
-----
            // Pcpg46.Dll を閉じる
            //-------------------------------------------------------
-----
            if ( FALSE == pcpg.Pcpg46wDllClose()) {

            pcpg.ErrorMessage(PCPG46_BSN_AUTO);
                        return ;
                                                // DLL の
アンロードに失敗しました
            }

    m_pcpgadInit.EnableWindow(TRUE);
    m_pcpgadClose.EnableWindow(FALSE);
    m_joystickStart.EnableWindow(FALSE);
    m_joystickStop.EnableWindow(FALSE);
            return ;

}

void CMCOperDlg::OnJoystickStop()
{
            // 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
//          KillTimer(1);
  joystickStart = 0;
  m_pcpgadInit.EnableWindow(FALSE);
  m_pcpgadClose.EnableWindow(TRUE);
  m_joystickStart.EnableWindow(TRUE);
  m_joystickStop.EnableWindow(FALSE);

}

void CMCOperDlg::OnJoystickStart()
{
            // 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

    int axis;

    counter = 0;
    for (axis = 0; axis < AXIS_NUM; axis++)
      meanForce[axis] = 0.0;
    joystickStart = 1;
```

```
  // SetTimer(1,5,NULL);
   m_pcpgadInit.EnableWindow(FALSE);
   m_pcpgadClose.EnableWindow(FALSE);
    m_joystickStart.EnableWindow(FALSE);
   m_joystickStop.EnableWindow(TRUE);
}


void CMCOperDlg::OnHandlemode()
{
            // 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_handlemode = 1;
               m_pcpgadInit.EnableWindow(FALSE);

m_pcpgadClose.EnableWindow(FALSE);

m_joystickStart.EnableWindow(FALSE);

m_joystickStop.EnableWindow(FALSE);
                m_xp.EnableWindow(TRUE);
                m_yp.EnableWindow(TRUE);
                m_zp.EnableWindow(TRUE);
                m_xm.EnableWindow(TRUE);
                m_ym.EnableWindow(TRUE);
                m_zm.EnableWin dow(TRUE);
}

void CMCOperDlg::OnButton3()
{
            // 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_stop.EnableWindow(FALSE);
   m_henkei.EnableWindow(TRUE);
            m_kakou.EnableWindow(TRUE);
   m_tikara.EnableWindow(TRUE);
            m_foce.EnableWindow(FALSE);
            CDC*pDC=m_pict.GetDC();
            CRect myRECT;
            m_pict.GetClientRect(myRECT);
            pDC->FillSolidRect(myRECT,RGB(255,255
,255));
    henkei=0;
            kakou=0;
            tikara=0;
}

void CMCOperDlg::OnJoystickmode()
{
            // 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_handlemode = 0;
               m_pcpgadInit.EnableWindow(TRUE);
              //
m_pcpgadClose.EnableWindow(TRUE);
             //
m_joystickStart.EnableWindow(TRUE);
              //
m_joystickStop.EnableWindow(TRUE);
               m_xp.EnableWindow(FALSE);
               m_yp.EnableWindow(FALSE);
               m_zp.EnableWindow(FALSE);
                m_xm.EnableWindow(FALSE);
               m_ym.EnableWindow(FALSE);
               m_zm.EnableWindow(FALSE);


}

void CMCOperDlg::OnSPINDLESTOP()
{
            // 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_spinstart.EnableWindow(TRUE);
            m_spinstop.EnableWindow(FALSE);
            sprintf(sData.command, "L Spindle stop
from client %d", counter);
    sock->SockWrite(sData);
   KillTimer(READTIMER_ID);

   m_message ="SPINDLE STOP";
   UpdateData(false);
}

void CMCOperDlg::OnSindleStart()
{
            // 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_spinstop.EnableWindow(TRUE);
```

```
            m_spinstart.EnableWindow(FALSE);
            sprintf(sData.command, "M SPINDLE
START %d", counter);
            sData.fv[0] = double(m_sm);
    sock->SockWrite(sData);
   KillTimer(READTIMER_ID);

   m_message ="SPINDLE START";
   UpdateData(false);
}

void CMCOperDlg::OnCStart()
{
            // 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%d", counter);
            if(m_C1 == TRUE){
              sData.fv[0] = double(1);
            }
            else if (m_C1 == FALSE){
              sData.fv[0] = double(0);
            }
            else if (m_C2 == TRUE){
              sData.fv[1] = double(1);
            }
            else if (m_C2 == FALSE){
              sData.fv[1] = double(0);
            }
            else if (m_C3 == TRUE){
              sData.fv[2] = double(1);
            }
            else if (m_C3 == FALSE){
              sData.fv[2] = double(0);
            }
            else if (m_C4 == TRUE){
              sData.fv[3] = double(1);
            }
            else if (m_C4 == FALSE){
              sData.fv[3] = double(0);
            }
            else if (m_C5 == TRUE){
              sData.fv[4]= double(1);
            }
            else if (m_C5 == FALSE){
              sData.fv[4] = double(0);
            }
            else if (m_C6 == TRUE){
              sData.fv[5] = double(1);
            }
            else if (m_C6 == FALSE){
               sData.fv[5] = double(0);
            }
            else if (m_C7 == TRUE){
              sData.fv[6] = double(1);
            }
            else if (m_C7 == FALSE){
              sData.fv[6] = double(0);
            }
            else if (m_C8 == TRUE){
              sData.fv[7] = double(1);
            }
            else if (m_C8 == FALSE){
              sData.fv[7] = double(0);
            }
            else if (m_C9 == TRUE){
              sData.fv[8]= double(1);
            }
            else if (m_C9 == FALSE){
              sData.fv[8] = double(0);
            }
            else if (m_C10 == TRUE){
              sData.fv[9] = double(1);
            }
            else if (m_C10 == FALSE){
              sData.fv[9] = double(0);
            }
            else if (m_C11 == TRUE){
              sData.fv[10] = double(1);
            }
            else if (m_C11 == FALSE){
              sData.fv[10] = double(0);
            }
            else if (m_C12 == TRUE){
              sData.fv[11] = double(1);
            }
            else if (m_C12 == FALSE){
              sData.fv[11] = double(0);
            }
            else if (m_C13 == TRUE){
```

```
            sData.fv[12]= double(1);
           }
           else if (m_C13 == FALSE){
            sData.fv[12] = double(0);
           }
           else if (m_C14 == TRUE){
            sData.fv[13] = double(1);
           }
           else if (m_C14 == FALSE){
            sData.fv[13] = double(0);
           }
           sock->SockWrite(sData);
           KillTimer(READTIMER_ID);
   UpdateData(false);*/
}


void CMCOperDlg::OnC1()
{
  // 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(m_C1 == TRUE){
                    m_C1 = FALSE;
            sData.fv[0] = double(0);
            m_message ="前面冷却ファン停止";
           }
           else if (m_C1 == FALSE){
                    m_C1 = TRUE;
            sData.fv[0] = double(1);
            m_message ="前面冷却ファン稼動";

           }
    sock->SockWrite(sData);
   KillTimer(READTIMER_ID);
   UpdateData(false);
           sprintf(sData.command, "a");
}

void CMCOperDlg::OnC2()
{
           // 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(m_C2 == TRUE){
                    m_C2 = FALSE;
            sData.fv[0] = double(0);
            m_message ="側面前冷却ファン停止";
           }
           else if (m_C2 == FALSE){
                    m_C2 = TRUE;
            sData.fv[0] = double(1);
            m_message ="側面前冷却ファン稼動";

           }
    sock->SockWrite(sData);
   KillTimer(READTIMER_ID);
   UpdateData(false);
           sprintf(sData.command, "b");

}

void CMCOperDlg::OnC3()
{
           // 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(m_C3 == TRUE){
                    m_C3 = FALSE;
            sData.fv[0] = double(0);
            m_message ="背面冷却ファン停止";
           }
           else if (m_C3 == FALSE){
                    m_C3 = TRUE;
            sData.fv[0] = double(1);
            m_message ="背面冷却ファン稼動";

           }
    sock->SockWrite(sData);
   KillTimer(READTIMER_ID);
   UpdateData(false);
           sprintf(sData.command, "c");

}

void CMCOperDlg::OnC4()
{
           // 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(m_C4 == TRUE){
                    m_C4 = FALSE;
            sData.fv[0] = double(0);
```

```
                m_message ="ATC 側前冷却ファン停止";
                }
                else if (m_C4 == FALSE){
                        m_C4 = TRUE;
                 sData.fv[0] = double(1);
                 m_message ="ATC 側前冷却ファン稼動";

                }
    sock->SockWrite(sData);
   KillTimer(READTIMER_ID);
   UpdateData(false);
                sprintf(sData.command, "f");

}

void CMCOperDlg::OnC5()
{
                // 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(m_C5 == TRUE){
                        m_C5 = FALSE;
                 sData.fv[0] = double(0);
                 m_message ="側面後冷却ファン停止";
                }
                else if (m_C5 == FALSE){
                        m_C5 = TRUE;
                 sData.fv[0] = double(1);
                 m_message ="側面後冷却ファン稼動";

                }
    sock->SockWrite(sData);
   KillTimer(READTIMER_ID);
   UpdateData(false);
                sprintf(sData.command, "e");

}

void CMCOperDlg::OnC6()
{
                // 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(m_C6 == TRUE){
                        m_C6 = FALSE;
                 sData.fv[0] = double(0);
                 m_message ="ATC 側後冷却ファン停止";
                }
                else if (m_C6 == FALSE){
                        m_C6 = TRUE;
                 sData.fv[0] = double(1);
                 m_message ="ATC 側後冷却ファン稼動";

                }
    sock->SockWrite(sData);
   KillTimer(READTIMER_ID);
   UpdateData(false);
                sprintf(sData.command, "d");

}

void CMCOperDlg::OnCh1()
{
                // 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(m_Ch1 == TRUE){
                        m_Ch1 = FALSE;
                 sData.fv[0] = double(0);
                 m_message ="前面左ヒーター停止";
                }
                else if (m_Ch1 == FALSE){
                        m_Ch1 = TRUE;
                 sData.fv[0] = double(1);
                 m_message ="前面左ヒーター稼動";

                }
    sock->SockWrite(sData);
   KillTimer(READTIMER_ID);
   UpdateData(false);
                sprintf(sData.command, "g");

}

void CMCOperDlg::OnCh2()
{
                // 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(m_Ch2 == TRUE){
                        m_Ch2 = FALSE;
                 sData.fv[0] = double(0);
                 m_message ="前面右ヒーター停止";
                }
```

```
		else if (m_Ch2 == FALSE){
			m_Ch2 = TRUE;
		  sData.fv[0] = double(1);
		  m_message ="前面右ヒーター稼動";

		}
	 sock->SockWrite(sData);
	KillTimer(READTIMER_ID);
	UpdateData(false);
		sprintf(sData.command, "h");

}

void CMCOperDlg::OnCh3()
{
		// 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(m_Ch3 == TRUE){
			m_Ch3 = FALSE;
		  sData.fv[0] = double(0);
		  m_message ="背面左ヒーター停止";
		}
		else if (m_Ch3 == FALSE){
			m_Ch3 = TRUE;
		  sData.fv[0] = double(1);
		  m_message ="背面左ヒーター稼動";

		}
	 sock->SockWrite(sData);
	KillTimer(READTIMER_ID);
	UpdateData(false);
		sprintf(sData.command, "i");

}

void CMCOperDlg::OnCh4()
{
		// 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(m_Ch4 == TRUE){
			m_Ch4 = FALSE;
		  sData.fv[0] = double(0);
		  m_message ="背面右ヒーター停止";
		}
		else if (m_Ch4 == FALSE){
			m_Ch4 = TRUE;
		  sData.fv[0] = double(1);
		  m_message ="背面右ヒーター稼動";

		}
	 sock->SockWrite(sData);
	KillTimer(READTIMER_ID);
	UpdateData(false);
		sprintf(sData.command, "j");

}

void CMCOperDlg::OnCh5()
{
		// 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(m_Ch5 == TRUE){
			m_Ch5 = FALSE;
		  sData.fv[0] = double(0);
		  m_message ="側面前ヒーター停止";
		}
		else if (m_Ch5 == FALSE){
			m_Ch5 = TRUE;
		  sData.fv[0] = double(1);
		  m_message ="側面前ヒーター稼動";

		}
	 sock->SockWrite(sData);
	KillTimer(READTIMER_ID);
	UpdateData(false);
		sprintf(sData.command, "k");

}

void CMCOperDlg::OnCh6()
{
		// 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(m_Ch6 == TRUE){
			m_Ch6 = FALSE;
		  sData.fv[0] = double(0);
		  m_message ="側面後ヒーター停止";
		}
		else if (m_Ch6 == FALSE){
			m_Ch6 = TRUE;
```

```
            sData.fv[0] = double(1);
            m_message ="側面後ヒーター稼動";

        }
    sock->SockWrite(sData);
   KillTimer(READTIMER_ID);
   UpdateData(false);
        sprintf(sData.command, "l");

}

void CMCOperDlg::OnCh7()
{
        // 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(m_Ch7 == TRUE){
            m_Ch7 = FALSE;
          sData.fv[0] = double(0);
          m_message ="ATC 側前ヒーター停止";
        }
        else if (m_Ch7 == FALSE){
            m_Ch7 = TRUE;
          sData.fv[0] = double(1);
          m_message ="ATC 側前ヒーター稼動";

        }
    sock->SockWrite(sData);
   KillTimer(READTIMER_ID);
   UpdateData(false);
        sprintf(sData.command, "m");

}

void CMCOperDlg::OnCh8()
{
        // 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(m_Ch8 == TRUE){
            m_Ch8 = FALSE;
          sData.fv[0] = double(0);
          m_message ="ATC 側後ヒーター停止";
        }
        else if (m_Ch8 == FALSE){
            m_Ch8 = TRUE;
          sData.fv[0] = double(1);
          m_message ="ATC 側後ヒーター稼動";

        }
    sock->SockWrite(sData);
   KillTimer(READTIMER_ID);
   UpdateData(false);
        sprintf(sData.command, "n");

}

void CMCOperDlg::Onkakou()
{
        // 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
        kakou=1;
   m_kakou.EnableWindow(FALSE);
   m_henkei.EnableWindow(FALSE);
        m_tikara.EnableWindow(FALSE);
        m_stop.EnableWindow(TRUE);
}

void CMCOperDlg::O nhenkei()
{
        // 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_kakou.EnableWindow(FALSE);
   m_henkei.EnableWindow(FALSE);
        m_tikara.EnableWindow(FALSE);
        m_stop.EnableWindow(TRUE);
}

void CMCOperDlg::Ontikara()
{
        // 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

    tikara=1;
        m_kakou.EnableWindow(FALSE);
   m_henkei.EnableWindow(FALSE);
        m_tikara.EnableWindow(FALSE);
        m_stop.EnableWindow(TRUE);
        CWnd* h=GetDlgItem(IDC_PICT);
      CDC*pDC=h->GetDC();
       pDC->MoveTo(30,0);
```

```
p D C ->LineTo(30,190);
p D C ->MoveTo(90,0);
p D C ->LineTo(90,190);
p D C ->MoveTo(150,0);
p D C ->LineTo(150,190);
                int y;
                for(y=60;y<=180;y=y+60){
                        pDC->MoveTo(27,y);
                pDC->LineTo(32,y);
                }
                for(y=60;y<=180;y=y+60){
                        pDC->MoveTo(87,y);
                pDC->LineTo(92,y);
                }
                for(y=60;y<=180;y=y+60){

    pDC->MoveTo(147,y);
                pDC->LineTo(152,y);
                }
                tu=0;
                xa=0;
                ya=30; yb=90; yc=150;
}
```