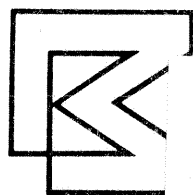
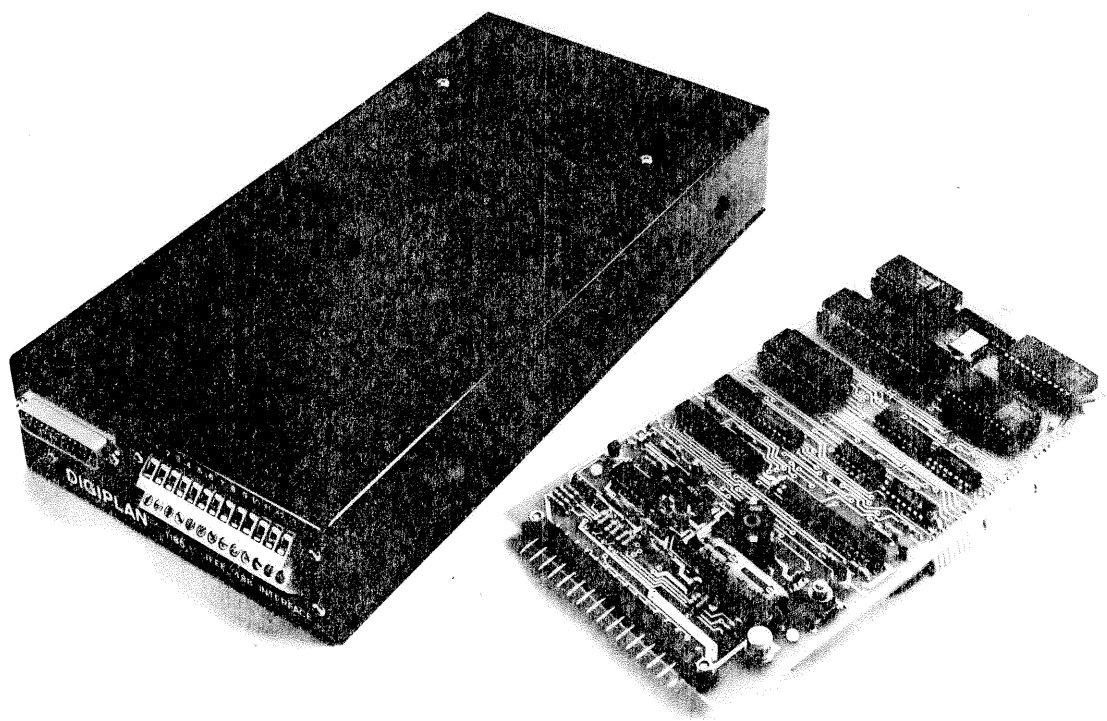


D I G I P L A N

1185 IEEE 488 INTERFACE

TECHNICAL SPECIFICATION



**DIGIPLAN LIMITED**

21/22 Balena Close, Creekmoor  
Poole, Dorset, BH17 7DX.  
Telephone: Broadstone (0202) 690911 Telex 417217

D I G I P L A N

1185 IEEE 488 INTERFACE

TECHNICAL SPECIFICATION

TYPE 1185-105

INTERFACE TYPES

1185-105 0-25000 steps/sec multiplex mode.  
0-5000 steps/sec in linear interpolation mode.

## INTRODUCTION

The 1185 Processor interface enables one or more stepper motor drives to be controlled directly from any processor such as the Commodore PET having an IEEE/IEC/HPIB interface bus. The facilities available include indexing and speed control as well as run/stop operation, with programmable acceleration and deceleration. Up to 32 motors may be controlled from a single data bus, with individual interface units being accessed by the controller using an address set up on switches within the interface.

The standard interface incorporates the Intel 8039 microprocessor and is supplied as a two-layer Euro-compatible PC card. Power and drive signal connections are made via a 32-way connector to DIN 41612 (Type D). A 26-way ribbon cable connector is used for the bus connection, allowing economic 'daisy chaining' in a multi-axis, rack mounted system.

Up to three stepper drives may be operated from a single interface in the multiplexed or time-shared mode. In this mode each axis may be driven individually, or up to three axes may be made to perform the same operation simultaneously.

Linear interpolation may be performed on two axis X and Y at a speed of up to 5000 s/s by supplying major and minor axis movements.

An optional mounting box is available which will accommodate the interface as a free-standing unit or it may be fitted underneath a Digiplan 1054 drive. In this case the power and signal connections appear on screw terminals and the bus connection is made by a standard industry-compatible D-connector. Alternative mounting arrangements will be made available later.

### PROGRAMMABLE CONTROL FACILITIES

Indexing - incremental indexing programmed in motor steps using either BCD or hexadecimal coding, 6 decades.

Speed Control - speed programmed in steps/sec using BCD or hexadecimal coding, 5 decades.

Acceleration and deceleration - linear acceleration and deceleration ramps generated automatically, with 16 alternative slopes.

Run/Stop operation - the motor may be run at a programmed speed, stopped as required or stopped on the next zero-phase position of the translator. The speed may be changed whilst the motor is running.

### Programmed Boost And Energise

Cancel facility - used to terminate a run, remove boost, de-energise or abort an index.

Stall detection - may be used on one selected axis in the multiplexed mode, or on any axis in the simultaneous mode.

Linear interpolation - at programmed speeds up to 5000 steps/sec on X and Y axis.

Emergency stop - hardwired connection which halts all axes driven by the interface, independently of the controller.

Dimensions -: Standard interface card 112mm wide, 220mm long, overall height 35mm.

Mounting box 292mm long, 152mm wide, 52mm deep. Suitable for fixing directly to underside of 1054 drive and includes separate mounting bushes in the base and on one side.

## 4. INDEXING

### 4.1 Increment

The interface module accepts incremental axis information programmed in motor steps. A board-mounted selector switch 'C' informs the interface of the data format being used.

C = 1 - BCD 6 digits

C = 0 - Hexadecimal 4 characters

The data arrives as ASCII characters 0-9 (0-F) with its most significant character first.

Leading zeros are not required. If no increment is programmed in a block requesting an index then the previous index increment applied to the axis will be repeated.

### 4.2 Direction

The direction in which the increment is applied will be positive unless an ASCII minus character is sent after the axis code. The plus sign is assumed but may be programmed if desired. In the case of a multiplexed interface it is possible to apply the same movement to any or all of the axes in different directions. (e.g. XY-325 applies 325 steps in a + direction on X and a - direction on Y at the same time, using the same acceleration rate).

## 5. SPEED

### 5.1 Speed

The speed information is always preceded by an ASCII '@' character and the rate data is programmed in steps per second in either BCD or Hexadecimal as defined by switch C - (see 'The increment', section 4.1).

C = 1 - BCD - 5 digits

C = 0 - Hexadecimal - 4 characters

Leading zeros are not required.

On initialization a speed of 400 s/s is stored and will be remembered unless overwritten by a new rate. In general it is only necessary to program a new rate if a change in speed of that axis is required. Note the speed information is used both in the index and run modes.

RATE CONTROL ACCURACY

Module type 1185-105 : 0 - 400 s/s                   + 1.7%  
                          400 s/s - 25000 s/s        <-1%

Linear interpolation mode : 400 s/s - 5000 s/s   + 1.7%

Speeds from 0 to 400 s/s are not accelerated and Bit 3 of the status Byte section 14.2 will not be set.

The speed of any axis in motion may be changed at any time by specifying the axis and new rate only. While the axis is responding by accelerating or decelerating, Bit 2 of the status work (section 14.2) will not be set. When the correct speed is reached Bit 2 will be set.

6. ACCELERATION & DECELERATION

The standard interface produces linear acceleration and deceleration ramps which have the same slope. The interface user can however select a suitable slope for the application for each axis and program that into the interface. As with other data the slope will be remembered until overwritten. The system is initialized with the longest acceleration/deceleration time.

The ASCII 'A' character is used to precede the slope number which is a single digit in the range 0-F; the smaller the number the longer the acceleration/deceleration time.

The following gives the approximate values for the acceleration rates.

<u>Programme Number</u>	<u>Steps/Second</u>
0	7000
1	10040
2	13030
3	18560
4	24500
5	33100
6	43750
7	61250
8	81666
9	111360
A	153125
B	204166
C	306250
D	408333
E	612500
F	1225000

In a time shared application the interface is used by up to three axes, so unless it is critical to system performance it is recommended to choose 'A' to suit the worst axis. The programmer can however change to a higher 'A' number on the lighter axes if they are used singly.

#### Examples of Indexing Instructions

1. (MLA) X-10 @ 3 g - Index X axis -ve 10 steps at 3 steps/sec.
2. (MLA) X Y 100 @ 50 g - Index X & Y axes + 100 steps at 50 steps/sec.
3. (MLA) X-Z 100 @ 20 g - Index X -ve & Z +ve 100 steps at 20 steps/sec.
4. (MLA) X Y @ 5 g - Index X & Y +ve the previous increment at 5 steps/sec.
5. (MLA) X Y Z - g - Index X & Y +ve and Z -ve the previous increment at the previous rate.

6. (MLA) X 0A6 § - Load interface with new acceleration rate 6 and zero index.
7. (MLA) XA3 § - Index X +ve the previous index at the previous rate with a new acceleration rate of 3.

Axes can be stopped by the use of the cancel code. Refer to section 11.

At the end of index the motion bit of the status work goes low and a Service Request is raised on the controller. See section 14.

### 7. RUNNING

Any axis can be started in the run mode by sending the run, ASCII 'R' after the axis code. The axis will run at the previously loaded rate unless a new rate is programmed after the run code. The acceleration/ deceleration rate can be changed by inserting a new 'A' code after the axis code.

1. (MLA) X R @ 50 § - Start running on the X axis in the +ve direction at 50 s/s.
2. (MLA) X Y-R @ 20 § - Start running on X +ve & Y -ve at 20 s/s.
3. (MLA) X-ZR § - Start running on X -ve and Z +ve at the previous rate.
4. (MLA) XYZR A 7 § - Start running on X, Y & Z +ve at the previous rate with a new acceleration rate of 7.
5. (MLA) X R @ 25000 § - Run axis +ve at 25000 s/s.
6. (MLA) X R @ 5000 § - Change speed X axis to 5000 s/s.
7. (MLA)\*§ - Stop axis

#### Note

Ex 5,6,7 are consecutive instructions.

Ex 1,2,3 & 4 can be terminated with (MLA)\*§ see section 11.

Axes can be stopped by the use of cancel codes. Refer to section 11. There is also a special terminate code (section 8) which stops the axis on zero phase of the translator, the state in which the stepper is energised at power on. See Drive Manual.

### 8. RUN STOP (ZERO PHASE)

In systems that require a mechanical datum it is recommended that this is accomplished by the controller using the 'run' command. A table switch is used to determine upon which revolution of the stepper the datum position lies. A disc switch on the motor shaft then has to indicate on which 1/50th part of the revolution the datum lies, and the final datum position is given by 'zero phase' of the translator logic. This method combined with a suitable sequence control will give a precise datum to within one motor step which can be re-established when power is removed and returned to the unit.

The run stop code 'ASCl '\*' is programmed after the appropriate axis or axes command which upon receipt will cause the motor to stop at the next zero phase state of the translator. If the axes have not previously been requested to RUN, then they will move in the direction received after the axes code to the next zero phase position.

1. (MLA) XZ \* § - Stop X & Z on their next zero phase position.
2. (MLA) XZ - \* § - Stop X & Z in neg. dir. on next zero phase positions.

When the axis actually comes to rest, the motion status bit is set low and a Service Request is made on the controller. See section 14. If zero phase is not detected after 11 pulses then a fault will be indicated see section 14.2.

### 9. ENERGISE

An axis or axes may be energised by sending the ASCII 'P' code (Power up). All axes are automatically energised on Power On. A drive must NOT be energised or de-energised while axes are in motion.

Examples --:

1. (MLA) XP5 § - Energise all axes and index X 5 steps only.
2. (MLA) XYZ P § - Energise all axes and index X, Y, Z the previous amount.

3. (MLA) P - Energise all axes connected.

Note

The 'P' code affects all axes associated with that interface. To de-energise axes refer to section 11. (This only applies to the axes connected).

10. BOOST

The torque of the stepper motor can be increased by the application of Boost.

An axis or axes may be used with Boost by sending the ASCII '>' code. It is only recommended that boost be used intermittently, for example during acceleration.

Examples -:

1. (MLA) XP > § - Energise and apply Boost to all multiplexed axes and index X only.

Note

The '>' affects all axes associated with the interface only if they are connected.

11. CANCEL FUNCTION

The cancel function has a number of uses. These are -:

1. Terminate Index
2. Terminate Run
3. Terminate Energise
4. Terminate Boost

To terminate Motion i.e. 1 & 2 above, all that is necessary is to program the ASCII '✖' code after the axis code, followed by ASCII '§'.

To end a latch function i.e. 3 & 4 it is necessary to follow the '✖' code with the function code. In the case of Energise, if motion is present it will be automatically cancelled.

To cancel the automatic boost facility send '✖>'.

Examples

1. (MLA) ✖ § - Stop motion on all axes.

- 2. (MLA)\*P - De-energise all axes.
- 3. (MLA)\*> - Remove boost on all axes and auto boost.

Note

If a cancel function is used whilst axes are in motion then data indicating the distance moved from the start of motion is available in the interface. See section 15.

In case 1 where motion is being arrested by the '\*' code, the motion bit in the status word will go low and a Service Request will be made on the controller. See section 14.

## 12. MOTION DETECTOR

A simple detector feature has been incorporated in the design. To use the facility it is necessary to connect the axis clock output to the stall axis clock input.

The axis using the facility also requires a toothed wheel and suitable detector fitted to the motor output shaft. The number of teeth on the wheel is defined by the translator used thus -:

<u>Translator</u> No. of steps/revolution	<u>Teeth on wheel</u>
200	4
400	8
600	12
800	16
1000	20

We recommend an aluminium wheel 3 inches in diameter with the teeth spaced equally giving equal tooth and gap. A metal proximity detector with a sensing area of about 8mm diameter and a band-width of 1 KHz is suitable.

(e.g. Pepperl & Fuchs NJ1-8G ME 24V NPN Output).

The stall detector is a simple way of detecting the stall condition of a stepper motor. It will not however detect the loss or gain of a few motor steps. It operates by counting step pulses to the drive between leading edges of pulses derived from the detector looking at the toothed wheel. If more than 100 motor pulses are counted a stall fault is produced, setting a status bit and raising a Service Request on the controller. Refer to Section 14.

### NOTE

1. Two - lead proximity detectors cannot be used unless external logic is added.
2. If the facility is not used then the stall axis clock input should be grounded to OV.

## 13. EMERGENCY STOP

Each interface module has an input, quite separate to the IEEE Bus which is required to be switched by a normally closed switch to the +24 volt supply if the system is allowed to run. If this line goes open circuit whilst axes

are in motion then the axes associated with that interface will decelerate and stop, a status bit will be set and a Service Request will be made. It is intended that this facility will be used for Safety Stop and extreme limit arrest. Normal limit switch operation should stop the axes via the controller using the cancel code.

13.2

EXTERNAL STALL FAULT

If this input is grounded the clocks will instantly stop and an SRQ sent to the controller with the stall fault bit set in fault status. The input cannot be disabled by the IEEE Bus. This input is intended to be used in multiplexed systems in conjunction with the PKS 255 stall detector card, when stall detection is needed on more than one axis. This input should be left unconnected if this facility is not used.

13.3

EXTERNAL DRIVE FAULT

If any of the fault inputs are open circuited or allowed to go a logic high, the drive clocks will instantly stop and an SRQ will be sent to the controller with the drive fault bit set in the fault status. The fault can only be cleared by removal of power to the drive. Unused drive inputs should be taken to OV.

14.2

STATUS REQUEST

Where the controller requires to know more detailed information about an interface the Status Request method should be used. Two complete status bytes are then available for each axis.

To obtain a status byte for a particular axis the controller must first send out the primary address of the axes (MTA) and follow it by one of the following secondary command group (SCG) codes. (See overleaf).

ATN	DATA I/O BITS								COMMAND
	8	7	6	5	4	3	2	1	
* 1	X	1	1	0	0	0	0	0	Multiplex Mode Status
* 1	X	1	1	0	0	1	0	0	Multiplex Mode Fault Status

NOTES

- \* logic levels to IEEE convention i.e. logic 1 =  $\leq 0.8V$   
0 =  $\geq 2.0V$

The appropriate interface, i.e. the one which has been selected by the SCG code will respond by sending back its status byte along with ASCII 'CR' and 'LF'. The control should on terminating a read send the general untalk command.

CR - Carriage Return  
LF - Line Feed

The axis status is interpreted as follows -:

<u>BIT NO.</u>	<u>BIT ALLOCATION</u>
1	Always set
2	Instruction complete
3	Acceleration complete
4	Motion
5	Zero Phase X
6	Zero Phase Y
7	Zero Phase Z
8	Parity

The fault status byte is interpreted as follows -:

<u>BIT NO.</u>	<u>BIT ALLOCATION</u>
1	Data Fault & Zero Phase Fault
2	Emergency stop
3	Drive Fault
4	Stall Fault
5	Always set
6	Not used
7	SRQ
8	Parity

NOTES

- Logic level to IEEE Convention i.e. logic 1 = 0.8V  
2 = 2.0V

The IEEE bit lines are low (0.8V) when a signal is present.

2. Stall fault, data fault and emergency stop will be automatically reset, removing the appropriate fault bit, after a status request has been made on an axis with one of the above faults.
3. Drive fault can only be cleared by removing power to the drive and re-applying it. The interface clear (IFC) signal must then be sent to all interfaces.
4. If any fault is present then the led on PC 1185-005 will be lit if motion is attempted.

### 15. AXIS POSITION REQUEST

In some applications, for example after an emergency stop, it is necessary for the controller to know how much movement took place before the axis was stopped.

By making an Axis Position Request the controller can obtain back from the interface the number of steps taken by the axis since it was last started. The data is returned in ASCII HEX code as a five character message, followed by characters ASCII 'CR' and 'LF'.

The data returned must be subtracted from FEFF0 to obtain the correct distance moved in the run mode.

To make an Axis Position Request the controller sends out the talk address (MTA) followed by one of the following secondary command group (SCG) codes.

ATN	DATA I/O BITS								Command
	8	7	6	5	4	3	2	1	
* 1	X	1	1	0	1	0	0	0	Multiplex Mode

#### NOTE

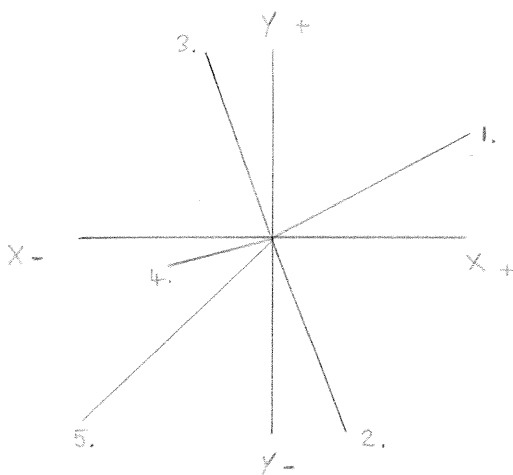
Logic levels to IEEE Convention i.e. logic 1  $\leq$  0.8V  
0  $\geq$  2.0V

The appropriate interface i.e. the one selected by the SCG code will respond by sending out five Hex characters, 'CR' and 'LF' in ASCII. For speeds > start/stop speed, axis when stopped with a '\*' code will stop and return a value for distance moved as a multiple of 16 steps.

17. LINEAR INTERPOLATION

In this mode the interface will control two axes X and Y simultaneously by generating a major axis clock and deriving a minor axis clock from it. Both axes will take the same time to complete their moves. Data must be supplied as two increments programmed in motor steps. The first increment must be that for the major axis, the second is for the minor axis. Only X and Y clock outputs can be used, so the major axis output need only be specified. Directions are assumed positive unless specified.

Examples For Linear Moves.



- |    |                  |   |
|----|------------------|---|
| 1. | X 400 L 200 \$   | - Major axis X 400 steps, minor axis Y 200 steps at previously programmed rate. |
| 2. | Y -400 L 150 \$  | - Major axis Y, negative 400 steps, minor axis X positive 150 steps.            |
| 3. | Y 400 L -150 \$  | - Major positive Y axis, minor negative X axis.                                 |
| 4. | X -200 L -80 \$  | - Major negative X axis, minor negative Y axis.                                 |
| 5. | X -400 L -400 \$ | - X and Y negative 400 steps (45°)  |

If different speeds are required in the above examples then this may be programmed from 0 - 5000 steps/sec.

Note

Maximum programmed movement of the major and minor axes in the linear interpolation mode is 32000 steps.

Examples Of Above Moves At Different Speeds

1. X 400 L 200 @ 5000 \$ - As before at 5000 steps/sec.
2. Y -400 L 150 @ 1000 \$ - As before at 1000 steps/sec.
3. Y 400 L -150 @ 100 \$ - As before at 100 steps/sec.
4. X -200 L -80 @ 20 \$ - As before at 20 steps/sec.

The Z axis may also be used but only as a major axis.

1. XZ 400 L 200 \$ - X and Z major axis positive,  
Y minor axis positive
2. YZ- 600 L 200 \$ - Y positive Z negative major axes,  
X minor axis positive

Programmed moves are stored in the interface, so that for repeat moves data does not need to be repeated.

Following Examples Must Be Consecutive

1. X 400 L 200 \$ - X 400 steps, Y 200 steps at previously programmed rate.
2. X L \$ - As above.
3. X- L - \$ - X negative 400, Y negative 200 i.e. will retrace previous move.
4. Y L - @ 100 - Y positive 400, X negative 200 at new rate of 100 steps/sec.

Relative speeds of X and Y clocks may be generated using the linear interpolator in the RUN mode (electronic gear box). This can be specified by inserting an 'R' at the end of the data.

Example

- (a) X 400 L 200 @ 1000 R \$ - Will run the X axis at 1000 steps/sec and the Y axis at 500 steps/sec.

The same formula for the direction of major and minor axes applies as before. Sending the '\*\$' command will cancel a run command for example -:

- (a) \*\$ - Stop axes motion.

The speed may also be changed while the axes are in motion by sending the major axis followed by the new speed and 'R'.

Following Examples Must Be Consecutive

1. X -400 L 200 @ 4000 R \$ - Will run X axis negative at 4000 steps/sec and Y axis positive at 2000 steps/sec.
2. X @ 800 R \$ - Change X axis speed to 800 steps/sec and Y will correspondingly change to 400 steps/sec.
3. \*\$ - Stop motion on all axes.

Note

When changing speed in the run mode it is vital that the speed boundary at 400 s/s is not crossed. If it is necessary to cross this boundary then the axes must be stopped and re-started at the new speed.

1. X 400 L 200 @ 200 R \$ )
  2. \*\$ )
  3. X L @ 500 R \$ )
- Change speed from 200 to 500 steps/sec.

Similarly

1. X 400 L 200 @ 5000 R \$ )
  2. X @ 401 R \$ )
  3. \*\$ )
  4. X L @ 100 R \$ )
- Change speed from 5000 steps/sec to 401 steps/sec to 100 steps/sec.

When in the linear interpolation mode position information can be obtained as in section 15. The following changes must however be made.

The information returned to the controller from the interface will be that of the major axis and will be 5 bytes of data the last byte being zero. This last byte should be ignored and the first 4 bytes only used.

Example

If the interface returned the following 5 bytes of data 01900 this is in the hex and ignoring the last byte will become 0190 or converted to decimal 400 steps.

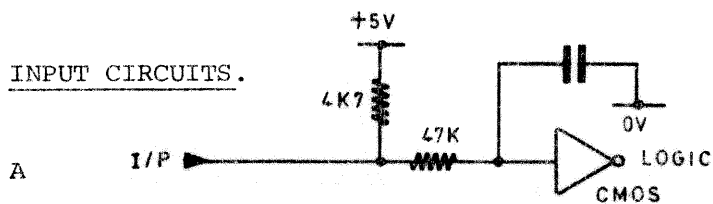
As in section 15 the value returned in an index mode is the number of steps left to complete an index. However in the run mode to obtain the number of steps run, from the axis being started the value returned must be subtracted from FEFF or decimal 65279.

LIST OF CONNECTIONS TO DRIVE

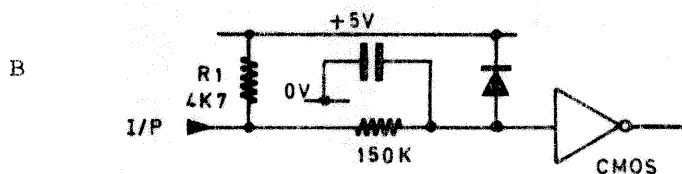
NOTE (see Drg. 1185-035) E1 for mounting box connections use terminal block nos.

<u>PIN NO.</u> <u>TERMINAL</u> <u>BLOCK</u>	<u>PIN NO. PCB</u> <u>TYPED DIN 41612</u>	<u>SIGNAL NAME</u>	<u>TYPE OF I/P OR O/P CIRCUIT</u>
4	a20	X AXIS CK	
9	c12	Y AXIS CK	
16	c16	Z AXIS CK	
3	a8	X AXIS DIR	OUTPUT CIRCUIT TYPE A
8	a16	Y AXIS DIR	
15	a24	Z AXIS DIR.	
6	a12	X AXIS FAULT	
11	a30	Y AXIS FAULT	
18	a28	Z AXIS FAULT	INPUT CIRCUIT TYPE A
5	c6	X AXIS ZERO PHASE	
10	c28	Y AXIS ZERO PHASE	
17	c26	Z AXIS ZERO PHASE	
21	c18	BOOST	OUTPUT CIRCUIT
19	c8	ENERGIZE	
25	a22	EMERGENCY STOP	INPUT CIRCUIT TYPE C
20	a18	STALL CK	INPUT CIRCUIT TYPE D
24	c20	PROX FB CK	INPUT CIRCUIT TYPE B
26	c22	EXT STALL FAULT	INPUT CIRCUIT TYPE A
23,14,12,7,2,	a14,c14	0V	
1,22,	a10, c10,	+24v	

INPUT CIRCUITS.

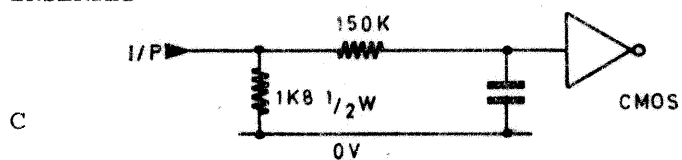


Logic high = 3.5 - 5.5v  
Logic low = 0 - 1v

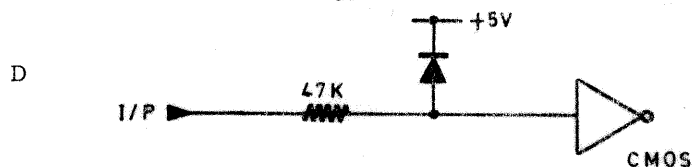


R1 INSERTED  
Logic high = 3.5 - 5.5v  
Logic low = 0 - 1v

INTERFACE DESPATCHED WITH R1  
INSERTED



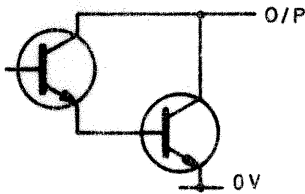
R1 REMOVED  
Logic high = 3.5 - 24v  
Logic low = 0 - 1v



Logic high = 3.4 - .2v  
Logic low = 0 - 1v

OUTPUT CIRCUIT

A.



DARLINGTON ARRAY

Logic High = O/C 30V MAX.  
Logic Low = 1V MAX & 15mA.

INTERFACE SUPPLY

The supply to the interface must be capable of supplying 4 watts, preferably +24V dc at  $\approx 150\text{mA}$  r.m.s. and peak currents of 1.2A. A high frequency capacitor should be connected across supply i.e. 47nf 40V disc.

IEEE - CONTROLLER CONNECTIONS

The connections between the IEEE stepper interface and the controller are via a D type connector. This is not the IEEE 488 standard connector but the more popular IEC 66 standard. IEEE 488 to IEC 66 adaptor cables and blocks can be obtained from TEKDATA (Telephone No. 0782 813631).

All data I/P & O/P are: Logic 1 = 2-5V  
Logic 0 = 0.4V

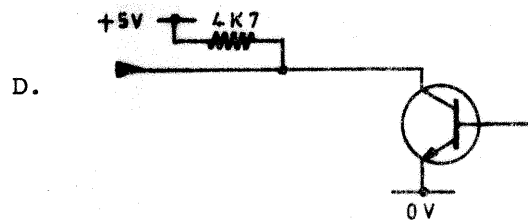
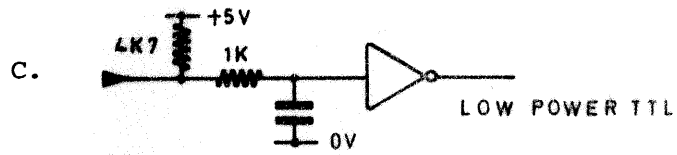
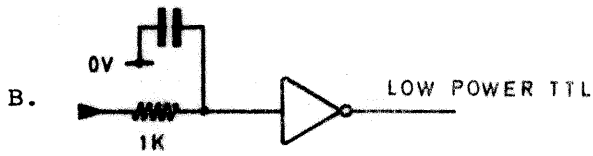
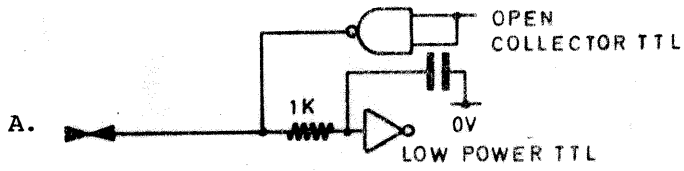
INPUT SINK CURRENT 0.2mA

OUTPUT SINK CURRENT 48mA

All control lines are as above except IFC which has input sink of 1mA.

IEEE D-CONNECTOR SIGNALS IEC66 STANDARD

Signal	D-Conn Pin	Type of I/P-O/P	PCB 3M Connector
D1O1	1 )	A	1
D1O2	2 )		3
D1O3	3 )		5
D1O4	4 )		7
REN	5	B	9
EOI	6 )	A	11
DAV	7 )		13
NRFD	8 )		15
NDAC	9 )		17
IFC	10	C	19
SRQ	11	D	21
ATN	12	B	23
SCREEN	13		-
D1O5	14 )	A	2
D1O6	15 )		4
D1O7	16 )		6
D1O8	17 )		8
GROUND	18		10
GND (6)	19		12
GND (7)	20		14
GND (8)	21		16
GND (9)	22		18
GND (10)	23		20
GND (11)	24		22
GND (12)	25		24





1185 IEEE 488 Check List

1. Check supply to interface is +24V DC.

Pin 1 - +24V ) Terminal Block On Box.

Pin 2 - 0V )

If connections are to card check type of connector, as pin connections change depending on type.

2. Check switch settings on 1185-010.  
When a switch is in the on position then the contact is closed and the switch is not selected i.e. '0'.

BCD mode switch C - NOT ON

Primary address 5 F & H - NOT ON

D, E & G - ON

A and B switch not used

3. Check that all drive fault inputs are at 0V. Link to 0V if not used.
4. Check that EMERGENCY STOP input is taken to +24V. Link to +24V if not used.
5. Check EXT stall input is at +5V, is open circuit input.
6. Check Stall Clock is taken to 0V if not used.
7. Check that REN line is low <0.5V
8. Check that IFC line is high >2V.
9. Check that a  $\$$  follows data to initiate an index or run.  
i.e. "X 400 @ 5000  $\$$ "
10. On HP controllers check that HB IB interface is switched to controller internally.
11. Check drive function and connections. See separate drive manual.
12. Check. Controller/Interface connections see following table.

IEC 66 - IEEE 488 - GPIB  
PET COMMODORE CONNECTIONS

1185 - 010 26 WAY - 3M 3429 SOCKET	IEC 66 25 WAY SOCKET D TYPE	IEEE 488 24 WAY TYPE 57 MICRORIBBON	PET COMMODORE PCB PLUG	SIGNAL NAME
1	1	1	1	DIO1
3	2	2	2	2
5	3	3	3	3
7	4	4	4	4
9	5	17	E	REN
11	6	5	5	EOI
13	7	6	6	DAV
15	8	7	7	NRFD
17	9	8	8	NDAC
19	10	9	9	IFC
21	11	10	10	SRQ
23	12	11	11	ATN
	13	12	12	SCREEN
2	14	13	A	DIO5
4	15	14	B	6
6	16	15	C	7
8	17	16	D	8
10	18	24	F	GROUND
12	19	18		GND (6)
14	20	19		(7)
16	21	20		(8)
18	22	21		(9)
20	23	22		GND (10)
22	24	23		(11)
24	25			(12)
26				GND

PET COMMODORE PIN NO.    1 - 12 UPPER PINS  
                                  A - F    LOWER PINS