(2)

June 12, 1992

Full Surface Testing of Grazing Incidence Mirrors

Principal Investigator: John L. Remo Ph.D.
E.R.G. Systems Inc.
St. James, New York 11780
(516) 584-5540

Seventh Quarter Report: Introductory Outline

The project goals for the first seven quarters have been achieved. These include:

1. Prototype design,
2. Procurement and testing of components,
3. Mathematics and algorithm development,
4. Interferogram and data reduction algorithms,
5. Construction of first prototype,
6. Automated operation software development,
7. Construction and evaluation of modified prototype,
8. Integration of automated hardware controls and software, and
9. Testing of the full surface interferometer on aspheric optical surfaces.
10. Initiation of user manual and documentation
11. Additional testing and modification
12. Software refinement and development

The eighth (final) quarter goals are:

1. Complete the writing of a users manual and documentation,
2. Additional testing and modification, and
3. Preparartion of the final report (eighth or final quarter).

Research is supported by SDIO/IST and managed by ONR
Contract # N00014-90-C-0246.

FULL-SURFACE INTERFEROMETRIC TESTING OF GRAZING INCIDENCE MIRRORS
SEVENTH QUARTERLY REPORT: 03/12/92 to 06/11/92

SUMMARY

In the present report, we present the work done during the period of March 12, 1992 to June 11, 1992.

The tasks include work on the preparation of the user manual and software documentation, as well as additional testing and modifications.

This report includes a detailed presentation on the following:

- The user manual describing the operation of the interferometer system. The system operation is menu driven using the function keys for the various steps
- A software package overview describing the various subprograms included in the package
- The program source code, including the main menu program, the motion control programs and the data processing programs
- The specification sheets of the motion control hardware, including microstepper and linear scanning stage.

During the next period we will work on the following tasks:

- Complete the user manual and documentation
- Complete the testing program and implement the final modifications
- Prepare the final report.

This report is divided into four parts:

1. The initial version of the user manual

2. Software package overview

3. Program source code which is over forty pages long and still undergoing refinement. Therefore we have only included the first two and last two pages in order to keep the size of his report managable. A complete version of the source code will be included in the final report. However, we can at any time provide, upon request, an up-to-date version of the software.

4. Specification sheets on the motion control system.

CONTENTS

1.0 User Manual: Operation of the Interferometer System

The cylindrical mirror testing package user manual includes the hardware motion controllers specification sheets with sample programs to test the motion control and system operation menu driven software package with the programs.

The motion control specifications list the model and maker of the controllers and motors, their electrical and mechanical specifications, wire connections of the communication ports and control software command lists.

The system operation package is menu driven type of functions. When type "MAINF" in DOS system, the display will show a message and the MAIN MENU on the screen. By pressing the Function keys for the desired operation, the user can manipulate the interferometer system and implement some specific control and data processing for the test mirrors. There are three operations in the main menu and are implemented by pressing function keys F1, F2 or F3, pressing the "ESC" key will exit the program and back to DOS system.

The MAIN MENU is shown in the following:

```
-    FSIS SYSTEM    -


F1 : TEST MIRROR SCANNING
F2 : SINGLE APERTURE MEASUREMENT
F3 : FULL SURFACE MEASUREMENT
ESC: EXIT
```

The functions of each operation are explained in the following:

F1. TEST MIRROR SCANNING

The TEST MIRROR SCANNING is to operate the interferometer system in the scanning mode, which allows the user to align the cylindrical test mirror with the interferometer and scan the full surface of the test mirror to view the complete interferogram.

When Function key F1 is pressed, the linear translation stage (NEAT) will be activated to move the test mirror along X-direction (cylindrical mirror axis) back and forth at a

certain speed to allow the user to view and examine the full interferogram of the test mirror. This processing can be repeated by pressing the F1 as needed.

F2. SINGLE APERTURE MEASUREMENT

The SINGLE APERTURE MEASUREMENT is to operate the interferometer system to acquire a desired aperture of the cylindrical test mirror interferogram, and compute the phase of the interferogram and display it in 3-D wire frame isometric plot or in thin line color contour plot.

F3. FULL SURFACE MEASUREMENT

The FULL SURFACE MEASUREMENT is to operate the interferometer system to scan the cylindrical test mirror in a multi-aperture mode. For the first test mirror, it is assigned five subaperture to cover the full mirror.

It takes the first subaperture interferograms and stores in the frame buffer then move the test mirror to the 2nd position, then takes the 2nd subaperture interferograms and stores in the frame buffer. The same procedure repeats until all five subaperture interferograms are acquired and stored in the frame buffer.

After all the subaperture interferograms being acquired. The program then move the SUB-MENU page for further data processing. It includes the following functions:

- FSIS SYSTEM -

F1 : 1ST APERTURE
F2 : 2ND APERTURE
F3 : 3RD APERTURE
F4 : 4TH APERTURE
F5 : 5TH APERTURE
F6 : COMPUTE ALL APERTURES
F7 : MERGE THE FULL SURFACE
F8 : REMOVE TILT PLANE OF THE FULL SURFACE PHASE MAP
ESC: MAIN MENU

The functions of the sub-menu are explained in the following:

F1 : 1ST APERTURE

When Function key F1 is pressed, the program will start to load the stored images of FIRST subaperture interferograms into frame buffer. Compute the phase map and display it in 3-D isometric wire frame plot and thin line contour plot, then store in the hard disk. The hard copy of the plot can be printed out in Laser Printer or other type of printer by pressing the keys "SHIFT" and "PRINT SCREEN" at the same time.

F2 : 2ND APERTURE

When Function key F2 is pressed, the program will start to load the stored images of SECOND subaperture interferograms into frame buffer. Compute the phase map and display it in 3-D isometric wire frame plot and thin line contour plot, then store in the hard disk. The hard copy of the plot can be printed out in Laser Printer or other type of printer by pressing the keys "SHIFT" and "PRINT SCREEN" at the same time.

F3 : 3RD APERTURE

When Function key F3 is pressed, the program will start to load the stored images of THIRD subaperture interferograms into frame buffer. Compute the phase map and display it in 3-D isometric wire frame plot and thin line contour plot and store the phase in the hard disk. The hard copy of the plot can be printed out in Laser Printer or other type of printer by pressing the keys "SHIFT" and "PRINT SCREEN" at the same time.

F4 : 4TH APERTURE

When Function key F4 is pressed, the program will start to load the stored images of FORTH subaperture interferograms into frame buffer. Compute the phase map and display it in 3-D isometric wire frame plot and thin line contour plot and store the phase in the hard disk. The hard copy of the plot can be printed out in Laser Printer or other type of printer by pressing the keys "SHIFT" and "PRINT SCREEN" at the same time.

F5 : 5TH APERTURE

When Function key F5 is pressed, the program will start to load the stored images of FIFTH subaperture interferograms into frame buffer. Compute the phase map and display it in 3-D phase map wire frame plot and thin line contour plot and stored the phase in the hard disk. The hard copy of the plot can be printed out in Laser Printer or other type of printer by pressing the keys "SHIFT" and "PRINT SCREEN" at the same time.

F6 : COMPUTE ALL APERTURES

When Function key F6 is pressed, the program will compute all the five subapertures interferograms start from first subaperture to fifth subaperture. First it load the stored images of 1st subaperture interferograms into frame buffer, then compute the phase and display it in 3-D wire frame plot and thin line contour plot and store the phase map in the hard disk. Then load the stored images of the 2nd subaperture interferograms into frame buffer, compute the phase and display it in 3-D wire frame plot and thin line contour plot and store in the hard disk. Repeat the same processing and display until all the five subapertures are computed and display and stored in the hard disk.

F7 : MERGE THE FULL SURFACE

When Function key F7 is pressed, the program will load all five subapertures phase maps from the hard disk and match the phase between the overlap areas between each subaperture with best fit algorithm. It does the following operations:

a. Load the phase maps of subaperture #3 and #2

b. Define the overlap area

c. Compute the phase difference between of subaperture #3 and #2 within the overlap area

d. Linear best fit a plane on the phase difference which is computed in step "c"

f. Adjust the phase of subaperture #2 by the linear best fit plane function which is computed in step "f"

g. Store the new phase map of subaperture #2 into the disk

h. Load the phase of subaperture #1 and new phase map of subaperture #2 which is computed in step "g"

i. Define the overlap area

j. Compute the phase difference between subaperture #2 and #1 within the overlap area

k. Linear best fit a plane on the phase difference which is computed in step "j"

l. Adjust the phase of subaperture #1 by the linear best fit

plane function which is computed in step "k"

m. Store the new phase map of subaperture #1 in the hard disk

n. Load the phase maps of subaperture #3 and #4

o. Define the overlap area

p. Compute the phase difference between subaperture #3 and #4 within the overlap area

q. Linear best fit a plane on the phase difference which is computed in step "p"

r. Adjust the phase of subaperture #4 by the linear best fit plane function which is computed in step "q"

s. Store the new phase map of subaperture #4 in the hard disk

t. Load the phase of subaperture #5 and new phase map of subaperture #4 which is computed in step "s"

u. Define the overlap area

v. Compute the phase difference between subaperture #4 and #5 inside the overlap area

w. Linear best fit a plane on the phase difference which is computed in step "v"

x. Adjust the phase of subaperture #5 by the linear best fit plane function

y. Store the new phase map of subaperture #5

z. Rearrange the new phase maps of subapertures #1 to #5 into a regular 120 x 120 full surface phase array and store it in hard disk

F8 : REMOVE TILT PLANE OF THE FULL SURFACE PHASE MAP

When the Function key F8 is pressed, the program will do the following operations:

a. Load the new full surface phase map which is computed in step "z" in Function F7

b. Fit the best fit plane on the phase map

c. Remove the tilt plane by subtracting the linear best fit function which is computed in step "b"

d. Display the tilt removed full surface phase map in 3-D wire frame plot and thin line contour plot.

f. Store the tilt removed phase map

ESC: MAIN MENU

When "ESC" key is pressed, the program will return back to MAIN MENU page.

2.0 Software Package Overview: Description of the Sub-Programs

The cylindrical mirror testing software package includes a main program file and some subprogram files. Basically the subprogram files are grouped in three groups: the translation stage motion control subprograms, the phase measuring subprograms and the full surface phase matching processing subprograms.

In the following, the lists of the program file names and their functions are described:

1. MAINF

This is the main program of the package, which includes two menus: MAIN MENU and SUB MENU.

There are three functions in the main menu:

1. TEST MIRROR SCANNING
2. SINGLE APERTURE MEASUREMENT
3. FULL SURFACE MEASUREMENT

In the third function "FULL SURFACE MEASUREMENT", there are sub menu to do the further processing of the multi-aperture interferograms. Eight functions are under this sub menu:

1. FIRST APERTURE
2. SECOND APERTURE
3. THIRD APERTURE
4. FORTH APERTURE
5. FIFTH APERTURE
6. COMPUTE ALL APERTURES
7. MERGE FULL SURFACE

8. REMOVE TILT PLANE OF THE FULL SURFACE PHASE MAP

2. **NEATREM1**

Motion control subprogram, called by Function 1 in Main Menu. It controls the test mirror stage moving forward 3 inches and backward to its home position

3. **NEATREM2**

Motion control subprogram, called by Function 3 in Main Menu. It controls the test mirror stage moving forward 1/2 aperture

4. **NEATREM3**

Motion control subprogram, called by Function 3 in Main Menu. It controls the test mirror stage moving backward 2 apertures

5. **C7**

Single Aperture Phase measuring subprogram, called by Function 2 in Main Menu. It acquires the interferograms, computes the phase and displays it in 3-D and contour plots

6. **C10 - C14**

Multi-Aperture Phase measuring subprograms, called by Function 3 in Main Menu, and Functions 1 to 5 in Sub Menu. It loads the interferograms of the subaperture into the frame buffer, computes the phase and displays it in 3-D and contour plots and stores it in the disk.

7. **C20 - C24**

Multi-Aperture Phase measuring subprograms, called by Function 3 in Main Menu, and Function 6 in Sub Menu. It loads the interferograms of five subapertures in sequence into the frame buffer, computes the phase and displays it in 3-D and contour plots and stores it in the disk.

8. **CC**

Multi-Aperture Phase measuring subprograms, called by Function 3 in Main Menu, and Function 7 in Sub Menu. It loads all the phase files of the five subapertures, matches the phase

between each subaperture by linear best fit algorithm of the phase difference within the overlap zone between the adjacent subapertures. It adjusts and merges five subapertures into a full surface file, display the full surface phase map and store it in disk

9. CRM

Multi-Aperture Phase measuring subprograms, called by Function 3 in Main Menu, and Function 8 in Sub Menu. It loads the full surface phase map file, fits the best plane, removes the best fit (tilt) plane, plots it and stores it in disk.

3.0 Software Package: Program Source Code

3.1 Main program

```
/*    MAIN PROGRAM: MAINF */


#include<stdio.h>
#include<stdlib.h>
#include<process.h>
#include<graph.h>
#include<signal.h>
#include<dos.h>
#include "svobj.h"
#include "pxobj.h"
#include "pxip8.h"
#include "pxmp.h"
int comswitch = 0 , merge = 0 ;
char message[78] ;
main(msgn,msgc)
int msgn ;
char *msgc[] ;
{ int i,j,ch;
  opensi(240,3,0,240,0,0) ;
  openoriel() ;
   _setvideomode(_ERESCOLOR) ;
  do { draw_para() ;
       while (!kbhit()) ; while (kbhit()) ch= getch() ;
       if (ch == 27)
       break ;
       if (ch > 58 && ch < 62)
       dographic(ch) ;
       } while (ch != 27) ;
  pxd_video('s',0L) ;
```

```
  pxd_close();
  _setvideomode(_TEXTC80) ;
}

callbasic(pgm)
char *pgm ;
{
  _setvideomode(_TEXTC80) ;
  system(pgm) ;
  _setvideomode(_ERESCOLOR) ;
}
draw_para()
{  char buf[30] ;
   int fkeyp = 25 ;
   mesg() ;
   d_text(3,18,"WELCOME TO CYLINDRICAL LENS MEASURING SYSTEM",15)
;
   d_text(7,fkeyp,"F1 : TEST MIRROR SCANING",15) ;
   d_text(9,fkeyp,"F2 : SINGLE APERTURE MEASUREMENT",15) ;
   d_text(11,fkeyp,"F3 : FULL SURFACE MEASUREMENT",15) ;
   d_text(15,fkeyp,"ESC: EXIT",15) ;
}
draw_para1()
{  char buf[30] ;
   int fkeyp = 25 ;
   mesg() ;
   d_text(3,24,"CYLINDRICAL LENS MEASURING SYSTEM",15) ;
   d_text(6,fkeyp,"F1 : 1ST APERTURE ",15) ;
   d_text(8,fkeyp,"F2 : 2ND APERTURE ",15) ;
   d_text(10,fkeyp,"F3 : 3RD APERTURE ",15) ;
   d_text(12,fkeyp,"F4 : 4TH APERTURE ",15) ;
   d_text(14,fkeyp,"F5 : 5TH APERTURE ",15) ;
   d_text(16,fkeyp,"F6 : COMPUTE ALL APERTURES",15) ;
   d_text(18,fkeyp,"F7 : MERGE THE FULL SURFACE",15) ;
   d_text(20,fkeyp,"F8 : REMOVE TILT PLANE OF FULL SURFACE PHASE
MAP",15) ;
   d_text(23,fkeyp,"ESC: MAIN MENU",15) ;
}
d_text(rr,cc,buf,color)
int rr,cc ;
char *buf ;
int color ;
{ _settextcolor(color);
  _settextposition(rr,cc);
  _outtext(buf);
}
dographic(ch)
int ch ;
{ char msg[100] ;
  switch(ch) {
    case 59 :
            callbasic("neatrem1") ;
```

```
            break ;
    case 60 :
            takeone() ;
            break ;
    case 61 :
            takemul() ;
            _setvideomode(_ERESCOLOR) ;
            merge = comswitch = 0 ;
            do { draw_para1() ;
              while (!kbhit()) ; while (kbhit()) ch= getch() ;
              sprintf(message,"                                    \0") ;
              if (ch == 27)
                 { _setvideomode(_ERESCOLOR) ;
                 break ;
                 }
              if (ch > 58 && ch < 67)
                 dographic1(ch) ;
              } while (ch != 27) ;
            _setvideomode(_ERESCOLOR) ;
            break ;
    default  : break ;
    }
}
dographic1(ch)
int ch ;
{ char msg[100] ;
  int i ;
  switch(ch) {
    case 59 :
            compute(0) ;
            break ;
    case 60 :
            compute(1) ;
            break ;
    case 61 :
            compute(2) ;
            break ;
    case 62 :
            compute(3) ;
            break ;
    case 63 :
            compute(4) ;
            break ;
    case 64 :
            for (i = 0 ; i < 5 ; i++)
             compute1(i) ;
            break ;
    case 65 :
            if (comswitch != 31)
            sprintf(message,"DATA  FILES  HAVE  NOT  COMPUTED  YET
!!!\0") ;
          else
```

Pages 11 through 45 contain SOURCE CODE and have deliberately been omitted: A complete listing of the FINAL SOURCE CODE will be included in the FINAL REPORT

```
      goto drawit
case4:
      x1=xh(m1):y1=yh(m1)
      x2=(h(m3)*xh(m2)-h(m2)*xh(m3))/(h(m3)-h(m2))
      y2=(h(m3)*yh(m2)-h(m2)*yh(m3))/(h(m3)-h(m2))
      savx = x2 : savy = y2
      goto drawit
case5:
      x1=xh(m2):y1=yh(m2)
      x2=(h(m1)*xh(m3)-h(m3)*xh(m1))/(h(m1)-h(m3))
      y2=(h(m1)*yh(m3)-h(m3)*yh(m1))/(h(m1)-h(m3))
      goto drawit
case6:
      x1=xh(m3):y1=yh(m3)
      if m > 1 then
      x2 = savx : y2 = savy
        else
      x2=(h(m2)*xh(m1)-h(m1)*xh(m2))/(h(m2)-h(m1))
      y2=(h(m2)*yh(m1)-h(m1)*yh(m2))/(h(m2)-h(m1))
      end if
      goto drawit
case7:
      if m > 1 then
      x1 = savx : y1 = savy
        else
      x1=(h(m2)*xh(m1)-h(m1)*xh(m2))/(h(m2)-h(m1))
      y1=(h(m2)*yh(m1)-h(m1)*yh(m2))/(h(m2)-h(m1))
      end if
      x2=(h(m3)*xh(m2)-h(m2)*xh(m3))/(h(m3)-h(m2))
      y2=(h(m3)*yh(m2)-h(m2)*yh(m3))/(h(m3)-h(m2))
      savx = x2 : savy = y2
      goto drawit
case8:
      x1=(h(m3)*xh(m2)-h(m2)*xh(m3))/(h(m3)-h(m2))
      y1=(h(m3)*yh(m2)-h(m2)*yh(m3))/(h(m3)-h(m2))
      x2=(h(m1)*xh(m3)-h(m3)*xh(m1))/(h(m1)-h(m3))
      y2=(h(m1)*yh(m3)-h(m3)*yh(m1))/(h(m1)-h(m3))
      savx = x1 : savy = y1
      goto drawit
case9:
      x1=(h(m1)*xh(m3)-h(m3)*xh(m1))/(h(m1)-h(m3))
      y1=(h(m1)*yh(m3)-h(m3)*yh(m1))/(h(m1)-h(m3))
      if m > 1 then
      x2 = savx : y2 = savy
        else
      x2=(h(m2)*xh(m1)-h(m1)*xh(m2))/(h(m2)-h(m1))
      y2=(h(m2)*yh(m1)-h(m1)*yh(m2))/(h(m2)-h(m1))
      end if
 drawit:
   x1 = int(x1)
   x2 = int(x2)
   y1 = int(y1)
```

```
            y2 = int(y2)
              line (x1,y1)-(x2,y2),colors(k)
         case0: next m

noneintri: next k
noneinbox: next i
noneinbox1:  next j
10202 end sub


sub box static
line(0,0)-(639,349),,b
line(0,14)-(639,14)
line(3,16)-(636,346),,b
end sub


sub dodata static
open "t.dat" for input as #1

input #1,top%,bot%,max,min
for i% = top% to bot%
    input #1,bd%(1,i%),bd%(2,i%)
    for j% = bd%(1,i%) to bd%(2,i%)
     input #1,zz%(j%,i%)
     next j%
    next i%
close
call tiltterm
call removetilt
end sub
sub removetilt static
xf = 119
xe = 0
    max = -32767
    min = 32767
    for i% = top% to bot%
     if xf > bd%(1,i%) then xf = bd%(1,i%)
     if xe < bd%(2,i%) then xe = bd%(2,i%)
     for j% = bd%(1,i%) to bd%(2,i%)
         zz%(j%,i%)= zz%(j%,i%)-(coe1+coe2*j%+coe3*i%)
         if max < zz%(j%,i%) then max = zz%(j%,i%)
         if min > zz%(j%,i%) then min = zz%(j%,i%)
     next j%
    next i%
yf = top% + 2
ye = bot% - 2
xf = xf + 3
xe = xe - 3

end sub
```

```
sub tiltterm static
     n = 0
     x = 0
     y = 0
     xy = 0
     xx = 0
     yy = 0
     w = 0
     wx = 0
     wy = 0
'    print top%,bot%
    for i% = top% to bot%
     for j% = bd%(1,i%) to bd%(2,i%)
          iii = i%
          jjj = j%
          zzz = zz%(j%,i%)
          n=n+1
          x=x+jjj
          y=y+iii
          xx=xx+jjj^2
          yy=yy+iii^2
          xy=xy+iii*jjj
          w=w+zzz
          wx=wx+zzz*jjj
          wy=wy+zzz*iii
          next j%
'         print n;i%;bd%(1,i%);bd%(2,i%),w;wx;wy
     next i%

     det=N*xx*yy+x*xy*y+x*y*xy-xx*y*y-N*xy*xy-x*x*yy
     coe1=(w*xx*yy+wx*xy*y+x*xy*wy-xx*y*wy-w*xy*xy-wx*yy*x)/det
     coe2=(N*wx*yy+x*y*wy+w*xy*y-wx*y*y-N*wy*xy-w*x*yy)/det
     coe3=(N*xx*wy+x*w*xy+x*y*wx-y*w*xx-N*xy*wx-x*x*wy)/det
end sub
```

4.0 Specification Sheets: Motion Control Systems

4.1 ORIEL STEPPER MIKE MOTOR AND CONTROLLER SPECIFICATION

A. Motor Controller

ORIEL RS 232 Stepper Interface Controller.

Specification:

Number of controlled stepper motors:	*Up to two, individually.*
Data I/O :	*300,600,1200,2400,4800,9600 baud, switch selectable.*
Compatible stepper motor:	*2 or 4 phase unipolar motor, 24V, 0.5 A max, per phase.*

Winding holding current:	*Reduced to 0.1 A per phase.*
Limit switch connections:	*Low logic level = limit reached.*
Maximum speed:	*Half step mode: 1000 steps / sec.*
Full step mode:	*500 steps / sec.*
Computer interface:	*RS 232.*
Power requirements:	*110/220-240 VAC 50/60 Hz, switch selectable.*

B. Motor

ORIEL Stepper Mike. Model 18500.

Specification:

Step Size:	*Half step mode: 1.3, 0.7 um.*
Full step mode:	*2 um.*
Maximum step rate:	*Half step mode: 1000 steps / sec.*
Full step mode:	*500 steps / sec.*
Maximum spindle speed:	*1 mm /sec.*
Maximum axial load:	*15.5 lbs.*
Uni-directional repeatability:	*< 2 um.*
Backlash:	*< 3 um.*
Travel Range:	*0.5", (13 mm).*

C. Wiring Specification of the Communication Port

Computer: RS232 9 pin (F) adapter.
- *2: (Red)*
- *3: (Green)*
- *5: (Black)*
- *6,7,8: Short circuit.*

Controller: 25 pin (M) adapter.
- *2: (Red)*
- *3: (Green)*
- *1,7: (Black)*

Controller to Motor : 10 lines Rainbow cable.

D. Motor Control Software Command Sets

RS 232 serial control, 7 data bit, 1 stop bit, no parity bit, ASCII data format.
Command Set:

A,B: Direct the flow of command, data, status inquiries to motor control register.
C: Clear absolute register to zero, (set logic origin).
D: Disable motor driver.
E: Enable the motor driver.
F: Change to full step mode.
G: Go absolute.
Format G +/- dddd, ',' is necessary to stop data field.
dddd is the number of steps to go.
H: Change to half step mode.
I: Inquire the status of motor. return 'd', d = 0 - 7
bit 0, direction: CW = 1 , CCW = 0.

bit 1, step size: half = 1 , full = 0.
bit 2, E/D status: Enable = 1, disable = 0.

Q: Inquire the status of translator. return 'd', d = 0 - 7
bit 0, motor: On = 1 , Off = 0.
bit 1, CW limit: yes = 1 , no = 0.
bit 2, CCW limit: yes = 1 , no = 0.

R: Set up the desired step rate for each motor.
Format: Rn, n = 0 - 7.

S: To start the moving.

T: Travel relative steps.
Format: Tddddddddd,

Z: Move to the logic origin. (the position of absolute register = 0)

@: Unconditional stop both motor.

>: Move single step in CW direction.

<: Move single step in CCW direction.

=: Start both motors simultaneously.
** After command set, eg AT1000,BT1000.*

+: Set direction to CW.

-: Set direction to CCW.

4.2 NEAT MOTOR AND CONTROLLER SPECIFICATION

A. Motor Controller

NEAT-310M Programmable Stepping Motor Controller.

Specification:

Drive Type:	*54 volt bipolar chopper*
	.7 to 3.5 Amps/Phase, half-coil
	1.5 to 7 Amps/Phase, full-coil.
Memory buffer:	*8 or 32 K bytes non-volatile program memory*
Velocity Range:	*40 to 327,640 steps / sec.*
	40 Hz resolution.
Position Range:	*-8,388,608 to +8,388,608 steps*
Computer interface:	*RS 232. / 8 bit parallel communication port*
Power requirements:	*115/230 V A.C. 50/60 Hz, 1.5 Amps*

B. Control Software Command Sets.

RS 232 serial control, 8 data bit, 1 stop bit, no parity bit, ASCII data format.
Command Set:

MR: Move relative.
Format: MRddddddddd ddddddddd range from -8,388,608 to + 8,388,608

MA: Move absolute.
Format: MAddddddddd ddddddddd range from -8,388,608 to + 8,388,608

MH: Move physical origin position.

MC: Move continuous.

Format MC+/-.

SP: Set absolute register counter value.

VI: Set initial velocity.

Format: VIddddddddd ddddddddd range from 40 to 327,680

default is 4000 steps / sec.

The primary resonance is at 400-1200 steps / sec. the maximum allowable start/stop frequency is load dependent, typically below 6000 / sec is safe.

VF: Set final velocity.

Format: VFddddddddd ddddddddd range from 40 to 327,680

AC: Set acceleration and deceleration.

Format: AC(+/-)ddddddddd , '+' for acceleration, '-' for deceleration, without sign for both.

ST: Stop moving immediately.

ME: Mnemonic Expansion.

Format: MFE for enable, MED for disable.

MF: Move finish signal feed back.

Format: MFE for enable, MFD for disable.

C. Wiring Specification of the Communication Ports

Computer: RS232 9 pin (F) adapter.

2: (Red)

3: (Green)

5: (Black)

6,7,8: Short circuit.

Controller: 25 pin (M) adapter.

2: (Green)

3: (Red

1,7: (Black)