Giant Metrewave Radio Telescope

NCRA • TIFR

National Centre for Radio Astrophysics

TATA INSTITUTE OF FUNDAMENTAL RESEARCH

Technical Report

# "Testing of Brushless motor, drive in C04 Antenna with Programmable Multi Axis Controller Configured as Position Loop and Velocity Loop"

By
Srinivasarao Beera
B. Amit Kumar
Shailender Bachal
GMRT-TIFR
Pune, India

Under the guidance of

Dr. Bal Chandra Joshi
Mr. Suresh Sabhapathy
Servo Group
GMRT-NCRA-TIFR
Pune, India

# Brushless Motor Testing at C04

Contents

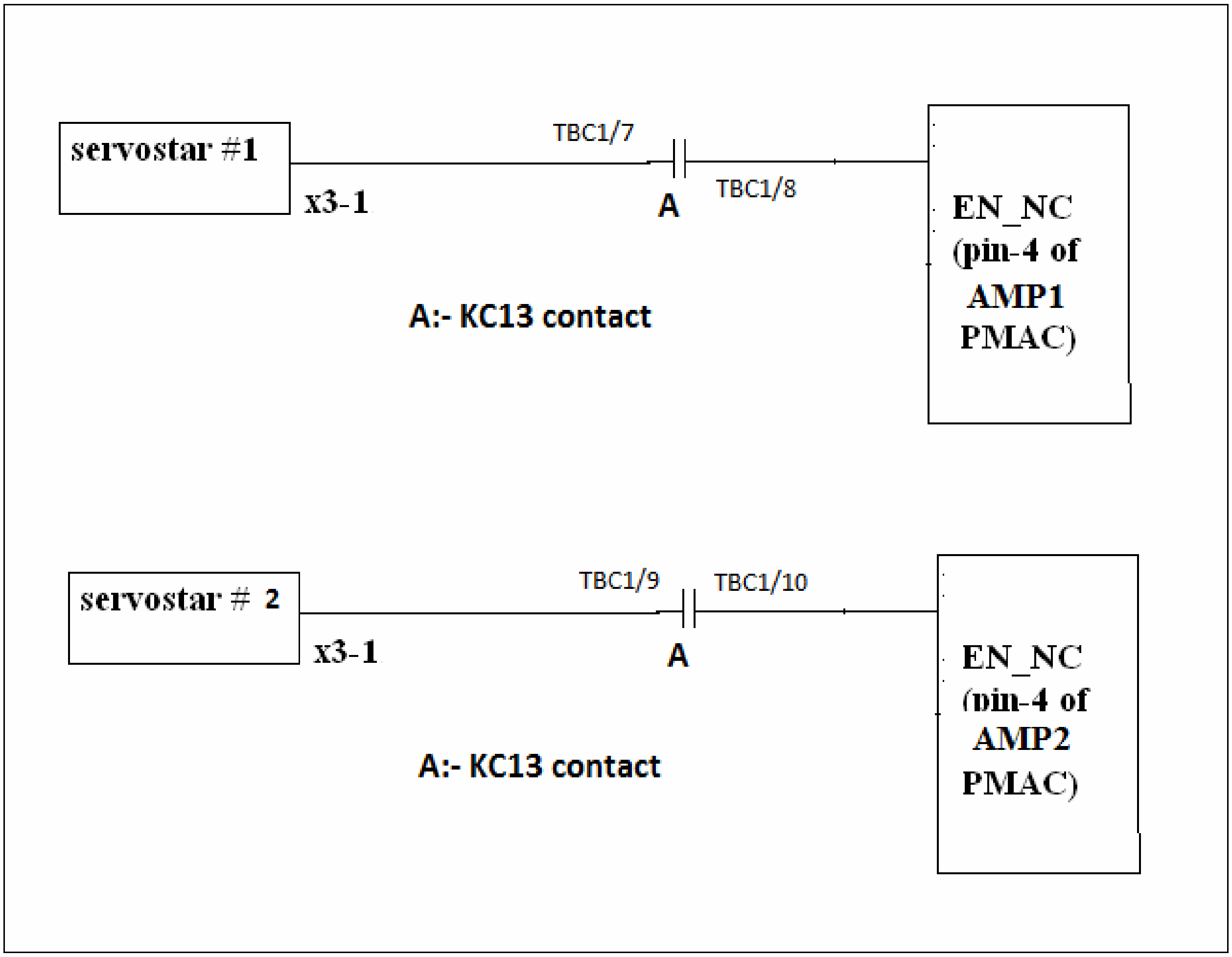
# 1. Safety and interlocks while testing

1.1) In existing antenna system we have safety interlock system which have connected to Amplifier overload (AOLD), Motor Overload (MOLD), +270deg limit switch, -270deg limit switch, Cwrap, 0deg limit switch so on in AZ axis and Amplifier overload (AOLD), Motor Overload (MOLD), 110 deg limit switch, 15 deg limit switch and STD (90 deg) limit switch so on in EL axis for safety of Antenna.

1.2) While brushless motor testing in C04 , following was done:

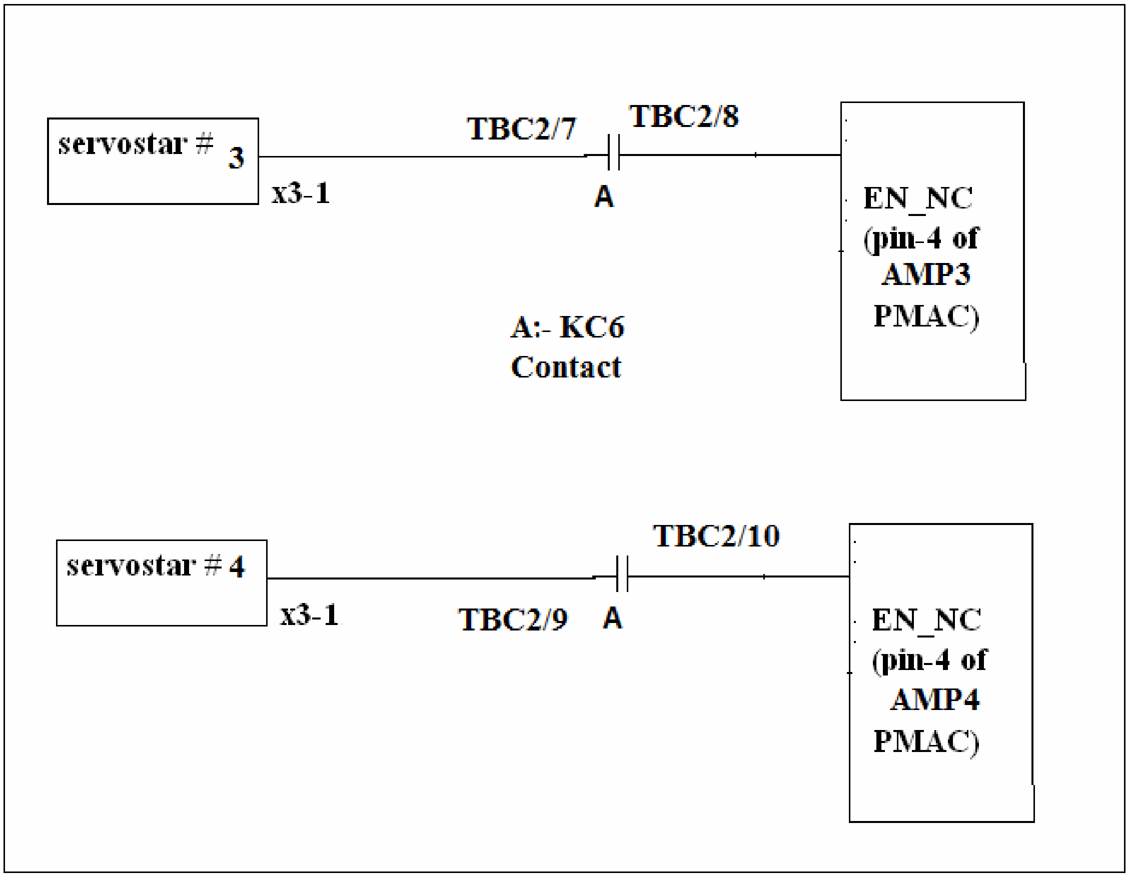
**Azimuth Axis:-**

a) KC13 contacts in control rack i.e. TBC1/7, TBC1/8 was given to SS712-1 Enable pin (pin no:-X3/1 of SS712), enable pin of PMAC (pin no:-Amp1/4) respectively. Similarly KC13 contacts in control rack i.e. TBC1/9, TBC1/10 was given to SS712-2 Enable pin (pin no:-X3/1 of SS712), enable pin of PMAC (pin no:-Amp2/4) respectively as given in below diagram.



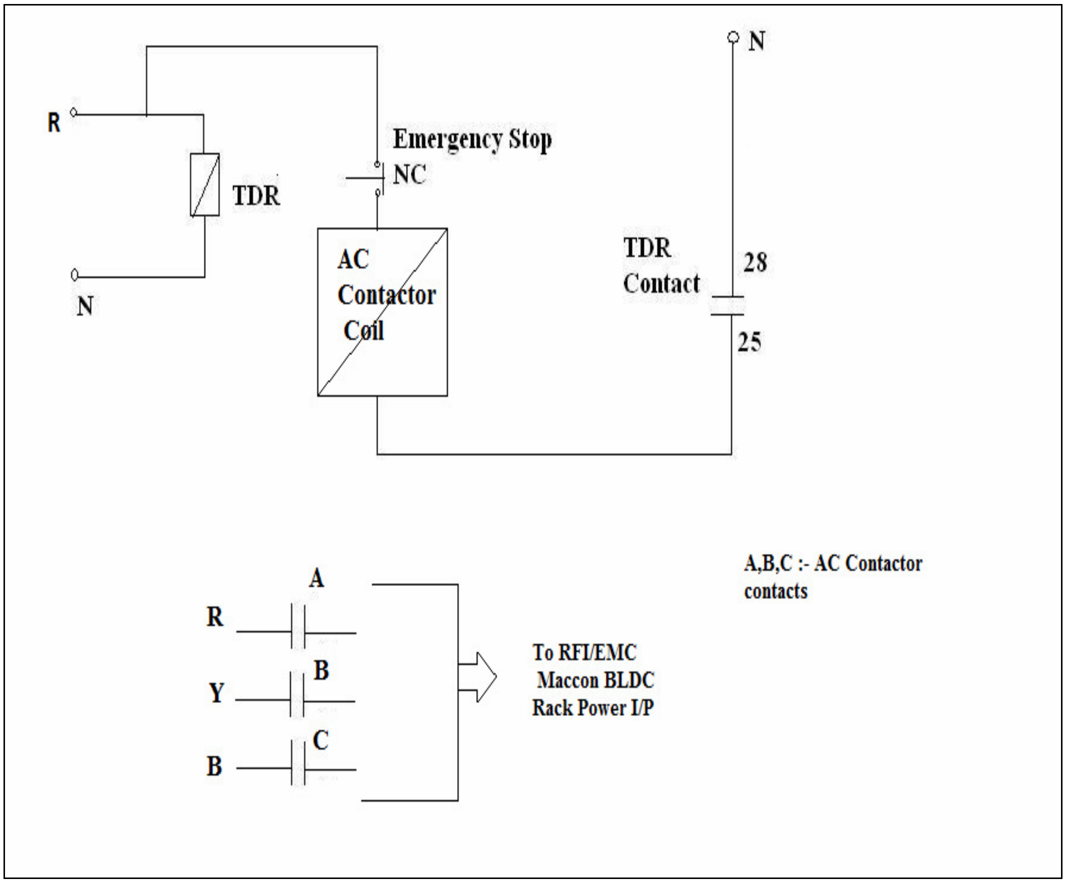
**Elevation Axis:-**

b) KC6 contacts in control rack i.e. TBC2/7, TBC2/8 was given to SS712-3 Enable pin (pin no:-X3/1 of SS712), enable pin of PMAC (pin no:-Amp3/4) respectively. Similarly KC6 contacts in control rack i.e. TBC2/9, TBC2/10 was given to SS712-4 Enable pin (pin no:-X3/1 of SS712), enable pin of PMAC (pin no:-Amp4/4) respectively as given in below diagram.



c) Any interlock failure will disable the Servostar (SS712).

d) We have taken extra Emergency switch in case of any emergency will stop total power to Brushless Rack as per diagram below.



Note:-

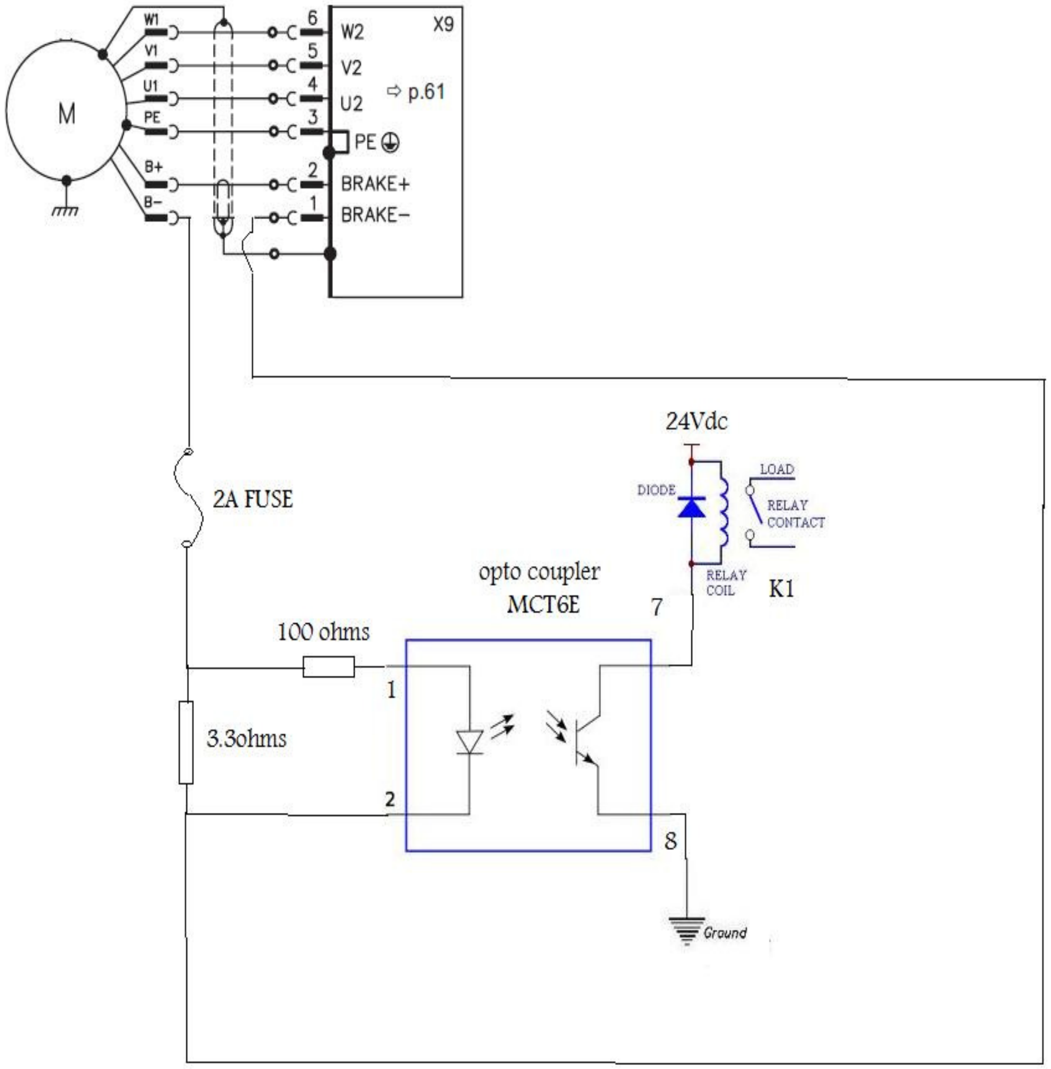
1. Checked Servostar disabling by pressing limit switches.
2. Checked total depower the system by pressing Emergency Stop.
3. Motor Overload contacts are short on connector side.
4. Amplifier Overload contacts are not used (always closed).

1.3) **Existing Brake card changes**: - short pin no. 17, 23 for EL and short 10, 12 for AZ for interlocks (i.e. brake release).

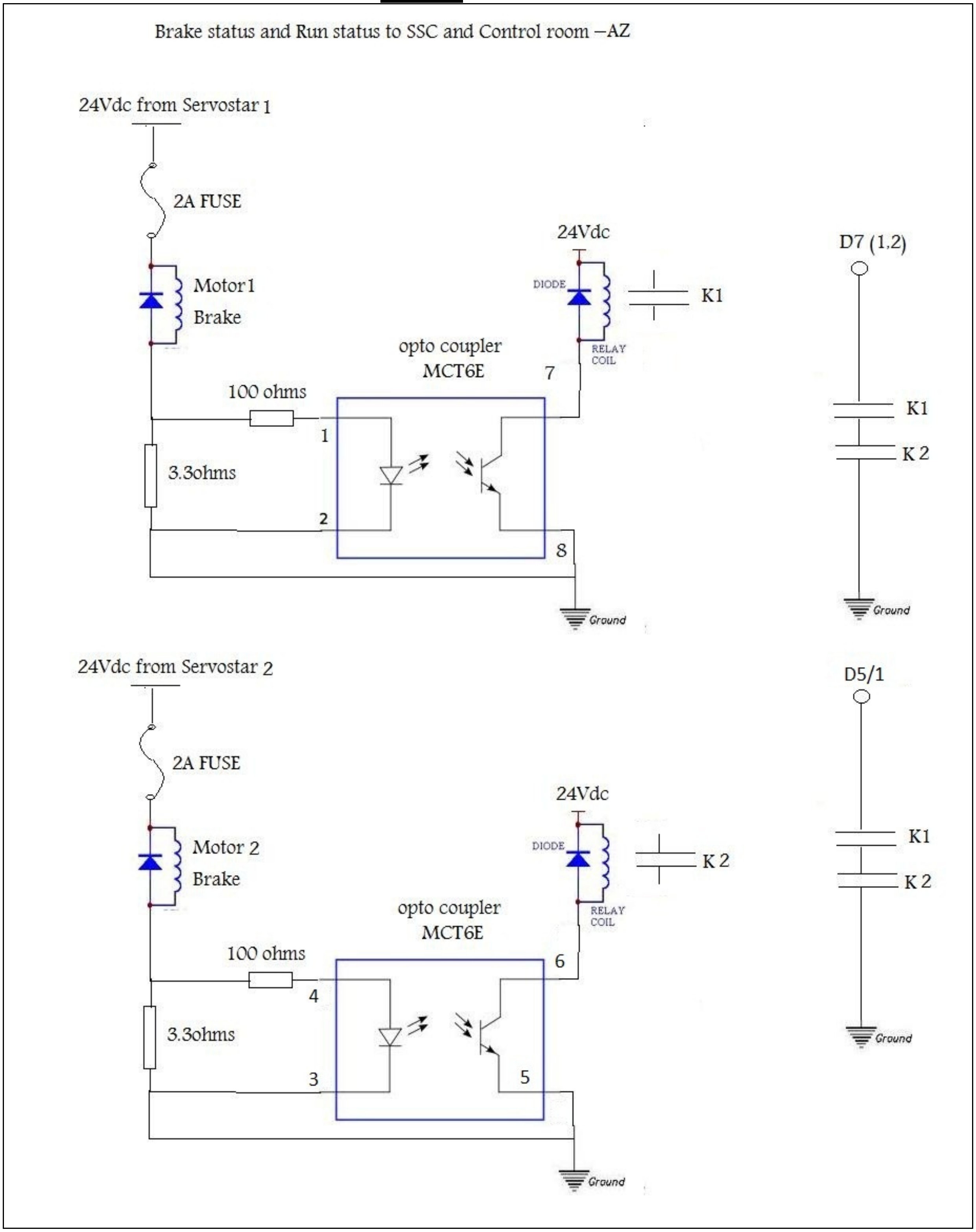
1.4) **AZ, EL Brake status, AZ, EL RUN flag to Control room and SSC**: - To provide Brake, RUN flag status to control room and SSC, a PCB was made. Schematic and wiring diagram is as below. Current through brake coil of BLDC is sensed by a resistor and OPTO-Coupler circuit which will energize a Relay. The relay contact is used for Brake release status and RUN flag of AZ and EL axes.

## Wiring Diagram for monitoring BL motor Brakes:-

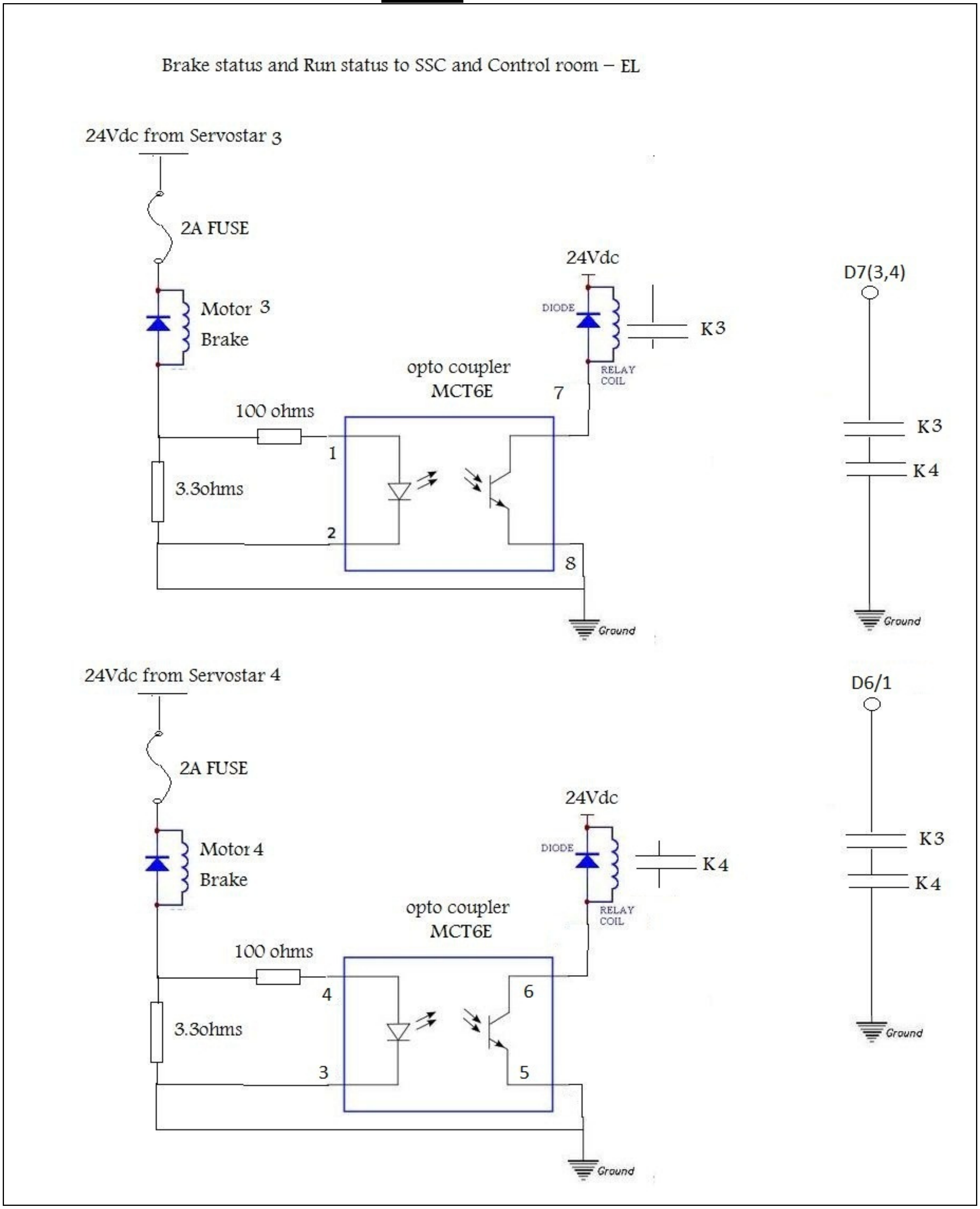
Brake Sensing wiring diagram



## Schematic Diagram for Brake status and Run Flag to SSC and Control room AZ axis



## Schematic Diagram for Brake status and Run Flag to SSC and Control room EL axis



## 2. Rack Switch on sequence:

1) Switch on BLDC power in the MSEB panel outside the Shielded cage in antenna base.

2) Switch ON 32 Amp MCB – 3 ph in Rack for incoming Main supply from Line filter.

3) Switch on 32 Amp MCB for powering on the 24 volt DC power supply,
   Status in SS712 is 12.

4) Switch on 3-ph 32 amp MCB for AZ & El axes to SS712- #1 and #2 & SS712- #3 and #4, Status in SS712 is P12.

5) Now switch the AC power to PC / Laptop after connecting RS232 port to SS712 - #1,#2 #3,#4(individually) and USB port to PMAC controller MAC2-BLC-2 (this will enable comm. to PMAC from PC).

6) Switch on 63 Amp MCB to Servo Rack (old) and switch on the Key switch of the Control rack.

7) Put the **mode** switch in **Manual** and press **AZ ON** or **EL ON** button for enabling interlock to drive. This step to be done only after PMAC is configured and ready for rotation checks.

## 3. ADC linearity check of MAC2-BLC-2 (useful when using velocity loop).

3.1) +/- 10 Analog voltage has connected through D type female connector to X11 (ANA CH #7) or/and X12(ANA CH #8) of BLC.

Pin2 of connector will go to +10V.
Pin6 of connector will go to -10V.

3.2) Go to PEWIN32 Pro and Download file "bsr_adc_blc_19032010.pmc" which has PLC30 program as given below.

```
i5=3                  ; PLC program ON for enabling in Terminal
                      ; Window
i7106=$1FFFFF   ; ADC strobe word default value for A/D
                      ; Conversion.
M5063->Y: $78115, 8, 16, S; ch7 A-D channel (X11)
M5064->Y: $7811D, 8, 16, S; ch8 A-D channel (X12)
            ; Memory location that store the 16 bit digital data proportional to
            ; Analog input in signed form, not in unsigned form. Signed form
            ; will give digital value from -32,678 to +32,678. Unsigned form
            ; will give only 0 to 65,536 digital values. Since we have used -/+
            ; 10 analog voltage operation, so we have to taken S.

Open plc30 clear
If (m5063 > 16383)
P0= (m5063-32768)*10/16383
Else
P0=m5063*10/16383
EndIf
; When # ADC Ch 8 is using, instead of M5063, M5064 will come in above PLC;
program.
Close
```

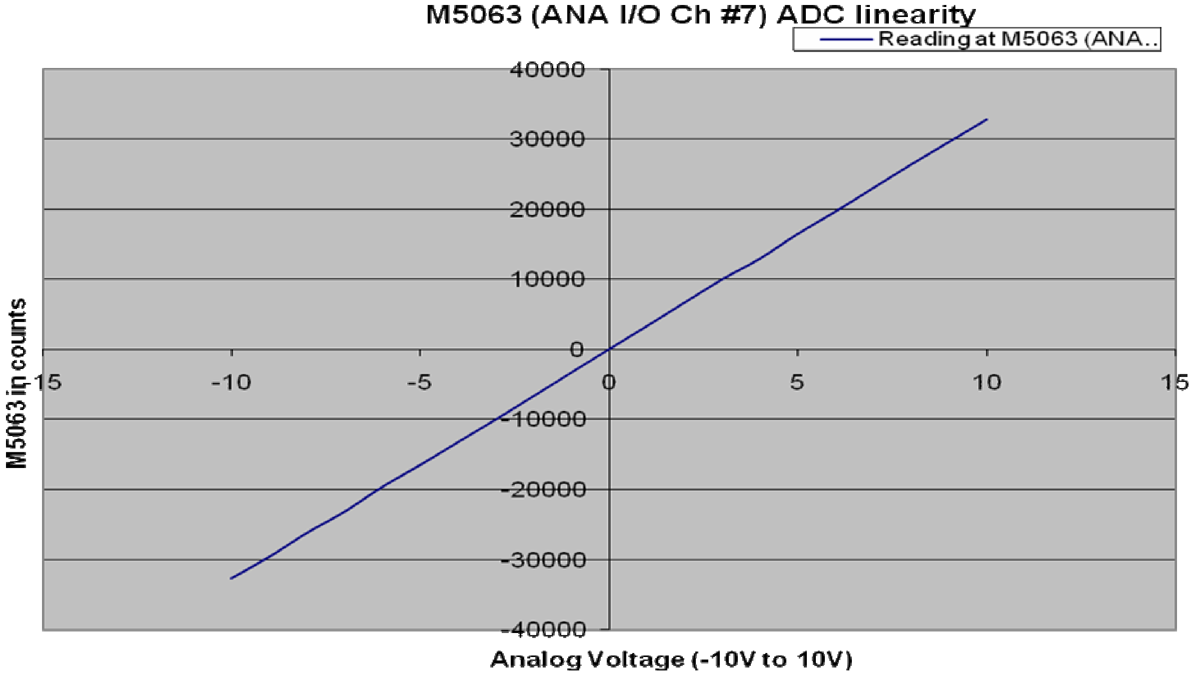
3.3) After downloading the file, go to Terminal Window and run the Plc30 by the Command ENABLE PLC30.

3.4) In VIEW menu open Watch Window add the M5063 (or/and M5064) which will show the digital value proportional to analog input.

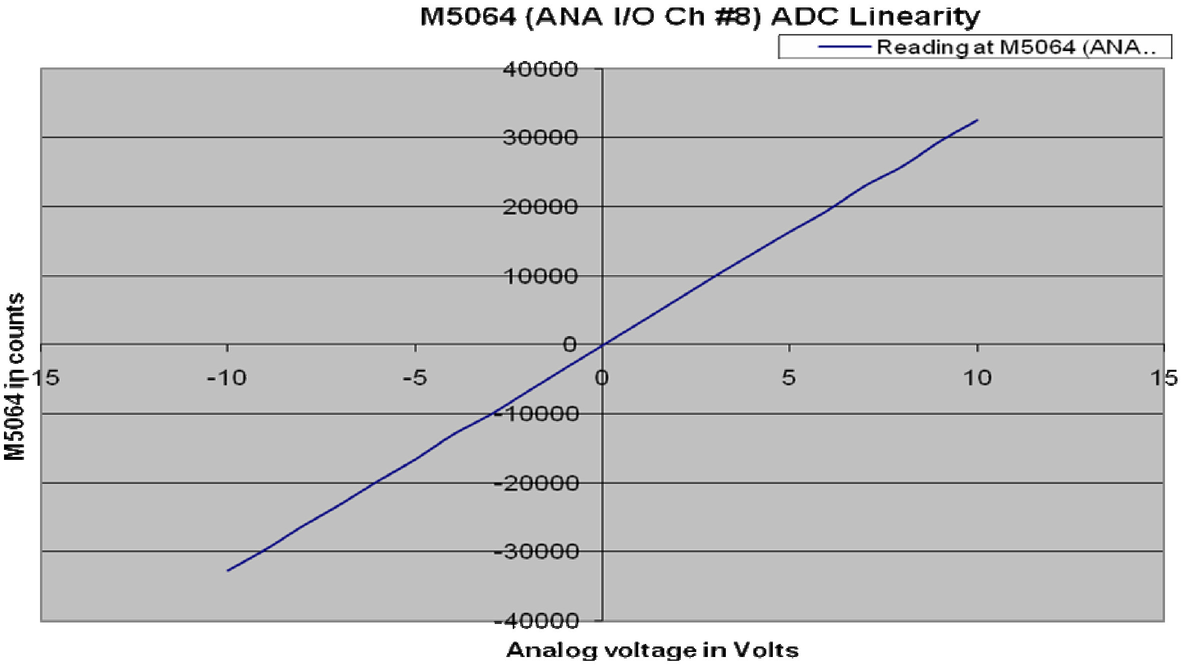
**Observation: -** M5063(or/and M5064) values are noted in the Watch Window for various input values ranging from -10 to 0 and 0 to +10 analog voltage given from an external power source to the ANA CH #7(or/and ANA CH #8).The values are tabulated and plotted as below.

| Input Analog Voltage(Volts) | Reading at M5063 (ANA I/O Ch #7) | Reading at M5064 (ANA I/O Ch #8) |
|---|---|---|
| -10 | -32687 | -32650 |
| -9 | -29625 | -29604 |
| -8 | -26202 | -26185 |
| -7 | -23170 | -23066 |
| -6 | -19627 | -19687 |
| -5 | -16518 | -16551 |
| -4 | -13253 | -12972 |
| -3 | -10037 | -10076 |
| -2 | -6697 | -6702 |
| -1 | -3337 | -3345 |
| 0 | -15 | -30 |
| 1 | 3295 | 3295 |
| 2 | 6652 | 6636 |
| 3 | 9984 | 10007 |
| 4 | 12928 | 13234 |
| 5 | 16462 | 16443 |
| 6 | 19643 | 19500 |
| 7 | 23010 | 23084 |
| 8 | 26350 | 25906 |
| 9 | 29560 | 29523 |
| 10 | 32767 | 32590 |

The plot between M5063 and corresponding Analog input



The plot between M5064 and corresponding Analog input



Conclusion: - ADC check through Ch#7 ADC (Ch #8 ADC) of MAC2_BLC2 is linear.

## 4. Servostar 712 settings

Open software 'Drive GUI ' for configuring the Drive Servostar 712

4.1) File download to SS712:-

Go to file and open “AZ1-SS712-settings “to SS712-1 (related to Azimuth 1 motor)
“AZ2-SS712-settings”to SS712-2 (related to Azimuth 2 motor)
“EL1-SS712-settings “to SS712-3 (related to Elevation 1 motor)
“EL2-SS712-settings”to SS712-4 (related to Elevation 2 motor)
The above files will be in “Desktop/C04_settings” folder.
Check for the following parameters:

4.2) Motor Selection = AKM73M (400),
SrNo = 19175,
Icont = 13.6 amp
Ipeak = 68 Amps
RPM = 1500 with Brake

4.3) Current Loop: - I2T warning = 80%
Kp_i = 37.2
I time = 0.725 m-sec

4.4) Velocity Loop: - Speed limit + ve = 1500 rpm,-ve = -1500 rpm
Over speed = 1799 rpm

4.5) Feedback: - Conn X2 - Resolver
No. of Poles = 2

4.6) Analog i/p1 - Current / Velocity Command (depends on Op-mode = 3 which is Analog Torque)
Offset = 103 (automatic) this will ensure differential i/p to SS712 = 0V
Current scale = 20 Amps / 10 Volt

4.7) Status: - Gives warning / error message eg: F08 – over speed, F04 – feedback loss

4.8) Monitor:-

Analog i/p = _________ Volts
I eff = _________ Amps
Velocity actual = _________ Rpm
I2T = _________ %
DC Bus Voltage = _________ Volts
Temperature = _________ deg.C

4.9) SS712 firmware should be 5.15 and DriveGUI version should be 2.20 build 0004.

## 5. Servostar Drive Maximum current settings

Maximum Continuous Torque 20Nm in GMRT, limited by Gearbox. Torque Constant of Brushless motor (AKM 73M) KTms =3.1Nm/A.

So Maximum Continuous Current is = Max. Cont. Torque/ KTms
= 20/3.1
= 6.5A

ICont = 6.5A.
IPeak = 13A (6.5A*2)

Note:-

1. Connect RS232 from PC to Servostar, Change the values of ICont. and I Peak.
2. These Values are identical for AZ and EL axis.

## 6. Simple Move Test.

6.1) Azimuth Axis 2 motors are connected to Servostar #1 and #2 which is Connected Ch #1 and Ch #2 of PMAC.Azimuth axis Encoder has Connected to Ch #5 through Interpolator (IBV102) which is useful when PMAC configured in Position Loop.

6.2) This test is for validation of Directions that is when open loop command has given to #1 (#1o5) or (#1o-5), #1 motor direction must be Positive or negative respectively. As well as Motor #2.

6.2) Go to PWIN32PRO and Download following program.

```
i7016, 4, 10=1
i100, 4,100=1
i122, 4,100=15
i119, 4,100=1
i123, 4,100=5
i124,4,100=$820001
i130, 4,100=500000
i131, 4,100=5000
i132, 4,100=5000
i133, 4,100=10000
i134, 4,100=0
i169, 4,100=16384
i500=1, i600=1
i7110=3, i7120=7 ; if motor and Encoder direction is not proper these values switch
                 ; between 3 and 7
```

6.3) See for 0 errors.

6.4) Go to Terminal Window and Give following commands

#1o0 #2o0 (Both the motor's brakes are released)
#1o3; Motor #1 should rotate in Positive direction (3 % of total torque)
Or #2o3; Motor #2 should rotate in Positive direction (3 % of total torque)
Or #1o-3 ; Motor #1 should rotate in Negative direction (3 % of total torque)
Or #2o-3 ; Motor #2 should rotate in Negative direction (3 % of total torque)

6.5) In Position window observe the Direction of 2 motors and #5 Axis (encoder) direction, all 3 should be in same directions.

6.6) The above Simple Move Test will be repeat for Elevation Axis.EL Axis 2 motors are connected to Servostar #3 and #4 which is connected Ch #3 and Ch #4 of PMAC. EL Encoder has connected through Interpolator to #6 (ENC6) of PMAC.

## 7. Calculation and validation of Scaling factor between Motor and Load Encoder.

### AZ Axis:-

**Calculation:-**

7.1) Resolver Count after

One revolution of big gear is = 4096*4*1488*12.6
= 307180399

| | |
|---|---|
| 4096 | = Resolution of Resolver |
| 4 | = Multiplied in PMAC |
| 1488* 12.6 | = Gear box and Gear ratio. |

7.2) Load Encoder Count after

One revolution of big gear is = 8192* 4 * 100
= 3276800

| | |
|---|---|
| 8192 | = Encoder resolution |
| 4 | = Quadrature decode |
| 100 | = Interpolator |

7.3) Motor to load Ratio = 307180399/3276800
= 97.7

**Validation:-**

7.4) Go to PWIN32PRO and Download Following program.

```
i7016, 4, 10=1
i100, 4,100=1
i122, 4,100=15
i119, 4,100=1
i123, 4,100=5
i124, 4,100=$820001
i130,4,100=500000
i131, 4,100=5000
i132, 4,100=5000
i133,4,100=10000
i134, 4,100=0
i169, 4,100=16384
i500=1, i600=1
i7110=3,i7120=7
```

7.5) See for 0 errors.

7.6) Go to Terminal Window and Give following commands

#1o0 #2o0 #5o0 (Both the motor's brakes are released, enabled #5 axis)
#1o5 or #2o5 or #2o5 or #2o-5

7.7) In Position window , Motor #1 and #5 axis(Encoder) readings are given below

| #1 Motor<br>Reading in counts | #5 axis (Encoder)<br>reading in counts | #5 axis (Encoder)<br>by calculation<br>(Motor1/97.7) |
|---|---|---|
| 1, 98,106 | 2,059 | 2,063 |
| -6, 90,000 | -74,169 | 71,875 |

**EL Axis:-**

**Calculation:-**

7.8) Resolver Count after

One revolution of big gear is = 4096*4*821.976*30.55
= 411424633.7

4096 = Resolution of Resolver
4 = Multiplied in PMAC
821.976* 30.55 = Gear box and Gear ratio.

7.9) Load Encoder Count after

One revolution of big gear is = 8192* 4 * 100
= 3276800

8192 = Encoder resolution
4 = Quadrature decode
100 = Interpolator

7.10) Motor to load Ratio = 411424633.7/3276800
= 125.5568

7.11) The above AZ scaling factor validation test will be repeated for the EL axis.

## 8. One motor(#1 or #2) in Closed loop with resolver and second motor(#2 or #1) brakes released

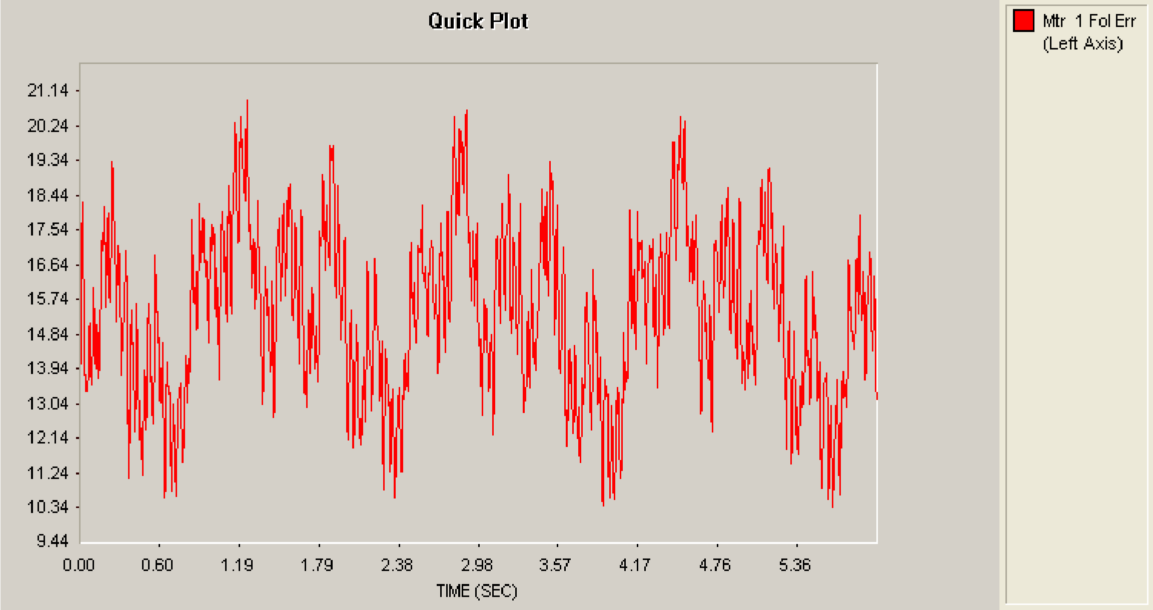
8.1) Go to PEWIN32PRO and download following Program.

```
Close
endg
del gat
i7016, 4, 10=1                ; true DAC Output
i100, 4,100=1                 ; activate axis
i122, 4,100=15                ; slow default speed
i119, 4,100=0.2               ; higher acc + dec
i123, 4,100=5                 ; homing speed
i124, 4,100=$20001            ; no limit switches
i130, 4,100=120000            ; PID Settings
i131, 4,100=1050
i132, 4,100=1050
i133, 4,100=10000
i134, 4,100=1
i169, 4,100=16384             ; 10V diffential DAC Output
; AZ load encoder setting (connected to ENC5 input)
i7110=3,i7120=7               ; changing counting direction
```

8.2) Motor #1 and motor #2 brakes are released ---- #1o0 #2o0
Motor #1, speed has given as 10 cts/ms i122=10
Motor #1 in closed and jog command ------------------------ #1j+

8.3) Go to PMAC PLOT pro , take X-axis as time (sec) and Y-axis as Motor #1 Following error.

The following Plot is Motor #1 Following Error.

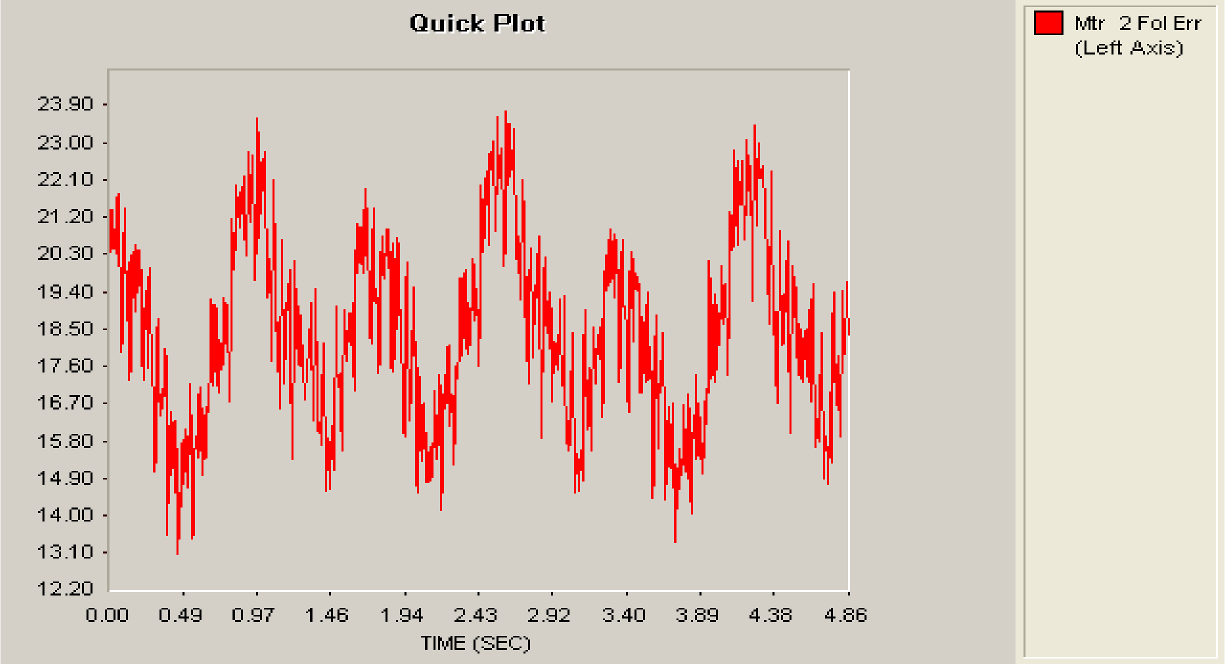


8.4) **Effective Current** through Motor #1 is **0.4A to 0.6A** (Checked in Servostar #1, drive related Motor #1) and **RPM** of Motor #1 is **35 to 50 rpm**.

8.5) Similarly Motor #1 kept in open loop and motor #2 kept in closed loop.

Motor #1 and motor #2 brakes are released ---- #1o0 #2o0
Motor #2, speed has given as 10 cts/ms i222=10
Motor #2in closed loop and jog command ------------------------ #2j+

The following Plot is Motor #2 Following Error.



8.6) **Effective Current** through Motor #2 is **0.4A to 0.6A** (Checked in Servostar #2 drives related Motor #2 and **RPM** of Motor #2 is **35 to 50 rpm**.

Note: - Effective current through Motor #1(or Motor#2) is 0.5 A to 1.5 A (Checked in Servostar #1or Servostar #2) and RPM of Motor #1(Motor #2) is 1000rpm.

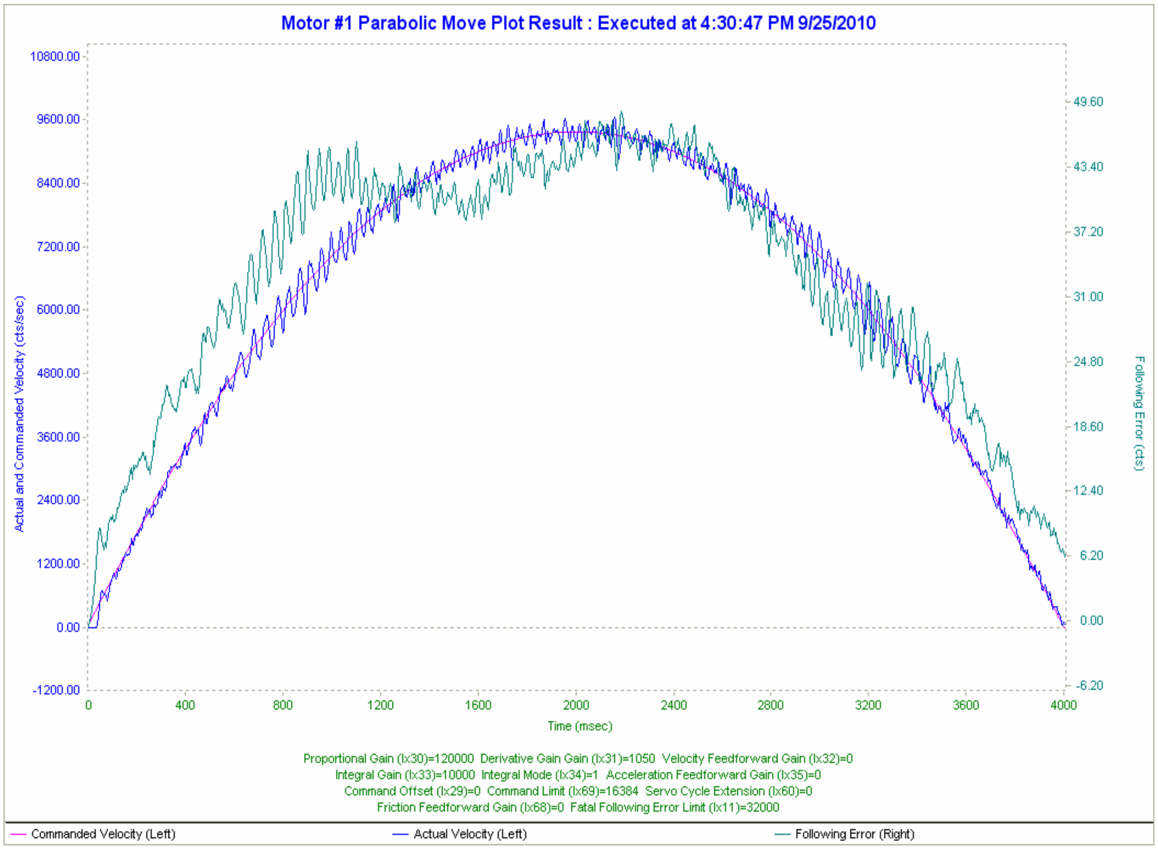
8.7) The above Test (one motor (#3 or #4)in closed loop with resolver and second motor(#4 or #3) brakes released) will be done in EL axis as above.

## 9. Tuning Motor #1 (or Motor #2) with Parabolic Profile in AZ axis and Motor #3(or Motor #4) in EL axis

Parabolic Move with following Parameters

I130=120000,
I131=1050,
I132=0,
I133=10000,
I134=1.

Go to PMAC TUNING PRO, select the parabolic profile and set the above PID values.

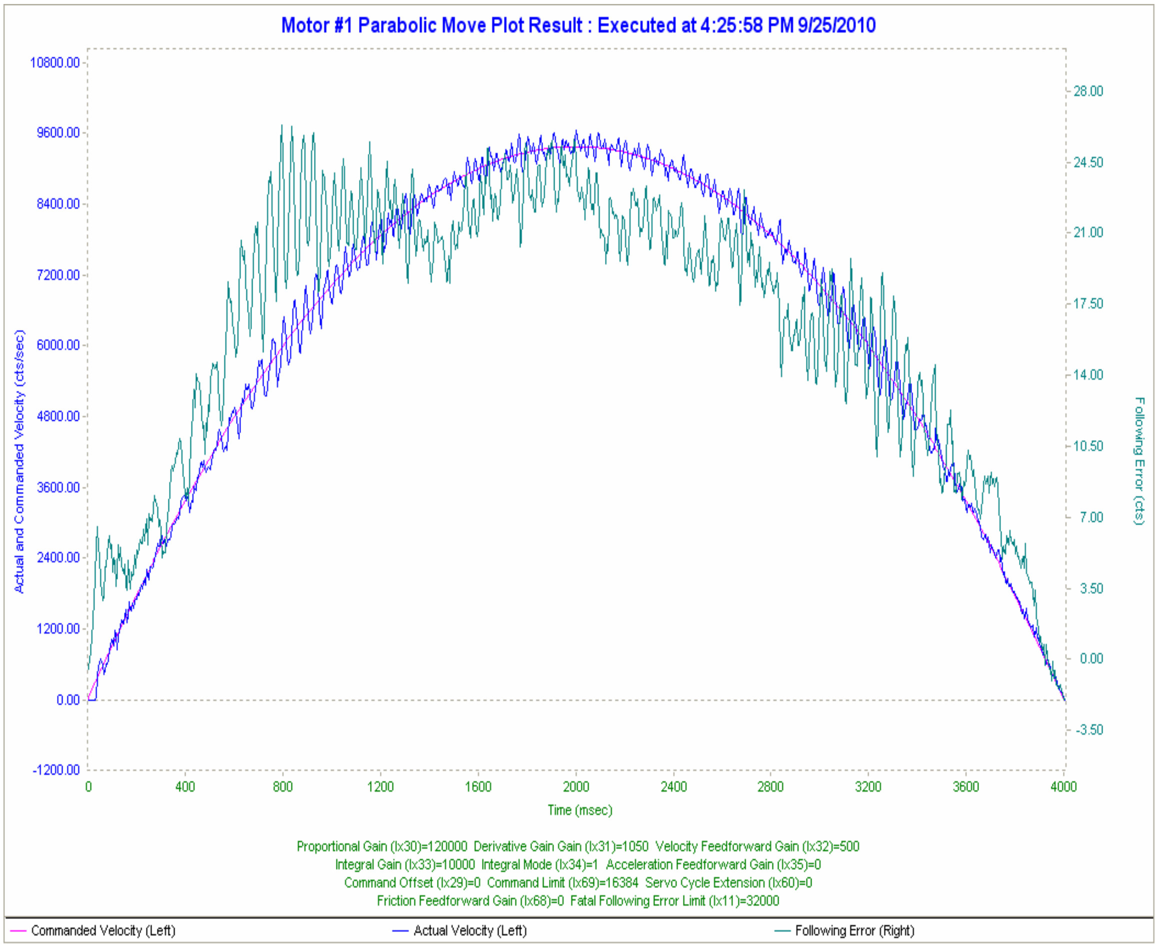


Proportional Gain (Ix30)=120000  Derivative Gain Gain (Ix31)=1050  Velocity Feedforward Gain (Ix32)=0
Integral Gain (Ix33)=10000  Integral Mode (Ix34)=1  Acceleration Feedforward Gain (Ix35)=0
Command Offset (Ix29)=0  Command Limit (Ix69)=16384  Servo Cycle Extension (Ix60)=0
Friction Feedforward Gain (Ix68)=0  Fatal Following Error Limit (Ix11)=32000

Note: - In above plot Correlation between Following error and Velocity is more and following error is also more. So for less correlation and Following error we have to increase Kvff.

Changed i132 from 0 to 500.

I130=120000,
I131=1050,
I132=500,
I133=10000,
I134=1.

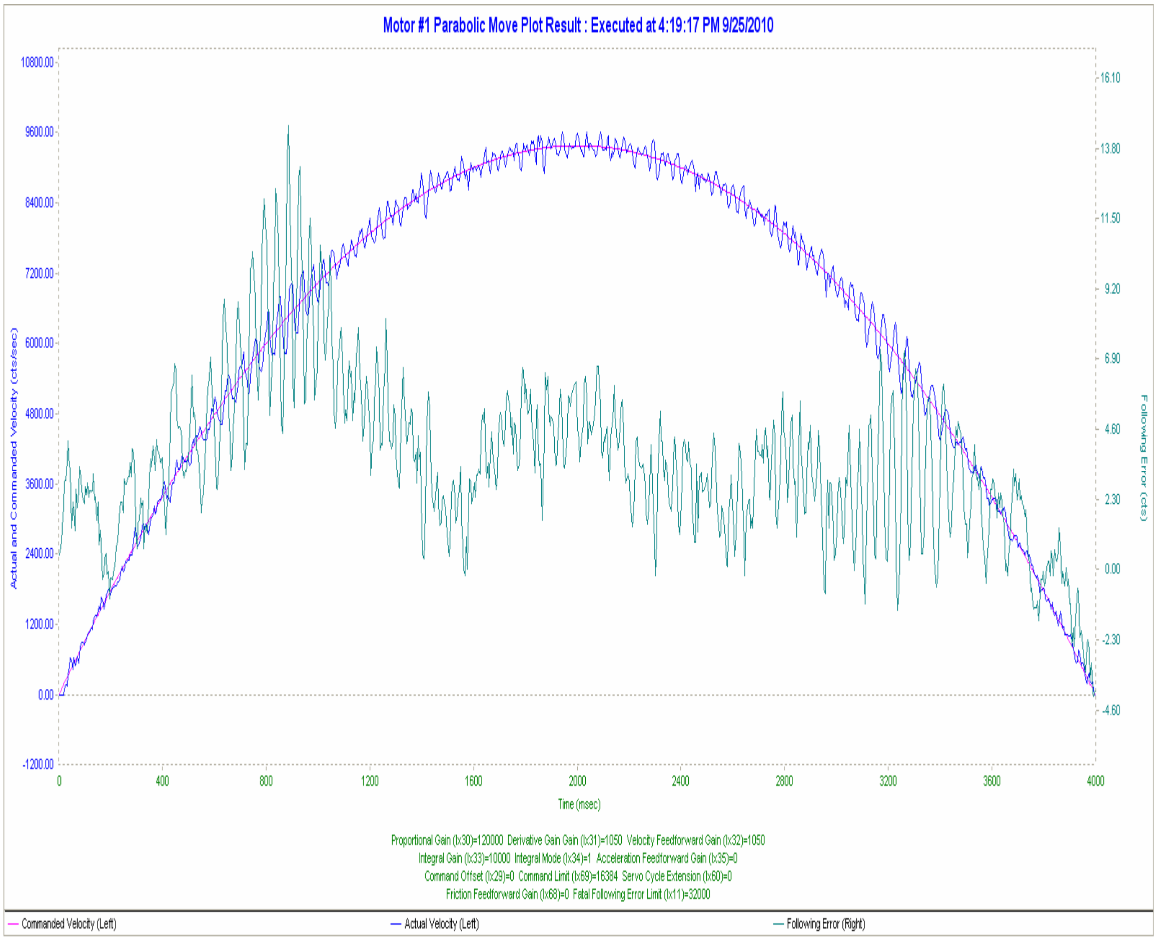


Proportional Gain (Ix30)=120000 Derivative Gain Gain (Ix31)=1050 Velocity Feedforward Gain (Ix32)=500
Integral Gain (Ix33)=10000 Integral Mode (Ix34)=1 Acceleration Feedforward Gain (Ix35)=0
Command Offset (Ix29)=0 Command Limit (Ix69)=16384 Servo Cycle Extension (Ix60)=0
Friction Feedforward Gain (Ix68)=0 Fatal Following Error Limit (Ix11)=32000

Note: - In above plot Correlation between Following error and Velocity is more and following error is also more. So for less correlation and following error we have to increase Kvff.

Changed i132 from 500 to 1050.

I130=120000<br>
I131=1050
I132=1050
I133=10000
I134=1

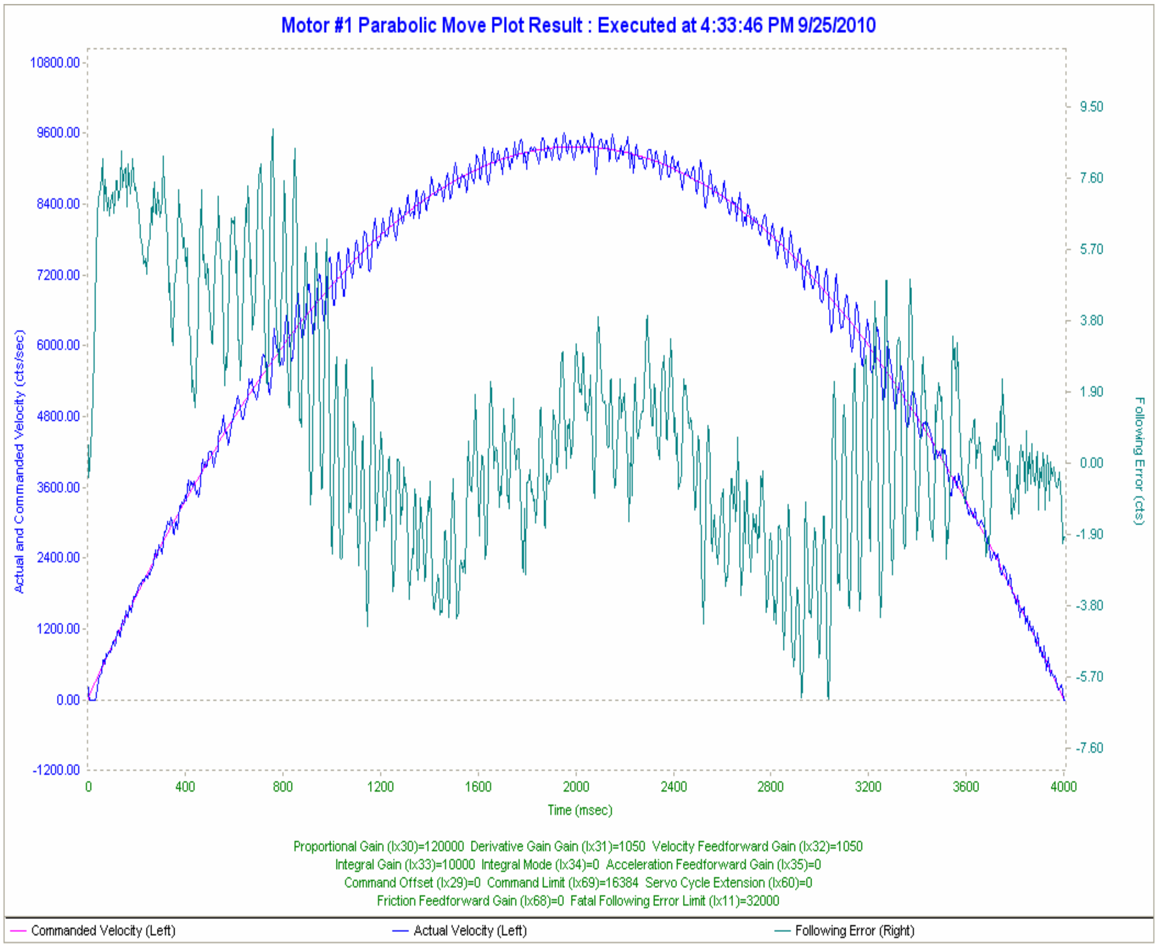


Proportional Gain (Ix30)=120000 Derivative Gain Gain (Ix31)=1050 Velocity Feedforward Gain (Ix32)=1050
Integral Gain (Ix33)=10000 Integral Mode (Ix34)=1 Acceleration Feedforward Gain (Ix35)=0
Command Offset (Ix29)=0 Command Limit (Ix69)=16384 Servo Cycle Extension (Ix60)=0
Friction Feedforward Gain (Ix68)=0 Fatal Following Error Limit (Ix11)=32000

Note: - In above Plot there is not much correlation between Following error and Velocity. So I133 (Kvff) value is accepted.

Changed i134 from 1 to 0.

I130=120000
I131=1050
I132=1050
I133=10000
I134=0

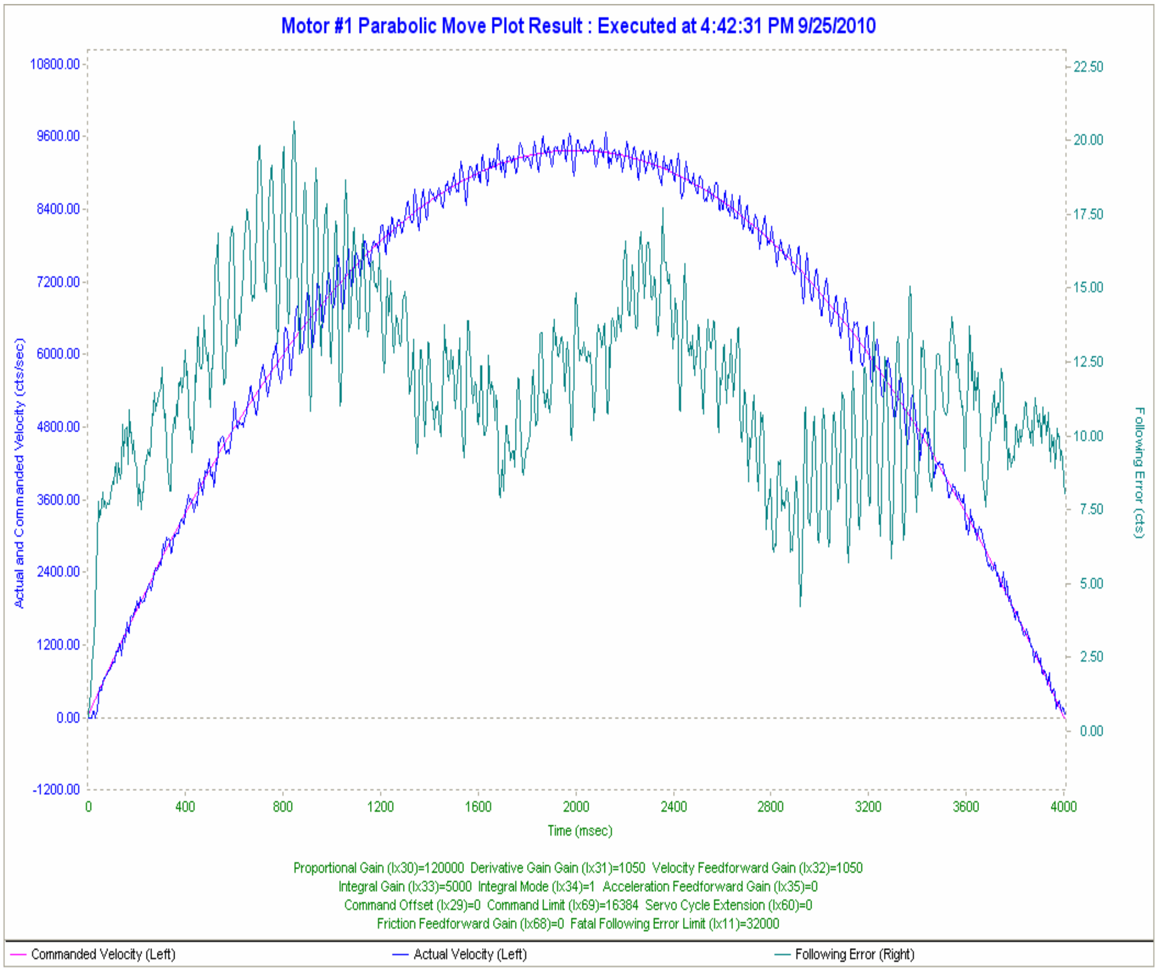

Motor #1 Parabolic Move Plot Result : Executed at 4:33:46 PM 9/25/2010

Proportional Gain (Ix30)=120000  Derivative Gain Gain (Ix31)=1050  Velocity Feedforward Gain (Ix32)=1050
Integral Gain (Ix33)=10000  Integral Mode (Ix34)=0  Acceleration Feedforward Gain (Ix35)=0
Command Offset (Ix29)=0  Command Limit (Ix69)=16384  Servo Cycle Extension (Ix60)=0
Friction Feedforward Gain (Ix68)=0  Fatal Following Error Limit (Ix11)=32000

Note: - In above Plot Following is less and the correlation between Following error and Velocity is there, but in inverted. So we have to make IM (I134) =1.

Changed i133 from 10000 to 5000.

I130=120000
I131=1050
I132=1050
I133=5000(decreased from 5000 from 10000)
I134=1



Proportional Gain (Ix30)=120000 Derivative Gain Gain (Ix31)=1050 Velocity Feedforward Gain (Ix32)=1050
Integral Gain (Ix33)=5000 Integral Mode (Ix34)=1 Acceleration Feedforward Gain (Ix35)=0
Command Offset (Ix29)=0 Command Limit (Ix69)=16384 Servo Cycle Extension (Ix60)=0
Friction Feedforward Gain (Ix68)=0 Fatal Following Error Limit (Ix11)=32000

Note: - In above Plot Ki (I133, Integral Gain) is 5000, so following error is increasing.

# 10. Test with Load Encoder by adding Backlash Algorithm.

10.1) Go to PWIN32PRO and set Kp and Kd.

10.2) If want to add notch filter click on Notch filter calculator. Enter resonant frequency which is the frequency of the mechanical resonance to notch out. To calculate resonant frequency, do step move from tuning screen and measure.

10.3) KP value will also change according resonant frequency.

10.4) Let Resonant frequency is 1000Hz, Click on calculate notch filter gains and implement.

```
I 530=12000          ;Motor #5 Proportional gain
I531=1000            ; Motor #5 Derivative gain
I532=0               ; Motor #5 Velocity FeedForward Gain
I533=0               ; Motor #5 Integral gain
I534=0               ;Motor #5 Integral Mode
I535=0               ; Motor #5 Acceleration FeedForward Gain
I536=0               ; Motor #5 Notch filter coefficient N1
I537=0               ; Motor #5 Notch filter coefficient N2
I538=-0.468234       ; Motor #5 Notch filter Coefficient D1
I539=0.078913        ; Motor #5 Notch filter Coefficient D2
(N1,N2,D1 and D2 have put the values such that Notch  filter at 1000Hz)
I568=400             ; Friction offset.
```

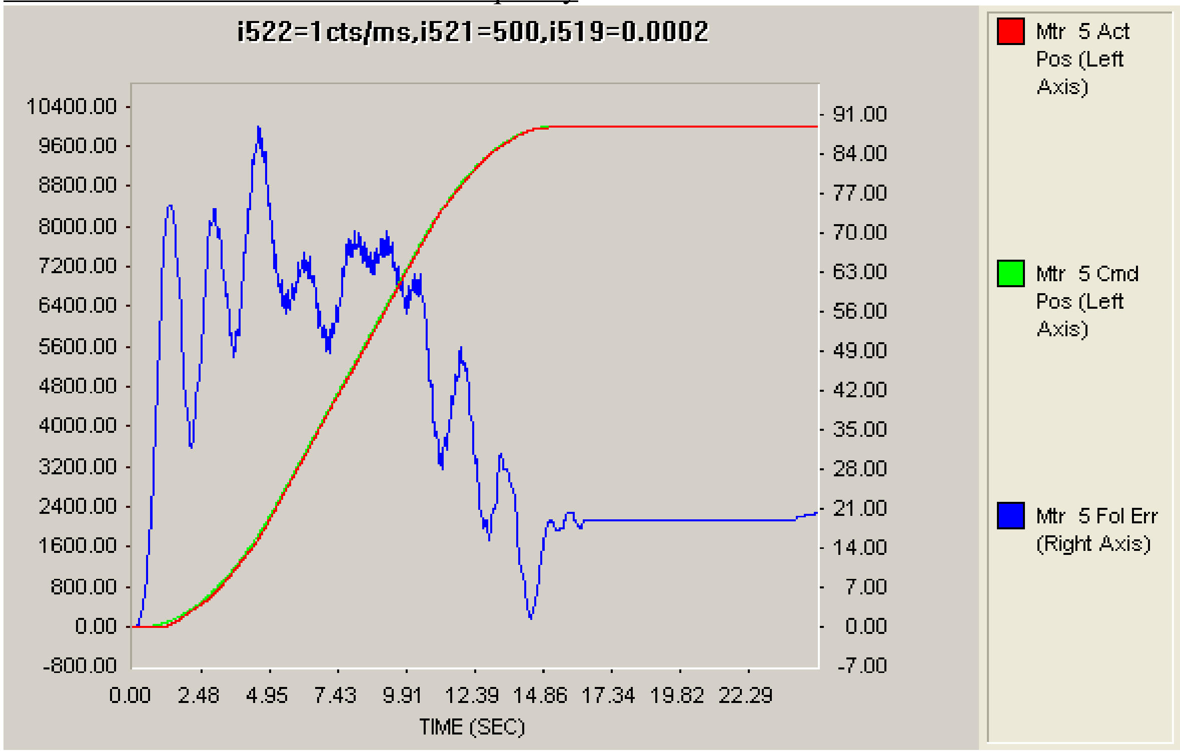
10.5) Add these values in File named “AZ preload position.pmc in Annexure-A” which will have I variable related motor #1,Motor #2 and Encoder #5, as well as Backlash Compensation algorithm PLC0.

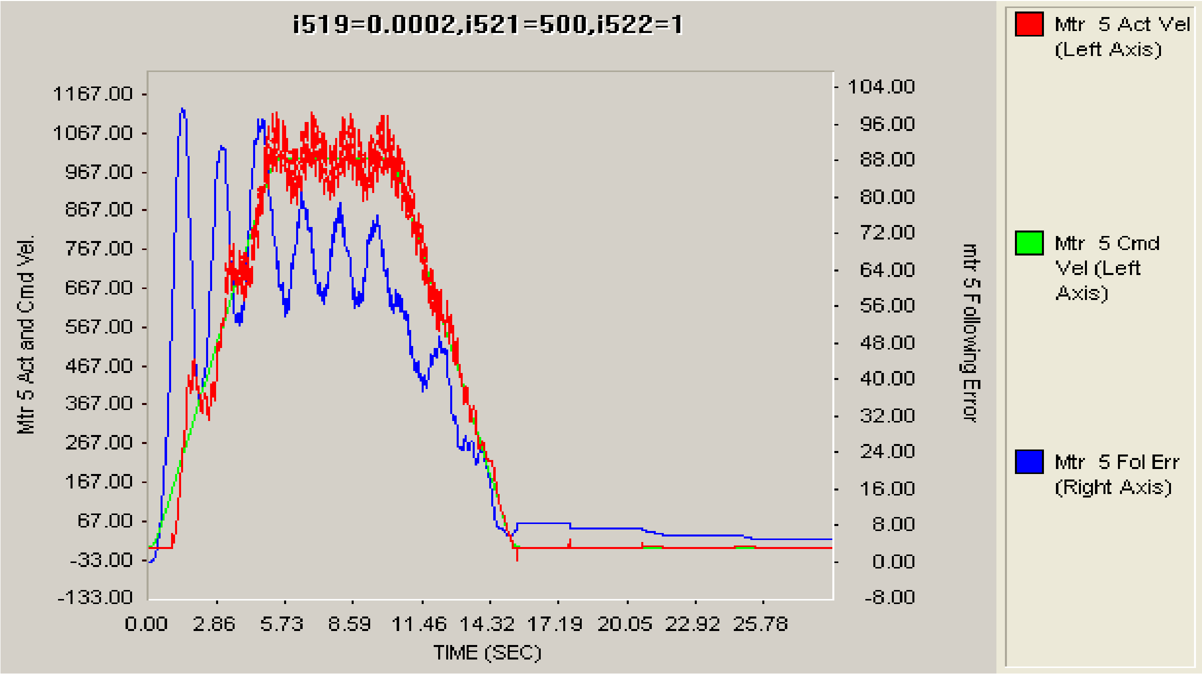
10.6) Go to Terminal Window.

```
 #1o0 #2o0 #5o0
I519=0.0002 cts/ms2
I521=500 ms
I522=1cts/ms
 Define gather
 Gather j=10000
 End gather
```

10.7) Go to PMAC PLOT PRO, take Motor #5 Act and Cmd Position in left Y-axis,Motor #5 following error in right axis. As well as Motor #5 Act and cmd Velocity in left Y-axis, Motor #5 following error in right axis.

Notch Filter with 1000hz resonant frequency





10.8) If want to add Low pass filter click on Low pass filter calculator. Enter Cutoff frequency which is actual frequency to use for the low pass filter in the servo loop. To Calculate cutoff frequency, do step move from tuning screen and measure.

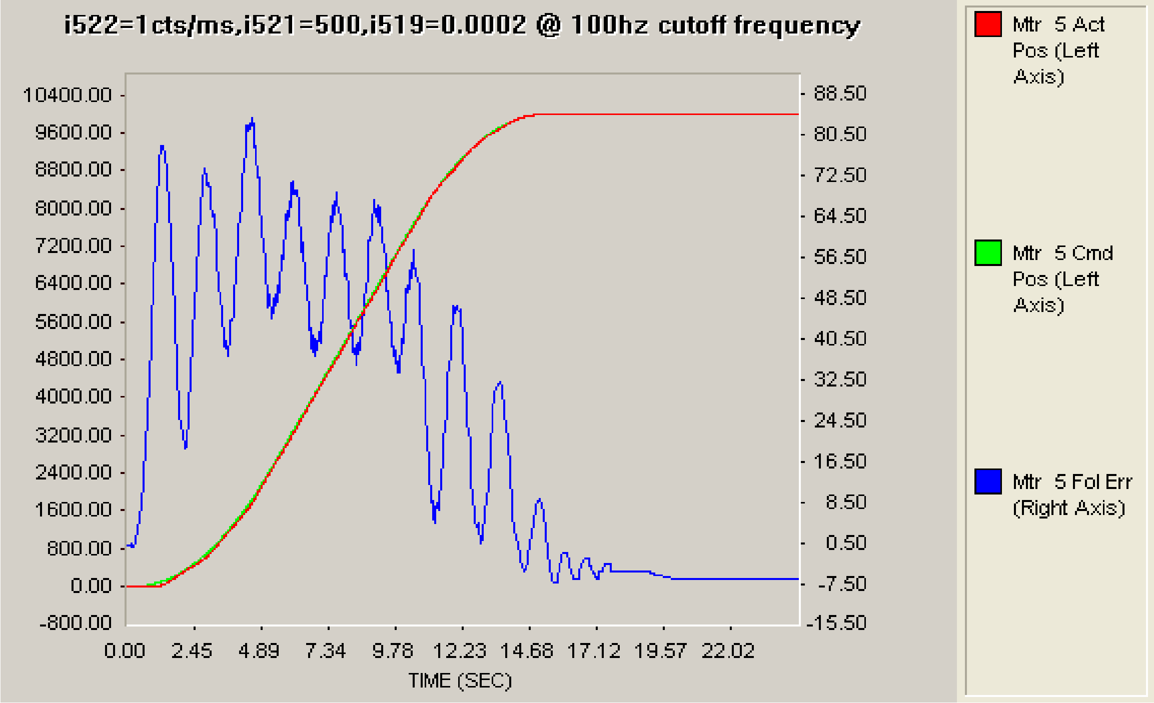
10.9) KP value will also change according Cutoff frequency.

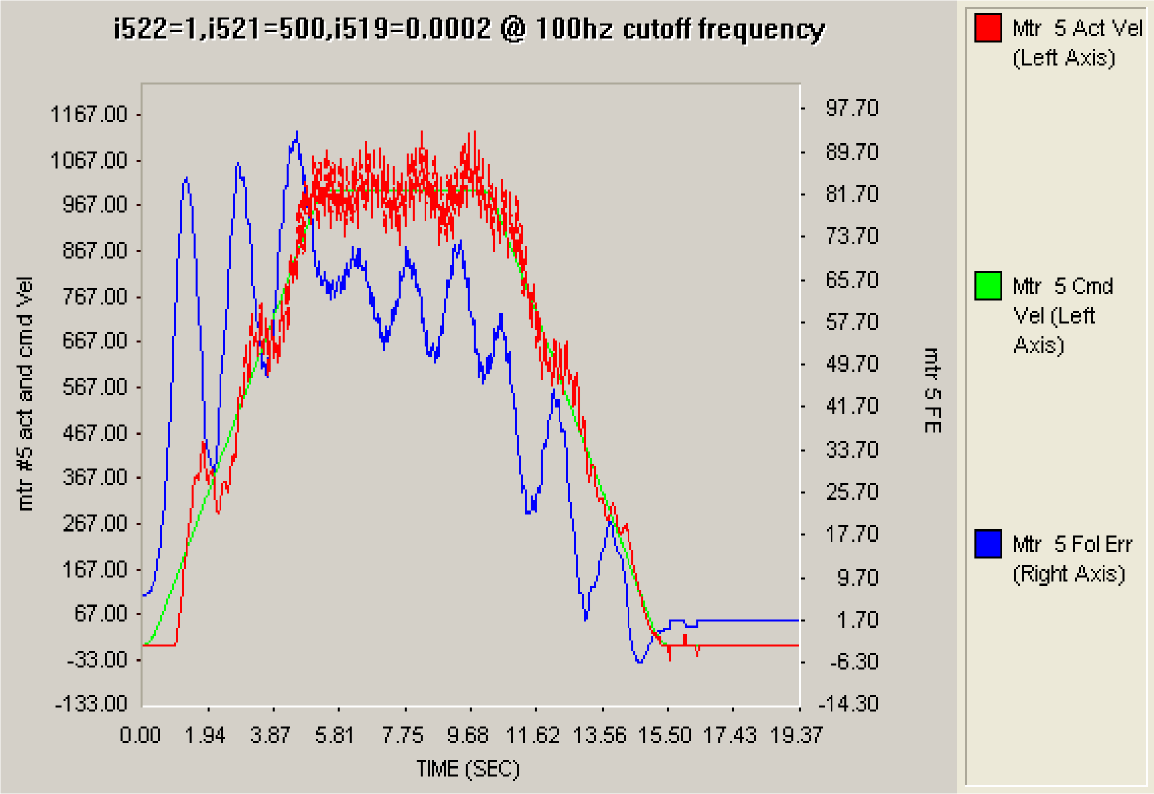
10.10) Let Cutoff frequency is 100Hz, Click on calculate low pass filter gains and Implement.

```
I 530=1000
I531=1000
I532=0
I533=0
I534=0
I535=0
I536=0
I537=0
I538=-1.62731
I539=0.67993
(I536, I537, I538 and I539 values set such that Low pass 2 order at 100Hz).
I568=400
```

10.11) Add these values in File named “AZ preload position.pmc in Annexure-A” which will have I variable related motor #1,Motor #2 and Encoder #5, as well as Backlash Compensation algorithm PLC0. Repeat 10.6 and 10.7 steps.

Low pass Filter with 100 Hz cutoff frequency





10.12) The above test will be repeated for EL axis.

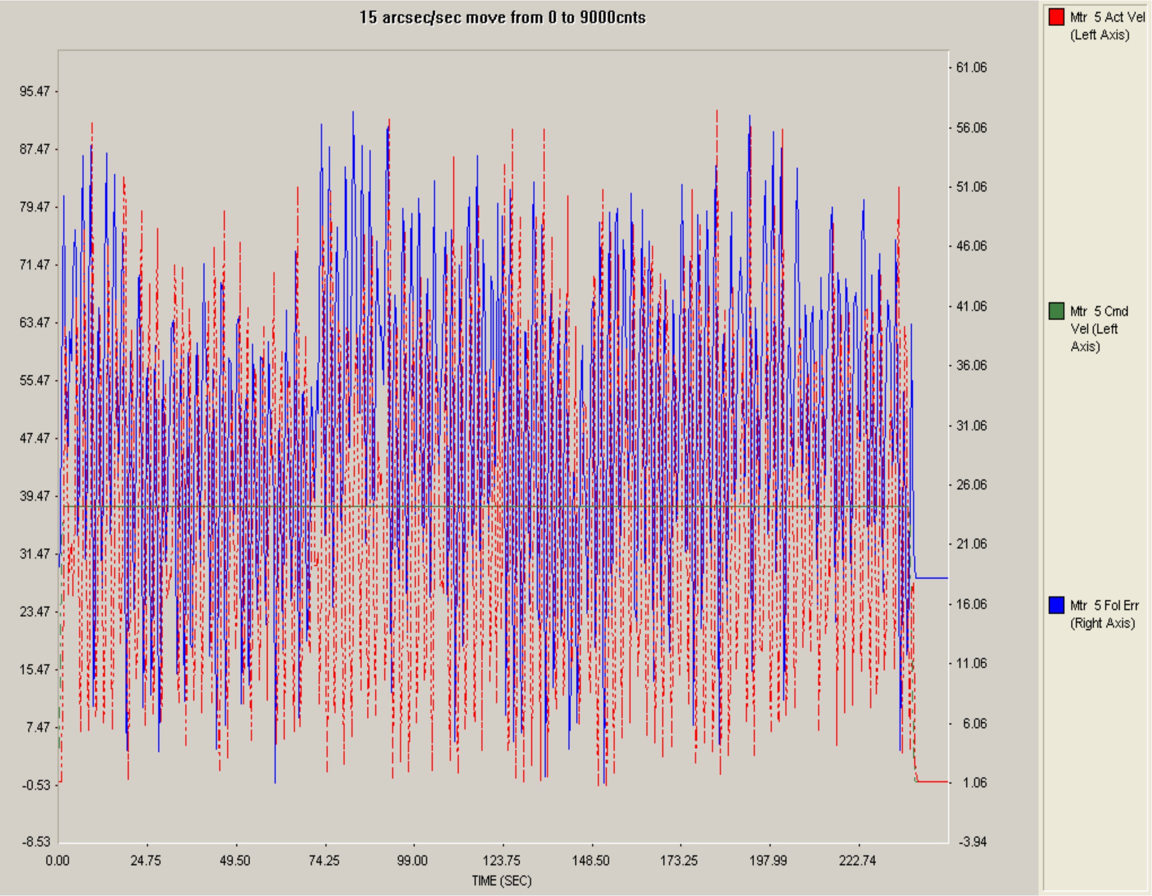
## 11. Running Antenna by giving command to Axis #5 and gathering data for the following values of i522.

I522=0.038 cts/ms → 15 arcsec/sec (low speed)
I522=0.38 cts/ms → 150 arcsec/sec
I522=4.5cts/ms → 30 deg /min (high speed)

11.1) Repeat Step 10 with i522=0.038 cts/ms (15 arcsec/sec move) instead of i522=1cts/ms by giving following commands in terminal window.

```
 #1o0 #2o0 #5o0
I519=0.0002 cts/ms2
I521=500 ms
I522=1cts/ms
 Define gather
 Gather j=9000
 End gather
```

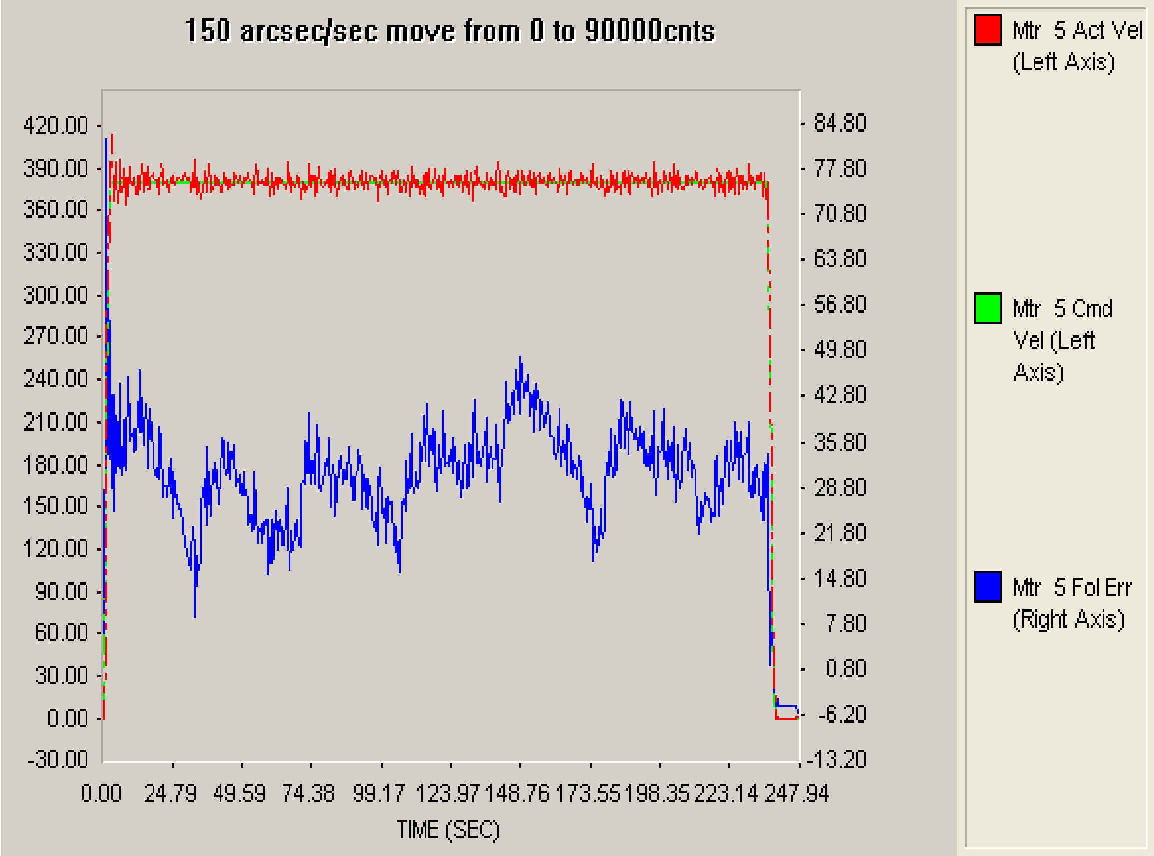
15 arcsec/sec move from 0 to 9000cnts with i522=0.038cnts/ms



11.2) With i522=0.38 cts/ms (150 arcsec/sec move) by giving following commands

```
 #1o0 #2o0 #5o0
I519=0.0002 cts/ms2
I521=500 ms
I522=1cts/ms
Define gather
Gather j=90000
End gather
```

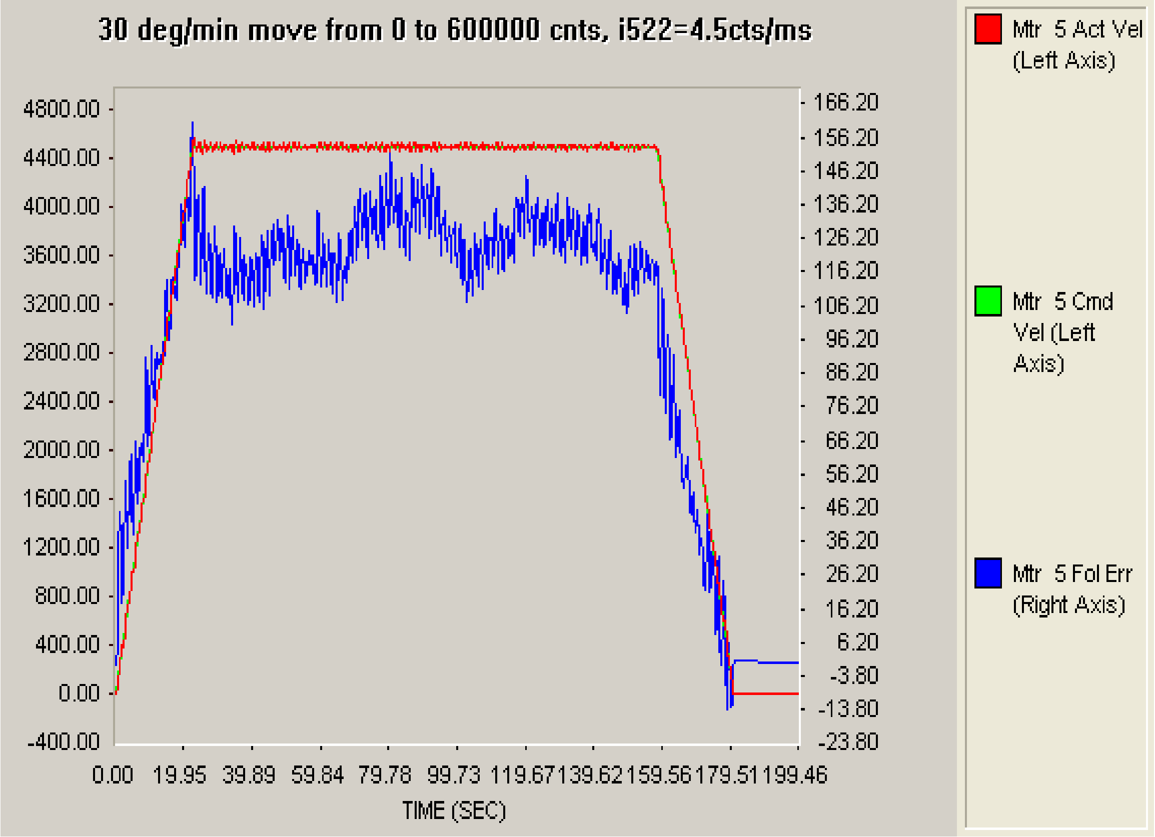
150 arcsec/sec move from 0 to 90000cnts with i522=0.38cnts/ms



11.3) With i522=4.5 cts/ms (30 deg/min move) by giving following commands

```
 #1o0 #2o0 #5o0
I519=0.0002 cts/ms2
I521=500 ms
I522=1cts/ms
Define gather
Gather j=600000
End gather
```

30deg/min move from 0 to 600000cnts with i522=4.5cnts/ms



11.4) The above tests will be repeated for EL axis.

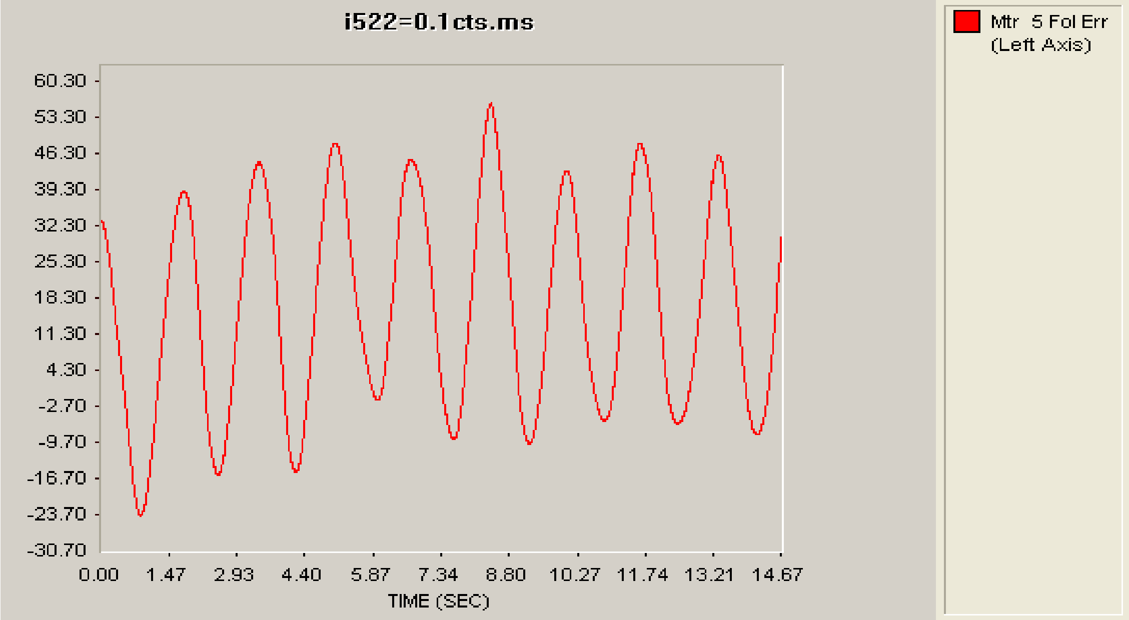
## 12. Running Antenna with Total PMAC position loop by giving i522=0.1cts/ms and i522=0.2 cts/ms.

12.1) Go to PMAC PEWIN32PRO and Download the file named "**AZ preload Position.pmc- Annexure-A"** which will contain I variables of Motor #1, Motor #2, Encoder (#5) and Backlash algorithm PLC0.

12.2) Go to terminal Window , give the following commands

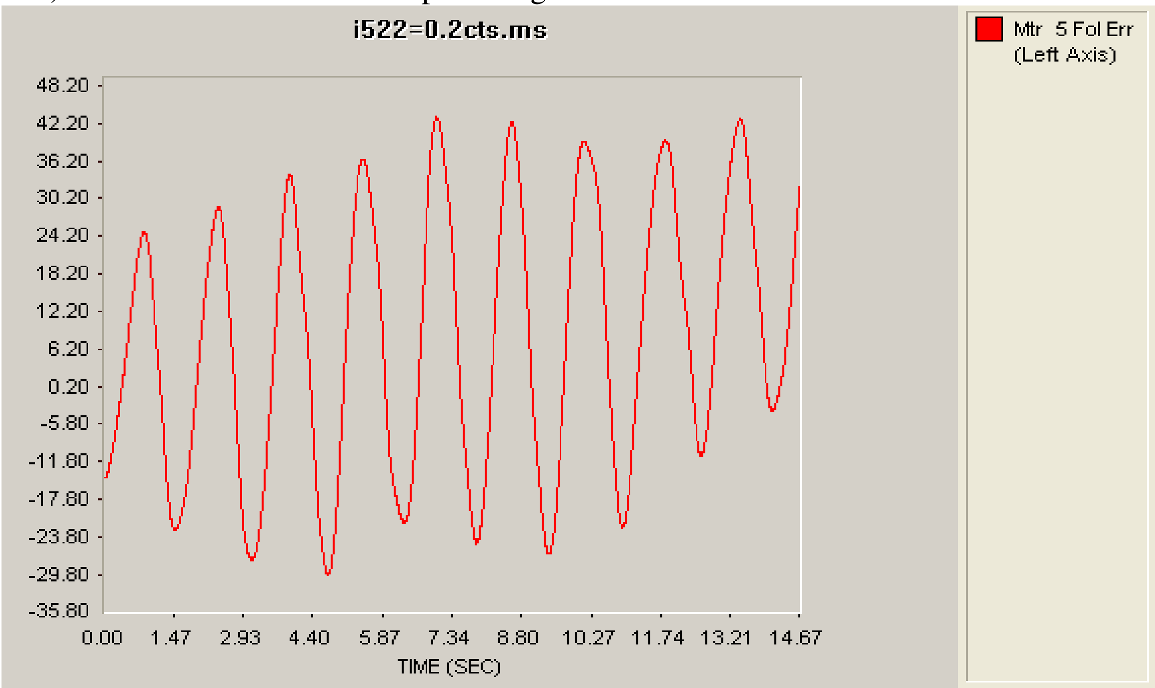
```
#1o0 #2o0 #5o0  ; enabling #1,#2 and #5 axis
#5j/            ; kept #5 axis in closed loop
I522=0.1        ; speed as 0.1cts/ms
#5j+            ; jog command
```

12.3) In PMAC PLOT PRO , take x-axis as time(sec) and y-axis as Motor #5 Following Error. The following Plot shows the Motor 5 following error.



12.4) Following Error is between -25 cts (10 arcsec) to 53 cts (21.2 arcsec).

12.5) With i522=0.2 cts/ms the plot has given as



12.6) Following Error is between -30cts (12 arcsec) to 42 cts (16.8 arcsec).

12.7) The above test will be repeated in EL axis.

# 13. Running Antenna with PMAC configured in Velocity loop-Azimuth Axis

13.1) For making this PMAC PID filter as velocity loop in AZ axis, we have to make position Feedback needs to be zero. For that we are taking load encoder feedback to #7 Channel, so while closing velocity loop, Position feedback will remain open at Channel #5 and position Feedback will be zero. At Ch #7 we can get encoder Data, which we can used for plotting data between Actual Velocity (arcsec/sec) and Time (sec).

13.7) Here Analog input is given to Ch#7 ADC channel of PMAC which address is given By M5063-> y: $78115, 8, 16, S.Means Analog value will be stored in Signed 16 Bits Form. This value will scaled and stored in Hand-wheel Position register (M567).So M567 is given by

**M567= 32*96*M5063* AzcmdvelScaling factor/ 32767**.

13.8) Go to “PEWIN PRO32 PRO “and download the following PLC0 program for AZ axis velocity loop.

```
#include "Tp2mvar.pmc"

close
endg
del gat

i7016, 2,10=1 ; true DAC Output

i100,2,100=1; activate axis
i122,2,100=15; slow default speed
i119,2,100=0.2; higher acc + dec

i123,2,100=5; homing speed
i124=$820001; no limit switches
i224=$820001

i130,2,100=120000; PID Settings
i131,2,100=1050
i132,2,100=1050
i133,2,100=10000
i134,2,100=1

i169,2,100=10000; 10V diffential DAC Output
; AZ load encoder setting (connected to ENC5 input)
i7110=7
```

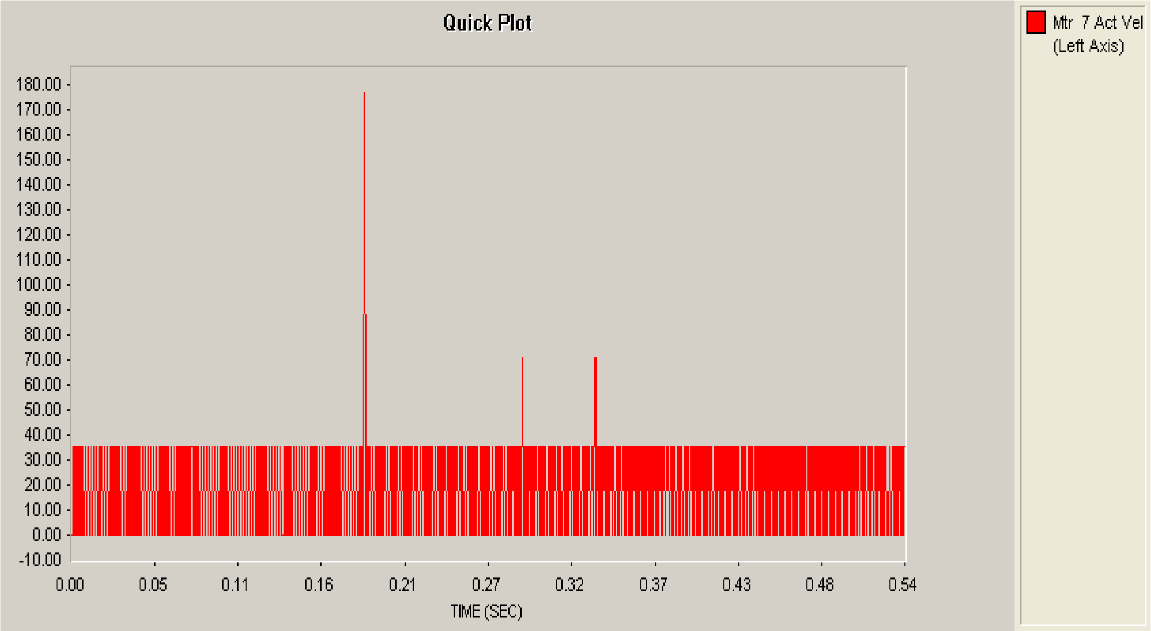
```
                                                                          ;
I7010=3
I7020=3

; definitions for the analog input reading
I7106 = $1FFFFF; ADC strobe word
M5063->Y:$78115,8,16,s ; ch7 A-D channel

; -------------------------------------------------------
; PLCC0 real-time task for torque offset and active damping

; standard AZ position/speed control loop at axis 5
; control output AZ of axis 5 distributed to axes 1 and 2
; adding a torque offset
; axes 1 and 2 must be activated via command o0
; when killing axes 1 and 2, torqueOffset must be reset to 0

#define AZvelocityLoad        M574    ; filtered (unfiltered is M566)
#define AZvelocityMotor1      M174    ; filtered (unfiltered is M166)
#define AZvelocityMotor2      M274    ; filtered (unfiltered is M266)
#define AZtorque1            M179
#define AZtorque2            M279
#define AZdesTorque           M568
#define AZcmdPos             M561
#define AZactPos             M562

#define AZposError           P1      ; position control deviation
#define AZFRICTION_Offset     P2      ;
#define AZD1                 P3      ; damping coefficient 1
#define AZD2                 P5      ; damping coefficient 2
#define AZGR                 P6      ; gear ratio
#define AZMAX_TORQUE          P7      ; Nm scaled to 16 bit integer
#define AZTORQUE_OFFSET       P8      ; Nm scaled to 16 bit integer
#define AZcmdVelScaling  P9

AZD1           = 0
AZD2           = 0
AZGR           = 8.64257
AZMAX_TORQUE    = 32768
AZTORQUE_OFFSET = 1000
AZcmdVelScaling = 190

I5 = 3     ; PLC program control enabled
I8 = 0     ; PLCC 0 called every sample
; motor encoders used for velocity feedback
```

```
I8008 = $E00100     ; sum of motor 1 and 2 encoders written into
; motor 5 velocity feedback register

i500 = 1
i503 = $350B
i504 = $3509
i506 = 1; enable master encoder (handwheel) in order
; for setting the desired vel input signal via
; m567 (scaled by 1/(32*i507)
i507 = 96
i508 = 96
I509 = 48    ; motor 5 velocity scaling factor
; set to half of the scaling of i508 since
; summation of to input = resolver are considered
i530 = 1000000; PID Settings
i531 = 128
i532 = 0
i533 = 0
i534 = 1
i538 = 0
i539 = 0
i519 = 0.0002
i522 = 1
i523 = 1
i524 = $20001
i568 = 0; Friction FF term
i511 = 0
i512 = 0; setting the error limits to zero in order to
; avoid any influence to the velocity loop

OPEN PLC 0 CLEAR

; analog input reading and scaling
m567 = 96 * 32 * m5063 * AZcmdVelScaling / 32767


; torque offset

If (AZdesTorque < 0)
AZtorque2 = AZdesTorque/2 - AZTORQUE_OFFSET;
If (AZtorque2 < -AZMAX_TORQUE/2)
AZtorque2 = -AZMAX_TORQUE/2
EndIf
```

```
AZtorque1 = AZdesTorque - AZtorque2
Else
AZtorque1 = AZdesTorque/2 + AZTORQUE_OFFSET;
If (AZtorque1 > AZMAX_TORQUE/2)
AZtorque1 = AZMAX_TORQUE/2
EndIf
AZtorque2 = AZdesTorque - AZtorque1
EndIf

; saturation

If (AZtorque1 > AZMAX_TORQUE/2)
AZtorque1 = AZMAX_TORQUE/2
EndIf
If (AZtorque1 < -AZMAX_TORQUE/2)
AZtorque1 = -AZMAX_TORQUE/2
EndIf
If (AZtorque2 > AZMAX_TORQUE/2)
AZtorque2 = AZMAX_TORQUE/2
EndIf
If (AZtorque2 < -AZMAX_TORQUE/2)
AZtorque2 = -AZMAX_TORQUE/2
EndIf

CLOSE   ; PLC 0
```

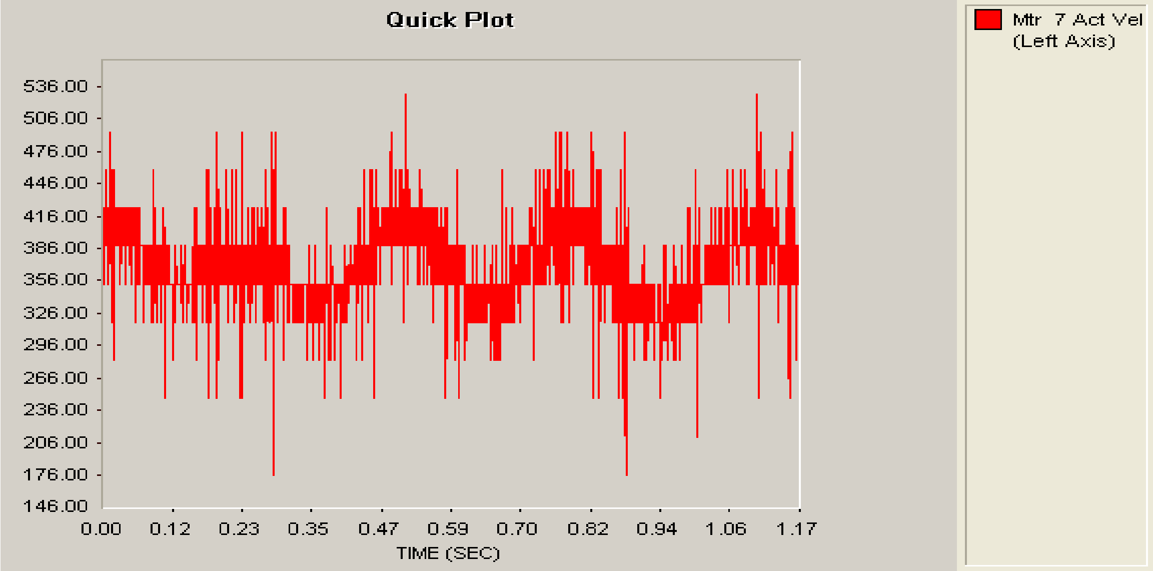
13.9) For running Antenna with **15 arcsec/sec** , the AzcmdvelScaling factor value should be **1.596,** for 150arcsec/sec, the AzcmdvelScaling factor value is **15.96** and for **30 deg/min**, the AzcmdvelScaling factor value is **190.**

13.10) For Azcmdvel Scaling factor = 1.596, Plot between Act Vel (cts/sec) and Time(Sec) is given below.



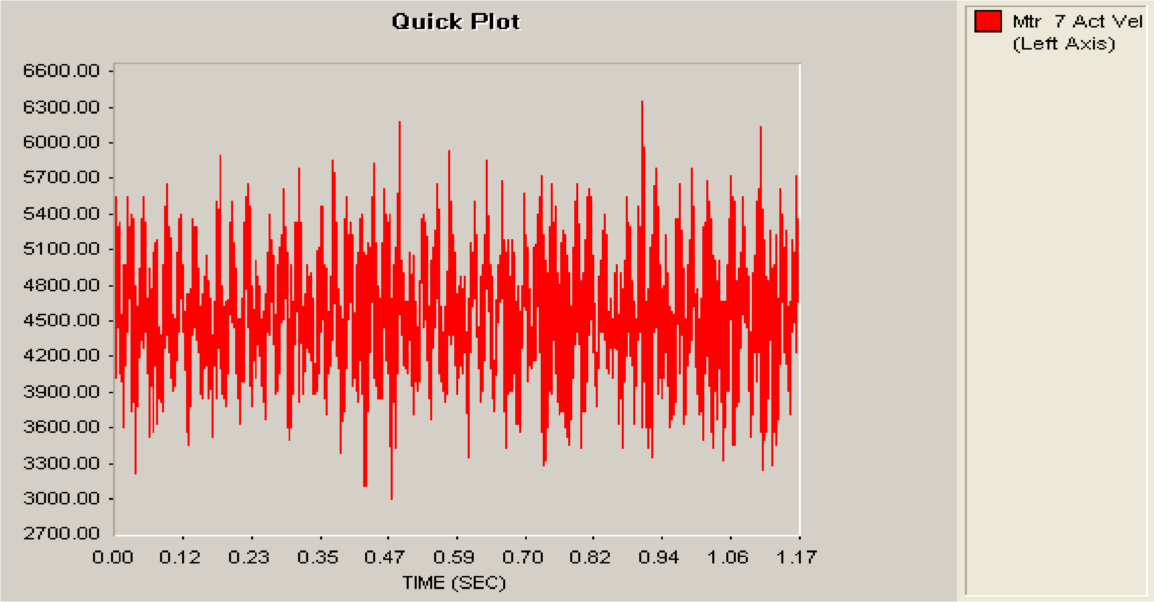
13.11) ActVel of Encoder is 36 cnts which corresponds to approximately**15 arcsec/sec** (36*0.4=14.4).

13.7) For Azcmdvel Scaling factor = 15.96, Plot between Act Vel (cts/sec) and Time(Sec) is given below.



13.8) ActVel of Encoder is 370 cnts which corresponds to approximately**150 arcsec/sec** (370*0.4=148).

13.9) For Azcmdvel Scaling factor = 190, Plot between Act Vel (cts/sec) and Time(Sec) is given below.



13.10) ActVel of Encoder is 1800 cnts which corresponds to approximately **30deg /min** (1800 *60/ 3600= 30).

# 14. Running Antenna with PMAC configured in Velocity loop-Elevation Axis

14.1) For making this PMAC PID filter as velocity loop in EL axis, we have to make position Feedback needs to be zero. For that we are taking load encoder feedback to #8 Channel, so while closing velocity loop, Position feedback will remain open at Channel #6 and position Feedback will be zero. At Ch #8 we can get encoder Data, which we can used for plotting data between Actual Velocity (arcsec/sec) and Time (sec).

14.2) Here Analog input is given to Ch#8 ADC channel of PMAC which address is given By M5064-> y: $7811D, 8, 16, S. Means Analog value will be stored in Signed 16 Bits Form. This value will scaled and stored in Hand-wheel Position register (M667).So M667 is given by

**M667= 32*96*M5064* ELcmdvelScaling factor/ 32767**.

14.3) Go to "PEWIN PRO32 PRO "and download the following PLC0 program for EL axis velocity loop.

```
#include "Tp2mvar.pmc"

close
endg
del gat

i7036,4,10=1                    ; true DAC Output
i300,4,100=1                    ; activate axis
i322,4,100=15                   ; slow default speed
i319,4,100=0.2                  ; higher acc + dec
i323,4,100=5                    ; homing speed
i324,4,100=$820001              ; no limit switches

i330,4,100=120000               ; PID Settings
i331,4,100=1050
i332,4,100=1050
i333,4,100=10000
i334,4,100=1

i369,4,100=16384                ; 10V diffential DAC Output


; #3 encoder is EL1 encoder and needs an oppoiste counting
; direction since the motor is mounted in the opposite orientation
; output of the BLC is hardwired in the opposite way (Servostar input)
```

```
i7030=3
; definitions for the analog input reading
I7106 = $1FFFFF                              ; ADC strobe word
M5064->Y:$7811D,8,16,s                       ; ch8 A-D channel

; ------------------------------------------------------
; PLCC0 real-time task for torque offset and active damping

; standard EL position/speed control loop at axis 6
; control output EL of axis 6 distributed to axes 3 and 4
; adding a torque offset
; axes 3 and 4 must be activated via command o0
; when killing axes 3 and 4, torqueOffset must be reset to 0

#define ELvelocityLoad       M674    ; filtered (unfiltered is M666)
#define ELvelocityMotor1      M374    ; filtered (unfiltered is M366)
#define ELvelocityMotor2      M474    ; filtered (unfiltered is M466)
#define ELtorque1            M379
#define ELtorque2            M479
#define ELdesTorque           M668
#define ELcmdPos              M661
#define ELactPos             M662

#define ELposError            P11      ; position control deviation
#define ELFRICTION_Offset     P12      ;
#define ELD1                 P13      ; damping coefficient 1
#define ELD2                 P15      ; damping coefficient 2
#define ELGR                  P16      ; gear ratio
#define ELMAX_TORQUE          P17      ; Nm scaled to 16 bit integer
#define ELTORQUE_OFFSET       P18      ; Nm scaled to 16 bit integer
#define ELcmdVelScaling               P19

ELD1          = 0
ELD2          = 0
ELGR           = 8.64257
ELMAX_TORQUE    = 32768
ELTORQUE_OFFSET = 1000
ELcmdVelScaling = 180

I5 = 3                          ; PLC program control enabled
I8 = 0                          ; PLCC 0 called every sample
 ; motor encoders used for velocity feedback
I8008 = $E00100     ; sum of motor 1 and 2 encoders written into
; motor 5 velocity feedback register
```

```
I8009 = $E00302     ; sum of motor 3 and 4 encoders written into
; motor 6 velocity feedback register

i600 = 1
i603 = $350B
i604 = $350A
i606 = 1                                ; enable master encoder (handwheel) in order
; for setting the desired vel input signal via
; m667 (scaled by 1/(32*i607)
i607 = 96
i608 = 96
I609 = 48                   ; motor 6 velocity scaling factor
; see scaling calculation of 21.04.09
; of motor 3 and 4
; considering the different resolution of motor and load
i630 = 1000000                  ; PID Settings
i631 = 128
i632 = 0
i633 = 0
i634 = 1
i638 = 0
i639 = 0
i619 = 0.0002
i622 = 1
i623 = 1
i624 = $20001
i668 = 0                    ; Friction FF term (not checked yet)
i611 = 0                    ; disable following Error
i612 = 0

OPEN PLC 0 CLEAR

; analog input reading and scaling
m667 = 96 * 32 * m5064 * ELcmdVelScaling / 32767

; torque offset

If (ELdesTorque < 0)
  ELtorque2 = ELdesTorque/2 - ELTORQUE_OFFSET;
  If (ELtorque2 < -ELMAX_TORQUE/2)
    ELtorque2 = -ELMAX_TORQUE/2
  EndIf
```

```
ELtorque1 = ELdesTorque - ELtorque2
Else
  ELtorque1 = ELdesTorque/2 + ELTORQUE_OFFSET;
  If (ELtorque1 > ELMAX_TORQUE/2)
    ELtorque1 = ELMAX_TORQUE/2
  EndIf
  ELtorque2 = ELdesTorque - ELtorque1
EndIf

; saturation

If (ELtorque1 > ELMAX_TORQUE/2)
  ELtorque1 = ELMAX_TORQUE/2
EndIf

If (ELtorque1 < -ELMAX_TORQUE/2)
  ELtorque1 = -ELMAX_TORQUE/2
EndIf

If (ELtorque2 > ELMAX_TORQUE/2)
  ELtorque2 = ELMAX_TORQUE/2
EndIf

If (ELtorque2 < -ELMAX_TORQUE/2)
  ELtorque2 = -ELMAX_TORQUE/2
EndIf


CLOSE   ; PLC 0
```

14.4) For running Antenna with 2**0 deg/min**, the ELcmdvelScaling factor value is **180.**

## 15. Interfacing Existing SSC (Station Servo Computer) as Position Loop with PMAC as Velocity Loop.

15.1) SSC (Station Servo Computer) will have Position Loop.

15.2) Analog Input (+/- 10V) from Analog card of SSC will connect to ANA I/O ch #7 of PMAC for AZ axis and Analog Input (+/- 10V) from Analog card of SSC will connect to ANA I/O ch #8 of PMAC for EL axis. The wiring diagram between SSC and RFI/EMC RACK for Analog i/p to PMAC as given below.

| F4 (FRC Connector) In SSC | BLDC Rack rear side 15 pin D type(male) | ANA I/O CH #7 of PMAC | ANA I/O CH #8 of PMAC |
|---|---|---|---|
| 19(AZ error+) | 9 | 6 | |
| 20(AZ error-) | 10 | 2 | |
| 21(EL error+) | 11 | | 6 |
| 22(EL error-) | 12 | | 2 |

15.3) Here PMAC will have Velocity Loop.

**Positioning Command to Azimuth Axis:-**

### 15. a) Running antenna with 150arcsec/sec speed by giving Position command of 10deg rotation from NEWSMU.

15. a.1) First Configure PMAC as Velocity Loop with AzcmdvelScaling factor =**15.96.** So when Total 10V analog input is giving, antenna will rotate with 150 arcsec/sec.

15. a.2) Let antenna is in 0 deg Position. Now give Position Command of 10 deg from NEWSMU to SSC. Then from Analog voltage from Analog card will be 10V. This Analog output from Analog card connected to PMAC ANA I/O ch #7. So Antenna will rotate with 150 arcsec/sec speed from 0 deg angle to 10 deg angle.

15.a.3) The table related first overshoot, second undershoot and Steady state error as given below

| S.No | Actual Angle | Target angle | First overshoot | Second undershoot | Steady state error |
|---|---|---|---|---|---|
| 1 | 002:00:22 | 012:00:00 | 012:10:00 | 012:06:00 | 012:00:20 |
| 2 | 012:49:00 | 022:50:00 | 022:55:00 | 022:48:00 | 022:50:06 |

### 15.b) Running antenna with 30 deg/min speed by giving Position command of 36deg rotation from NEWSMU.

15. b.1) First Configure PMAC as Velocity Loop with AzcmdvelScaling factor =**190.** So when Total 10V analog input is giving, antenna will rotate with 30 deg/min.

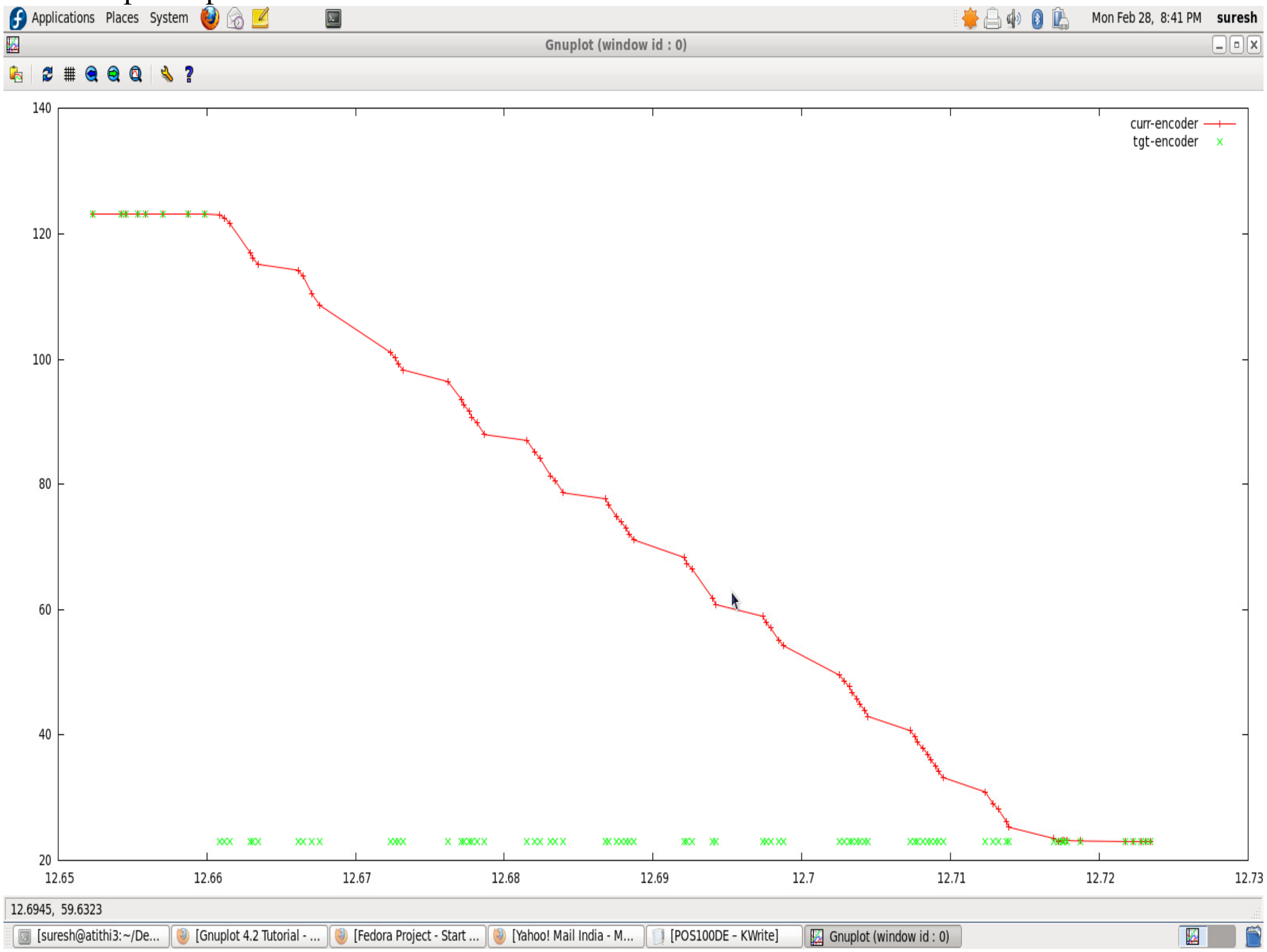
15.b.2) Let antenna is in 0 deg Position. Now give Position Command of 36 deg from NEWSMU to SSC. Then from Analog voltage from Analog card will be 10V. This Analog output from Analog card connected to PMAC ANA I/O ch #7. So Antenna will rotate with 30 deg/min speed from 0 deg angle to 36 deg angle.

15.b.3) The Table related first overshoot, second undershoot and Steady state error as given below

| S.No | Actual Angle | Target angle | First overshoot | Second undershoot | Steady state | Error (Target – Actual Position) |
|---|---|---|---|---|---|---|
| 1 | 000:00:00 | 036:00:00 | 036:35:52 | 036:00:09 | 036:00:09 | 000:00:09 |
| 2 | 026:38:20 | 036:00:00 | 036:12:07 | 035:59:47 | 035:59:57 | 000:00:13 |

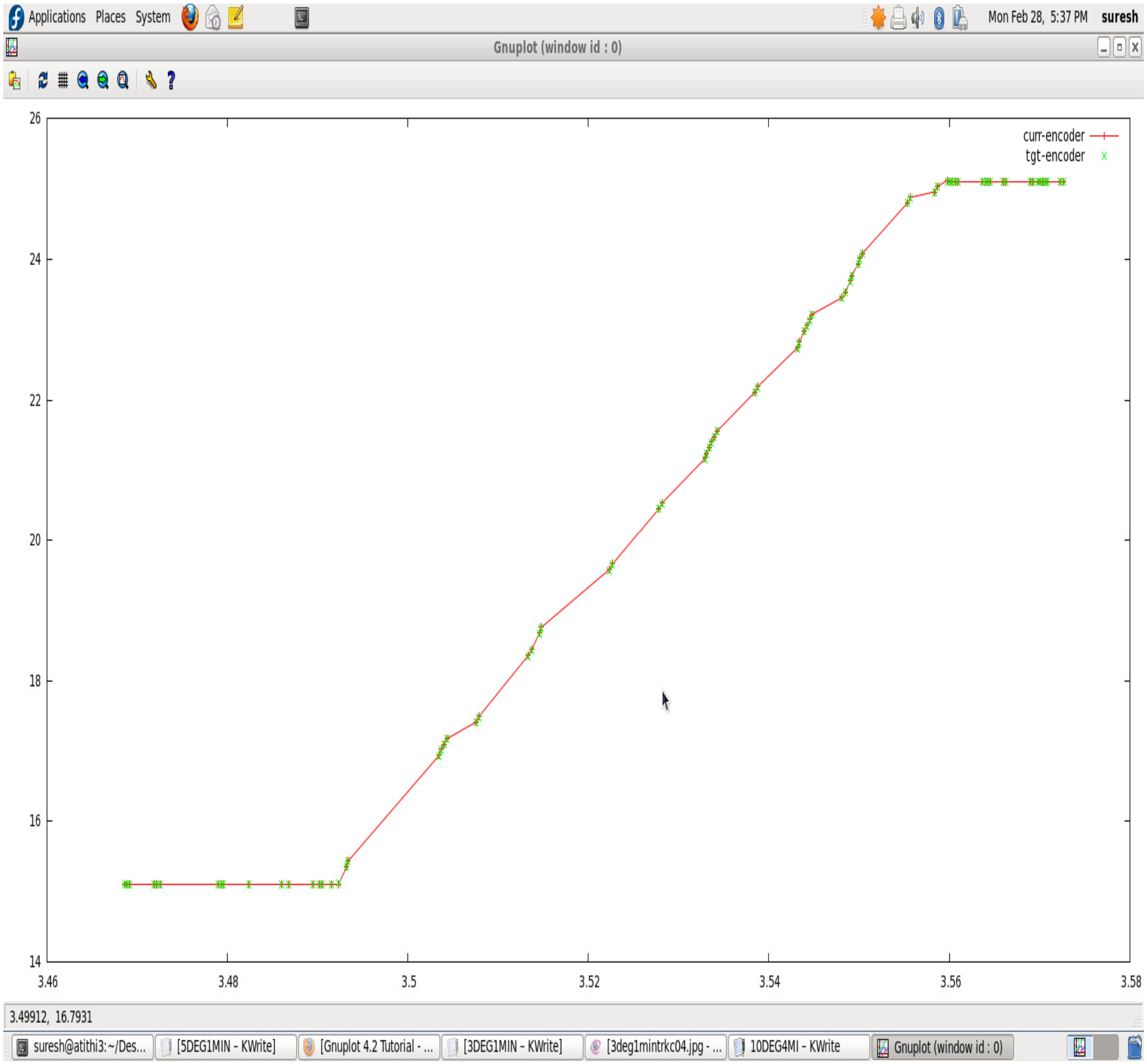
## 15.c) Running antenna with 30 deg/min speed by giving Position command of 100 deg rotation from NEWSMU.

When positioning command of 100 deg from Newsmu has given, the following is the reponse plot.



**Tracking Command to Azimuth Axis:-**

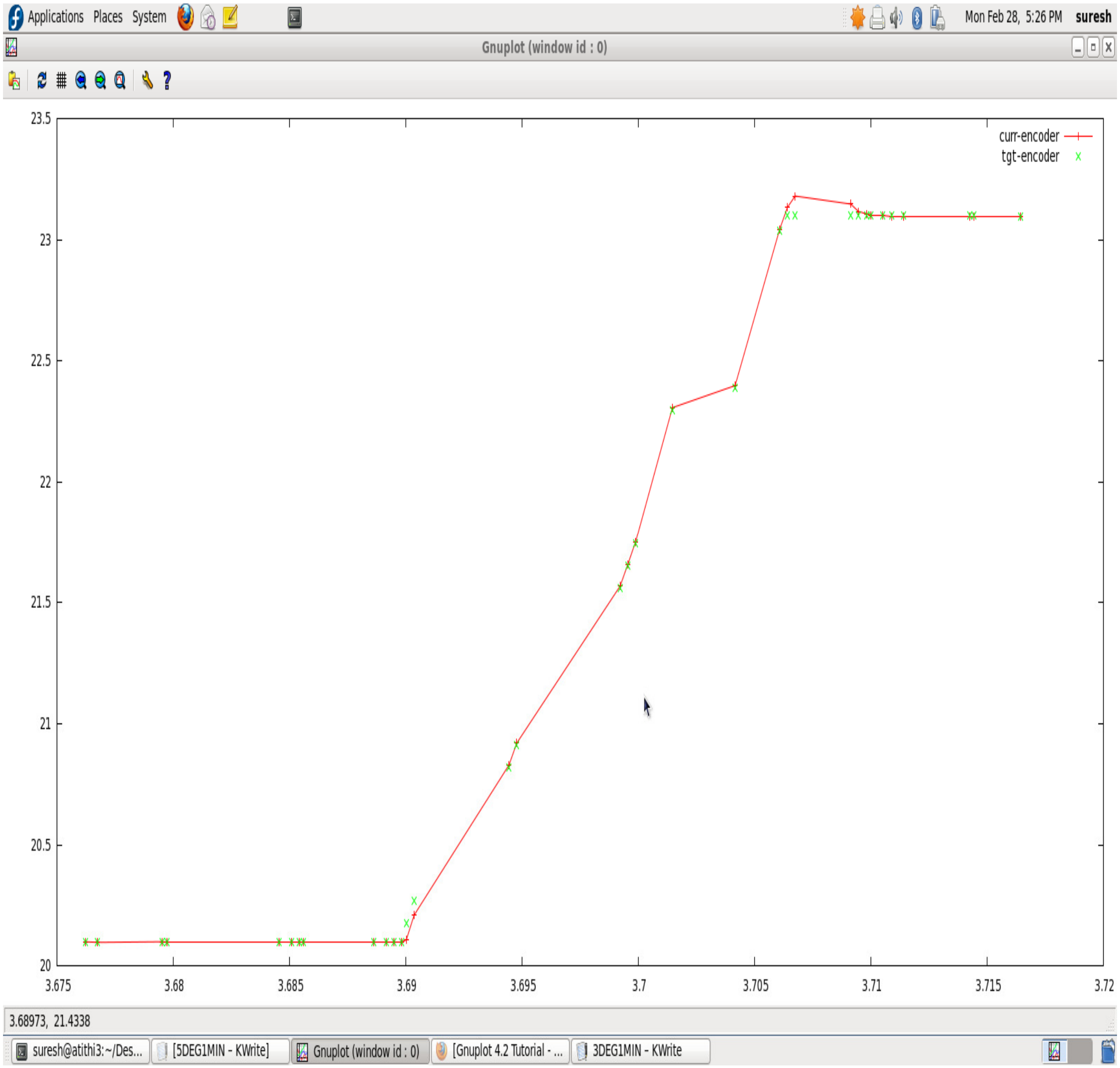
10 deg per 4 min track:-



X-axis:- float time (3.49225 float = 03:29:25 hrs and 3.55972 float = 03:33:31)
Y-axis:- Position in deg( 15 deg to 25 deg).

| Time in hrs | Position in deg |
|---|---|
| 3:29:25<br>3:33:31 | 15.1216<br>25.1007 |
| Approximately = 4 min. | Approximately = 10 deg. |

3 deg per 1 min:-



X-axis:- Time Float (3.69003 float = 03:41:21 hrs and 3.70677 float = 03:42:24)
Y-axis:- Position in deg( 20 deg to 23 deg).

| Time in hrs | Position in deg |
|---|---|
| 3:41:21<br>3:42:24 | 20.0980<br>23.1743 |
| Approximately = 1 min. | Approximately = 3 deg. |

**Positioning Command to Elevation Axis: -** Give Positioning command like Azimuth axis as described above.

**Tracking Command to Elevation Axis: -** Give Tracking command like Azimuth axis as described above.

## 16. Monitoring of Motor currents and speeds (Resolver) :-

16.1) For obtaining Analog outs (currents and tacho) from SS712, Have used “X3C Connector of POS I/O Monitor card”.

16.2) The following wiring diagram is between “F4 (FRC connector) of SSC to “X3C connector of SS712 -1, 2, 3, 4” for getting Current and tacho feedback.

| F4 (FRC connector of SSC) | SS712 -1 (AZ-1) | SS712-2 (AZ-2) | SS712-3 (EL-1) | SS712-4 (EL-2) |
|---|---|---|---|---|
| 3 (AZM1-CR) | 19 pin of X3C | | | |
| 4(AZM2-CR) | | 19 pin of X3C | | |
| 5(ELM1-CR) | | | 19 pin of X3C | |
| 6(ELM2-CR) | | | | 19 pin of X3C |
| 7(AZTacho-1) | 17 pin of X3C | | | |
| 8(AZTacho-2) | | 17 pin of X3C | | |
| 9(ELTacho-1) | | | 17 pin of X3C | |
| 10(ELTacho-2) | | | | 17 pin of X3C |

Note:-

1. 18 and 20 pin of X3C connector must be grounded.
2. DriveGUI setup software should be “Version 2.20 build 0004” and firmware should be 5.15 using upgrade tool V3.15.
3. In SS712 setting change the following parameters :-

   3.1) ANOUT1 =1 (for V-act)
   3.2) ANOUT2=2 (for I-act)
   3.3) AN1TRIG= XXXX. (for scaling Analog out1)
   3.4) AN2TRIG = XXXX (for scaling Analog out2)
   3.5) ANOFF1 = XXXX ( for adding offset to Analog out1)
   3.6) ANOFF2 = XXXX ( for adding offset to Analog out2)

## 17. Step Response of Velocity Loop in Azimuth :-

17.1) Connect AZ encoder to ENC5 of PMAC instead of connecting to SSC.
17.2) Add Following commands in PLC0 program for getting Encoder data to #7 axis.

```
i700=1
i703=$3505
```

17.3) Keep analog input voltage at 0V from Power supply (put the SW in OFF) before beginning the test. This analog i/p connects to ANA I/O Ch #7 of PMAC.
17.4) Go to File and Download “AZ_Velocity_PLC.pmc” PLC0 program in PEWIN32 PRO.

**Procedure:**

First open ‘PMAC PLOT PRO’ from toolbar for plotting the response

Click detail plot,
|
Keep gather period = 10
|
Go to
Item to gather,
|
Scaling and Processing,
|
Items to plot ---- define gather buffer
|
Begin gathering
|
Adjust analog i/p to ADC = 200mV and switch ON the ADC I/p supply
|
End gather after 10000ms
|
Upload data
|
And plot data.

In order to get multiple step response switch the power supply on / off several times after ‘begin gather’ and before ‘end gather’.

The following Commands are to be given in TERMINAL window before giving STEP Input

```
Enable plc0,
#1o0 #2o0 #5o0 #7o0        ; enable channels 1, 2, 5 and 7,
 #5j/                      ;   close Channel 5,
```

**Example illustrating the procedure**

17.5) Keep P9 (CmdVel Scaling factor) =190 and keep analog input 200mV from Power supply.

17.6) Procedure for Step response:-

17.6.1) Go to **PMAC Plot Pro**

17.6.2) Detail Plot

17.6.3) Press →Item to Gather→ Take Gather Period as 10
→Take Source1 for M5063 (Command Vel Signal)
→Take Source2 for M762 (Actual Position) → OK

17.6.4) Press → Scaling and Processing → Edit Source1 → Change Item Name as M5063 → Scale factor as 0.00055→ differentiate as none→OK

(Here the default scale of 0.000326 is changed to 0.00055 to increase the Response to match to the Step input. )

17.6.4) Edit Source2→ Change Item Name as M762 → Scale factor as 0.000326 differentiate as Once(Velocity)→OK→ OK

(M762 is Actual Position of Load Encoder, but we need Actual Velocity. So we are Differentiating once to Actual Position for getting Actual Velocity)

- Press → Items to plot → Edit → Horizontal Axis as Time
→ Left Vertical Axis as M5063 and M762
→ OK
- Press **Define Gather Buffer**
- Press **Begin Gathering**
- After 2000ms switch on 5V analog Input Power supply.
- After 15000ms Press **End Gathering**.
- Press **Upload Data**.
- Press **Plot Data**

**Note:-** The above procedure of “Step response of Velocity loop” will repeat for EL axis, except Analog i/p to ANA I/O Ch#8.

## 18. Sine Sweep Response of Velocity Loop in Azimuth:-

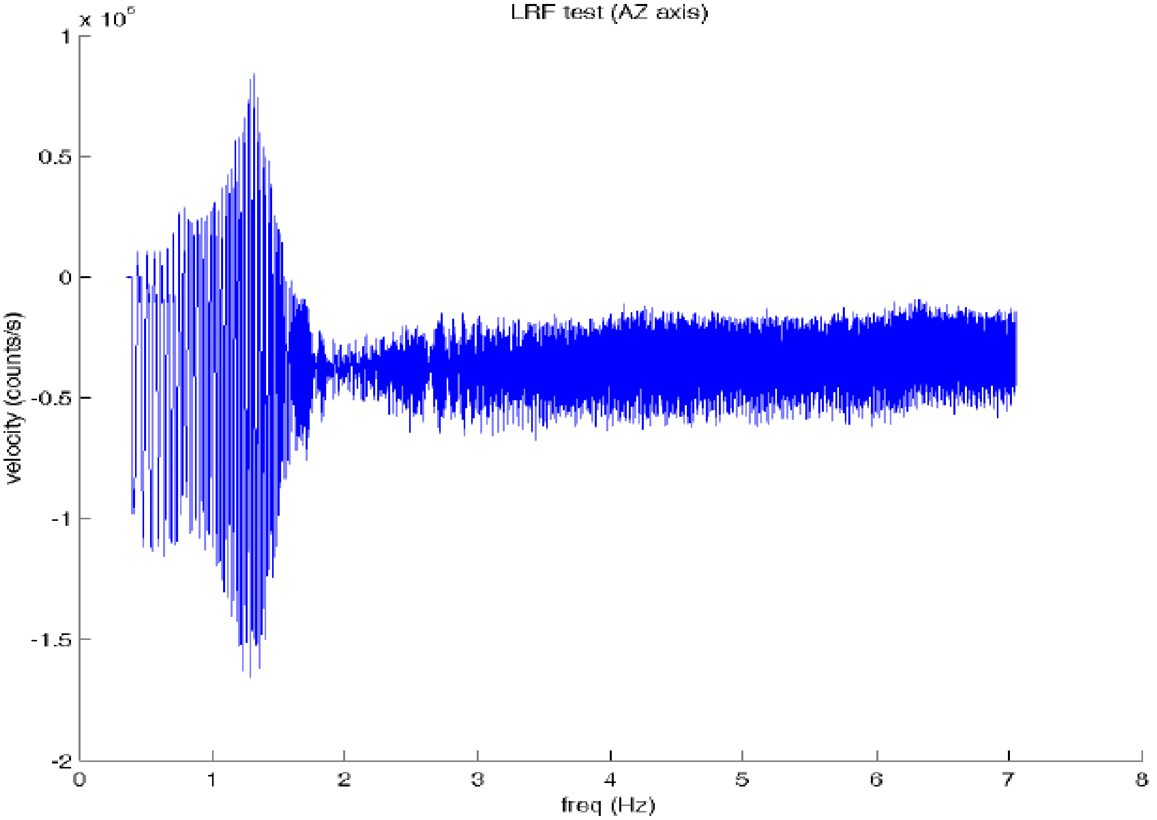
18.1) For Sine Sweep response, the above step response will be repeated except i/p to ANA I/O Ch #7 of PMAC is Sine Sweep from Function generator.

18.2) Sine Sweep from Function generator setting will be as follows

| | |
|---|---|
| 0.1 Hz to 15 Hz | - frequency range |
| 500s | - Sweep Time |
| 200mV | - Amplitude |

18.3) The following plot is Sine Sweep response (0Hz to 7Hz frequency range, 290s Sampling time and gather period 100 cycles = 44 ms) of Velocity loop in AZ axis.

18.4) The Resonant Frequency is approximately = 1.24Hz.



# 19. Installation of " POS I/O – Monitor" in SS712:-

19.1) SS712 Servostars which we got from Maccon are not having Monitor Analog out option, only we can get Position feedback (resolver) to PMAC (9 pin D type male). It means SS712 is having only "POS- I/O "Card in Slot-2 and Slot3 is empty.

19.2) For installing "POS I/O-Monitor" card in Slot2, remove "POS I/O" from Slot2.

19.3) Install "POS I/O- Monitor" card in Slot2 and "POS I/O" card in Slot3.

19.4) After installing New card in SS712 will give F20 (Slot error) error.

19.5) For removing F20 error, give following commands in Terminal Window in "DriveGUI"

19.6.1) CALCPOSIO
19.6.2) SAVE
19.6.3) COLDSTART

19.6) Now F20 error will go.

19.7) SS712 servostar have come with Firmware 5.04 version. For upgrading Firmware to 5.15 version, download " firmware 5.15 and Upgrade Tool V3.15" from Kollmorgen website.

19.8) Go to Upgarde tool and add firmware " 5.15 version" and download to SS712.

# 20. Annexure-A

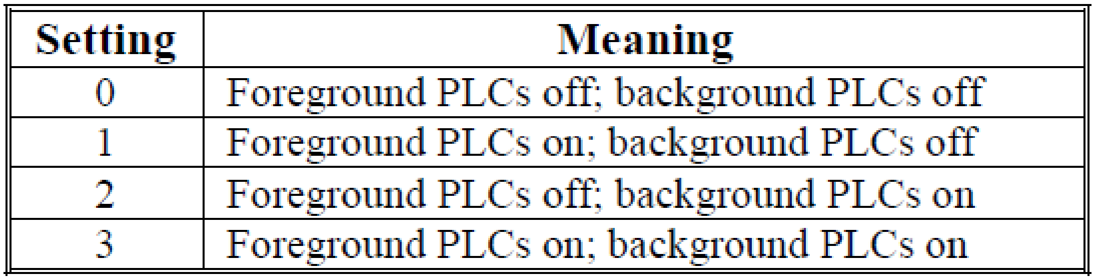
```
#include "Tp2mvar.pmc"

close
endg
del gat

i7016, 2,10=1                          ; true DAC Output
i100, 2,100=1                          ; activate axis
i122, 2,100=15                                 ; slow default speed
i119, 2,100=0.2                        ; higher acc + dec
i123, 2,100=5                          ; homing speed
i124, 2,100=$820001                    ; no limit switches
i130, 2,100=120000                     ; PID Settings
i131, 2,100=1050
i132, 2,100=1050
i133, 2,100=10000
i134, 2,100=1
i169, 2,100=16384                      ; 10V diffential DAC Output
; AZ load encoder setting (connected to ENC5 input)
i7110=7                                ; changing counting direction
i7010=3
i7020=3
i7030=3

; ---------------------------------------------------------
; PLCC0 real-time task for torque offset and active damping

; standard AZ position/speed control loop at axis 5
; control output AZ of axis 5 distributed to axes 1 and 2
; adding a torque offset
; axes 1 and 2 must be activated via command o0
; when killing axes 1 and 2, torqueOffset must be reset to 0

#define AZvelocityLoad        M574    ; filtered (unfiltered is M566)
#define AZvelocityMotor1      M174    ; filtered (unfiltered is M166)
#define AZvelocityMotor2      M274    ; filtered (unfiltered is M266)
#define AZtorque1           M179
#define AZtorque2           M279
#define AZdesTorque           M568
#define AZcmdPos             M561
#define AZactPos            M562
```

```
#define AZposError          P1    ; position control deviation
#define AZFRICTION_Offset    P2    ;
#define AZD1                P3    ; damping coefficient 1
#define AZD2                P5    ; damping coefficient 2
#define AZGR                P6    ; gear ratio
#define AZMAX_TORQUE          P7    ; Nm scaled to 16 bit integer
#define AZTORQUE_OFFSET       P8    ; Nm scaled to 16 bit integer

AZD1          = 0
AZD2          = 0
AZGR          = 8.64257
AZMAX_TORQUE    = 32768
AZTORQUE_OFFSET = 1000

I5 = 3                        ; PLC program control enabled
I8 = 0                        ; PLCC 0 called every sample

; motor encoders used for velocity feedback
I8008 = $E00100    ; sum of motor 1 and 2 encoders written into
; motor 5 velocity feedback register

i500 = 1
i503 = $3505
i504 = $3509
i508 = 192
I509 = 1          ; motor 5 velocity scaling factor
; see scaling calculation of 17.04.09
; of motor 1 and 2
; considering the different resolution of motor and load

i530 = 12000   ; PID Settings (Kp also depends on which filter is using)
i531 = 1000
i532 = 0
i533 = 0
i534 = 1

i538 =; this value depends on which filter is using that is (1000Hz notch or 100Hz lowpass)
i539 = ;this value depends on which filter is using that is (1000Hz notch or 100Hz lowpass)

i519 = 0.0002
i522 = 1
i523 = 1
i524 = $20001
i568 = 400                  ; Friction FF term
```

```
OPEN PLC 0 CLEAR

; ====================================================================
; AZ PART
; ====================================================================

AZposError = AZcmdPos - AZactPos;

; torque offset

If (AZdesTorque < 0)
AZtorque2 = AZdesTorque/2 - AZTORQUE_OFFSET;
If (AZtorque2 < -AZMAX_TORQUE/2)
AZtorque2 = -AZMAX_TORQUE/2
EndIf
AZtorque1 = AZdesTorque - AZtorque2
Else
AZtorque1 = AZdesTorque/2 + AZTORQUE_OFFSET;
If (AZtorque1 > AZMAX_TORQUE/2)
AZtorque1 = AZMAX_TORQUE/2
EndIf
AZtorque2 = AZdesTorque - AZtorque1
EndIf

; saturation

If (AZtorque1 > AZMAX_TORQUE/2)
  AZtorque1 = AZMAX_TORQUE/2
EndIf
If (AZtorque1 < -AZMAX_TORQUE/2)
  AZtorque1 = -AZMAX_TORQUE/2
EndIf
If (AZtorque2 > AZMAX_TORQUE/2)
  AZtorque2 = AZMAX_TORQUE/2
EndIf
If (AZtorque2 < -AZMAX_TORQUE/2)
  AZtorque2 = -AZMAX_TORQUE/2
EndIf

CLOSE   ; PLC 0
```

# 21. Annexure-B(Important I-Variables):-

21.1) **I5 (PLC Program Control):-**
I5 controls which PLC programs may be enabled. There are two types of PLC programs: the foreground programs (PLC 0 and PLCC 0), which operate at the end of servo interrupt calculations, with a repetition rate determined by I8 (PLC 0 and PLCC 0 should be used only for time-critical tasks and should be short); and the background programs (PLC 1 to PLC 31, PLCC 1 to PLCC 31) which cycle repeatedly in background as time allows.



| Setting | Meaning |
|---|---|
| 0 | Foreground PLCs off; background PLCs off |
| 1 | Foreground PLCs on; background PLCs off |
| 2 | Foreground PLCs off; background PLCs on |
| 3 | Foreground PLCs on; background PLCs on |

After assigning I5 for PLC , have to give ENABLE PLC command in terminal window of PEWIN32PRO for enable PLC and Disable PLC is for disabling plcs.

21.2) **I8(Real Time Interrupt period):-**
I8 controls how often certain time-critical tasks, such as PLC 0, PLCC 0, and checking for motion program move planning, are performed. These tasks are performed every (I8+1) servo cycles, at priority level called the “real-time interrupt” (RTI). A value of 2 means that these tasks are performed after every third servo interrupt 3 means every fourth interrupt, and so on.

21.3) **Ixx01(Motor XX activation control):-**
Ixx00 determines whether Motor xx is de-activated (Ixx00 = 0) or activated (Ixx00 = 1).

21.4) **Ixx03(Motor XX Position Loop feedback address):-**
Ixx03 tells the Turbo PMAC where to look for its position feedback value to close the position loop for Motor xx. It contains the address of the register where the motor will read its position feedback value.

21.5) **Ixx04(Motor XX Velocity loop feedback address):-**
Ixx04 tells the Turbo PMAC where to look for its *position* feedback value to close the *velocity* loop for Motor xx. It contains the address of the register where the motor will read its position feedback value.

21.6) **Ixx05(Motor XX master position address):-**
Ixx05 specifies the address of the register for "master" position information of Motor xx for the position following, or electronic gearing, function. The position following function is only enabled if Ixx06 is set to 1 or 3.

21.7) **Ixx06(Motor XX Position following enable):-**
Ixx06 controls the position following function for Motor xx. It determines whether following is enabled or disabled.

21.8) **Ixx07(Motor XX Master Position Scaling factor):-**
Ixx07 controls with what scaling the master (hand wheel) register gets multiplied when extended into the full-length register.

21.9) **Ixx08(Motor XX Position scale factor):-**
Ixx08 specifies the multiplication scale factor for the internal position registers for Motor xx.Source position registers are multiplied by Ixx08 as the get extended into the full-length motor position registers. For most purposes, this is transparent to the user and Ixx08 does not need to be changed from the default. There are two reasons that the user might want to change this from the default value. First, because it is involved in the "gear ratio" of the position following function -- the ratio is Ixx07/Ixx08 – the value of Ixx08 might be changed (usually raised) to get a more precise ratio. The second reason to change this parameter (usually lowering it) is to prevent internal saturation at very high gains or count rates (velocity).

21.10) **Ixx09(Motor XX Velocity Scale factor):-**
Ixx09 specifies the multiplication scale factor for the internal actual velocity registers for Motor xx. Source position registers for the velocity loop are multiplied by Ixx09 before they are compared and used in the velocity loop. For most purposes, this is transparent to the user and Ixx09 does not need to be changed from the default.

21.11) **Ixx11(Motor XX fatal Following error limit):-**
Ixx11 sets the magnitude of the following error for Motor xx at which operation will shut down. When the magnitude of the following error exceeds Ixx11, Motor xx is disabled (killed).

21.12) **Ixx12(Motor XX warning Following error limit):-**
Ixx12 sets the magnitude of the following error for Motor xx at which a warning flag goes true. Setting this parameter to zero disables the warning following error limit function. If this parameter is set greater than the Ixx11 *fatal* following error limit, the warning status bit will never go true, because the fatal limit will disable the motor first.

21.13) **Ixx19(Motor XX Maximum Jog/Home Acceleration):-**
Ixx19 sets a limit to the commanded acceleration magnitude for jog and home moves, and for **RAPID**-mode programmed moves, of Motor xx.

21.14) **Ixx20(Motor XX Maximum Jog/Home Acceleration Time):-**
Ixx20 establishes the time spent in acceleration in a jogging, homing, or programmed **RAPID** mode move.

21.15) **Ixx21( Motor XX Jog/Home S-Curve time):-**
Ixx21 establishes the time spent in each “half” of the “S” for S-curve acceleration in a jogging, homing, or **RAPID**-mode move.

21.16) **Ixx22(Motor XX jog speed):-**
Ixx22 establishes the commanded speed of a jog move, or a programmed **RAPID**-mode move (if Ixx90=0) for Motor xx. Direction of the jog move is controlled by the jog command.

21.17) **Ixx23 (Motor XX home speed and direction):-**
Ixx23 establishes the commanded speed and direction of a homing-search move for Motor xx.

21.18) **Ixx24(Motor XX flag mode control):-**
Ixx24 is a set of 24 individual control bits – bits 0 to 23. Currently bits 0 and 11 to 23 are used.
In GMRT Brushless project , this value is $820001. For explanation of all bits , see Turbo SRM.

21.19) **Ixx29(Motor XX output):-**
When Turbo PMAC is not commutating Motor xx (Ixx01 Bit 0 = 0), Ixx29 serves as the offset for the single command output value, usually a DAC command.

21.20) **Ixx30(Motor XX PID proportional Gain):-**
Ixx30 provides a control output proportional to the position error (commanded position minus actual position) of Motor xx.

21.21) **Ixx31(Motor XX PID derivative Gain):-**
Ixx31 subtracts an amount from the control output proportional to the measured velocity of Motor xx. It acts effectively as an electronic damper. The higher Ixx31 is, the heavier the damping effect is.

21.22) **Ixx32(Motor XX PID Velocity Feed forward Gain):-**
Ixx32 adds an amount to the control output proportional to the desired velocity of Motor xx.

21.23) **Ixx33(Motor XX PID Integral Gain):-**
Ixx33 adds an amount to the control output proportional to the time integral of the position error for Motor xx.

21.24) **Ixx34(Motor XX Integral Mode):-**

Ixx34 controls when the position-error integrator is turned on. If it is 1, position error integration is performed only when Motor xx is not commanding a move (when desired velocity is zero). If it is 0, position error integration is performed all the time.

21.25) **Ixx35(Motor XX PID Acceleration Feed forward Gain):-**

Ixx35 adds an amount to the control output proportional to the desired acceleration for Motor xx.

21.26) **Ixx36(Motor XX PID Notch Filter coefficient N1):-**

Ixx36, along with Ixx37-Ixx39, is part of the 2nd-order "notch filter" for Motor xx, whose main purpose is to damp out a resonant mode in the motor/load dynamics. This filter can also be used as a low-pass filter and a velocity-loop integrator. This parameter can be set according to instructions in the Servo Loop Features section of the manual.

21.27) **Ixx37(Motor XX PID Notch Filter coefficient N2):-**

Ixx37 is part of the "notch filter" for Motor xx. See Ixx36 and the Servo Loop Features section of the manual for details.

21.28) **Ixx38(Motor XX PID Notch Filter coefficient D1):-**

Ixx38 is part of the "notch filter" for Motor xx. See Ixx36 and the Servo Loop Features section of the manual for details.

21.29) **Ixx39(Motor XX PID Notch Filter coefficient D2):-**

Ixx39 is part of the "notch filter" for Motor xx. See Ixx36 and the Servo Loop Features section of the manual for details.

21.30) **Ixx68(Motor XX friction Feed forward):-**

Ixx68 adds a bias term to the servo loop output of Motor xx that is proportional to the *sign* of the commanded velocity. That is, if the commanded velocity is positive, Ixx68 is added to the output. If the commanded velocity is negative, Ixx68 is subtracted from the output. If the commanded velocity is zero, no value is added to or subtracted from the output.

21.31) **Ixx69(Motor XX Output command limit):-**

Ixx69 defines the magnitude of the largest output that can be sent from Turbo PMAC's PID position/velocity servo loop.

21.32) **I7m06( Servo IC m ADC strobe):-**

I7m06 controls the ADC strobe signal for machine interface channels on Servo IC m. The 24-bit word set by I7m06 is shifted out serially on the ADC_STROB lines, MSB first, one bit per DAC_CLK cycle starting on the rising edge of the phase clock. The value in the LSB is held until the next phase clock cycle.

21.33) **I7mn0(Servo IC m channel n Encoder/Timer decode Control):-**

I7mn0 controls how the input signal for Encoder n on a PMAC2-style Servo IC m is decoded into counts. As such, this defines the sign and magnitude of a "count". The following settings may be used to decode an input signal.

I7mn0 = 0: Pulse and direction CW
I7mn0 = 1: x1 quadrature decode CW
I7mn0 = 2: x2 quadrature decode CW
I7mn0 = 3: x4 quadrature decode CW
I7mn0 = 4: Pulse and direction CCW
I7mn0 = 5: x1 quadrature decode CCW
I7mn0 = 6: x2 quadrature decode CCW
I7mn0 = 7: x4 quadrature decode CCW
I7mn0 = 8: Internal pulse and direction
I7mn0 = 9: Not used
I7mn0 = 10: Not used
I7mn0 = 11: x6 hall-format decode CW*
I7mn0 = 12: MLDT pulse timer control
(Internal pulse resets timer; external pulse latches timer)
I7mn0 = 13: Not used
I7mn0 = 14: Not used
I7mn0 = 15: x6 hall-format decode CCW*

21.34) **I8000-I8191(Conversion Table setup lines):-**

I8000 to I8191 form the 192 setup lines of the Turbo PMAC's Encoder Conversion Table (ECT).The main purpose of the ECT is to provide a preprocessing of feedback and master data to prepare it for use by the servo loop. For detailed explanation of about ECT see Turbo PMAC.

## 22. Annexure-C(Setting up analog inputs):-

The Brick Motion Controller can be ordered with two or four 16-bit hi-resolution analog to digital converters. The Brick Motion Controller uses the Burr Brown ADS8361.When selected for bipolar mode, differential inputs allow the user to apply input voltages to ±10 volts(20Vp-p). To read the A/D data, the user needs to set the ADC strobe word for the second gate array to I7106 = $1FFFFF. Also, the user needs to create M-variable definitions that point to the ADC inputs (M-variables that are used are suggested ones) for channels 5 to 8.

**Bipolar**

The data received is a signed 16-bit number scaled from –10V to +10V (-32768cts to 32767cts).

```
M5061->Y:$78105,8,116,s ;ch5 A-D channel
M5062->Y:$7810D,8,16,s ;ch6 A-D channel
M5063->Y:$78115,8,16,s ;ch7 A-D channel
M5064->Y:$7811D,8,16,s ;ch8 A-D channel
```