EuroSystem
EuroStep

HARDWARE REFERENCE GUIDE

Servo/Stepper Positioning System

Issue 2.1

Ref: MN00100-001;esd2_1/MC/1193

Copyright © Optimised Control Ltd 1995.

All rights reserved.

This manual is copyrighted and all rights are reserved. This document may not, in whole or in part, be copied or reproduced in any form without the prior written consent of Optimised Control.

Optimised Control makes no representations or warranties with respect to the contents hereof and specifically disclaims any implied warranties of fitness for any particular purpose. The information in this document is subject to change without notice. Optimised Control assumes no responsibility for any errors that may appear in this document.

Optimised Control Ltd
178 - 180 Hotwell Road
Bristol
BS8 4RP
U.K.

Telephone:	(+44) (117) 987 3100
Fax:	(+44) (117) 987 3101
BBS:	(+44) (117) 987 3102

IBM is a registered trademark of International Business Machines, Inc.

Manual Revision History

Issue	Revision	Date	Reference	Comments
2.0	000	Oct 91	MN00100; hard2/mc/1091	Include sections on keypad and display and revise for clarity and completeness
2.1	001	Nov 93	MN00100-001; ESD2_1/MC/11 93	Revised for clarity and completeness Remove servo setup section. Now covered in Getting Started Guide. Include details on options board and the 3rd axis servo board

1. Introduction

EuroSystem and EuroStep are both 3 axis intelligent motor controllers that provide a comprehensive range of features for controlling servo and step motors. EuroSystem allows servo and step motors to be mixed in the same multi-axis system with a highly flexible programming language, MINT. EuroStep provides support for up to three open loop step motors using the same programming language, MINT.

Features:

- *Stand-alone operation or controlled by host over RS232/485 serial link, up to 16 cards on multidrop*
- *Easy to use BASIC-like motion control language, MINT*
- *28k bytes non-volatile program/data memory*
- *On board program editor*
- *Circular and linear interpolation*
- *Microstepping control up to 200kHz*
- *Step, Direction and Boost outputs compatible with industry standard drives*
- *8 uncommitted digital inputs and outputs for machine control*
- *3 Limit switches, 3 Home switches and Error input*
- *Two 10 bit analogue inputs for interface to joy-stick or sensors*
- *Pulse follower input*
- *+5V, ±12V power requirement*
- *Expansion port for option boards such as extended I/O, keypad and display*

For servo systems, you are advised to read the section 'Servo System Set-up' which explains, in detail, how to test for the correct wiring of amplifiers to EuroSystem and the setting of system gains.

1.1. MINT

MINT, standing for Motion INTerpreter, is a structured form of BASIC, custom designed for EuroSystem/EuroStep. MINT has been written to allow you to very quickly get 'up and running' with simple motion programs, whilst providing a wide range of more powerful commands for complex applications.

In addition to the usual BASIC commands, plain English keywords are used to control motion and input/output. These keywords allow control of motor position, speed, torque including interpolation and synchronisataion of multiple axes. You also have full software control over the basic motor control parameters. MINT supports both servo and stepper systems, allowing different motors to be mixed in the same multi-axis system.

2. EuroSystem/EuroStep I/O

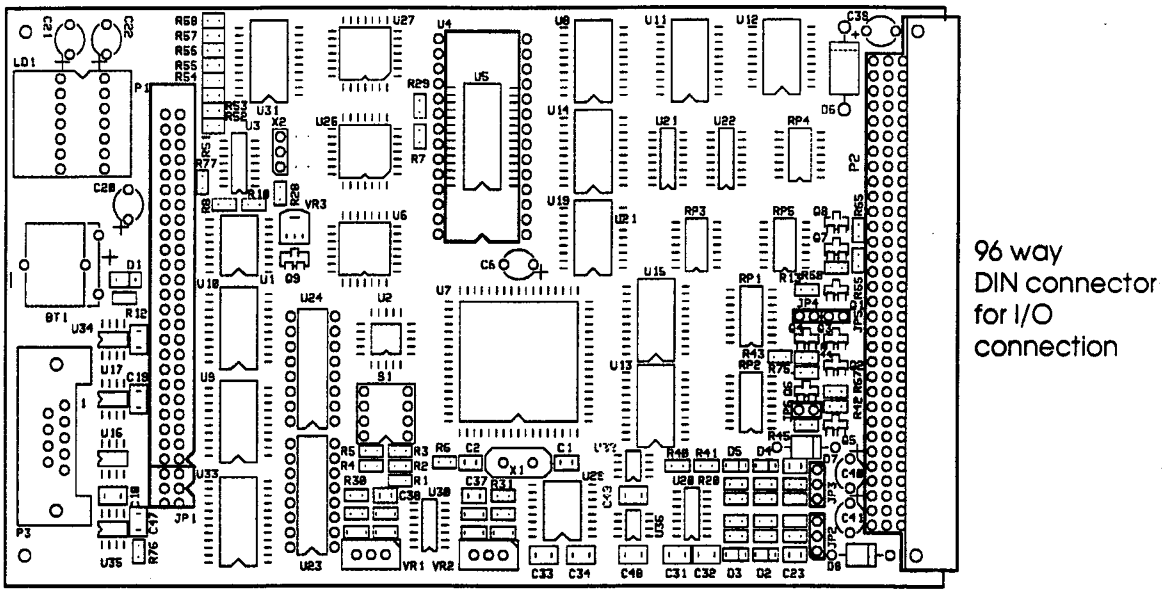
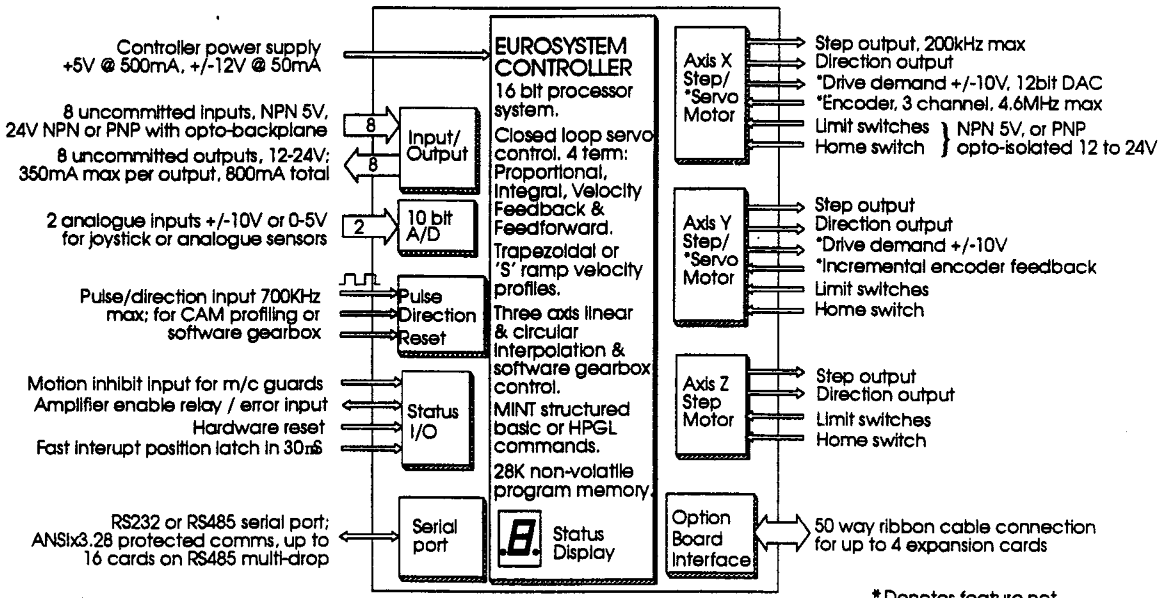


Figure 1.1:EuroSystem/EuroStep Controller layout.



- Operating temperature: 0 - 50 ° C (extended temperature spec on request).
- Dimensions: 3U standard eurocard 100 by 160 mm.
- Connector: DIN 41612, 96-way, plugs into opto-isolated or standard back-plane.
- Weight: 150gms.

*Denotes feature not applicable to EuroStep controller

Figure 1.2: EuroSystem/EuroStep Controller Block Diagram

2.1. 96-way DIN Connector Pinout

I/O associated with EuroSystem and EuroStep are illustrated in Figures 1.2 and Figures 1.3. They include limit switch inputs, home switch inputs, amplifier demand outputs and error output. All connections are brought out at one end of the board on a DIN 41612 96-way connector. Normally you would use a backplane (isolated or non-isolated) for your connections, but for OEM applications the pin-out is explained in more detail below:

c			b			a		
Vcc	1	❑	Vcc	1	❑	Vcc	1	❑
Vcc	2	❑	Vcc	2	❑	Vcc	2	❑
GND	3	❑	GND	3	❑	GND	3	❑
USR-OUT-6	4	❑	USR-OUT-7	4	❑	USR-OUT-COM	4	❑
USR-OUT-3	5	❑	USR-OUT-4	5	❑	USR-OUT-5	5	❑
USR-OUT-0	6	❑	USR-OUT-1	6	❑	USR-OUT-2	6	❑
CHB-1	7	❑	CHA-0	7	❑	CHB-0	7	❑
IDX-1	8	❑	IDX-0	8	❑	CHA-1	8	❑
!CHA-1	9	❑	!IDX-0	9	❑	!IDX-1	9	❑
!CHB-0	10	❑	!CHA-0	10	❑	!CHB-1	10	❑
ERROR-IN	11	❑	ERROR-OUT	11	❑	GND	11	❑
RESET-IN	12	❑	GND	12	❑	GND	12	❑
GND	13	❑	BOOST-1	13	❑	BOOST-0	13	❑
PULSE-2	14	❑	PULSE-1	14	❑	PULSE-0	14	❑
DIR-2	15	❑	DIR-1	15	❑	DIR-0	15	❑
BOOST-2	16	❑	GND	16	❑	DSR	16	❑
GND	17	❑	GND	17	❑	DTS	17	❑
USR-IN-4	18	❑	FAST-INT	18	❑	USR-IN-2	18	❑
USR-IN-3	19	❑	USR-IN-5	19	❑	USR-IN-7	19	❑
USR-IN-6	20	❑	USR-IN-1	20	❑	RXD	20	❑
USR-IN-0	21	❑	RTS	21	❑	TXD	21	❑
OPT-4	22	❑	OPT-2	22	❑	CTS	22	❑
HOME-2	23	❑	OPT-3	23	❑	OPT1	23	❑
PULSE-IN	24	❑	HOME-1	24	❑	LIMIT-2	24	❑
DIR-IN	25	❑	LIMIT-1	25	❑	STOP	25	❑
HOME-0	26	❑	RESET-CNTR	26	❑	LIMIT-0	26	❑
DEMAND-0	27	❑	DEMAND-1	27	❑	-ANALOGUE-2	27	❑
+ANALOGUE-2	28	❑	+ANALOGUE-1	28	❑	-ANALOGUE-1	28	❑
+12V	29	❑	+12V	29	❑	+12V	29	❑
AGND	30	❑	AGND	30	❑	AGND	30	❑
-12V	31	❑	-12V	31	❑	-12V	31	❑
SCRN	32	❑	SCRN	32	❑	SCRN	32	❑

96-way connector pin-out

Signal	Comments	Pin No.
Drive Demands		
DEMAND-0	Demand signal out, +/-10V, axis 0	c27
DEMAND-1	Demand signal out, +/-10V, axis 1	b27
Encoders		
CHA-0	Encoder channel A, true, axis 0	b7
!CHA-0	Encoder channel A, compliment, axis 0	b10
CHB-0	Encoder channel B, true, axis 0	a7
!CHB-0	Encoder channel B, compliment, axis 0	c10
IDX-0	Encoder INDEX, true, axis 0	b8
!IDX-0	Encoder INDEX, compliment, axis 0	b9
CHA-1	Encoder channel A, true, axis 1	a8
!CHA-1	Encoder channel A, compliment, axis 1	c9
CHB-1	Encoder channel B, true, axis 1	c7
!CHB-1	Encoder channel B, compliment, axis 1	a10
IDX-1	Encoder INDEX, true, axis 1	c8
!IDX-1	Encoder INDEX, compliment, axis 1	a9
Home/Limits		
LIMIT-0	Limit input, axis 0	a26
HOME-0	Home input, axis 0	c26
LIMIT-1	Limit input, axis 1	b25
HOME-1	Home input, axis 1	b24
LIMIT-2	Limit input, axis 2	a24
HOME-2	Home input, axis 2	c23
Miscellaneous		
PULSE-IN	Pulse input, pulse follower	c24
DIR-IN	Direction input, pulse follower	c25
RESET-CNTR	Timer 2 reset input	b26
RST-IN	System reset, input	c12
STOP	Stop execution, input	a25
ERROR-OUT	Motion/System error output	b11
ERROR-IN	Motion/System error input	c11
FAST-INT	User hardware INTERRUPT	b18
Digital Inputs		
USR-IN-0	User input, bit 0	c21
USR-IN-1	User input, bit 1	b20
USR-IN-2	User input, bit 2	a18
USR-IN-3	User input, bit 3	c19
USR-IN-4	User input, bit 4	c18
USR-IN-5	User input, bit 5	b19
USR-IN-6	User input, bit 6	c20
USR-IN-7	User input, bit 7	a19
Digital Outputs		
USR-OUT-0	User output, bit 0	c6
USR-OUT-1	User output, bit 1	b6
USR-OUT-2	User output, bit 2	a6
USR-OUT-3	User output, bit 3	c5
USR-OUT-4	User output, bit 4	b5
USR-OUT-5	User output, bit 5	a5
USR-OUT-6	User output, bit 6	c4
USR-OUT-7	User output, bit 7	b4
USR-OUT-COM	Common diode clamp	a4

Analogue Inputs		
+ANALOGUE-1	Analogue input 1, non-inverting	b28
-ANALOGUE-1	Analogue input 1, inverting	a28
+ANALOGUE-2	Analogue input 2, non-inverting	c20
-ANALOGUE-2	Analogue input 2, inverting	a27
Stepper		
PULSE-0	Pulse output stepper axis 0	a14
DIR-0	Direction output stepper axis 0	a15
BOOST-0	Boost or full/half step output stepper axis 0	a13
PULSE-1	Pulse output stepper axis 1	b14
DIR-1	Direction output stepper axis 1	b15
BOOST-1	Boost or full/half step output stepper axis 1	b13
PULSE-2	Pulse output stepper axis 2	c14
DIR-2	Direction output stepper axis 2	c15
BOOST-2	Boost or full/half step output stepper axis 2	c16
Serial Port		
TXD	Transmitted data (TxD true)	a21
RXD	Received data (RxD true)	a20
RTS	Request to send (TxD compliment)	b21
CTS	Clear to send (RxD compliment)	a22
DSR	Connected to DTR	a16
DTR	Connected to DSR	a17
Option Board		
OPT1	Option board I/O	a23
OPT2	Option board I/O	b22
OPT3	Option board I/O	b23
OPT4	Option board I/O	c22
Power Supply and References		
-12V	-12V @ 100mA	c31,b31,a31
+12V	+12V @ 100mA	c29,b29,a29
5V	+ 5V @ 500mA	c1,c2,b1,b2, a1,a2
GND	0V Digital ground	
AGND	0V Analogue ground	c30,b30,a30
SCRN	0V Power ground / Cable screen	c32,b32,a32

Inputs are always in their active (high) state unless they are pulled low by external circuitry, providing fail safe operation. Before the system will operate, you must connect the limit switches and the stop switch either to their respective switches or to ground. Failure to do so will result in a limit error on the controller and a stop condition. This is indicated by the LED status display showing either an 'L' or an 'S'.

The following sections explain each of the I/O types in detail. A pin-out table lists all the relevant pins on the DIN connector and the backplane for that I/O function. The pin number on the DIN connector is given in brackets. The backplane detail shows the connector block followed by the pin name.

2.2. Switch Types

The inputs are active high and must be tied to ground through normally closed switches. The system will fail to operate with normally open switches.

2.3. Limit Inputs

Pinout:

Din Connector	Backplane	Description
LIMIT-0 (a26)	HOME/LIMIT: L0	Limit switch axis 0
LIMIT-1 (b25)	HOME/LIMIT: L1	Limit switch axis 1
LIMIT-2 (a24)	HOME/LIMIT: L2	Limit switch axis 2
GND	HOME/LIMIT: Gnd	Digital ground

In a typical application, the limit switch inputs would be connected to normally closed micro switches on the axis. Hitting the limit switch will cause the switch to become open circuit, the respective input is pulled high internally, resulting in a limit error and the axis coming to an immediate stop. This is indicated by an 'L' on the LED status display and the ERROR keyword which will return 3. See the MINT Programming Guide to more details on the ERROR keyword and error handling within MINT.

Because there is only one limit input per axis, if two end-of-travel limit switches are fitted then they must both be connected in series as shown below.

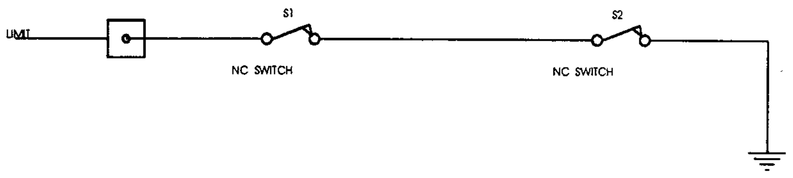


Figure 2.1 Dual Limit input switch connection.

A possible problem with this arrangement is that is it not possible to determine in software which end-of-travel switch has been hit by simply reading the limit input. This can be overcome by introducing a double pole limit switch at one end of the axis and connecting this to the home input or a spare digital input. This means that when a limit switch is hit the program can determine whether this is the forward or reverse limit by reading the status of the home switch or relevant digital input.

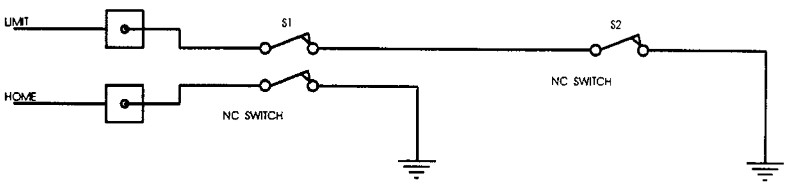


Figure 2.2 Dual Limit and Home/Input connection.

In many applications it is necessary to have separate limit and home switches. Since the controller has only limit input per axis it is best to arrange the active areas of the switches as shown below.

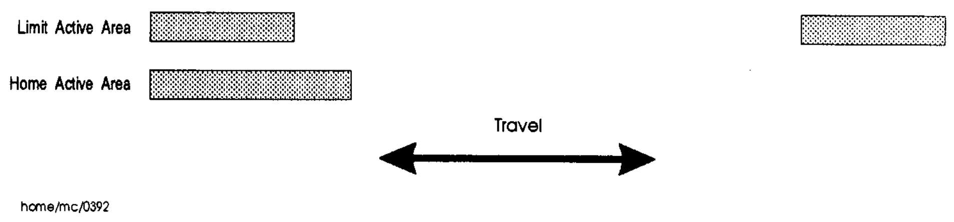


Figure 2.3 Limit/Home switch active areas.

Hitting a limit switch causes an error which is handled in software by the ONERROR subroutine (see the MINT Programming Guide). By reading both the state of the limit and home inputs, the left or right limit switch can be determined as shown in the table..

HOME	LIMIT	
on	on	Left limit hit
off	on	Right limit hit

If the home position is midway between the two limits, the active area of the home should be made to extend to either the right or the left to the limit active area.

In an application where not all 3 axes are used, the redundant axes must have their respective limit switches grounded for normal operation. Alternatively the MINT keyword DISLIMIT can be used to disable the detection of limit switches.

The state of the limit switches can be read using the MINT LIMIT keyword. A value of 1 (logical one) will be returned if the limit switch is high or floating and 0 (logical zero) if grounded.

Figure 2.4 shows the input buffer circuit and normal connections for all motion inputs.

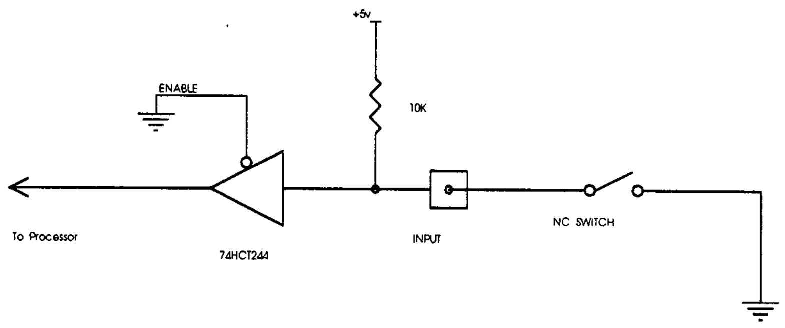


Figure 2.4 Input buffer circuit for Limit, Home, Stop, Error in, Pulse, Dir and Reset counter.

2.4. Home Inputs

Pinout:

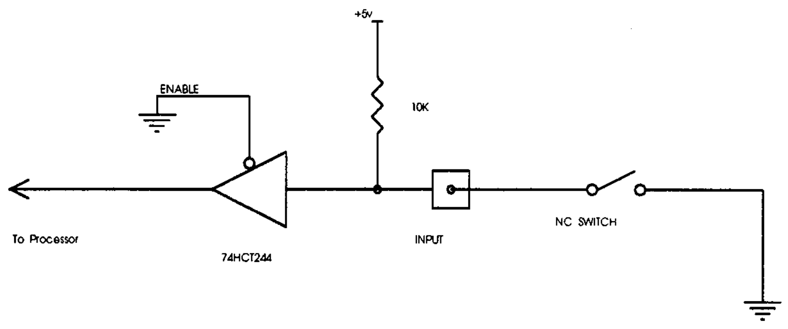
Din Connector	Backplane	Description
HOME-0 (c26)	HOME/LIMIT: H0	Home switch axis 0
HOME-1 (b24)	HOME/LIMIT: H1	Home switch axis 1
HOME-2 (c23)	HOME/LIMIT: H2	Home switch axis 2
GND	HOME/LIMIT: Gnd	Digital Ground

Three home inputs are provided, one for each axis. Like the limit switches these are active high and require normally closed switches for normal operation. Unlike the limit switches however, they do not have to be grounded if they are not used.

If the limit switch is used as a datum point, both the limit input and the home input must be connected together. Noisy environments can cause glitches on the limit switches which may result in a limit error during homing. A 100nF ceramic capacitor connected between the limit input and ground should eliminate this problem.

The state of the home switches can be read using the MINT HOME keyword. A value of 1 will be returned in the home switch is high or floating and 0 if grounded.

The input buffer circuit is shown below:



2.5. General I/O

EuroSystem/EuroStep provide 8 uncommitted digital inputs and 8 uncommitted digital outputs. These are accessed through the MINT keywords IN and OUT.

2.5.1. Digital Inputs

Pinout:

Din Connector	Backplane	Description
USR-IN-0 (c21)	IN: 0	User input bit 0
USR-IN-1 (b20)	IN: 1	User input bit 1
USR-IN-2 (a18)	IN: 2	User input bit 2
USR-IN-3 (c19)	IN: 3	User input bit 3
USR-IN-4 (c18)	IN: 4	User input bit 4
USR-IN-5 (b19)	IN: 5	User input bit 5
USR-IN-6 (c20)	IN: 6	User input bit 6
USR-IN-7 (a19)	IN: 7	User input bit 7
GND	IN: Gnd	Digital ground

The 8 inputs are active high with schmidt trigger buffers, the controller will read a logical 1 if the input is +5V or unconnected and logical 0 if the input is grounded. The inputs can take a maximum voltage input of 5V using the standard backplane. Using the opto-isolated backplane, voltages up to 24V may be connected to the inputs.

The input buffer circuit for the user inputs is slightly different from that for motion inputs. The circuit is shown in figure 2.5, along with some input configurations.

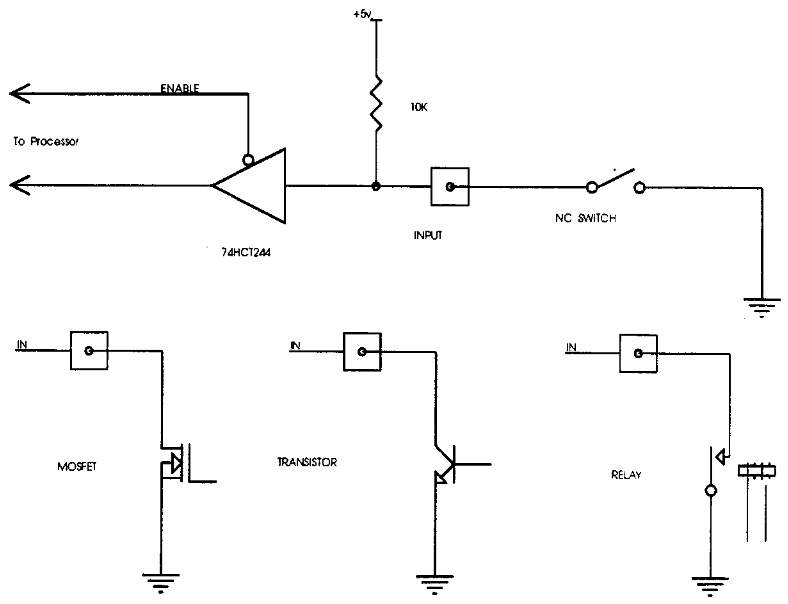


Figure 2.5 User inputs buffer circuit and some input connections.

2.5.2. Digital Outputs

Pinout:

Din Connector	Backplane	Description
USR-OUT-0 (c6)	OUT: 0	User output bit 0
USR-OUT-1 (b6)	OUT: 1	User output bit 1
USR-OUT-2 (a6)	OUT: 2	User output bit 2
USR-OUT-3 (c5)	OUT: 3	User output bit 3
USR-OUT-4 (b5)	OUT: 4	User output bit 4
USR-OUT-5 (a5)	OUT: 5	User output bit 5
USR-OUT-6 (c4)	OUT: 6	User output bit 6
USR-OUT-7 (b4)	OUT: 7	User output bit 7
USR-OUT-COM (a4)	OUT:CM	Common diode clamp
GND	OUT: Gnd	Digital ground

The uncommitted digital outputs are driven by an octal darlington array (ULN2803 device). Each output is capable of sinking 50mA when ON, and can withstand 50V when OFF. The outputs have a Common diode clamp connection accessible from the backplane through USR-OUT-COM. This should be connected to the external supply from which you are sourcing current. The circuit is shown in figure 2.6.

For switching high currents it is recommended that the outputs be used to drive solid state relays.

Example:

```
OUT = 15
```

this example in MINT will turn on outputs 0 to 3 (i.e. outputs are grounded) and turn outputs 4 to 7 off. See the MINT Programming Guide for more details on the OUT keyword.

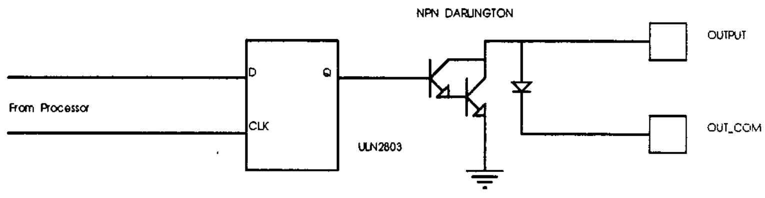


Figure 2.6 Output driver circuit.

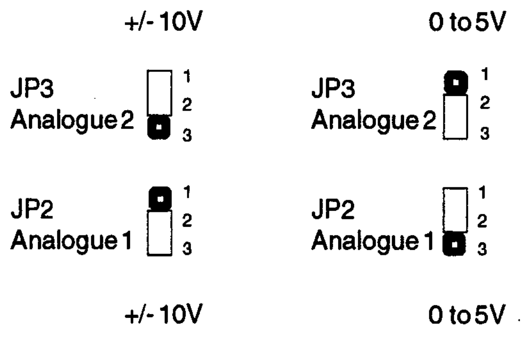
2.5.3. Analogue Inputs

Pinout:

Din Connector	Backplane	Description
+ANALOGUE-1 (b28)	ANALOGUE: 1+	Analogue input 1, non-inverting
-ANALOGUE-1 (a28)	ANALOGUE: 1-	Analogue input 1, inverting
+ANALOGUE-2 (c28)	ANALOGUE: 2+	Analogue input 2, non-inverting
-ANALOGUE-2 (a27)	ANALOGUE: 2-	Analogue input 2, inverting
AGND (c30,b30,a30)	ANALOGUE: AG	Analogue Ground
SCRN (c32,b32,a32)	ANALOGUE: SC	Power ground/Screen

Two independent analogue inputs are provided, each with 10 bit resolution in the range of ±10V or 0-5V. These may be used for analogue sensor input or to provide a low cost joy-stick interface.

Each input is buffered and has a low pass filter to reject noise, (-3dB @ 2Khz). For ±10V operation, the inputs are differential which helps reduce the problems associated with differing ground potentials. This may be by-passed using jumpers JP2 and JP3 to allow 0 to 5V single ended operation, see figure 1.1 for location of jumpers.



Jumper Connection	Voltage Range
1 and 2	±10V
2 and 3	0 to 5V

On no account must the input voltage exceed the maximum rating shown above.

In the 5V configuration the inputs are single ended inputs through the positive input and are referenced to the controller analogue ground. In ±10V configuration the inputs are differential and not referenced to the controller ground. Normally the negative input is connected to the analogue ground of the external equipment. It is important that this connection is made to external <u>analogue</u> ground, and not to external <u>system</u> ground. Connection to the external system ground may result in erroneous input readings caused by the large return currents associated with motor control.

Each analogue input signal should be connected to EuroSystem using a screened twisted pair cable, and the cable screen should be connected to the SCRN (SC) input on the EuroSystem backplane. No other connection should be made to the cable screen, ie. connect the screen at one end only.

The analogue inputs can be read in MINT as 10 bit values using the keywords ANALOGUE1 (A1) and ANALOGUE2 (A2).

2.5.4. Pulse and Direction

Pinout:

Din Connector	**Backplane**	**Description**
PULSE-IN (c24)	MISC: PI	Pulse follower input
DIR-IN (c25)	MISC: DI	Pulse follower direction
RESET-CNTR (b26)	MISC: RC	Reset counter
GND	MISC: G	Digital ground

EuroSystem/EuroStep provide a pulse follower input which consists of 3 signals:

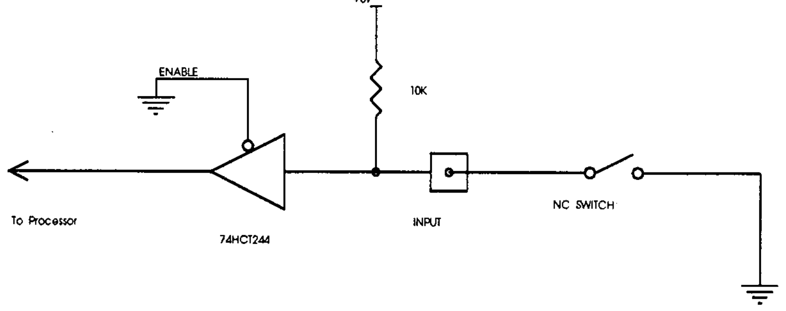
- PULSE-IN: Pulse train input
- DIR-IN: Direction of pulse train
- RESET-CNTR: Reset pulse counter

The pulse and direction inputs (TTL level or open collector signals) go directly into an up-down counter. This counter changes on BOTH the rising and falling edges of the incoming pulse train. RESET-CNTR will reset the counter to 0 on a rising edge and should be tied to ground if not used. The counter value can be read in MINT using the TIMER keyword. This would commonly be used where one channel of an encoder provides the pulse train, and the index pulse resets the counter every revolution. The direction input, DIR-IN, determines whether the counter increments or decrements on each edge. If the direction signal is left unconnected or taken high (5V) then the counter will increment, if it is taken low (0V) then the counter will decrement.

The pulse input has a very fast response time, maximum input frequency 700kHz, and will capture spurious voltage spikes if care is not taken over cable connections. An individually screened cable should be used to connect to this signal, and the neighbouring Gnd connection may be used to earth the respective screen.

By using an external conversion circuit, the counter input can be used to accept a quadrature encoder signal.

The buffer circuit for these inputs is shown below:



2.6. Stepper Outputs

Pinout:

Din Connector	Backplane	Description
PULSE-0 (a14)	STEP: P0	Pulse output stepper axis 0
DIR-0 (a15)	STEP: D0	Direction output stepper axis 0
BOOST-0 (a13)	STEP: B0	Boost output stepper axis 0
PULSE-1 (b14)	STEP: P1	Pulse output stepper axis 1
DIR-1 (b15)	STEP: D1	Direction output stepper axis 1
BOOST-1 (b13)	STEP: B1	Boost output stepper axis 1
PULSE-2 (c14)	STEP: P2	Pulse output stepper axis 2
DIR-2 (c15)	STEP: D2	Direction output stepper axis 2
BOOST-2 (c16)	STEP: B2	Boost output stepper axis 2
GND	STEP: Gnd	Digital ground

The stepper signals are also brought out onto IDC connectors on the backplane.

EuroSystem and EuroStep provide PULSE, DIRECTION and BOOST signals for 3 independent stepper motor drives or micro stepper drives with a maximum output frequency of 200kHz.

All the stepper outputs are open collector and are driven using a darlington array (ULN2803 device). The BOOST output for the 3rd axis is driven using an npn transistor (BC337 device). Each output is capable of sinking 100mA continuously when ON, and can withstand 50V when OFF.

The boost outputs can be turned on and off using the MINT keywords, BOOSTON and BOOSTOFF respectively. This output can also be used for FULL/HALF STEP control or other drive control inputs.

The drive circuits for these outputs are shown in Figure 2.7.

The pulse outputs from the controller have fast fall times and as such can be a source of interference to neighbouring signals. You are strongly recommended to use an individually screened cable for the pulse signal and to run other signals separately. The screen for this signal should be connected to a good ground point, e.g. the ground stud on the backplane and should only be connected at one end.

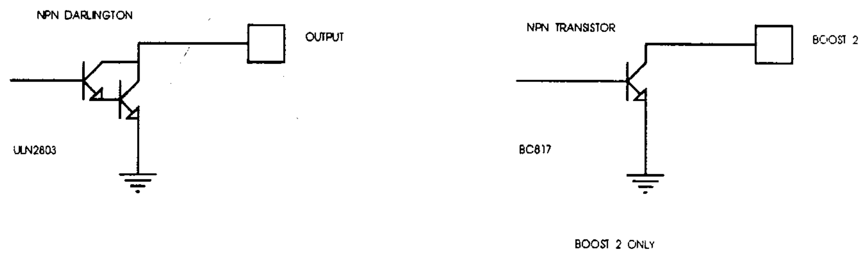


Figure 2.7 Stepper output driver circuits.

2.7. Servo Outputs

Not applicable to EuroStep

Pinout:

Din Connector	**Backplane**	**Description**
DEMAND-0 (c27)	ANALOGUE: D0	Demand signal out ±10V
DEMAND-1 (b27)	ANALOGUE: D1	Demand signal out ±10V
AGND (c30,b30,a30)	ANALOGUE: AG	Analogue Ground
SCRN (c32,b32,a32)	ANALOGUE: SC	Power Ground/Screen

EuroSystem provides two ±10V (±5%) analogue outputs for motor demand, one for each servo axis. A 12 bit DAC is used which gives a resolution of 4.9mV. The signals are brought out on screw terminals on the backplane and are repeated on a 10 way IDC connector labelled 'DRIVE DEMANDS' for connection to Optimised Control amplifiers.

During power up, both the analogue outputs are disabled so that no demand is given to the drives before the controller has control over the system. The output voltage will fall to less than 100mV when the outputs are disabled.

After connecting motor, encoder and the required inputs you must ensure that the sense of your encoder is correct with respect to the motor armature connections. This is covered in detail in the Getting Started Guide.

2.8. Power

Pinout:

Din Connector	**Backplane**	**Description**
Vcc (c1,c2,b1,b2,a1,a2)	POWER: +5	+5V @ 500mA
+12V (c29,b29,a29)	POWER: +12	+12V @ 50mA
-12V (c31,b31,a31)	POWER: -12	-12V @ 50mA
GND	POWER: G	Digital Ground

The power connections are also brought out through a 10 way IDC connector, 'power' on the backplane. The connector is compatible with our own EuroAmp/8 and EuroAmp/2 backplanes.

Power requirements for EuroSystem/EuroStep are:

+5V at 500mA
±12V at 50mA
Ground

Power can be derived from all Optimised Control amplifiers otherwise an external power supply is required.

The voltage must be kept at strictly +5V (±5%). Should the voltage drop below 4.75V the controller will be reset and held in that state until the voltage rises to above 4.9V. Reset is characterised by the LED status display showing an '8' with the decimal dot lit or being blank. The controller will be held in reset for a further 50ms once the voltage is restored.

Two other ground potential signals are used: analogue ground (AGND), used in the analogue input/output circuit, and screen (SCRN), a cable screen connection for the serial link and encoder inputs.

2.8.1. Battery Back-up

Program and data are retained in battery backed RAM while the controller is turned off. The battery, a rechargeable Nickel Cadmium (NiCad), is charged when the controller is powered up and will retain memory contents for at least 12 months if not recharged. The battery will take approximately 6 days to recharge after a complete discharge.

2.9. Encoder

Not applicable to EuroStep

Pinout:

Din Connector	**Backplane**	**Description**
CHA-0 (b7)	Brought out on D-type	Channel A axis 0
CHB-0 (a7)		Channel B
IDX-0 (b8)		Index
!CHA-0 (b10)		Channel A complement
!CHB-0 (c10)		Channel B complement
!IDX-0 (b9)		Index complement
CHA-1 (a8)		Channel A axis 1
CHB-1 (c7)		Channel B
IDX-1 (c8)		Index
!CHA-1 (c9)		Channel A complement
!CHB-1 (a10)		Channel B complement
!IDX-1 (a9)		Index complement

EuroSystem provides an interface for two independent three channel incremental encoders (CHA, CHB, INDEX) and operates with both single ended TTL or differential TTL output types. It is recommended that line driver outputs be used in all applications, since this gives increased noise immunity. It is important that each encoder cable is screened independently and that the screen is connected at the controller end only. Maximum cable length is dependent on the encoder specification, but should be kept as short as possible.

The input receiver circuit allows encoders with either single ended or differential line drivers to be used and is shown in figure 2.8.

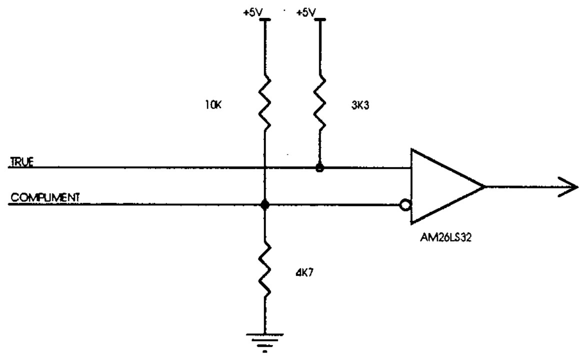


Figure 2.8 Encoder line receiver circuit.

2.10. Miscellaneous

2.10.1. Error Output

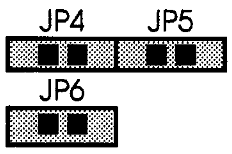
Pinout:

Din Connector	Backplane	Description
ERROR-OUT (b11)	MISC: EO	Error out signal
GND	MISC: G	Digital Ground

The error output provides a signal to external devices such as motor drives. The error signal is given when the controller detects a fault such as an end-of-travel limit switch open, maximum following error exceeded or a programming error. The output is jumper selectable as shown in the table below.

It is essential that the error output is connected to the drive and configured to the correct polarity with servo drives, since the state of the analogue outputs is undefined (but always below 100mV) during power-up. The error output ensures that the drive is disabled during this period. Once powered-up the error signal is maintained until deliberately removed by software. This allows time for system gains to be set before the motors are enabled. For stepper drives the error signal may be connected or not as desired. If not used all jumpers should be removed.

The jumpers JP4 to JP6, found next to the 96 way DIN connector, are used to select the output type.



JP6	JP5	JP4	Normal	Error
		❑	Floating	12V
	❑		0V	Floating
	❑	❑	0V	12V
❑	❑		12V	Floating
❑	❑		Floating	0V
❑	❑	❑	12V	0V

JP6 is used to invert the sense of the signals.

2.10.2. Stop Input

Pinout:

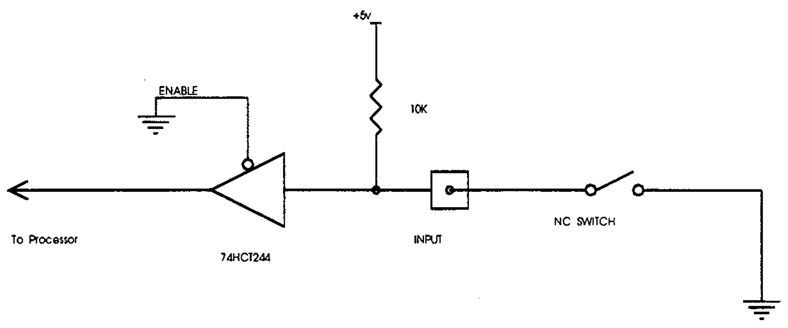
Din Connector	Backplane	Description
STOP (a25)	HOME/LIMIT: ST	Stop input
GND	HOME/LIMIT: Gnd	Digital Ground

The stop input provides a controlled stop on all axes when asserted. The stop input is active high and must be connected to a normally closed switch. If the stop input is not used it must be connected to ground otherwise the controller will be in stop condition, indicated by an 'S' on the LED status display.

The stop input is useful where a controlled stop is required such as machine guards where free wheeling could be dangerous.

The state of the stop input can be read in MINT using the STOPSW keyword. A subroutine within MINT can be called in response to a rising edge on the stop input.

The stop input buffer circuit is shown below:



2.10.3. Reset Input

Pinout:

Din Connector	Backplane	Description
RESET-IN (c12)	HOME/LIMIT: RS	Reset input
GND	HOME/LIMIT: Gnd	Digital Ground

The reset input will perform a hardware reset when the line is pulled low. This can be useful to provide an external reset of the system where turning the power off to the controller would prove to be inconvenient.

2.10.4. Error Input

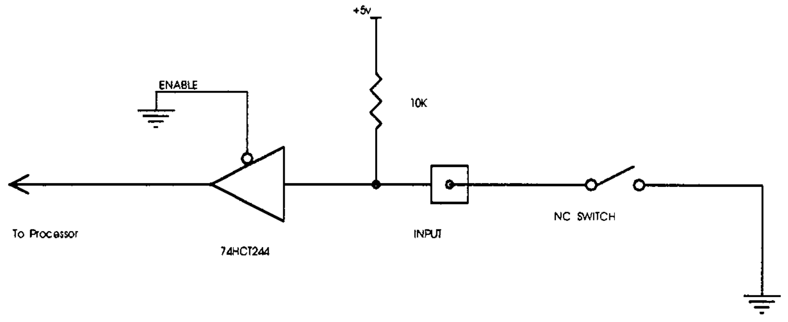
Pinout:

Din Connector	Backplane	Description
ERROR-IN (c11)	MISC: EI	Error input
GND	MISC: G	Digital Ground

The error input is used to detect error conditions in other parts of the system such as PLCs, or from the motor drives so that if a fault occurs on one motor the whole system is stopped.

The sense of this input is software selectable for active high or active low, it may also be disabled in software. The input is pulled up internally if no external connection in made.

The error input buffer circuit is shown below:



2.10.5. Fast Interrupt

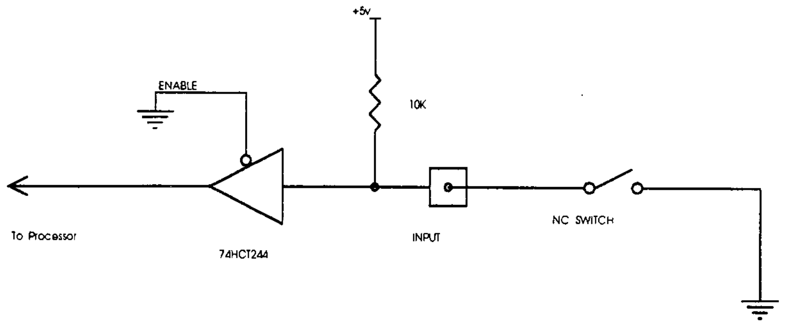
Pinout:

Din Connector	Backplane	Description
FAST-INT (c50)	MISC: FI	Error input
GND	MISC: G	Digital Ground

The fast interrupt is an external interrupt into the processor available with custom software only. This has a service time in the region of 25 microseconds as opposed to MINT interrupts which have a service time in the region of milliseconds. The primary function of the fast interrupt is to record axis positions which is provided as an option to the standard MINT software.

The Fast Interrupt has a very fast response time and will capture spurious voltage spikes if care is not taken over cable connections. An individually screened cable should be used to connect to this signal, and the neighbouring Gnd connection may be used to earth the respective screen.

The fast-interrupt buffer circuit is shown below:



2.10.6. Option Board Connections

Pinout:

Din Connector	**Backplane**	**Description**
OPT-1 (a23)	PWR/OPT: O1	Option 1
OPT-2 (b22)	PWR/OPT: O2	Option 2
OPT-3 (b23)	PWR/OPT: O3	Option 3
OPT-4 (c22)	PWR/OPT: O4	Option 4
GND	PWR/OPT: G	Digital Ground

OPT-1 to OPT-4 are connections which allow signals from custom designed option boards to be brought out to the backplane. These signals are only required when the option board is physically mounted on the option header and has only a small number of connections, for example a 4th stepper drive output which only requires three outputs and two chips.

2.11. Serial Port

EuroSystem/EuroStep has a full duplex serial port which can be either RS232 or RS485. The serial port is set up for the following configuration:

- 9600 Baud
- 1 start bit
- 8 data bits
- 1 stop bit
- No parity

MINT will transmit a carriage return/line feed (<CR><LF>) combination but only expects a carriage return (<CR>) from the host terminal.

cTERM and PC/MINT are pre-configured for use with EuroSystem/EuroStep.

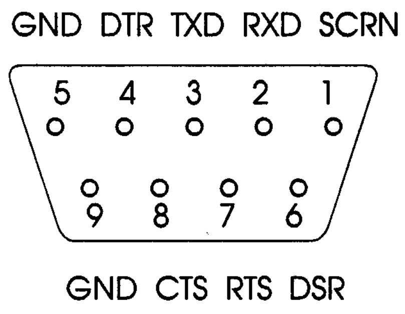
2.11.1. RS232

Pinout:

Din Connector	Backplane	Description
RXD (a20)	Brought out on D-type	Receive Data
TXD (a21)		Transmit Data
CTS (a22)		Clear to Send
RTS (b21)		Request to Send
DSR (a16)		Data Set Ready
DTR (a17)		Data Terminal Ready
SCRN (c32,b32,a32)		Screen

The RS232 connections are brought out onto a male 9-way D-type connector on the front of the controller. The same signals are also brought out onto the backplane.

The RS232 port is configured as a DTE (Data Terminal Equipment) unit so it is possible that EuroSystem will operate with any DCE (Data Communications Equipment) or DTE equipment. Full duplex transmission with CTS/RTS handshaking is supported. Both the output and input circuitry are single ended and operate between ±12V.



serial/mc

RS232 D-Type connector pinout

Pin No.	Signal Name & Function	Type
1	SCRN : Cable screen	Input
2	RXD : Receive Data	Input
3	TXD : Transmit Data	Output
4	DTR : Data Terminal Ready	Output
	(Internal connection to pin 6)	
5	GND : Signal Ground	
6	DSR : Data Set Ready	Input
	(Internal Connection to pin 4)	
7	RTS : Request to Send	Output
8	CTS : Clear to Send	Input
9	GND : Signal Ground	

The following table shows the wiring required for a standard IBM PC 25 way or 9 way connector:

Controller Pin No.	Signal Name and Function	Wire to: 25 Way	Wire to: 9 Way
1	SCRN : Cable screen	-	-
2	RXD : Receive Data	2	3
3	TXD : Transmit Data	3	2
4	DTR : Data Terminal Ready (Internal connection to pin 6)	6	6
5	GND : Signal Ground	7	5
6	DSR : Data Set Ready (Internal Connection to pin 4)	20	4
7	RTS : Request to Send	5	8
8	CTS : Clear to Send	4	7
9	GND : Signal Ground	7	9

2.11.2. RS485

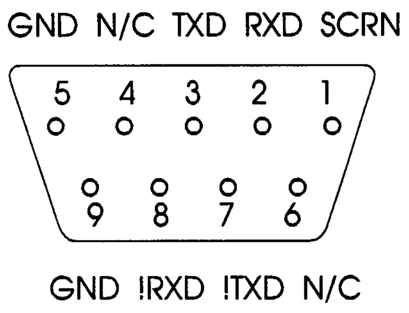
Pinout:

Din Connector	**Backplane**	**Description**
RXD (a20)	Brought out on D-type	Receive Data True
TXD (a21)		Transmit Data True
CTS (a22)		Receive Data Compliment
RTS (b21)		Transmit Data Compliment
DSR (a16)		Data Set Ready
DTS (a17)		Data Terminal Ready
SCRN (c32,b32,a32)		Screen

The RS485 connections are brought out onto the male 9-way D-type connector on the front of the controller and on the backplane.

The RS485 supports a full multi-drop protocol. Both the output and input signals are differential and operate between 0 and 5V.

The D-Type signals are also brought out on the 96-way connector.



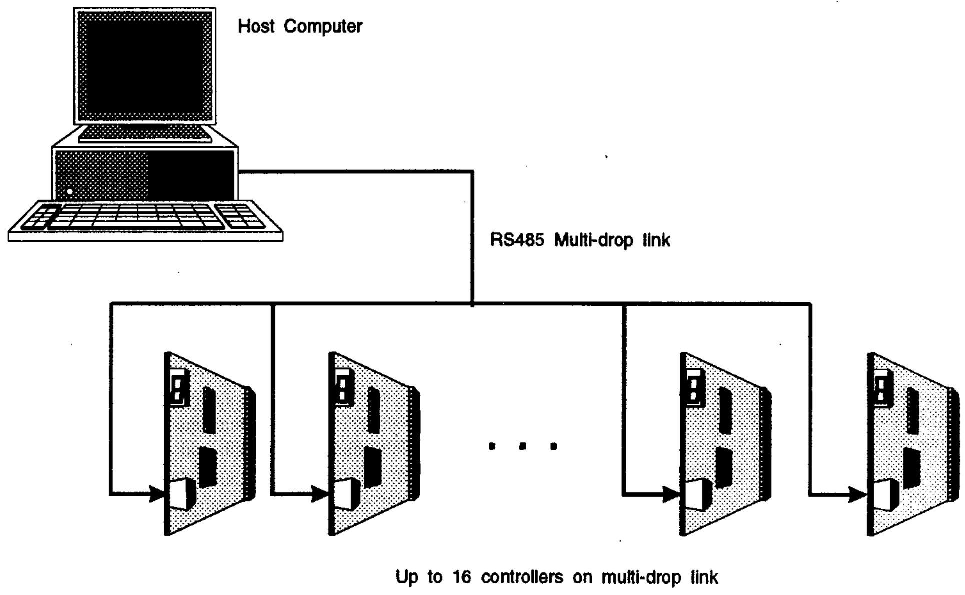
RS485/mc

RS485 D-Type connector pinout

Pin No.	Signal Name & Function	Type
1	SCRN : Cable screen	Input
2	RXD : Receive Data	Input
3	TXD : Transmit Data	Output
4	Not connected	
5	GND : Signal Ground	
6	Not connected	
7	!TXD : Transmit Data Compliment	Output
8	!RXD : Receive Data Compliment	Input
9	GND : Signal Ground	

2.11.2.1. RS485 Multi-Drop

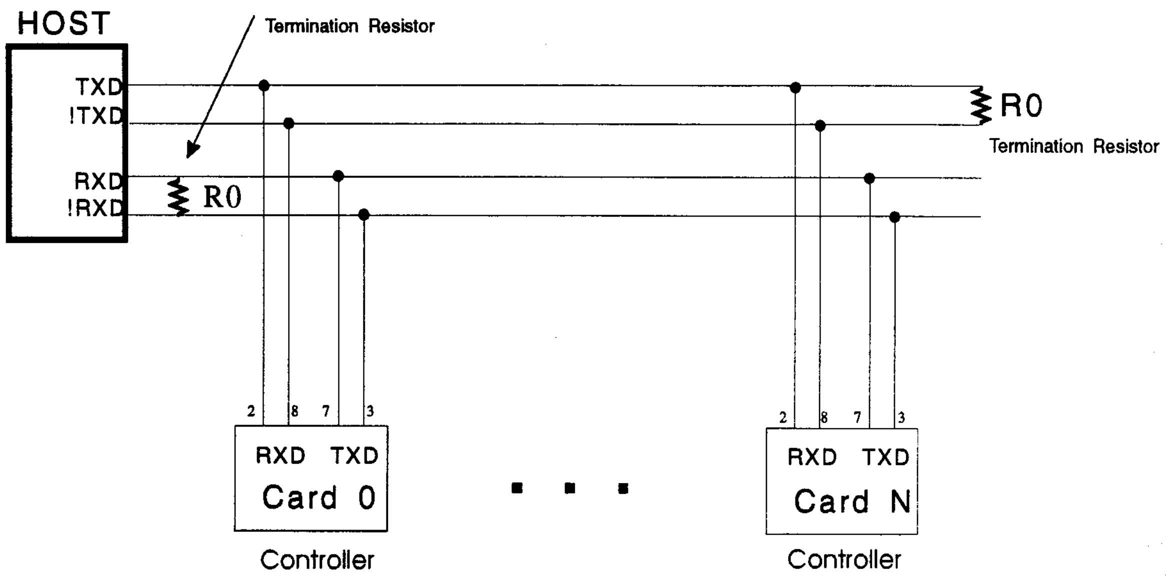
A multi-drop system can be easily configured using a ribbon cable and IDC D-Type connectors to the controller cards. A multi-drop layout is shown in the diagram below.



multi/mc/0392

EuroSystem supports up to 16 cards on the serial line, where each card is distinguished by a unique address set by a 4 bit DIP switch. See section 3 for more details on the card address. Software details on multi-drop can be found in the MINT Programming Guide.

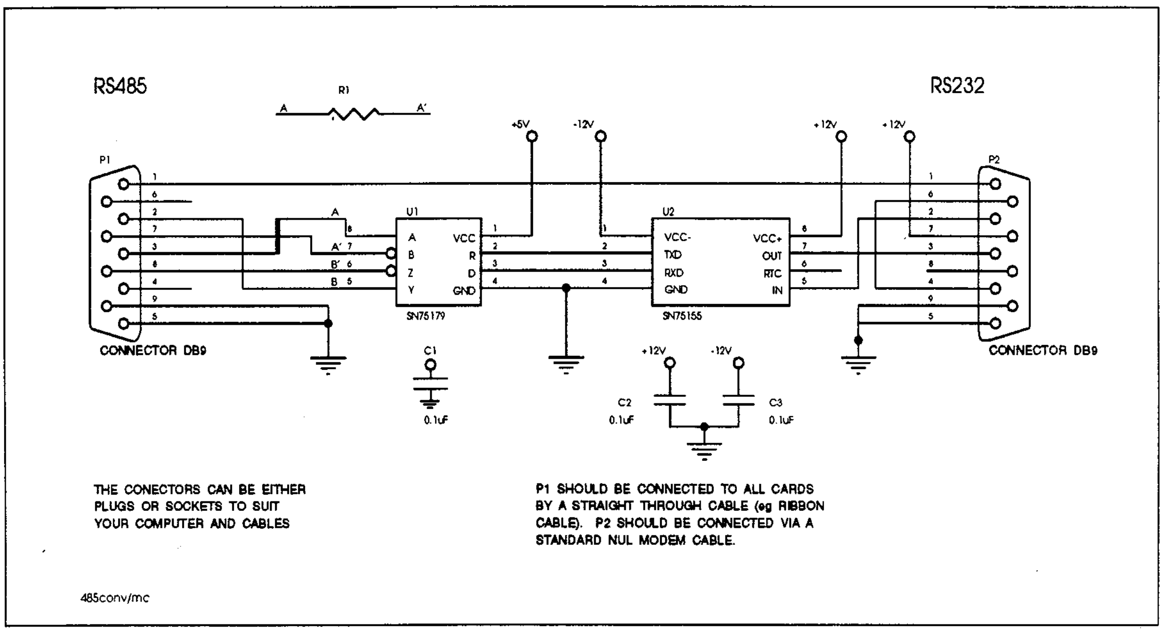
Note that the RXD and TXD lines must be terminated with 120 ohm resistors.



mult485/mc

2.11.2.2. RS485 to RS232 Converter

The following shows a schematic for an RS485 to RS232 converter. This is useful for connecting an IBM PC, using a null modem cable, to a multi-drop link and using the PC as a host.



THE CONECTORS CAN BE EITHER PLUGS OR SOCKETS TO SUIT YOUR COMPUTER AND CABLES

P1 SHOULD BE CONNECTED TO ALL CARDS BY A STRAIGHT THROUGH CABLE (eg RIBBON CABLE). P2 SHOULD BE CONNECTED VIA A STANDARD NUL MODEM CABLE.

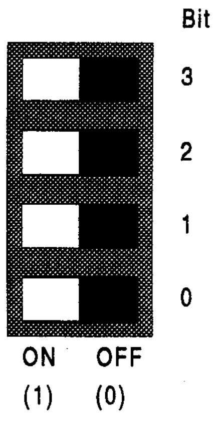
The 485 line must be correctly terminated by placing 120 ohm resistors across A and !A (A complement) and B and !B as near to the driver (SN75179) as possible. If the 485 line is not correctly terminated, characters transmitted over the serial link may be corrupted.

The 232 connector must be connected as close to the PC serial output as is possible to effectively give the PC an RS485 output.

3. Card Address

Up to 16 controllers can be connected together over a multi-drop RS485 link for host control. Each card on the link is distinguished by a unique address set by a 4 bit DIP switch located next to the processor.

Controller Address Switch:



DIP/MC

Switch Position 3210	Address	LED
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	10	A
1011	11	b
1100	12	C
1101	13	d
1110	14	E
1111	15	F

On power up, the controller will show the address number on the LED status for about 1 second. The DIP switch value can be read in MINT using the CARD keyword.

If the controller is used in a stand-alone system, the card address should be set to address 0. A RS232 controller will always default to address 0 regardless of the switch positions.

4. EuroSystem/EuroStep Backplane Connections

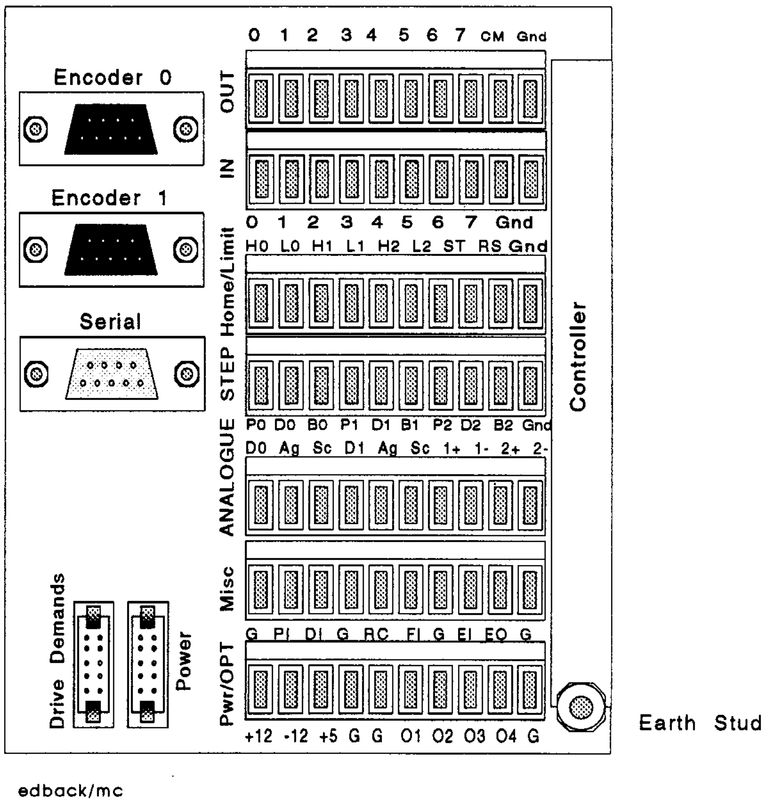


Figure 4.1: EuroSystem/EuroStep Backplane

Your backplane, figure 4.1, has been designed to provide easy connection between EuroSystem/EuroStep and your application signals. All signals with the exception of the encoders and RS232/485 are connected to via seven removable ten way connectors. Encoders and RS232/485 are connected to via nine pin D type connectors.

You may also link directly to our range of stepper and servo products and their respective backplanes with the simple addition of a ribbon cable. Stepper signals for all three axes have been duplicated and made available on three, ten way ribbon cable headers. Servo drive signals and power are also available on independent ten way ribbon cable headers.

Connections to the backplane are as follows. For a full description of the function of each signal refer to section 2.

4.1. Digital Outputs: OUT

Signal	Function
O0	Digital output bit 0
O1	Digital output bit 1
O2	Digital output bit 2
O3	Digital output bit 3
O4	Digital output bit 4
O5	Digital output bit 5
O6	Digital output bit 6
O7	Digital output bit 7
CM	Common diode connection
Gnd	Digital ground

The outputs have a Common diode clamp connection accessible from the backplane through USR-OUT-COM. This connection is for use with inductive loads and should be to the external supply from which you are sourcing current.

4.2. Digital Inputs: IN

Signal	Function
I0	Digital input bit 0
I1	Digital input bit 1
I2	Digital input bit 2
I3	Digital input bit 3
I4	Digital input bit 4
I5	Digital input bit 5
I6	Digital input bit 6
I7	Digital input bit 7
GND	
GND	Digital ground

4.3. HOME/LIMIT

Signal	Function
H0	Home Input 0
L0	Limit Input 0
H1	Home Input 1
L1	Limit Input 1
H2	Home Input 2
L2	Limit Input 2
ST	Stop Input
RS	Reset Card Input
Gnd	Digital Ground

Note:

The limit and stop inputs must be grounded if not used for the system to operate correctly.

4.4. Stepper Drive Connections: STEP

Signal	Function
P0	Pulse Output 0
D0	Direction Output 0
B0	Boost Output 0
P1	Pulse Output 1
D1	Direction Output 1
B1	Boost Output 1
P2	Pulse Output 2
D2	Direction Output 2
B2	Boost Output 2
Gnd	Digital Ground

The pulse output from the controller has a fast fall time and as such it can be a source of interference to neighbouring signals. You are strongly recommended to use an individually screened cable for the pulse signal and to run other signals separately. The screen for this signal should be connected to the ground stud on the backplane and not at the drive end.

4.5. Servo Drive Connections: ANALOGUE

Some of the signals on this connector are duplicated on the servo ribbon header (see later). It is possible to use either the ribbon cable connection or this connection for any axis, but not both.

Signal	Function
D0	Demand Output 0 (demand +)
AG	Analogue Ground (demand -)
SC	Cable Screen
D1	Demand Output 1 (demand +)
AG	Analogue Ground (demand -)
SC	Cable Screen
1+	Analogue Input 1 (differential +ve)
1-	Analogue Input 1 (differential -ve)
2+	Analogue Input 2 (differential +ve)
2-	Analogue Input 2 (differential -ve)

4.6. Miscellaneous: MISC

The Fast Interrupt and Pulse input have very fast response times and will capture spurious voltage spikes if care is not taken over cable connections. Individually screened cables should be used to connect to these signals, and the neighbouring Gnd connection may be used to earth the respective screen.

Signal	Function
G	Ground
PI	Pulse Input
DI	Direction Input
G	Ground
RC	Reset Counter Input
FI	Fast Interrupt Input
G	Ground
EI	Error Input
EO	Error Output
G	Ground

4.7. Power: PWR/OPT

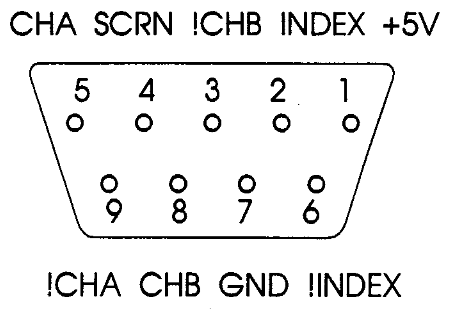
The power signals are duplicated on a ten way ribbon connector and as such care should be taken not to connect to both of these simultaneously.

Signal	Function
+12	+12V power supply input
-12	-12V power supply input
+5	+5V power supply input
G	Digital Ground
G	Digital Ground
O1	Option 1
O2	Option 2
O3	Option 3
O4	Option 4
G	Digital Ground

O1 to O4 are used to bring out signals from option boards.

4.8. Encoder Connections

The encoder inputs are brought out onto 9 pin 'D' type female sockets. The encoder should be wired to a 9 pin 'D' male plug.



encoder/mc

Pin No.	Signal Name & Function	Type
1	+5V : Power to Encoder	Output
2	INDEX : Index Mark	Input
3	!CHB : Channel B Compliment	Input
4	SCRN : Cable Screen	Input
5	CHA : Channel A	Input
6	!INDEX : Index Complement	Input
7	GND : Signal Ground	
8	CHB : Channel B	Input
9	!CHA : Channel A Compliment	Input

4.9. Servo Drive: IDC Connector

Function	Pin			Pin	Function
Servo Error Out	1	❑	❑	2	Error In
Cable Screen	3	❑	❑	4	Analogue Ground
Demand_0 +ve out	5	❑	❑	6	Demand_0 +ve out
Demand_0 -ve (Agnd)	7	❑	❑	8	Demand_1 -ve (Agnd)
Demand_1 +ve Out	9	❑	❑	10	Demand_1 +ve Out

Note:

Demand _0 -ve and demand _1 -ve are common and are connected to analogue ground.

4.10. Power: IDC Connector

Function	Pin			Pin	Function
Dgnd	1	❑	❑	2	Dgnd
Dgnd	3	❑	❑	4	Dgnd
+5V	5	❑	❑	6	+5V
+12V	7	❑	❑	8	+12V
-12V	9	❑	❑	10	-12V

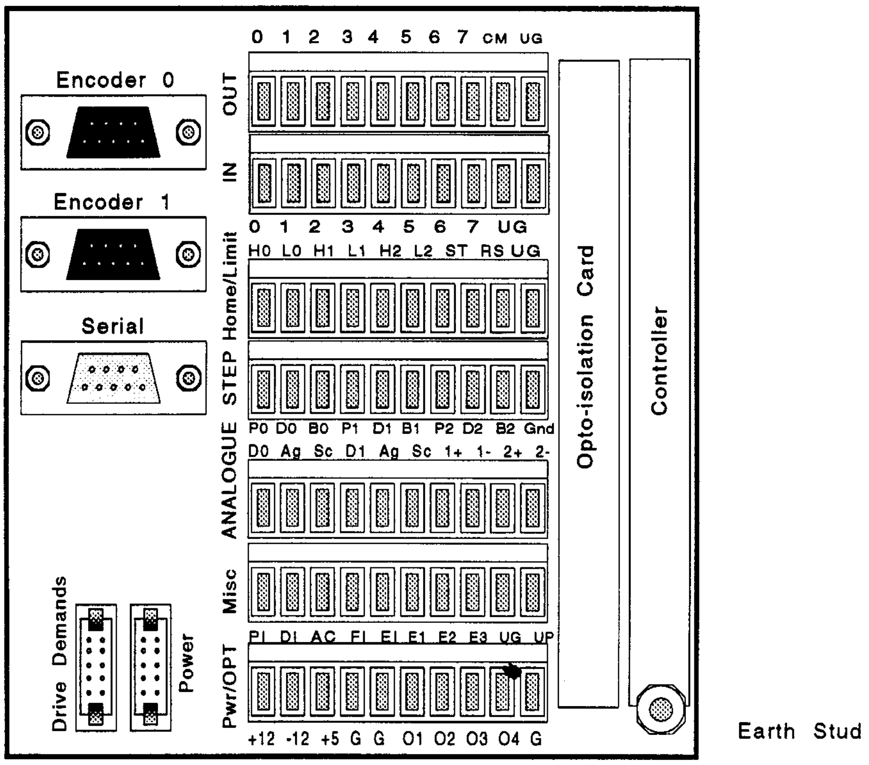
4.11. Earth Stud

This connection should be used as the main earth point for your controller. It should be connected to a low impedance star earth point within the system.

4.12. RS232/485

The RS232/485 port available on the front of the card is duplicated on the backplane.

5. Optically Isolated Backplane



edbck_o/mc/1193

Figure 6.1: Optically Isolated Backplane

The optically isolated backplane isolates the following signals:

- Digital user inputs
- Digital user outputs
- Pulse following inputs (PULSE, DIRECTION, RESET)
- Limit inputs
- Home inputs
- Reset input
- Stop input
- Error input
- Fast interrupt
- Error output

The optically isolated backplane has been designed to have the same connection details as the standard backplane, with the exceptions of the miscellaneous block (see section 6.1).

In the optically isolated backplane system all active components are mounted on a small signal conditioning card which connects to the backplane via a 96 way DIN 41612 'R' type connector. This allows different signal conditioning options to be offered, such as NPN or PNP type inputs and outputs, or custom conditioning cards, for instance, connection to proximity sensors.

5.1. Differences from Standard Backplane

5.1.1. Error Output

The error output is isolated with a change over relay because of the wide variety of voltages and polarities the output may be connected to. The contact rating of the relay is 2A @ 30Vdc or 0.6A @ 150Vac. The connections to the relay are brought out on the MISC connector through E1, E2, E3.

Connector	Function
E1	Error output common
E2	Error output normally open
E3	Error output normally closed

5.1.2. MISC Connector

Because of the additional power requirements and extra Error output connections, the MISC connector is re-arranged as follows.

PI	Pulse Input
DI	Direction Input
AC	Accumulator Clear (reset counter) Input
FI	Fast Interrupt Input
EI	Error Input
E1	Error Output Common
E2	Error Output Normally Open
E3	Error Output Normally Closed
UG	User (Isolated) Ground
UP	User (Isolated) Power

5.1.3. User (Isolated) Power

The opto isolated backplane system is designed to interface EuroSystem to external voltages of up to 24V. If a voltage other than 24V is used, some component values must be changed to suit the voltage used, contact Optimised Control for details. With a 24V voltage, the card is compatible with most PLC's.

Power is connected through the MISC connector on pins 9 and 10, labelled UP (user power) and UG (user ground) respectively.

The Limit/Home, Digital Inputs and Pulse following inputs, being isolated, have a User Ground (UG) reference connection. The stepper outputs (pulse, direction and boost), which are not isolated, have a controller ground (Gnd) reference connection.

The user outputs have their own power connections. See section 6.2.2 for details.

5.2. Signal Connections

5.2.1. All Inputs

All inputs applies to user inputs, homes, limits, reset, stop, fast interrupt, pulse, direction and reset counter.

When using the **NPN** signal conditioning card (current sink) the inputs will become active when taken to isolated ground (UG), providing the switch can sink 10mA.

When using the **PNP** signal conditioning card (current source), connections to the inputs must be able to source 10mA (5mA min) into a 2k2 load (24V will give 11mA) for the input to become active. This current must drop to less than 1mA to ensure that the input becomes inactive.

5.2.2. User Outputs

This applies to user outputs only.

Two power connections are specific to the output driver circuit, they are OG and CM. These terminals are connected differently for the different signal conditioning cards (NPN or PNP).

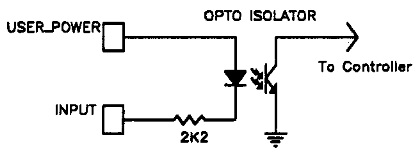
For **NPN** outputs, OG is connected to UG, and CM is a flyback diode connection for inductive loads. The NPN output driver can sink a maximum of 400mA continious on a single channal, with a total maximum of 800mA for all 8 channels.

For **PNP** outputs, OG is connected to UP, and CM is connected to UG. Note that with this driver the flyback diodes are connected internally. The PNP output drivers can source a maximum of -350mA continuous on a single channel, with a total maximum of -750mA for all 8 channels.

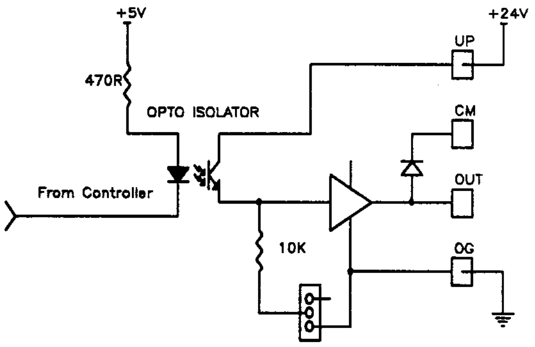
5.2.3. Jumpers

The jumper, J1, on the signal conditioning card is factory set and must not be moved.

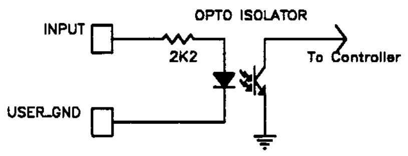
5.2.4. Opto Isolation Circuit Diagrams



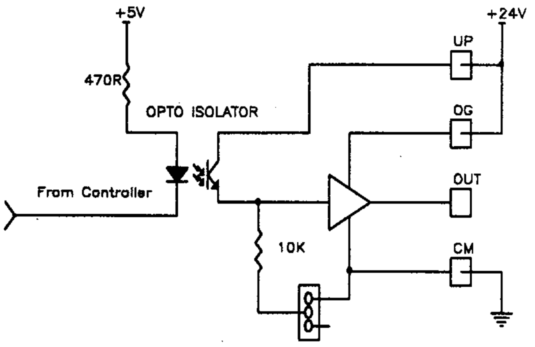
NPN INPUTS



NPN OUTPUTS

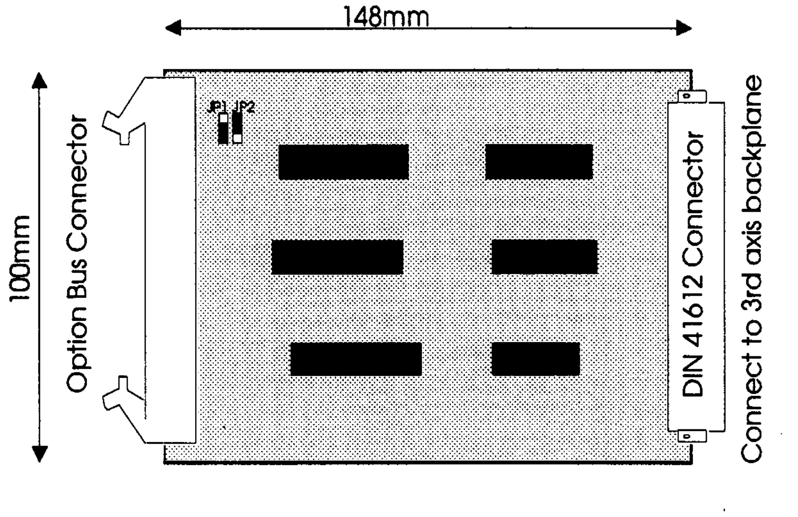


PNP INPUTS

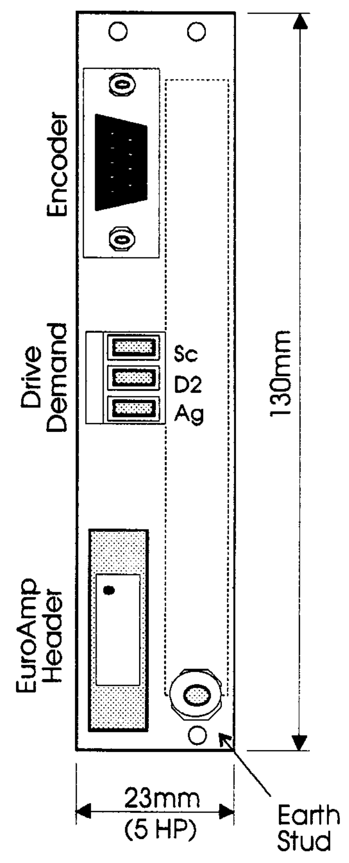


PNP OUTPUTS

6. 3rd Axis Servo Option Board



3rd axis servo board



Ref: 3axis/mc/1193

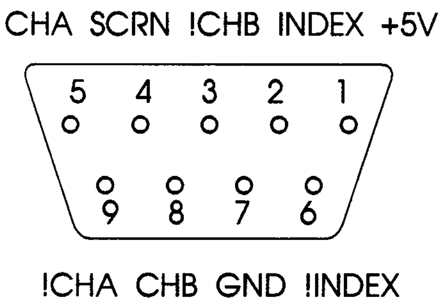
3rd axis backplane

The 3rd axis servo option board provides additional hardware to control a third axis of servo motion for the EuroSystem controller. The board has an incremental encoder input which supports three channel differential or single-ended encoders, and a 12 bit resolution analogue output to control high performance servo drives. Limits and home inputs are provided as standard on the EuroSystem controller for the 3rd axis servo, as is support by MINT.

Connection between EuroSystem and the 3rd axis board board is provided by a ribbon cable from the EuroSystems's option bus connector. Connections to the servo drive and encoder are made via a backplane which sits along side the EuroSystem backplane. The connection details of the backplane are as follows:

6.1. Encoder Connections

The encoder input is brought out onto 9 pin 'D' type female sockets. The encoder should be wired to a 9 pin 'D' male plug.



encoder/mc

Pin No.	Signal Name & Function	Type
1	+5V : Power to Encoder	Output
2	INDEX : Index Mark	Input
3	!CHB : Channel B Compliment	Input
4	SCRN : Cable Screen	Input
5	CHA : Channel A	Input
6	!INDEX : Index Complement	Input
7	GND : Signal Ground	
8	CHB : Channel B	Input
9	!CHA : Channel A Compliment	Input

6.2. Drive Demand Terminal Block

Terminal	Function
Sc	Cable Screen
D2	Demand output 2 (demand +)
Ag	Analogue Ground (demand -)

6.3. EuroAmp Header: IDC Connector

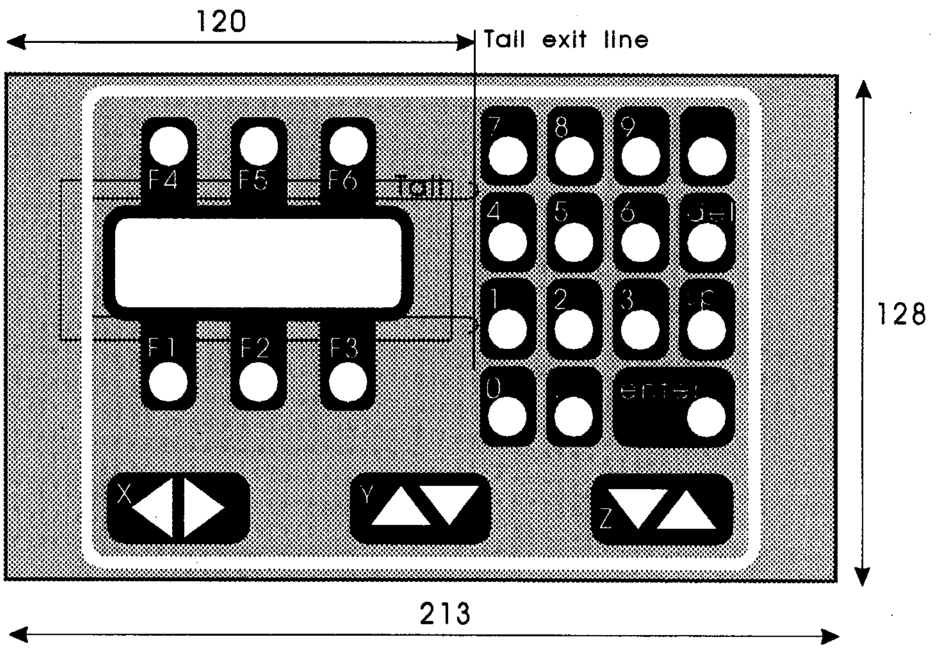
The EuroAmp header provides a direct connection to the EuroAmp range of amplifiers via a 10 way IDC connector.

Function	Pin			Pin	Function
Drive enable (error out)	1	❑	❑	2	Error In
Cable Screen	3	❑	❑	4	Analogue Ground
Demand_2 +ve Out	5	❑	❑	6	Demand_2 +ve Out
Demand_2 -ve (Agnd)	7	❑	❑	8	Demand_2 -ve (Agnd)
Not connected	9	❑	❑	10	Not connected

6.4. Earth Stud

This connection should be taken to a low impedance star earth point within the system.

7. Keypad and Display Option



keypad/mc/0392

Figure 7.1: Operator Panel Layout

The keypad and operator terminal provide a general purpose operator terminal suitable for stand alone machines of all types. The operator terminal is cost effective for simple functions, such as replacing thumb wheel switches and providing simple diagnostics, or may be used as a fully interactive programming panel for special purpose machine control.

The keypad interface board allows Hitachi LCD displays or compatible units to be driven from the EuroSystem controller directly. The board also provides connection for up to 64 keys arranged on an 8 by 8 matrix using normally open switches. MINT allows users to define the value of each key in the matrix using a keyword KEYS, which means that the physical order of connection is not important.

Alphanumeric LCD displays of up to 40 characters by 2 lines or 20 by 4 lines may be used. Connection is made via a 34 way IDC ribbon cable between the interface board and a daughter board mounted on the panel,

A piezoelectric buzzer is attached to the display panel. A short beep (using the BEEP keyword) can be used to acknowledge a key press, whilst a series of beeps can be used to attract operator attention. The high pitch of the buzzer makes it audible over general industrial noise from machinery.

MINT software supports the keypad and display as if it were a standard serial terminal. MINT statements PRINT, INPUT, CLS, LOCATE etc. can be used with the display, BEEP activates the buzzer. Key presses cause characters to be placed in the serial port buffer so that they can be read as normal by INKEY and INPUT. The READKEY function returns the value of the key that is currently pressed. This is an enhancement not normally available on serial terminals, very useful for jogging motors when the operator holds his finger on a key. The following keywords are supported:

Keyword	Description
BEEP	Sound the buzzer
BINARY	Print a binary number
BOL	Send cursor to beginning of line
CLS	Clear screen
INKEY	Read a key from the serial port buffer
INPUT	BASIC formatted input
KEYS	Define layout of keyboard
LOCATE	Locate cursor at column, row
PRINT	BASIC formatted print
READKEY	Read value of key currently pressed
TERM	Direct output to LCD or serial port

A general purpose operator panel is available, suitable for many machine control applications. The panel is available as a kit of parts, comprising of adhesive matrix board and display, so that it can be mounted on the front of a machine control panel with a cut-out for the display. The matrix panel is designed so that the connecting tail exits from the middle of the board, through the side of the display cut-out. Therefore when stuck in place, the operator panel is sealed to IP65.

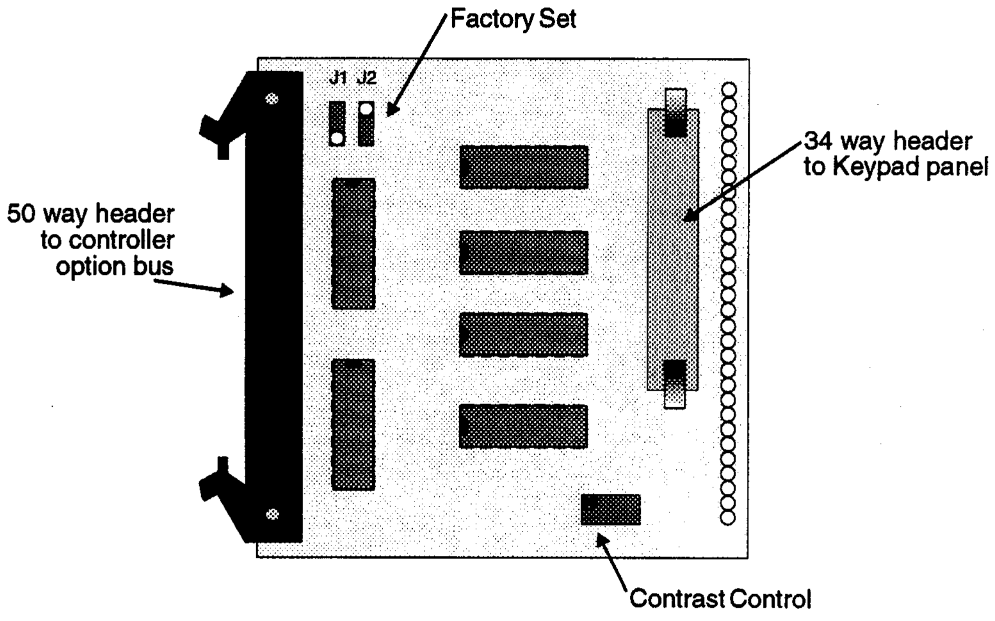
A fully assembled panel is also available, comprising the display and matrix board mounted on an aluminium panel. The assembled operator panel occupies half a standard 19" 3U high eurocard rack.

The keypad interface makes it easy to build your own operator panel, using push buttons and a display, which is exactly suited to your application. For a really professional finish, it may be preferable to commission a custom keypad, but this is expensive. An alternative is to custom print the front of our standard keypad, which we can arrange for quantities of 25 off, which gives savings on tooling.

The operator panel incorporates six function keys placed above and below the display, so that the function of each key can be indicated by printing a legend on the top and bottom lines of the display. This allows menu driven operator interfaces to be written readily, with the function of each function key changing depending on the menu. The middle two lines are used for messages or operator prompts.

7.1. Specification

7.1.1. Keypad and Display Interface Board


Factory Set
J1 J2
34 way header to Keypad panel
50 way header to controller option bus
Contrast Control

Ref: keyint/MC/1193

Figure 7.2: Keypad and Display Interface Board

- Maximum cable run between interface board and operator panel 5M
- Display types: Hitachi or compatible alphanumeric LCD modules.
- Keypad interface: 64 keys maximum on 8x8 matrix. Operating voltage 5V, keys should be normally open types, 200mOhm maximum contact resistance.
- Piezoelectric Buzzer: 6.5 KHz, 80 Db @ 10cm mounted on the back of the display

Operator Panel:

- Overall dimemsions: 128 x 213mm. Occupies half a standard 19" 3U high eurocard rack.
- 27 keys: X, Y, Z cursor keys, numeric keypad and 6 user definable function keys.
- Self adhesive backing providing protection against ingress of fluid and dust to IP65.
- 20 character by four line green backlit LCD display. Character height: 4.75mm. Viewing area: 76.0 x 25.5mm

7.2. Interface Connection Details

7.2.1. Connecting to EuroSystem

The keypad and display interface board is connected to EuroSystem on the option bus via a 50 way IDC connector. Ensure that pin 1 of EuroSystem to connected to pin 1 of the keypad and display interface board.

7.2.2. Connecting to the Panel

The keypad interface is connected to the panel via a 34 way IDC cable. The maximum permissable length of the cable is 5m.

The 34 way pin out is given below:

Function	Pin			Pin	Function
Keypad: Row 1	1	❑	❑	2	LCD Display: V$_{ss}$ Ground
Keypad: Row 2	3	❑	❑	4	LCD Display: V$_{dd}$ +5V logic
Keypad: Row 3	5	❑	❑	6	LCD Display: V$_{o}$ Power Supply for LCD
Keypad: Row 4	7	❑	❑	8	LCD Display: RS Register Select
Keypad: Row 5	9	❑	❑	10	LCD Display: RW Read/Write (Connected to GND)
Keypad: Row 6	11	❑	❑	12	LCD Display: E1 Start Pulse
Keypad: Row 7	13	❑	❑	14	LCD Display: DB0 Data Bit 0
Keypad: Row 8	15	❑	❑	16	LCD Display: DB1 Data Bit 1
Keypad: Column 1	17	❑	❑	18	LCD Display: DB2 Data Bit 2
Keypad: Column 2	19	❑	❑	20	LCD Display: DB3 Data Bit 3
Keypad: Column 3	21	❑	❑	22	LCD Display: DB4 Data Bit 4
Keypad: Column 4	23	❑	❑	24	LCD Display: DB5 Data Bit 5
Keypad: Column 5	25	❑	❑	26	LCD Display: DB6 Data Bit 6
Keypad: Column 6	27	❑	❑	28	LCD Display: DB7 Data Bit 7
Keypad: Column 7	29	❑	❑	30	LCD Display: V$_{led}$ Backlight Supply
Keypad: Column 8	31	❑	❑	32	Buzzer: R
Buzzer: BK	33	❑	❑	34	Buzzer: BL

Alternatively, the connections are brought out onto a 41612 Type B DIN connector for use with custom backplanes etc. The connector supports the buzzer, the LCD display and an 8 row by 6 column keypad. The connector has the following pin out:

	Pin	Function
❑	1	Buzzer: R
❑	2	Buzzer: BK
❑	3	Buzzer: BL
❑	4	Keypad: Row 1
❑	5	Keypad: Row 2
❑	6	Keypad: Row 3
❑	7	Keypad: Row 4
❑	8	Keypad: Row 5
❑	9	Keypad: Row 6
❑	10	Keypad: Row 7
❑	11	Keypad: Row 8
❑	12	Keypad: Column 1
❑	13	Keypad: Column 2
❑	14	Keypad: Column 3
❑	15	Keypad: Column 4
❑	16	Keypad: Column 5
❑	17	Keypad: Column 6
❑	18	LCD Display: V_{ss} Ground
❑	19	LCD Display: V_{dd} +5V logic
❑	20	LCD Display: V_o Power Supply for LCD
❑	21	LCD Display: RS Register Select
❑	22	LCD Display: RW Read/Write (Connected to GND)
❑	23	LCD Display: E1 Start Pulse
❑	24	LCD Display: DB0 Data Bit 0
❑	25	LCD Display: DB1 Data Bit 1
❑	26	LCD Display: DB2 Data Bit 2
❑	27	LCD Display: DB3 Data Bit 3
❑	28	LCD Display: DB4 Data Bit 4
❑	29	LCD Display: DB5 Data Bit 5
❑	30	LCD Display: DB6 Data Bit 6
❑	31	LCD Display: DB7 Data Bit 7
❑	32	LCD Display: V_{led} Backlight Supply

8. Additional Options

The following options are available for EuroSystem and EuroStep.

8.1. Three Channel Encoder Board: ES/08

The 3 axis encoder board is used primarily for providing 3 independent master input signals for encoder following, software gearboxes and Cams. Additionally it can be used with a stepper system to provide accurate position feedback. Note that this is not closed loop stepper control, it only provides confirmation of the stepper position.

The 3 channel encoder board requires a firmware upgrade.

Note: The Three Channel Encoder Board cannot be used in conjunction with the Third Axis Servo Board on a EuroSystem controller. The third axis must be turned off before use (CONFIG[2] = _off).

8.2. Prototyping Board:ES/PROT

The prototyping board consists of a EuroSystem option bus interface hardware, a 33x38 matrix of plated holes on a 0,1" grid for assembling wire-wrap circuits. A 3x32 matrix (with mounting holes) is provided for DIN 41612 type A,B,C,D and F connectors.

The prototyping board can be used to provide additional I/O, for example relay contacts or analogue inputs. Support in MINT is through use of the PEEK and POKE keywords.

8.3. I/O Expansion Board: ES/01/P & ES/01/N

The 24 I/O board provides 24 additional inputs and 24 additional outputs, optically isolated. NPN or PNP can be provided.

The board is supported within the firmware as standard using the XIO keyword.

8.4. Smart (Memory) Card Interface: ES/07

The memory card interface consists of an interface board and either a 32K or 64K credit card sized memory card. Given one of two firmware options, the memory card can be used for:

1. Program storage (/MA - Memory card Auto boot). A program, configuration file and array data can be stored on a 32K memory card. If `AUTO` is detected within the configuration file held on the memory during power up, the program file and configuration file are copied into the controller's memory and execution begins as normal. To load the array data, the command `LOAD -3` must be inserted into the program. When programs etc are downloaded from cTERM, these are stored directly into the controller memory. To save to the memory card, use `SAVE -1` for the program, `SAVE -2` for the configuration and `SAVE -3` for array data. The /MA option can be used to update software in the controller memory by powering the system up with the memory card in place. The memory card can then be removed since the program and configuration file will be held in the controller memory.

2. Expandable memory (/MX - Memory card eXpansion). In this instance, the program and configuration files are stored on the memory card when they are downloaded from the host PC. When **AUTO** or **RUN** is executed, the controller will tokenise the code and place the tokenised code in the controller's memory. The tokenised code will then be executed. Array data can be saved and loaded using **SAVE -3** and **LOAD -3** respectively. It should be noted that with the /MX option, MINT does not support the command line editor although a program can be listed using the **LIST** keyword. Files can be stored on either a 32K or 64K memory card. In order support larger programs, a maximum of 100 variables and 80 subroutines can be defined. It should also be noted that with this option a memory card must be in place at all time.

Note that both these options only allow one set of array data per card. The array data must be loaded in to the controller address space by use of the **LOAD -3** command. Conversely, the **SAVE -3** command must be used to save the array data to the memory card.

8.5. Encoder Splitter/Buffer Board: ES/08/E

This is not an option that fits onto the controllers option bus but is a stand alone PCB that takes an encoder signal, either single ended or differential and gives 2 differential outputs. This is useful for 'chaining' an encoder signal from a master across a number of controllers.

9. Appendix A: Technical Specifications

9.1. Technical Specification: Non-Isolated

9.1.1. Motor Control I/O

Stepper outputs:	Step/Direction/Boost
	Axis 0,1,2 open collector
	100 mA sink each
	10 Hz to 200Hz frequency output
Analogue outputs:	±10V, 10 mA sink/source
	12 bit resolution
Encoder inputs:	Minimum requirements: Channel A, Channel B single ended TTL
	Preferred: Channel A, Channel B, Index, differential TTL line driver
	Encoder frequency 1MHz maximum
	9 way D-type female connector on backplane
Limit inputs:	Axis 0,1,2
	Connect to ground through normally closed switches
Home inputs:	Axis 0,1,2
	Connect to ground through normally closed switches
Stop input:	Motion inhibited when high, axes decelerate to stop
Reset input:	Pull low for hardware reset
Error input:	Motion inhibited. Acitive high or low, configured by software.
Enable output:	Active high (12V) or low (0V), hardware configured.
	100mA maximum sink/source, totem pole.

9.1.2. Machine Control I/O

User inputs:	8 input lines, logical one when 5V, logical zero when 0V
User outputs:	8 output lines, open collector
	400 mA max sink per output, 800mA total sink
Analogue inputs:	2 inputs
	±10V or 0-5V jumper selectable
	10 bit resolution
Pulse/Direction:	TTL 700kHz maximum
	Pulse input counts rising and falling edges
	Reset counter edge triggered, tie to ground if not used
Serial Port:	RS232 or duplex RS485
	9 way D-type male connector on front panel, duplicated on backplane
	9600 baud, 1 start bit, 8 data bits, 1 stop bit, no parity
	No handshaking
	RS485 supports multi-drop of up to 16 cards

9.1.3. General

Power:	+5V at 500mA
	±12V at 50mA
Operating temperature:	0 - 45°C
Connectors:	Backplane connector: DIN 41612, 96-way
	Option bus connector: 50 way IDC header
	Power/drive demands: 10 way IDC headers

9.2. Technical Specification: Isolated

All isolated signals are marked with *.

9.2.1. Motor Control I/O

Stepper ouputs:	Step/Direction/Boost Axis 0,1,2 open collector 100 mA sink each 10 Hz to 200Hz frequency output
Analogue outputs:	±10V, 10 mA sink/source 12 bit resolution
Encoder inputs:	Minimum requirements: Channel A, Channel B single ended TTL Preferred: Channel A, Channel B, Index, differential TTL line driver Encoder frequency 1MHz maximum 9 way D-type female connector on backplane
Limit inputs*:	Axis 0,1,2 Connect to ground (NPN)/power (PNP) through normally closed switches
Home inputs*:	Axis 0,1,2 Connect to ground (NPN)/power (PNP) through normally closed switches
Stop input*:	Motion inhibited when active, axes decelerate to stop
Reset input*:	Pull low for hardware reset
Error input*:	Motion inhibited. Acitive high or low, configured by software.
Enable output*:	Active high (12V) or low (0V) on 10 way IDC, hardware configured Change over relay on screw terminations

9.2.2. Machine Control I/O

User inputs*:	8 input lines, logical one when in-active, logical zero when active
User outputs*:	8 output lines, open collector NPN:400 mA max sink per output, 800mA total sink PNP:300 mA max sink per output, 750mA total sink
Analogue inputs:	2 inputs ±10V or 0-5V jumper selectable 10 bit resolution
Pulse/Direction*:	TTL 35kHz maximum Pulse input counts rising and falling edges Reset counter edge triggered, tie to ground if not used
Serial Port:	RS232 or duplex RS485 9 way D-type male connector on front panel, duplicated on backplane 9600 baud, 1 start bit, 8 data bits, 1 stop bit, no parity No handshaking (Hardware handshaking for HPGL) RS485 supports multi-drop of up to 16 cards

9.2.3. General

Power:	+5V at 500mA ±12V at 50mA Isolated supply voltage
Operating temperature:	0 - 45°C
Connectors:	Backplane connector: DIN 41612, 96-way Option bus connector: 50 way IDC header Power/drive demands: 10 way IDC header, not isolated

Software Reference Manual

Applications and Utilities Diskette

Issue 1.01

Ref: utils/mc 0394:M00106-002

Copyright © Optimised Control Ltd 1995.

All rights reserved.

This manual is copyrighted and all rights are reserved. This document may not, in whole or in part, be copied or reproduced in any form without the prior written consent of Optimised Control.

Optimised Control makes no representations or warranties with respect to the contents hereof and specifically disclaims any implied warranties of fitness for any particular purpose. The information in this document is subject to change without notice. Optimised Control assumes no responsibility for any errors that may appear in this document.

Optimised Control Ltd
178 - 180 Hotwell Road
Bristol
BS8 4RP
U.K.

Telephone:	(+44) (117) 987 3100
Fax:	(+44) (117) 987 3101
BBS:	(+44) (117) 987 3102

IBM is a registered trademark of International Business Machines, Inc.

Revision History

Issue	Date	Reference/Comments
1	Mar 92	utils/mc/0392;M00106-001 First release

Ref: utils/mc/0392;M00106-001

1. Applications and Utilities Diskette

Supplied with every control system is a diskette containing application programs and utilities to assist you with the system. Also included are various routines, mainly written in C, to assist the implementation of host computer communications, whether using MINT/3.28 or MINT's host computer communication. The diskette contains the following:

- *cTERM: Terminal emulator configured for the controller*
- *MINT application programs*
- *C source code routines for:*

  a) File upload/download

  b) MINT/3.28 and MINT Host Computer Communications protocol
- *MINT Utilities:*

  a) MINT code squasher utility. This utility will squash MINT code enabling you to gain memory from the controller.

  b) A lines utility to print MINT programs with line numbers.

To use the utilities, you require the following:

- *IBM compatible PC with 3.5" diskette*
- *DOS version 2.0 or higher*
- *Serial communication ports COM1 or COM2*

To make the diskette easier to use, files have been put into sub-directories. The files and directories found on the disk are:

\(root)

CTERM.EXE	cTERM terminal emulator
CTERM.DEF	cTERM defaults file
DECIMAL.MNT	Example program showing use of keypad and display
KEYPAD.MNT	Example program showing use of keypad and display
KEYPAD.CFG	Keypad configuration file
JOYSTICK.MNT	XY table example program
XYPLOT.MNT	'b' plot demonstration program
XYPLOT.CFG	Configuration file for XYPLOT

\APPS

Contains all the programs discussed in the setting up and application sections of the Getting Started Guide.

ENCODER.MNT	Test encoder connections
MOTOR.MNT	Test motor polarity
FOLLERR.MNT	Test following error and gains
MOVE1.MNT	Test a positional move
CONFIG.CFG	Sample configuration file
FIRST.CFG	First MINT configuration file
FIRST.MNT	First MINT program
FEEDER.CFG	Cut to length config file
FEEDER.MNT	Cut to length program file
TEACH.MNT	XY Teach and replay program
SPOOLER.MNT	Spooler coil winding
INFEED.MNT	Product infeed
BATCH.MNT	Program showing use of batch processing

\MANUAL

Contains program examples used in the MINT Programming Guide

INTRO.MNT	Introduction MINT program
INTRO.CFG	Introduction configuration file
TEMPLATE.CFG	Template configuration file
TEMPLATE.MNT	Template program file
CHAR.MNT	Example use of character constants
NONVOL1.MNT	Template program for non-volatile variables
NONVOL2.MNT	Template program for non-volatile variables with defaults
TEACH.MNT	Teach and Learn XY example program
ARRAY.BAS	GWBasic array download example
3AXES.CFG	3 axes servo config file
CORRECT.MNT	Correct for missing steps

\COMMSON

Contains C and QBasic routines for implementing MINT Host Computer Communications on an IBM compatible PC.

COMMS.C	C source code
COMMS.BAS	QBasic source code

\MINT328

Contains C routines for implementing MINT/3.28 on an IBM compatible PC.

M328.C	MINT/3.28 C source code

\SERIAL

Turbo C routines for implementing interrupt driven serial communications.

SERIAL.C	C source code for interrupt comms
SERIAL.H	Header file for SERIAL.C

\UPDOWN

C routines for uploading and downloading files from an IBM PC.

UPDOWN.C	C source code for file upload/download
UPDOWN.H	Header file for UPDOWN.C

\UTILS

Various MINT utility programs.

LINES.EXE	Display MINT programs with line numbers
SQUASH.EXE	MINT code squasher utility
BASIC.TAB	MINT keyword table for SQUASH
MOTION.TAB	Motion keyword table for SQUASH
CONSTANT.TAB	Constant keyword table for SQUASH

2. cTERM v2.0

cTERM version 2.0 is a powerful terminal emulator program supplied specifically for use with the EuroSystem range of controllers. cTERM offers the following features:

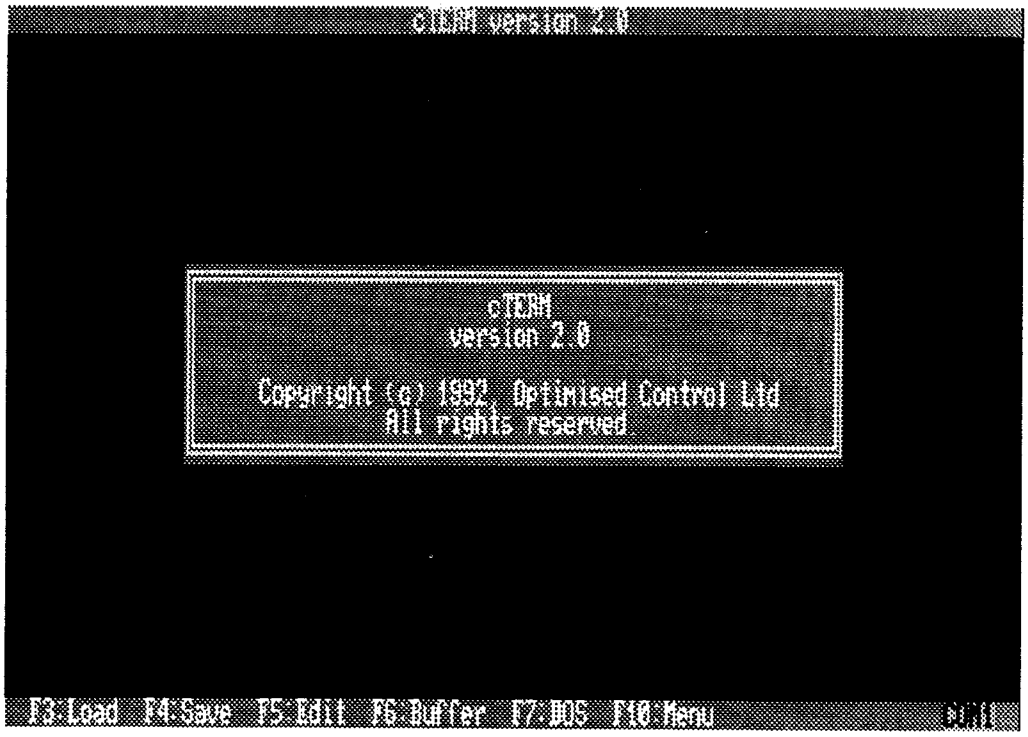
- *Automatically sets up the serial port for the controller*
- *Easy to use 'pop-down' menu structure*
- *File upload and download*
- *File upload and download from the DOS command line allowing file upload/download from within other application programs*
- *Powerful debugging facilities for controller communications*
- *Facility to call your favourite program editor from within cTERM. The editor is called with the relevant filename for editing.*

cTERM is called by simply typing:

```
CTERM
```

at the DOS prompt. cTERM can also be called with various options, see the section on cTERM Options for more details.

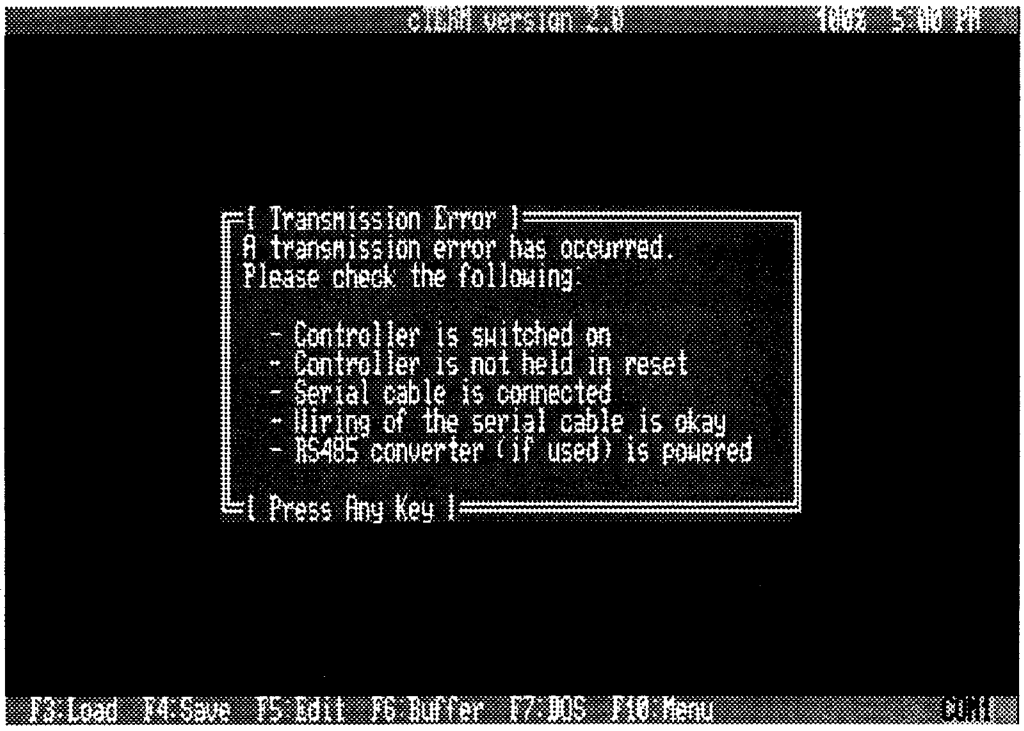
cTERM will start up with the following screen:



```
cTERM version 2.0

cTERM
version 2.0

Copyright (c) 1992, Optimised Control Ltd
All rights reserved

F3:Load  F4:Save  F5:Edit  F6:Buffer  F7:DOS  F10:Menu          COM1
```

copyrite.scr/mc

After a period of about 4 seconds, or on pressing a key, you will be taken direct to the terminal screen. Provided a controller is connected and powered up, characters typed at the keyboard should be echoed back to the terminal. If cTERM cannot communicate with the controller the following error message will be display:



```
                         cTERM version 2.0                    100% 5:00 PM

[ Transmission Error ]
A transmission error has occurred.
Please check the following:

  - Controller is switched on
  - Controller is not held in reset
  - Serial cable is connected
  - Wiring of the serial cable is okay
  - RS485 converter (if used) is powered

[ Press Any Key ]

F3:Load  F4:Save  F5:Edit  F6:Buffer  F7:DOS  F10:Menu                  COM1
```

transerr.scr/mc

2.1 cTERM Options

cTERM can be called with various options using the following syntax:

```
cTERM [options]
```

by typing:

```
CTERM /h
```

a list of the following options will be displayed:

```
/c      Do not use the CRC checking on file upload/download
/com2   Select COM2 for communications
/d      Load a defaults file
/m      Force mono attributes
/l      Download a file from the DOS prompt
/s      Upload a file from the DOS prompt
```

Example:

```
CTERM /m /com2
```

will call up cTERM in mono mode and use COM2 for serial communications. Note that cTERM can accept any number of arguments.

2.1.1 /c: No CRC

Format:

```
CTERM /c
```

cTERM on file upload and download will automatically add a Cyclic Redundancy Checksum to the end of the file to ensure correct transmission of the files. To suppress the check, use the /c option.

2.1.2 /com2: Select COM2

Format:

```
CTERM /com2
```

Using the /com2 option, all serial communications will be addressed through communications port 2 (COM2). The active communications port (COM1 or COM2) will be displayed in the top right corner of the screen.

2.1.3 /d: Load a defaults file

Format:

```
CTERM /d:<filename>
```

The defaults file is used to store the filenames used in a cTERM session. By default cTERM will load the defaults file *cTERM.DEF* which can be placed anywhere within the DOS path (see your DOS manual for an explanation of the path). If you require to load a defaults file other than cTERM.DEF, then call cTERM with the /d option as shown in the example:

```
CTERM /d:xyplot
```

will load the defaults file *xyplot.def*. Note that no extension is necessary since cTERM will use the extension .DEF for all default files.

Default files can also be loaded and saved from the Options menu.

2.1.4 /m: Mono screen

Format:

```
CTERM /m
```

The /m will force cTERM to use monochrome attributes. cTERM will automatically detect a monochrome monitor but many mono LCD displays emulate colour displays. Use the /m option to give a more acceptable display on LCD screens.

2.1.5 /l: Load (download) file

Format:

```
CTERM /l<buffer>:<filename>
```

The /l options allows files to be downloaded from the DOS command line without calling up the cTERM terminal screen.

Using the above format, <buffer> is a number relating to one of the following MINT files:

<buffer>	File
1	Program file
2	Configuration file
3	Array data

For example:

```
CTERM /l1:xyplot
```

will download the file XYPLOT.MNT to the program buffer. Note that the the following default extension is attatched to the file if no extension is given:

File	Extension
Program	.MNT
Configuration	.CFG
Array	.ARR

CTERM will display any error message and also return the error as a DOS errorlevel number. The errorlevel numbers:

errorlevel	Error
0	No error
1	Cannot open file or file not found
2	Cannot download the file
3	Timeout on serial transmission
4	Checksum error

Note:

If you require to load a file over serial communications port 2, the /com2 option *must* be places before the /l option as follows:

```
CTERM /com2 /l1:test
```

2.1.6 /s: Save (upload) file

Format:

```
CTERM /s<buffer>:<filename>
```

The /s options allows files to be uploaded from the DOS command line without calling up the cTERM terminal screen.

Using the above format, <buffer> is a number relating to one of the following MINT files:

<buffer>	File
1	Program file
2	Configuration file
3	Array data

For example:

```
CTERM /s1:xyplot
```

will upoad the program file to XYPLOT.MNT. Note that the the following default extension is attatched to the file if no extension is given:

File	Extension
Program	.MNT
Configuration	.CFG
Array	.ARR

CTERM will display any error message and also return the error as a DOS errorlevel number. The errorlevel numbers:

errorlevel	Error
0	No error
1	Cannot open file or file not found
2	Cannot download the file
3	Timeout on serial transmission
4	Checksum error

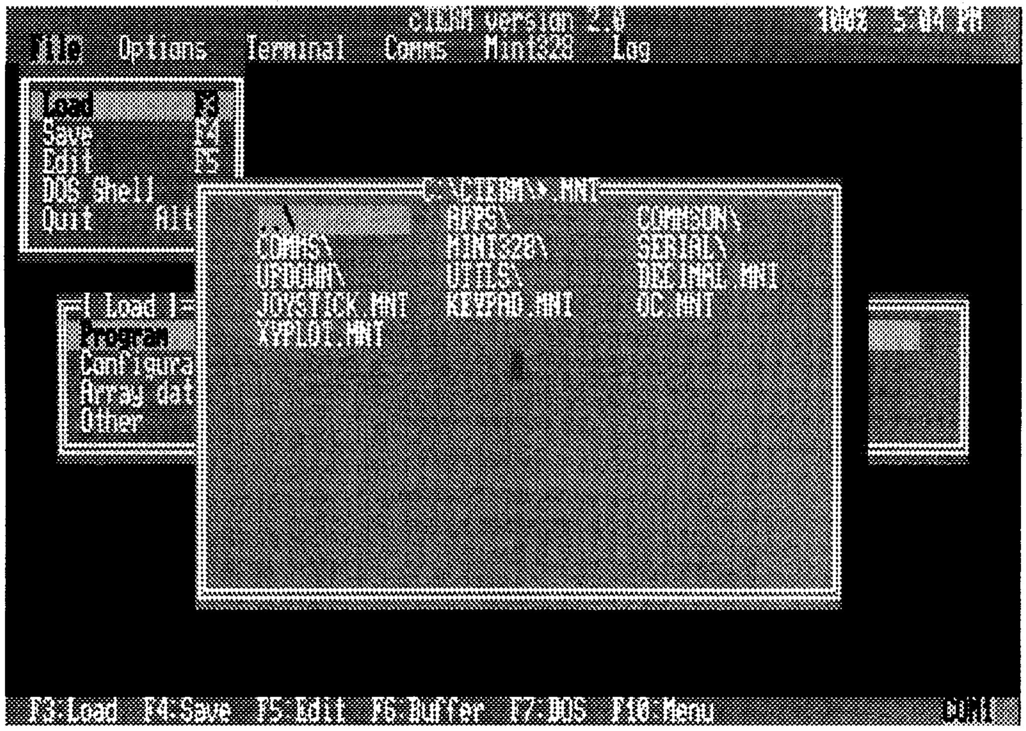
Note:

If you require to save a file over serial communications port 2, the /com2 option *must* be places before the /s option as follows:

```
CTERM /com2 /s1:test
```

2.2 Filenames

Where filenames are requested within cTERM, a list of the file in the current directory can be called up by pressing F2. For example: a program filename is requested, pressing F2 will display all files with a .MNT extension. Pressing Enter on the filename will automatically enter the name into the field and perform the required action. You can also move around a disk's directory structure. Files prefixed with '\' are directories. Pressing Enter on a directory name will move you down into the directory. To move up a directory, press Enter on the file '..\' (the parent directory).



```
C:\CTERM\*.MNT
..\            APPS\          COMMSON\
COMMS\         MINT328\       SERIAL\
UPDOWN\        UTILS\         DECIMAL.MNT
JOYSTICK.MNT   KEYPRO.MNT     OC.MNT
XYPLOT.MNT
```

dir.scr/mc

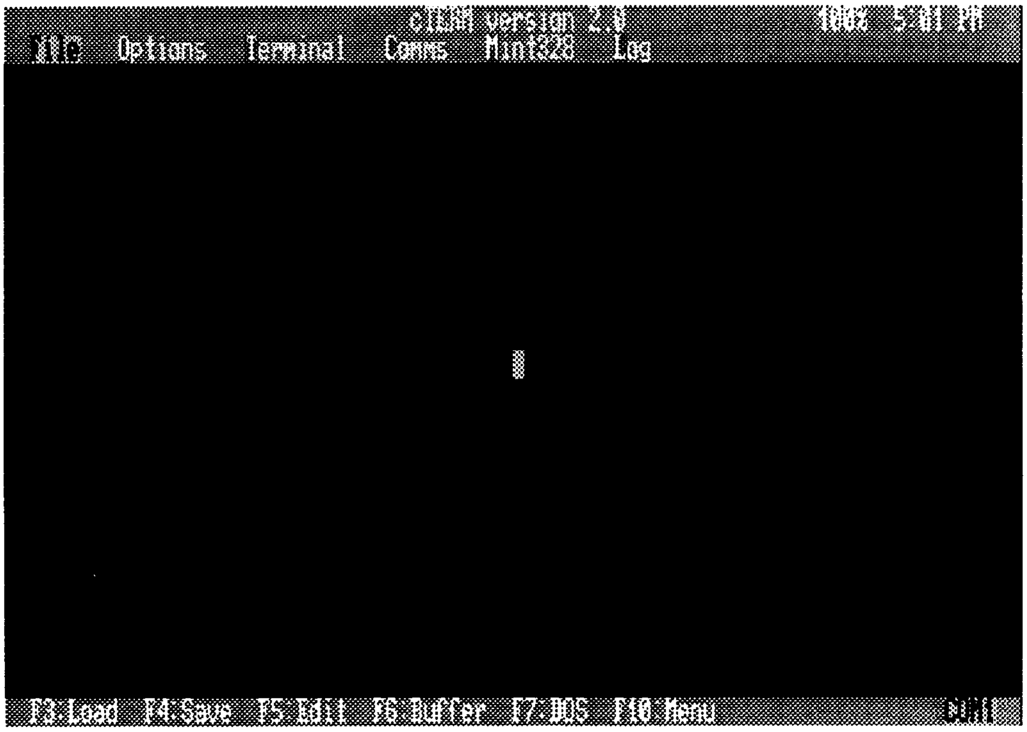
The default file extensions are:

Extension	File
.MNT	MINT Program file
.CFG	MINT Configuration file
.ARR	MINT Array data file
.CMS	Comms protocol data file
.328	MINT/3.28 data file
.DEF	cTERM defaults file
.LOG	cTERM log file

If you require a file that does not use a default file extension, press Shift-F2 instead of F2. This will display all files in the directory.

2.3 cTERM Menu Structure

Pressing F10 from the terminal screen will open up the main menu:


cTERM version 2.0
File Options Terminal Comms Mint328 Log
F3:Load F4:Save F5:Edit F6:Buffer F7:DOS F10:Menu
COM1

mainmenu.scr/mc

The menu options are:

FILE:	Allows upload and downloading of files to the controller. Files can also be edited using an external editor.
OPTIONS:	Gives you various options such as a timer, clearing the screen and loading and saving default file names.
TERMINAL:	Opens up the terminal emulator screen.
COMMS:	Gives you facilities for testing the communications protocol (COMMSON).
MINT328:	Gives you facilities for testing MINT/3.28
LOG:	Opens up a log file to record all serial communications.

The current menu item is shown by a highlight bar. To move on to the next menu item use the arrow keys or press the marked in yellow on red (or in caps). Pressing return will open up the menu. In some cases this may open up another menu or may open up a data entry screen. Pressing ESC will move you back up a menu level to the previous menu.

2.3.1 File Menu

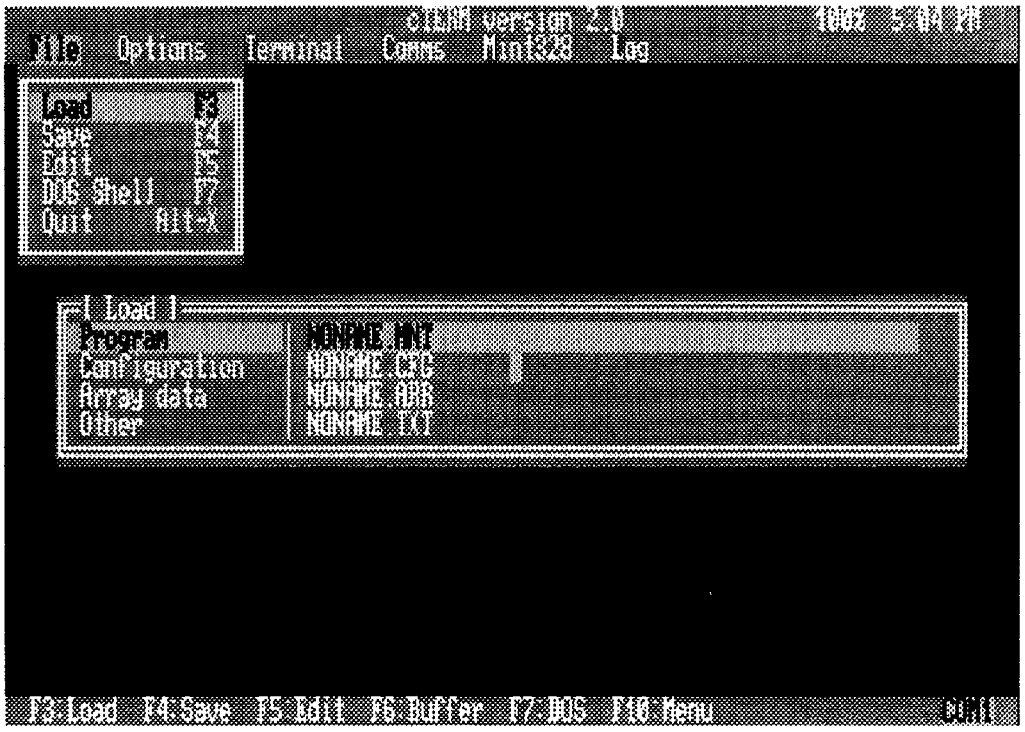
The FILE menu will allow you to upload and download program, configuration and array files as well as giving you the facility to call an external editor to edit a file. The FILE menu options are:

LOAD:	Load (download) a file into the controller.
SAVE:	Save (upload) a file from the controller.
EDIT:	Call an external editor to edit program, configuration and array files.
DOS SHELL:	Call DOS. Typing EXIT from the DOS command line will return you cTERM.
QUIT:	Quit cTERM and return to DOS.

Quick access keys are shown next to the menu option.. For example, pressing F3 will bring up the load menu.

2.3.1.1 Menu: Load/Save

The file load and save options allow you to download and upload the three files of MINT (program, configuration and array) automatically. The load and save menus are available from the terminal by pressing F3 and F4 respectively and will display:


```
                          cTERM version 2.0
 File  Options  Terminal  Comms  Mint328  Log
 Load        F3
 Save        F4
 Edit        F5
 DOS Shell   F7
 Quit     Alt-X

 [ Load ]
 Program          NONAME.MNT
 Configuration    NONAME.CFG
 Array data       NONAME.ARR
 Other            NONAME.TXT

 F3:Load  F4:Save  F5:Edit  F6:Buffer  F7:DOS  F10:Menu          COM1
```

load.scr/mc

Select the appropriate file and press return. You will now be requested to enter the filename. If a new filename is entered, the old one will be cleared otherwise it can be edited using the arrow keys. If you are not sure of the name of the file press F2 which will display a list of all the files with the appropriate extension. Pressing return on the filename will upload or download the file. Pressing Shift-F2 will display all the files in the current directory should you use a file extension other than the default.

If the file cannot be uploaded or downloaded a message "Cannot upload/download file" will be display. This may be due to one of the following reasons:

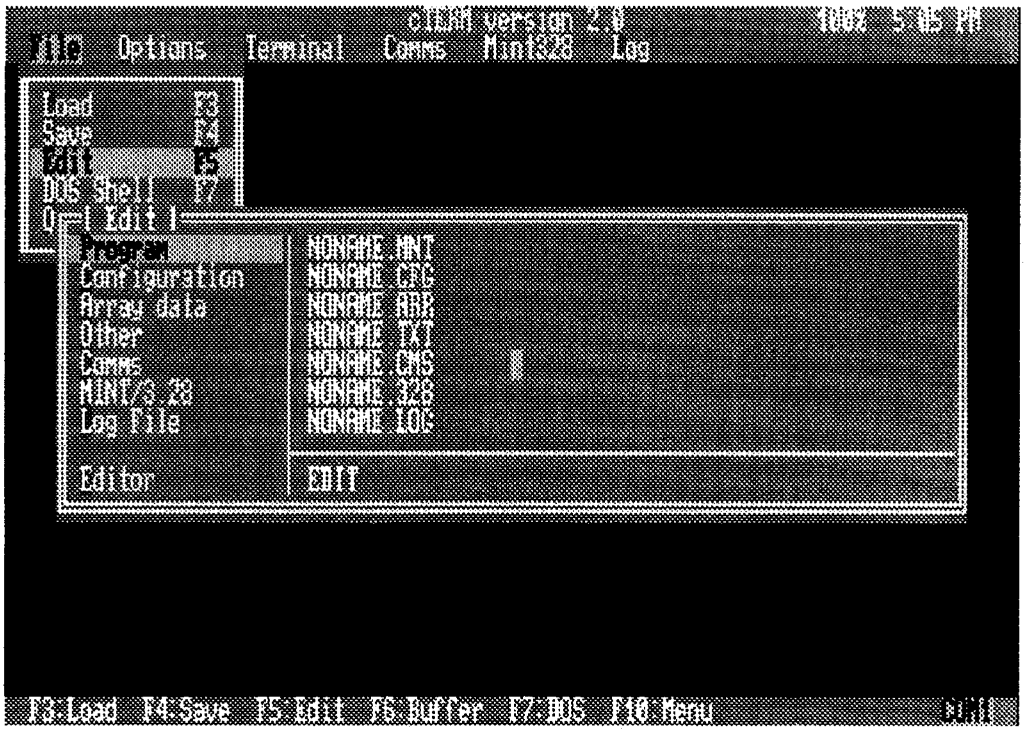
- *The serial cable is not connected*
- *The controller is switched off*
- *A program is executing, in which case abort program execution using Ctrl-E*
- *The communications protocol may be running in which case abort comms execution.*
- *RS232 to 485 convertor (if used) is not operating.*

Fix the problem and re-try. It is usually best to return to terminal mode and press Enter a few times. If a P> or C> prompt is seen, you should be okay to upload/download the file.

The options Program to Array automatically send the LOAD and SAVE commands and a CRC to the controller. If this is not required, use the "Other" option. Loading a "other" file will download the file contents only. If you request "Save Other", then on returning to the terminal screen, anything transmitted from the terminal screen will be saved to the file. This is useful for saving data from an executing program. Pressing ESC will close the file.

2.3.1.2 Menu: Edit

cTERM allows you to edit the MINT files by calling an external editor of your choice. Choosing the edit option (or pressing F5 from the terminal screen) will display:



```
                         cTERM version 2.0                    1002  5:05 PM
File  Options  Terminal  Comms  Mint328  Log

Load       F3
Save       F4
Edit       F5
DOS Shell  F7
Q[ Edit ]
   Program          NONAME.MNT
   Configuration    NONAME.CFG
   Array data       NONAME.ARR
   Other            NONAME.TXT
   Comms            NONAME.CMS
   MINT/3.28        NONAME.328
   Log file         NONAME.LOG

   Editor           EDIT

F3:Load  F4:Save  F5:Edit  F6:Buffer  F7:DOS  F10:Menu                COM1
```

editor.scr/mc

Selecting any of the file and entering a filename will call the editor with the filename given. The editor can be changed by moving to the Editor option and entering a new name. You can then save this as your default editor by using the Save Default menu option within the Options menu and selecting CTERM.DEF as the defaults file. Every time cTERM is called, the defaults file, CTERM.DEF will be loaded unless another defaults file is requested.

2.3.2 Options Menu

The options menu is:

TIMER:	Turns a timer on and off. The timer will time between successive beeps and can be used to time routines within MINT programs.
CTS:	Turns the CTS checking of cTERM on and off.
CLEAR SCREEN:	Clears the terminal screen and homes the cursor.
CLEAR BUFFER:	Clears the serial port buffer. On slow machines, cTERM may have trouble keeping up with large amounts of data over the serial port. This option will allow cTERM's internal serial buffer to be flushed.
SAVE DEFAULTS:	Save all filenames to a defaults file.
LOAD DEFAULTS:	Restore all saved filenames from a defaults file.

2.3.2.1 Menu: Timer

The cTERM timer option allows you to time MINT programs between successive beeps. For example:

```
BEEP
MOVEA = 100 : GO : PAUSE IDLE
BEEP
```

On receiving the first BEEP, the timer start, when the second BEEP is received, the timer will stop and its value in seconds.milliseconds will be displayed to the terminal screen.

2.3.2.2 Menu: Load/Save Defaults

The load and save defaults options are used to save the filenames used within cTERM to a file for future reference. The following filenames are saved to the defaults file:

Program (.MNT)
Configuration (.CFG)
Array data (.ARR)
"Other" filename
Comms protocol data file (.CMS)
MINT/3.28 data file (.328)
Log file (.LOG)
Editor filename

cTERM will load the defaults file, CTERM.DEF, when loaded from DOS. Using the /d option, another defaults file can be loaded. For example:

```
CTERM /d:keypad1
```

will load the defaults file KEYPAD1.DEF

2.3.3 Terminal

The terminal option opens up the terminal screen which allows communication with the controller. If, for some reason, there is no communication with the controller, an error message will be displayed.

There are various hot keys avaialable from the terminal window to allow file upload/download etc. The hot keys are labelled across the bottom of the screen:

F3:	Calls up the file download (load) menu. Press ESC to return control to the terminal.
F4:	Calls up the file upload (save) menu. Press ESC to return control to the terminal.
F5:	Call up the external editor menu. Control is returned to the editor menu after the editor has been called. Press ESC to return control to the terminal.
F6:	Flushes the serial port buffer.
F7	Call the DOS Shell. Type EXIT from the DOS command line to return to cTERM

Shift-F8:	Aborts MINT Host Computer Communications Protocol.
Alt-X:	Abort cTERM and return to DOS.

2.3.4 Comms Menu

The comms menu facilitates the testing of the MINT Host Computer Communications Protocol over a multi-drop line and has the following options:

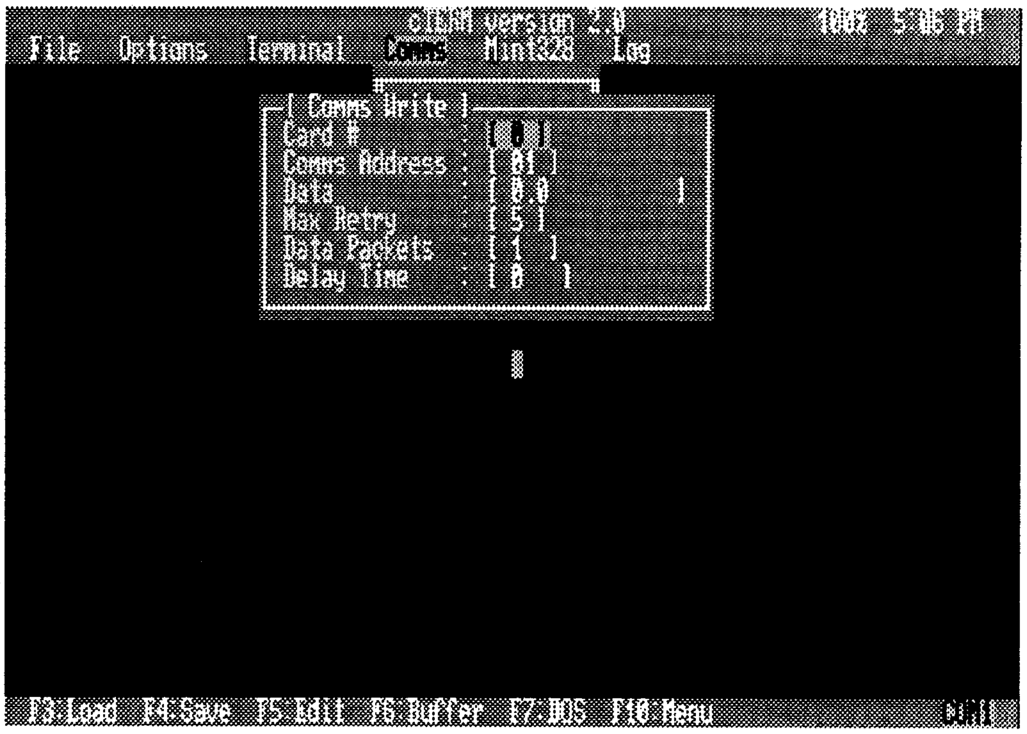
- *Write to a comms address*
- *Read a comms address*
- *Abort comms protocol*
- *Send a comms data file*

You are advised to read section 13 in the MINT Programming Guide for details on the MINT Host Computer Communications Protocol.

To aid debugging of a mutli-drop comms system, a log file can be opened to record all data going up and down the serial port. The log file is discussed in a later section.

2.3.4.1 Menu: Comms Write

A data entry window is opened to allow you to enter data to be sent to any one of 16 controllers.


cTERM version 2.0
File Options Terminal Comms Mint328 Log
F3:Load F4:Save F5:Edit F6:Buffer F7:DOS F10:Menu

```
[ Comms Write ]
Card #          : [ 0 ]
Comms Address   : [ 01 ]
Data            : [ 0.0              ]
Max Retry       : [ 5 ]
Data Packets    : [ 1  ]
Delay Time      : [ 0   ]
```

cwrite.scr/mc

The parameters are:

Card #:	Card number to send data to. Card numbers 10 to 15 are entered as A to F respectively.
Comms Address:	The communication address (pigeon hole) to send the data to. Valid address numbers are between 0 and 99.
Data:	The data to send. Any valid MINT number is allowed in this field.
Max Retry:	This determines the maximum number of retries, if data is corrupted or cannot get through, before giving up.
Data Packets:	The number of data packets to send. For example, a value of 10 will send a total of 10 data packets to the controller.
Delay Time:	Determines the delay between sending data packets where 1 represents 1/18th of a second. This allows you to see what is going on in the communications window. Pressing any key will override the delay and process the next data packet.

When the data has been entered, it is transmitted. All transmission is displayed in the Communications Window (see later).

2.3.4.2 Menu: Comms Read

A data entry window is opened to allow you to read data from a comms address from any one of 16 controllers. The field parameters are:

Card #:	Card number to send data to. Card numbers 10 to 15 are entered as A to F respectively.
Comms Address:	The communication address (pigeon hole) to send the data to. Valid address numbers are between 0 and 99.
Max Retry:	This determines the maximum number of retries, if data is corrupted or cannot get through, before giving up.
Data Packets:	The number of data packets to send. For example, a value of 10 will send a total of 10 data packets to the controller.
Delay Time:	Determines the delay between sending data packets where 1 represents 1/18th of a second. This allows you to see what is going on in the communications window. Pressing any key will override the delay and process the next data packet.

When the data has been entered, it is transmitted. All transmission is displayed in the Communications Window (see later). The result of the comms address is shown in the communications window.

2.3.4.3 Menu: Abort Comms

Before program execution can be aborted, using Ctrl-E, the communications protocol must be aborted by sending a special data packet. Using the Abort Comms menu option the required data packet will be sent to abort comms.

Note

Comms can be aborted from the terminal screen by pressing Shift-F8.

2.3.4.4 Menu: Load File

Different data packets can be created in a special format using a standard text editor. Once created, these files can be transmitted to the controllers using the Load File option. This allows files to be created that can send data to more than one controller on a mutli-drop link. The data packets are made up as follows:

A read data packet is made up as follows:

```
R;<card #>;<comms address>
```

Example:

```
R;0;10
```

will read comms address 10 on card 0.

A write data packet is made up as follows:

```
W;<card #>;<comms address>;<data>
```

Example:

```
W;1;23;1234.5
```

will write 1234.5 to comms address 23 on card 1.

A comms data file may look like:

```
: Sample file for comms
W;0;01;1234.123
R;0;01
W;0;01;4321.321
R;0;01
W;0;99;1234.123
R;0;99
W;0;99;4321.321
R;0;99
```

2.3.5 MINT328

The MINT328 menu facilitates the testing of MINT/3.28 for host computer communications over a multi-drop line and has the following options:

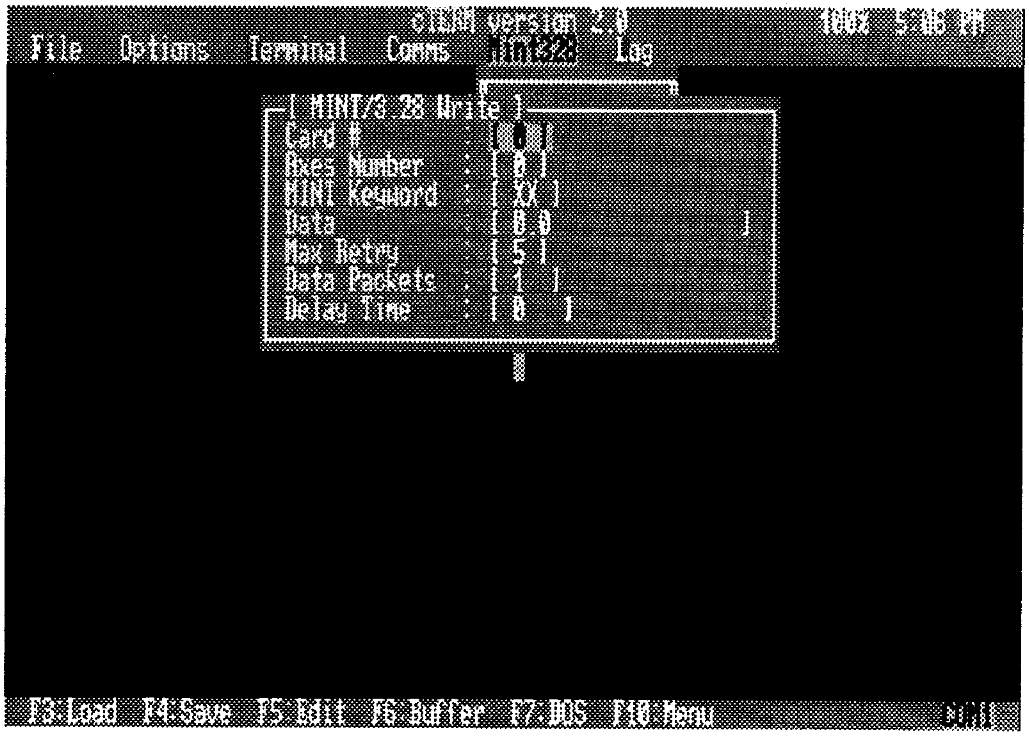
- *Write to a motion keyword*
- *Read a motion keyword*
- *Send a motion command*
- *Send a data file*

You are advised to read section 14 in the MINT Programming Guide for details on MINT/3.28.

To aid debugging of a mutli-drop comms system, a log file can be opened to record all data going up and down the serial port. The log file is discussed in a later section.

2.3.5.1 Menu: Write

A data entry window is opened to allow you to write data to be a MINT/3.28 keyword to any one of 16 controllers.


cTERM version 2.0
File Options Terminal Comms Mint328 Log
F3:Load F4:Save F5:Edit F6:Buffer F7:DOS F10:Menu COM1

```
[ MINT/3.28 Write ]
Card #         : [ 0 ]
Axes Number    : [ 0 ]
MINT Keyword   : [ XX ]
Data           : [ 0.0                    ]
Max Retry      : [ 5 ]
Data Packets   : [ 1 ]
Delay Time     : [ 0  ]
```

mwrite.scr/mc

The field parameters are:

Card #:	Card number to send data to. Card numbers 10 to 15 are entered as A to F respectively.
Keyword:	Two letter abbreviated MINT keyword.
Axis:	Axis number. This conforms to the MINT/3.28 axis number, for example an axis number of 3 represents axes 0 and 1.
Data:	The data to send. Any valid MINT number is allowed in this field. Data to more than one axis must be seperated by commas. Note that the semi-colon syntax as with MINT in not supported within MINT/3.28
Max Retry:	This determines the maximum number of retries, if data is corrupted or cannot get through, before giving up.
Data Packets:	The number of data packets to send. For example, a value of 10 will send a total of 10 data packets to the controller.
Delay Time:	Determines the delay between sending data packets where 1 represents 1/18th of a second. This allows you to see what is going on in the communications window. Pressing any key will override the delay and process the next data packet.

When the data has been entered, it is transmitted. All transmission is displayed in the Communications Window (see later).

2.3.5.2 Menu: Read

A data entry window is opened to allow you to read a MINT/3.28 keyword from one of 16 controllers. The field parameters are:

Card #:	Card number to send data to. Card numbers 10 to 15 are entered as A to F respectively.
Keyword:	Two letter abbreviated MINT keyword.
Axis:	Axis number. This conforms to the MINT/3.28 axis number, for example an axis number of 3 represents axes 0 and 1.
Max Retry:	This determines the maximum number of retries, if data is corrupted or cannot get through, before giving up.
Data Packets:	The number of data packets to send. For example, a value of 10 will send a total of 10 data packets to the controller.
Delay Time:	Determines the delay between sending data packets where 1 represents 1/18th of a second. This allows you to see what is going on in the communications window. Pressing any key will override the delay and process the next data packet.

When the data has been entered, it is transmitted. All transmission is displayed in the Communications Window (see later).

2.3.5.3 Menu: Command

A data entry window is opened to allow you to send a MINT/3.28 command to any one of 16 controllers. The field parameters are:

Card #:	Card number to send data to. Card numbers 10 to 15 are entered as A to F respectively.
Keyword:	Two letter abbreviated MINT keyword.
Axis:	Axis number. This conforms to the MINT/3.28 axis number, for example an axis number of 3 represents axes 0 and 1.
Max Retry:	This determines the maximum number of retries, if data is corrupted or cannot get through, before giving up.
Data Packets:	The number of data packets to send. For example, a value of 10 will send a total of 10 data packets to the controller.
Delay Time:	Determines the delay between sending data packets where 1 represents 1/18th of a second. This allows you to see what is going on in the communications window. Pressing any key will override the delay and process the next data packet.

When the data has been entered, it is transmitted. All transmission is displayed in the Communications Window (see later).

Menu: Load File

Different data packets can be created in a special format using a standard text editor to allow a mixture of read, write and command data packets. Once created, these files can be transmitted to the controllers using the Load File option. This allows files to be created that can send data to more than one controller on a mutli-drop link. The data packets are made up as follows:

A read data packet is made up as follows:

```
R;<card #>;<keyword>;<axis>
```

Example:

```
R;0;PS;1
```

will read the position of axis 1 on card 0.

A write data packet is made up as follows:

```
W;<card #>;<keyword>;<axes>;<data>
```

Example:

```
W;1;VR;3;100,200
```

will setup a vector relative move on axes 0 and 1 on card 1 with relative coordinates of 100,200. Note that <axes> is coded, where 3 represents axes 0 and 1. See section 14 in the MINT Programming Guide for further details.

A command data packet is made up as follows:

```
C;<card #>;<keyword>;<axes>
```

Example:

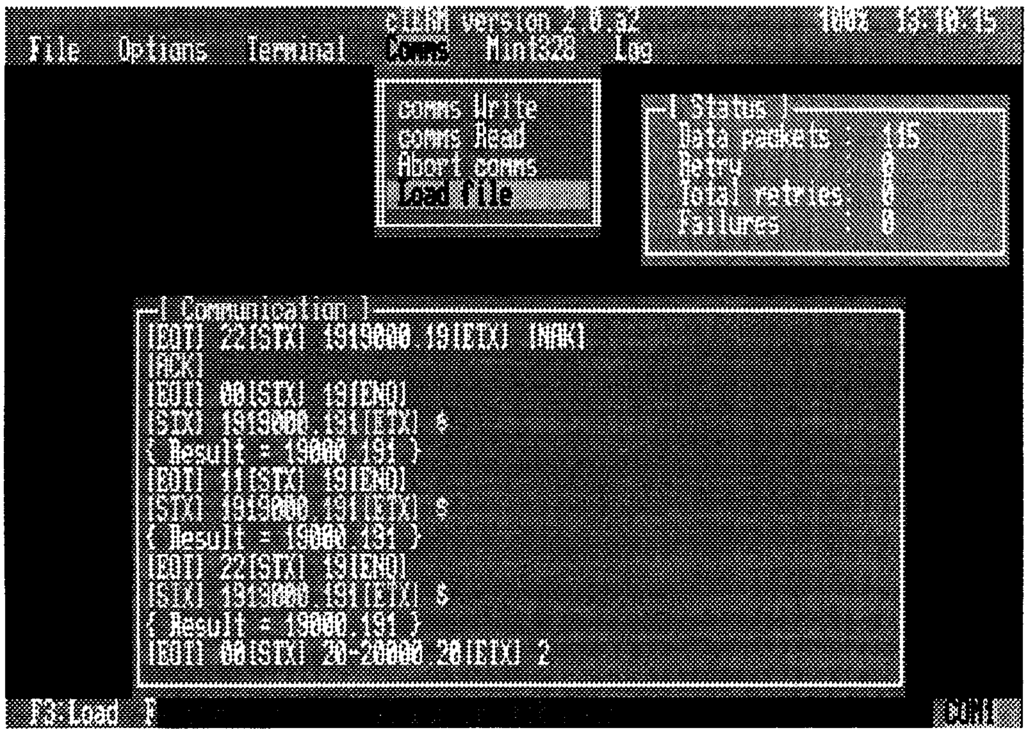
```
C;0;ST;0
```

will stop motion on card 0, axis 0

A MINT/3.28 data file may look like the following:

```
: Sample file for MINT/3.28
C;0;RE;6
: 2 axes of motion only
W;0;GN;3;2,2
W;0;KV;3;10,10
W;0;SF;3;2000,2000
W;0;SP;3;20,20
W;0;AC;3;750,750
W;0;RP;3;2.5,2.5
```

2.3.6 Communcations Window



```
[ Status ]
Data packets : 115
Retry        : 0
Total retries: 0
Failures     : 0
```

```
[ Communication ]
[EOT] 22[STX] 191900 19[ETX] [NAK]
[ACK]
[EOT] 00[STX] 19[ENQ]
[STX] 191900.191[ETX] $
{ Result = 1900.191 }
[EOT] 11[STX] 19[ENQ]
[STX] 191900.191[ETX] $
{ Result = 1900.191 }
[EOT] 22[STX] 19[ENQ]
[STX] 191900.191[ETX] $
{ Result = 1900.191 }
[EOT] 00[STX] 20-20000.20[ETX] 2
```

commwin.scr/mc

The communication window is used to display all data packet transmissions over the serial port. cTERM uses colour to represent various data packets and messages. These are:

Colour	Meaning
Yellow	Transmitted data
Cyan	Received Data
Light Grey	Contents of serial buffer on flushing
White	Comments (also enclosed in curly braces)

All ASCII characters below 32 are represented by their control names. For example, if ASCII 6 is received, this will be displayed to the screen as "[ACK]". Note that space (ASCII 32) is shown as "[SPC]" within data packets to distinguish it from spaces within messages. Spaces are used to aid clarity when reading data packets.

It should be noted that before a data packet is transmitted, the serial port buffer is flushed. To aid debugging, any flushed characters are displayed in the communication window in light grey.

The status window shows the number of data packets sent, the total number of retries and the number of retries for the current data packet.

2.4 Log Menu

The Log menu is used to open and close serial communications log files. When a log file is opened, all communications over the serial port will be recorded within this log file. For example:

The log file can be used to determine any problem areas within data communications.

Example:

```
{Log file opened March 30, 1992; 9:27 AM}
[EOT]00[STX]011234.123[ETX][CAN]
[ACK]
[EOT]00[STX]01[ENQ]
[STX]011234.121[ETX][SUB]
{ Result = 1234.121 }
[EOT]00[STX]014321.321[ETX][CAN]
[ACK]
[EOT]00[STX]01[ENQ]
[STX]014321.320[ETX][EM]
{ Result = 4321.320 }
[EOT]00[STX]991234.123[ETX][EM]
[ACK]
[EOT]00[STX]99[ENQ]
[STX]991234.121[ETX][ESC]
{ Result = 1234.121 }
[EOT]00[STX]994321.321[ETX][EM]
[ACK]
[EOT]00[STX]99[ENQ]
[STX]994321.320[ETX][CAN]
{ Result = 4321.320 }
```

All log entries in curly braces { } are comments placed by cTERM and are not data items. For example, "{ Retry }" is used to tell you that a data packet retry took place. The most common comments are:

{ Retry }	Data packet retry
{ Timeout }	Communications timed out
{ Failed }	Data packet failed to be successfully transmitted.

All ASCII characters below 32 are represented by their control names. For example, if ASCII 6 is received, this will be displayed in the log file as "[ACK]". Note that space (ASCII 32) is shown as "[SPC]" within data packets to distinguish it from spaces within messages.

The log file is closed by using the "Close log file" menu option. The log file can be viewed using the file edit facility.

3. LINES: Program Line Numbers

LINES is a utility to provide line numbers with MINT programs. This is especially useful if you require the program listing in a document and need to refer to line numbers.

LINES accepts one or two parameters, the input filename and the output filename. If no output filename is given, the program, with line numbers, will be displayed to the screen.

Example:

```
LINE myprog.mnt lines.mnt
```

will add line numbers to the file *myprog.mnt* and write the output to the file *lines.mnt.*

The file extension must be provided in both cases.

Example program:

```
REM An example program
AXES[0,1]
RESET[0,1,2]

GOSUB init
GOSUB main

END

#init
  slow = 10
  fast = 50
  esc = 'f'
RETURN

#main
 ...
```

Example program with line numbers:

```
 1> REM An example program
 2> AXES[0,1]
 3> RESET[0,1,2]
 4>
 5> GOSUB init
 6> GOSUB main
 7>
 8> END
 9>
10> #init
11>   slow = 10
12>   fast = 50
13>   esc = 'f'
14> RETURN
15>
16> #main
17>  ...
18>
```

Note that the output file is not compatible with MINT and must not be downloaded into the controller. It is only to be used as an aid to debug and document programs.

4. SQUASH: MINT Source Code Squasher

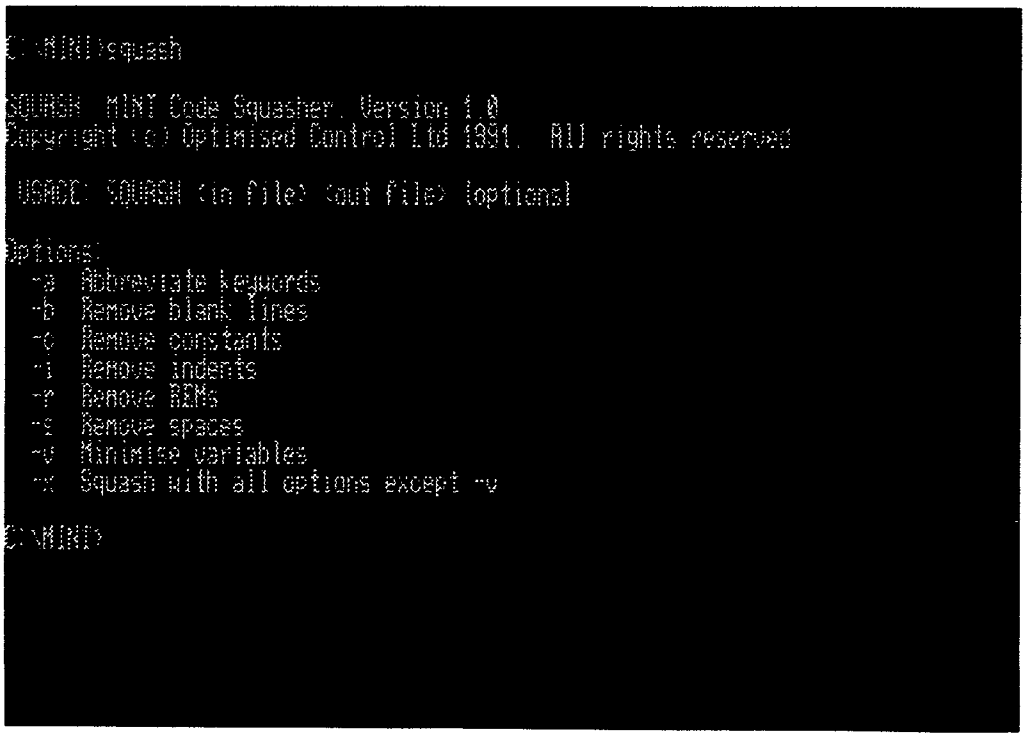
You may find that your programs are nearing the memory limit of MINT. To overcome this problem, it is often necessary to remove REM statements or remove spaces to gain more memory. However, this often leads to unreadable source code that is difficult to maintain. SQUASH is a utility that will automatically remove your REMs and spaces for you and thus increase your memory. By using SQUASH, you maintain a fully documented program, SQUASH it and run the squashed code on the controller. SQUASH will do more than just remove REMs and spaces it can also replace all keywords with their respective abbreviation, a task that would take a long time by hand. The full list of tasks that can be performed by SQUASH are:

- *Remove all REM statements.*
- *Remove all blank lines, including those left by removing a REM statement.*
- *Abbreviate all motion variables and the keywords PRINT, AND, OR and NOT.*
- *Replace constant keywords with their respective value.*
- *Remove all indents.*
- *Remove all unnecessary spaces.*
- *Minimise all user variables.*

You have complete control over the amount of squashing that takes place. Typing:

```
SQUASH
```

without parameters will display all the options as shown:



```
C:\MINT>squash

SQUASH  MINT Code Squasher. Version 1.0
Copyright (c) Optimised Control Ltd 1991.  All rights reserved

 USAGE: SQUASH <in file> <out file> [options]

Options:
  -a  Abbreviate keywords
  -b  Remove blank lines
  -c  Remove constants
  -i  Remove indents
  -r  Remove REMs
  -s  Remove spaces
  -v  Minimise variables
  -x  Squash with all options except -v

C:\MINT>
```

sq1.scr/mc

The syntax of SQUASH is:

```
SQUASH <in file> <out file> [options]
```

where <in file> is the file you wish to squash, and <out file> is the squashed file. To avoid over-writing your source code, <out file> must be a different to <in file> otherwise squashing will not take place.

The correct file extension must be provided for both the input file and the output file. For example:

```
SQUASH myprog.mnt squashed.mnt /r
```

will remove all REM statements from the program file *myprog.mnt*, and place the new file into *squashed.mnt.*

4.1 Options

SQUASH accepts option parameters to control the amount of optimisation that is done on your source code. These options are described in detail in the following sections. Please note that the following is always performed on the source code:

- *All keywords will be put into upper case and user variables into lower case.*
- *All user variables over 10 characters in length will be truncated in length to 10 characters.*

4.1.1 /a: Abbreviate keywords

Format:

```
SQUASH in.mnt out.mnt /a
```

All motion keywords and the keywords PRINT, AND, OR and NOT are replaced by their abbreviated keywords. For example:

```
WHILE IN0 AND IN2
  JOG[1] = ANALOGUE1 - 512
ENDW
```

will become:

```
WHILE I0 & I2
  JG[1] = A1 - 512
ENDW
```

The motion keywords are held in the ASCII file MOTION.TAB allowing for easy updating with new versions of MINT.

4.1.2 /b: Remove blank lines

Format:

```
SQUASH in.mnt out.mnt /b
```

Removes all blank lines. If the /r option (remove REMs) is also selected, a blank line left after taking out the REM will also be removed.

Example:

```
a = POS[1]

REM Move axis 0 to position 1
MOVEA = a : GO
```

will become:

```
a = POS[1]
REM Move axis 0 to position 1
MOVEA = a : GO
```

using the option /b/r, the above will become

```
a = POS[1]
MOVEA = a : GO
```

4.1.3 /c: Replace constants

Format:

```
SQUASH in.mnt out.mnt /c
```

All constant keywords are replaced by their respective values. For example:

```
PULSE = 1
WHILE IN0 = _on
  OFFSET = 10
  PAUSE MODE = _pulse
ENDW
```

becomes:

```
PULSE = 1
WHILE IN0 = 1
  OFFSET = 10
  PAUSE MODE = 7
ENDW
```

The constant keywords and their values are held in the ASCII file CONSTANT.TAB allowing for easy updating with new versions of MINT.

4.1.4 /i: Remove indents

Format:

```
SQUASH in.mnt out.mnt /i
```

Removes all spaces at the beginning of a line. For example:

```
LOOP
  IF IN0 DO
    MOVEA = 10 : GO
  ENDIF
ENDL
```

becomes:

```
LOOP
IF IN0 DO
MOVEA = 10 : GO
ENDIF
ENDL
```

to remove spaces within lines, use the /s option.

4.1.5 /r: Remove REMs

Format:

```
SQUASH in.mnt out.mnt /r
```

Remove all REM's from the program. If the REM leaves a blank line, this can be removed with the /b option.

Example:

```
REM Check key presses
IF key = 'A' THEN GOSUB manual : REM Manual mode
```

becomes:

```

IF key = 'A' THEN GOSUB manual
```

Note that ":REM" is removed not just the REM statement.

4.1.6 /s: Remove spaces

Format:

```
SQUASH in.mnt out.mnt /s
```

Remove all unnecessary spaces but not indents (use /i). For example:

```
WHILE POS < 100 AND POS > -100
  JOG = ANALOGUE1 - 512
ENDW
```

becomes:

```
WHILE POS<100 AND POS>-100
  JOG=ANALOGUE1-512
ENDW
```

If the /a (abbreviate keywords) option is used, this will become:

```
WHILE POS<100&POS>-100
  JOG=ANALOGUE1-512
ENDW
```

removing a further 2 spaces from the first expression.

4.1.7 /v: Minimise variables

Format:

```
SQUASH in.mnt out.mnt /a
```

All variables will be replaced by the variables 'A' to 'Z' and '_A' to '_X'. Replacement will occur in the order that the variables are found in the program file. For example:

```
DIM answer(10)
DIM xpos(20)
DIM ypos(20)

GOSUB init
GOSUB main
END

#init
  finalPos = 100
  startPos = 0
  FOR x = 1 to 20
    xpos(x) = 0 : ypos(x) = 0
  NEXT
RETURN
```

will become:

```
DIM a(10)
DIM b(20)
DIM c(20)

GOSUB init
GOSUB main
END

#init
  d = 100
  e = 0
  FOR f = 1 TO 20
    b(f) = 0 : c(f) = 0
  NEXT
RETURN
```

Caution:

Variables defined in the configuration file and used in the program file are not cross referenced. However, if the minimise variables option is selected, a cross reference file will be produced. The cross reference file for the above example will be:

```
a  = answer
b  = xpos
c  = ypos
d  = finalPos
e  = startPos
f  = x
```

The cross reference file is given the same name as the output program file but with a ".var" extension. For example:

```
SQUASH myprog.mnt myprog2.mnt /v
```

will produce a cross reference file called *myprog2.var*.

4.1.8 /x: Squash with all options

```
SQUASH in.mnt out.mnt /x
```

The program will be squashed with all options set with the exception of /v (minimise variables).

4.2 Keywords

The keywords are held in the ASCII files BASIC.TAB, MOTION.TAB and CONSTANT.TAB which relate to the Basic MINT keywords, the motion keywords and abbreviations and the constant keywords and their values. This will allow easy updating of keywords when new keywords are introduced to MINT.

The keyword files must be placed either in the DOS path or the current directory to be found.

4.3 SQUASH Examples

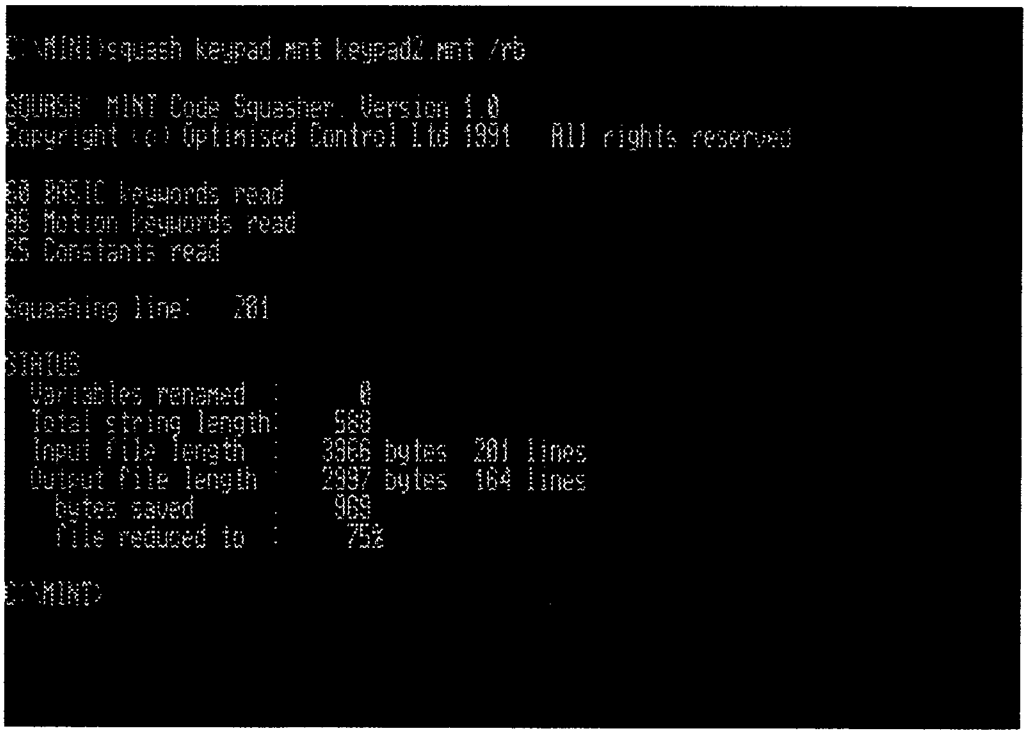
Example 1:

```
SQUASH keypad.mnt keypad2.mnt /rb
```

will remove all REMs and blank lines from the file *keypad.mnt* to file *keypad2.mnt*

Note that an input filename and an output file name must be given. If the output filename is the same as the input, a message will be displayed.

Once complete, status information will be given on the conversion as shown:



```
C:\MINT>squash keypad.mnt keypad2.mnt /rb

SQUASH: MINT Code Squasher. Version 1.0
Copyright (c) Optimised Control Ltd 1991   All rights reserved

80 BASIC keywords read
96 Motion keywords read
25 Constants read

Squashing line:   201

STATUS
  Variables renamed  :     0
  Total string length:   560
  Input file length  :  3366 bytes  201 lines
  Output file length :  2397 bytes  164 lines
    bytes saved      :   969
    file reduced to  :    75%

C:\MINT>
```

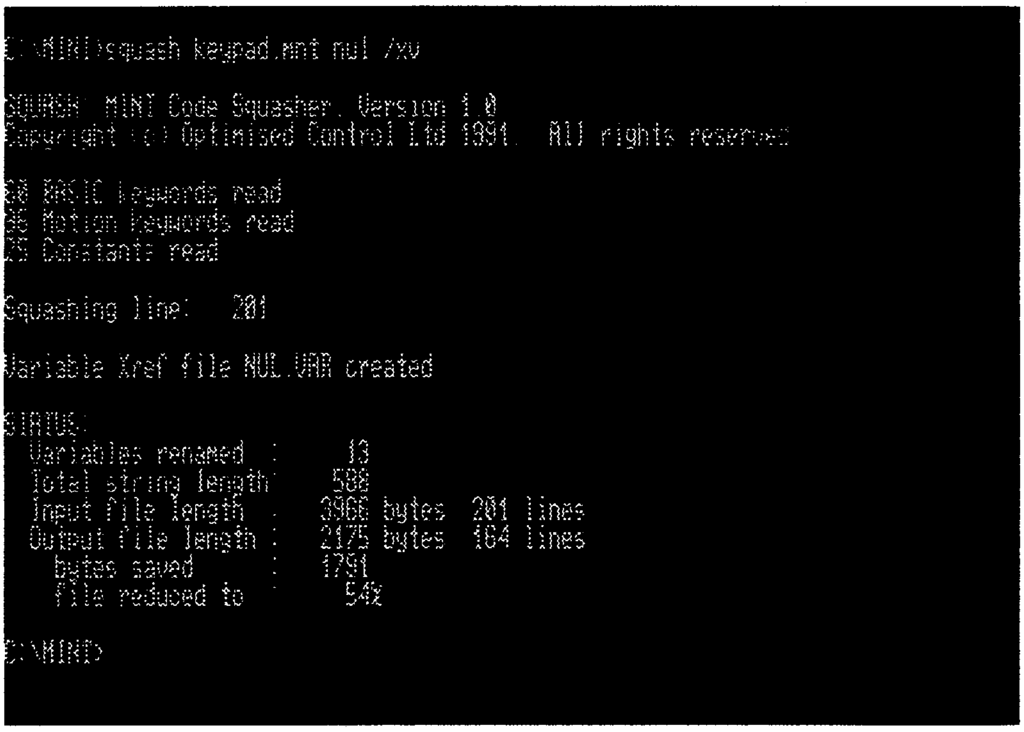
sq2.scr/mc

The field, "Length of strings", shows how much memory is taken up by strings. This can prove useful in optimising their use.

Example 2:

```
SQUASH keypad.mnt nul /xv
```

show only the status after performing a complete optimisation including minimising variables. Note that making the output file NUL, the output is effectively lost.



```
C:\MINT>squash keypad.mnt nul /xv

SQUASH  MINT Code Squasher. Version 1.0
Copyright (c) Optimised Control Ltd 1991.  All rights reserved

38 BASIC Keywords read
36 Motion Keywords read
25 Constants read

Squashing line:   201

Variable Xref file NUL.VAR created

STATUS:
  Variables renamed  :     13
  Total string length:    588
  Input file length  :   3966 bytes  201 lines
  Output file length :   2175 bytes  164 lines
    bytes saved      :   1791
    file reduced to  :     54%

C:\MINT>
```

sq3.scr/mc

Applications and Utilities Diskette: Addendum

cTERM Version 2.03
steTERM Version 2.03

Issue 1

Ref: ct1/mc/0593; MN00132-001

cTERM and steTERM Updates

Support is provided for HPGL files:

- Since HPGL does not echo its received characters, cTERM has the option of switching on local echo.
- File Download has the option of downloading HPGL files. This takes into account hardware handshaking. The default file extension used is .PLT.

- Directories and drives can be changed from within the File menu.
- cTERM would time out when downloading large files which would result in a "File Corruption" error.
- Options selected from the Options menu are now saved when "save Defaults" is selected.

HPGL Support

cTERM now provides the facility for local echo of characters entered at the keyboard. Selecting Local Echo to ON in the Options menu will turn on this facility. The options menu has been updated to:

`TIMER`	`OFF`	Turn timer on and off
`CTS`	`ON`	Turn off CTS (hardware handshaking)
`Local echo`	`OFF`	Turn on local each. Keypresses are shown on the terminal
`VT100`	`ON`	VT100 emulation off
`clear Screen`		Clear the terminal screen
`clear Buffer`		Clear the serial buffer
`save Defaults`		Save files names and options to the defaults file
`Load defaults`		Load files names and options from a defaults file

When using HPGL, Local echo should be turned on and VT100 turned off.

steTERM

steTERM provides a terminal emulator over the STE bus in the same way as cTERM does over a serial link. steTERM provides all the facilities found in cTERM with the following differences.

- Support for the Dual Port RAM (DPR). The DPR contents are shown in a window at the top of the terminal screen.
- Facility to select a control card over the STE bus. This is done by pressing F9 and pressing return over the required control card.
- CTS on/off is not supported in the Options menu.

Dual Port RAM

The DPR contents are shown in a window above the terminal screen. This window can be closed by selecting "Display DPR" in the STE menu.

Various function keys are used to control the function of the DPR:

Function Key	Function
F8	Moves through the DPR pages 0 to 2
Ctrl-F8	Allows selective DPR contents to be written to.
F9	Pops up a menu for selecting a controller
Ctrl-F9	Toggles the DPR page lock. Pressing Ctrl-F9 will lock the DPR contects until Ctrl-F9 is pressed again.

Function Key Reference

Function Key	Function
F2	Gives a directory listing when requested for a file
F3	Call up file download menu
F4	Call up file upload menu
F5	Call up file edit menu
Shift-F5	Clear the terminal screen
F6	Clear the serial port buffer
F7	Shell out to the DOS prompt
F8	Moves through the DPR pages 0 to 2
Ctrl-F8	Allows selective DPR contents to be written to.
F9	Pops up a menu for selecting a controller
Ctrl-F9	Toggles the DPR page lock. Pressing Ctrl-F9 will lock the DPR contects until Ctrl-F9 is pressed again.
F10	Call up the main menu