

RRR-robot : instruction manual

Citation for published version (APA):

van Beek, A. M. (1998). *RRR-robot : instruction manual*. (DCT rapporten; Vol. 1998.012). Technische Universiteit Eindhoven.

Document status and date:

Published: 01/01/1998

Document Version:

Publisher's PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

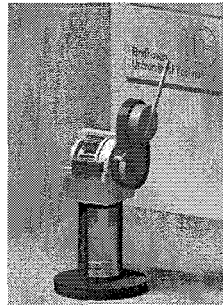
www.tue.nl/taverne

Take down policy

If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.



RRR-robot

Instruction manual

A.M. van Beek

WFW report 98.012

Trainee project

Supervisor: ir. L. Kodde

Faculty of Mechanical Engineering
Eindhoven University of Technology (TUE)
March 1998



Operational warnings

1. Do not operate the robot unsupervised, or without the emergency stop within reach. The motors can rotate at high speed with high torque; beware of the rotating radius of the robot.
2. Do not touch any part of the robot when the power is switched on. Dangerously high voltage is present at the location of brush terminals (shielded with plexiglas covers), and inside the octagonal shaped, silver colored housing.
3. Keep clear of the rotating radius of the robot while the servos are powered *and* enabled.
4. Do not exceed the following velocities in order to stay below the allowed moment for alternating load, DM60: 0.5 [rps], DM30: 1 [rps], and DM15: 1.5 [rps].
5. Do not operate the DM15 servo while the DM60 and/or DM30 are at a continuous standstill in order to prevent wear or damage to the power sliprings due to welding effects.
6. Do not operate the DM30 servo while the DM60 is at a continuous standstill in order to prevent wear or damage to the power sliprings due to welding effects
7. Ensure that the motor phases V_A , V_B , and V_C are connected correctly. If not, the encoder feedback-loop results in unstable behavior, i.e., the motor will start rotating at maximum velocity and with maximum torque in a direction opposite to the desired trajectory.
8. After modifying the Simulink controller, always perform a “test run” to see if error messages occur.
9. When performing an experiment, do not start the controller before the servos are powered and enabled; the introduced tracking error may cause a large and unwanted response.
10. Do not use the enable circuit as an alternative emergency stop. There is a considerable delay (approximately one second) before opening the circuit has effect.
11. Use the emergency stop if the control software or the operating system crashes; or if error messages occur after the Simulink controller has been started.
12. Use the emergency stop and perform a (hard) reboot of the PC containing the MultiQ I/O board, if the encoder read-out becomes discontinuous or otherwise erratic.
13. Do not apply more than 5.3 [V] to the (analog) inputs of the MultiQ I/O board to avoid overload. As a consequence, do not use any of the analog driver outputs without additional overload protection since a surge of 8 [V] occurs when the power is switched on.

14. Ensure that the power is switched off before opening the Driver cabinet. Dangerously high voltage is present inside this unit.
15. Use the base connection to move or lift the robot (approximate weight 90 [kg]). Do not use the octagonal shaped, silver colored housing to lift the robot; unless with extreme care to prevent damage to the spherical joint at the base of the robot.

Contents

Operational warnings	iii
Terminology	1
1 Introduction	3
1.1 Objective	3
1.2 Getting started	4
2 General description	7
3 Manipulator frame	11
3.1 Composition	11
3.2 Assembly and installation	12
3.3 Wiring	18
3.3.1 Power connections	18
3.3.2 Signal connections	20
3.4 Operation	23
3.5 Inertia parameters	23
4 Joint actuation System	25
4.1 Dynaserv motors	25
4.2 Driver cabinet	26
4.3 Dynaserv drivers	27
4.4 Dynaserv brakes	27
5 Control System	29
5.1 MultiQ plug-in and terminal board	29
5.2 External controls	30
5.3 Personal Computer	31
5.3.1 PC Hardware	31
5.3.2 PC software	31
6 Maintenance and inspection	33
A Addresses	35

B Components	37
B.1 Wincon software manual	38
B.2 MultiQ board manual	70
B.3 Manual: Dynaserv DD Servo-Actuator DM/SD series	91
B.4 Manual: BE-A/B Type Dynamic Brake	113
B.5 Cabinet manual	123
B.6 Power sliprings	133
B.7 Signal sliprings	135
B.8 Conceptual design	137

Terminology

When applicable, parts are numbered from the base to the end-effector, i.e., the stationary base is link 0 (black parts), the rotating octagonal silver housing is the top of link 1, link 2 is green, and the end-effector is link 3.

Table 1: Terminology

Short	Synonym	Description
A/D	AD(C)	Analog-to-Digital (Converter)
AC		Alternating Current
BDC	Brushless DC	Brushless Direct Current (motor)
BE60	Brake 1	Electro-mechanical brake BE1060B
BE30	Brake 2	Electro-mechanical brake BE1030B
CTD	GTD	“Centrale/Gemeenschappelijke Technische Dienst”
D/A	DA(C)	Digital-to-Analog (Converter)
DC		Direct Current
DM60	Motor/servo 1	Dynaserv Motor DM1060B50*1, max. torque 60 [Nm]
DM30	Motor/servo 2	Dynaserv Motor DM1030B50*1, max. torque 30 [Nm]
DM15	Motor/servo 3	Dynaserv Motor DM1015B50*1, max. torque 15 [Nm]
Driver cabinet		Shielded cabinet for drivers, brakes and line filters
MultiQ	PC plug-in board	board with ADC, DAC, encoder inputs, and digital I/O
PSR-1	Power Slipring 1	SM140-6 and -2 combined
PSR-2	Power Slipring 2	M-SM204-12
RTW	Real Time Workshop	Matlab toolbox to generate C-code
SSR-1	Signal Slipring 1	Capsule AC-267 modified
SSR-2	Signal Slipring 2	Capsule AC-267 modified
SSR-3	Signal Slipring 3	Litton 12-ring capsule
SD60	Driver 1	Driver SD1060B52-2, driving DM60
SD30	Driver 2	Driver SD1030B52-2, driving DM30
SD15	Driver 3	Driver SD1015B52-2, driving DM15
Terminal board		Board with connectors to the internal MultiQ board
Watcom	Watcom C/C++	Watcom C/C++ compiler version 10.6
Wincon	WinCon V2.0	Software to interact with real-time running C-code

Chapter 1

Introduction

1.1 Objective

The RRR-robot is designed as a manipulator-like system (with three rotational degrees of freedom), to test a variety of advanced nonlinear control strategies. Since for high-speed tracking of complex trajectories, Coriolis and centrifugal torques form an essential part of the occurring nonlinear effects, the main requirement of this robot is to highlight these velocity dependent torques. This has led to two important features of the RRR-robot:

- the use of sliprings to facilitate unconstrained rotation of each link, and
- the use of direct-drive servos.

In addition, the system has a Simulink based control interface for fast implementation of various control strategies.

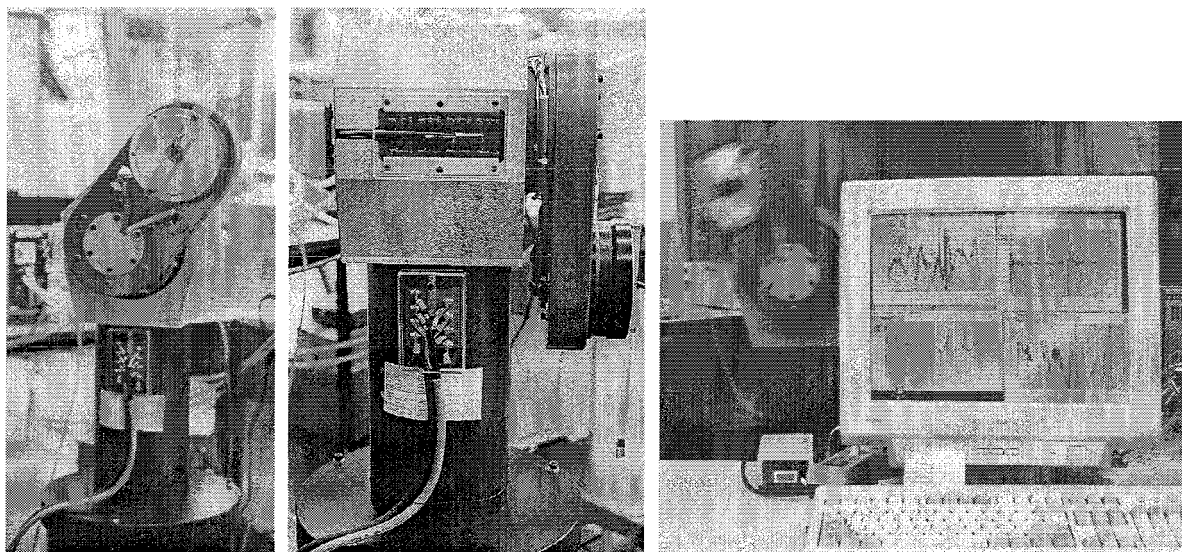


Figure 1.1: RRR-robot pictures, notice the brush block terminals to connect the two power sliprings.

The aim of this manual is to describe the composition and use of the RRR-robot as a whole. With regard to the individual components, only key parameters, and non-documented features

(from personal communications and experiences) are listed. For more detailed information, the reader is referred to the included component manuals in Appendix B. Before actually operating the robot, the user should also read the Wincon software manual in Appendix B.1.

1.2 Getting started

The procedure listed below can be used to get familiar with the basic operation of the RRR-robot and its control system. However, it is recommended to read the remaining of this manual and Appendix B.1 first in order to understand the entire background.

1. Switch on the PC.
2. Start Windows. Matlab should start automatically (in the directory `c:\rrr\demo\`).
3. Inside the Matlab Command Window type: `demo`. The Simulink window with the controller should open.
4. Double click on the button `init` inside the Simulink window. The matrices K_p and K_d should be read into the Matlab workspace.
5. Select **Generate and Built Realtime Code** from the Code menu of the Simulink window. A DOS window should open, displaying the compilation procedure. This should be completed without errors (warnings may be disregarded). After successful compilation, close the DOS window.



Do not yet enable the servos.

6. Select **Start** from the Simulation menu of the Simulink window. Now Wincon should start automatically and the controller should run “dry” without error messages.



After modifying the Simulink scheme, always perform a “test run” with the controller before enabling the servos. If error messages occur, recompile the controller.

7. Stop the controller by pressing the traffic light inside the Wincon window.
8. Use the File menu of the Wincon window to load `demo`. Now a number of plot windows should appear (if not, push the Scopes/Workspace button, and select the desired plot variables).
9. Make sure the controller is not running (the traffic light is red, the time display is frozen).
10. Switch on the central power of the Cabinet driver (if necessary pull out the emergency button on the cabinet).
11. Enable the servos by switching on the separate power supply.
12. Put one hand on the emergency stop.



Use the emergency stop if error messages occur after the Simulink controller has been started; if the control software or the operation system crashes; or if the encoder read-out becomes discontinuous or otherwise erratic.

-
13. Start the controller (push the traffic light or select start form the Simulink window). After 1 second delay, each joint should start moving at a constant velocity.

Chapter 2

General description

The RRR-robot is actuated by three brushless direct-drive servos with maximum torques of 60, 30 and 15 [Nm]. Each motor has its own driver to generate the driving stator phases (both are shown in Fig. 2.1 on the right). The so-called *Dynaserv* servo is of an outer-rotor type with internal encoder, and internal bearings, providing direct coupling between the outside housing and the attached link. Basically, the robot is constructed by simply combining these servos, i.e., joints and the links in a chain structure.

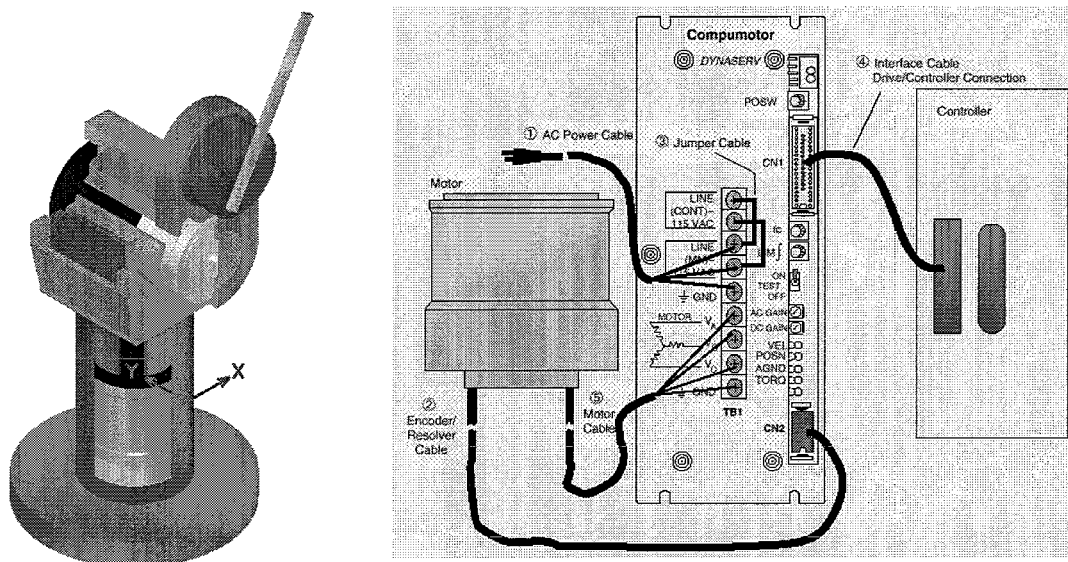


Figure 2.1: On the left, manipulator frame (servos, sliprings, and other frame parts); and on the right, the connections between one *Dynaserv* motor-driver pair.

To facilitate unconstrained rotation of all joints – the connection to and from the DM30 and DM15, shown in Fig. 2.1 by cable ② and ⑤ – are implemented using (power and signal) sliprings. The DM60 is connected directly from the base.

Servos, sliprings, and other frame parts (e.g., links) together constitute the *Manipulator frame* (shown in Fig. 2.1 on the left). The assembly of the manipulator frame, and all required wiring is described in Chapter 3.

To bring the DM60 and DM30 (with considerable inertia) to a controlled stop, electro-mechanical brakes are placed between motor and driver to short the motor coil. Line filters are used to filter out external disturbances on the power supply (cable ①). Drivers, brakes and line filters are mounted in a shielded *Driver cabinet*. This cabinet has a central power supply which can be switched of by an external emergency stop. Servos, drivers, brakes, and line filters constitute the *joint actuation system* described in Chapter 4.

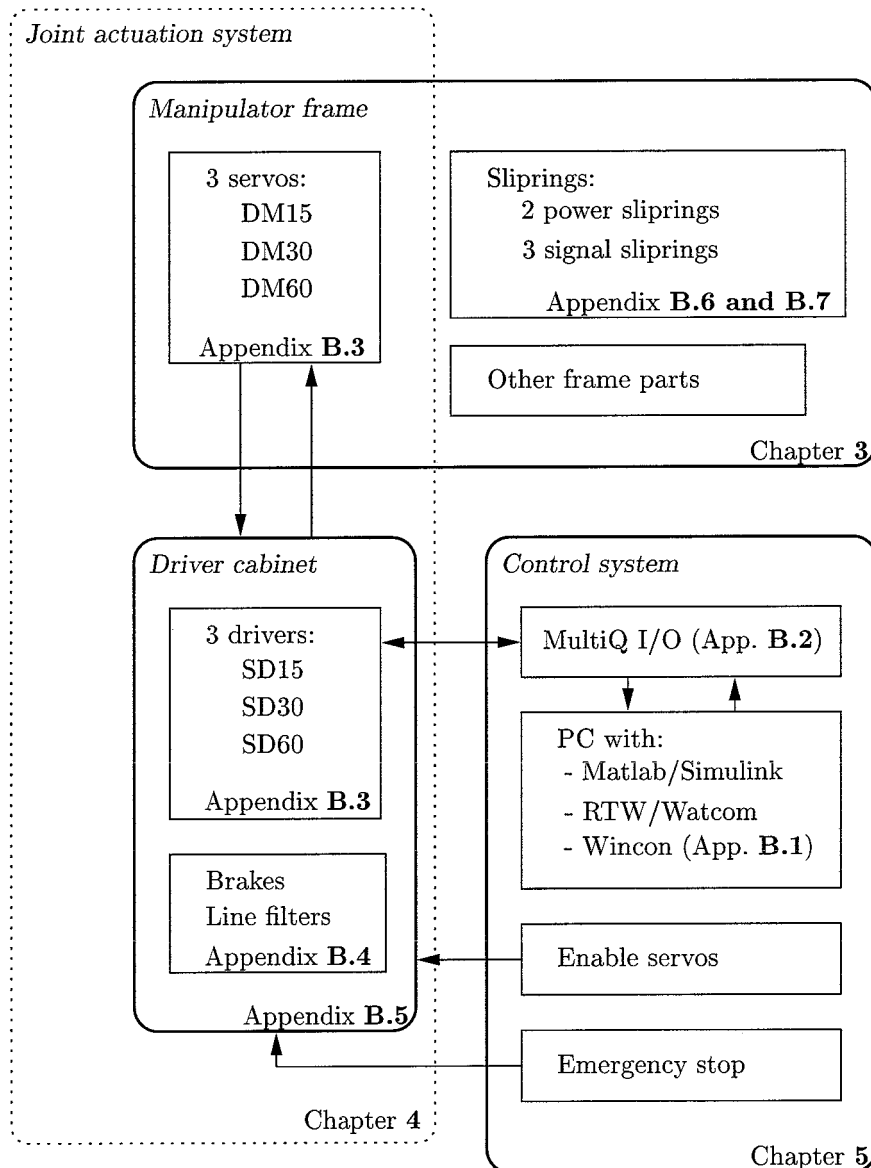


Figure 2.2: Schematic description and reference of the RRR-robot components.

In Chapter 5, the *Control system* is addressed. A PC plug-in control board, the MultiQ (Fig. 2.3), is used to communicate with the motor drivers, e.g., read-in the output of the encoders, and apply the torque command voltage. The terminal board of the MultiQ is placed inside a casing (the "controller" in Fig. 2.1), and connected by a 50-pole connection (cable ④) to each driver.

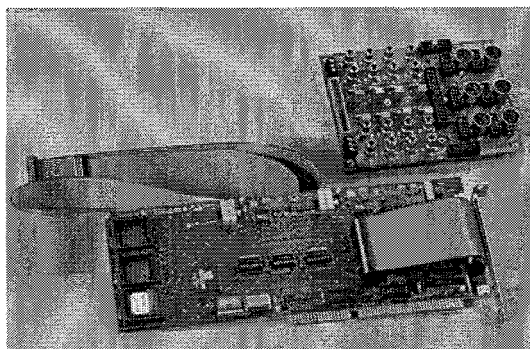


Figure 2.3: MultiQ plug-in board, placed inside the PC (bottom), and terminal board (top right)

The actual control algorithm is implemented in Simulink. If desired, the algorithm can be tested and analyzed by means of simulation. If the results are satisfactory, the simulink blocks can be automatically converted to C-code by the Real Time Workshop (RTW), and subsequently compiled by the Watcom C-compiler. The compiled C-code can be executed directly (using DOS), or in interaction with the Wincon software (using Windows). Using Wincon selected Simulink variables can be displayed and modified in real-time.

In Fig. 2.2 all main components of the RRR-robot are depicted schematically.

Chapter 3

Manipulator frame

3.1 Composition

The manipulator frame consists of servos, sliprings and non-standard frame parts (e.g. links) manufactured by the CTD. Furthermore, it includes all internal wiring. In Table 3.1 the main components are listed, together with their origin and mass. In Fig. 3.1, the conceptual design, all components are assembled. Based on this design the CTD constructed the non-standard components. Therefore, the design sketch can only serve as an indication for the composition of the final RRR-robot (several dimensions may be slightly altered).

Table 3.1: RRR-robot main components (see Fig. 3.1). With the exception of part 8, all parts manufactured by the CTD are made of Aluminum. The masses are either measured, estimated, calculated (indicated with a *), or supplied by the manufacturer.

No.	Part	Source	Appendix	Mass [kg]
1.	Base	CTD	B.8	15*
2.	Spherical joint	CTD	B.8	2*
3.	Housing DM60	CTD	B.8	6*
4.	DM60	Litton	B.3	12
5.	Signal slipring 1 (SSR-1)	Litton	B.7	< 0.1
6.	Rotor extension DM60	CTD	B.8	} 3.504
7.	Power slipring 1 (PSR-1)	Litton	B.6	
8.	Bearing 165-120-22	CTD	B.8	< 0.5
9.	Bearing cover	CTD	B.8	< 0.1
10.	Housing DM30	CTD	B.8	15.5
11.	Stator extension DM30	CTD	B.8	1.620
12.	DM30	Litton	B.3	7.5
13.	Signal slipring 2 (SSR-2)	Litton	B.7	< 0.1
14.	Rotor extension DM30	CTD	B.8	1.482
15.	Power slipring 2 (PSR-2)	Litton	B.6	3.464
16.	Arm 2	CTD	B.8	3.908
17.	DM15	Litton	B.3	5.5
18.	Signal slipring 3 (SSR-3)	Litton	B.7	< 0.1

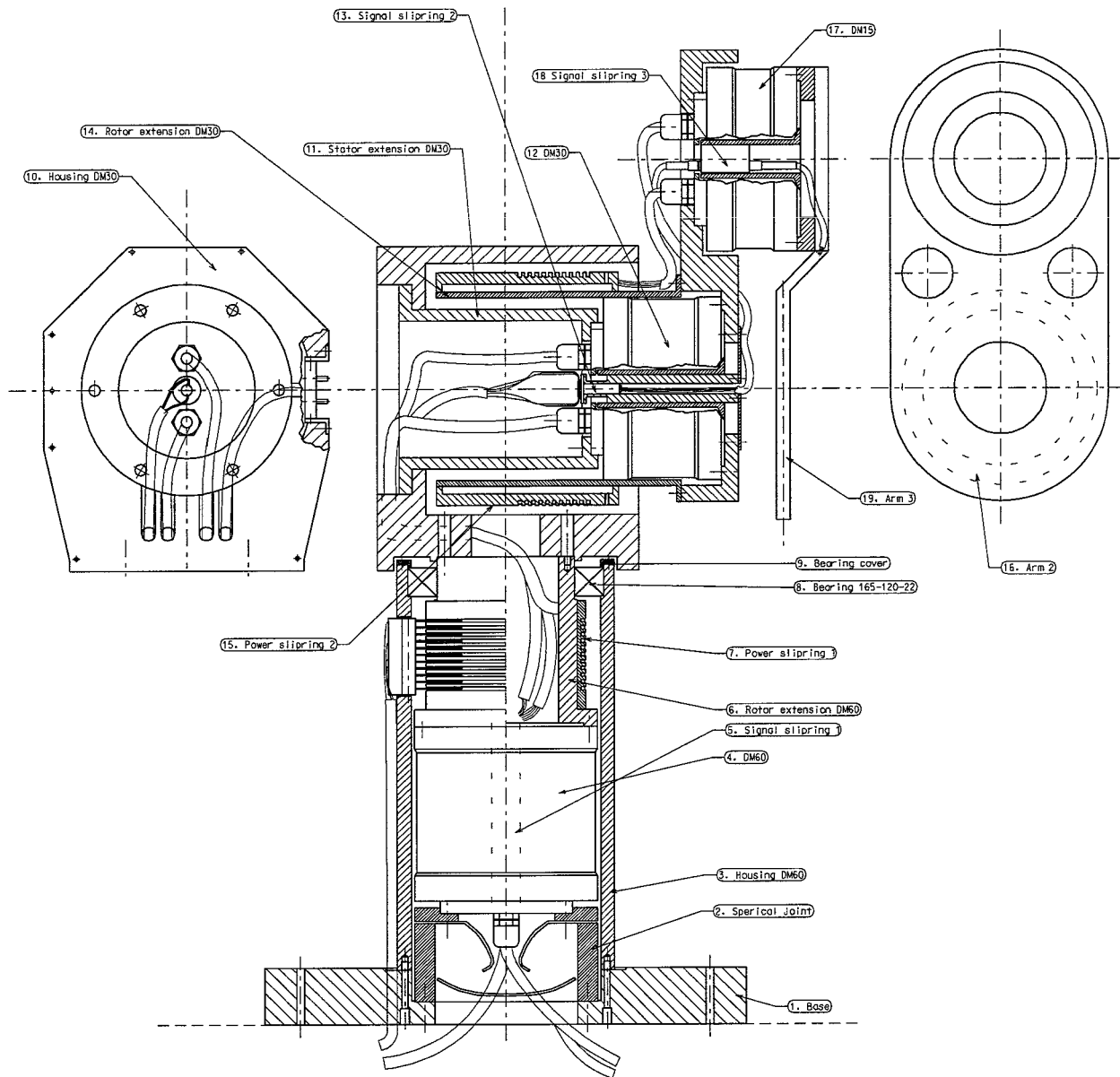


Figure 3.1: RRR-robot conceptual design, used as a basis for the realization of the various non-standard components. Note that some dimensions were slightly altered during the manufacturing

3.2 Assembly and installation

In the following, the assembly steps of the frame components are listed. Here, the emphasis is on the mechanical connections. In Section 3.3, the wiring of the sliprings will be treated in more detail.

1. The signal sliprings (5., 13., and 15.) are clamped and glued inside the three motors. The wires to and from the sliprings are bundled and shielded.

2. The Spherical joint (2.) and Housing DM60 (3.) are connected to the Base (1.) with two series of bolts at the bottom of the Base (1.).
3. The DM60 (4.) is bolted to the Spherical joint (2.), through the Base.

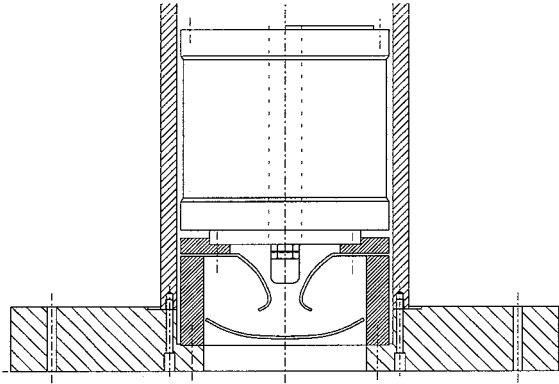


Figure 3.2: Step 2 and 3: connections made from the bottom, three sets of bolts are tightened while the Base stands on its side.

4. PSR-1 (7.) is placed around Rotor extension DM60 (6.) and secured with radial screws. Both are mounted on the rotor of the DM60 (4.). The 8 slipping cables are extended (see Table 3.3, third column). Now the brush block (part of 7.) can be placed.

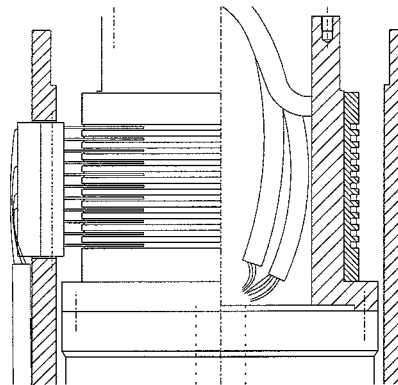


Figure 3.3: Step 4: PSR-1 (including brush block) and Rotor extension DM60. The three cable bundles inside the Rotor extension are, from left to right, encoder extension cables to DM15 and DM30; and the 8 extended PSR-1 leads.

5. The Bearing (8.) and the Bearing cover (9.) are mounted.

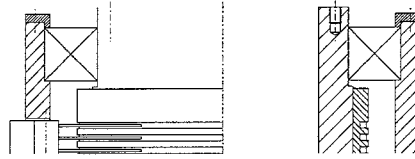


Figure 3.4: step 5: additional bearing.

6. Housing DM30 (10.) is placed on Rotor extension DM60 motor (6.). The cables for data and power originating from PSR-1 (7.) and SSR-1 (5.) enter Housing DM30 through 4 slots (which can be seen at the backside of Housing DM30 when the cover is removed). For details and the connection to PSR-2 (15.) see Table 3.3 and Table 3.5.
7. The bolts (screwed into plugs) connecting Housing DM30 (10.) to the Rotor extension DM60 (6.) are fastened from the inside of the housing. This is done *before* Stator extension DM30 (11.) is in place.

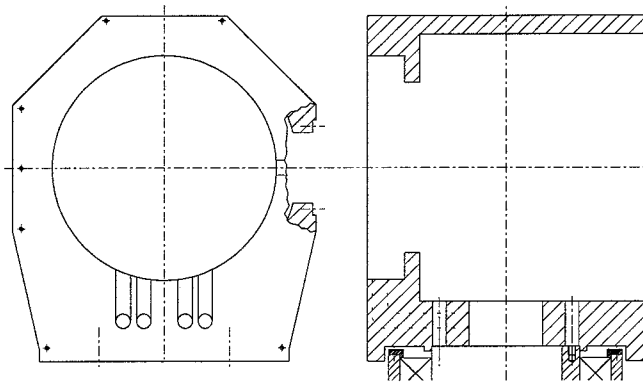


Figure 3.5: Step 6 and 7: mounting Housing DM30 (10.)

8. Stator extension DM30 (11.) is bolted to Housing DM30 (10.).

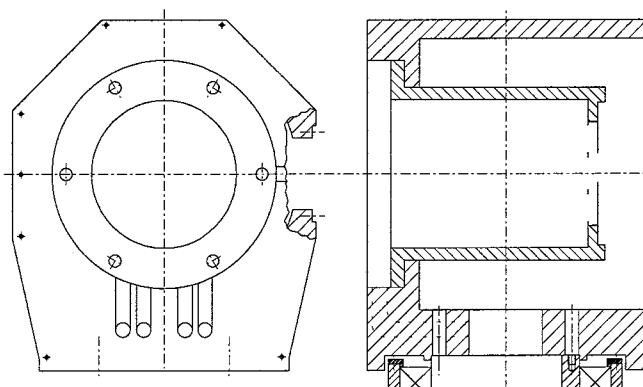


Figure 3.6: Step 8: mounting Stator extension DM30 (11.)

- DM30 (12.) is mounted on Stator extension DM30 (11.) through the back of Housing DM30 (10.). The DM30 motor cable, the DM30 encoder cable and the PSR-2 bundles can be connected with the appropriate cables (entering through the 4 slots). For details see Table 3.3 and Table 3.5.

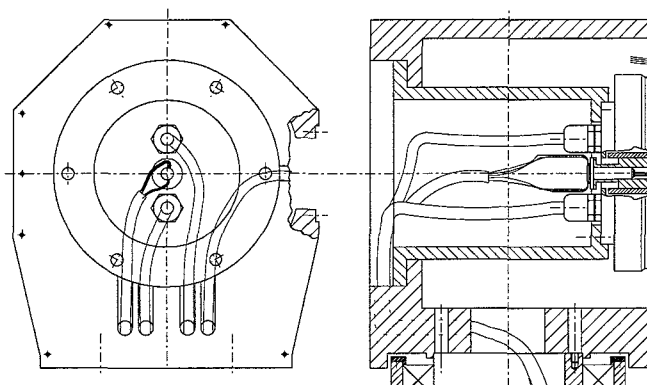


Figure 3.7: Step 9: mounting DM30 (12.).

- PSR-2 (15.) is mounted on Rotor extension DM30 (14.). Both are placed over DM30 (12.) and bolted to arm2.

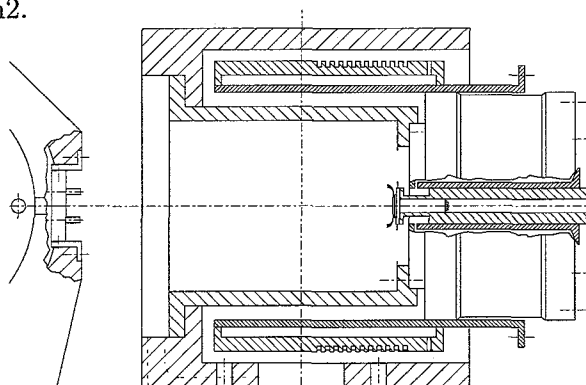


Figure 3.8: Step 10: PSR-2 (15.) and Rotor extension DM30 (14.) are placed over DM30 (12.)

- Arm 2 (16.) is bolted to the DM30 motor, and Rotor extension DM30 (14.) is fixed to Arm 2. Now the brush block (part of PSR-2) can be placed, and subsequently connected to 8 of the 4 leads originating from PSR-1 (7.).

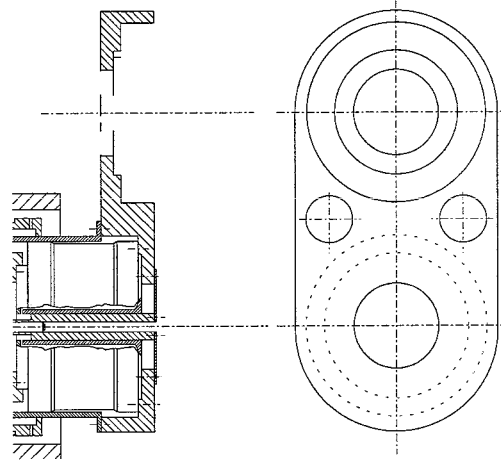


Figure 3.9: Step 11: Mounting arm 2 (16.) and securing the Rotor extension DM30 (14.).

- DM15 (17.) is mounted on Arm 2. (16.) The 12 leads from PSR-2 (15.) are grouped in sets of three and connected to the DM15 motor cable. The bundle from SSR-2 (13.) is split up and connected to the DM15 encoder cable and SSR-3 (18.).

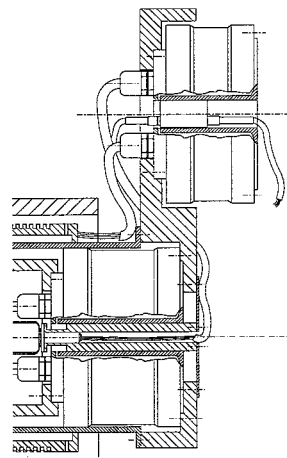


Figure 3.10: Step 12: DM15 (17.) mounting and connecting.

- Arm 3 (19.) is bolted to the DM15 motor

Remark

- It is possible to remove DM30 and everything connected to it as a whole:
 - remove the PSR-2 brushes,
 - disconnect the power to DM30 and all signal wires in the DM30 housing, and
 - carefully support the parts to be disconnected, and unscrew the bolts (inside housing DM30) which connect the stator extension to the DM30.

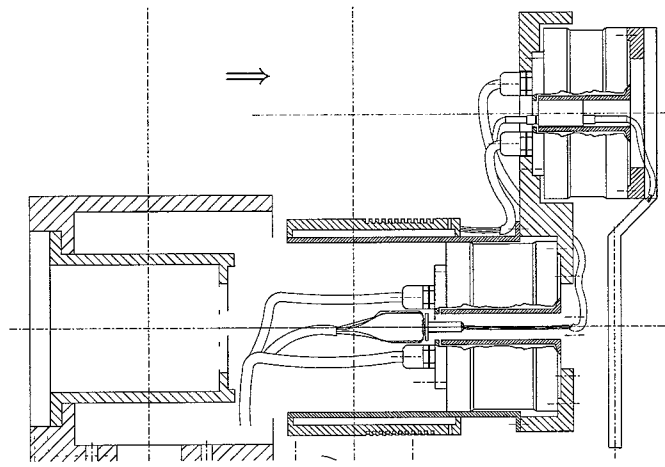


Figure 3.11: Partial disassembly

3.3 Wiring

3.3.1 Power connections

The DM60 motor is directly connected to the SD60 driver with its (extended) motor cable (1=red, 2=white, 3=black, and 4=green), see Fig. 3.12.

To connect the DM30 and the DM15, power sliprings are used: PSR-1 (7.) and PSR-2 (15.). For the location of these sliprings see Fig. 3.1 and assembly steps 4 and 10. The main specifications of these rings are summarized in Table 3.2.

Table 3.2: Main specifications of power sliprings PSR-1 and PSR-2.

property	PSR-1	PSR-2
	SM140-6 and -2	M-SM204-12
number of circuits	8	12
maximum current/circuit	15 [A]	6 [A]
maximum total load	15 [A] \times 50 [V]/circuit	-
maximum velocity	250 [rpm]	150 [rpm]
circuit resistance	< 0.1 [Ohm]	-

PSR-1 is connected from the base, through a brush-block terminal with 8 contact points, schematically depicted by the 8 “x”’s in Fig. 3.12 (right).

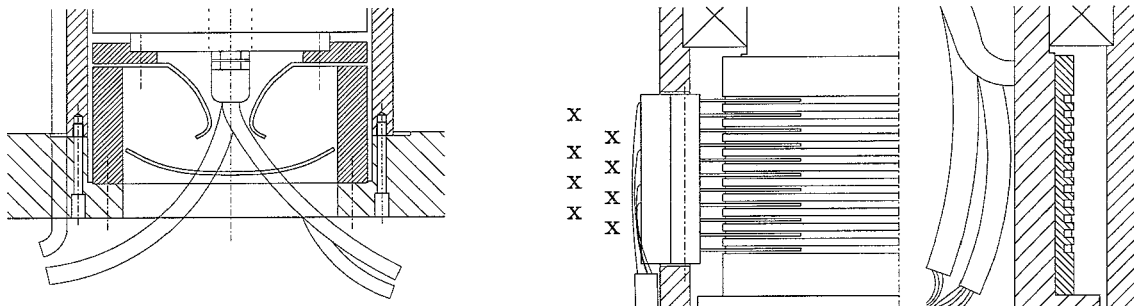


Figure 3.12: Connections to the DM60 (left) and PSR-1 with brush-block terminal (right). Each “x” denotes a clamp bolt (see also Fig. 1.1).

In Table 3.3, the same symbols are used to describe the motor cable connections. For example the DM15 motor cable from the SD15 driver with 4 leads is connected to the bottom 4 clamps of the PSR-1 terminal. Inside PSR-1, 4 leads are extended and (through slot 4 of Housing DM30, depicted by [o o o x]) led to the PSR-2 terminal shown in Fig. 3.13.



Do not push any of the power leads in Housing DM30 (green, black, yellow and red) further back into slot 3 and 4. Then they might come into contact with power slipring 2.

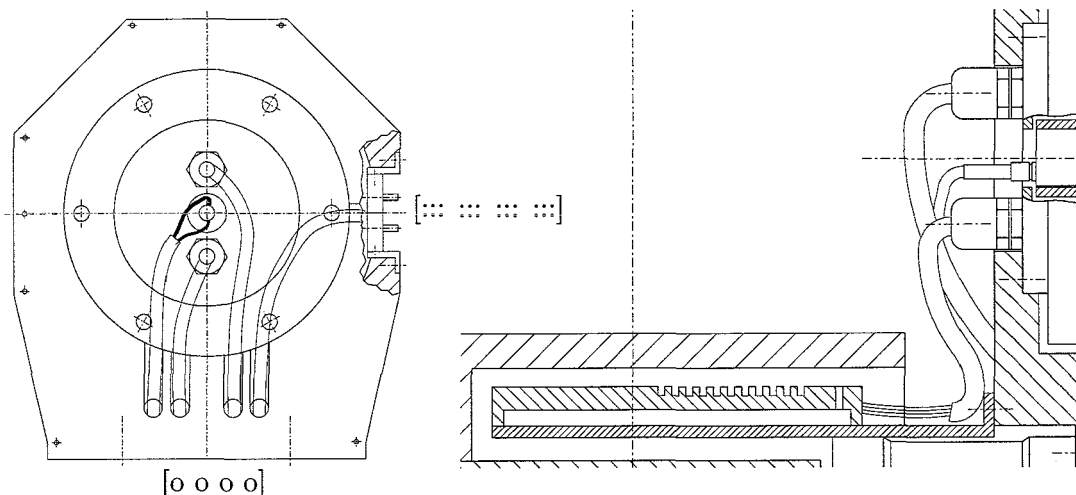


Figure 3.13: Cable slots and brush-block PSR-2 (left), and PSR-2 (right). Each “.” denotes a brush (see also Fig. 1.1).

The PSR-2 terminal has 24 brushes (12 circuits, 2 brushes per circuit) denoted by [::: :: :: :::]. Groups of 6 brushes are interconnected to form 4 channels each consisting of 3 circuits. The same is done on the side of DM15 to power this last servo.

Table 3.3: Connections to power slipping 1 and 2 (PSR-1 and PSR-2). The “x” in [o o x o] denotes a cable entering motor housing 2 via slot 3. The “x” in [::: :: :: :: xxx] denotes a connection to the fourth set of 6 brushes on the horizontal terminal block of slipping 2.

driver SD30		terminal PSR-1	inside PSR-1	housing DM30		
VA red	x		red	via slot 3		
VB white		x	yellow	[o o x o]		
VC black	x		black	to DM30		
GND green		x	green			
driver SD15					terminal PSR-2	side PSR-2/ to DM15
GND green	x		green	via slot 4	::: :: :: :: xxx	green
VC black		x	black	[o o o x]	::: :: :: xxx :::	black
VB white	x		yellow	to PSR-2	::: xxx :: :::	white
VA red		x	red		xxx :: :: :::	red

3.3.2 Signal connections

The encoder cable of the DM60 is directly connected with the SD60 driver. To transfer the encoder signals of the DM30 and the DM15 and to transfer additional measurement signals, three signal sliprings are used: SSR-1 (5.), SSR-2 (13.), and SSR-3 (18.). The dimensions and main specifications of these rings are listed in Fig. 3.14 and Table 3.4.

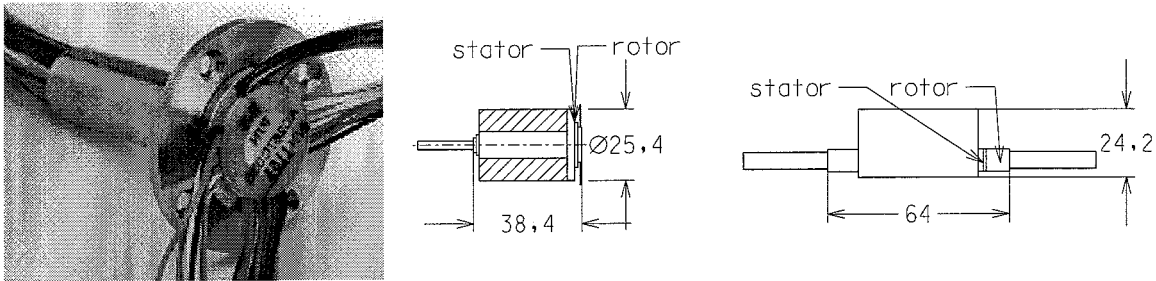


Figure 3.14: Signal sliprings: on the left, a photograph of the original AC-267 slipring; in the middle, the modified AC-267 (used for SSR-1 and SSR-2); and on the right, SSR-3.

The stator of each sliprings is clamped and glued on the inside of the servos (see position 5., 13., and 18. in Fig. 3.1), i.e., attached to the DM rotors. Since the slipring friction torques are small, the slipring leads are used to fix the slipring rotors to the DM stators. For the SSR-1 and SSR-2, tie-raps are used to tie 2 bundles of 18 wires to the motor cable and the encoder cable. Note that, every wire is attached separately to the slipring (see Fig. 3.14), i.e. there is no mechanical connection between the (white) bundle of 18 wires and the slipring itself. Below the base, a cable relieve clamp is placed to prevent any damage to SSR-1.



Do not pull any of the bundled slipring cables (white) originating from SSR-1 (below the base), or SSR-2 (in the octagonal shaped, silver colored Housing DM30 (10.)).

Table 3.4: Main specifications of the signal sliprings (see Fig. 3.14).

property	SSR-1 and 2 Capsule AC-267 modified	SSR-3 Litton 12-ring capsule
number of circuits	36	12
maximum current/circuit	1.2 [A]	1 [A]
maximum total load	25 [A]	50 [W]
maximum velocity	100 [rpm]	120 [rpm]
total friction torque	0.0054 [Nm]	0.0023 [Nm]

To reduce the sensitivity of the signal data to external disturbances (e.g. originating from the servos) two precautions have been taken:

- Slipring leads are bundled and provided with an external shield.

- For the encoder feedback, redundant circuits are used to cancel out possible disturbances. In total 10 circuits are used: 2 for power, and 8 for data. The latter consist of 4 twisted pairs transmitting the pulse trains A, \bar{A} (A inverse), B, and \bar{B} . Assuming a disturbance acts both A (or B) and the inverted, it can be canceled out as shown in Fig. 3.15.

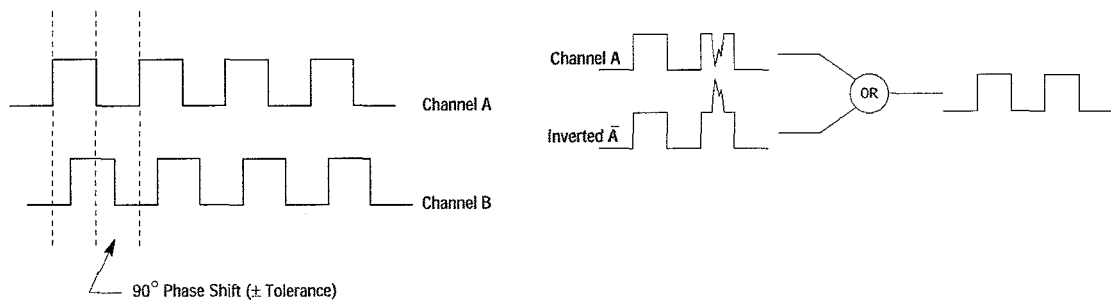


Figure 3.15: On the left, basic quadrature encoder output; and on the right complementary outputs to cancel out disturbances.

The slipring wires are connected to the encoder leads of DM30 and DM15. On Arm 3, 12 wires are available for future sensors. All slipring wires are numbered on both the stator and rotor side. In Table 3.5 all connections are listed. For example, the blue wire with label 27 goes through the base to the rotor of SSR-1. On the stator side another blue wire (part of bundle 2) enters Housing DM30 through the second slot, and is connected to a blue-brown wire (again with label 27) leading to the rotor of SSR-2. On the stator side of SSR-2, the blue-brown wire is connected to a grey wire (label 3) belonging to SSR-3, and ending on link 3.

Table 3.5: Connections to the 36-circuit signal sliprings (SSR-1 and SSR-2) mounted inside the DM60 and the DM30 motor. Wires are numbered with the least significant number closest to the end of the wire, e.g. wire 31: —3 1- or -1 3—. SSR-1.1 [x o o o] denotes signal slipring 1, bundle 1, entering motor house 2 through the utmost left slot (Fig. 3.13).

no.	SSR-1.1 [x o o o]	encoder DM30	
1.	red	1. red	
2.	orange	1. shield	
3.	black-orange	2. black	
4.	black-red	2. black	
5.	blue	3. blue	
6.	blue-white	4. blue-white	
7.	blue-orange	5. brown	
8.	white-yellow	6. brown-white	
9.	black-blue	7. green	
10.	green-red	8. green-white	
11.	blue-red	9. orange	
12.	red-white	10. orange-white	
	SSR-1.2 [o x o o]	SSR-2.2	encoder DM15
13.	grey-red	grey-red	1. red
14.	(dark) grey-red	red	1. shield
15.	black-purple	black-purple	2. black
16.	black-yellow	black	2. black
17.	blue-brown	blue-red	3. blue
18.	blue-yellow	blue-white	4. blue-white
19.	brown	grey	5. brown
20.	brown-white	red-white	6. brown-white
21.	green	blue-yellow	7. green
22.	green-white	white-yellow	8. green-white
23.	orange	brown-red	9. orange
24.	orange-white	red-yellow	10. orange-white
			SSR-3
25.	purple	blue-(purple)	1. blue
26.	red-yellow	black-yellow	2. purple
27.	blue	blue-brown	3. grey
28.	blue-red	blue	4. brown
29.	purple-white	black-white	5. black
30.	white	yellow	6. yellow
	SSR-1.1 [x o o o]	SSR-2.1	SSR-3
31.	blue	blue-black	7. white
32.	grey	brown-white	8. brown-white
33.	black-green	black-red	9. red
34.	black	red	10. black-white
35.	black-white	green	11. orange
36.	yellow	purple-white	12. green

Remarks

- Two parallel circuits are used for one of two the encoder-power leads to both the DM30 and the DM15 encoder: wire 3 and 4 to SSR-1.1 and wire 15 and 16 to SSR-1.2 (and SSR-2.2). In a non-parallel configuration, two circuits could be saved, resulting in two additional data channels from the Base to Arm 2 (no further since SSR-3 has 12 circuits).
- Because SSR-3 has only 12 circuits, SSR-2 has 12 unused circuits (on bundle 1). Currently, these wires are connected to the shield of SSR-2.1.

3.4 Operation

Lifting

Preferably, use the Base (1. in Fig. 3.1) to move or lift the frame (approximate weight 90 [kg]). If this is not possible, use a strap around Housing DM60 (4., black). Do not use the octagonal shaped, silver colored Housing DM30 (10.) to lift the robot; unless with extreme care to prevent damage to the Spherical joint (2.) at the base of the robot.



Slipping standstill

Using the power slipping to transfer power while at a continuous standstill may cause wear and/or damage to the brushes and ring surfaces due to welding effects.



3.5 Inertia parameters

After completion of the robot, *Microstation Modeler* was used to generate a 3D model of the robot in order to calculate the inertia tensors of each joint-link pair expressed in the appropriate link coordinates (see Fig. 3.16).

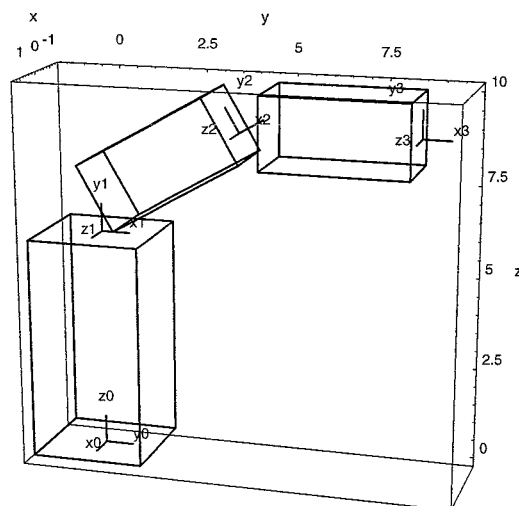


Figure 3.16: Orientation of the coordinate frames: O_0 is placed at the center-base of link 1; O_1 at the intersection of the DM60 and DM30 rotation axes; O_2 at the intersection of the DM15 rotation axis and the line parallel to the length axis of Arm 2, through its center of gravity; and O_3 at the end of the line parallel to the length axis of Arm 3 through its center of gravity. Note that Arm3 is not yet present, so O_3 is located at the end of a virtual link with length 0.3 [m].

In Table 3.6, all moving components are grouped to form joints (the motors) and links.

Table 3.6: Moving parts: joints and links

part	subcomponents	mass [kg]
Joint 1	DM60	12
Link 1 (vertical)	Rotor extension DM60, PSR-1, Housing DM30, and Stator extension DM30	20.6
Joint 2	DM30	7.5
Link 2 (green)	Rotor extension DM30, PSR-2 and Arm 2	8.9
Joint 3	DM15	5.5
Link 3	Arm 3 (not present)	-

The combined inertia tensor of Link i and Joint i with respect to frame i is given by J_i :

$$J_1 = \begin{pmatrix} 2.2334 & -2.6560 \cdot 10^{-7} & 2.7755 \cdot 10^{-8} & 0 \\ -2.6560 \cdot 10^{-7} & 0.2213 & -0.01456 & -6.1425 \\ 2.7755 \cdot 10^{-8} & -0.01456 & 2.2623 & -0.2270 \\ 0 & -6.1425 & -0.2270 & 32.9305 \end{pmatrix} \quad (3.1)$$

$$J_2 = \begin{pmatrix} 0.1921 & 1.9029 \cdot 10^{-4} & -0.2068 & -2.8439 \\ 1.9029 \cdot 10^{-4} & 0.7340 & 0 & 0 \\ -0.2068 & 0 & 0.6474 & -1.0222 \\ -2.8439 & 0 & -1.0222 & 16.3123 \end{pmatrix} \quad (3.2)$$

$$J_3 = \begin{pmatrix} 0.01238 & 0 & 0 & 0 \\ 0 & 0.01190 & 0 & 0 \\ 0 & 0 & 0.012 & 0 \\ 0 & 0 & 0 & 5.4763 \end{pmatrix} \quad (3.3)$$

Chapter 4

Joint actuation System

4.1 Dynaserv motors

The *Dynaserv* motors are self-contained units containing precision ball bearings, magnetic components, and integral feedback. The motor is outer-rotor, providing direct motion of the outside of the motor. On the inside, the motor has a hollow core (diameter 25 [mm]) which is part of the rotor. With regard to the stator, only the base part is visible.

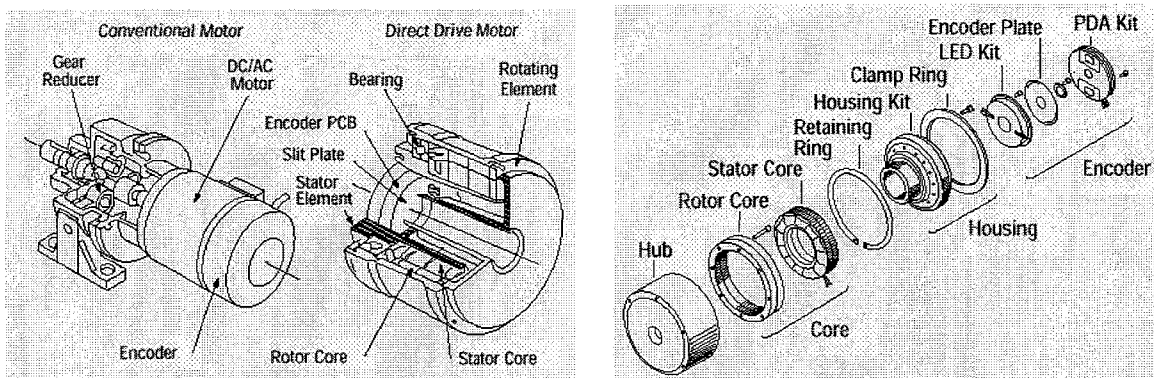


Figure 4.1: On the left, Dynaserv motor v.s. a conventional motor; and on the right, an exploded view of the Dynaserv servo.

In Table 4.1, the main specifications of the motors are summarized. The listed maximum torque and maximum velocity cannot be achieved simultaneously, as illustrated in Fig. 4.2.

The torque of each motor can be controlled by sending a command voltage to its respective driver (± 8 [V]). Although Litton stated the motor torques listed in Table 4.1, the relation between torque and voltage is not linear. Preliminary measurements confirm the trend shown in Fig. 4.2 (taken from the on line manual of Compumotor): Moving Arm 2 to an horizontal position (14.62 [Nm], or 49 % of 30 [Nm]) required input voltages between 2.6 and 2.9 [V] (31 % to 34 % of 8.5 [V]).

Table 4.1: Main specifications of the Dynaserv motors. Source: Appendix B.3 and personal communications with Litton (marked by *).

property	unit	DM60 Motor 1	DM30 Motor 2	DM15 Motor 3
max. output torque	[Nm]	60	30	15
max. velocity	[rps]	2.4	2.4	2.4
mass	[kg]	12	7.5	5.5
inertia	[kg m ²]	0.023	0.015	0.012
allowed axial load	[N]	3 10 ⁴ (compression), 3 10 ⁴ (tension)		
allowed moment load	[Nm]	200 (static load), 60 (alternating load*)		
allowed velocity	[rps]	0.5	1	1.5
torque constant*	[Nm/V]	15.6	7.8	3.9
max. power	[kW]	2.2	2.0	1.6



Do not exceed the following velocities in order to stay below the allowed moment for alternating load, DM60: 0.5 [rps], DM30: 1 [rps], and DM15: 1.5 [rps].



Ensure that the motor phases V_A , V_B , and V_C are connected correctly. If not, the position feedback-loop results in unstable behavior, i.e., the motor will start rotating at maximum velocity and with maximum torque in a direction opposite to the desired trajectory.

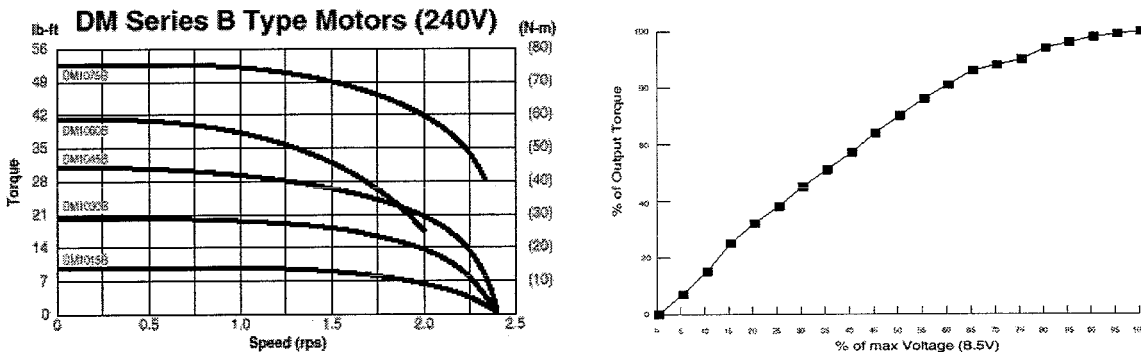


Figure 4.2: On the left, the DM series motor performance (torque v.s. velocity), on the right the saturation of the motor output torque as a function of the command voltage send to the driver.

4.2 Driver cabinet

The Driver cabinet contains three Dynaserv drivers (mounted on rails for access to the jumpers and the notch filter), line filters and two brakes (for DM60 and DM30). All drivers and brakes are powered by one central power supply which can be controlled by a switch, and two emergency stops (on the cabinet and remote).

Ensure that the power is switched off before opening the Driver cabinet (e.g. to change driver settings). Dangerously high voltage is present inside this unit.



On the back of the cabinet are connectors for (from left to right): the control of the brakes (two \times 6 pol. DIN connectors), the encoder feedback from the motors (three \times 15 pol. sub D), the power to the motors (three \times 5 pol. Harting), and the remote emergency stop. Furthermore, here the extended CN-1 terminals leave the cabinet.

4.3 Dynaserv drivers

Since the Dynaserv motor is a brushless DC motor, the commutation of currents is accomplished by measuring the rotor position using an encoder. The driver is powered by 220 [V] AC (cable ① in Fig. 2.1). Based on the encoder feedback (cable ②) and the internal/external controller, the motor phases V_A , V_B , and V_C (cable ⑤) to actuate the motor are determined.

Each Driver has several control modes:

- Position Control Mode (I-PD, P-P and P-I type)
- Speed Control Mode (P and PI type)
- Torque Control Mode

When using the external controller (PC and MultiQ) the driver is set to the torque control mode (for more detailed information on the required jumper settings, see Appendix B.3)

Furthermore, each driver has a first-order lag filter and a notch filter to reduce vibrations. At present both are unused.

Although, the manual recommends a maximum filter offset voltage of 0.05 [V], this can, and should be set below 0.005 [V] to avoid a noticeable offset in the torque command voltage.



The control system interfaces with each driver through the CN-1 terminal (cable ④ in Fig. 2.1). From the Driver cabinet all CN-1 connections are extended to the casing of the MultiQ plug-in board.

When switching on the power, a surge of approximately 8 [V] occurs at the driver outputs (CN-1, VEL, POSN and TORQ in Fig. 2.1)



4.4 Dynaserv brakes

The DM60 and DM30 are equipped with an electro-mechanical brake to bring the robot to a controlled stop. Each brake consists of a board with power electronics and relays to short the motor phases (V_A , V_B and V_C) either directly (at low velocities), or by using a dissipative RC-circuit (at high velocities). Simply stated, the brake is wired by cutting the motor cable (cable ⑤ in Fig. 2.1), and connecting both sides to the board. For more details see Appendix B.4.

Each brake has a separate power supply, and is mounted inside the Driver cabinet. On the backside of the Driver cabinet are connections to engage the breaks. If the power is switched off (power failure or emergency stop), the brakes are engaged automatically.

Chapter 5

Control System

5.1 MultiQ plug-in and terminal board

The MultiQ I/O board from Quanser Consulting has 8×13 bits ADC, 8×12 bits DAC, 8 digital I/O, 6 encoder inputs, and 3 hardware timers. Also included is a terminal board with several connectors.

Using separate test software the I/O times of the MultiQ board can be measured:

Digital input (16 [bit])	5 μ s
Digital output (16 [bit])	2 μ s
Encoder read (24 [bit])	5.5 μ s
ADC (16 [bit])	20 μ s
DAC (16 [bit])	5 μ s

For a computed torque controller based on encoder data only (see Fig. 5.2), the total I/O time amounts to 31.5 μ s. Using external timing it was verified that a sampling frequency of 1 [kHz] can be achieved without interruptions. Higher frequencies require the use of one of the on-board clock timers, and were not yet tested.

The terminal board (see Fig. 2.3) is located inside a casing with three 50-pole connectors. Here, the three driver-CN1 extensions (originating from the back of the Driver cabinet) are inserted. Inside the casing, the MultiQ inputs, and outputs are connected to the right CN1 pins. Furthermore, through here, the servo enable signal is fed to the drivers (originating from an independent power supply).

Connections to the three driver-CN1 extensions inside the casing:

- Encoder read (pin 13, 14, 29 and 30):

The 4 encoder outputs are connected to a AM26LS32 line driver (white=[13. A+], brown=[14. A-], green=[28. B+], yellow=[30. B-], black=ground). The outputs are connected to the first three encoder DIN connectors (0, 1, and 2).

Use the emergency stop and perform a (hard) reboot of the PC containing the MultiQ I/O board, if the encoder read-out becomes discontinuous or otherwise erratic.



- Torque command (pin 49 and 50):

Pin [49. VIN] is connected to the core of a coax cable, the shield is connected to pin [50. AGND]. Using tulip connectors the three cables are plugged in analog MultiQ output 0, 1 and 2.

- Enable servo (pin 23 and 24):
See Section 5.2.

Although, an analog velocity output is present (pin 17 and 18), currently, it is unused because:

- Differentiating the encoder yielded better (without a bias) results.
- A surge of 8 [V] occurs at the CN1 outputs when the power is switched on. Without additional overload protection, this may cause damage to the MultiQ board:



Do not apply more than 5.3 [V] to the (analog) input of the MultiQ I/O board to avoid overload.

- A/D conversions are the most time consuming operations of the MultiQ board.

5.2 External controls

Enable Servos and Brakes Both servos and brakes are enabled, i.e., made ready for operation, by closing a circuit (like the one shown in Fig. 5.1) between two pins of each driver-CN1 terminal (inside the terminal board casing), and each brake-CN2 terminal (extended to the back of the Driver cabinet).



Do not use the enable circuit as an alternative emergency stop. There is a considerable delay (approximately one second) before opening the circuit has effect.

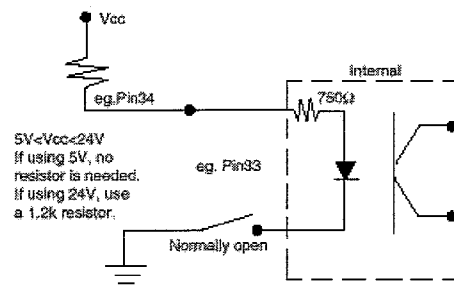


Figure 5.1: Example (enable) circuit for both the three drivers and the two brakes. On the driver-CN1, the circuit is between pin [24. VCC], and [23. $\overline{\text{SRVON}}$]. On the brake-CN2, it is between [3. VCC] and, [4. $\overline{\text{SRVON}}$]. Normally the circuit is open, i.e. the $\overline{\text{SRVON}}$ -signal is set to HIGH, and driver (or brake) does not accept commands. After closing the circuit, i.e., setting $\overline{\text{SRVON}}$ to LOW, the driver (or brake) is ready for operation.

Emergency Stop



Always keep the (remote) emergency stop within reach while operating the robot. Use the emergency stop if error messages occur after the Simulink controller has been started; if the control software or the operation system crashes; or if the encoder read-out becomes discontinuous or otherwise erratic.

5.3 Personal Computer

5.3.1 PC Hardware

The PC configuration consists of a Pentium Pro 200 with 32 MB RAM, it has three free PCI slots and three ISA slots.

5.3.2 PC software

The PC software for controlling the robot consists of several programs (between brackets are the *untested* updated versions):

- The operating system Windows 3.11 (Windows 95).
- Matlab 4.2c (Matlab 5.1) and Simulink 1.3 (Simulink 2.1) to design, analyze, and implement the controller.
- The Real-Time Workshop 1.1C (RTW 2.1) to convert the controller – consisting of Simulink blocks – to C-code.
- The Watcom C-compiler 10.6 (Microsoft visual C compiler) to compile the generated C-code to an executable.
- Wincon V2.0 (Wincon V3.0) to run the executable under windows while displaying and modifying selected Simulink variables.

When performing an experiment, do not start the control algorithm before the servos are powered and enabled; the introduced tracking error may cause a large and unwanted response.



An example of a Simulink controller is shown in Fig. 5.2. This diagram also displays the trajectory generation, the blocks for the data acquisition board, and the feedback-loop signal flow.

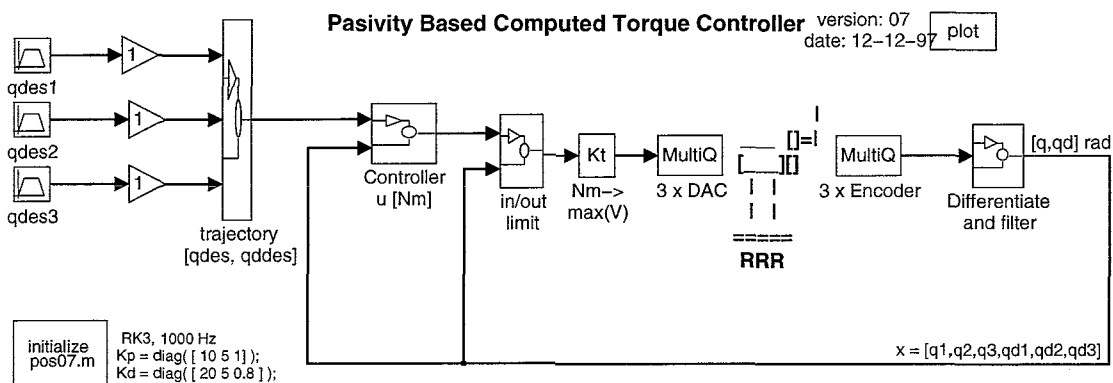


Figure 5.2: Simulink block scheme of a Passivity Based Computed Torque Controller.

After modifying the Simulink controller, always perform a “test run” to see if error messages occur.



Chapter 6

Maintenance and inspection

Motors Check daily if changes in noise level or excessive vibration occur. After 20.000 hours or 5 years, some parts may need to be replaced. When an overhaul or motor disassembly is required, contact Litton. (Appendix A).

Driver cabinet To prevent problems due to poor insulation, periodically remove accumulated dust inside the Driver cabinet and the individual drivers.

Sliprings To prevent problems due to poor insulation, periodically remove accumulated dust inside the Base (1.) and Pedestal; inside Housing DM30 (10.); and behind the (silver colored) cover on Arm 2 (16.).

Additional frame parts The bearing 165-120-22 (8.) provided by the CTD has a lifespan of at least 15 year, and requires no maintenance.

Control system Periodically, defragment the hard disk of the PC.

Appendix A

Addresses

Litton Precision Products International

product: Dynaserv servos, sliprings and brakes
address: Griendstraat 10
2921 LA Krimpen a/d Yssel
The Netherlands
tel. 0180-596888
fax. 0180-596899
contact person: Rolf van de Weijer
littonnl@worldaccess.nl
web site: <http://www.litton.com>
technical support servos: Reto Muff (Switzerland)
tel. 0041-1313 1450
fax. 0041-1313 1255
email: RMuff94444@aol.com

CTD

product: frame manufacturing
contact: Erwin Dekkers
W-hal 1.58
tel. 3356

GTD Groep Electronica/Energietechniek

product: Driver cabinet
Realization: B. Viveen
tel. 3494

Quanser Consulting Inc.

product: MultiQ I/O board and Wincon
address: 102 George Street
Hamilton, Ontario
Canada
tel. 001905 527-5208
fax. 001905 570-0177
sales@quanser.com
web site: <http://www.quanser.com>

Scientific Software Benelux

product: Matlab
address: Bleulandweg 1B
2803 HG Gouda
tel. 0182-537644
fax. 0182-570380

AC-2000 Software Ingenieurs

product: Watcom C/C++
address: Standerdmolen 8-302
3990 DB Houten
PO box 83
tel. 030-635 1840
fax. 030-635 1839

Appendix B

Components

In the following sections several component manuals and drawings are included:

B.1 Wincon software manual	38
B.2 MultiQ board manual	70
B.3 Manual: Dynaserv DD Servo-Actuator DM/SD series	91
B.4 Manual: BE-A/B Type Dynamic Brake	113
B.5 Cabinet manual	123
B.6 Power sliprings	133
B.7 Signal sliprings	135
B.8 Conceptual design	138

B.1 Wincon software manual

Instruction manual, 32 pages

WinCon V2.0

Quanser Consulting Inc.



WinCon™, Windows™ based Realtime Controller for SIMULINK™

Quanser Consulting Inc

1.0 General Description

WinCon is a realtime program that will run SIMULINK controllers in Realtime under Windows. In order to use the controller you will need the following:

- MATLAB™
- SIMULINK™
- Real-Time Workshop™
- Watcom™ C++ compiler version 10.a or higher
- Windows™ 3.11 or Win95™
- DOS™ version 6.2
- 8 MB RAM
- IBM 486 compatible computer or higher, with math co-processor
- 1 Mbytes of free hard disk space

You will also need the SIMULINK device drivers for the data acquisition board you are using (A/D, D/A, Encoder input).

To date the following devices are supported:

- MultiQ : A/D, D/A, Quadrature Encoder interface board (Quanser Consulting Inc)
- DT2811: A/D, D/A interface board (Data Translation)
- CIO-DAS16: A/D, D/A interface board (Computer boards)

The general functional description of WinCon is the following: Draw the SIMULINK controller and generate the realtime code (RTC) using the Real-Time Workshop. Run WinCon and load the RTC you created into WinCon and run it. You now have the controller you created running in realtime, at the sample rate you selected, under Windows. You can plot all the "Scope" variables you had selected in SIMULINK while the controller is running using the plotting facilities of WinCon. The data buffer allows you to store up to 16348 points per variable. Scope data can be saved to disk at will. You can change windows while the controller is running and start another Windows application without interfering with the performance of the running controller. Specifically you can start up SIMULINK, load the diagram of the controller you are running and change parameters in realtime while the controller is running in order to instantaneously observe the effects of the gains, filters or other parameters on system behaviour! Once you are satisfied with the performance, you can start WinCon independent of SIMULINK, load the RTC and run it in realtime. You can also run WinCon along with the RTC from SIMULINK on a computer that does not have SIMULINK on it.

2.0 Setting up

It is assumed that you are well versed in Windows, MATLAB and SIMULINK

2.1 Install WinCon

- Start Windows
- Insert the WinCon distribution diskette into your a: drive
- From Program Manager Select File
- Select Run from the Pull Down Menu
- Type a:setup Hit enter
- Select Full Installation.. Follow the instructions on the screen

2.2 Copying sample files

- make a directory `c:\winsim` on your hard drive
- copy the contents of `a:\winsim` to `c:\winsim`

2.3 Configuring the directories for the makefiles

2.3.1 Using any text editor (eg notepad) edit the file

```
c:\matlab\codegen\rt\tmf\wc_watc.tmf
```

and modify the lines

```
WATCOM_ROOT = C:\WATCOM  
WINCON_ROOT = C:\WINCON
```

to match the drive and directories for where you have installed WATCOM and WinCon

2.3.1.2 Re Compilers

- **Watcom 10.0a users: change all references of "binw" to "binb" in the file `c:\matlab\codegen\rt\tmf\wc_watc.tmf` (global substitute "binw" to "binb")**
- **Watcom should be able to generate Win3.11 code even if you are running on Win95 (ie 16 bit compile)**

2.3.2 Edit `autoexec.bat` and ensure that `c:\watcomc\binw` is in the PATH

also ensure that the lines:

```
SET INCLUDE=C:\WATCOM\H;C:\WATCOM\H\WIN  
SET WATCOM=C:\WATCOM  
SET EDPATH=C:\WATCOM\EDDAT
```

Are appropriately directed in the `autoexec.bat` file

2.3.3 Edit `c:\matlab\matlabrc.m` and add the following directories to the MATLABPATH:

```
c:\winsim
c:\matlab\codegen\rt\wincon\devices
c:\matlab\codegen\rt\dos\devices
```

for example the following should be part of your `c:\matlab\matlabrc.m`,

```
matlabpath([...
'C:\winsim',...
';C:\matlab\codegen\rt\wincon\devices',...
';C:\matlab\codegen\rt\dos\devices',...
';C:\MATLAB\toolbox\local',...
.
.
';C:\MATLAB\toolbox\fuzzy\fuzdemos',...
]);
```

3.0 Operation

The guidelines for operating WinCon are given below. In order to use the full potential of WinCon, remember that in Windows you switch active windows by pressing [Ctrl Tab]. You can also switch to a desired Window by entering [Ctrl Esc] and selecting the Task from the Task List (or use [Alt Esc]).

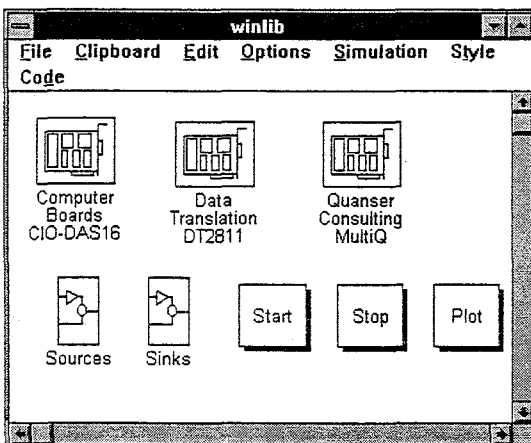


Figure 3.1 winlib supplied with WinCon

Analog Input (Mask)	
Block name: ADC	OK
Block type: Analog Input (Mask)	
Quanser Consulting MultiQ I/O Card	Cancel
Base I/O Address:	Help
<input type="text" value="0x320"/>	
AD Channels to Use:	
<input type="text" value="0"/>	
Effective Number of Bits:	
<input type="text" value="13"/>	
Sample Time (sec):	
<input type="text" value="str2num(cg_get(bdroot, 'Real-time step size'))"/>	
Access hardware during simulation (0 = no; 1 = yes):	
<input type="text" value="0"/>	

Figure 3.2 Configure the board to the right address

1) **Install a data acquisition board into your computer.** To date, the following boards are supported by WinCon/SIMULINK:

- a) MultiQ - Quanser Consulting
- b) DT2811 - Data Translation
- c) CIO DAS16 - Computer Boards

If you have a different board, Quanser Consulting will write the appropriate drivers at a nominal fee.

2) **Draw the controller in SIMULINK.** Draw the SIMULINK controller as you usually would. The data acquisition blocks and other WinCon blocks are accessed by typing `winlib` in the MATLAB Command Window. Make sure the Data acquisition blocks are configured for the right base address of the device. If there is any data you would want to monitor while the controller is running, attach them to a SIMULINK Scope and select an appropriate name for the block.

Note the sample time of the data acquisition blocks. You **must** ensure that the sample time of these blocks is the same as the actual sample time of the controller. In case of MultiQ drivers, this is automatically performed by using the `cg_get` MATLAB function as shown in Figure 3.2. You may override this by typing in the sample time you want.

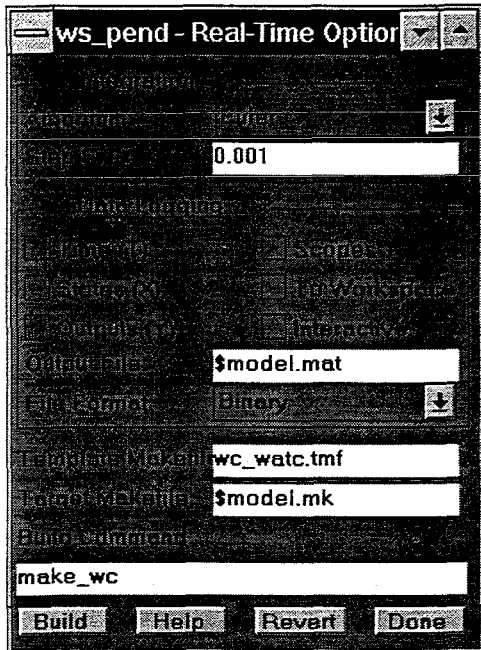


Figure 3.3 Realtime Options

3) **Configure the Real Time Code Generator.** Select Code from the menu of the drawing of the controller. Select Real Time Options:

- a) **Step size:** set the controller sampling period (eg .005 seconds)
- b) **Template Makefile:** `wc_watc.tmf`
- c) **Build Command:** `make_wc`

Note about Template makefile: You can also use `wc_watg.tmf`

You may make these choices the **default** by editing the file : `builddopt.m` (typically under `c:\matlab\toolbox\local`)

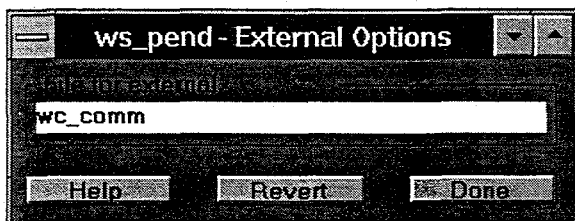


Figure 3.4 External Options

4) **Configure the simulation :** Select Simulation in the controller drawing. Select External (). Also from Simulation select External Options and enter `wc_comm` under File for External Simulation. Note that from this point on, the word "simulation" is a *misnomer*. You will be running a realtime controller but it will be referred to as Simulation from within SIMULINK.

5) **Save the drawing in a file** under the directory `c:\winsim` (or any other directory you create.) It is highly recommended that the controllers and the associated `m` files reside in a separate directory from MATLAB or WinCon. It is also good practice to start the file name with the characters `wc_` in order to find them easily later on.

6) **Change the MATLAB data directory** Switch to the MATLAB Command Window ([Ctrl Esc]) and type `cd c:\winsim` (or the directory you chose in part 5). This ensures that all generated code and associated files are stored in that directory. If you do not change directory from inside MATLAB, the generated code will be added to the MATLAB directory and may create unnecessary confusion.

7) **Generate and Build Realtime Code** Switch back to the SIMULINK drawing ([Ctrl Esc] or [Alt Tab]) and select Code and Generate and Build Realtime. This starts the compilation of the diagram in to C code that can be linked with WinCon. The output file for this operation has the extension `.wc1` . This means it is a SIMULINK controller that can link with WinCon.

8) Start the Controller From the SIMULINK drawing, select Simulation and Start. This starts WinCon, loads the controller for which you just generated code and starts the realtime controller. Note that this operation also starts a SIMULINK simulation which is automatically paused. You can also start and stop the simulation by typing `start` and `stop` from within the MATLAB Command Window (MCW). The `start` command automatically pauses the simulation but keeps the controller running. You may find this a more convenient method of starting the controller from MATLAB. Note that these commands start and stop the controller in the block diagram in the last window that you had clicked on before going to the MATLAB command window. Make sure you issue the command after you have clicked on the window of the controller you want to start. If WinCon is already up and the controller you want loaded, you can start the controller from any window by entering **[Alt Pause]** on the keyboard.

To stop the controller, you can either

- Stop the controller *from any Window* with the **[Pause]** key on the keyboard (this is the fastest and easiest way)
- Stop Simulation in the SIMULINK Window [Ctrl T] or issue `stop` from the MCW
- Stop the controller from the WinCon Main Window (WMW), click on the Traffic Light Button, or select File-Stop from the pull down menu.

You can also use the buttons that are in `winlib` (Figure 3.1). You can drag copies of these buttons into the SIMULINK diagram itself. double clicking on these buttons will start and stop the controller.

The following table lists all the possible methods of starting and stopping the controller:

From where	How	Condition
From SIMULINK Menu of Diagram	[Ctrl T] or Simulation Menu	SIMULINK up Strats WinCon, loads diagram RTC
From SIMULINK Diagram itself	Double click Start button Double click Stop button	SIMULINK up (Obtain Stop and Start from <code>winlib</code>)
From MATLAB Command line	Start Stop	Last SIMULINK diagram that was active
From WinCon	Traffic light Button Or File/Run Menu item	WinCon loaded
<i>From Anywhere</i>	<i>[Alt pause]</i> <i>[Pause]</i>	<i>WinCon loaded</i>

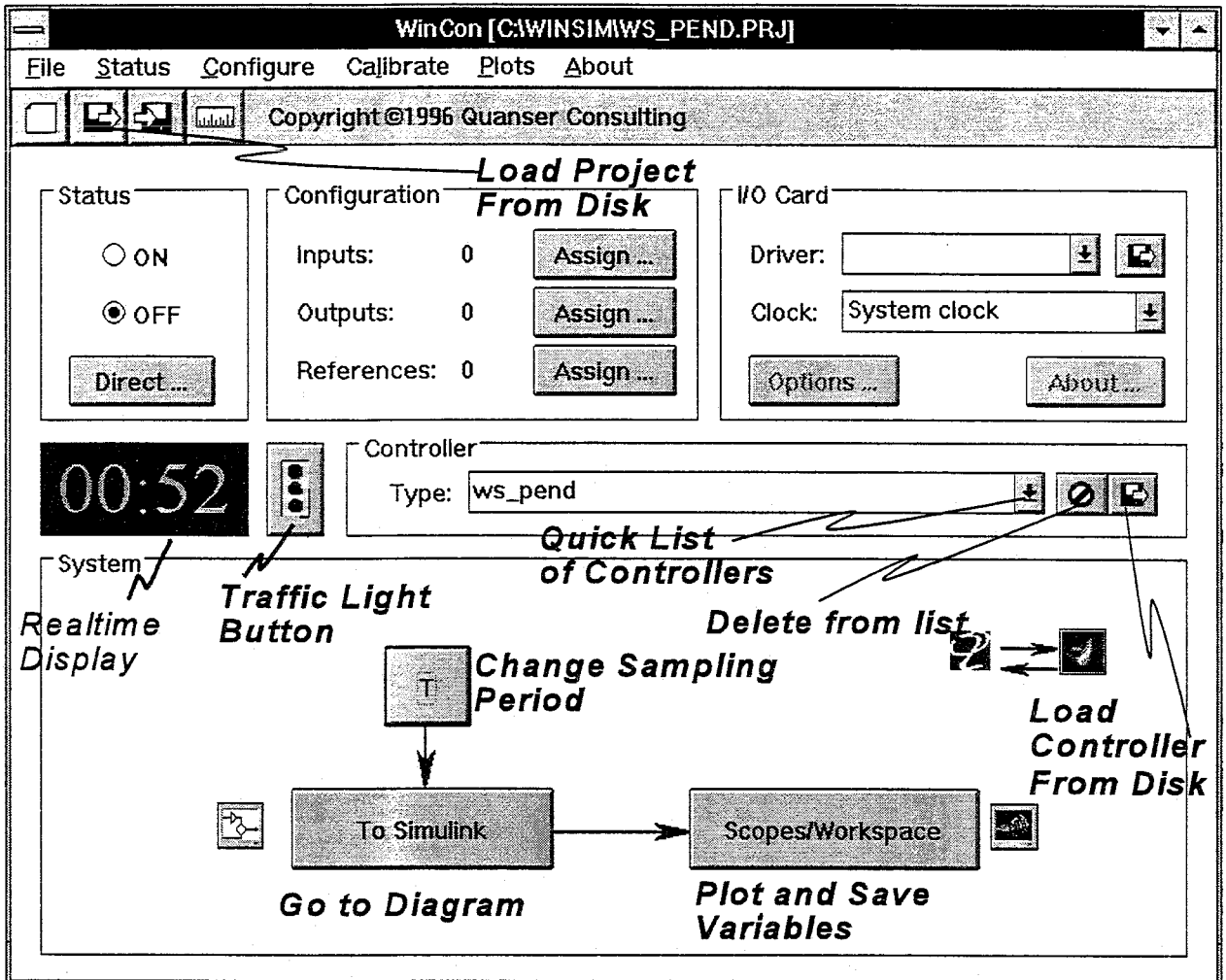


Figure 3.5 WinCon Main Window

9) Plotting realtime and saving data

"Scope" data selected in SIMULINK may be plotted in realtime by clicking on the Scopes/Workspace button in the WinCon main window. Select the variable you want to plot and click OK. If you want more than one variable plotted on the same graph (eg reference and output) click on the first variable then hold down the [Ctrl] key and click the left mouse button on the other variables you wish to plot. Once you are satisfied with the variable list click [OK].

Once the plot is showing you can alter some of the plot parameters such as Update Frequency, Buffer duration, axes etc... You can also select if you want plot windows to be a child of WinCon or be treated as an independent window accessible from the Windows Task List (Ctrl Esc). This is done using the Plot / Are Children menu in the WinCon Main Window.

If you would like to open another plot window, you may do so using the Scopes/Workspace button or by clicking on File - New in any active plot Window. From an active plot Window you may switch to the WinCon main window or any other plot that is in the background using the Window Menu in the Plot Window.

If you want to change the variables being plotted in a given Plot Window, click on File/Variables in the Plot Window and select the new variables you want to plot.

You can also save the drawn data in two formats: (.m and .mat). Select File from the Plot Window and Save. Select the directory (we highly recommend that you place the data files in a different directory eg c:\winsim\data) Select the type of file you want to save from the selection list. **Caution: If you select a ".m" type file for output, make sure you do not use the same file name as your controller as you will overwrite your controller diagram with the data file!** Enter a filename (no extension) and click [OK]. The ".m" file will contain an *ascii* file of the variables you plotted in WinCon as well as a MATLAB plot statement. You can then switch to MATLAB and enter the file name. This will automatically plot the data you saved. If the file is saved as a ".mat" file, it will be in binary format and you will have to **load** it into MATLAB using the MATLAB `load` command. The data will consist of individual vectors with the Scope names you had chosen. If the variable name has a blank in it, the blank will be replaced with an underscore(_). There will also be a variable holding the time vector for the data named T. Both of these plot features are very useful if you want to perform parameter estimation or make hard copies of the plot to include in a report.

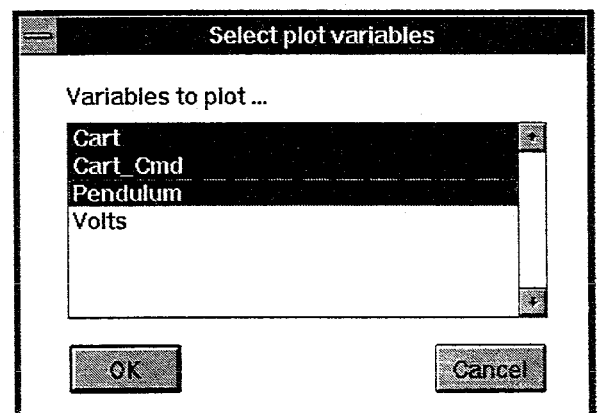


Figure 3.6 Selecting plot variables

In summary, the following features are available from a Plot window menu:

Under File

New Open a new plot window

Save Saves the plotted data in a .m or .mat file

Variables Select which Scope variables you want to plot

Close Closes the plot window

Under Update

- **Fixed Interval or Realtime:** Selects whether you want to plot in realtime or plot just one trace and hold the trace on the plot(single sweep).

- **Buffer (T_b) :** Is the duration of the data buffer for the plot. This is a circular buffer of duration T_b. Only the last T_b seconds of data are available for plotting and saving.

- **Frequency (P_{ts}):** The sampling period of the data buffer. The fastest frequency possible is the actual sampling frequency of the controller. P_{ts} may be larger than the actual sampling period (T_s) thus allowing decimation. You can then save a longer duration of data by compromising resolution. The maximum number of points per trace is 16000.

Under Axes

- **Auto Scale or Fixed:** Select whether or not the scale of the y axis is adjusted to fit the last sweep maximum and minimum. In Fixed mode, you select the scale.

- **Time (T_p):** This is the duration of the displayed plot along the time axis. T_p is always less than or equal to the buffer duration T_b. If T_p is less than T_b and you are plotting in Fixed mode, then a scroll bar appears under the plot that lets you scroll through the data over the last T_b seconds.

- **Grid** Self explanatory

Under Window

- **Legend** Self explanatory

- **Main Window** Takes you back to the WinCon main Window. A handy feature to help you navigate through many open windows.

- **Other plot windows list** Takes you to another open plot window. Also helps you navigate through a maze of plot windows.

Furthermore, if you want to access the plotting facilities straight from the SIMULINK diagram, you can drag and drop the [plot] button from winlib (Figure 1.1) into the SIMULINK diagram. Double clicking on this box is just like clicking on the [Scopes/Workspace] button in WinCon.

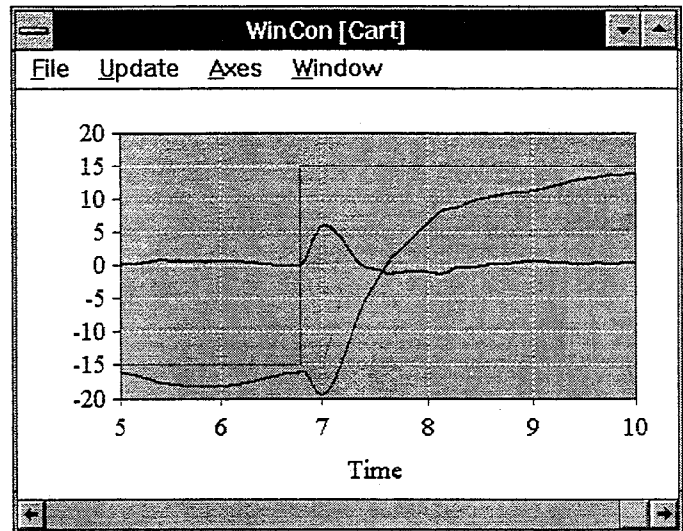


Figure 3.7 Sample WinCon plot window

10) Changing controller parameters

If WinCon and SIMULINK are up, and the loaded controller in WinCon is the same as the diagram in SIMULINK, then parameters changes in the diagram automatically take effect in the running controller. **It is usually good practice to stop the controller in SIMULINK, change the parameter and then start it again.** This is especially true if you are changing filter parameters drastically and the system you are controlling is sensitive to parameter changes. If the diagram is altered in any way (eg, cutting connections, adding blocks) you will need to regenerate realtime code as in step 7.

11) Saving and Running a WinCon Project

You may save a running WinCon controller as an independent project, including the status of all plots you had opened. Do this from the WinCon main window and select File/Save as. Select a directory and filename then click OK. You can then load this new project file (.prj extension) into WinCon in the future. You can start the loaded project from WinCon by entering [Alt Pause] or clicking on the Traffic Light Button (TLB) to the right of the Time display in WinCon. Starting the controller from WinCon, will start the controller that was originally saved during Realtime Code Generation. If you want to use a controller with different parameters, Stop the controller using the pause or TLB. Click the To SIMULINK button. This starts MATLAB, and loads the SIMULINK diagram of the controller you have loaded into WinCon. You can then Start from SIMULINK which will now start the WinCon Controller and will allow you to change controller parameters. Once you are satisfied with controller performance, it is advisable to generate the final realtime code so that the project you eventually save will contain the desired parameters. This way you can start WinCon with a project that contains your "final" controller.

As in any application, practice makes perfect. Try out a few simple systems (our examples) first before you start a serious project. Learn the sequence of operations, you'll eventually become an expert!

12) Notes on Step size In WinCon, you may select the source of the timer interrupt that controls the sampling frequency. For sampling frequencies below 1KHz, you may select the Windows System clock. This is done by selecting Clock, in the I/O Card option of WinCon (on the screen, top right). If you want to sample at a higher frequency, you have to use an external hardware clock tied to an interrupt line on the PC. MultiQ supports this option. If you have other cards, a WinCon driver must be written for that card. Contact Quanser Consulting in order to have a Clock driver written for your card. The card must be equipped with a programmable realtime clock that can be tied to interrupt lines.

Be careful when selecting the step size. Make sure that it is not too small. Read the section on **Maximum Sampling Frequency** under *Hints and Troubleshooting* at the end of the manual.

4.0 Examples

4.1 Simple loopback

This is an example that everyone should try. Not only do you practice the use of WinCon but you also test your A/D, D/A board. The example is a loopback system. You need to wire the output of channel # 0 on the D/A subsystem of your board to the input of channel #0 on the A/D subsystem on your board. This is shown in Figure 4.1.1

The "controller" will simply apply a sine function to the D/A output and will measure the voltage from the A/D. No other hardware is required for this example.

1) Start MATLAB for windows and type `cd c:\winsim` . Type `ws_lpbk` This loads the diagram shown in Figure 4.1.2. You could have generated this diagram yourself using SIMULINK.

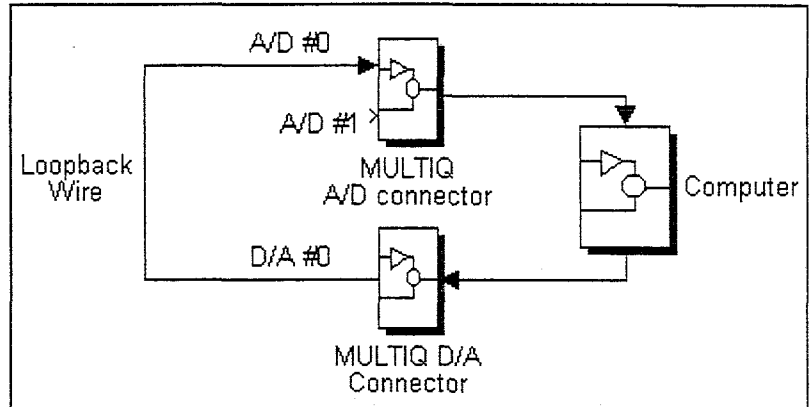


Figure 4.1.1 Wiring for loopback experiment

2) If you have a Multi board you can skip to step 4

3) If you do not have a Multi board installed in your system, switch to the MATLAB Command Window and type `doslib`. Double click on the card you have from the list of cards available in `doslib`. Copy one A/D block and one D/A block to the `ws_lpbk` diagram. Delete the Multi A/D block in the `ws_lpbk` diagram and replace it with your board's A/D block. Delete the Multi D/A block and replace it with the D/A block for your card. Make sure the drivers are configured for the correct base address and channel. You can select the A/D board you are using from the `doslib` or `winlib` Simulink blocks.

4) Select Code from the menu and click on Generate and Build Realtime . This generates the realtime code for the diagram. Wait until the compilation is complete. A DOS window is automatically opened and upon completion the message :

```
* * * * * Make of ws_lpbk.wcl is complete * * * * *
```

is printed in the DOS Window. You can then close the DOS window.

5) Click on Simulation/Start in the `ws_lpbk` SIMULINK window. This starts WinCon, loads the compiled controller and starts running it at the sampling frequency specified in Real Time Options in SIMULINK. Alternatively, you could switch to the MATLAB Command Window and type `start` to start the controller and automatically pause the simulation. (See section 3.8).

6) Switch to WinCon using [Alt Tab]

7) Click on Scopes/Workspace. The names of scopes defined in `ws_lpbk` will appear in a list. Plot both variable on the same graph by selecting [Click] Command then [Ctrl Click] on Measured and then select

[OK]. Note that the commanded voltage has an amplitude of 10 volts. However the saturation block limits it to 5 volts. Therefore the real output to the D/A is a clipped sinusoid at 5 Volts. This voltage is measured by the A/D (due to loopback wire) and displayed under the variable Measured. The plot will display a 10 V peak sine wave and a clipped sine wave. The command and the measured value should otherwise superimpose. If we apply a voltage smaller than 5 Volts p-p at the command, then the measured and command would (should) match exactly.

8) Click To SIMULINK from WinCon. This takes you to SIMULINK. Stop the controller [Ctrl T] . Click on the signal generator and change the source to a triangular wave. Keep the Plot Window of WinCon open so you can monitor the changes you are making. Start the Controller [Ctrl T] and note that the change you made took effect in WinCon by observing the plot. Change the signal generator parameters and observe that you are changing parameters in realtime.. without interruption in control !

9) Select a signal source different from the original sine wave (eg. triangular) , start the controller and switch to WinCon. Plot the variables as before. With the plot active, save this "controller" as a project by selecting File - Save as from the WinCon main window and select the filename as c:\winsim\ws_lpbk

10) Click on To SIMULINK. Stop the controller (or issue stop from MATLAB). Save the changed diagram (with a triangular wave as signal source). Close SIMULINK. Close MATLAB. Close WinCon.

11) Open WinCon from scratch. From File/Open load c:\winsim\ws_lpbk.prj. This is the project you

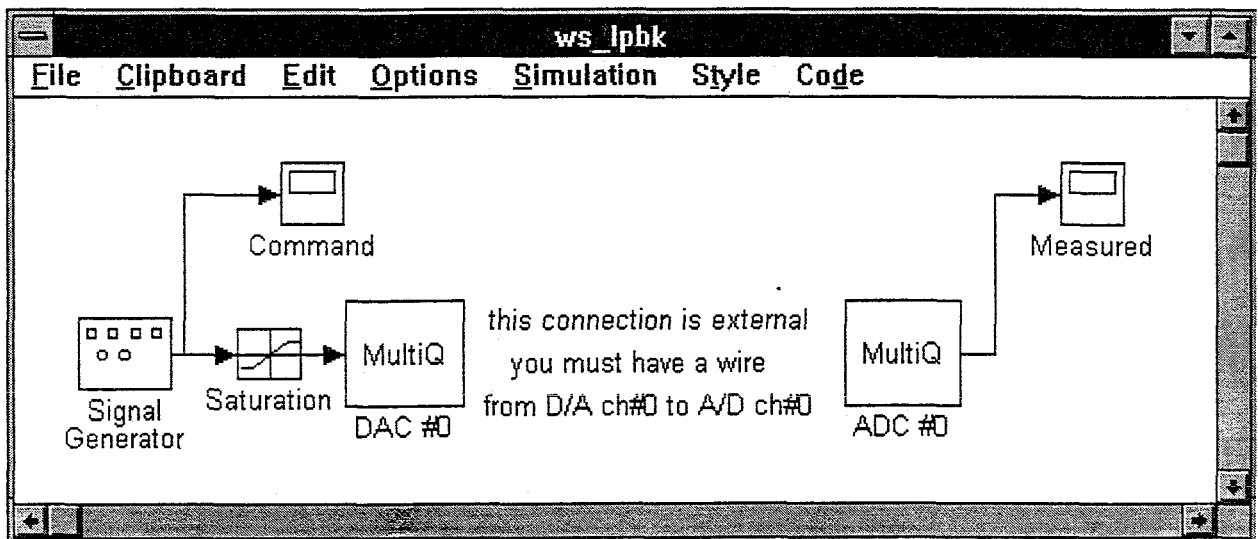


Figure 4.1.2 SIMULINK diagram of loopback "controller"

saved in step 9. Start the controller directly from WinCon. What are the plots showing? Triangular or Sine waves? It *will be sine waves* because the compiled code in step 4 was using a sine wave! Stop the controller using the Pause button. Click on To SIMULINK. This starts MATLAB and loads ws_lpbk.m diagram into SIMULINK. Click on Simulation - Start. Switch to WinCon. What do the plots show now? A *triangular wave* because you just loaded the parameters from the SIMULINK diagram into the running controller! *This is where you have to be cautious. WinCon runs realtime generated code that can only be changed via SIMULINK or re-compiling.*

4.2 PID Controller

This example demonstrates a PID controller that positions the output of the Servo plant shown in Figure 4.2.1. Start MATLAB from Windows, change directory to `c:\winsim` and type, `ws_srv_r`. This brings up the diagram of the controller shown in Figure 4.2.2. (Note this example is shown in Win95).

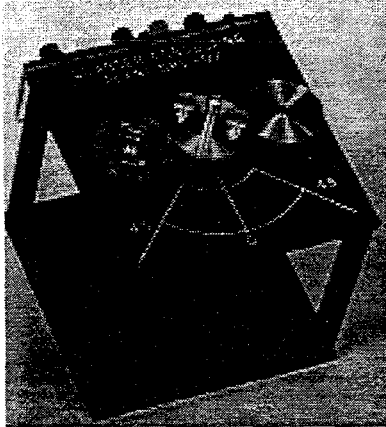


Figure 4.2.1 Rotary position Servo plant

The controlled output of the Servo Plant is measured using the A/D channel #0. The command to the system is generated from the Command Generation block. The difference between the command and the actual angle is the error signal which is multiplied by a proportional gain K_p . The error is also integrated and multiplied by a gain K_i . The measured variable (eg motor angle) is also differentiated using a high pass filter and multiplied by a gain K_d . The three gains are the PID gains of the controller. The sum of these terms is applied to the D/A output channel #0. The signal from the Command Generation block could be anything you want. In this example it is a signal generator applying a square wave of ± 45 degrees. You may replace this block with another A/D channel for example which has a potentiometer attached to it. The controlled system will track the signal generated by the command generation block. The PID gains in the Figure could be constants or variables defined in the MATLAB Command Window. If you enter variable names in the PID gains (or in any other block for that matter) they must have a value assigned to them before you can generate the

realtime code. Once you generate the code, the controller (.wcl extension file) will have these gain values hard coded into it. Changing the values of the variables in the MATLAB command window will not immediately change the value in a running controller. You will have to either stop and start the simulation or, open the dialogue box for the block which uses the variable name `init` and then click [OK]. This can be done while the simulation is running.

To start this controller perform the following operations:

- 1) Start MATLAB in Windows (If it is not already up. Do not start multiple copies)
- 2) type `cd c:\winsim` in the MATLAB Command Window
- 3) Load `ws_srv_r` by typing `ws_srv_r` in the MATLAB Command Window. This loads the SIMULINK drawing shown in Figure 4.2.1
- 4) Click on Code. Select Generate and Build Realtime from the Pull Down Menu. This starts the real time code generator and compiles the code to be linked with WinCon.
- 5) Once the code has been generated, you will see a DOS window with the message:

```
* * * * * Make of ws_srv_r.wcl is complete * * * * *
```

You can close the DOS window to minimize clutter. MATLAB does not close the DOS windows automatically.

- 6) Click on Simulation/Start.
- 7) The system starts WinCon, configures it and starts running the controller in realtime.
- 8) You can now plot and change gains as you please. You can save the project in WinCon so that the

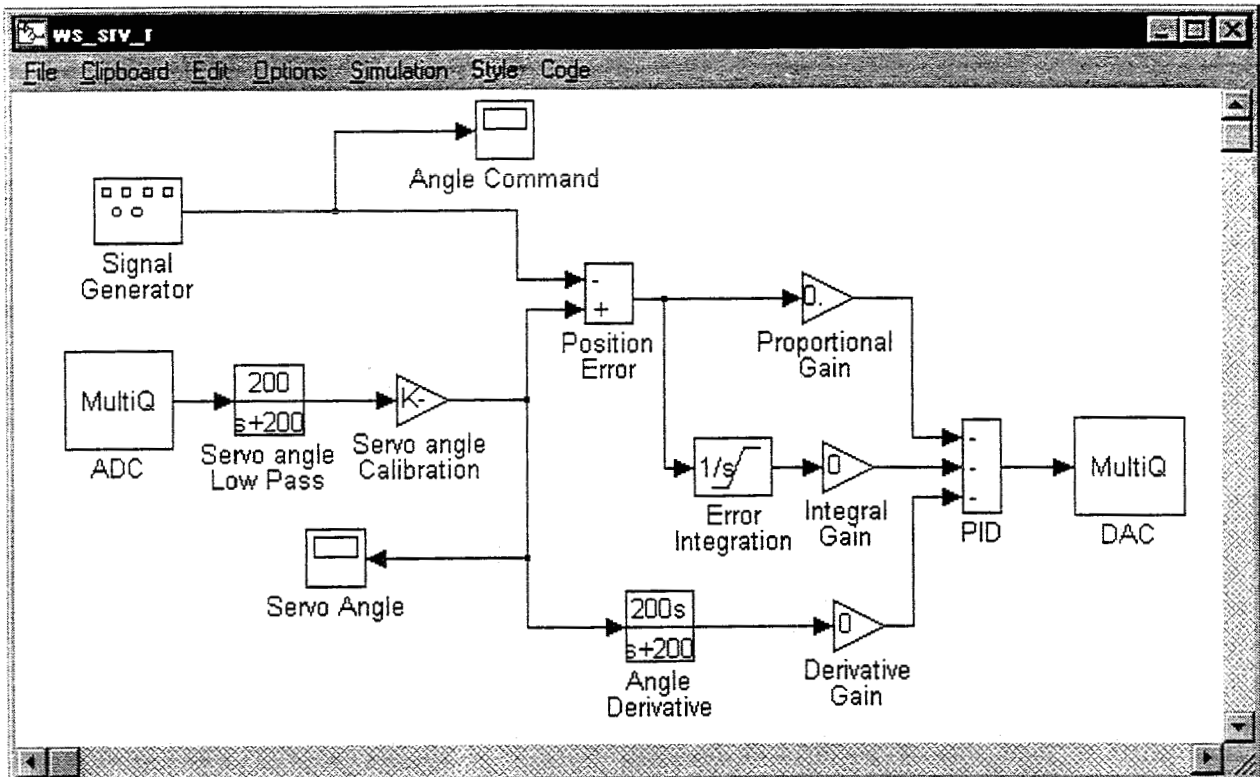


Figure 4.2.2 SIMULINK controller of position servo (PID)

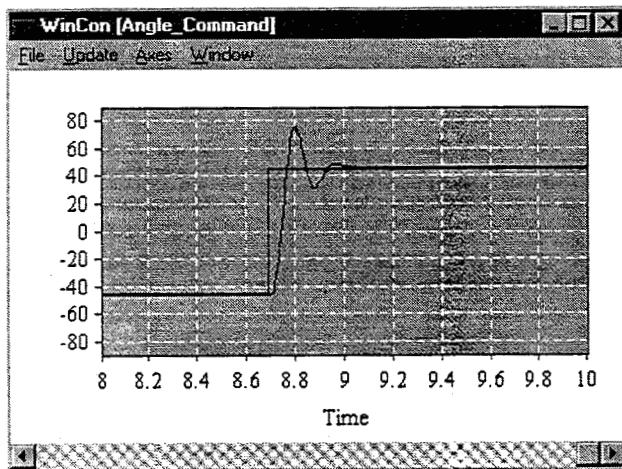


Figure 4.2.3 Step response when derivative gain is zero (WinCon plot)

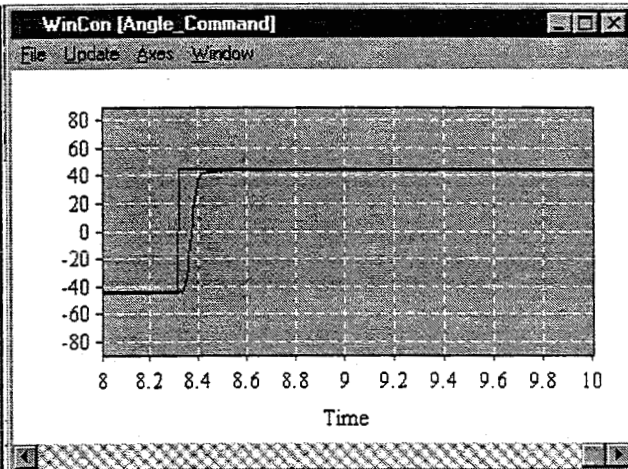


Figure 4.2.4 Step response when derivative gain is 0.002

controller can be started from WinCon (without SIMULINK) and run from WinCon. Figure 4.2.3 shows the response of the output to the square wave input when the derivative gain is zero. Note the overshoot and ringing. Figure 4.2.4 shows the response when the derivative gain is set to .002 (while the controller runs). Note the improved behaviour.

4.3 Inverted Pendulum

This example covers the Inverted Pendulum IP-01 shown in figure 4.3.1. The system is wired as shown in Figure 4.3.2.

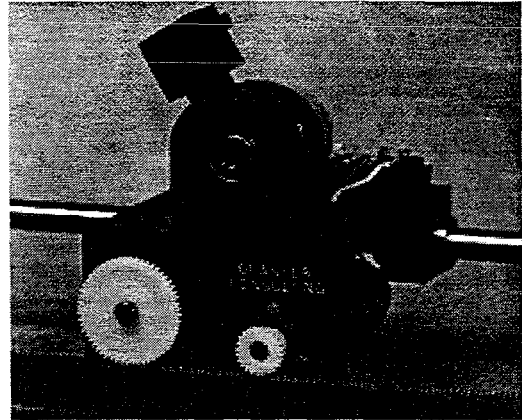
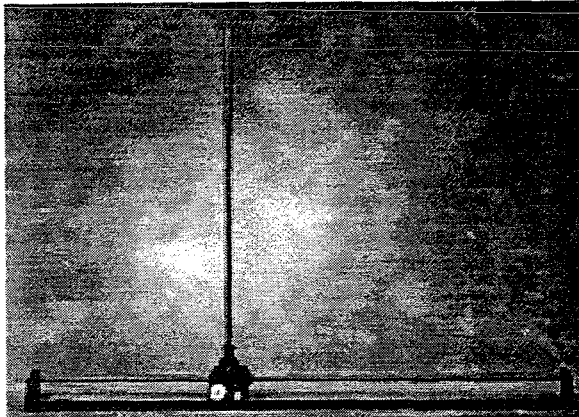


Figure 4.3.1 Inverted pendulum hardware

The inverted pendulum has two sensors, one for the cart position and one for the inverted pendulum angle. These sensors are attached to the MultiQ A/D inputs channel 0 & channel 1 respectively using the Quanser Quick Connect Module. The system is driven by a DC motor which is powered via a power amplifier on the power module PA-0103. The input to the power amplifier is obtained from the D/A channel #0 on the MultiQ board.

The required controller is a state feedback controller that feeds back the following voltage to the motor:

$$V_m = [k_1 \quad k_2 \quad k_3 \quad k_4 \quad k_5] \begin{bmatrix} e \\ \alpha \\ \dot{x} \\ \dot{\alpha} \\ e \end{bmatrix}$$

The control system design is beyond the scope of this manual. For details see "A comprehensive and modular laboratory for control system design and implementation" by Quanser Consulting.

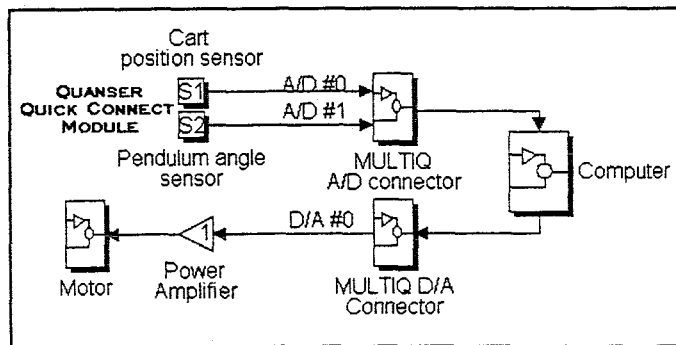


Figure 4.3.2 Wiring the inverted pendulum

Load the controller `ws_pend` into SIMULINK.

The diagram shown in Figure 4.3.3 will appear. The measurements are voltages from the sensors and are measured using A/D channel#0 and A/D channel #1. These are applied to bias removal blocks and then fed to a calibration and differentiation block. The output of this block results in the states $[x \quad \dot{x} \quad \alpha \quad \dot{\alpha}]$. A signal generator generates the commanded cart position which is subtracted from the actual cart position to obtain the cart position error e . These are then fed to the state feedback controller which computes the integral of the error and applies the desired gains k_1 to k_5 to obtain the applied voltage to the motor. This

values is then sent to the D/A channel #0 on the MultiQ to drive the motor through the power amplifier

4.3.1 Bias removal blocks

The two bias removal blocks are "grouped" SIMULINK blocks. Viewing one of these blocks (double click on the block) reveals the structure shown in Figure 4.3.4. This block has two inputs. In_1 is the voltage measured from the sensor while In_2 is from the clock which keeps track of time from the moment the simulation starts. The clock controls the position of a SIMULINK switch. As soon as the time is greater than 0.1 seconds (set in the dialogue box of the switch), the output of the switch is the value applied at the first line. In the first 10 mseconds of the simulation, the output of the switch is the input applied at line 3. So, for the first 10 msec., the output of the block is $(In_1 - In_1) = 0$. When the switch trips over, the output of the switch is maintained to the last value before it switched over. Thus, the block defines the zero position of the device to be the position it was in at $t = 10$ msec. The purpose for this block, is so the user can hold the pendulum upright and the cart at $x = 0$. The user then starts the controller which for the first 10 msec does not do anything but measure the bias on the sensors. It then uses the measured bias to obtain the actual angle and cart positions.

4.3.2 Calibration and Differentiation block

This block converts the the cart position voltage and pendulum angle voltage to the appropriate units. The cart sensors measures 8 cm per volt while the pendulum angle sensor measures 14.5 degrees per volt. The voltages are first low-pass filtered and then multiplied by the appropriate calibration constants as shown in Figure 4.3.5. After converting the voltages to the desired units, the variables are applied to Scopes, which will be available in WinCon for plotting. The two signals are then fed to limited differentiators to obtain the cart velocity and pendulum angular velocity. Note that we use limited differentiators rather the "pure" derivatives to reduce noise.

4.3.3 State Feedback Block

The last block before the output block is the state feedback controller which implements the feedback gains to obtain the voltage that will be applied to the controller. The system requires five gains which are implemented as "sliders" in SIMULINK. While the controller is running you can click on the sliders to change the gains during operation. Note that the integrator is a "limited integrator" block which we suggest you always use instead of a regular unlimited integrator. This way, if the integrator reaches the maximum value it will stop integrating and will immediately start integrating in the opposite direction when the input signal changes sign. This is a more effective method of implementing an integrator in a real system. Note that the integrator state is automatically reset whenever the controller is stopped and re-started. The output of this block is fed directly to the MultiQ D/A block. The channel of the D/A is selected in the D/A dialogue box.

4.3.4 Safety Stop As a safety feature, the controller is automatically turned off if the pendulum angle exceeds 10 degrees in either direction.

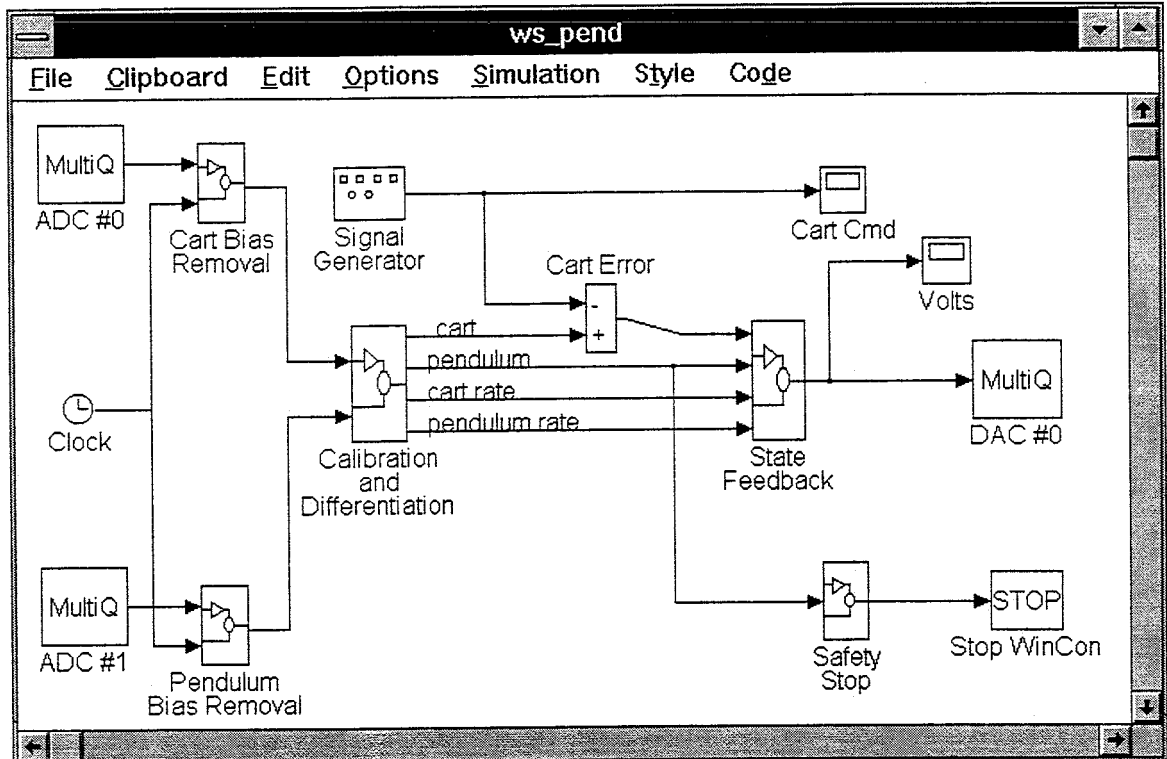


Figure 4.3.3 SIMULINK controller of Inverted pendulum

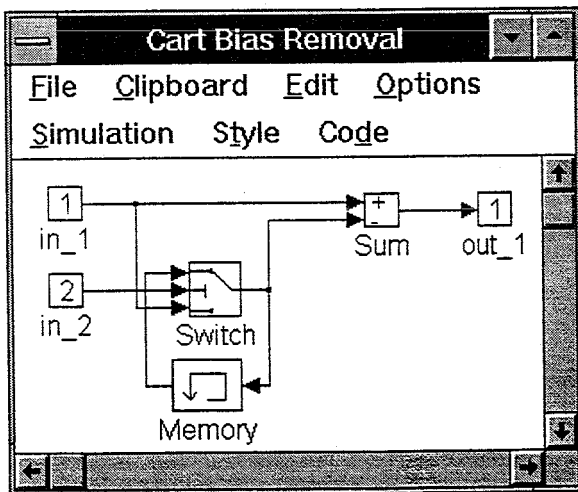


Figure 4.3.4

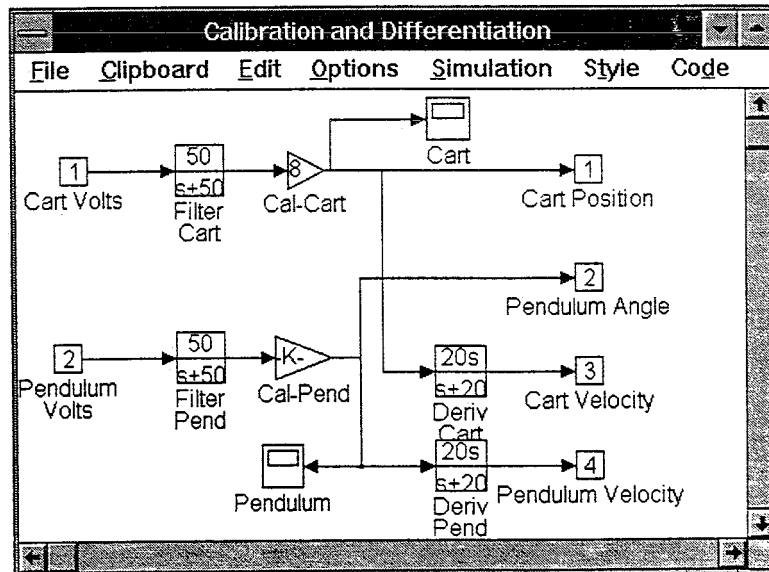


Figure 4.3.5

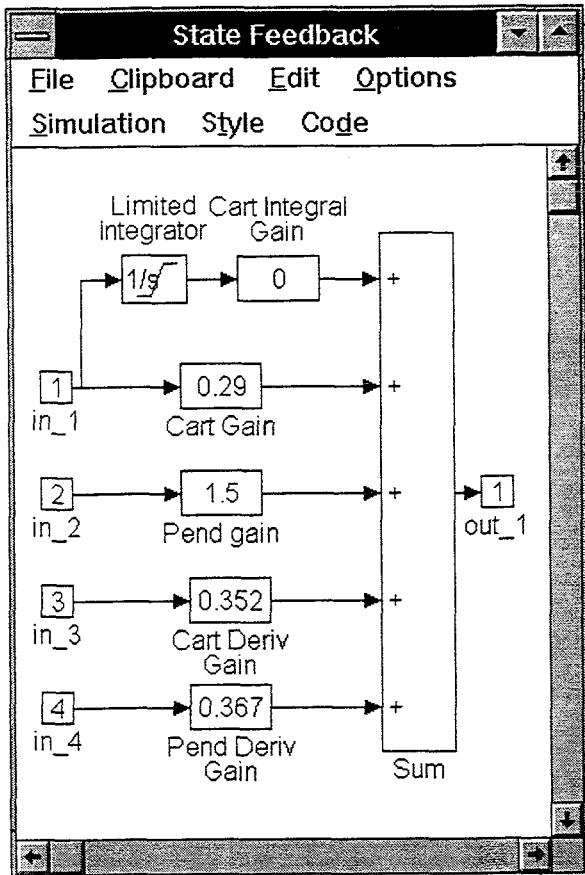


Figure 4.3.6

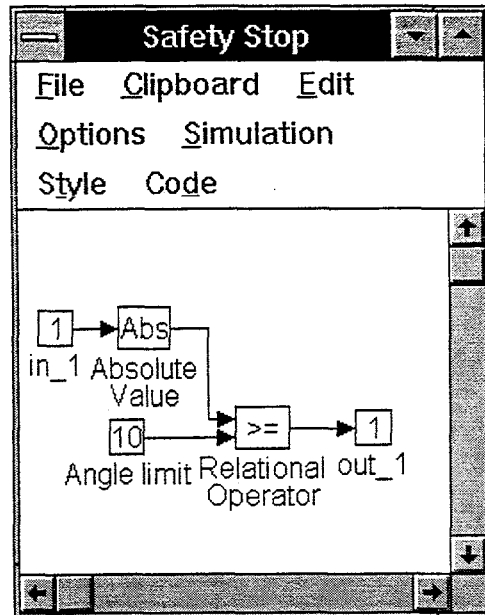


Figure 4.3.7

4.3.4 Running the controller

To start this controller perform the following operations:

- 1) Start Matlab in Windows (If it is not already up. Do not start multiple copies)
- 2) type `cd c:\winsome` in the Matlab Command Window
- 3) Load `ws_pend` by typing `ws_pend` in the Matlab Command Window. This loads the SIMULINK drawing shown in Figure 4.3.3
- 4) Click on Code. Select Generate and Build Realtime from the Pull Down Menu. This starts the real time code generator and compiles the code to be linked with WinCon.
- 5) Once the code has been generated, you will see a DOS window with the message:

```
* * * * * Make of ws_pend.wcl is complete * * * * *
```

You can close the DOS window to minimize clutter. Matlab does not close the DOS windows automatically.

- 6) Hold the pendulum upright and the cart in the middle of the track.
- 7) Click on Simulation/Start.
- 8) The system starts WinCon, configures it and starts running the controller in realtime.

9) You can now plot and change gains as you please. You can save the project in WinCon so that the entire controller can be started from WinCon (without SIMULINK) and run from WinCon.

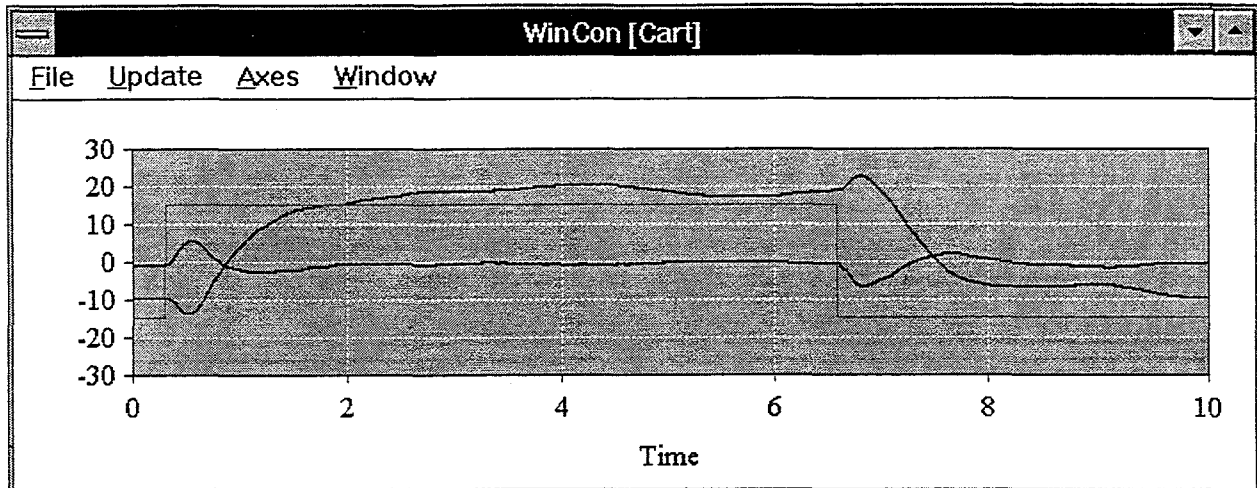


Figure 4.3.7 Plots obtained in realtime from WinCon

Figure 4.3.7 shows the WinCon plots that are obtained. The variables plotted are the signal generator, the cart position and the pendulum angle. Note the non-minimum phase behaviour in cart position.

4.4 Linear flexible joint frequency response

In some cases, you may want to evaluate the frequency response of a system. This example demonstrates the linear flexible joint system (LFJC) and its controller performance using a sine-sweep signal. The LFJC consists of the Inverted pendulum cart (IP-01) coupled to a load cart via a spring as shown in Figure 4.4.1. The input to the system is the voltage applied to the pendulum cart while the output is the position of the load cart. Note that the load cart (on the right) is not powered. The controller block diagram is shown in Figure 4.4.2. It consists of a state feedback controller that feeds back the positions of the two carts and their respective velocities. The commanded position is subtracted from both cart positions in order to maintain the length of the spring at rest. The interesting feature in this system is that you can change the gains from the feedback cart as shown in Figure 4.4.3 which is an expansion of the block named Control. By setting the two gains from the load cart to zero, we are not using output feedback and positioning the output cart by simply positioning the input cart and letting the spring do the rest. A better controller is obtained by feeding back the position and velocity of the output cart as well (non-zero gains). In order to evaluate the controller, we could apply square wave inputs to the command end look at the overshoot and damping. In this example however we will apply a sine wave of varying frequency and observe the resonance in the system (almost a bode plot.. but not quite).

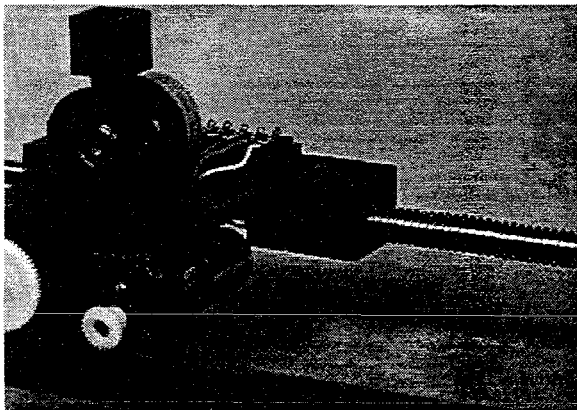


Figure 4.4.1 Hardware for flexible joint experiment

The Command is generated from the Chirp block, which when unmasked shows the diagram in Figure 4.4.4. A sawtooth signal is applied to a MATLAB function block which generates a sine function whose frequency depends on the value of the sawtooth signal. The sawtooth is held at zero for the first and last five seconds of the period. The rate of change of the chirp signal is selected to be unity per second and is thus changing like time. The reason we use a sawtooth signal instead of the Time block is that we want to repeat the sine sweep every 30 seconds. Figure 4.4.5 shows the input command (chirp signal) and load cart position when the feedback gains from the load cart are set to zero. Note the resonance in the system. Using LQR designed gains for the system, we can dampen the response to obtain the plots shown in Figure 4.4.6. The FFT's of these signals are compared in Figures 4.4.7 and 4.4.8. All of the plots

LFJC1

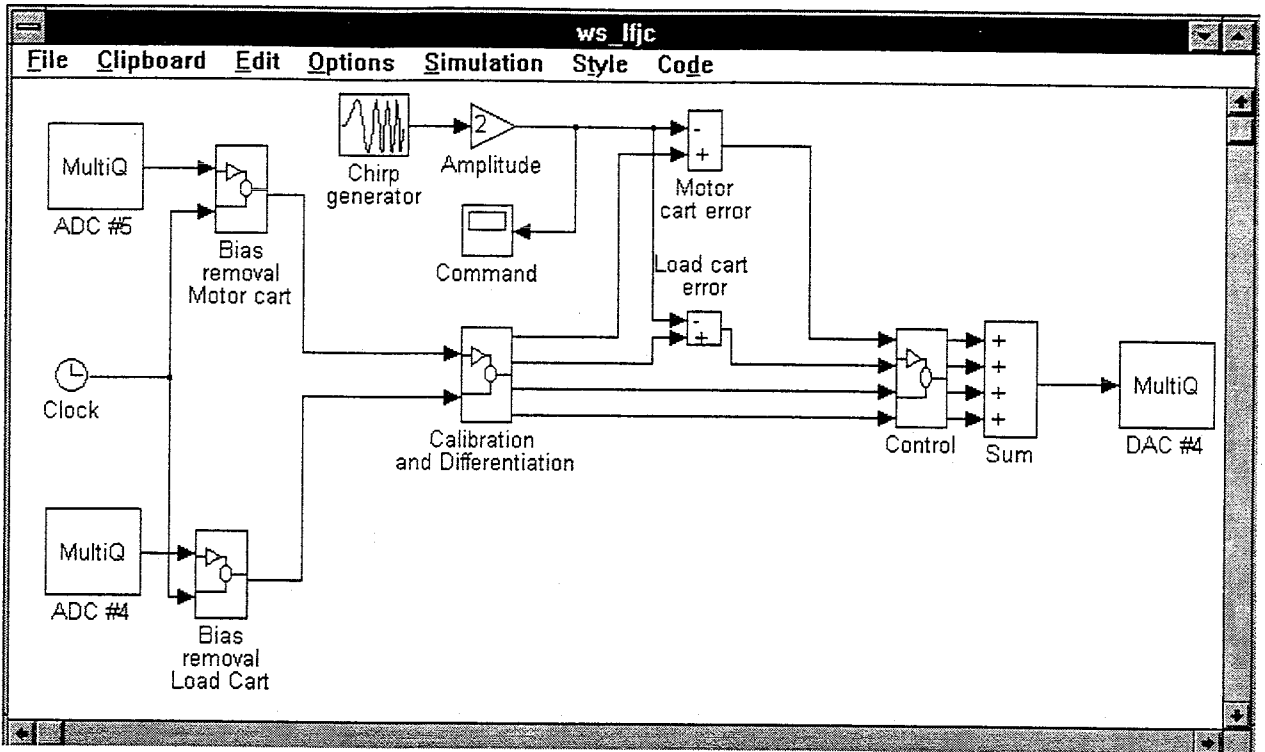


Figure 4.4.2 Controller for flexible joint

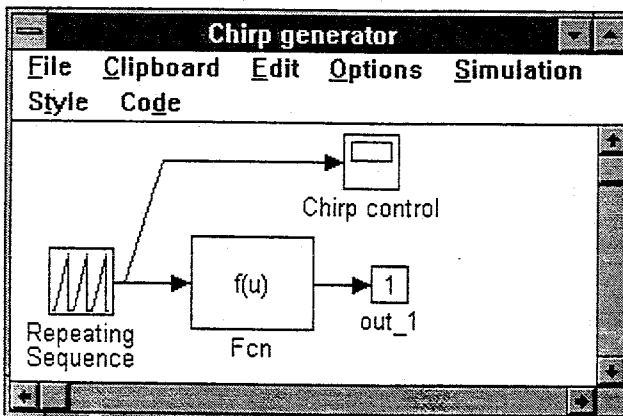


Figure 4.4.3 Chrip generator

are MATLAB plots obtained from data saved during the controller execution.

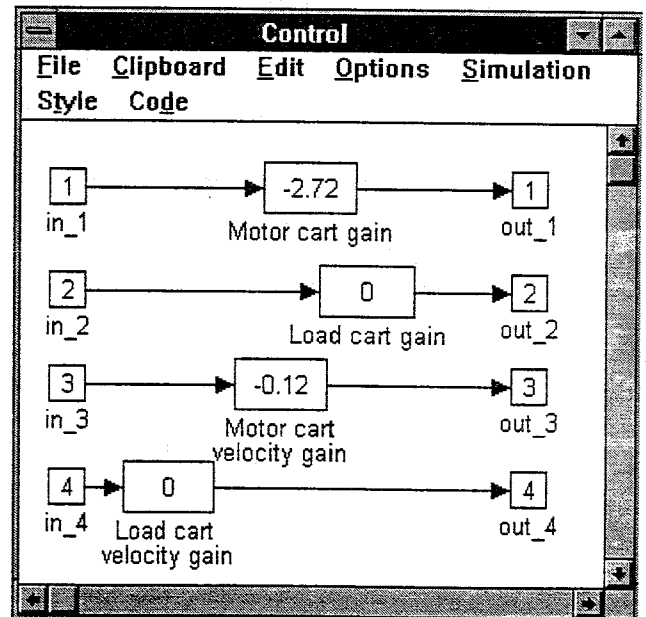


Figure 4.4.4 Feedback gains

2 MASSES & SPRING EXAMPLE

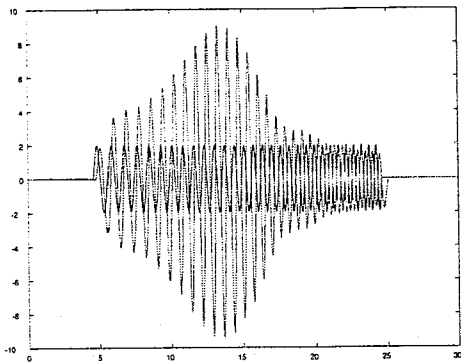


Figure 4.4.4 Chirp signal and load cart position when $K_2=K_4 = 0$

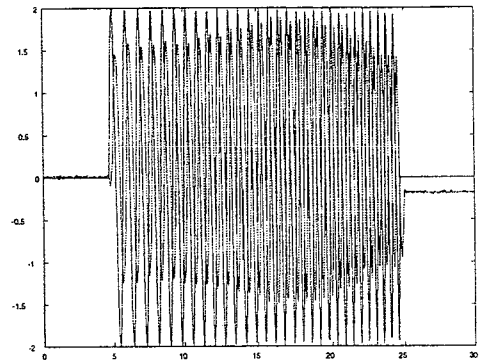


Figure 4.4.6 Chirp signal and cart output when K_2 and K_4 have been designed using LQR

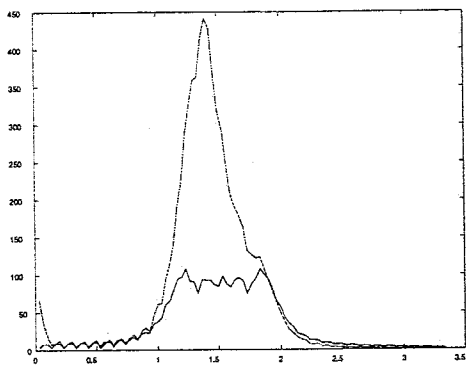


Figure 4.4.7 FFT of signals in Figure 4.4.5

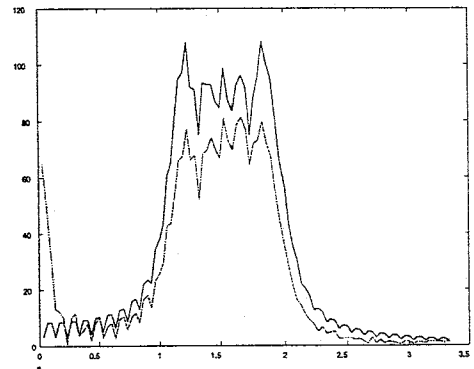


Figure 4.4.8 FFT of signals in Figure 4.4.6

4.4 3 DOF Helicopter

The 3 DOF helicopter experiment shown in Figure 4.4.1 is a prototype of a new Quanser consulting experiment under development. The **preliminary controller** is shown in Figure 4.4.1 and the associated blocks are described below. The controlled variables are the Pitch, Travel and Elevation of the helicopter which are measured via encoders on the apparatus. The outputs of the controller are voltages applied to two motors, namely the Front motor and the Back motor, through D/A channels #0 & #1. The system is commanded via a joystick which generates two analog voltages measured via A/D channel #0 and A/D channel #1.

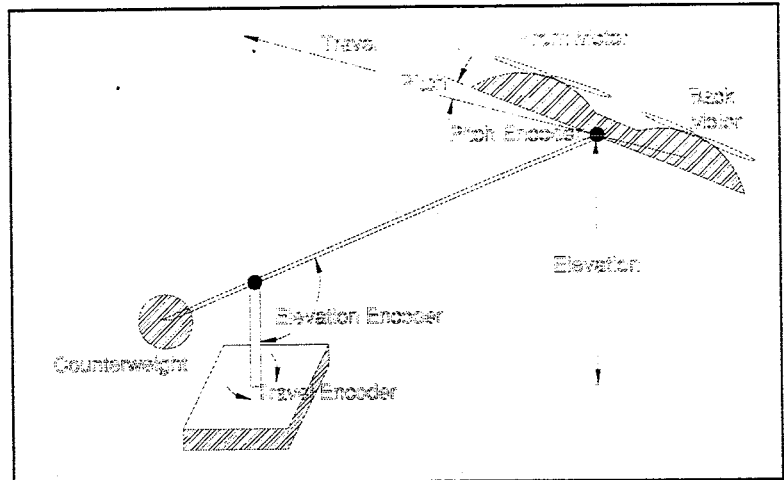


Figure 4.4.1 Diagram of 3D helicopter experiment

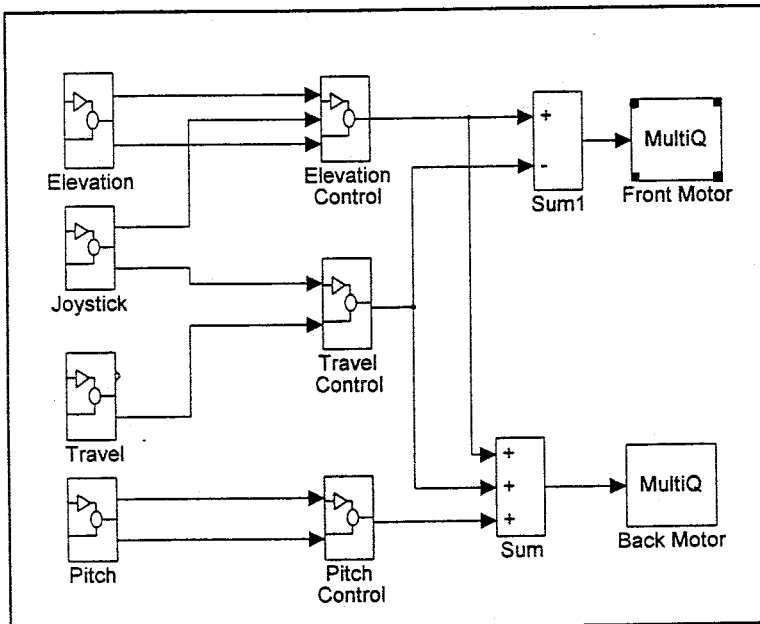


Figure 4.4.1 3DOF helicopter controller (file ws_heli3.m)

4.4.1 Joystick block The joystick block measures two analog voltages and after filtering and calibration, generates an elevation command and a travel rate command. Note that the elevation command is generated by integrating the signal, while the Travel rate command does not integrate the signal. A deadband is applied to the signals in order to eliminate jitter.

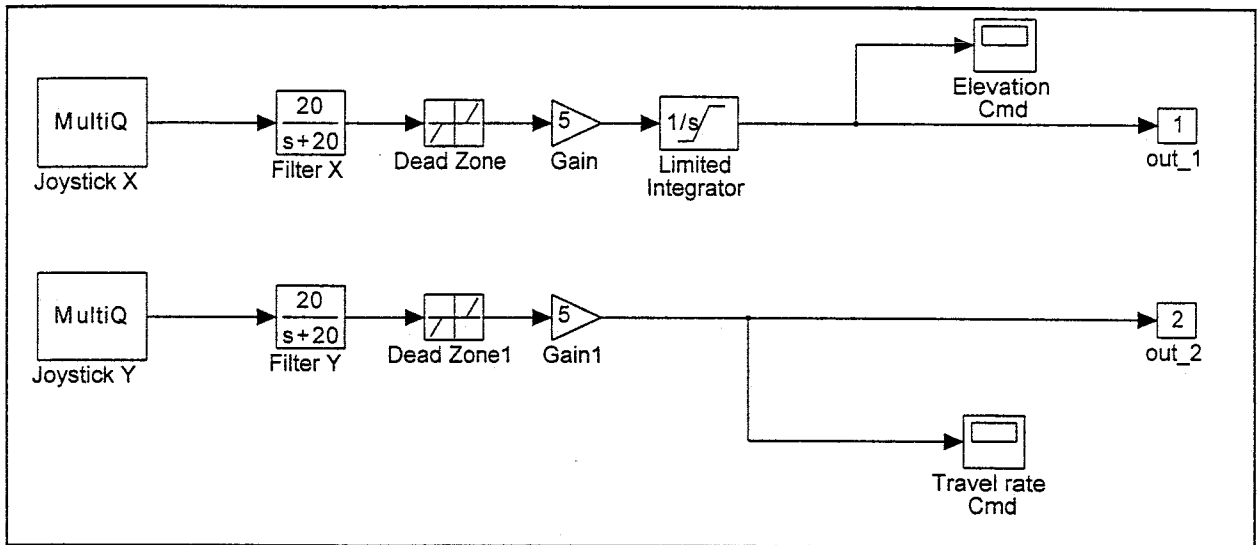


Figure 4.4.2 Joystick block of helicopter Controller

4.4.3 Elevation, Travel and Pitch blocks

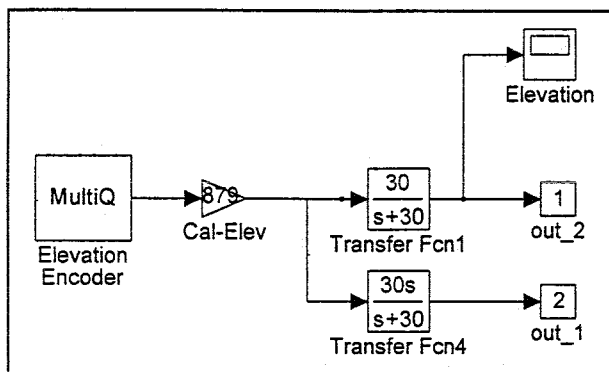


Figure 4.4.3 Expanded Elevation block

These block measure the three controlled variables and low pass filter them as well as generate their derivatives using a limited differentiator (high pass filter). One of the blocks is shown expanded in Figure 4.4.3.

The encoder measurement is obtained from the MultiQ board, which is then multiplied by the calibration constant. The block also applies the measured value to a "scope" which will be accessible from WinCon.

4.4.4 Controller blocks

These blocks generate the voltages that will be applied to the two motors. Each block is a controller using one of the measured variables and its command. Note that there is no command for the pitch axis. The controller tries to maintain the pitch horizontal. The travel rate command, causes disturbances in pitch, which in turn result in travel.

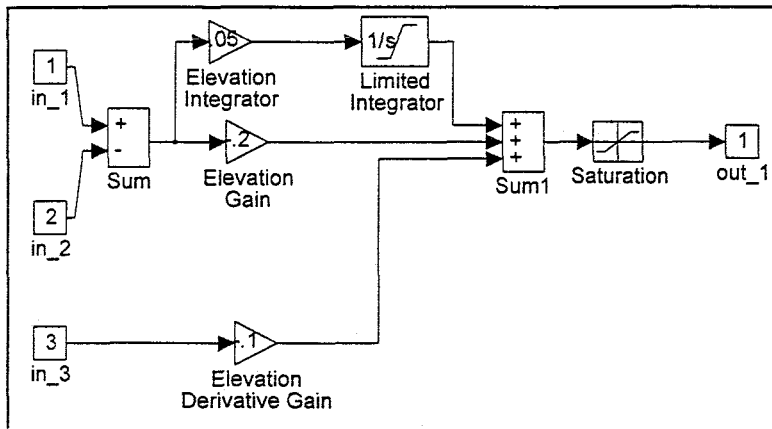


Figure 4.4.4 Elevation Control block

The voltages generated from the control blocks are then sent to summers which distribute the voltages to the two motors in the system. Note that this is a MIMO system but the controllers are designed as SISO systems.

Figure 4.4.7 shows the response of the Elevation axis to a commanded elevation derived from the joystick input.

Figure 4.4.8 shows the travel rate response to the commanded travel rate. The variation in pitch during this manoeuvre is shown in Figure 4.4.9

In order to run the controller follow the steps given in the previous examples.

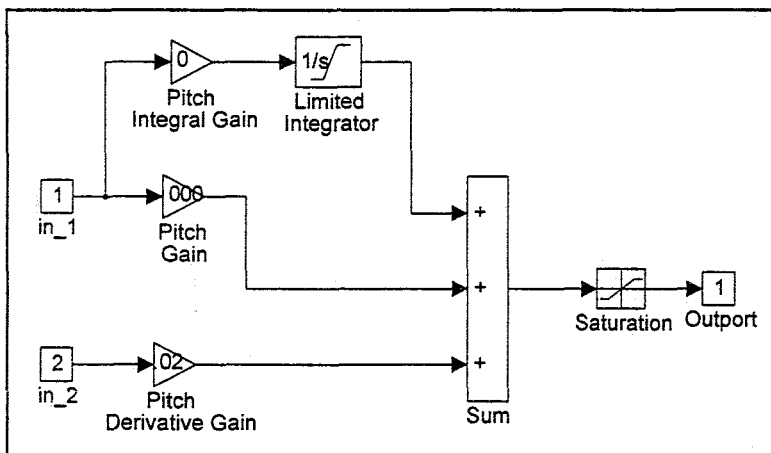


Figure 4.4.5 Pitch control block

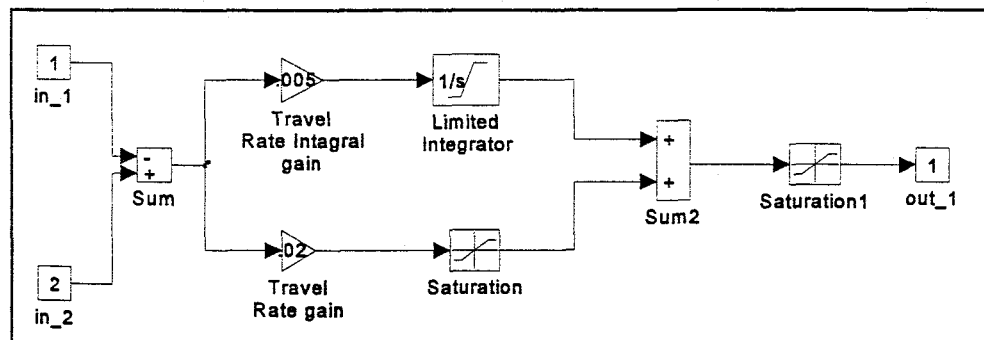


Figure 4.4.6 Travel control block

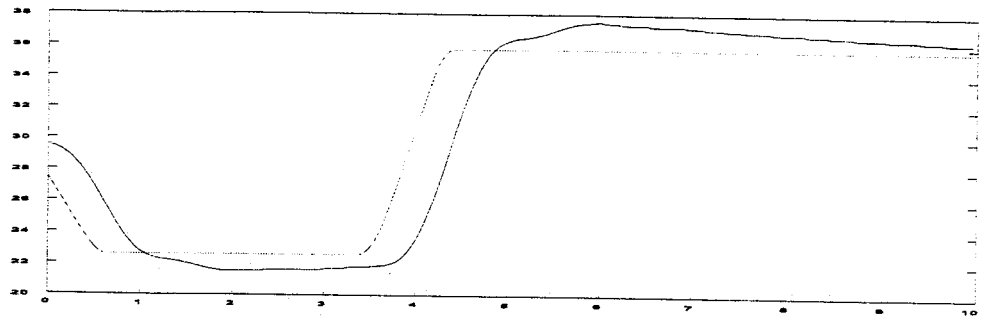


Figure 4.4.7 Elevation response to an elevation command

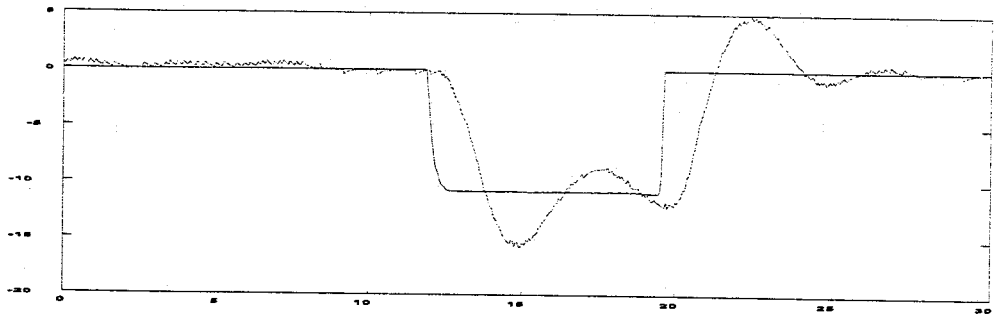


Figure 4.4.8 Travel rate response to an travel rate command

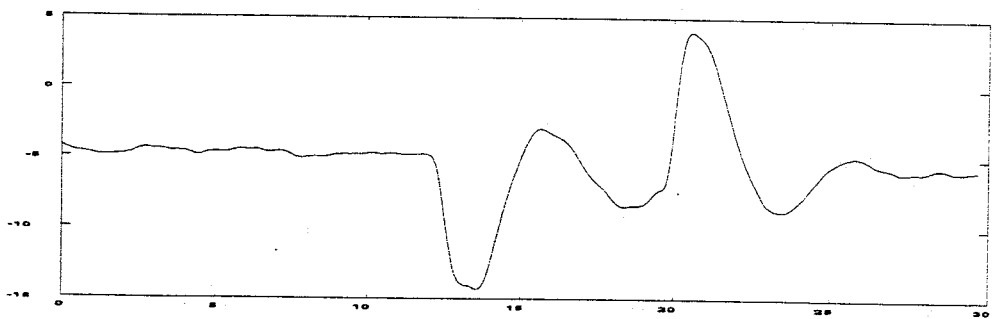


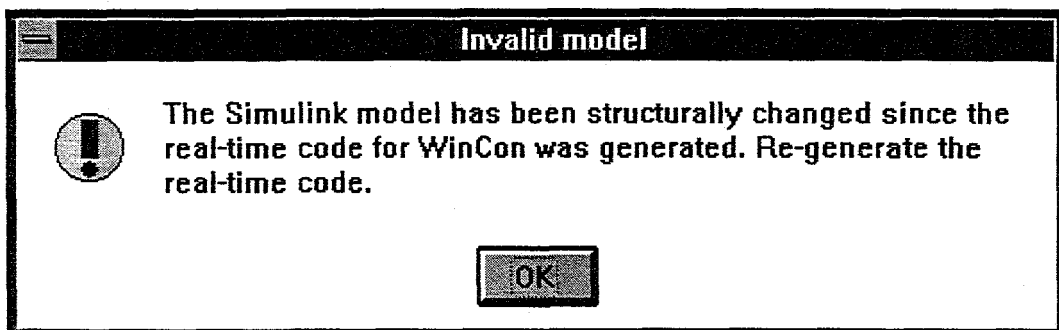
Figure 4.4.9 Pitch reaction to travel rate command shown in Figure 4.4.8

5.0 Hints and Troubleshooting

5.1 Discrete and multirate blocks

Discrete time transfer functions have a Sample time parameter which must be entered when the block is defined. If the sample time is shorter than or equal to the Step size selected under Realtime Code Options, then the block is executed at every Step (size). If the Sample time is longer than the Step size, then you have defined a multirate system. The discrete block will then be evaluated at an integer multiple of the Step size closest to the Sample time you have defined when you generated the realtime code. This means that if you change the Sample time of a discrete block during a simulation, it will not take effect. The reason for this is that you have already generated code which fixes the Sample time for that block. Also note that the coefficients computed for a discrete transfer function, must be based on the Sample time you chose and not the Step size, [eg converting from continuous time to discrete use `c2dm(nmu, den, Sample_Time)` and not `c2dm(nmu, den, Step_size)`]. Otherwise, the coefficients will not be compatible with the actual implementation. Normally you would select the Sample time of a discrete block to be the same as the Step size you selected for Real-time options unless you want a multirate system.

5.2 Parameter changes and the message



If you change connections in the diagram or modify the number of blocks, you will need to re-compile the code. This is also true if you change the dimensions of a matrix or vector which is being used in a SIMULINK block. For example, if you are using a numerator for a transfer function `num = [.2 3.1]` you decide to change it to `num = [3.1]` in the MATLAB Command Window, this is interpreted as a change in the block diagram. You may try to change it to `num = [0 3.1]` but you will still get the message that a change has occurred. This is because SIMULINK detects "0" as a no connection and a "1" as a connection. For this reason, if you want to change a value to zero or one without re-compiling, you should use a "very small value" for a zero. For example, changing the value of `num` to `[0.000001 3.1]` will result in practically the same filter but will not cause a re-compile message to arise. The same principle applies for a value of 1.

5.3 Changing a variable in the MATLAB Command Window and reflecting the change into the running SIMULINK controller

Suppose you are using a MATLAB variable named `A` in a SIMULINK block. A value must be assigned to the variable before you can generate realtime code. You assign the value of `A` from the MATLAB Command Window. Once you generate the Real-time code and you are running the controller, you can change the value of `A` in the MATLAB Command Window. The value does not change in

SIMULINK/WinCon until you either

- a) while the controller is running, open the dialogue box for the block in which the variable A is referenced and close it.
- b) Stop and Start the controller. The new variable value will be automatically updated in the SIMULINK/WinCon controller without opening the dialogue box. The reason the value changed in MATLAB is not automatically updated in SIMULINK is a SIMULINK specific characteristic.

5.4 Summing junction

Summing junction signs cannot be changed once the code has been generated. This is SIMULINK specific characteristic. Do not open a Summing junction dialogue box while running code. This stops the controller and switches Simulation Options to Normal. Always switch it back to External.

5.5 Controller does not start

If you click Start and the controller does not start, check the External switch under Simulation-Options, also make sure you have configured the Realtime Code Options correctly as well as the External File. See section 3.4.

5.6 Integration methods

The choice of integration method for continuous time blocks is made from the Code - Real-time Options dialogue window. The choices under Simulation-parameters will not change integration methods of Real-time Code. Once you compile with a specific integration method, you can only change it by re-compiling. We suggest RK3 since it performs better than Euler but generates less code than RK5. If you choose "None", then the outputs of continuous time blocks will be zero.

5.7 WinCon Configuration

WinCon should always be configured with zero Inputs/Outputs/References (Configure from WinCon Main Window) while running SIMULINK controllers. This option is only for custom written controllers from Quanser Consulting.

5.8 STOP Control/Simulation block

The STOP block from SIMULINK/Sinks does not work under external mode. If you want to stop the controller use STOP WinCon. Obtain STOP WinCon by typing `winlib` in the MATLAB command Window and selecting SOURCES.

5.9 Random Noise blocks

These blocks are very useful for parameter estimation experiments. The block from SIMULINK/Sources does not work in external mode. Use the two choices from the SOURCES under `winlib`.

5.10 Other compilers

To date we support Watcom V10.0a or higher. For other compilers please contact Quanser Consulting.

We will be happy to assist you in creating the correct templates and makefiles for your compiler.

5.11 CRASH!

If the system returns with a Windows error message. Close everything down and reboot. It is a fact of life that crashes sometimes occur (the less often the better of course.. but they do occur).

Who has not seen the cryptic message:

"An error occurred in your application if you choose to ignore ..."

It is a worthwhile habit to save all your work before re-compiling! WinCon has been rigorously tested and we hope that we have ironed out the bugs. If there is a consistent crash occurring please contact us with details. We will attempt to resolve the problem expeditiously. If the controller itself crashes, it is likely that you have selected a sampling period that is too small for the controller/computer combination you have. Start with a slow sampling period then move faster if necessary.

☞ During realtime operation, please do not be hasty in changing windows or clicking the mouse buttons. If the hourglass symbol appears (system busy), wait until it is removed from the screen before you start another operation.

5.12 Maximum sampling frequency

Realtime controllers run in the background. Apart from obtaining realtime control, you want your computer to perform other tasks while the controller runs. How fast can the controller run such that you can still use another application in the foreground? There is no precise answer to this question. It will depend on the complexity of the controller and the speed of the processor you are using. Furthermore if your controller is too complex, you may not achieve the sampling speed you specified. If the operations to be performed (computation delay) take longer than one sample period, the next sample period will be ignored! Note that computational delay from one sample to the next cannot be assumed to be constant. It depends on the actual values being operated upon and if you have a nonlinear controller it depends on the statements being executed. So how can you verify that the controller is sampling at the rate you specify?

Here are a few methods:

a) The simplest method of determining if you are sampling too fast is to observe behaviour of the foreground task. For example, If the mouse cursor is not reacting as fast as it usually does, graphics is not updating quickly or the realtime display is skipping seconds, then the controller has taken over the processor and there is little time left in the foreground to perform other operations. This is usually a good indication that you should reduce the sampling rate.

b) A simple and relatively precise method would be to compare the realtime display in WinCon with a hand held digital chronometer. Start the controller and the chronometer at the same time. Compare the WinCon displayed time with the chronometer. If after ten minutes of operation, the controller and the chronometer are in synch (within 1 second due to delays in starting the controller and the discrepancy between your two hands), then the controller is running at the sample speed without skipping beats. If there is a discrepancy, then you are sampling too fast. The delay between the two times can be used to approximate how many beats were skipped and you can then estimate the average sampling period. As a rule of thumb, you should let 25% of processor time be available for your foreground tasks. For example, if

after 10 minutes of running a 1 KHz controller the WinCon clock was 30 seconds behind the chronometer then the controller skipped approximately 30,000 samples. That means you acquired only 570,000 samples in 10 minutes and the average sampling period was 1.053 mseconds. The average sampling frequency was 949Hz. You should limit your sampling frequency to $0.75 \times 949 = 711$ Hz ($T_s = .0014$ sec.) in order to allow enough time in the foreground to perform other tasks. Running the controller at this speed would also guarantee that no beats are missed.

c) If you want an accurate measure, you can dedicate one of your D/A channels to switch its output between +/-1 volts at every sample. You can then examine the output of the D/A on the oscilloscope and see if you are attaining the speed you want. If you are sampling faster than the time it takes to execute the controller, the output will not be a square wave of the frequency you expect (half the sampling frequency since you switch signs at every sample).

If your system requires faster sampling rates than can be achieved on the PC, you may want to consider

- 1) a faster computer
- 2) Running in DOS
- 3) a dedicated controller (ie a DSP).

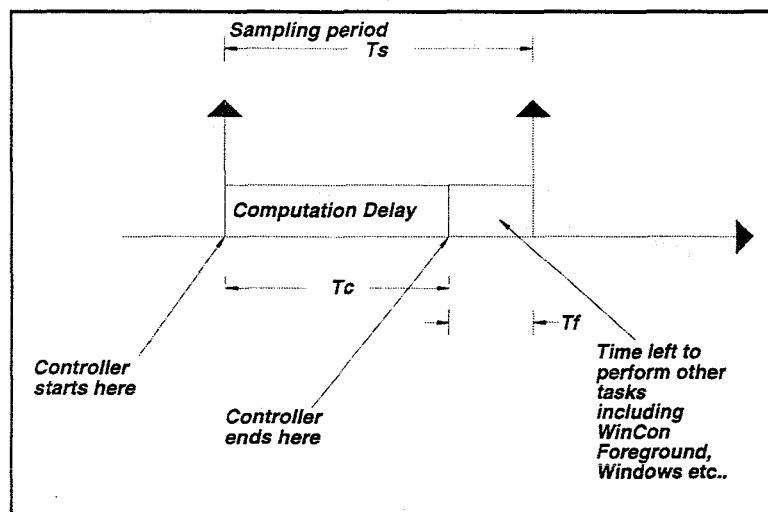


Figure 5.2 Timing Diagram during realtime control [WinCon/start only. Does not apply when WinCon is stopped]

- 1 x limited integrator
- 1 x absolute value
- 1 x relational operator
- 1 x D/A output
- 1 x Stop WinCon

The natural question to ask is if one can predict the maximum sample rate for a given controller. This depends on the code that has been generated and the data acquisition board.

Our benchmark is that the following operations (inverted pendulum, example 4.3) can be performed at 3 kHz leaving little time (around 5%) for other tasks on a Pentium 90 MHz.

- 2 x A/D acquisitions
- 4 x 1st order continuous time filters
- 1 x signal generator
- 6 x scalar gain
- 7 x additions
- 3 x scopes
- 1 x realtime clock

One can then estimate that if you double the above complexity, then the fastest rate possible is around 1500 Hz if you use the same computer. Note that the data acquisition blocks typically take longer to execute than the other blocks. So if you maintain the same I/O channels and add some other blocks to the above, you may still be able to achieve around 3 KHz... it all depends on complexity.

Predicting the maximum sample rate accurately is usually not possible. The safest method to obtain this measure for a given controller is to try the controller without connecting the plant. You then keep increasing sampling frequency until there is considerable slowing of the tasks in the foreground. If the computer hangs while performing this test, you should reboot the system. The reason the computer would hang is that the sample rate was so high that there was no time left to interact with the user!

5.13 FUZZY Controllers

The file `c:\matlab\toolbox\fuzzy\ffis.c` does not work with the realtime workshop. You will need to download an update for this file from the Mathworks website at: <http://www.mathworks.com>.

Fuzzy controllers will require much more computational power than a state feedback controller. Sampling periods must be selected carefully. Our experience indicates that a 2 input, 1 output FUZZY controller with 5 membership functions at each input, 9 rules and 5 membership functions at the output can only run at 200 Hz. Sample rates faster than that will not function properly.

5.14 Disk Space Make sure there are at least 10MBytes of free hard disk space on your system.

5.15 Networked systems WinCon is a realtime system and network operations will interfere with the performance. WinCon is designed for a standalone system and Quanser Consulting does not guarantee performance on networked systems.

5.16 Versions!

Much too often, all software changes versions (that is a drag). Type `ver` in the Matlab command window. You will get a display as shown below. These are the versions we have been using. If you are using earlier versions there is no guarantee things will work. Later versions should *normally* cause no problems.

```

ver
MATLAB Version 4.2c.1
MATLAB Site Identification Number: *****
No Contents.m file for c:\winsim
No Contents.m file for c:\matlab\codegen\rt\wincon\devices
No Contents.m file for c:\matlab\codegen\rt\dos\devices
No Contents.m file for c:\matlab\codegen\rt\dos\quanser
MATLAB Toolbox Version 4.2a 25-Jul-94
SIMULINK model analysis and construction functions. Version 1.3c 15-August-94
SIMULINK demonstrations and samples. Version 1.3c 15-August-94
SIMULINK block library. Version 1.3c 15-August-94
SystemBuild 3.0 model import into SIMULINK. Version 1.0a 09-Mar-94
Real-Time Workshop Version 1.1c 25-May-95
Control System Toolbox. Version 3.0b 3-Mar-93
Nonlinear Control Design Toolbox. Version 1.0 5-Nov-93
Signal Processing Toolbox. Version 3.0b 10-Jan-94
User Interface Utilities. Version 1.3 19-Oct-95
System Identification Toolbox. Version 4.0 30-May-95
Fuzzy Logic Toolbox. Version 1.0, 1-19-95
Fuzzy Logic Toolbox Demos. Version 1.0, 1-19-95
    
```


B.2 MultiQ board manual

Instruction manual, 21 pages

MULTIQ™

8 Analog to Digital Converters

8 Digital to Analog Converters

6 Quadrature input decoders/counters

8 Digital inputs

8 Digital outputs

3 Realtime clocks

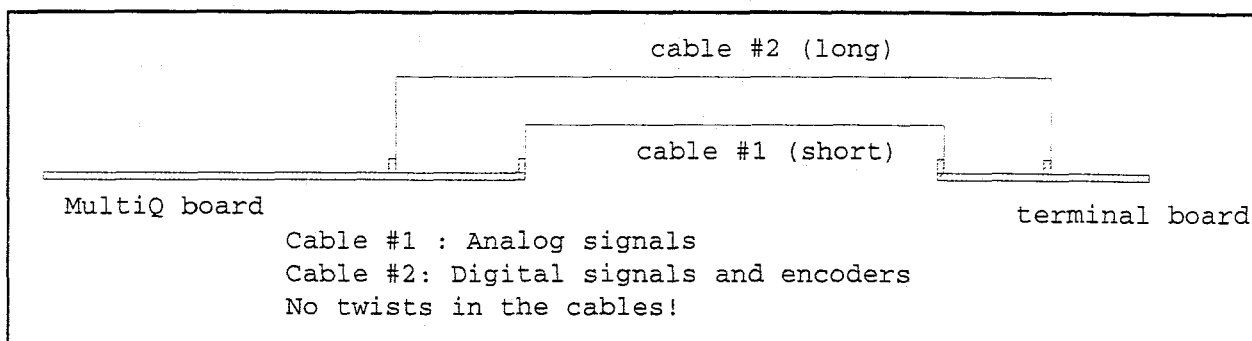
PROGRAMMING MANUAL

Quanser Consulting

CAUTION

The flat ribbon cable connectors are keyed and must be inserted into the **MULTIQ** board and the **TERMINAL** board with the correct *orientation*. Furthermore, the short cable should be used for the Analog signals and the long cable should be used for the digital signals and encoders.

CAUTION



The board is static electricity sensitive. Be very careful with electrical discharge. Always touch a ground plane **BEFORE** you handle the board.

CAUTION

Some of the wiring you perform to the terminal board carries **POWER**. Be sure your wiring is correct as applying power incorrectly may either damage the device you are connecting to, the board or your computer!

NEED HELP ?

CALL US AT (905) 527 5208 or FAX your question to (905) 570 1906

OR EMAIL TO : QUANSER @NETACCESS.ON.CA

MULTIQ™ I/O BOARD

Quanser Consulting

1.0 General description

The MULTIQ is a general purpose data acquisition and control board which has 8 single ended analog inputs, 8 analog outputs, 8 bits of digital input, 8 bits of digital output, 3 programmable timers and up to 6 encoder inputs decoded in quadrature. Interrupts can be generated by either of the three clocks, one digital input line and the end of conversion from the A/D.

The system is accessed through the PC bus and is addressable via 16 consecutivememory mapped locations which are selected through a DIP switch located on the board.

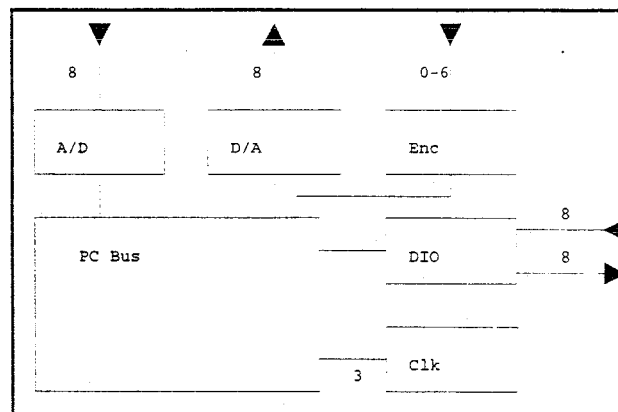


Figure 1 Block diagram of MULTIQ

2.0 Principles of operation

2.0.1 Terminal board

Turn the PC off and insert the MultiQ into a bus slot in your PC. Connect the cables as shown on the front page of this manual. The terminal board supplied with the board is shown in Figure 2.

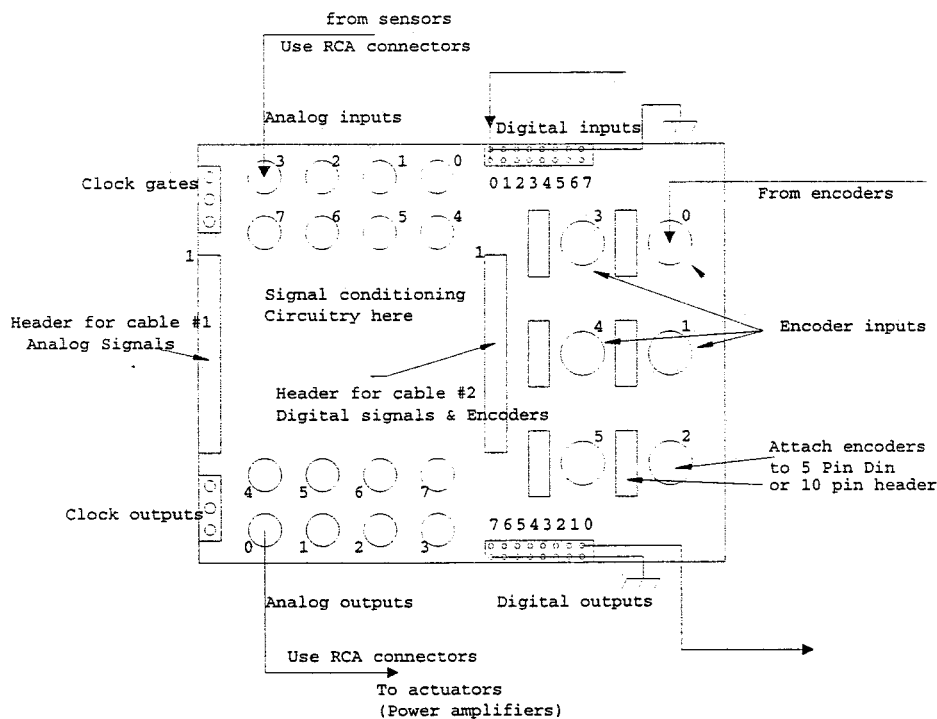


Figure 2 Terminal board

All wiring to the board is performed through the terminal board. Analog inputs and outputs are connected via RCA connectors. Digital I/O is via 16 pin headers and Encoder inputs are through 5 Pin Din (Stereo) connectors or via 10 pin headers.

2.1 Analog to digital conversion

The A/D of the MULTIQ is a single ended bipolar signed 13 bit binary (12 bit plus sign) A/D. You can perform a conversion on one of 8 channels by selecting the channel and starting a conversion. the EOC_I (end of conversion interrupt) bit in the STATUS REGISTER indicates that the data is ready and can be read. The data is read by issuing 2 consecutive 8 bit reads from the AD_DATA register.

The data returned is two 8 bit words which must be combined to result in a 16 bit signed word. 5 volts input maps to 0xFFFF while 0 volts maps to 0x0 and -5 Volts maps to 0xFFFF000.

2.1.1 Wiring to the A/D

↗ ≤ 5.3

All inputs to the A/D multiplexer are single ended in the range +/- 5 Volts and should be wired via the RCA connectors on the terminal board labelled Analog Inputs.

2.1.2 Signal conditioning of the Analog Input signals

If you wish to low pass filter the input signal before data acquisition you may do so by populating the area labelled Signal Conditioning shown in Figure 2. The circuit for each input channel is shown in Figure 3. **It is highly recommended that you attach a capacitor at least to the input of each A/D. This is simply done by soldering a capacitor into the appropriate holes.** The factory configuration is Ra = short, Rb = open and C = open. The choice of the component values depends on the output impedance of the sensor and the sampling frequency of your data acquisition program. Typically you select a cutoff frequency to be less than half the sampling frequency (or even smaller).

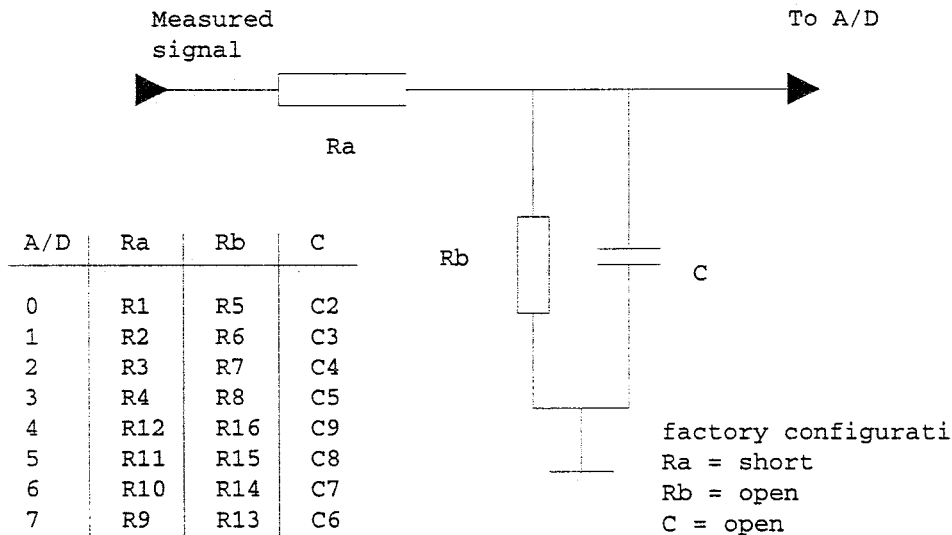


Figure 3 Signal conditioning circuit for Analog input signals

2.2 Analog output

The digital to analog (D/A) converters are 12 bit unsigned binary. An input of -5 volts maps to 0x000, 0 volts to 0x3FF and 5 volts maps to 0xFFF. Your program should write a 12 bit number (0 to 4095) to the appropriate register and should latch the data. The analog outputs change when the data is latched.

2.3 Encoder Inputs

The board can be equipped with up to six encoder decoders. (Models -2E, 4E and 6E). The encoders data is decoded in quadrature and used to increment or decrement a 24 bit counter. With 24 bits, you can obtain 16,777,215 counts. With a 2000 line encoder in quadrature, this results in 8000 counts per revolution and 2097 revolutions can be measured without overflowing the counters. Higher counts can be handled by software.

2.3.1 Wiring to the Encoder inputs.

Each encoder input is equipped with one 5 pin DIN socket and one 10 pin header on the terminal board. you can use either of these to connect an encoder to the board. The connectors supply +5V and GND to bias the encoders and receives an 'A' channel and a 'B' channel from the encoder. **You must use 5 Volt output encoders only.** Figure 4 shows the pin definition for the 5 pin DIN connector as well as the 10 pin header of the encoders inputs and the method of wiring.

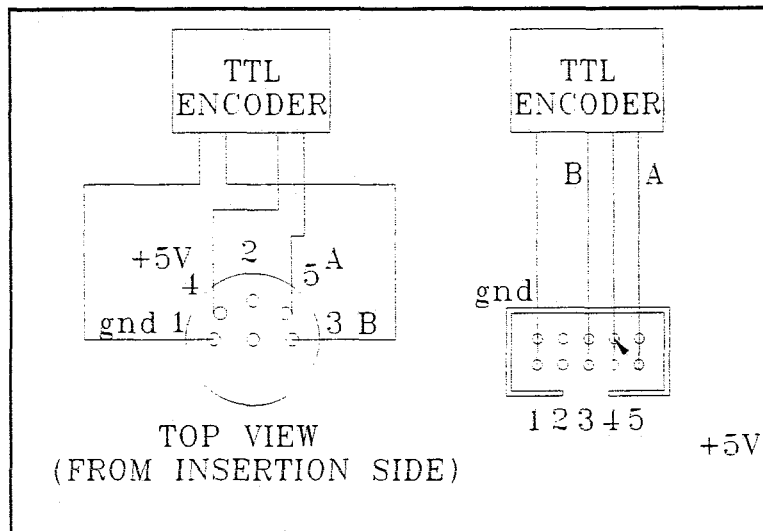


Figure 4 Encoder connections to the 5 pin DIN connector or the 10 pin header.

2.6 Realtime clocks

The board is equipped with three independent programmable clock timers. Each timer can be programmed to run at a frequency between 2 MHz Hz and 30.52 Hz. The principle of operation is to write a divisor (N) to the desired clock and the output frequency will be $2.0/N$ MHz. (N) is a 16 bit integer value between 2 and 65535 (0xFFFF). The output of any of the three clock can be tied to an interrupt line using a jumper on the board. The outputs are available on the terminal board for monitoring or triggering external devices.

2.4 Digital inputs

The board can read 8 digital input lines mapped to one I/O address. The digital input is normally high ('1') and results in a low ('0') when the line is pulled to GND. Digital input line #0 can be tied to an interrupt using the jumpers supplied.

2.5 Digital outputs

The board can control 8 individual digital outputs mapped to one I/O address. Writing a '0' to the appropriate bit results in zero volts (TTL LOW) at the output while writing a '1' results in 5 Volts (TTL HIGH).

3.0 PROGRAMMING

3.1 Base address selection

The base address is selected by using the dip switches(SW2) on the board. Factory configuration is 0x320. make sure there are no other devices on that address and up to Base+0xF (0x32F factory configuration).

Base Address											
B	A	9	8	7	6	5	4	3	2	1	0
0	0	SW2-F	SW2-E	SW2-D	SW2-C	SW2-B	SW2-A	Decoded on board			

3.1.1 Factory configuration

Base Address								
0	A	9	8	7	6	5	4	3
0	0	SW2-F	SW2-E	SW2-D	SW2-C	SW2-B	SW2-A	Decoded on board
0	0	1(OFF)	1(OFF)	0(ON)	0(ON)	1(OFF)	0(ON)	0
3			2				0	

3.2 Board Registers The table below shows the registers on the MULTIQ. Each register is described in its appropriate section.

Base +	Read	Write	Size
0	DIGIN_PORT	DIGOUT_PORT	8 Bit
1			
2		DAC_DATA	16bit
3			
4	AD_DATA	AD_CS	16 bit write 8 bit read
5			
6	STATUS	CONTROL	16 bit
7			
8	N/A	CLK_DATA	8 bit
9	N/A	N/A	N/A
A	N/A	N/A	N/A
B	N/A	N/A	N/A
C	ENC_1	ENC_1	16 bit
D			
E	ENC_2	ENC_2	16 bit
F			

3.2.1 Digital port : Base + 0 (DIGIN_PORT & DIGOUT_PORT)

3.2.1.1 Write (DIGOUT_PORT)

This is the Digital output port. A 16 bit write to this port outputs the 16 bit data to the digital output header on the terminal board. Eg. writing a 0x00e5 (binary 11100101) results in bits 0,2,5,6,7 to go high.

DIGIN_PORT Write															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	DO 7	DO 6	DO 5	DO 4	DO 3	DO 2	DO 1	DO 0

3.2.1.2 Read (DIGIN_PORT)

This is the Digital input port. A 16 bit read from this board returns the digital levels at the header labelled Digital input on the terminal board. The inputs are tied high and a read with nothing connected to the header results in reading a 0xFFFF. A returned value of 0x FFa6 (last 8 bits 10100110) means that bits 0,3,4 and 6 have been pulled low by an external device (for example a switch). Note that the high word is always ones.

DIGIN_PORT Read															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	DI 7	DI 6	DI 5	DI 4	DI 3	DI 2	DI 1	DI 0

3.2.2 D/A data Port : Base + 2 (DAC_DATA)

3.2.2.1 Write (DAC_DATA)

A write to this port sets up the Data for the D/A output. The data should be written to bits(DO11 to DO0). A value of 0 puts out -5 Volts, a value of 0x1FF puts out 0 Volts and 0xFFF puts out +5 Volts. The output channel is selected by writing to bits (DA2 DA1 DA0) of the CONTROL REGISTER and the output is latched when a (11) is written to bits (LD0 LD1) of the CONTROL REGISTER.

DAC_DATA Write															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NA	NA	NA	NA	AO 11	AO 10	AO 9	AO 8	AO 7	AO 6	AO 5	AO 4	AO 3	AO 2	AO 1	AO 0

3.2.2.2 Read (NOT APPLICABLE)

3.2.3 A/D Register : Base + 4 (AD_CS and AD_DATA)

3.2.3.1 Write (AD_CS)

A write to this register (any data) initiates a conversion after the A/D has been properly set up using the CONTROL REGISTER. The program must wait for (EOC) to go high in the STATUS REGISTER *before* initiating a conversion.

3.2.3.1 Read (AD_DATA)

This is an **8 bit read register** and contains the high byte on the first read and the low byte on the second read. The structure of the two 8 bit reads is shown below:

AD_DATA Read (First time)							
7	6	5	4	3	2	1	0
SIGN	SIGN	SIGN	SIGN	AI 11	AI 10	AI 9	AI 8

AD_DATA Read (Second time)							
7	6	5	4	3	2	1	0
AI 15	AI 14	AI 13	AI 12	AI 11	AI 10	AI 9	AI 8

To convert the data to a voltage the two bytes should be combined into a 16 bit word as follows:

```
integer_data = (high_byte<<8)|(low_byte&0xFF);
volts = integer_data * 5.0/4096;
```

where (<<) is the shift left operator, (|) is bitwise 'OR' and (&) is bitwise 'AND'. This means mask off the left 8 bits of the low byte just in case there is extraneous noise, and then merge it with the high byte shifted left 8 bits. This results in a number between (-4096) and (4095). These are mapped to -5V and +5V.

3.2.4 CONTROL REGISTER: Base + 6 (STATUS & CONTROL)

3.2.4.1 Write (CONTROL)

CONTROL REGISTER Write															
X	X	X	LD1	LD0	CLK	S/H	CAL	AZ	EN	A2	A1	A0	DA2	DA1 RC1	DA0 RC0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Bits 0-2 (DA0 DA1 DA2) : Select the D/A channel number. eg writing a 0x03 selects channel D/A ch3

Bits 0-1 (RC0 RC1) : Select the realtime clock register

Bits 3-5 (A0 A1 A2) : Select the analog input channel you want to Multiplex to the A/D

Bit 6: (MX) Enable the 8 channel multiplexer

Bit 7: (AZ) Enable Auto Zero on the A/D

Bit 8: (CAL) Enable Autocalibration on the A/D

Bit 9: (S/H) Disable Sample and Hold on the A/D **Keep this bit high all the time**

Bit 10 (CLK): Select base clock frequency for the A/D. 1 is 4 MHz, 0 is 2 MHz. (Always use 4 Mhz). **Keep this bit high all the time**

Bit 11-12(LD0,LD1): Latch data to the selected D/A channel when both bits are set high.

3.2.4.2 Read (STATUS)

STATUS REGISTER Read																
												EOC_I	EOC	CT1	CT2	CT0
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	

Bits 0-2(CT0,CT1 CT2): Counter timeout states for the three timers.

Bit 3:(EOC) End of conversions on the A/D. Goes high when A/D is ready for another conversion.

Bit 4:(EOC_I) End of conversion Interrupt. Goes high when A/D conversion is complete.

3.2.5 Clock data register: base + 8 CLK_DATA

3.2.5.1 Write (CLK_DATA)

This an **8 bit** write only register that accesses any of the four registers on the realtime clock chip. The register (CLK_0 to CLK_4, see section on programming) is selected by writing to bits (RC1 RC0) of the CONTROL REGISTER and then writing the data to CLK_DATA.

CLK_DATA Write							
7	6	5	4	3	2	1	0
CD	CD	CD	CD	CD	CD	CD	CD
7	6	5	4	3	2	1	0

3.2.5.2 Read (NOT APPLICABLE)

3.2.6 ENCODER REGISTERS Base + 0xC and Base + 0xD (ENC1 & ENC_2)

3.2.6.1 Write (ENC_1)

Writing to this register selects which of the 6 encoder counter values you want to read next. The encoder number is specified in bits (E2 E1 E0)

ENC_1 Write															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	R1	R0	E2	E1	E0

Writing to bits (R1 R0) will reset the encoders to a zero count. To reset an even numbered encoder (0,2,4) write the encoder number into bits (E2 E1 E0) and write a '1' to bit R0. To reset an odd numbered encoder (1,3,5) write the encoder number to bits (E2 E1 E0) and write a '1' to bit R1.

Writing to ENC_2 is not applicable

3.2.6.1 Read (ENC_1 and ENC_2)

A 16 bit read from ENC_1 returns the low 16 bits of the 24 bit counter selected by the write described above. A 16 bit read from ENC_2 returns the high eight bits of the 24 bit counter located in the low eight bits of the data.

ENC_1 Read															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN	EN	EN	EN	EN	EN	EN	EN	EN	EN	EN	EN	EN	EN	EN	EN
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

ENC_2 Read															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NA	NA	NA	NA	NA	NA	NA	NA	EN	EN	EN	EN	EN	EN	EN	EN
								23	22	21	20	19	18	17	16

4.0 SAMPLE PROGRAMS

The following are sample programs written in Turbo C and can be used by your main program. These functions are included in the file **multiq.driv**. The file also contains the definitions of the register locations based on a base address 0x320 as shown below:

```
#define base_port 0x320

#define digin_port    base_port + 0x00
#define digout_port   base_port + 0x00
#define dac_cs        base_port + 0x02
#define ad_cs         base_port + 0x04
#define status_reg    base_port + 0x06
#define control_reg   base_port + 0x06
#define clk_reg       base_port + 0x08
#define enc_reg1      base_port + 0x0c
#define enc_reg2      base_port + 0x0e

#define AD_SH         0x200 /* active low */
#define AD_AUTOCAL    0x100 /* active high */
#define AD_AUTOZ      0x80  /* active high */
#define AD_MUX_EN     0x40  /* active high */
#define AD_CLOCK_4M  0x400 /* high = 4 MHz */

/* IMPORTANT */
/* sample and hold disabled to prevent exrtaneous sampling */
/* and fix the clock speed to 4 MHz */

#define CONTROL_MUST (AD_SH | AD_CLOCK_4M)

unsigned int control_word = CONTROL_MUST;
```

4.1 D/A operation

- Write the analog output channel number to bits (DA2 DA1 DA0) of CONTROL along with a (11) to LD1 and LD0 of the CONTROL REGISTER
- Write the actual data to DAC_DATA (16 bit write lowest 12 bits carry the data).
- release the latch by writing a (00) to both bits (LD1 LD0) of the CONTROL REGISTER.

4.1.1 Sample C function

```
void daout(int ch, int ivalue)
{
    output(control_reg, 0x1800 | ch | CONTROL_MUST);
    output(dac_cs, ivalue);
    output(control_reg, CONTROL_MUST);
}
```

4.1.2 Reset the D/A outputs

```
void reset_da(void)
{
    int zero_v;
```

```

zero_v = vtoi(0.0); /* see this function below */
daout(0,zero_v);
daout(1,zero_v);
daout(2,zero_v);
daout(3,zero_v);
daout(4,zero_v);
daout(5,zero_v);
daout(6,zero_v);
daout(7,zero_v);
}

```

4.1.3 Voltage to integer conversion

This function is used to convert a desired voltage (in volts) to the appropriate integer value for the D/A.

```

int vtoi(float v)
{
return(ceil( v*2048/5.+2047));
}

```

4.2 A/D OPERATIONS

4.2.1 Calibrating the A/D

This can be performed **once only** at the start of a program. Once calibrated, the offset and gain are used for all subsequent measurements. To calibrate:

- 1) write a '1' to bit 'CAL' and to 'S/H' of CONTROL REGISTER
- 2) write a '1' to 'S/H' of CONTROL REGISTER
- 3) wait for EOC to go high in STATUS REGISTER

4.2.1.1 Sample C function

```

void reset_ad(void)
{
/* start calibration */
outport(control_reg, AD_AUTOCAL | CONTROL_MUST);
outport(control_reg, CONTROL_MUST);
while( (inport(status_reg)&0x08) == 0x00 );
}

```

4.2.2 Acquiring a sample

- 1) select the channel and write it to control register bits (A2 A1 A0) along with a '1' to (EN) , a '1' to (S/H) and a '1' to (CLK) bits of the CONTROL REGISTER. Also write a '1' to bit (AZ) if you want auto zero before the sample. Note that autozero takes longer and is not normally necessary.
- 2) wait until (EOC) in STATUS REGISTER goes high.
- 3) Initiate a conversion by a write to AD_CS (any value)
- 4) wait until EOC_1 in STATUS REGISTER goes high
- 5) read high byte from AD_DATA
- 6) read low_byte from AD_DATA

4.2.2.1 Sample C Function

```
int adin(int ch)
{
  unsigned int hb,lb;
  int toolong,maxcnt;
  maxcnt = 30;
  nosound();
  control_word = CONTROL_MUST | AD_MUX_EN | (ch<<3); /* select channel and enable mux start S/H*/

  /* use the next line instead of above line if you want to auto zero before every sample */
  /*control_word = CONTROL_MUST | AD_AUTOZ | AD_MUX_EN | (ch<<3);*/
  /* NOTE THAT IT IS SLOWER WITH AUTO ZERO */

  outport(control_reg,control_word);
  toolong = 0;
  while( ((inport(status_reg)&0x8) == 0x00 ) && (toolong <maxcnt) ) toolong++;
  if(toolong>=maxcnt) sound(400);
  outportb(ad_cs,0);
  while( (inport(status_reg)&0x10) == 0x00 );
  hb = inport(ad_cs) & 0xff;
  lb = inport(ad_cs) & 0xff;
  outport(control_reg,CONTROL_MUST);
  return ( (hb<<8) | lb);
}
```

Note the limited wait loop:

```
toolong = 0;
while( ((inport(status_reg)&0x8) == 0x00 ) && (toolong <maxcnt) ) toolong++;
if(toolong>=maxcnt) sound(400);
```

- which ensures that the wait for EOC is left if it takes too long . If this happens, an error has occurred on the A/D Chip (National Semiconductor ADC1251) during a conversion and the chip is not ready for a conversion after sufficient waiting. *This is not usually necessary but is good practice.* If this error occurs, the computer will generate a sound until issuing 'nosound()' from the calling C program. If you do not want the sound to go on just delete the 'sound(400)' statement.

4.2.3 Integer to voltage conversion

The following function converts from an integer value read by the A/D to a floating point value in volts.

```
float itov(int iv)
{
  return(iv*5/4095.);
}
```

4.3 Digital input operation:

- Read a 16 bit word from DIGIN_PORT

4.3.1 Sample C Function

```
int digin(void)
```

```

{
return inport(digin_port);
}

```

4.4 Digital output operation

- Write a 16 bit word to DIGOUT_PORT

4.4.1 Sample C Function

```

void digout(int dig_value)
{
outport(digout_port,dig_value);
}

```

4.5 Encoder operations

4.5.1 Encoder reset

- write to ENC1 the encoder channel you want to reset.
- write to bit R0 of ENC1 a '1' if the channel number is even (along with the channel number)
- write to bit R1 of ENC1 a '1' if the channel number is odd (along with the channel number)

4.5.1.1 Sample C function

```

void enc_reset(int ch)
{
outport(enc_reg1,ch);
if( (ch == 0) ||(ch == 2)|| (ch == 4)) outport(enc_reg1,((ch&0x07)|0x8));
if( (ch == 1) ||(ch == 3)|| (ch == 5)) outport(enc_reg1,((ch&0x07)|0x10));
}

```

4.5.2 Encoder read

- write to ENC_1 the channel number you want to read into the lower 3 bits(E2 E1 E0)
- read from ENC_1 the low byte (16 bits)
- read from ENC_2 the high byte (16 bit read, mask off the 8 most significant)
- merge the two data to obtain a 'long signed int'

4.5.2.1 Sample C function

```

long int enc_in(int ch)
{
unsigned int low_word, high_word;
unsigned long result;

outport(enc_reg1,ch);
low_word = inport(enc_reg1);
high_word =inport(enc_reg2);
result = high_word & 0xff ;
if(result & 0x80) result = result | 0xff00; /*convert to signed 32 bit*/
result = (result << 16) | low_word;
return ((long) result);
}

```

4.6 Clock operations

The three clocks are imbedded in a single integrated circuit (INTEL 82C54). Four registers located on the clock chip are addressed via bits (RC1 RC0) of the MULTIQ's CONTROL REGISTER. In order to write a desired value to a specific clock register, first write to (RC1 RC0) of the CONTROL REGISTER the value for the register you want to access and then write the desired value to the CLOCK DATA REGISTER (Base + 8).

RC1	RC0	Clock IC register
0	0	CLOCK 0 DATA REGISTER
0	1	CLOCK 1 DATA REGISTER
1	0	CLOCK 2 DATA REGISTER
1	1	CLOCK COMMAND REGISTER

The CLOCK COMMAND REGISTER has the following bits:

CLOCK COMMAND REGISTER							
7	6	5	4	3	2	1	0
SC1	SC0	RW1	RW2	M2	M1	M0	BC

The clock you want to perform an operation on is selected using bits (SC1 SC0) and the mode of operation is selected using bits (M2 M1 M0).

In order to program a specific clock to run at a given frequency, you must first select a divider for the clock. For example if you want to run at a frequency 'F' the divisor is obtained by using the equation:

$$DIV = CEIL(2e6/Freq)$$

where 2e6 Hz is the base frequency for all three clocks. This will be a 16 bit integer value (0 to 65535). Note that if Freq is smaller than (2e6/65535) the clock will actually run much faster than you expect!

Next you need to write to the CLOCK COMMAND REGISTER the clock number into (SC1 SC0) and select mode2 into (M2 M1 M0)(Baud rate Generator). You do this by **first writing to the CONTROL REGISTER** bits (RC1 RC0) a (1 1) indicating you will be writing to the CLOCK COMMAND REGISTER next and then you write the desired data .

After you select the clock number and the mode of operation into the CLOCK CONTROL REGISTER, write to the CLOCK DATA REGISTER (bits RC1 RC0 in the CONTROL REGISTER) the low byte and then the high byte of the divisor DIV. At this point, the clock you selected will start running at the frequency you specified.

4.6.1 Sample C functions

```
void clockdiv(int clk_num,int div_value)
{
    unsigned int lb,hb;
    lb = div_value & 0xff;
    hb = (div_value & 0xff00)>>8;
}
```

```

control_word = 3 | CONTROL_MUST;
output(control_reg,control_word);/*select register 3 of RTC */
outputb(clk_reg,((clk_num<<6)|0x34));
control_word = clk_num | CONTROL_MUST;
output(control_reg,control_word);
outputb(clk_reg,lb);
outputb(clk_reg,hb);
}

```

/ this function sets the clock frequency of the desired clock. It calls clockdiv() */*

```

void set_clk_freq(int clk_num,float clk_freq)
{
float base_freq = 2000000;
int divider;
divider = ceil(base_freq/clk_freq);
clockdiv(clk_num,divider);
}

```

The states of the three clock can be monitored through bits (CT2 CT1 and CT0) of the STATUS REGISTER.

5.0 Tying to interrupts

You may wire the following lines to some of the interrupt lines on the PC bus.

Bit	Status
CT0	Clock Timer 0 overflow
CT1	Clock Timer 1 overflow
CT2	Clock timer 3 overflow
EOC_I	End of conversion interrupt
DIO	Digital input bit 0

The interrupts lines that can be tied to are

Interrupt#	Normally used by (on standard PC)	Vector address
3	COM1: Serial port	0xB
5	PC Fixed disk controller	0xD
7	LPT1: Printer	0xF
9	RESERVED	RESERVED
10	UNUSED	0x72
11	UNUSED	0x73
12	UNUSED	0x74
15	UNUSED	0x77

The interrupt lines should be physically connected on the board using the jumpers provided. You should be certain that no other device is tied to the interrupt line you want to use.

5.1 Writing interrupt service routines

The following is a short explanation on how to write interrupt service routines on an IBM PC compatible system. More detailed information can be obtained from any good book on PC architecture and the C compiler you are using.

An interrupt service routine (ISR) is initiated every time a specified interrupt line associated with the ISR goes high. The interrupt mask register on the PC is located at address 0x21. It is an eight bit port and in order to activate a certain interrupt line, you must write a '0' to the associated bit in the interrupt mask register. For example if you want to allow interrupts only from lines 2 and 7 you must write a (01111011) or (0x7b) to memory location 0x21.

Each interrupt lines causes a jump to the address located in the vector table shown above. For example if you want a function:

```
extern void interrupt far newtimer(void);
```

to be executed every time interrupt #5 occurs, you set it up in the following manner:

```
disable(); /* disable interrupts */
oldtimer = getvect(0xd); /*save old isr address, see vecctor in column 3 of table above */
setvect(0xd,newtimer); /* setup the new isr */
int_mask = inportb(0x21); /* get the present mask */
outportb(0x21,int_mask&0xdf); /* write out a new mask that sets bit 5 to 0) */
enable(); /* enable interrupts, at this point newtimer is active and will be executed every time line 3 goes high */
```

Now what happens inside the isr is also very important.
The ISR should have the following structure:

```
extern void interrupt far newtimer(void)
{
asm fsave data87 /* assembly code to save floating point processor status */
_clear87(); /* clear the floating point processor */
disable(); /* disable other interrupts */

/* here is where you write your code */
/* this will be executed every time the interrupt occurs */

asm frstor data87 /* restore floating point processor status */
outportb(0x20,0x65); /* acknowledges interrupt to 8259 */
enable();
}
```

data87 must be declared globally as

```
char *data87[94];
```

6.0 Testing your board

The programs 'test_mq.c', 'test_int.c' and the driver file 'multiq.drv' are supplied with the board. In order to compile these programs you need Turbo C and Turbo assembler. Executable versions are also supplied.

- make a directory MULTIQ on your hard drive
- copy all files to that directory
- Run the program **test_mq**

- The program is interactive and you can plot selected data in realtime. You can select the parameters associated with the letters in brackets []. For example entering the letter [v] you will be prompted to enter a voltage which will be output to D/A channel specified by hitting [o]. The program is interrupt driven and all variables are monitored on the screen in realtime. The A/D channel associated with the letter [i] can be plotted in realtime as well as the encoder input associated with the letter [e]. Entering [z] will reset the encoder selected. You can output a digital word to the digital output port by entering [d] followed by a hexadecimal number. You can select the data you want to plot in realtime by hitting [V]. Hit 'x' to exit realtime plotting. Selecting [T] allows you to alter the duration of the x axis of the realtime plot.

[i]: A/D input channel number
[o]: D/A output channel number
[v]: output volts
[d]: Digital word out
[e]: Encoder channel
[z]: Encoder reset
[F]: Sampling Frequency(Hz)
[T]: X axis time duration(Sec)
[V]: Y axis variable:
[R]: Realtime plot
[Q]: Quit program

The program **test_mq** uses the IBM PC SOFTWARE INTERRUPT and changes the speed of the realtime clock used for time of day. Run this program to test all the functions on the board except the realtime clocks.

Reset the time of day from DOS if necessary.

The program `test_int` uses the MULTIQ CLOCK #1 for hardware interrupts on interrupt line #5. In order to run this program you should install the jumper labelled CTC1! to pin interrupt 5.(FACTORY CONFIGURATION)

6.1 Compiling the source code

If you need to change anything in the source code of the above programs, you will need to recompile them. Use the following lines to obtain new executable code.

Turbo C users

```
tcc -r -B -f87 -ml %1 graphics.lib
```

Borland C users

```
tcc -r -B -f87 -ml %1 graphics.lib
```

where %1 is the name of the program.

Note Turbo assembler (TASM) should be in the PATH as well as (TC\BIN) or (BC\BIN). The file EGA VGA.BGI must be present in the directory.

6.2 Execution speed

The program `speed.c` tests the speed of the various operations on the board using the drivers listed above. The speeds are obtained on a 66MHz PC486 and may vary with the processor you use. The speeds do not solely depend on the board but also on execution time of the instructions in the drivers. For example, a single A/D conversion is performed in only 8 μ seconds but it really takes 19.2 μ seconds to actually acquire the data into the program. The extra time is due to bus access (inport, outport wait loops).

The results from 'speed.c' are tabulated below. These are calculated by obtaining an average over one million consecutive operations using the driver functions given above.

Function	Execution speed in μ seconds
Digital input (16 bit)	2
Digital output (16 bit)	2
Encoder read (24 bit)	5.5
Analog todigital conversion (13 bit)	19.2
Digital to analog conversion (12 bit)	5.0

The above information is important when determining what is the maximum sampling frequency you can set in an ISR. Suppose you would like to sample all 8 analog input channels, output to all 8 channels, read 6 encoders and perform 16 bit digital I/O in a single interrupt service routine. These would take approximately $(2+2+5.5 \times 6 + 19.2 \times 8 + 5 \times 8) = 230.6$ microseconds. Therefore the ISR should be called at a frequency slower than 3.8 KHz. If you want to perform calculations in an ISR (which you typically would for a controller), then the time for calculations should also be taken into consideration. Assuming the calculations you are performing take another 230 microseconds, then the ISR should execute slower than 1.9 KHz. You would

also like to have some time left over for foreground jobs. Lets's say 50% of processor time left for foreground operations(plotting, user interaction, etc), then the ISR should be set to a maximum of 950 Hz. This is the suggested maximum sampling frequency when the board is being used at full capacity. Of course, the less channels are use and the simpler the controller, the faster you can set the speed of the ISR.

7.0 Selecting interrupt sources

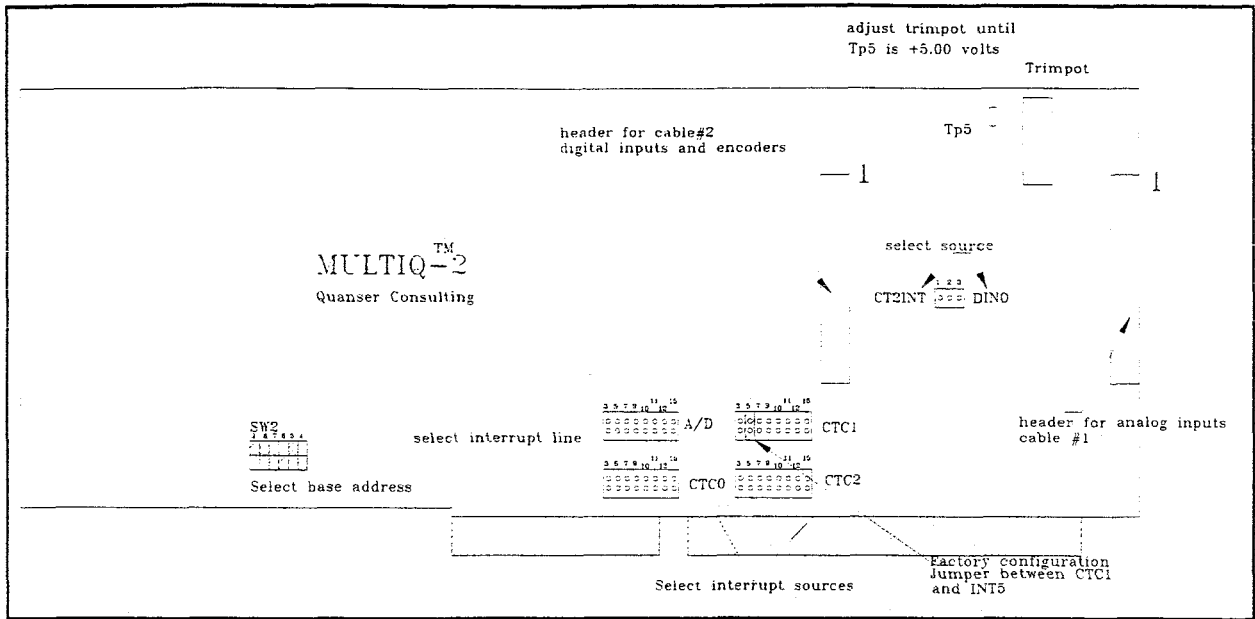
Select the interrupt source using the jumpers at the headers labelled **Interrupts**. USE ONE INTERRUPT JUMPER PER HEADER.

1) Placing a jumper at the header labelled A/D will cause an interrupt from EOC_I to occur at the line to which the jumper is attached. ie if the jumper is attached between A/D and the pin labelled '5', an EOC_I will cause an interrupt number 5 to occur.

2) Placing a jumper at the header labelled CTC0 will cause an interrupt from CLOCK 0 to occur at the line to which the jumper is attached. ie if the jumper is attached between CTC0 and the pin labelled '5', an interrupt number 5 will occur at the frequency at which CLOCK 0 is operating.

3) Placing a jumper at the header labelled CTC1 will cause an interrupt from CLOCK 1 to occur at the line to which the jumper is attached. ie if the jumper is attached between CTC1 and the pin labelled '5', an interrupt number 5 will occur at the frequency at which CLOCK 1 is operating. **THIS IS FACTORY CONFIGURATION.**

4) Placing a jumper at the header labelled CTC2 will cause an interrupt from CLOCK 2 **OR FROM** Digital Input #0 to occur at the line to which the jumper is attached. The source depends on the jumper labelled CTC2INT. If the jumper is between pins (12) then the source is clock 2, if the jumper is between (23) then the source is DIN0. With the CTC2INT jumper located at (12) and one jumper at (CTC2,5) an interrupt number 5 is generated at the speed of CLOCK2. With the CTC2INT jumper located at (23) and one jumper at (CTC2,5) an interrupt number 5 is generated every time the digital input number 0 is pulsed from LOW to HIGH to LOW



INTRODUCTION

Thank you for purchasing our DYNASERV DD Servo-Actuator (New Driver Model). The DYNASERV is a high torque, high velocity, highly accurate outer rotor type servo-actuator which can be used in a wide range of field applications related to FA (Factory Automation), including industrial robots, indexes, etc.

This instruction manual covers the DM/SD series. Be sure to read this instruction manual prior to operating the DYNASERV.

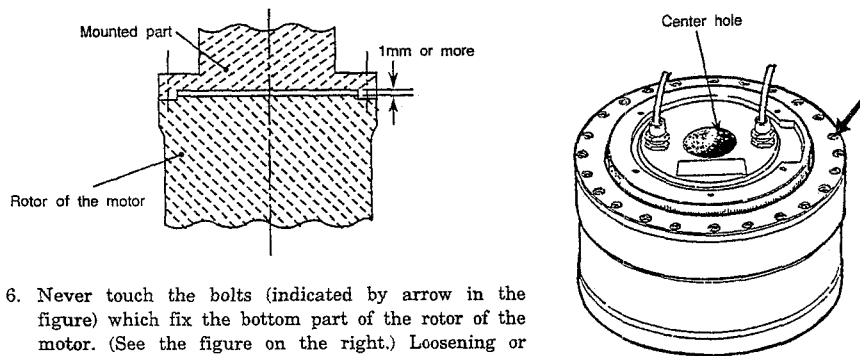
CAUTIONS

- Copying or the reproduction by any means of all or any part of the contents of this manual without permission is strictly prohibited.
- Yokogawa Precision Corp. reserve the right to change the contents of this manual without prior notification.
- While every effort has been made to ensure accuracy in the preparation of this manual, if you should however notice any discrepancies, errors or omissions, kindly contact your dealer or the authorized service personnel of Yokogawa Precision Corp. or it's authorized agency.
- Yokogawa Precision Corp. shall bear no responsibility for indirect or consequential damages such as, but not so as to limit the foregoing, the loss of profit, or the loss of production, caused by the use of our products in accordance with this manual.

-92-

Warning on Installation and Operation

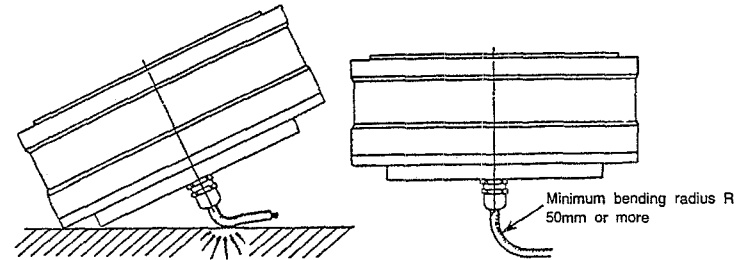
1. Never install the motor with the rotor fixed and the stator set free for rotation.
2. Ensure that the power is switched off when removing the side panel of the driver for jumper setting, etc. Dangerously high voltage is present inside the unit.
3. The motor rotates at high speed with high torque. Beware of the rotating radius of the load when operating the motor with the load installed.
4. Ensure adequate grounding at the ground terminal.
5. When installing a load to the rotor of the motor, allow a space of 1mm or more between the top surface of the motor and the surface of the load in order to maintain the proper alignment of the surfaces. Never apply any force or press fit any materials into the center hole. (See the figure below.)



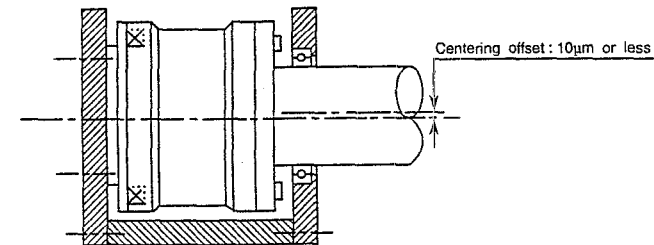
6. Never touch the bolts (indicated by arrow in the figure) which fix the bottom part of the rotor of the motor. (See the figure on the right.) Loosening or tightening these bolts may change the electrical commutation angle, and may result in faulty rotation.
7. Materials easily affected by magnetism must never be brought close to the motor as the surface of the motor is magnetized.
8. Install the motor in an appropriate location as the motor is not dust proof, watertight or oil proof.
9. If the motor is used with oscillating rotation movements with a small angle (50° or less), then carry out a running-in operation with back-and-forth movement about 10 times, each move exceeding an angle of 90° at least. The running-in operation must be carried out every 10,000 times of back and forth oscillation movements in order to ensure proper lubrication of the bearings.
10. Compatibility of the motor with the driver or vice versa when they are of a same model is possible only when they are of the same type. (i.e. when the motor code is DM1□□□□50*1, and the driver code is SD1□□□□52, the □□□□ of the motor and the driver shall be the same.)
11. Never disassemble or modify the motor or the driver. When such disassembling or modification is required, consult Yokogawa Precision Corp. or its authorized agency. Yokogawa Precision Corp. or its authorized agency, accepts no responsibility for disassembled or modified motor and driver.

Warning on Installation and Operation

12. If the motor is placed on the floor or the like as shown below when carrying or installing the DYNASERV, the cable is bent by the weight of the motor and this bending may cut the conductor wire. When placing the motor, be sure to use a supporting base which protects the cable from being bent. The minimum bending radius shall be 50mm or more when installing the motor with the cable being bent. Do not apply bending force repeatedly to the cable when it is used. The cable specifications do not include application with a robot.



13. Appropriate centering and alignment must be carried out when connecting the motor to a load. The shaft metal of the motor may get damaged if the centering offset remains $10\mu\text{m}$ or more.



14. Never carry out a withstanding voltage test. Carrying out this test ever accidentally may damage the circuits. When such withstanding voltage tests are required, consult Yokogawa Precision Corp. or its authorized agency.

CONTENTS

INTRODUCTION	i
Warning on Installation and Operation	ii
1. PRODUCT OUTLINE	1
1.1 DYNASERV, DM/SD Series	1
1.2 Standard Product Configuration	1
1.3 Model and Specification Code	2
2. FUNCTIONAL DESCRIPTION	3
2.1 Motor Section	3
2.2 Driver Section	3
2.3 Driver Panel Surface	4
3. PREPARATION FOR OPERATION	5
3.1 Initial Setting	5
(1) Setting of the Jumper Switches in the Driver Box	5
(2) Jumper Settings Done Prior to Shipment	6
(3) Switch, Volume Settings Done Prior to Shipment	6
3.2 Control Mode Setting	7
(1) Control Mode Types	7
(2) Feedback Pulse and Position Command Pulse Settings /JP1	7
a) <RATE#1 to 2> Jumpers	8
b) <UD/AB> Jumpers	8
(3) Velocity Signal Filter Setting /JP2	8
(4) Origin Pulse Output Signal Setting /z+, z-(Pin#43, 44)	8
(5) Positioning Completion Width Setting /S1	8
(6) Mechanical Resonance Filter(Notch Type) Adjustment (Optional)	9
3.3 External Wiring	10
(1) External Connection Outline Diagram	10
(2) Connection between the Motor and the Driver	10
(3) Typical Wiring Example (In the Position Control Mode)	11
(4) Connection to External Controller	12
(5) List of Interface	13
a) Input	13
b) Output	13
3.4 Installation	15
(1) Motor-section Mounting	15
a) Installation Location	15
b) Mechanical Coupling	15
(2) Driver Section Mounting	16
a) Installation Location	16
b) Mounting Procedure	16
3.5 Wiring Cables	17
(1) Cable Sizes and Rated Currents	17
(2) Wiring Cautions	17

4. OPERATION CAUTIONS	18
4.1 Input and Output Signal Cautions	18
(1) Position Command Pulse Input Signal (PULSE±)	18
(2) Motor Rotating Direction Command Input signal (SIGN±)	18
(3) Velocity Command Input (VIN)	18
(4) Velocity Monitoring Output (VELMON)	18
(5) A/B Phase, UP/DOWN Pulse Output Signals (A/U±, B/D±)	18
a) A/B Phase Output Pulse	18
b) UP/DOWN Output Pulse	19
4.2 Power ON/OFF	19
5. CONTROL MODE AND ADJUSTMENT	20
5.1 Position Control Mode Adjustment	20
(1) I-PD Type Position Control	20
(2) P Type Position Control	20
(3) Position Control System Adjustment Procedure	21
(4) Procedure for Adjustment without Measuring Instruments	22
5.2 Velocity Control Mode Adjustment	23
(1) PI Type Velocity Control	23
(2) P Type Velocity Control	23
(3) Adjustment of Velocity Control System	24
5.3 Torque Control Mode Adjustment	24
6. MAINTENANCE AND INSPECTION	25
6.1 Motor Section	25
6.2 Driver Section	25
7. TROUBLESHOOTING AND MEASURES	25
7.1 Motor Trouble and Measures	25
7.2 List of LED Display	27
7.3 Procedure for Error Correction	28
8. OTHERS	31
8.1 Standard Specification	31
(1) Standard Motor Specification	31
(2) Standard Driver Specification	31
(3) Environmental Specification	32
8.2 Torque vs Velocity Characteristic	32
8.3 Dimensional Outline Drawing	33
(1) Motor (A Series)	33
(2) Motor (B Series)	33
(3) Driver	34
8.4 Driver Block Diagram	35

-76-

1. PRODUCT OUTLINE

1.1 DYNASERV, DM / SD Series

The DYNASERV series are high speed, high torque, highly accurate outer rotor type servo actuators which embody the measurement technology and control technology developed during Yokogawa Electric Co.'s long experience in the field.

The DYNASERV is composed of a motor section incorporating an encoder, and a driver section.

There are two DYNASERV mode types—the 4 models of A Series with an output torque of 50–200N·m, and 4 models of B Series with an output torque of 15–60N·m. The A and B Series have an outer diameter of 264mm and 160mm respectively and a cylindrical hole of 58mm and 25mm diameter respectively.

There is a driver model for each motor.

The drivers are available in types so that they can correspond to motor types. The drivers are available also with 100–115V power supply type and 200–230V power supply type.

Table 1.1 DYNASERV Models DM/SD Series

Motor Model No.	Driver Model No.		Max. Torque (N·m)	Rated Output (W)	Rated Rotation Speed (rps)
	Former SR Driver Exchangeable				
A Series	DM1200A50*1	SD1200A52-□	200	820	1
	DM1150A50*1	SD1150A52-□	150	630	
	DM1100A50*1	SD1100A52-□	100	410	
	DM1050A50*1	SD1050A52-□	50	220	
B Series	DM1060B50*1	SD1060B52-□	60	450	1.5
	DM1045B50*1	SD1045B52-□	45	380	
	DM1030B50*1	SD1030B52-□	30	250	2
	DM1015B50*1	SD1015B52-□	15	125	

□ / 1: 100–115V, 2: 200–230V

1.2 Standard Product Configuration

The standard product set consists of the following components. When after unpacked, ensure that the product corresponds to the correct model, and also ensure that the types and quantities of standard accessories are also correct.

Table 1.2 Standard Products

Part Name	Q'ty	Remarks
Motor section	1	
Driver section	1	
Connector (for CN1 terminal)	1	Manufactured by Honda Tsushin Kogyo : MR-50LM
Connector (for CN2 terminal)	1	Manufactured by Honda Tsushin Kogyo : MR-16LM

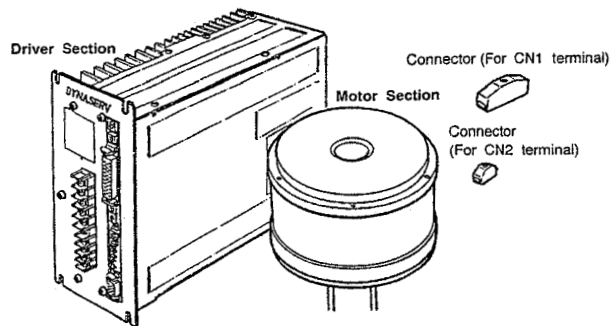
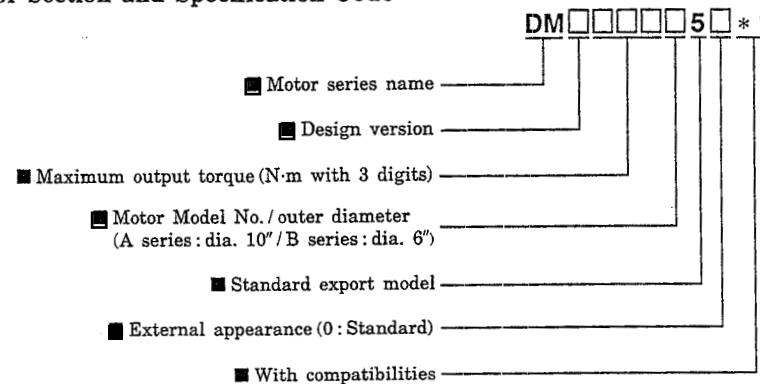


Figure 1.1 Standard Products

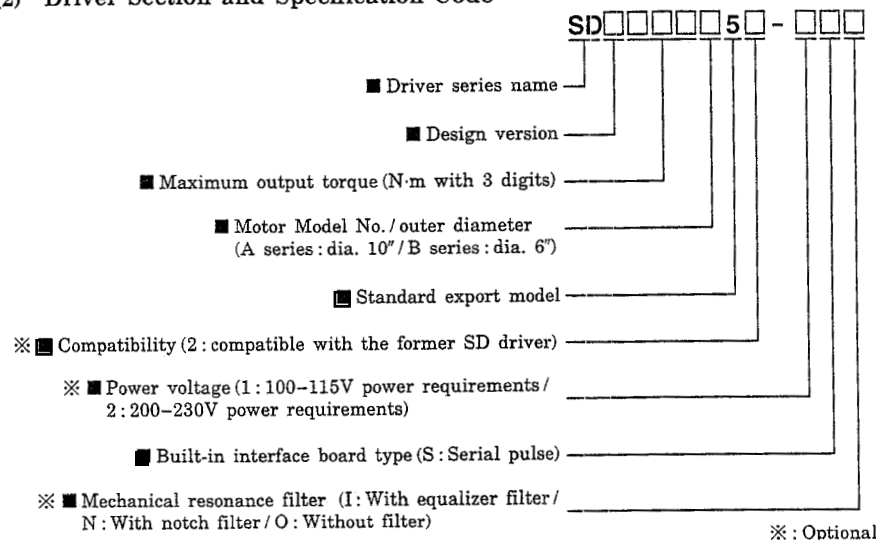
1.3 Model and Specification Code

The DYNASERV, DM / SD Series motor and driver Model Nos. specification code of the rating nameplate are as shown in the following.

(1) Motor Section and Specification Code



(2) Driver Section and Specification Code



Note : The motor and the driver are compatible within the same model type. For compatibility, the upper five digits of the motor code type (DM□□□□□) and the driver code type (SD□□□□□) shall be the same.

-95-

2. FUNCTIONAL DESCRIPTION

2.1 Motor Section

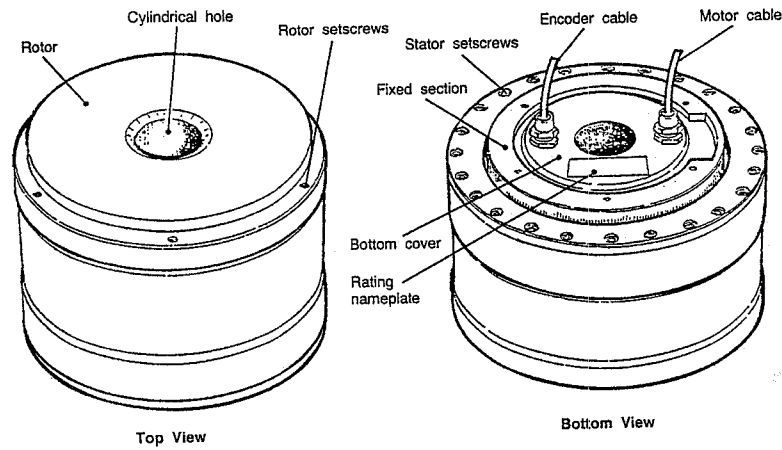


Figure 2.1 Parts Name of the Motor

2.2 Driver Section

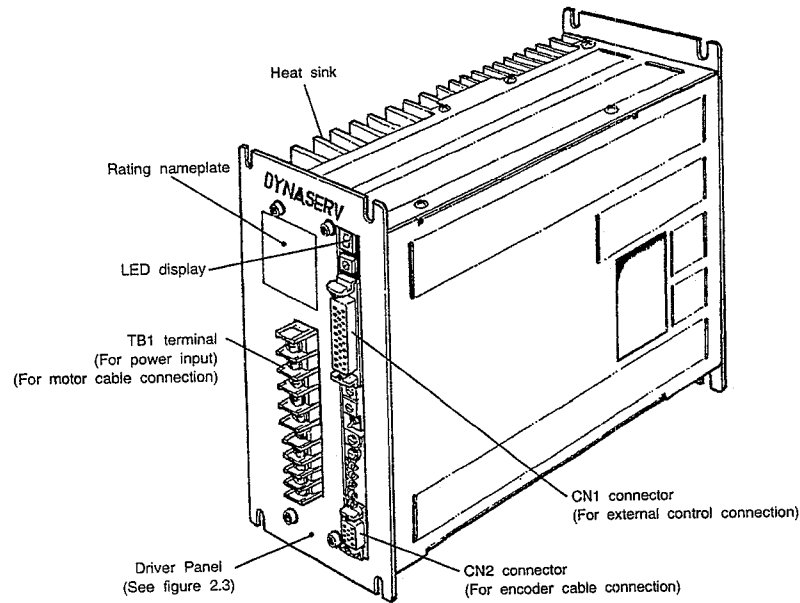


Figure 2.2 Parts Name of the Driver

2.3 Driver Panel Surface

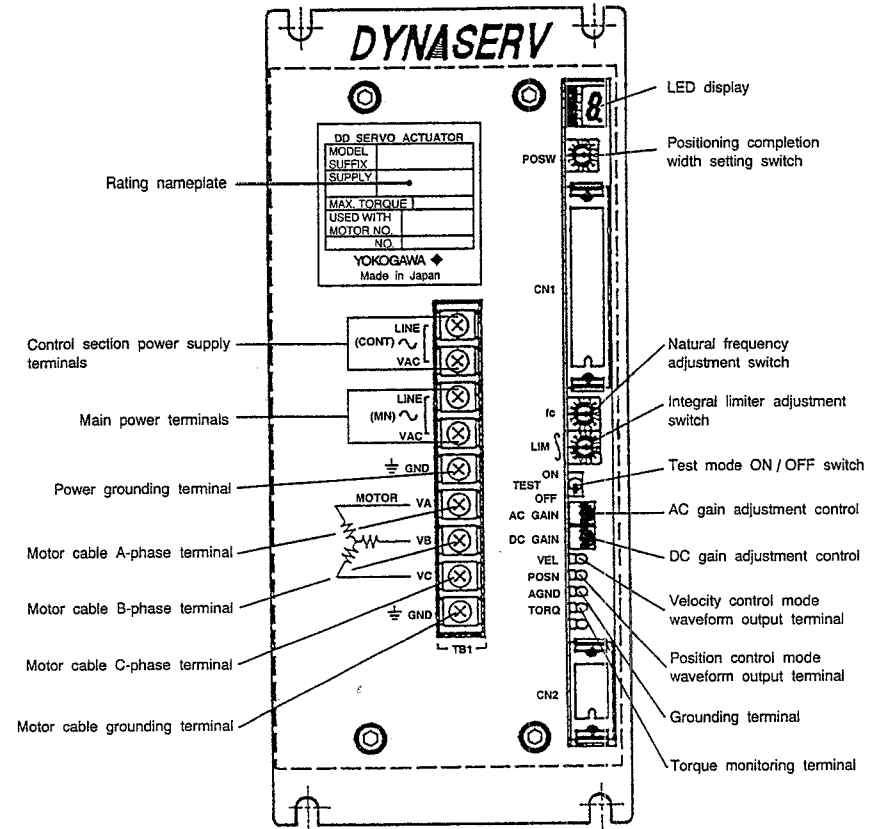


Figure 2.3 Name and Explanation of the Controls and Terminals on the Driver Panel

-96-

3. PREPARATION FOR OPERATION

3.1 Initial Setting

(1) Setting of the Jumper Switches in the Driver Box

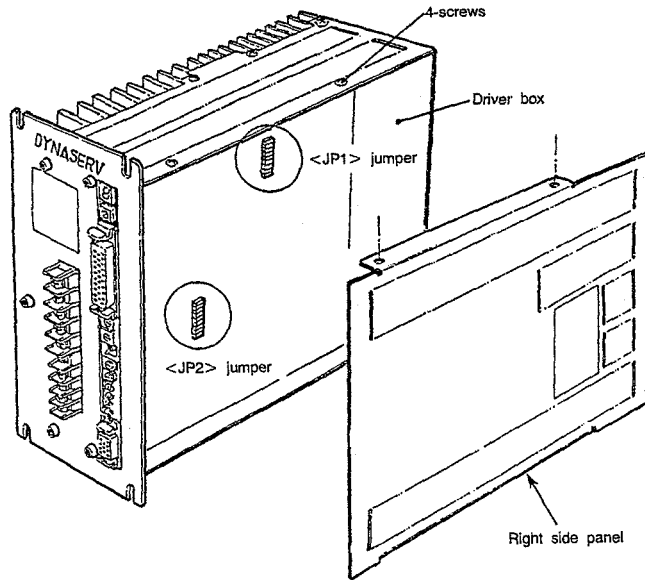


Figure 3.1 Setting of the Jumper Switches in the Driver Box

Certain jumpers, switches and variable resistors within the driver box may need to be set by the customer. However, prior to shipment, they are set as shown on the next page. See the figure above for their locations.

To remove the side plate of the driver box, unscrew the 4-screws shown in the figure above.

CAUTION

However, prior to commencing any operation, always turn OFF the power. Further, never touch the high-voltage generation section, even with the power turned OFF.

For setting and adjustment procedures, see the following pages. Never touch the switches and variable resistors other than those specified.

(2) Jumper Settings Done Prior to Shipment

<JP1> Jumper

- MODE : See the next page
- CALB : See the next page
- RATE#1 : Position command pulse multiplying factor setting
- RATE#2 : Position command pulse multiplying factor setting
- UD/AB : With jumper / A / B-phase, Without jumper / Up / Down pulse
- VFFH : Velocity feed forward amount setting (Note 1)
- VFFM : Velocity feed forward amount setting (Note 1)
- VFFL : Velocity feed forward amount setting (Note 1)
- GAIN H : DC gain magnification setting (Note 2)

<JP2> Jumper

- I : Velocity I type control
- P : Velocity P type control
- 100 : Velocity detection filter (Hz) selection (Open when a mechanical resonance filter is installed)
- 200 : Velocity detection filter (Hz) selection (Open when a mechanical resonance filter is installed)
- PV : Mode selection
- VEL : Velocity input
- TORQ : Torque input
- ALM
- TLIM : Open for standard models

Note : indicates setting prior to shipment.

(Note 1)

VFFH	VFFM	VFFL	Velocity Feed Forward Amount (%)
Shorted	Shorted	Shorted	100
Shorted	Shorted	Open	95
Shorted	Open	Shorted	90
Shorted	Open	Open	85
Open	Shorted	Shorted	80
Open	Shorted	Open	75
Open	Open	Shorted	70
Open	Open	Open	65

(Note 2)

Type	Gain Magnification
With jumper	DC Gain×13
Without jumper	DC Gain×1

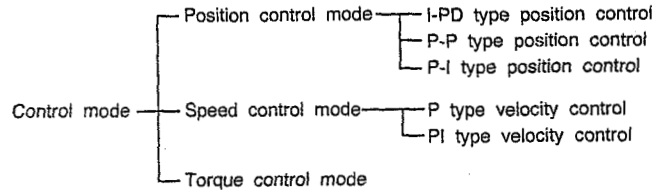
(3) Switch, Volume Settings Done Prior to Shipment

Switch Name Volume Name	Setting Status
DC GAIN	Minimum position
AC GAIN	Minimum position
POSW	Set to "0"
fc	Set to "0"
I. LIM	Set to "0"
TEST	Set to "OFF"

3.2 Control Mode Setting

(1) Control Mode Types

The following 6 control modes are available for the DYNASERV DM /SD Series.



The following table shows the validity or invalidity of the switches and variable resistors related to the control mode and the jumper pin settings for each control mode.

Table 3.1 List of Control Modes and Jumper Pin Switch Settings

Section	Jumper Name Switch Name	Position Control			Velocity Control		Torque Control Mode
		I-PD Mode	P-P Mode	P-I Mode	P Mode	PI Mode	
JP1	MODE	Shorted	Shorted	Shorted	Open	Open	Open
	CALIB	Open	Open	Open	Open	Open	Shorted
	RATE #1	○	○	○	○	○	○
	RATE #2	○	○	○	○	○	○
	UD/AB	○	○	○	○	○	○
	VFFH	○	○	○	×	×	×
	VFFM	○	○	○	×	×	×
	VFFL	○	○	○	×	×	×
GAIN H	○	○	○	○	○	×	
JP2	I	Open	Open	Shorted	Open	Shorted	Open
	P	Shorted	Shorted	Open	Shorted	Open	Open
	100	○	○	○	○	○	○
	200	○	○	○	○	○	○
	PV	Shorted	Shorted	Shorted	Shorted	Shorted	Open
	VEL	Open	Open	Open	Shorted	Shorted	Open
TORQ	Open	Open	Open	Open	Open	Shorted	
	DC GAIN	○	○	○	○	○	×
	AC GAIN	×	×	○	×	○	×
	POSW	○	○	○	×	×	×
	fe	○	○	○	×	×	×
	ILM	○	×	×	×	×	×
	TEST	○	○	○	○	○	×

Note : ○ : Validity, When the set value exerts an influence on motor operation.
 × : Invalidity, When the set value does not exert an influence on motor operation.

(2) Feedback Pulse and Position Command Pulse Settings / JP1

The servo driver receives a signal from the encoder built into the motor, then outputs an A/B phase or UP/DOWN pulse signal to a higher-level controller. Jumper pins related to the feedback pulse signal are <RATE#1 to 2> and <UD/AB>.

In addition, the position command pulse signal multiplication factor is determined by the setting of <RATE#1 to 2>.

a) <RATE#1 to 2> Jumpers

The adjustment of these jumpers can change the position command pulse signal by 1 to 8 times. (See the table on the right.) However, changes in the multiplication factor also change the resolution.

Set Value		Multiplying Factor
<RATE#1>	<RATE#2>	
Shorted	Shorted	1
Open	Shorted	2
Shorted	Open	4
Open	Open	8

b) <UD/AB> Jumpers

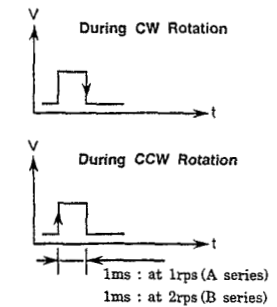
The selection of these jumpers enables the selection of the A/B phase or the UP/DOWN phase. The shorted jumper results in the A/B phase, and the open jumper, the UP/DOWN phase.

(3) Velocity Signal Filter Setting / JP2

These jumpers are used to select the velocity signal filter cut-off frequency. The cut-off frequency is set to 100Hz with <100> shorted, and it is set to 200Hz with <200> shorted. However, when the resonance filter is connected, these jumpers must be kept open.

(4) Origin Pulse Output Signal Setting / z+, z-(Pin#43, 44)

This setting for the detection signal of a reference zero (origin) position is achieved by equally dividing the angles of one rotation of the motor (for series A it is 100 and for series B it is 60). When the reference zero position is detected, the following potential signal will be output. Upon detecting the reference zero position, the potential of the detection signal changes from H (high) to L (low) when the motor is rotating in the CW (clockwise) direction; and it changes from L to H when the motor is rotating in CCW (counterclockwise) direction. The direction CW or CCW in this case are defined as direction of rotation upon viewing the motor from the load side.



(5) Positioning Completion Width Setting / S1

When positioning in the position control mode is completed, the CN1 connector COIN signal is set to ON. This positioning completion width can be selected by the [POSW] switch on the front panel.

The table on the right shows the relationship between [POSW] switches with <POSW 0, 1> signal of the CN1 connector set to H and the positioning completion width.

At the same time, when setting the position completion width using <POSW 0, 1> signal, set the [POSW] switch in 4 steps as shown in the table. With a combination of H and L of the <POSW> signals, the same selection as the [POSW] switch can be carried out.

Table 3.2 Setting of the Positioning Completion Width

[POSW] Switch Setting	Positioning Completion Width (Unit : Pulse)*	<POSW> Signal Setting		POSW Switch
		POSW 1	POSW 0	
0	1	H	H	0
1	5	H	L	
2	20	L	H	
3	100	L	L	4
4	2	H	H	
5	10	H	L	
6	40	L	H	
7	200	L	L	8
8	4	H	H	
9	20	H	L	
A	80	L	H	
B	400	L	L	C
C	8	H	H	
D	40	H	L	
E	160	L	H	
F	800	L	L	

* : 1 pulse = 1/max. resolution

(6) Mechanical Resonance Filter (Notch Type) Adjustment (Optional)

The following explains the adjustment procedure when the mechanical resonance filter (notch type) is installed as an option. The board of the filter is located as shown below just inside the square cut-out on the side panel. The controls $\langle fn_1 \rangle$ and $\langle fn_2 \rangle$ on the board are used to set the notch frequencies at the first stage and the second stage respectively. The frequencies can be set within the range from 150Hz to 1.5kHz (the frequencies are factory-set to 1.5kHz when shipped).

Use the controls $\langle Q1 \rangle$ and $\langle Q2 \rangle$ to change the setting of the Q values. The Q values can be set within a range from 0.5 to 2.5 (0 to 20k Ω) (the Q values are factory-set to 2.5 at the time of shipping). The offset voltage shall be readjusted when the Q value has been changed. This voltage is to be adjusted using adjustment controls so that the voltage difference between $\langle TP1 \rangle$ and $\langle TP3 \rangle$ becomes 50mV or less.

The first-order delay filter is also located on this board. The frequencies can be selected from 20/80Hz, 30/120Hz and 40/160Hz, using a jumper. In addition, using an appropriate pair of C and R, a desired filter frequency can be set. The frequencies of the first-order delay filter are factory-set to 20/80Hz at the time of shipping.

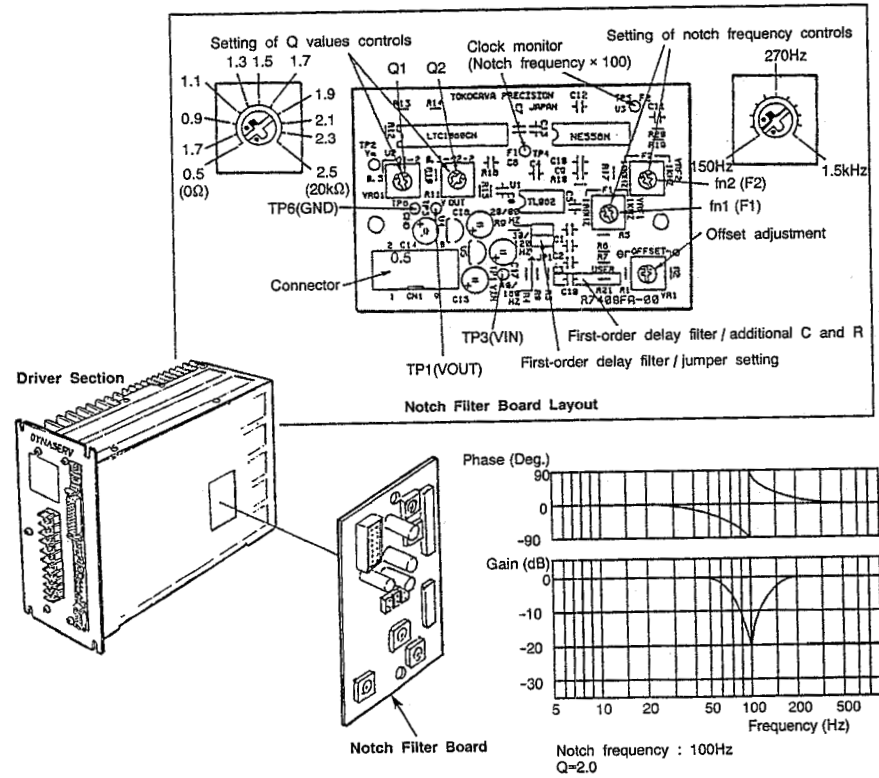
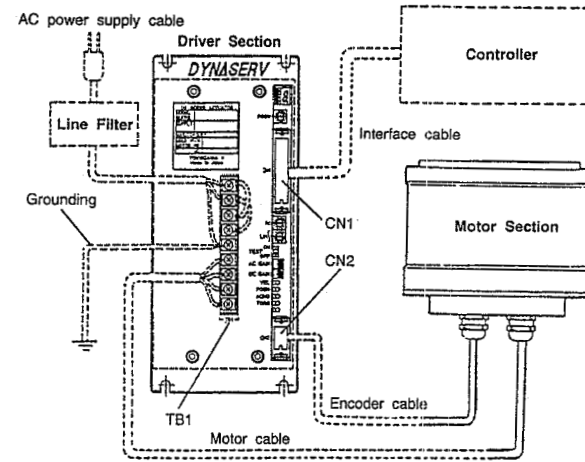


Figure 3.2 Mechanical Resonance Filter (Notch Type) Adjustment (Optional)

3.3 External Wiring

(1) External Connection Outline Diagram



Note : The items shown by the dotted lines should be prepared by the customer.

Figure 3.3 External Connection Outline Diagram

(2) Connection between the Motor and the Driver

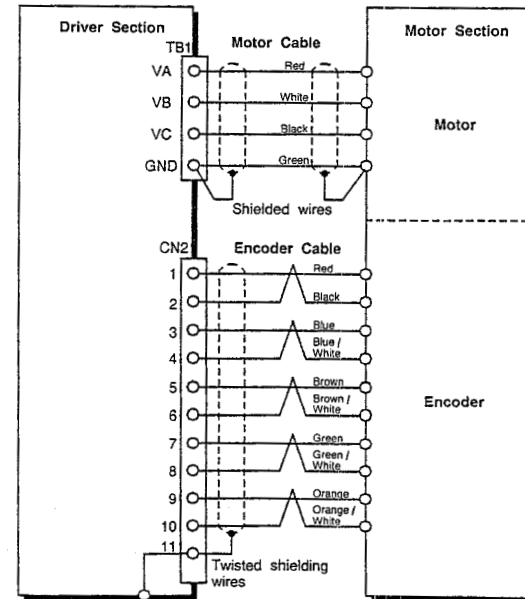


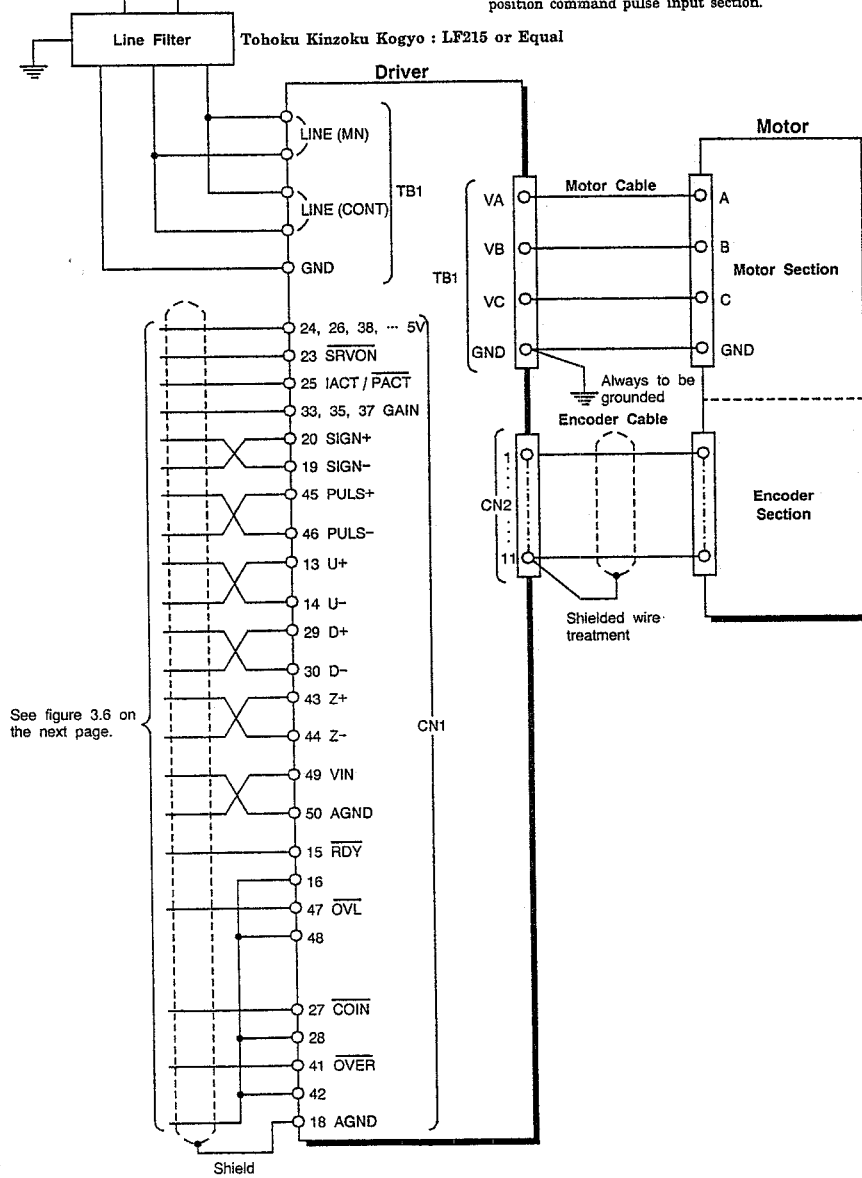
Figure 3.4 Connection between the Motor and the Driver

-66-

(3) Typical Wiring Example (In the Position Control Mode)

100-115V / 200-230V AC +10% to 15% 50/60Hz

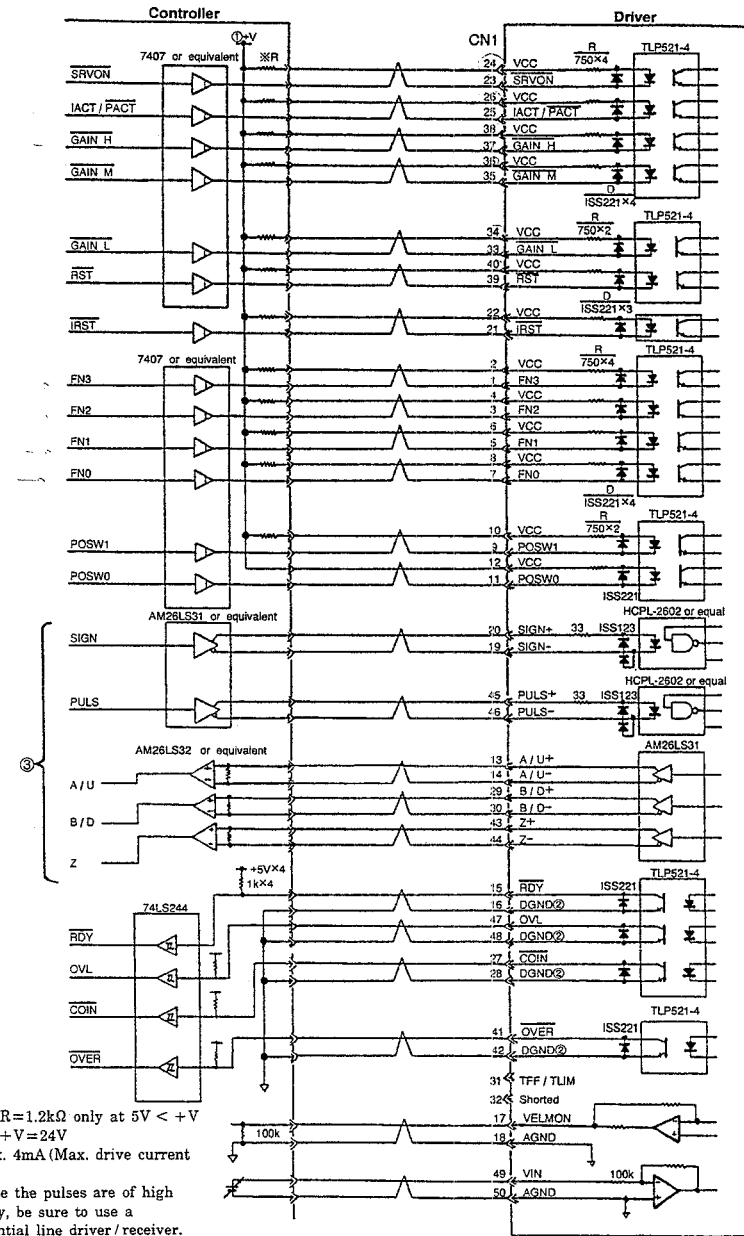
Note : Prepare the 5V power supply on the user's side.
Carry out complete noise rejection treatment in the position command pulse input section.



See figure 3.6 on the next page.

Figure 3.5 Typical Wiring Example

(4) Connection to External Controller
(CN1 terminal I/O signal connection and external signal processing)



- ① Insert R=1.2kΩ only at 5V < +V < 24V + V=24V
- ② Approx. 4mA (Max. drive current 15mA)
- ③ Because the pulses are of high velocity, be sure to use a differential line driver/receiver.

Figure 3.6 Connection to External Connector

(5) List of Interface

a) Input

Table 3.3 List of Input Interface

Signal Name	Pin No.	Meaning	Details
FN 3 FN 2 FN 1 FN 0	1 (2) 3 (4) 5 (6) 7 (8)	Compliance setting (Servo stiffness setting) (See Note 1.)	The signal for setting the <fc> switch on an interface is a 4-bit positive logic binary number which can be set in 16 steps of fc=1 to 16Hz (See Note 2.)
POSW 1 POSW 0	9 (10) 11 (12)	Positioning completion pulse width end (See Note 1.)	Signal for setting a deviation counting value for outputting the positioning completion pulses. Four step setting can be made in any range of 1 to 100, 2 to 200, 4 to 400 and 8 to 800 together with POSW switch setting.
SIGN+ SIGN-	20 19	Rotating direction command	The motor rotates CW with this signal set to H and CCW with the same signal set to L. (When viewed from the load side, it is the same hereafter.)
IRST	21 (22)	Integral capacitor reset	The integral capacitor in the velocity loop is shorted.
SRVON	23 (24)	Servo ON	The motor is set to the servo ON status 0.2 sec. after this signal is set to L to set the driver to the command wait status.
LACT/FACT	25 (26)	Integral/Proportional action selection	Integral action is selected when this signal is set to H and proportional action is selected when this signal is set to L in the position control mode.
TLIM/TFF	31 (32)	Torque Limit Torque feed forward	For input of torque limit and torque feed forward (option)
GAIN H GAIN M GAIN L	37 (38) 35 (36) 33 (34)	Gain selection	Signal to select the variable DC gain range (See Note 3.)
RST	39 (40)	CPU reset	The driver control section is initialized with this signal set to L for more than 50μ seconds.
PULS+ PULS-	45 46	Position command pulse	Driver position command pulse signal
VIN	49	Velocity command input Torque command input	Set to the maximum number of revolutions at ±10V input. CW direction /+10V, CCW direction /-10V. #50 pin: GND. For torque command ±8V
AGND	50	Analog input GND	Velocity/torque input analog GND

Note : () indicates Vcc signal input's terminal.
 (Note1) : FN0 to 3 and POSW 0, 1 are logically wired in the "OR" configuration with the rotary switch and jumper pin (JP1/GAIN H)
 When using external controller, set this rotary switch to the "0" position and GAIN H to open.

b) Output

Table 3.4 List of Output Interface (1/2)

Signal Name	Pin No.	Meaning	Details
A+ / U+ A- / U- B+ / D+ B- / D-	13 14 29 30	Position feedback pulse signal	Pulse signal to indicate the motor rotating position. Either A/B phase or UP/DOWN phase pulse can be selected by the jumper on the board.
RDY	15 (16)	Servo ready	The motor is ready to operate with this signal set to L. This signal is set to the H level about 3 seconds after driver power-ON.
VELMON	17 (18)	Velocity monitoring	Signal for monitoring the number of motor revolutions to output positive voltage for CW rotation and negative voltage for CCW rotation. Velocity detection sensitivity is as shown on the following table. (See Note 3.) Velocity detection sensitivity is not guaranteed for the number of motor revolutions in the range exceeding ±7.5V.

Note : () indicates GND signal output.

Table 3.4 List of Output Interface (2/2)

Signal Name	Pin No.	Meaning	Details
COIN	27 (28)	Positioning completion signal	This signal is set to L when the deviation counter value becomes less than the POSW switch set-value.
OVER	41 (42)	Deviation counter overflow or overspeed	Deviation counter overflow signal is output only in the position control mode, and this signal is set to L when the deviation counter value exceeds 32767. The overspeed signal is set to L when feedback pulse output frequency becomes greater than about 3MHz. It is set to L if the number of motor revolutions exceeds ±7.5 V in the position control or velocity control mode.
Z+ Z-	43 44	Origin pulse	Signal for detecting the original positions obtained by equally dividing 1 revolution of the motor (100 for the A series and 60 for the B series), and changes from H to L during CW rotation and from L to H during CCW rotation.
OVL	47 (48)	Overload	Set to H during overload, it simultaneously reduces motor current automatically to 1/3.

Note : () indicates GND signal output.

(Note 2)

FN 3	FN 2	FN 1	FN 0	fcSW Position	fc (Hz)
H	H	H	H	0	1
H	H	H	L	1	2
H	H	L	H	2	3
H	H	L	L	3	4
H	L	H	H	4	5
H	L	H	L	5	6
H	L	L	H	6	7
H	L	L	L	7	8
L	H	H	H	8	9
L	H	H	L	9	10
L	H	L	H	A	11
L	H	L	L	B	12
L	L	H	H	C	13
L	L	H	L	D	14
L	L	L	H	E	15
L	L	L	L	F	16

(Note 3)

GAIN H	GAIN M	GAIN L	Gain * Magnification
H	H	H	1
H	H	L	4
H	L	H	7
H	L	L	10
L	H	H	13
L	H	L	16
L	L	H	19
L	L	L	22

Note : * : The product of this GAIN value and the variable resistor position (0.5 to 5.5) becomes the total gain.

(Note 4)

Model	Velocity Detection Sensitivity (V / rps)	Velocity Detection Limit (rps)
DM1015B to DM1060B	5 / 2.0	3.0
DM1050A to DM1200A	5 / 1.0	1.5

3.4 Installation

When the product is delivered, first check the product type and Model No. as well as for the presence or absence of accessories and for the exact combination of the motor and the driver.

(1) Motor-section Mounting

The motor-section can be mounted either vertically or horizontally. However, incorrect mounting and unsuitable mounting location may shorten the motor service life and cause trouble. Therefore, always observe the following.

a) Installation Location

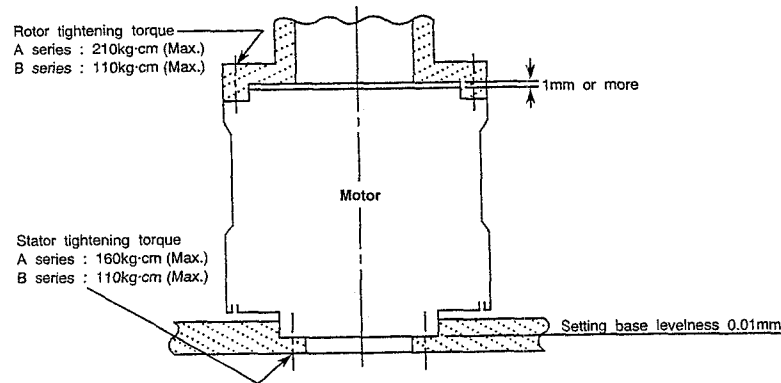
The motor section is designed for indoor use. Therefore, the installation location must be such that:

- There are no corrosive and explosive gases.
- Ambient temperature is between 0 and 45°C
- Dust concentration is low, with adequate air ventilation and low humidity.

Note : The DYNASERV is not drip proof or oil proof, so it should be covered by a suitable drip proof and oil proof cover.

b) Mechanical Coupling

- When coupling a load with the motor rotor section, make sure there is a clearance of more than 1mm between the motor upper surface and the load.
- Secure the motor rotor and stator by tightening the setscrew with torques of less than the following values as given below.
- Motor base levelness deviation must be maintained less than 0.01mm.



Note : When tightening the screws, always apply LOCTITE 601 or equivalent to these screws to lock them.

Figure 3.7 Tightening Torque

(2) Driver Section Mounting

The standard driver is designed for rack mounting.

a) Installation Location

■ When there is a heat generation source near the installation location, ensure that temperature does not exceed 50°C in the approximately of the driver by providing an appropriate heat shield or cover, etc.

■ When there is a vibration generating source near to the driver then mount the driver on the rack with appropriate vibration insulators.

■ Further the installation must be at a location when the humidity is low, and when the surrounding environment is free from high temperature, dust, metal powders and corrosive gases.

b) Mounting Procedure

■ Normally, the driver is rack mounted (L-shaped angle brackets) with its driver panel facing forward and its top and bottom surfaces horizontal. However, it may be mounted with its driver panel facing upward. Always avoid mounting it with its panel surface facing sideways or upside down. (See the figure(b) below.)

■ Mount the driver using 4-screw holes at the top and bottom of the driver panel.

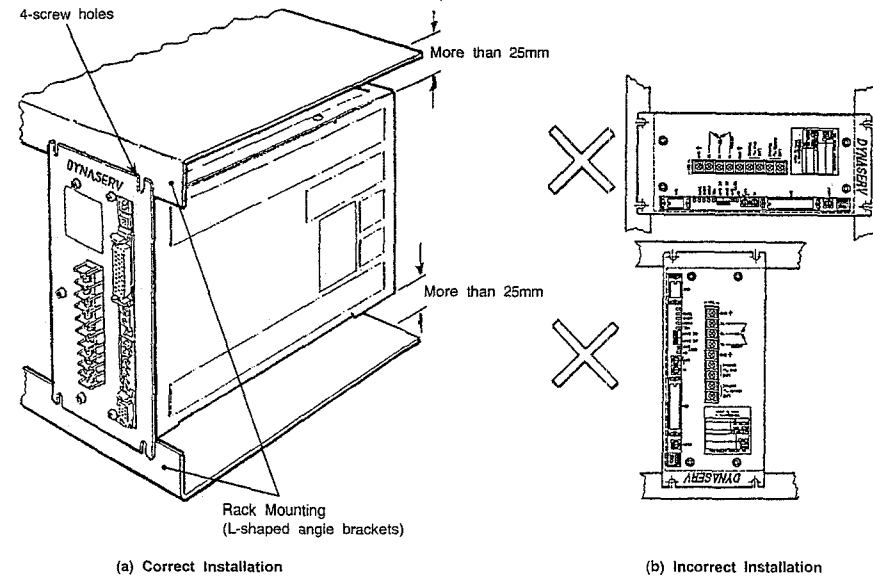


Figure 3.8 Driver Section Mounting

-102-

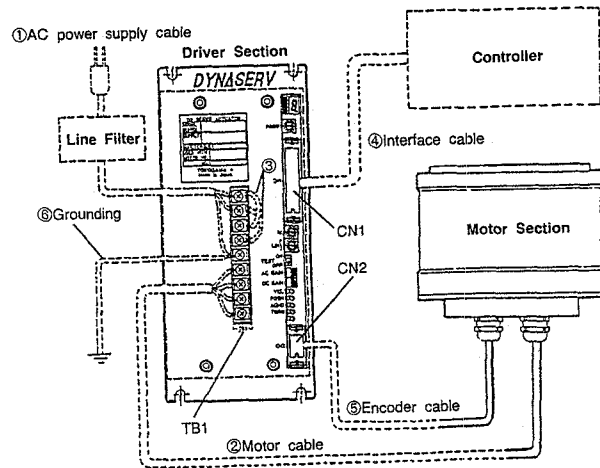
3.5 Wiring Cables

(1) Cable Sizes and Rated Currents

Table 3.6 Cable Sizes and Rated Currents

		A Series	B Series
Input	①AC power supply cable	Current (A)	20
		Cable size	HIV : More than 2.0, Length : Within 30m
	②Motor cable	Current (A)	20
		Cable size	HIV : More than 2.0, Length : Within 30m
③Jumper wire	Current (A)	20	
	Cable size	HIV : More than 2.0m	
Output	④Interface cable	Current (A)	DC100mA·Max.
		Cable size	※Twisted pair collectively shielded wire, Length : Within 3m
	⑤Encoder cable	Current (A)	DC150mA·Max.
		Cable size	※Twisted pair collectively shielded wire, Length : Within 30m
⑥Grounding	Cable size	HIV : More than 2.0m	

- Notes :
1. Current values: r.m.s. of rated currents
 2. Cable size : Cross sectional area in mm²
 3. Cross sectional area of conductor marked with ※ : More than 0.2mm² tin-plated twisted wire
 4. Outer sizes of the cables used for CN1 and CN2 : Less than dia. 14mm or dia. 9mm, respectively
 5. Cable size is obtained under the condition that ambient temperature is 40°C and the rated current flows through 3 bundled leadwires.
 6. HIV : Heat resistant polyvinyl chloride insulated wire maintains insulation resistance up to an operation temperature of 75°C



Note : The items shown by the dotted lines should be prepared by the customer.

Figure 3.9 Wiring Cables

(2) Wiring Cautions

- Use the specified multi-core twisted pair cables with collective shielding for the interface and the encoder cables. Ensure proper end shield connections.
- Use thick conductors as grounding cables as much as possible. Ground the DYNASERV through a resistance of less than 100Ω
- Since high voltage, large current flows through the motor and the AC power cables, ensure proper wiring connections.

4. OPERATION CAUTIONS

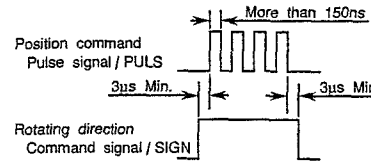
4.1 Input and Output Signal Cautions

(1) Position Command Pulse Input Signal (PULSE±)

This is a drive position command pulse signal. The pulse signal uses positive switching logic with a minimum pulse width of 150ns.

(2) Motor Rotating Direction Command Input Signal (SIGN±)

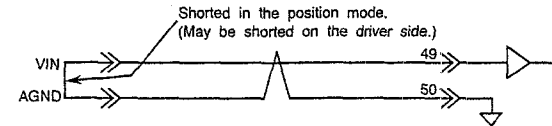
A signal indicating the motor rotation. The motor rotates in the CW direction with this signal set to H and CCW direction with this signal set to L. Timing of this signal with respect of the positioning command pulse signal at the output is as shown below.



Note : The pulse should be set to active H. This means that current does not flow through the driver photo-coupler when the pulse is not output.

(3) Velocity Command Input (VIN)

An analog input signal is used as the motor rotating velocity command value. The maximum velocity in the CW direction at +6V, and the maximum velocity in the CCW direction at -6V. (DR Std Series) (In the -6V to +6V, input range, input impedance is 100kΩ.)

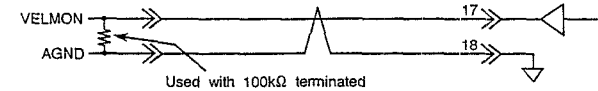


(4) Velocity Monitoring Output (VELMON)

Motor analog velocity monitoring output

Output voltage : At maximum velocity +6V (CW)

At maximum velocity -6V (CCW) (output impedance is 1kΩ.)



(5) A/B Phase, UP/DOWN Pulse Output Signals (A/U±, B/D±)

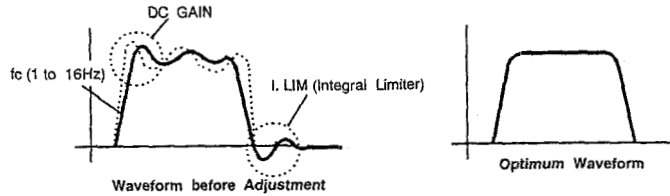
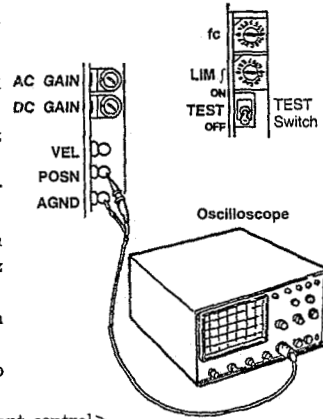
Pulse signals to indicate the motor position. The following 2 pulse output status can be selected by jumpers on the controller board.

(3) Position Control System Adjustment Procedure (See the Following Figure.)

The position control system can be adjusted in the test mode. Turning ON the test switch at the front of the driver generates a 2.5Hz square-wave position command signal inside the driver to output the motor position to the POSN signal terminals. At this time, ensure that the motor exhibits reciprocal action at very small rotating angles.

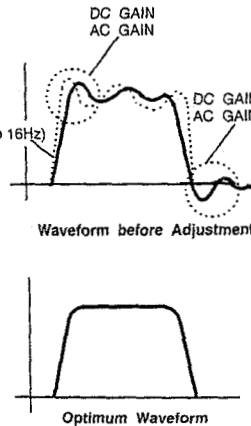
① The adjustment procedure for I-PD type position control in the test mode is as follows.

- Step 1: Connect an oscilloscope to the <POSN> signal terminals.
- Step 2: Set the CN1 connector <SERVO> signal to L. At this time, set the TEST switch to <OFF>.
- Step 3: Set the <TEST switch> at the front of the driver to ON.
- Step 4: Adjust the <fc switch>. Its variable range is from 1 to 16Hz and it should be set to about 10Hz (scale graduation : 9) under normal load conditions. Set the <I. LIM switch> to a large value within the range where there is no hunting. Select the <GAIN H to L signal so that they match the load condition. Fine adjustment is done by the <DC gain adjustment control>.
- Step 5: Perform the above adjustments such that the POSN signal becomes a square wave.
- Step 6: Set the <TEST switch> at the front of the driver to OFF.
- Step 7: Set the CN1 connector <SERVO> signal to H.



② The adjustment procedure for P-I type position control in the test mode is as follows.

- Step 1: Connect an oscilloscope to the <POSN> signal terminals.
- Step 2: Set the CN1 connector <SERVO> signal to L. At this time, set the TEST switch to <OFF>.
- Step 3: Set the <TEST switch> at the front of the driver to ON.
- Step 4: Adjust the <fc switch>. Its variable range is from 1 to 16Hz and it should be set around the center position under normal load conditions. Set the <AC gain control> to a large value within the range in which there is no hunting. Fine adjustment is done by the <DC gain control>.
- Step 5: Perform the above adjustments such that the POSN signal becomes a square wave.
- Step 6: Set the <TEST switch> at the front of the driver to OFF.
- Step 7: Set the CN1 connector <SERVO> signal to H.



(4) Procedure for Adjustment without Measuring Instruments

The preceding section demonstrates the procedure for performing adjustments while monitoring the waveform ; this section demonstrates an adjustment procedure that does not use any measuring instruments. These adjustment methods are valid only in the case of the position control mode (I-PD type, the setting at the time of shipping).

- 1) Calculate or otherwise verify the load inertia. In order to make use of this adjustment method, the load inertia must be known accurately. At this time, calculate the load multiple (K) by dividing the load inertia (J_L : kg-m² units) by the motor (DYNASERV) rotor inertia (J_M).
- 2) Set the <TEST> switch on the driver front panel to [ON].
- 3) Take the computed load multiple and refer to the tables of adjustment settings for the individual DYNASERV models (see pages 22 through 23). For example, suppose that K is [15] for a DM1200A ; thus the "5" range applies for this case. Next, follow this row to the right for the setting values.
- 4) First, look at the value in <DC gain> "Column 1". Because the value is [25], turn the <DC gain> control to [25].
When the value for either A or B series is within range 1 or 2 (DC gain to be set is 5 or less), change the DC gain switching signal to <H> before carrying out the setting.
- 5) Similarly, take the "Column 1" values for <fc> and <LIM f> in the same row, and set their respective controls to those values.
- 6) When the above settings have been completed, set the <TEST> switch to [OFF] to complete the adjustments.

Note : The GAIN value for signal selection shown below is multiplied by the DC GAIN level value to obtain the total gain.

GAIN H	H	H	H	H	L	L	L	L
GAIN M	H	H	L	L	H	H	L	L
GAIN L	H	L	H	L	H	L	H	L
GAIN	1	4	7	10	13	16	19	22

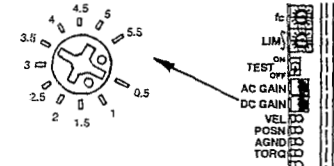


Table 5.1(a) Setting of the DYNASERV Controls (for A Series)

Range	Load Multiple: K			Setting Value			Load Multiple: K			Setting Value		
	SD1200A52	SD1150A52	SD1100A52	DC GAIN	fc	LIM f	SD1050A52	DC GAIN	fc	LIM f		
1	to 1.1	to 0.9	to 0.5	3	E	6	No load	3	E	6		
2	1.1 to 2.2	0.9 to 1.8	0.5 to 1.2	4	D	6	to 0.4	4	D	6		
3	2.2 to 5.4	1.8 to 4.6	1.2 to 3.5	8	C	4	0.4 to 1.8	8	C	4		
4	5.4 to 11	4.6 to 9.3	3.5 to 7.2	13	B	3	1.8 to 4.1	13	B	3		
5	11 to 21	9.3 to 19	7.2 to 15	25	A	3	4.1 to 8.7	25	A	3		
6	21 to 53	19 to 47	15 to 37	50	9	1	8.7 to 23	50	9	1		
7	53 to 75	47 to 66	37 to 52	65	8	2	23 to 32	65	8	2		
8	75 to 107	66 to 94	52 to 75	80	7	1	32 to 46	80	7	1		
9	107 to 160	94 to 141	75 to 112	100	5	2	46 to 69	100	5	2		
10	160 to 320	141 to 282	112 to 224	110	2	4	69 to 138	110	2	4		

-105-

Table 5.1(b) Setting of the DYNASERV Controls (for B Series)

Range	Load Multiple: K				Setting Value		
	SD1060B52	SD1045B52	SD1030B52	SD1015B52	DC GAIN	fc	LIM /
1	to 1.9	to 1.6	to 1.2	to 0.4	3	E	6
2	1.9 to 3.4	1.6 to 3.0	1.2 to 2.4	0.4 to 1.1	4	D	6
3	3.4 to 7.7	3.0 to 6.9	2.4 to 5.7	1.1 to 3.2	8	C	4
4	7.7 to 15	6.9 to 14	5.7 to 11	3.2 to 6.7	13	B	3
5	15 to 30	14 to 27	11 to 22	6.7 to 14	25	A	3
6	30 to 73	27 to 67	22 to 56	14 to 35	50	9	1
7	73 to 103	67 to 93	56 to 78	35 to 49	65	8	2
8	103 to 146	93 to 133	78 to 112	49 to 70	80	7	1
9	146 to 219	133 to 199	112 to 168	70 to 105	100	5	2
10	219 to 438	199 to 398	168 to 336	105 to 209	110	2	4

5.2 Velocity Control Mode Adjustment

In the velocity control mode, the motor rotating angle is controlled so as to correspond to the velocity command voltage (-10V to +10V) from the higher-level controller. The two control methods can be selected in the velocity control mode.

The following table shows the relationship between velocity command voltage and motor velocity.

Model	Velocity / Input Voltage (rps / V)
DM1015B to DM1060B	2 / 10
DR1050A to DM1200A	1 / 10

(1) PI Type Velocity Control

The use of integral / proportional action in velocity control achieves smooth, disturbance-resistant control. This is the same control mode used in the conventional DC/AC servo motor control. In this control mode, only the two <DC gain> and <AC gain> adjustment controls are adjusted.

a) <DC Gain>

The combination of the driver CN1 connector GAIN 0 to 2 signals results in an adjustment range of from 0.5 to 120 times.

b) <AC Gain>

Velocity loop band damping is adjusted.

(2) P Type Velocity Control

Since velocity control is effective only in proportional action, response is fast but is strongly influenced by disturbances in the controlled motor. In this control mode, only the <DC gain> variable resistor at the front of the driver is adjusted. While in this velocity mode, the test switch becomes invalid.

(3) Adjustment of Velocity Control System

Adjustment of velocity control system can be carried out in the test mode.

By turning the test switch on the front of the driver to ON, applies a 2.5Hz square waveform signal to the speed input in the driver, and the motor starts moving back and forth movements, repeatedly, at a small rotating angle. Under this condition, observe the <VEL signal> at the front panel on an oscilloscope, and adjust <DC gain> and <AC gain> so that <VEL signal> becomes an optimum waveform as shown in the figure below.

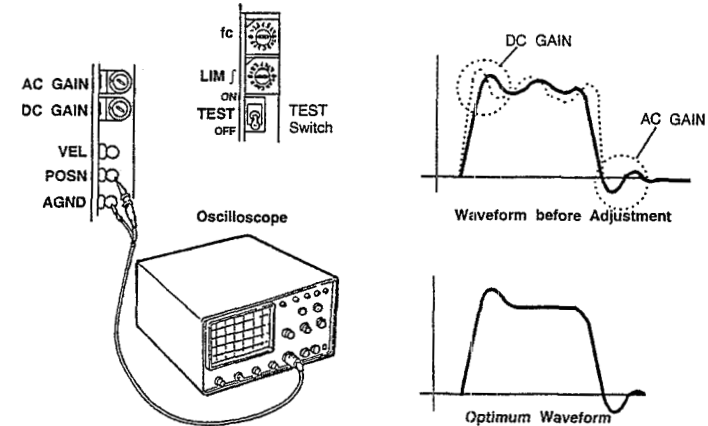


Figure 5.1 Adjustment in the Speed Control System

5.3 Torque Control Mode Adjustment

In the torque control mode, current flows through the motor corresponding to the current command voltage (-8V to +8V) from the higher-level controller. Motor output torque depends on the current. Therefore, torque is 0 at 0V of command voltage, and the maximum torque is produced at 8V.

Note : When desirous of using the torque control mode, carefully plan and design the velocity & position control loops and a proper interlocking system such that, the final control system meets the exact specifications of the application.

-106-

6. MAINTENANCE AND INSPECTION

6.1 Motor Section

Only simple daily checks need be carried out on the Motor section. Check for noise or excessive vibration which is not normal. Never disassemble the Motor section. If the condition of the Motor section is not normal after 20,000 hours of use or after five years from the installation, replace the Motor section together with the Driver section. This time duration may change depending on the environmental and operating conditions where the Motor is used.

6.2 Driver Section

There is no need of daily maintenance for the Driver part. However, clean the Driver part periodically to prevent it from poor insulation caused by accumulated dust.

7. TROUBLESHOOTING AND MEASURES

7.1 Motor Trouble and Measures

Whenever any abnormal condition occurs while operating the motor, check the LED display on the front panel of the driver. Take appropriate countermeasures as shown below if the cause of the abnormal condition is determinable by the indication of the LED display.

When the motor does not function normally, even after the following measures have been taken, immediately cease operation and contact the Yokogawa Precision Corp. or it's authorized agent.

Table 7.1 Motor Troubleshooting and Measures (1/2)

Trouble	Estimated Cause	Inspected Item	Measures	Page for Reference
The motor is not servo locked.	◆ No AC power is fed.	Wiring inspection	Apply the specified AC power	Pages 10, 11
	◆ The servo ON (SRVON) terminal is set to H.	Inspection	Set to L.	Page 12
	◆ The CPU reset (RST) terminal is set to L.	Inspection	Set to H.	Page 12
	◆ The integral capacitor reset (IRST) terminal is set to L.	Inspection	Set to H.	Page 12
	◆ fc, I LIM, DC gain is small.	Inspection	To be adjusted to an appropriate value.	Pages 21 to 24
The motor does not start.	◆ Under over load	Operate the motor under no load.	When starting the motor, lighten the load or replace the motor with a large output motor.	
	◆ Incorrect external wiring	Inspect wiring.	Re-wire correctly by referring to the connection diagram.	Pages 10, 11
	◆ fc, I LIM, DC gain is small.	Inspection	To be adjusted to an appropriate value.	Pages 21 to 24
Motor rotation is unstable.	◆ Imperfect connection	Check the connection of each phase of A, B, C and GND.	Re-wire correctly by referring to the connection diagram.	Pages 10, 11
	◆ The motor and driver combination is inappropriate.	Check the combination Nos. on the nameplate.	If the combination is incorrect, then return the pair to YPC or it's authorized agent.	Page 2

Table 7.1 Motor Troubleshooting and Measures (2/2)

Trouble	Estimated Cause	Inspected Item	Measures	Page for Reference
The motor overheats.	◆ Ambient temperature is high.	Check to see if ambient temperature is more than 45°C	Lower the ambient temperature to below 45°C	
	◆ The motor is overloaded.	Operate the motor under no load.	When starting the motor, lighten the load, or replace the motor with a larger output motor.	
Abnormal sound is produced.	◆ Incorrect mounting	Loosen set screws.	Tighten the screws.	
	◆ Bearing trouble	Check for sound and vibration near the bearing.	Motor replacement (Contact us.)	
	◆ Mounting base vibration	Check the mounting base.	Reinforce the mounting base.	
Abnormally small motor torque	◆ Incorrect motor / driver combination	Check the combination Nos. on the nameplate.	If the combination is incorrect, then return the pair to YPC or it's authorized agent.	Page 2
	◆ Motor is overloaded.	Check the OVL signal.	Recheck the operation. Lighten the load.	Pages 13, 14
	◆ fc, I LIM, DC gain is small.	Inspection	Adjust to appropriate value.	Pages 21 to 24
Motor runs out of control.	◆ Incorrect motor / driver combination	Check the combination Nos. on the nameplate.	If the combination is incorrect, then return the pair to YPC or it's authorized agent.	Page 2
	◆ Inappropriate jumper setting	Inspection	Perform correct jumper setting.	Pages 5 to 9
	◆ Imperfect connection	Check motor / encoder connection.	Re-wire correctly by referring to the connection diagram.	Pages 10, 11
Position is dislocated.	◆ Incorrect A/B-phase and U/D-pulse jumper, selection	To be inspected.		Pages 5 to 9
	◆ Command pulse rate and width are not as specified.	Check the command pulse width.		Pages 19, 20
	◆ Feedback pulse rate and receive circuit response speed are not as specified.	Check the feedback pulse rate (3MHz Max.) and receive circuit response speed.		Pages 19, 20
	◆ Both ends of the feedback pulse transmission cable shield are not connected to the earth.	To be inspected. If so, connect the driver to AGND and the controller to SG.		

7.2 List of LED Display

A seven segment LED is mounted on the front panel of the driver to display the normal/ abnormal status of the motor and driver. Display details are as shown in the following tables.

Table 7.2 List of LED Display

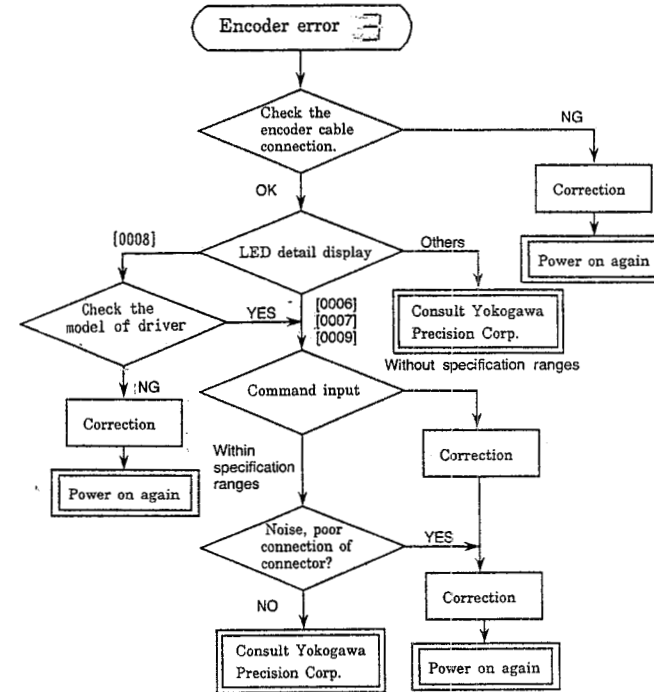
LED Display		Status	Display Details with TEST Switch ON (Serial display of 4-digit hexadecimal notation)	Measures / Page for Reference
Servo OFF	Servo ON			
0	0.	Normal status	No detail display	
1	1.	Speed over	No detail display	See pages 28 to 30
2		RAM error	Indiscriminate	Reparation required
3		Encoder error	0000 : Open circuit (SIG Ø, SIG 1 stop) 0001 : Open circuit (SIG 1 stop) 0002 : Open circuit (SIG Ø stop) 0006 : Abnormal frequency (Smoother error) 0007 : Abnormal frequency (Incorrect interruption, detected) 0008 : Error induced due to mechanical eccentricity of motor's internal construction. 0009 : Abnormal frequency (Divided error)	See pages 28 to 30
5		Error of power supply	Indiscriminate	Reparation required
6		Over count shut down	Indiscriminate	See pages 28 to 30
	E.	Counter overflow	No detail display	See pages 28 to 30
7		ROM error	ROM checksum code 4 digits	Reparation required
8		Abnormal main power supply	No detail display	See pages 28 to 30
8.		Reset status of driver Power supply error	No detail display	Release reset Reparation required
9		CPU error	0000 : Watch dog timer (WDT) error	Reparation required
A		Amplifier error	0001 : Over voltage (OVV) signal ON 0003 : Over current (FAULT) signal ON	See pages 28 to 30
E	E.	Overload error	No detail display	See pages 28 to 30

Note : Consult Yokogawa Precision Corp. or it's authorized agency when reparation is required.

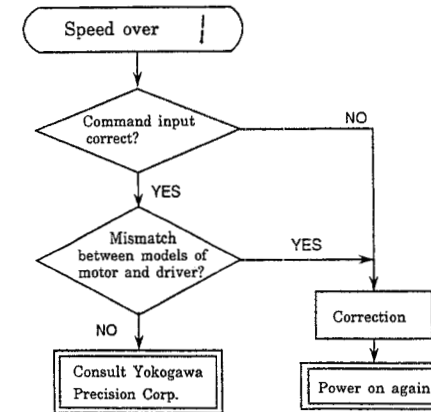
-807-

7.3 Procedure for Error Correction

(1) Encoder Error

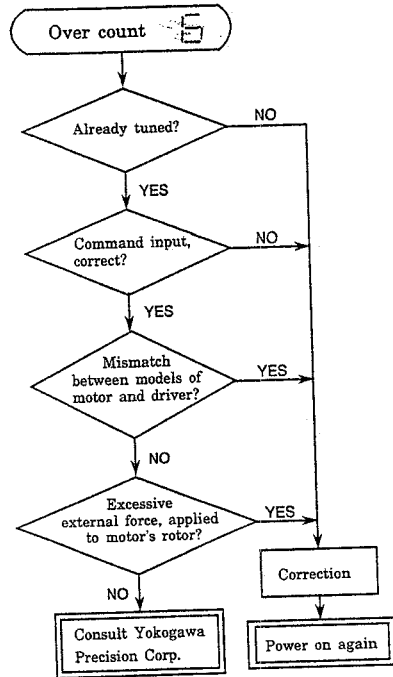


(2) Over Speed

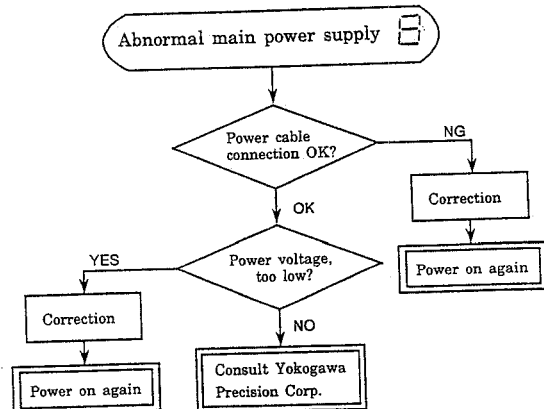


-109-

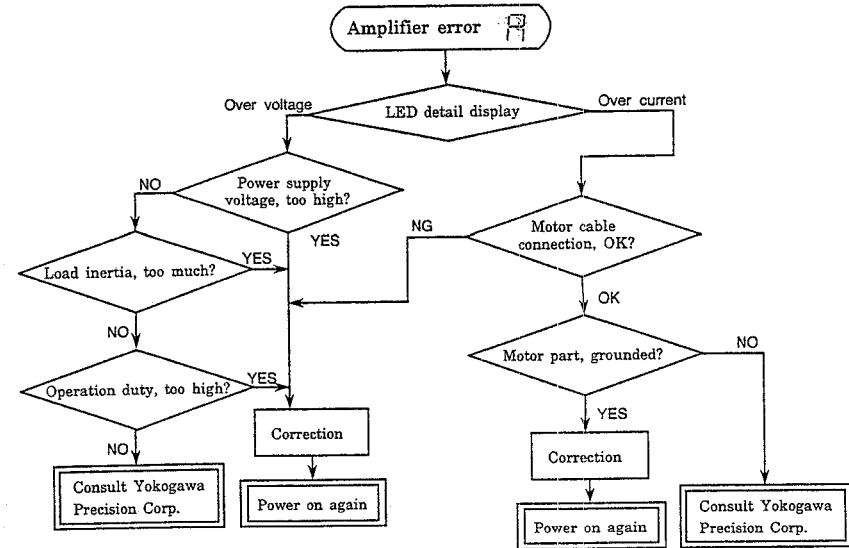
(3) Over Count



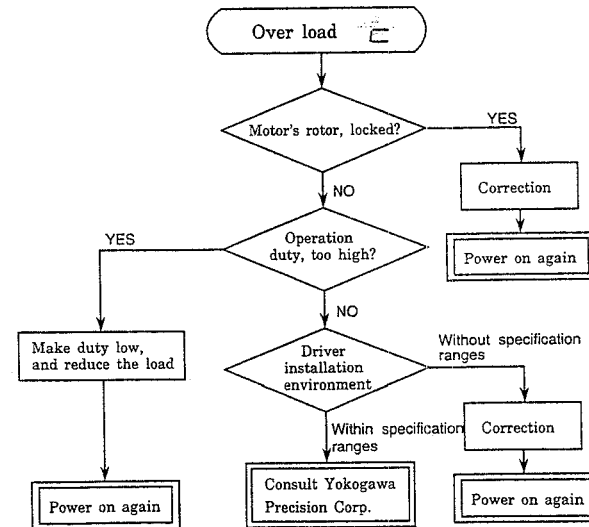
(4) Abnormal Main Power Supply



(5) Amplifier Error



(6) Over Load



8. OTHERS

8.1 Standard Specification

Item	Unit	A Series		B Series	
		□DM1200A50-1	□DM1100A50-1	□DM1060B50-1	□DM1045B50-1
Maximum output torque	N·m (kgf·m)	200 (20)	150 (15)	60 (6.0)	45 (4.5)
Rated rps (100V/200V)	rps	0.5/1.0	1.0/1.0	1.0/1.5	1.0/2.0
Encoder resolution	p/rev	1000	1000	30 (3.0)	15 (1.5)
Positioning repeatability	sec	±2	±2	±2	±1.5
Positioning accuracy	sec	±16	±16	±2	±1.5
Velocity ripple	%	3	3	3	3
Permissible axial load	N(kgf)	4×10 ⁴ (4×10 ³)	2×10 ⁴ (2×10 ³)	3×10 ⁴ (3×10 ³)	1×10 ⁴ (1×10 ³)
Permissible moment load	N·m (kgf·m)	400 (40)	2×10 ⁻⁶ (2×10 ⁻⁵)	200 (20)	2.5×10 ⁻⁶ (2.5×10 ⁻⁵)
Permissible axial load	mm/N	3×10 ⁻⁶ (3×10 ⁻⁵)	4×10 ⁻⁷ (4×10 ⁻⁶)	3×10 ⁻⁶ (3×10 ⁻⁵)	1×10 ⁻⁶ (1×10 ⁻⁵)
Axial stiffness	mm/kgf	19	24	12	0.6
Moment stiffness	rad/N·m (rad/kgf·m)	14.5	19	168	148
Weight	kg	188	163	23×10 ⁻³	19×10 ⁻³
Length L (See dimensional view.)	mm	107×10 ⁻³	142×10 ⁻³	119×10 ⁻³	86×10 ⁻³
Rotor inertia	kg·m ²	107×10 ⁻³	142×10 ⁻³	119×10 ⁻³	86×10 ⁻³

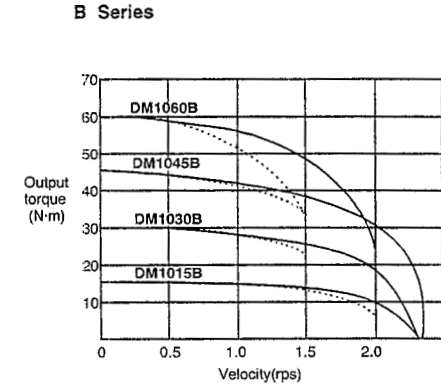
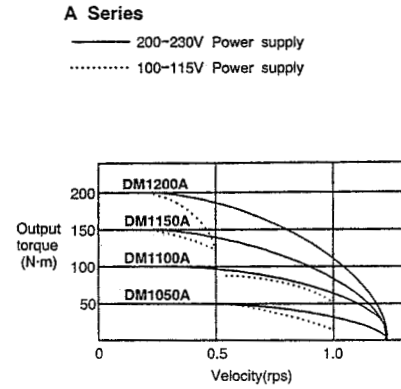
(2) Standard Driver Specification

Driver Name	Series Name	A Series		B Series	
		SD1100A52-1	SD1000A52-1	SD1060B52-1	SD1045B52-1
Per 100V Power Requirements	SD1100A52-1	SD1000A52-1	SD1060B52-1	SD1045B52-1	SD1045B52-1
Per 200V Power Requirements	SD1100A52-2	SD1000A52-2	SD1060B52-2	SD1045B52-2	SD1045B52-2
Speed input signal	Analog voltage: DC±10V/1.2pps Serial pulse/1.25MHz MAX.				
Positioning input signal	Analog voltage ±8V/Maximum torque Serial pulse/1.57MHz MAX.				
Torque command signal	Analog voltage ±8V/Maximum torque Serial pulse/1.57MHz MAX.				
Rotation direction command signal	H: CW / L: CCW				
Speed output	+6V (CW) to -6V (CCW)				
Encoder output	Phase A, Phase B (30KHz Max.) Zero positioning signal (100p/rev.) Phase A, Phase B (30KHz Max.) Zero positioning signal (60p/rev.)				
Alarm output	Over-current, Over-voltage, Heat sink temperature alarm, Voltage low alarm, Encoder abnormal, CPU abnormal				
Monitor output	2.5Hz Step response output (Pulse mode)				
Maximum input power capacity (VA)	3.0	3.0	2.7	2.4	2.0
Power source	100-115V or 200-230V AC ±10%, -15% 50/60Hz				
Weight (kg)	6				

(3) Environmental Specification

		Motor Section	Drive Section	Notes
Ambient operating conditions	Temperature	0 to 45°C	0 to 50°C	
	Humidity	20 to 85% R.H	20 to 90% R.H	Non condensing
Stage conditions	Temperature	-20 to 85°C	-20 to 85°C	
	Humidity	20 to 85% R.H	20 to 90% R.H	Non condensing
Operating environment		No corrosive gases, Dust-free environment		

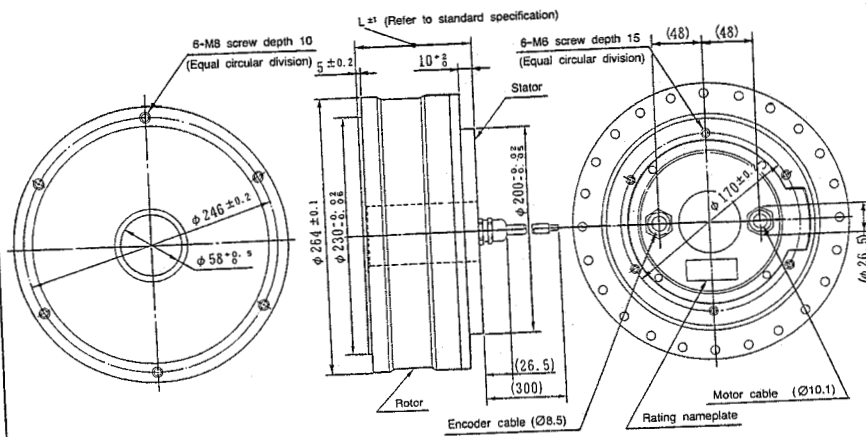
8.2 Torque vs Velocity Characteristic



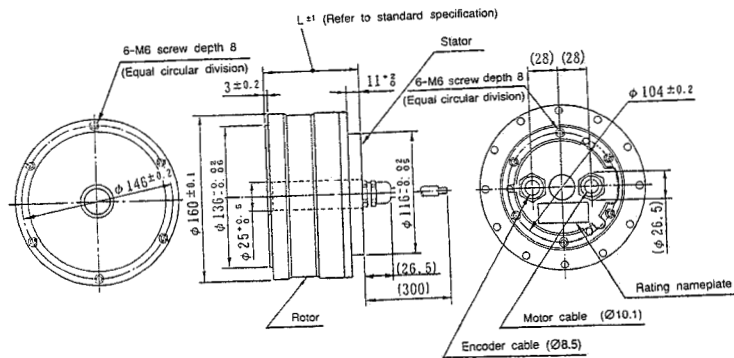
8.3 Dimensional Outline Drawing

Unit : mm

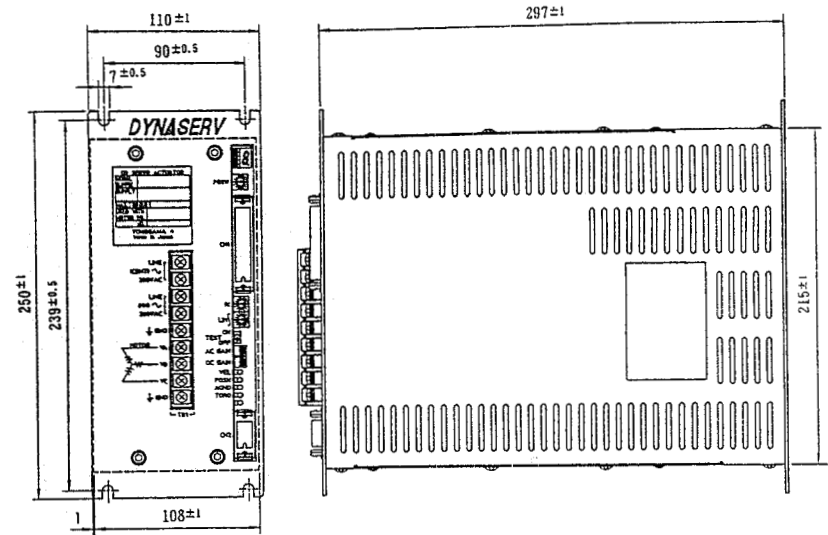
(1) Motor (A Series)



(2) Motor (B Series)

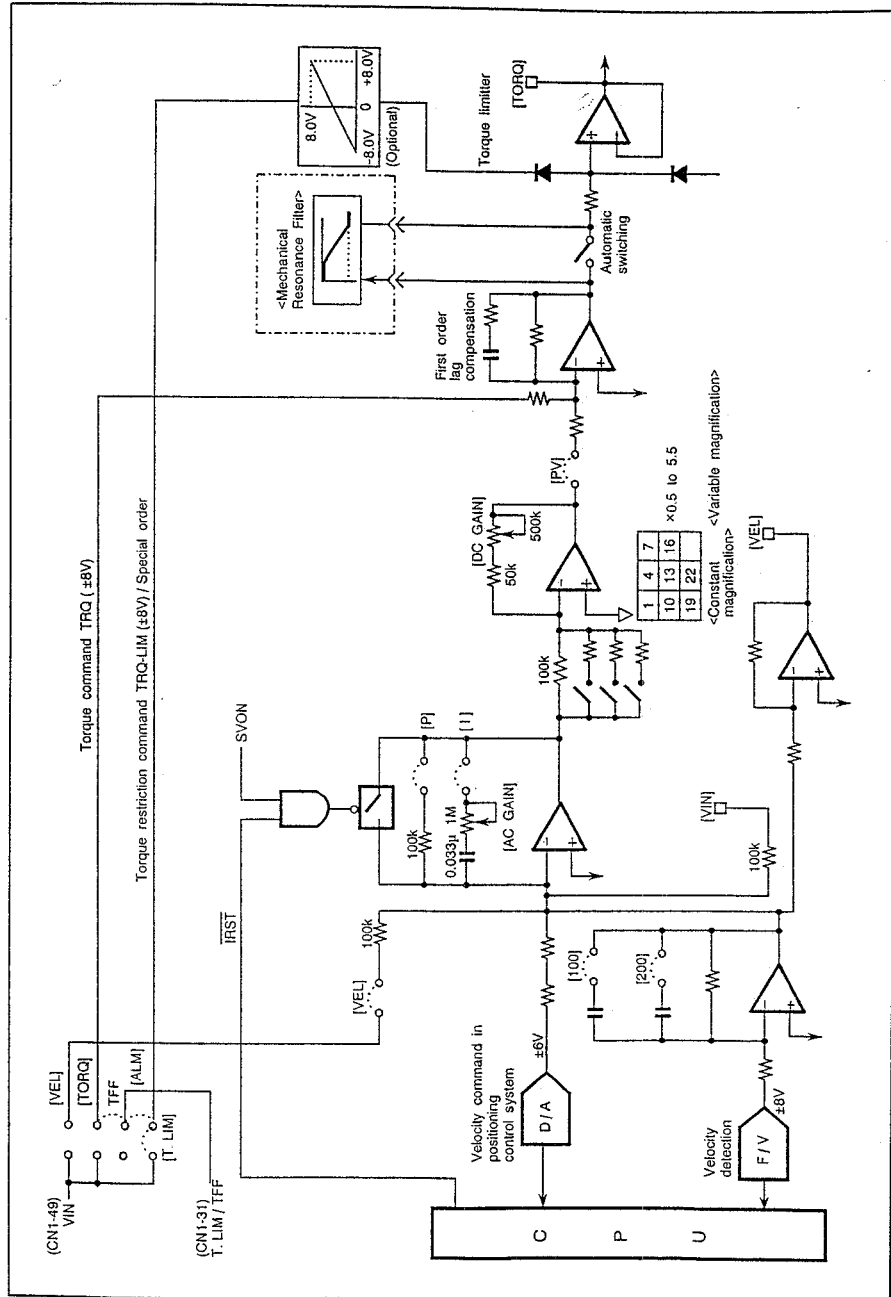


(3) Driver



Unit : mm

8.4 Driver Block Diagram



- 112 -

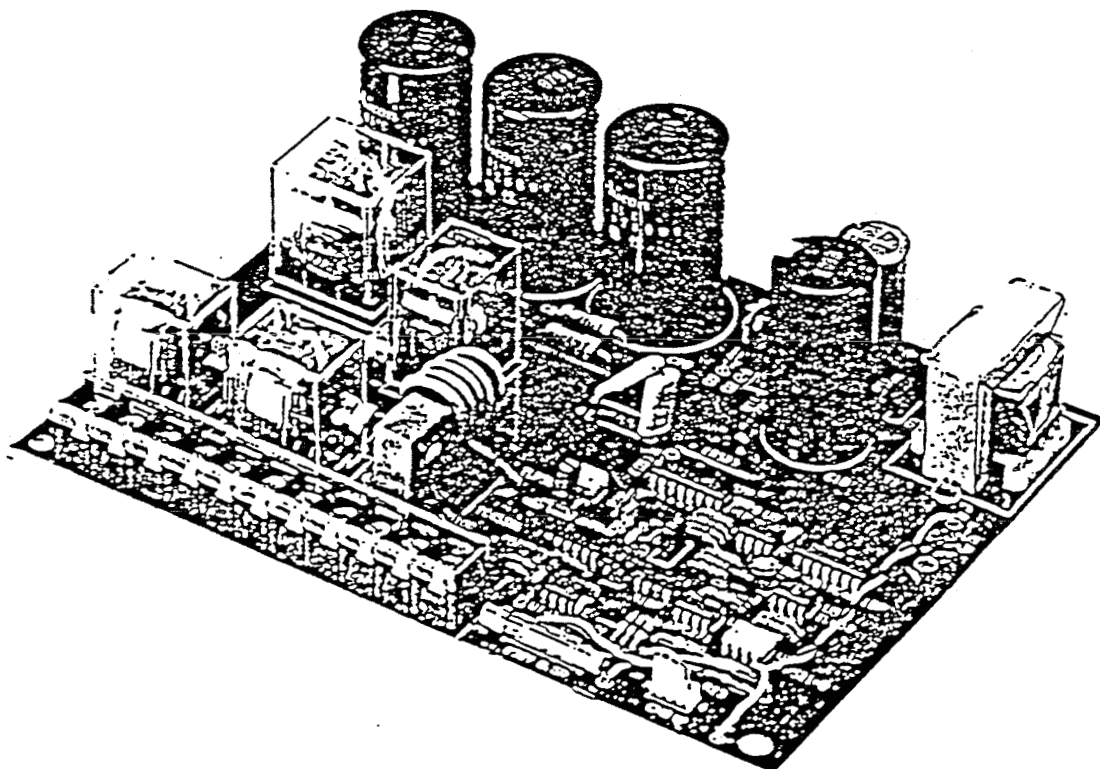
B.4 Manual: BE-A/B Type Dynamic Brake

Instruction manual, 10 pages

Instruction Manual

BE-A/B Type Dynamic Brake
(DYNASERV dedicated
board type)

Instruction Manual



-114-

Introduction

Thank you for purchasing our DYNASERV-dedicated dynamic brake. This generation type brake was developed for DYNASERV DD motors and is of simple construction, making it easy to assemble into industrial machinery such as robots with built-in DYNASERV and hence, suitable for a wide range of applications.

This instruction manual describes those items that are considered to be necessary when using this brake so that its functions and usage cautions are fully understood prior to commencing operation.

Cautions

- Copying of part or all of the contents of this manual is strictly prohibited.
- The contents of this manual may be subject to change without notice.
- This manual is prepared carefully, but if any mistakes and/or omissions are found, please contact our sales or service representative immediately.
- Any damage or indirect damage due to our unintentional mistakes as a result of operation in accordance with this instruction manual may not our responsibility.

Contents

1. Outline	3
2. Operational Cautions	3
3. Specifications	5
4. Model No.	5
5. Product Configuration	6
6. Dimensional Outline Drawing and Mounting Diagram	7
7. Interface	8
8. Wiring	10
9. Circuit Configuration and Operation	12
(1) Circuit configuration	12
(2) Operation	12
(3) Interface circuit configuration	13
(4) Signal and operation status	14
10. Operation Procedure	15
(1) Preparation	15
(2) Connection	15
(3) Operation procedure	15
11. Trouble and Measures	16

1. Outline

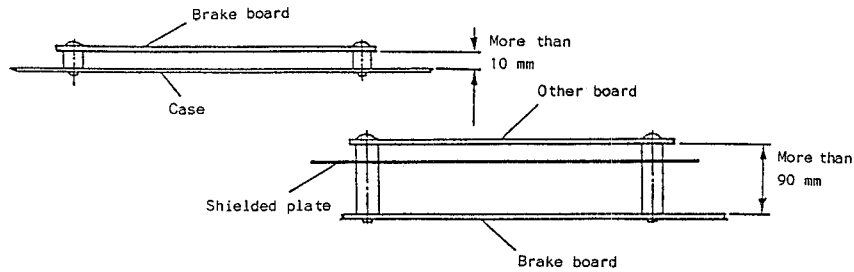
This negative action type electric power generation brake was developed especially for DYNASERV and has the following features.

- ◆ Simple construction ... No mechanical devices are required in the motor section, and efficient torque control is achieved simply by connecting this brake between the motor and driver sections.
- ◆ Power failure compensation ... Even during power failure, the same control torque as that at power-ON is available, through the use of a built-in power failure compensator.
- ◆ Both velocity change and capacitor types are available. ... Both velocity change and capacitor versions of this brake are available. The former is suitable for either high or low velocity applications while the latter is for use in the high-velocity area, making the model range suitable for a wide range of applications.
- ◆ Maintenance free

2. Operational Cautions

- (1) Because the brake was designed especially for use with DYNASERV, it may not display its specified performance when used with other motors.
- (2) When the brake has been activated, do not attempt to rotate the stopped motor by force, as doing so may overheat the internal resistor.
- (3) When coupling this brake to a DYNASERV, do not mistake the connection of the A, B and C phases and GND, especially in the motor and driver sections, as doing so may stop DYNASERV from operating normally.
- (4) When this brake is operated repeatedly, operate it at minimum intervals of 1 minute, otherwise the internal resistor may overheat.
- (5) Both 100 V AC and 200 V AC power supply voltages are available. Always check to make sure that the correct one is being used.

(6) When installing the brake board, separate it from the case by more than 10 mm or keep it away from other boards more than 90 mm when the other board is laid on top of the parts installation side of the brake board. (See the figure below.) Also, if necessary, install a shielded plate as shown in the following figure.



3. Specifications

3. 1 Common Item and Enviromental

Parameter	Value
Structure	Printed board type
Dimensions(L×W×H) [mm]	210×170×55 (Max.)
Weight [g]	1,000
Operating Temperature [°C]	0~50
Operating Humidity [%]	20~90RH (Non condensing)
Storage Temperature [°C]	-20~85
Storage Humidity [%]	20~90RH (Non condensing)
Atmosphere	No corrosive gases, Dust-free atmosphere
Power source	100/200VAC+10%-15% 50/60Hz
Power consumption [W]	10 (Max.)

3. 2 Braking specification

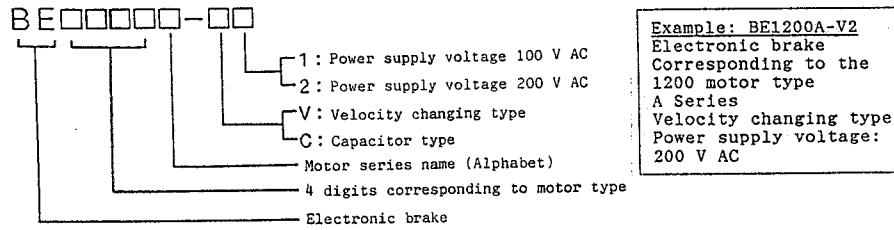
Model	Applied DYNASERV Model	Load cond. ($J_m \times 30$) [kg·m ²]	Speed cond. (Max.) [rps]	Braking angle [°] (Typical)	
				Speed changing type	Capacitor type
BE1015B	DM1015B	0.372	2.4	389	396
	DR1015B	0.651	"	795	805
BE1030B	DM1030B	0.465	2.4	279	287
	DR1030B	0.744	"	541	551
BE1045B	DM1045B	0.589	2.4	247	258
	DR1045B	0.806	"	418	430
BE1060B	DM1060B	0.713	2.4	216	231
	DR1060B	1.023	"	369	385
BE1075B	DM1075B	0.837	2.4	216	231
	DR1075B	1.023	"	369	385
BE1050A	DM1050A	2.976	1.2	226	232
	DR1050A	5.580	1.8	665	676
BE1100A	DM1100A	3.689	1.2	158	162
	DR1100A	6.200	"	256	260
BE1150A	DM1150A	4.402	1.2	152	157
	DR1150A	7.130	"	244	248
BE1200A	DM1200A	5.177	1.2	178	180
	DR1200A	8.835	"	304	305
BE1300A	DR1300A	10.540	1.0	96	101
BE1400A	DR1400A	12.400	0.8	60	64
BE1070E	DR1070E	2.635	2.4	711	724
BE1100E	DR1100E	3.100	2.4	627	640
BE1130E	DR1130E	3.875	1.2	174	181
BE1160E	DR1160E	4.340	1.2	157	164
BE1220E	DR1220E	5.270	1.2	145	152
BE1250E	DR1250E	5.735	1.2	136	145

(Note) J_m : Rotor inertia of DYNASERV

-116-

4. Model No.

Product Model No. has the following meaning.

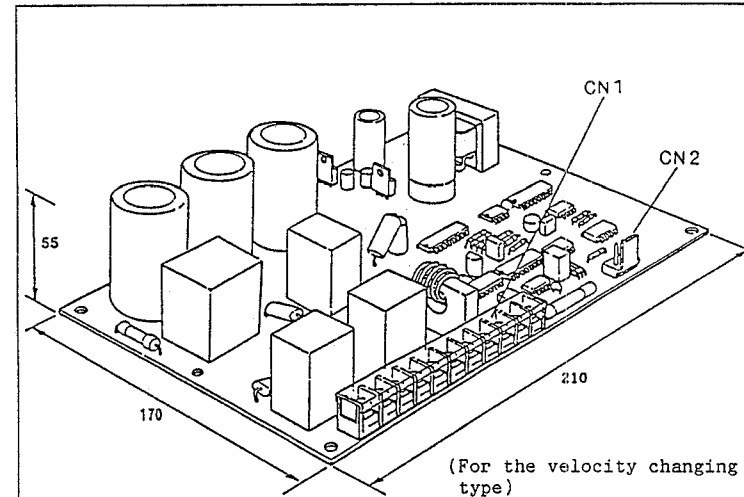


5. Product Configuration

This brake consists of the following device and accessories when it is purchased.

Name	Q'ty	Remarks
Mainframe	1	
Standard accessories	Connector	1 Type No. 5051-04 (Made by MOLEX)
	Terminals	4 Type No. 5159TL (Made by MOLEX)

6. Dimensional Outline Drawing and Mounting Diagram (Unit: mm)



(For the velocity changing type)
 Figure 6-1 Dimensional Outline Drawing of Dynamic Brake

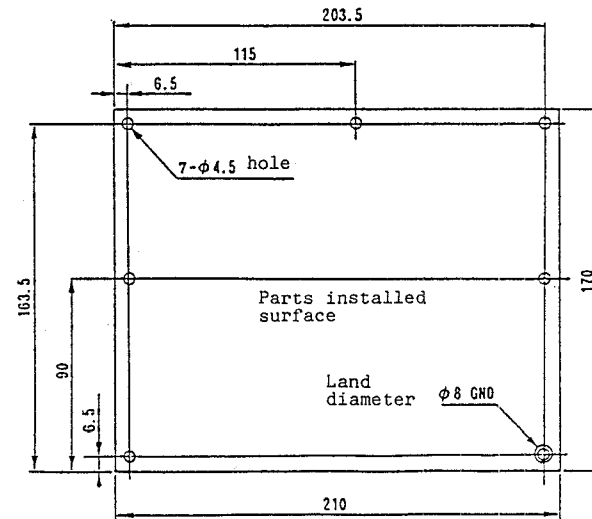


Figure 6-2 Mounting Diagram

-117-

7. Interface

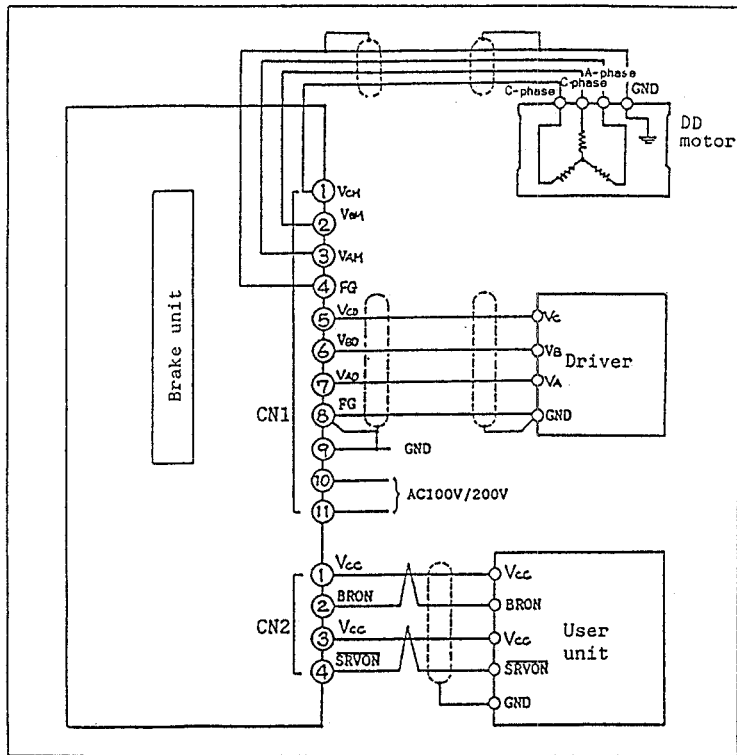


Figure 7-1 Interface Connection Diagram

Connector No.	Signal name	Meaning
CN1-1	V _{CM}	Motor C-phase
2	V _{BM}	Motor B-phase
3	V _{AM}	Motor A-phase
4	FG	Frame grounding
5	V _{CD}	Driver C-phase
6	V _{BD}	Driver B-phase
7	V _{AD}	Driver A-phase
8	FG	Frame grounding
9	GND	Grounding
10	AC	AC input
11	AC	AC input

Connector No.	Signal name	Meaning
CN2-1	V _{CC}	Power supply voltage
2	BRON	Brake ON
3	V _{CC}	Power supply voltage
4	SRVON	Servo ON

V_{CC} = + 5V to + 12 V

- 811 -

8. Wiring

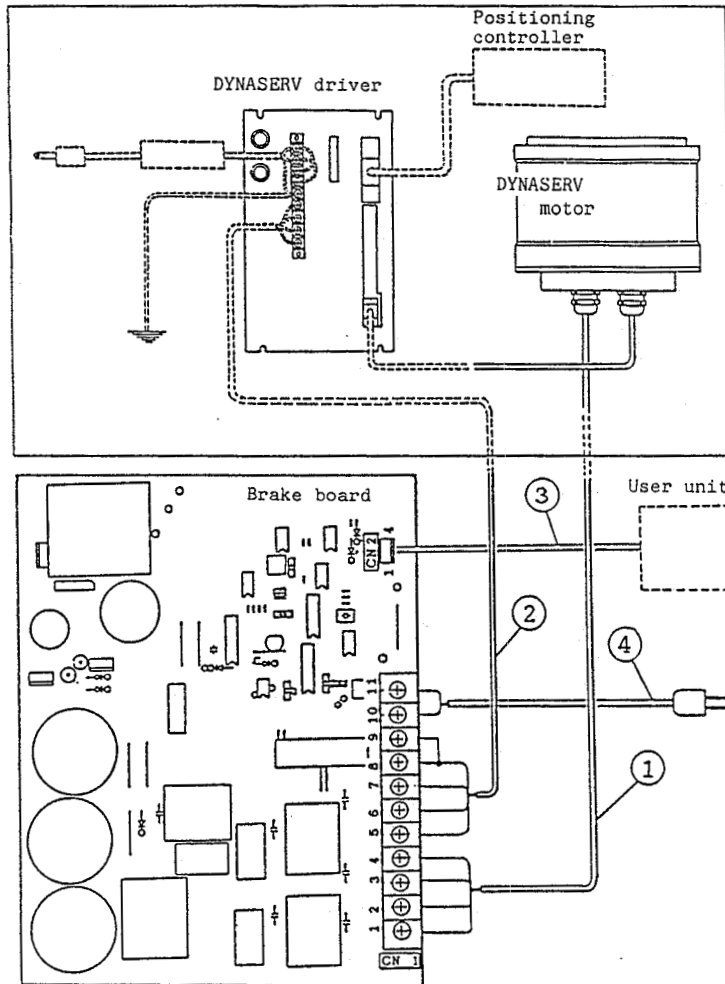


Figure 8-1 Wiring Diagram

Notes: Cables used for wiring related to the brake in the above wiring diagram should be as follows.

- ① Same specifications as those of the power cable from the motor: Current capacity 20 A (A Series), or 15 A (B series)
Cable size: HIV 2.0 mm² or more,
Length : less than 30 m (① + ②)
- ② Same as in ①
- ③ Current capacity: More than 100 mA DC
Twisted pair collectively shielded wire (core cross section: More than 0.2 mm², zinc plated, twisted soft copper wire) Length: Less than 30 m
- ④ Current capacity: More than 1A AC

To avoid miss operation by electrical noise, wiring should take care of as follows

- (1) To insert surge current absorb circuit, when used solenoid, relay, and other magnetic switch on line or closed.
- (2) To insert noise filter on power source line and input signal line, when existing high frequency noise generated source on line or closed.

- 611 -

9. Circuit Configuration and Operation

(1) Circuit configuration

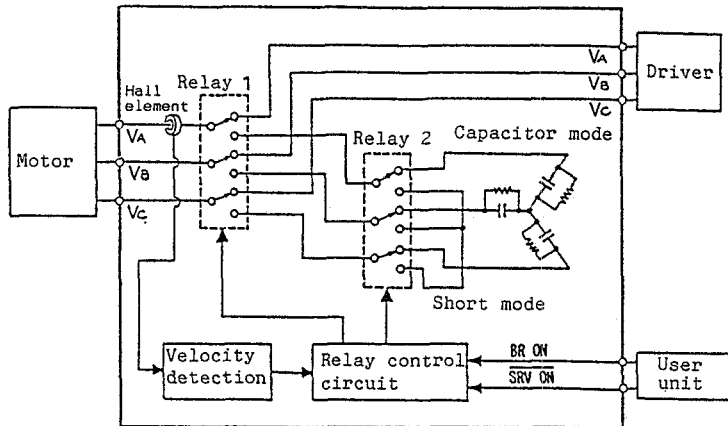


Figure 9-1 Circuit Configuration Block Diagram

Note: In the capacitor type circuit configuration there is no "Relay-2", but the motor output is directly connected to the capacitor circuit from "Relay-1".

(2) Operation

• Velocity changing type

Both high and low-velocity types are effective. Greater braking torque is obtained with the coil shorted than that of the capacitor type at low velocity. It is possible automatically to select the capacitor mode and the short mode by detecting the velocity.

If the <BRON> and <SRVON> signal from the user unit is set to "H" or the power supply is suspended, "Relay-1" is turned OFF and the motor is connected to the capacitor mode of "Relay-2".

Next, as shown in Figure 9-2, braking torque in the capacitor mode becomes smaller than in the short mode, so "Relay-2" is turned OFF to short the motor.

• Capacitor type

This is effective in the high-velocity region. The effect of motor coil inductance becomes large at high velocity, and therefore it is restricted by a capacitor which converts rotating energy to thermal energy. Operation is the same as that of the velocity changing type, but no short mode is available.

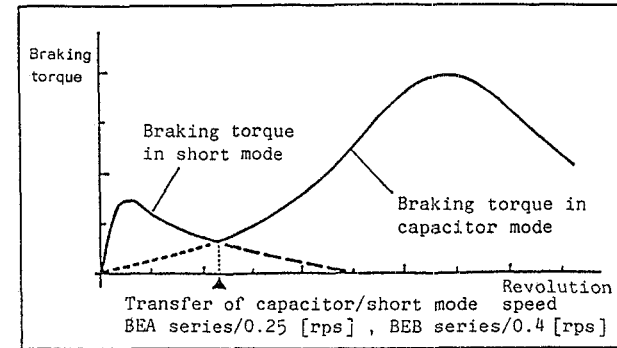


Figure 9-2 Generated Torque Diagram

From the above, when selecting the brake, refer to the following table.

Motor rotation speed [rps]		Braking
A Series	B Series	
0 to 1.2	0 to 2.4	Velocity changing type
0.25 to 1.2	0.4 to 2.4	Capacitor type
0 to 0.25	0 to 0.4	Short mode

(3) Interface circuit configuration

The CN2 terminal on this brake consists of a photocoupler as shown in the following diagram.

Inputting <VCC> activates the circuit, but the "H" or "L" input signal in this case has the following meaning.

"H": <VCC> level voltage (VCC = +5 V to +12 V)

"L": <GND> level voltage

Note: Other pins ③ and ④ on the CN2 terminal have the same circuit configuration.

-120-

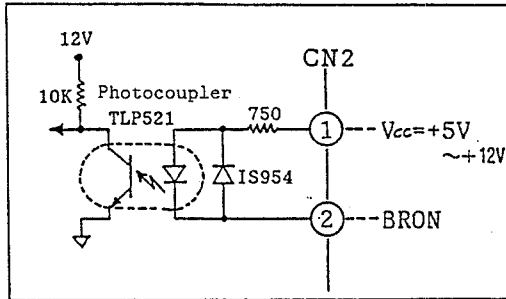


Figure 9-3 Interface Circuit Configuration

(4) Signal and operation status

<BRON>	<SRVON>	<Relay 1>	Connection with motor	Brake power
-	-	OFF	Short	OFF
L	L	ON	Driver	ON
L	H	ON	Driver	ON*
H	H	OFF	Capacitor, short	ON**

Note: *: In this status, a robot can be programmed directly.

** : When the power is turned from ON to OFF, this means a power failure has occurred, but operation is the same as in the power-ON status due to power failure compensation.

(Note) The relay chatters for few second only after the power source is turned off. However this is not trouble by back up capacity of power circuit capacitor.

10. Operation Procedure

(1) Preparation

Turn OFF the driver power of DYNASERV in use.
Remove the power cable from the motor.

(2) Connection

Connect and wire the brake between the motor and the DYNASERV driver in accordance with the connection diagram.

(3) Operation procedure

- ① Prior to turning ON the DYNASERV power, turn ON the brake power.
- ② Disconnect the connection of the brake CN2 terminal, or set the <BRON> and <SRVON> signal to "H".
- ③ Check to see if the brake is activated in this status.
- ④ Next, connect the brake's CN2 terminal, then set the <BRON> and <SRVON> signals to "L".
- ⑤ Check to see if the brake is released in this status.
- ⑥ Turn ON the DYNASERV's driver power.

Set the <SRVON> signal to "L" and the mode to the test mode, then check to see if the brake is activated normally.

If the above operation causes no abnormality, the brake is operating normally.

-121-

11. Trouble and Measures

Trouble	Probable cause	Inspecting items	Measures
The motor does not start	Incorrect connection	Check the connection of the motor's A, B and C phases and GND.	Make sure that the wiring is correct by referring to the wiring diagram.
	Incorrect power supply wiring	Check the wiring.	Make sure that the wiring is correct by referring to the wiring diagram.
	Fuse burnt out in the brake circuit	Check the fuse.	Replace the board if the fuse is burnt out.
Motor rotation is instable.	Incorrect connection	Check the connection of the motor's A, B and C phases and GND.	Repair the faulty section.
The brake is not activated.	Incorrect signal wiring	Check the wiring.	Make sure that the wiring is correct by referring to the wiring diagram.

Note: If it is assumed that the brake is faulty, the board itself should be replaced. Therefore, do not replace or repair any parts on the board.

If the brake fails or is damaged in the normal operating status due to defect attributed to faulty manufacture, within one year of the date of purchase, it will be repaired free of charge.

-122-

B.5 Cabinet manual

Instruction manual, 10 pages



Centrale Technische Dienst
Stafgroep E/E

Documentatie nummer

3506/001/23/23/01

Onderwerp

DYNASERU ACTUATORS

Project : 3506/001/23/23/01, 16-01-1997 1/1

Dok.nr : 35060012.323

Bedr.Ond : GTD/EE

Betreft : **Dynaserv Servo-Actuator**



GEMEENSCHAPPELIJKE TECHNISCHE DIENST

GROEP ELEKTRONICA / ENERGIETECHNIEK

ONDERWERP : *Dynaserv Actuators*

OPDRACHTGEVER : *J.J.F.J.Garenfeld*

TELEFOON : 2824

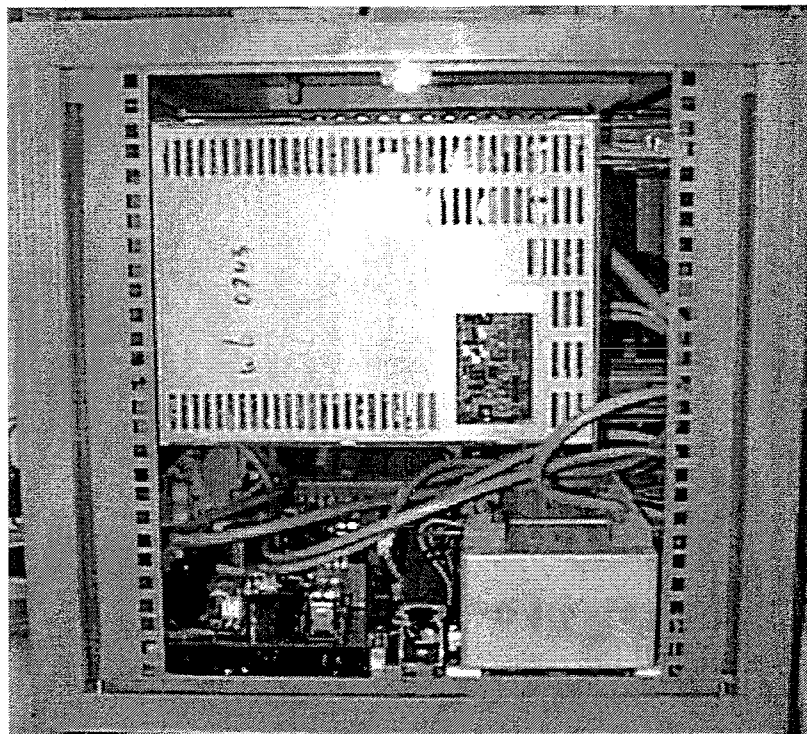
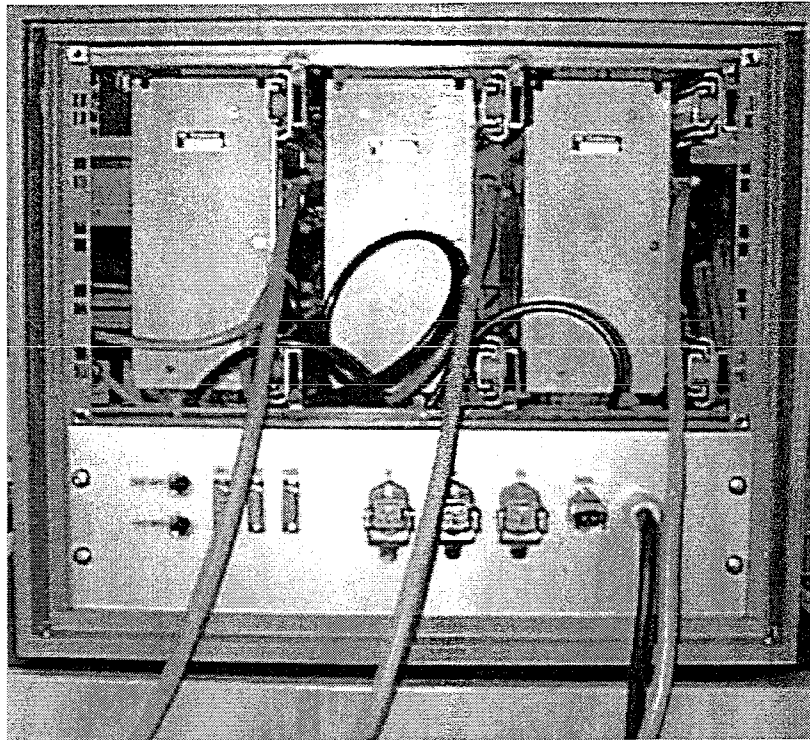
ONTWERP :


TELEFOON :

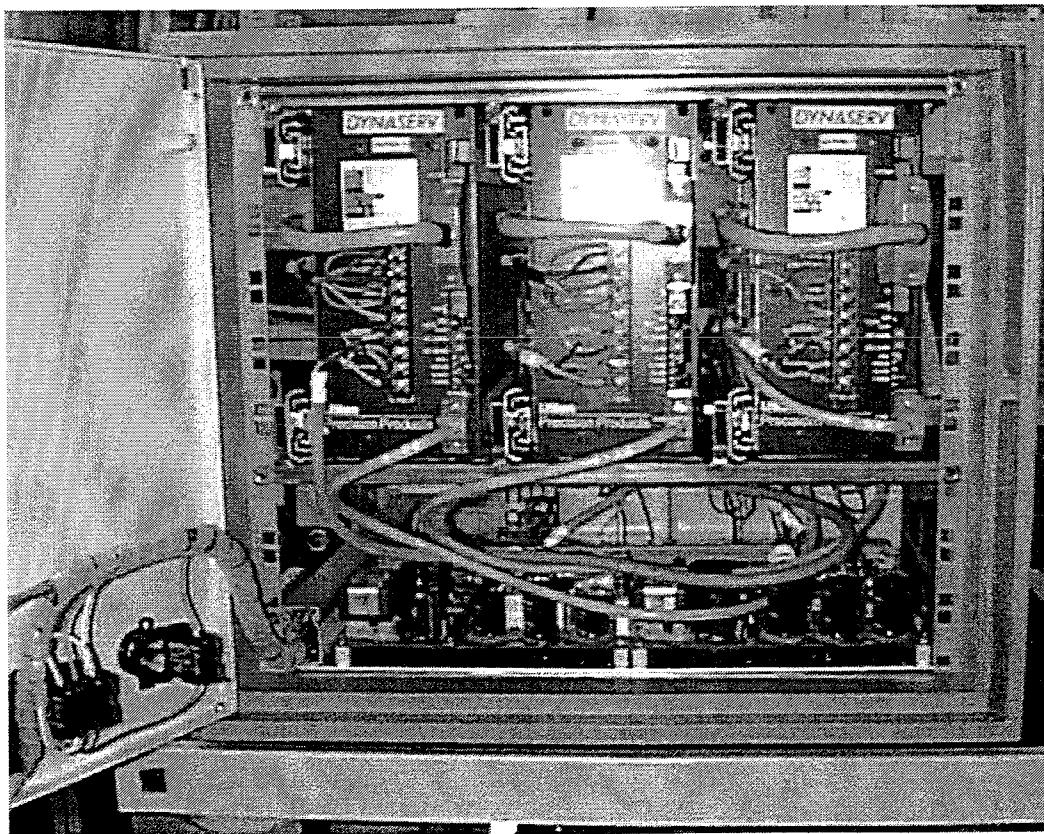
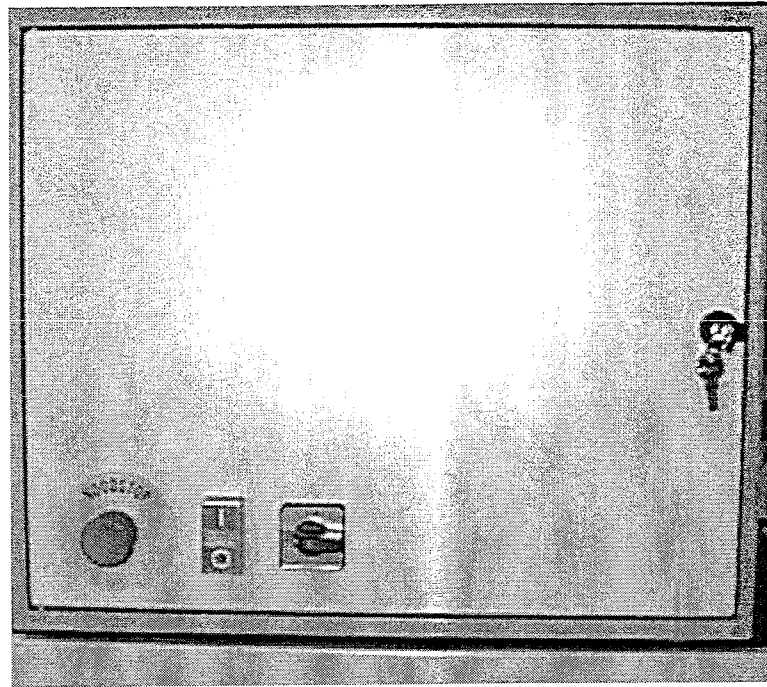
REALISATIE : *B.Viveen*


DOCUMENTATIE : *B.Viveen*

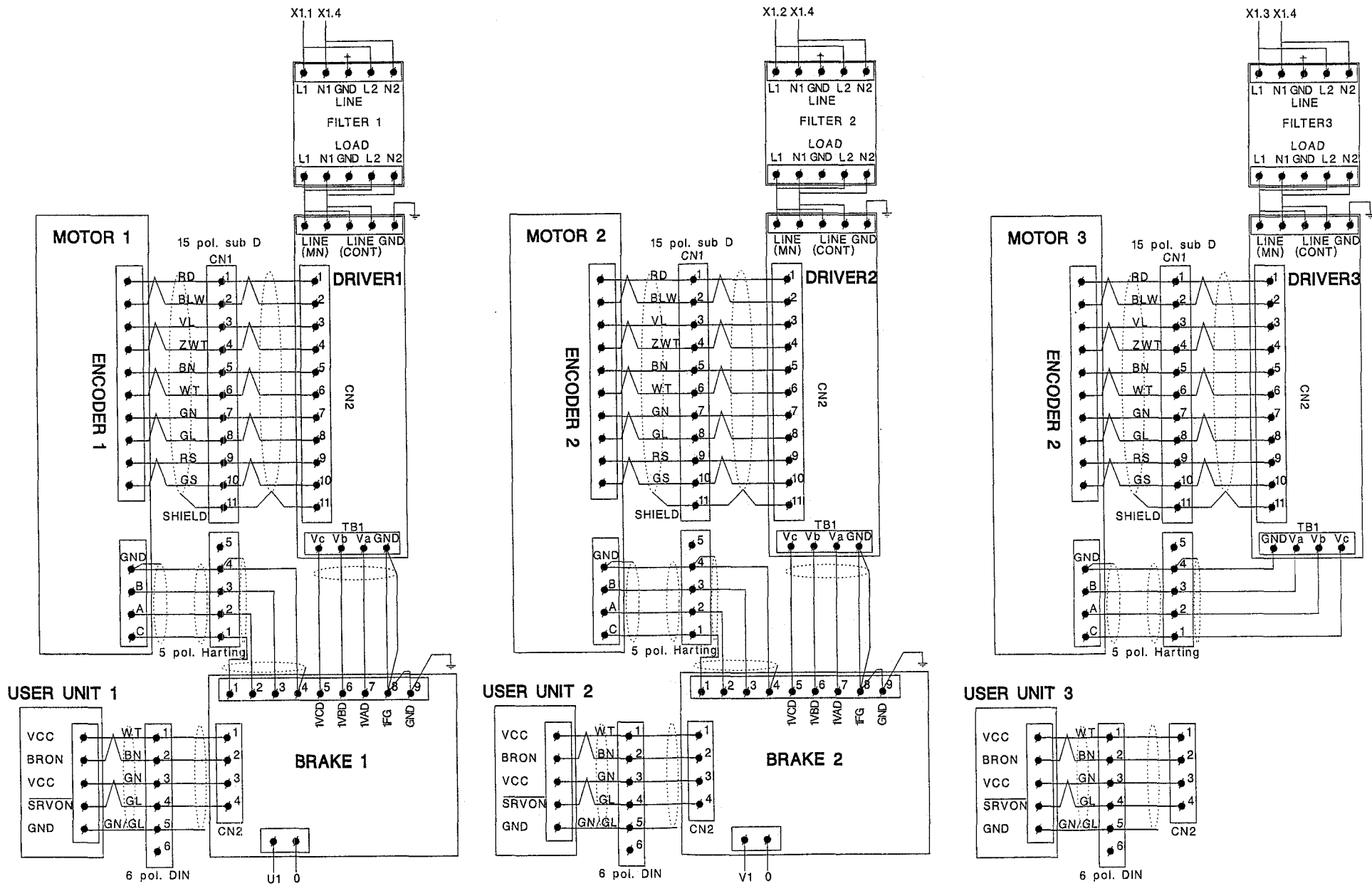
INHOUD :



	Omschrijving	Opmerkingen	Opdracht nr.	
	voorzijde	foto1.ged	schaal 3:1	3506/001/23/23/01
			get. B. Viveen; tel.	Formaat A4
			geo.;	Documentatie nr. 35060012.32
			dat. 23-01-98	

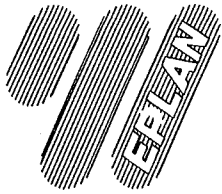


	Omschrijving	Opmerkingen	Opracht nr.	
	voorzijde	foto1.ged	schaal 3:1	3506/001/23/23/01
			get. B. Vliveen tel.	Formaat A4
			geo. dat. 23-01-98	Documentatie nr. 35060012.32



-127-

Omschrijving		OVERZICHT_SCHEMA.GED		Opmerkingen	
SERVO VERSTERKERS					
schast		get.B.Viveen tel. 5012		Format 3506/001/23/23/01	
gec.		dal. 23-01-98		Documentatie nr. 35060012.323	



EPLAN

Hengelder 56
Postbus 246
NL-6900 AE Zevenaar
Tel.: 08360-91790

KLANT :
PROJEKTNAAM : SERVO-SYSTEEM
TEKENINGNUMMER : GTD E/E
OPDRACHTGEVER :

Fabrikant (firma) :
Pad (zonder \EPLAN4\P) : 3506/001/23/2301
Projektnaam :
Fabrikaat :
Type :
Installatieplaats :
Projektleider :
Bijzonderheden :

Vervaardigd op : 27.NOV.1997 Hoogste paginanummer : 4
Laatste wijziging : door : Aantal pagina's : 4

-128-

		tek	Jan v Heerebeek
KS1	28.NOV.1997	datum	23.JAN.1998

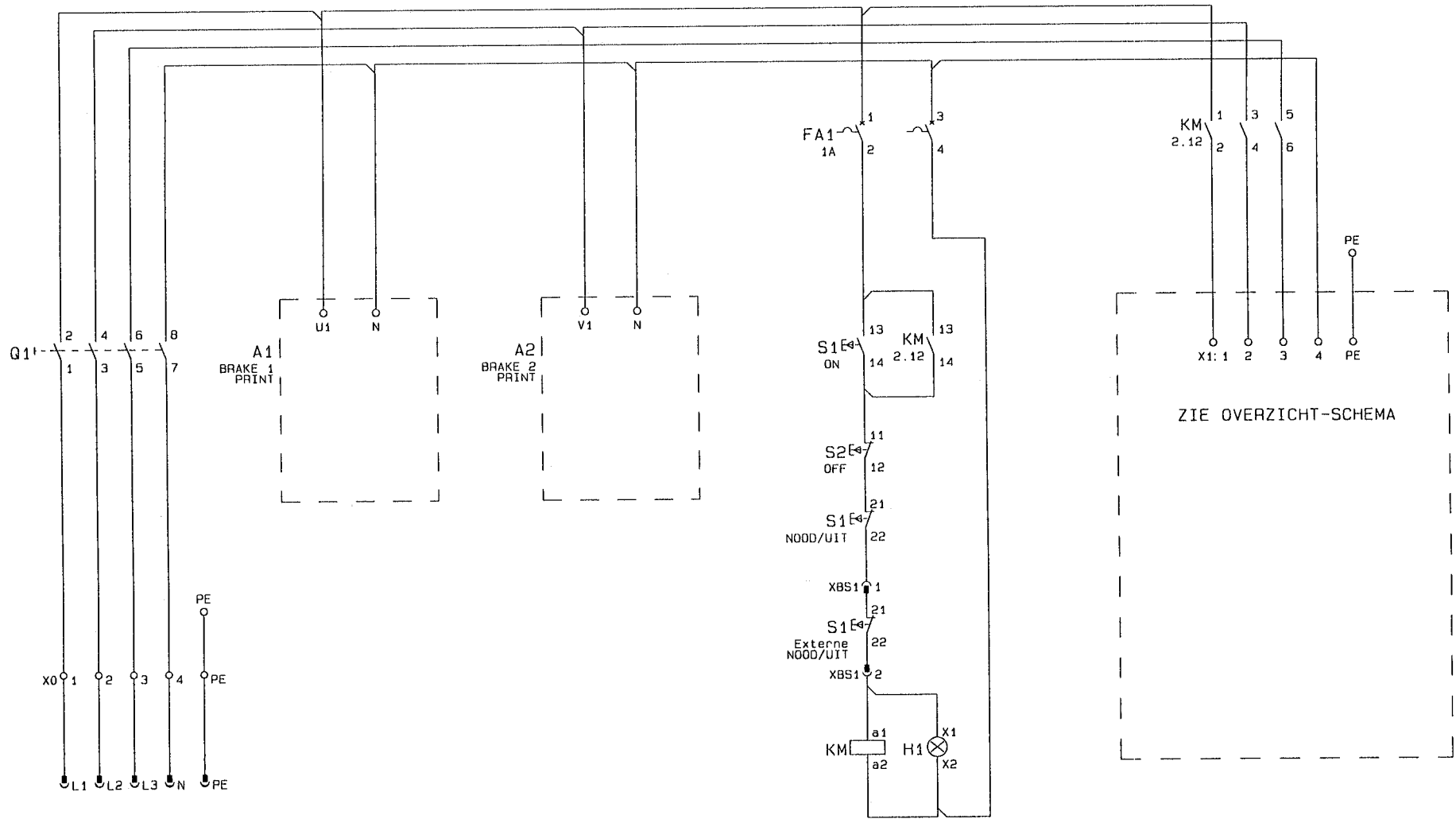
SERVO-SYSTEEM



VOORBLAD

storingen tel 3497/3499 opdracht uren 3506/001/23/2301

-129-



VOEDING
3*F+N+PE

M	V
2.16	
2.17	
2.17	
2.13	

tek	Jan v Heerebeek
datum	23 JAN. 1998
WS1	23 JAN. 1998

SERVO-SYSTEEM



HOOFDSTROOMSCHEMA

storingen	opdracht	uren	3506/001/23/2301	B1. 2
tel 3497/3499				

-130-

BESTELBON TECH.UNIE

VIBO POLIMEX
 engelseweg 127
 tel: 0492 576911
 fax: 0492 576290

FABRIKANT	ARTIKELNUMMER	BESTELNUMMER	OMSCHRIJVING	onderdeelcode	pag/pad	AANTAL	FL/eenh exc1. BTW	KORTING %	KORTING FL	BTW 17,5 %	TOTAAL
KLOCKNER MOEL	T0-2-8293/E	688 820	hoofdsch. 3 polig + nul front 20A	01	2.3	1					
				01	2.4	1					
TELEMECANIQUE	GB2-CD06	645 226	stuurstroom automaat 1A	FA2	2.6	1					
KLOCKNER MOEL	QDDL-11/10	699 546	AAN UIT DRUKKER MET LAMPJE	S1	2.6	1					
KLOCKNER MOEL	BK11	698 647	DRUKKNOP ELEMENT 1MAAK 1VERBREEK	S1	2.6	1					
KLOCKNER MOEL	EF	698 951	LAMPHOUDER BK ELEMENT	S1	2.6	1					
KLOCKNER MOEL	GIL 220K	469 296	LAMPJE 220V	S1	2.6	1					
				S2	2.6	1					
KLOCKNER MOEL	RPV	698 852	nood uit knop ontgrendeling door trekken	S1	2.6	1					
KLOCKNER MOEL	BK11	698 647	DRUKKNOP ELEMENT 1MAAK 1VERBREEK	S1	2.6	1					
KLOCKNER MOEL	FAK-RT/V/KC11/1	669 184	nood uit ext. ontgrendeling door trekken	S1	2.6	1					

PARAAF BESTELLER.....

PARAAF BEDRIJFSBUREAU.....

totaal FL.....

2

4

	tek	27.NOV.1997	
26.NOV.1997	datum	23.JAN.1998	

SERVO-SYSTEEM



storingen tel 3497/3499	opdracht nummer	uren	3506/001/23/2301	81. 3
----------------------------	--------------------	------	------------------	-------

BESTELBON TECH.UNIE

VIBO POLIMEX
 engelseweg 127
 tel: 0492 576911
 fax: 0492 576290

FABRIKANT	ARTIKELNUMMER	BESTELNUMMER	OMSCHRIJVING	onderdeelkode	pag/pad	AANTAL	FL/eenh excl. BTW	KORTING %	KORTING FL	BTW 17,5 %	TOTAAL
KLOCKNER MOEL	DIL 00 AM	697 623	hoofdmagn.schak.3P/5,5KW/spoelsp.220V	KM	2.6	1					
KLOCKNER MOEL	11 DIL M	697 821	hulpcontacten DIL 0,1,2 M 1M1V	KM	2.6	1					
				H1	2.7	1					
				A1	2.13	1					
				M1	2.13	1					
				A2	2.15	1					
				M2	2.15	1					
				A3	2.17	1					
				M3	2.17	1					

-131-

PARAAF BESTELLER.....

PARAAF BEDRIJFSBUREAU.....

totaal FL.....

3

	27.NOV.1997
tek	Jan v Heerebeek
28.NOV.1997	datum 23.JAN.1998

SERVO-SYSTEEM



storingen	opdracht	uren	3506/001/23/2301	Bl. 4
tel 3497/3499	nummer			

Project : 3506/001/23/23/01, 16-01-1997 1/1

Dok.nr : 35060012.323

Bedr.Ond : GTD/EE

Betreft : **Dynaserv Servo-Actuator**

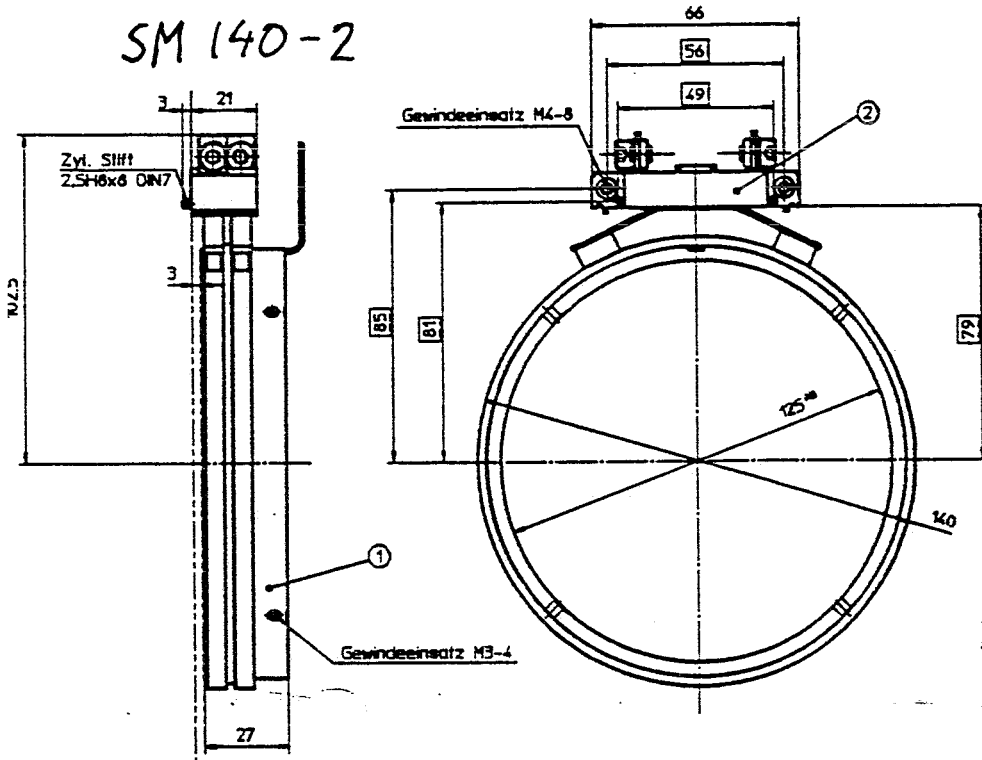


Aansluiting 50 pol. Amphenol connector

1 ROOD	27 ROOD/BLAUW
2 BLAUW	28 GRIJS/ROSE
3 ROSE	29 ZWART
4 GRIJS	30 VIOLET
5 GEEL	31 WIT
6 GROEN	32 BRUIN
7 BRUIN	33 ROSE
8 WIT	34 GRIJS
9 ROSE/GEEL	35 GEEL
10 ROSE/GROEN	36 GROEN
11 GRIJS/GEEL	37 ZWART/ROOD
12 GRIJS/GROEN	38 ZWART/BLAUW
13 ZWART/WIT	39 ROOD/GRIJS
14 ZWART/BRUIN	40 ROOD/ROSE
15 ROOD/WIT	41 ZWART/ROSE
16 ROOD/BRUIN	42 ZWART/GRIJS
17 BLAUW/WIT	43 ZWART/GEEL
18 BLAUW/BRUIN (afscherming)	44 BLAUW/GRIJS
19 ROSE/WIT	45 ZWART/GEEL
20 ROSE/BRUIN	46 ZWART/GROEN
21 GRIJS/WIT	47 ROOD/GEEL
22 GRIJS/BRUIN	48 ROOD/GROEN
23 GEEL/WIT	49 BLAUW/GEEL
24 GEEL/BRUIN	50 BLAUW/GROEN
25 GROEN/WIT	
26 GROEN/BRUIN	

B.6 Power sliprings

SM 140-2



Features:

- Heavy Duty design for industrial use
- Long life time
- 2/4/6/12 rings ¹⁾
- max. 15 Amps. per ring
- custom-made versions available

Technical Data:

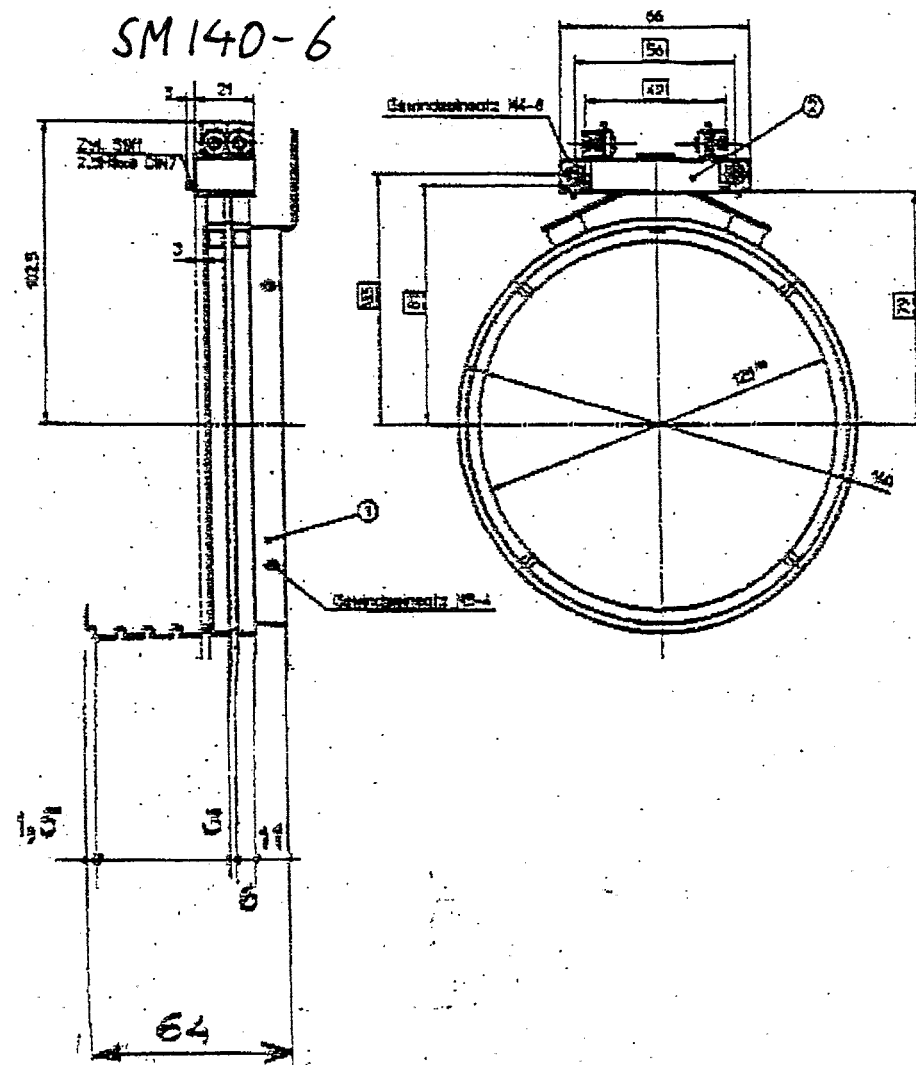
Dimensions:	see drawing
No. of rings:	2/4/6/12 ¹⁾
Max. operating current:	15 A (at 50 V DC)
Operating voltage:	nom. 50 V DC max. 220 V DC* (the max. value is dependant on the actual operating current)
Circuit resistance:	≤ 0,1 ohm
Insulation resistance at 500 V DC:	10 ⁶ M
Dielectric strength:	1000 V _{eff} (60s)
Rotation speed:	max. 250 rpm
Operating temp.:	0 to 65°C ¹⁾
Protection:	rotor: IP 00 terminals: IP 00

° If different data is required please use our specification sheet!

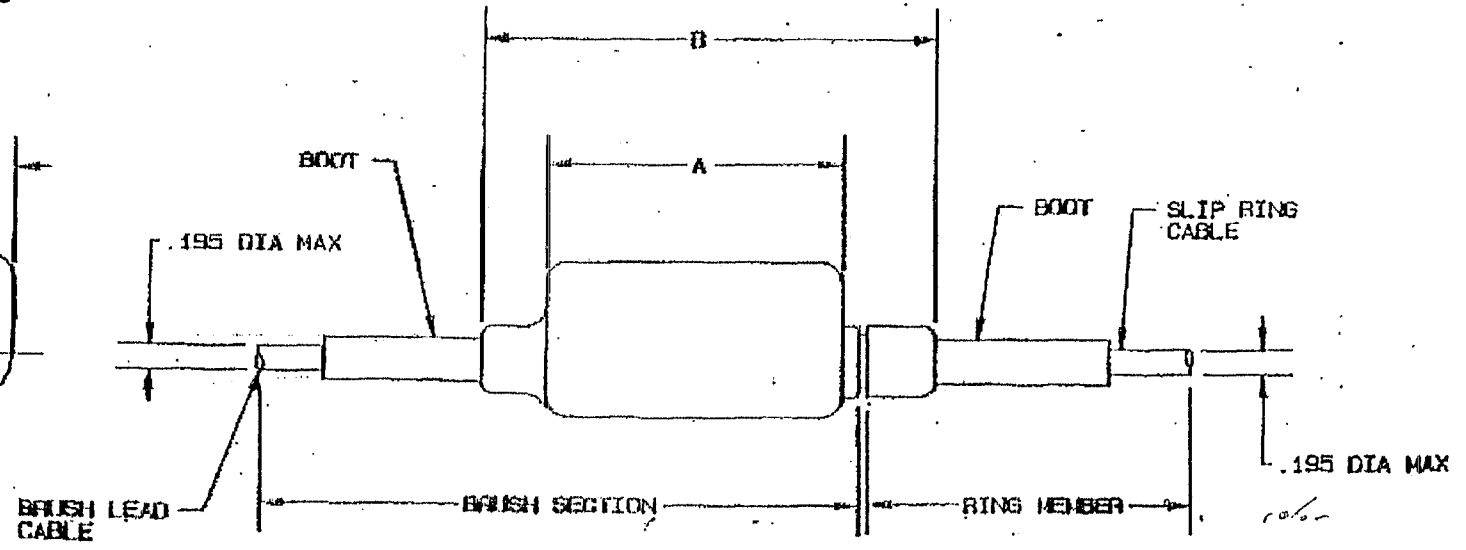
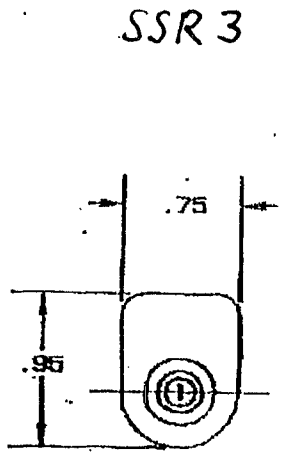
* according to insulation group of "VDE"!

¹⁾ others on request

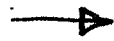
SM 140-6



B.7 Signal sliprings

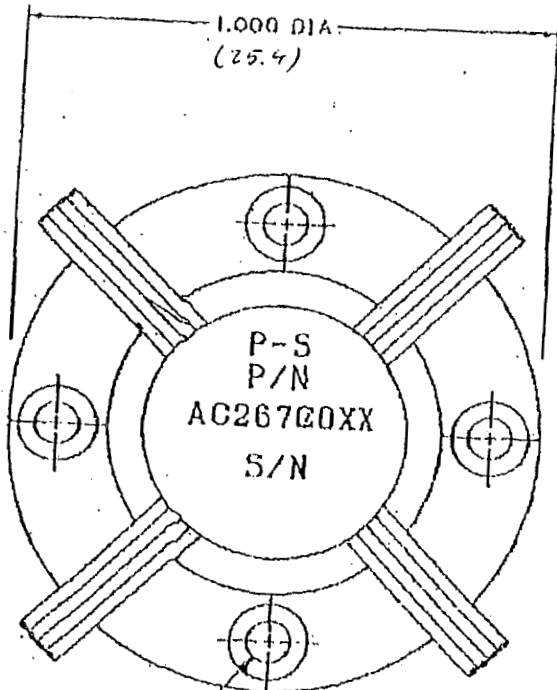


ND. OF RINGS	A	B
4	.85	1.75
12	1.65	2.50

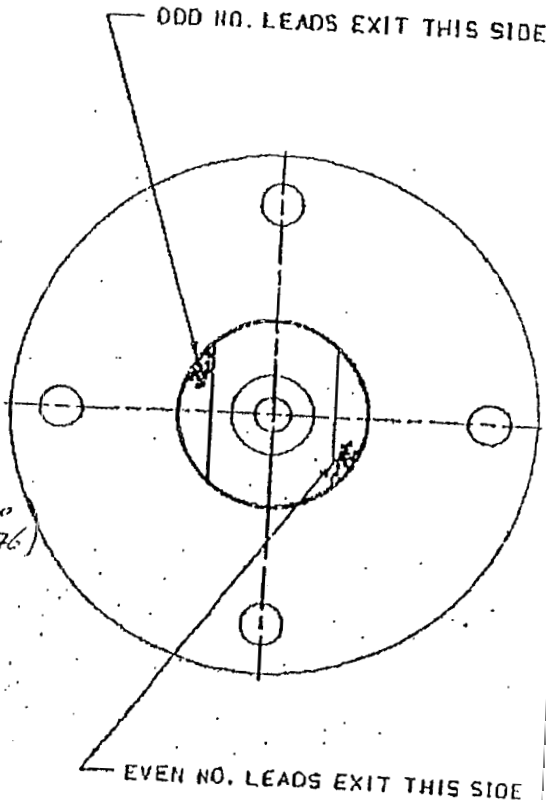
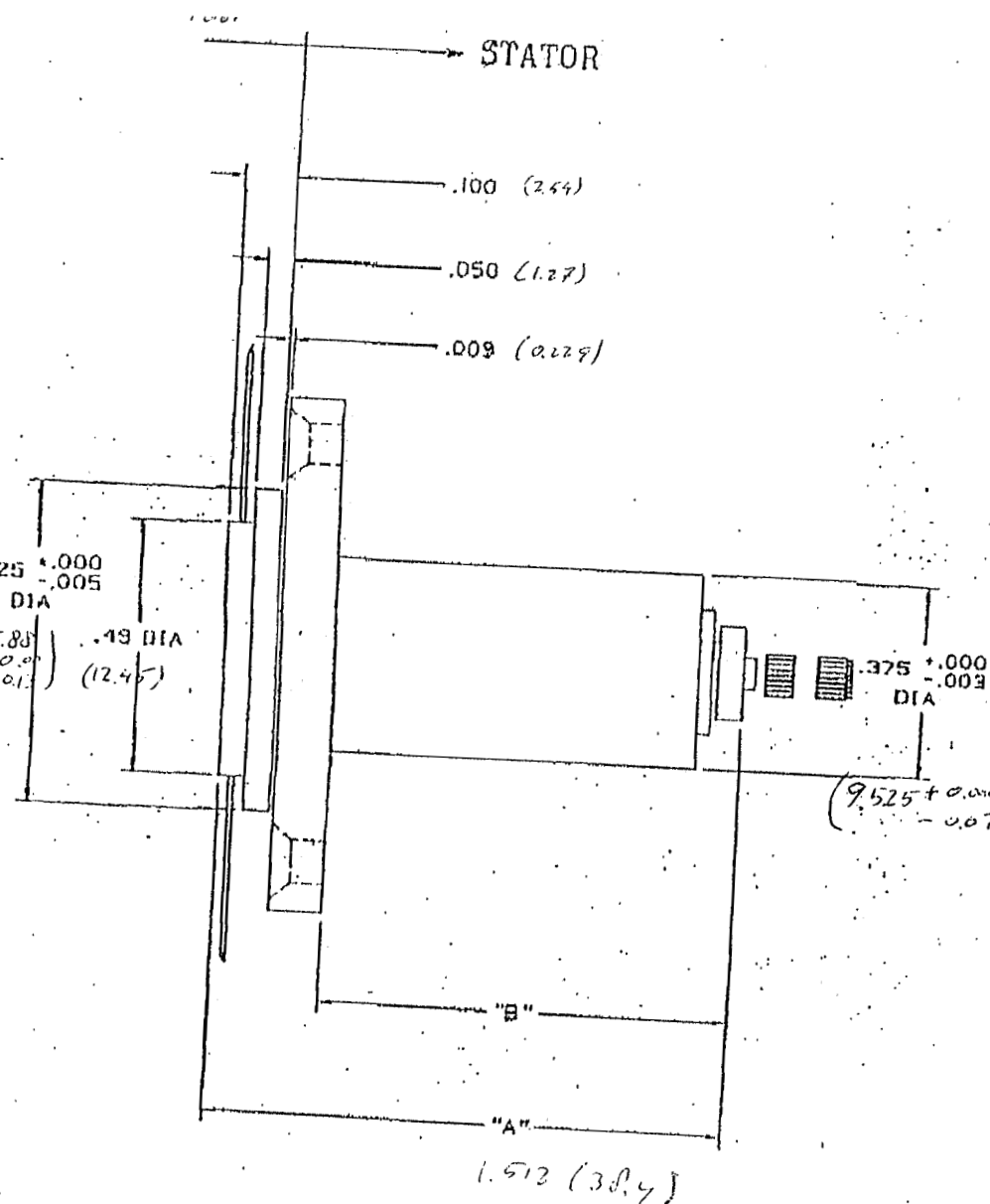


SSR1-2

-136-



.081 DIA THRU, C'SINK 82° TO
 .146 DIA, 4 HOLES TO SP OR A
 .812 DIA B.C.



1.512 (38.4)

B.8 Conceptual design

The RRR-robot conceptual design drawing in this Section was used as a basis for the manufacturing of the various non-standard components by the CTD. Note that some dimensions were slightly altered during the manufacturing.