iot332 DIGITAL I/O TRAM
USER'S MANUAL

Document Number **50300-30332** Rev **B**

Sunnyside Systems Limited
Nettlehill Road, Houstoun Industrial Estate, Livingston, West Lothian, EH54 5DL, United Kingdom
Telephone: 01506 444911 Facsimile: 01506 441040
VAT Reg. No.: GB 502 5346 81 Reg. No. SC117771 Scotland

WARRANTY NOTICE

Hardware

All boards/modules supplied by Sunnyside Systems Ltd. carry a full one year warranty against failure which occurs during normal use and is due to faulty materials or workmanship. Should a failure occur which requires a product to be returned for repair, the following points apply.

1. Warranty is valid for one year from date of purchase.

2. A Return Material Authorisation (RMA) is required. This can be obtained from the original supplier of the product and should be returned along with it.

3. The product must be returned either in its original packing material, or if this is not available, only in such packing material as is approved by Sunnyside Systems Ltd.

4. If the warranty period has expired, the full cost of repair and carriage is the responsibility of the user.

5. Any attempt to modify or repair the product will void the warranty.

6. Having obtained a return number, (point 2 above), complete the Request for Repair form at the back of this manual and return it with the board. This will ensure that the product is repaired and returned without delay.

Software

Sunnyside Systems Ltd. warrants that diskettes are free from defects in material and workmanship, assuming normal use, for a period of ninety days from date of purchase. Should a defect occur during this time, the faulty diskette should be returned to the original supplier, along with dated proof of purchase. It will then be replaced free of charge.

CONTENTS

1.0 INSTALLATION

Remove the iot332 from its container and inspect it. If there is any sign of damage to the board, its components or the floppy disc then immediately contact the carrier and inform your supplier/distributor or Sunnyside Systems Ltd.

Do not attempt to use the board or floppy disc if they appear to be damaged.

1.1 Prerequisites

To use the Sunnyside Systems iot332 module and associated software you will need the following equipment :-

1. An IBM PC or compatible with a hard disc drive.
2. An Inmos B004 or compatible TRAM motherboard for the PC.
3. Sunnyside Systems Ltd. spt156 Signal Processing TRAM or a compute TRAM which acts as the root processor.
4. 3L Ltd. parallel 'C' compiler or Inmos D7214 'C' toolset installed as instructed in the user documentation supplied with these packages.

1.2 Hardware Installation

The iot332 is installed in the computer as follows :-

1. Remove the power from the computer.
2. Configure the TRAM motherboard base address and link speed as shown in the user's guide supplied with the motherboard. The values should be set as follows :-

   Board base address = 150 Hex
   Link speed = 20Mbit/s

3. Plug the iot332 onto the motherboard, aligning the orientation triangle on the iot332 with the triangle on the chosen motherboard slot.
4. Install the motherboard into the computer. Refer to the user's guide for both the motherboard and computer before attempting this operation.
5. Reconnect the power to the computer.

1.3 Software Installation

The iot332 is supplied with a 5.25", 360 kbyte floppy disc and a 3.5", 720 kbyte disc, both of which contain the following :-

* A test program ('**test332**') which allows the iot332 to be tested.
* 'C' language routines which control the operation of the iot332.

The test program ('**test332**') should be run as soon as the software is installed to verify that the

board is operating correctly (refer to section 2.0).

The discs also contain a file called '**read_adt.me**' which gives some general information about the software and also explains how to install it onto the computers hard disc.

1.3.1 Installing the Software (3L Compiler)

A batch file called '**install**' has been provided to simplify the process of loading the software onto the hard disc. The hard disc is assumed to be drive C:. Insert the floppy disc into drive A: and follow the steps listed below.

The commands shown in **bold** are the ones to be typed. For example if your floppy disk drive is drive A and your hard disk is drive C then type...

C>a:\3l\install a c

After a few seconds a message will appear on the screen and the software will be automatically transferred to the hard disc.

The install process will create a directory called \iot332 on the hard disc. All of the supplied Sunnyside Systems software will be stored in this directory. Upon successful installation of the iot332 software, the directory will contain the following file structure.

Directory :- C:\IOT332

Files :- read_332.me iot332_4.bin iot332_8 .bin

Directory :- C:\IOT332\UTILS

Files :- test332 .exe

Directory :- \IOT332\EXAMPLE

Files :-	iot332 .h	iot332_4.bin	iot332_8 .bin
	ex332 .c	ex332 .cfg	make8 .bat
	ex332_4 .lnk	ex332_8 .lnk	make4 .bat
	view .exe		

Directory :- \IOT332\SOURCE

Files :- iot332 .c iot332 .h

1.3.2 Installing the Software (Inmos Toolset)

A batch file called '**install**' has been provided to simplify the process of loading the software onto the hard disc. The hard disc is assumed to be drive C:. Insert the floppy disc into drive A: and follow the steps listed below.

The commands shown in **bold** are the ones to be typed. For example if your floppy disk drive is drive A and your hard disk is drive C then type...

C>a:\inmos\install a c

After a few seconds a message will appear on the screen and the software will be automatically transferred to the hard disc.

The install process will create a directory called \iot332 on the hard disc. All of the supplied Sunnyside Systems software will be stored in this directory. Upon successful installation of the iot332 software, the directory will contain the following file structure.

Directory :- C:\IOT332

Files :- read_332.me iot332.tco

Directory :- C:\IOT332\UTILS

Files :- test332 .exe

Directory :- \IOT332\EXAMPLE

Files :- iot332 .h iot332 .tco
ex332 .lnk make .bat
ex332 .cfs ex332 .c

Directory :- \IOT332\SOURCE

Files :- iot332 .c iot332 .h

2.0 GETTING STARTED

After the hardware and software have been installed, a confidence test should be run to ensure that the installation has been successful and that the hardware is functioning correctly. This is very simply done by following the procedure in section 2.2 which causes the program '**test332**' to run and report any hardware problems which it finds.

The confidence check 'toggles' the iot332 outputs and reads them to check for correct operation. The outputs can be checked using an oscilloscope which will show each output constantly switching between TTL high and low levels.

2.1 Factory Settings

The iot332 has a number of jumper links which are used to set the operating conditions of the module. The position and function of these jumpers is shown in Figure 2.1.1, along with their factory set condition.

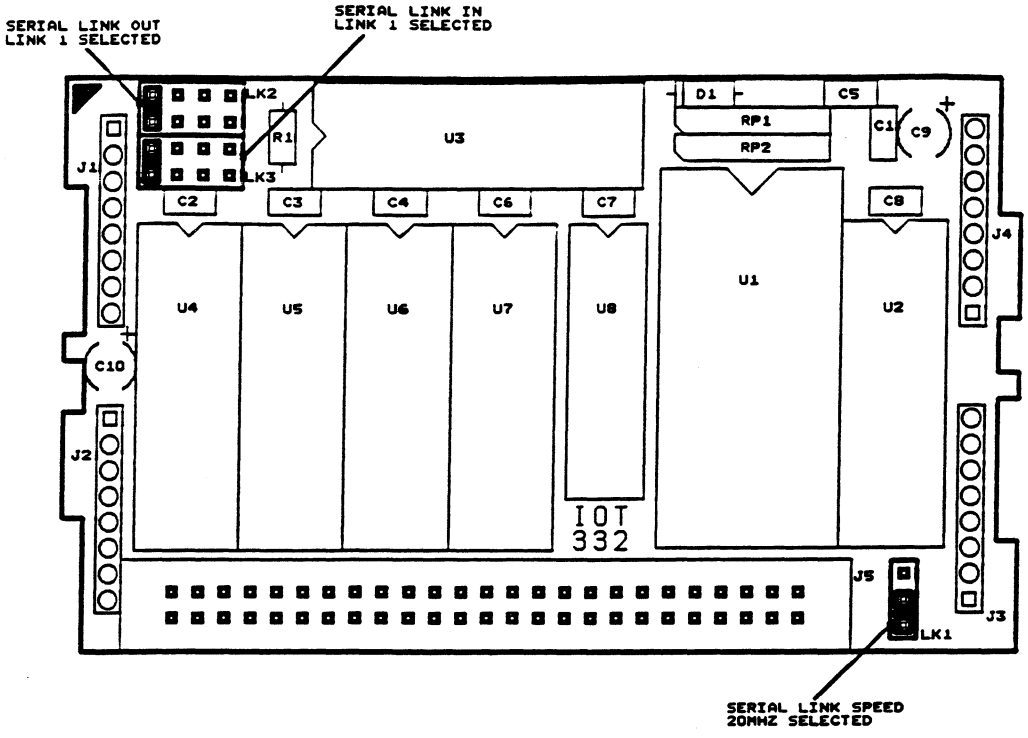


Figure 2.1.1 Jumper Positions and Factory Set Conditions

2.2 Confidence Test

Before running the confidence test, change the jumper settings for the serial link configuration as shown in Figure 2.2.1 below. The jumper links should be set such that the iot332 communicates on link0.

The test simulates the action of an eight bit counter on each of the four groups (ports) of inputs/outputs, (the 32 I/O lines are organised as four ports, each with eight lines). The maximum frequency is present on the LSB I/O pin, the minimum on the MSB I/O pin.

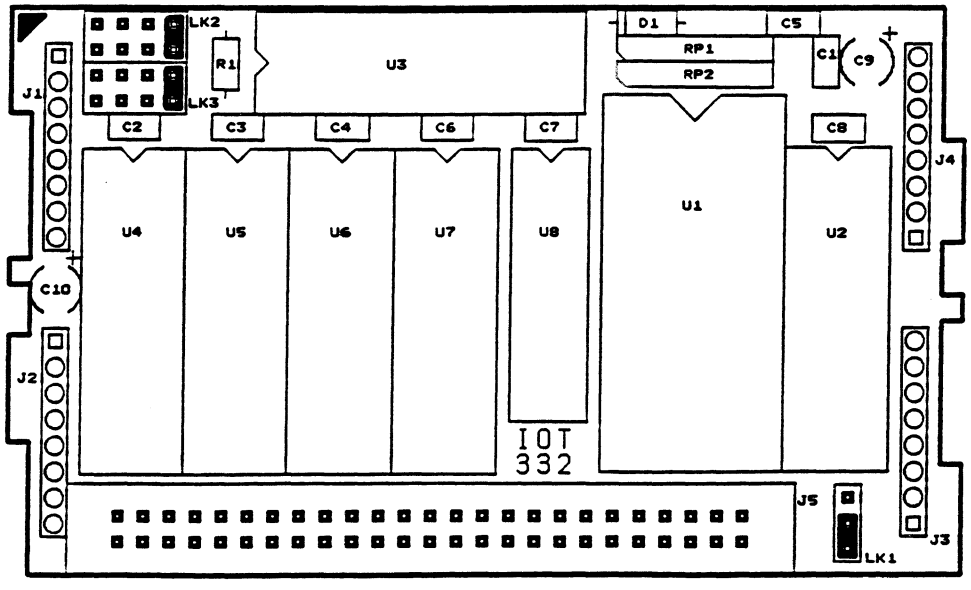


Figure 2.2.1 Confidence Test Serial Link Positions

Insert the iot332 module into slot0 of the motherboard, taking care to align the orientation triangles correctly.

To run the confidence test, the following commands shown in **bold** should be typed...

Log on to hard disc C: (if required).....

A>**c:**
C>

Change to the iot332 utilities directory.....

C>**cd \iot332\utils**
C>

Run the confidence test.....

C>**test332**
C>

The confidence test will check that the iot332 hardware is functioning correctly. Successful completion will result in the test continuing indefinitely until terminated by pressing any key on the computer keyboard. If errors are detected during the test, an error message will be displayed and the test will be aborted. If this happens, refer to Section 6.0, Troubleshooting for help in identifying possible causes of the problem.

Upon successful completion of the confidence test, the jumper links for the serial communications should be set as shown in Figure 2.2.2 below. The iot332 will now be set to communicate on serial link 1.

The iot332 should also be moved to a free slot on the motherboard where it will act as a slave to the root Transputer module.

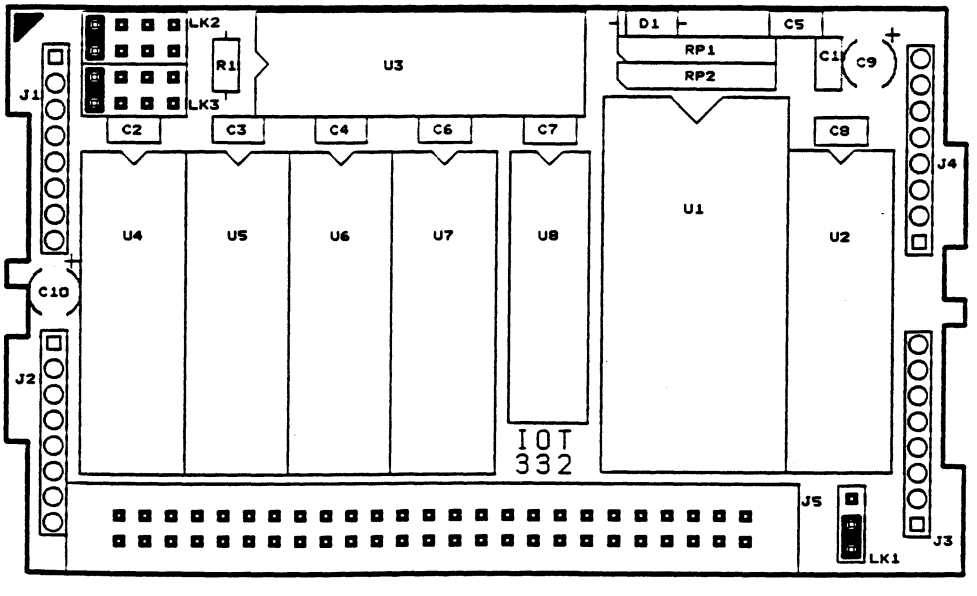


Figure 2.2.2 Link Positions After Confidence Test

3.0 USING THE LIBRARY ROUTINES

The software supplied with the iot332 contains routines and functions which help to make the module easy to use. With the aid of this software, it is possible to configure the I/O lines as inputs or outputs as required and to read/set TTL compatible levels on each input/output line.

By way of an example, this section uses the supplied software to configure the iot332 I/O lines, (16 as inputs and 16 as outputs). The status of each of the lines is displayed on the screen. The output lines are set by the program to behave like two eight bit counters. Note that unless the lines configured as inputs are driven, they will default to an undefined level. One option is to connect the inputs and outputs together. In this case the input and output data would track each other. The steps to be followed in order to create, assemble, compile, link and finally run the program are described below.

3.1 Running an Example Program (ex332.c) [3L Compiler]

In this example we will use the software routines and functions supplied on the Sunnyside Systems disc to configure the I/O lines on the iot332, set each of the output lines to a pre-determined level and read the level on each of the input lines. During the example we will perform the following procedure...

1. Compile and link the Transputer software.
2. Download and run the Transputer software.

Before attempting the example, ensure that the Sunnyside Systems iot332 board with a TRAM motherboard are installed within your PC and that the Sunnyside software is resident on your hard disc. If this is not the case then please refer to section 1.0, Installation. The 3L parallel 'C' should also be installed on your hard disk and their directories included in the DOS 'path' command.

A TRAM which can act as the root processor must be installed on the motherboard. The iot332 is slave to the root Transputer, linked through link 2 on the root module and link 1 on the iot332. For details on how to configure the links on the motherboard please consult the appropriate manual. For the iot332 link settings please refer to Section 4.5 of this manual.

STEP 1 : Changing to the EXAMPLE directory

Change your current directory to the iot332\example directory on your hard disc...

C>cd \iot332\example

List the directory contents (C>**dir**) and check that all of the following files are present...

iot332.h
iot332_4.bin
iot332_8.bin
ex332.c
ex332.cfg
ex332_4.lnk
ex332_8.lnk
make4.bat
make8.bat

The files listed above (apart from the ex332*.*) are duplicated in other iot332 directories on the hard disc so there need be no fear of irreparably corrupting files.

STEP 2 : Compiling the EXAMPLE program (ex332)

We must now compile the code that is to run on the Transputer. This is done using the 3L 'C' compiler.

The 'C' source code to be run on the Transputer is contained within the file **ex332.c**. The compile command will depend upon the type of Transputer on which the code will be executed, either T400 or T805. If you are unsure of the Transputer type fitted to the TRAM module in your system then look at the top of the device. The following will be printed on the top of the IC depending upon its type...

IMST400 : device is a T400
IMST805 : device is a T805

To compile the example program with a T400 transputer type the following...

C>t4c ex332.c

To compile the example program with a T805 transputer type the following...

C>t8c ex332.c

If you now list your directory contents there should be an additional file **ex332.bin**. If the compiler fails to run, check that the root Transputer module has been properly installed.

STEP 3 : Linking the EXAMPLE (ex332.c) program

It is now necessary to link the example program to the rest of the Transputer code. Included within the directory are two make files, one for each transputer type...

make4.bat : Make file for T400 transputer
make8.bat : Make file for T805 transputer

To link the software for a T400 Transputer type the following...

C>make4 ex332

To link the software for a T800 Transputer type the following...

C>make8 ex332

STEP 4 : Running the EXAMPLE (ex332.c) program

Having compiled and linked the Transputer source code, we are now ready to download and run the example program.

To run the code we need to invoke the 3L 'alien file server' program. This is a program which runs on the host PC and downloads code to the root Transputer in the system. It will also handle any communications made by the root Transputer to the host computer, for example printing messages to the screen.

To run the code type the following...

C>afserver -:b ex332.app

The status of the I/O lines is displayed on the screen. If the outputs are connected to the inputs then the data read back will track the data set on the output pins.

3.2 Running an Example Program (ex332.c) [Inmos Toolset]

In this example we will use the software routines and functions supplied on the Sunnyside Systems disc to configure the I/O lines on the iot332, set each of the output lines to a pre-determined level and read the level on each of the input lines. During the example we will perform the following procedure...

1. Compile and link the Transputer software.
2. Download and run the Transputer software.
3. Display the data collected by the iot332.

Before attempting the example, ensure that the Sunnyside Systems iot332 board with a TRAM motherboard are installed within your PC and that the Sunnyside software is resident on your hard disc. If this is not the case then please refer to section 1.0, Installation. The Inmos D7214 'C' Toolset should also be installed on your hard disk and their directories included in the DOS 'path' command.

A TRAM which can act as the root processor must be installed on the motherboard. The iot332 is slave to the root Transputer, linked through link 2 on the root module and link 1 on the iot332. For details on how to configure the links on the mother board please consult the appropriate manual. For the iot332 link settings please refer to Section 4.5 of this manual.

STEP 1 : Changing to the EXAMPLE directory

Change your current directory to the iot332\example directory on your hard disc...

C>cd \iot332\example

List the directory contents (C>**dir**) and check that all of the following files are present...

iot332.h
iot332.tco
ex332.c
ex332.cfs
ex332.lnk
make.bat

The files listed above are duplicated in other iot332 directories on the hard disc so there need be no fear of irreparably corrupting files.

STEP 2 : Compiling the EXAMPLE program (ex332)

We must now compile the code that is to run on the Transputer. This is done using the Inmos 'C' compiler.

The 'C' source code to be run on the Transputer is contained within the file **ex332.c**. The compile command will depend upon the type of Transputer on which the code will be executed, either T400 or T805. If you are unsure of the Transputer type fitted to the TRAM module in your system then look at the top of the device. The following will be printed on the top of the IC depending upon its type...

IMST400 : device is a T400
IMST805 : device is a T805

To compile the example program with a T400 transputer type the following...

C>icc ex332.c /T400

To compile the example program with a T805 transputer type the following...

C>icc ex332.c /T805

If you now list your directory contents there should be an additional file **ex332.tco**. If the compiler fails to run, check that the root Transputer module has been properly installed.

NOTE:- All the *.tco files on the installation disk have been compiled with the /T800 switch. If a T400 is fitted to the spt156 or root Transputer then it will be necessary to recompile them with the appropriate switch (/T400).

STEP 3 : Linking the EXAMPLE (ex332.c) program

It is now necessary to link the example program to the rest of the Transputer code. Included within the directory is a make file to simplify the link and configure process, type...

make ex332 T400 : Make for T400 Transputer
make ex332 T805 : Make for T805 Transputer.

STEP 4 : Running the EXAMPLE (ex332.c) program

Having compiled and linked the Transputer source code, we are now ready to download and run the example program.

To run the code we need to invoke the file server program. This is a program which runs on the host PC and downloads code to the root Transputer in the system. It will also handle any communications made by the root Transputer to the host computer, for example printing messages to the screen.

To run the code type the following...

C>iserver /sb ex332.btl

The status of the I/O lines is displayed on the screen. If the outputs are connected to the inputs then the data read back will track the data set on the output pins.

4.0 HARDWARE

4.1 Overview

The iot332 is a programmable digital input/output module in which the 32 TTL compatible lines are organised as 4 ports each with 8 lines. Each port can be selected, under software control, as either input or output. All I/O lines are fully TTL compatible with up to 64mA current sink capability.

Each of the lines in an output port can be independently set or cleared under software control. The values last written to the output lines are stored in an on-board latch and can be read back for verification purposes.

The Transputer interface is via an Inmos CO11 Link Adapter, user selectable from any one of the four links.

External I/O signals are taken through a 50-way IDC connector which is pin compatible with the Arcom range of signal conditioning cards.

The iot332 is a size two TRAM conforming to the Inmos electrical and mechanical format for TRAM modules.

4.2 Specifications

Electrical

Typical at 25 C and rated power supplies unless otherwise stated

INPUT/OUTPUT CONFIGURATION	
Number of Channels	32
Organisation	4 ports, 8 bits per port
Programmability	Each port independently programmable as input or output
Power-up State	All ports initialised as inputs
INPUT CHARACTERISTICS	
High Level Voltage	2V, min
Low Level Voltage	0.8V, max
High Level Current	20uA, max
Low Level Current	700uA, max
Max/Min Input Voltage (without damage)	+5.5V, -0.5V
OUTPUT CHARACTERISTICS	
High Level Voltage	2.4V, min
Low Level Voltage	0.6V, max

High Level Current	15mA, max
Low Level Current	64mA, max
POWER REQUIREMENTS	
iot332	+5 Volt (+/- 5%) @ 100mA
ENVIRONMENTAL	
Operating Temperature	0 C to +60 C
Storage Temperature	-25 C to +85 C
Relative Humidity	5% to 85% non condensing
MECHANICAL	
Format	Size 2 TRAM, 2.15" x 3.66"
Input/Output Connector	50 Way IDC Type, Arcom Pinout Compatible

4.3 Architecture

A block diagram of the iot332 board is shown in Figure 4.3.1. below.

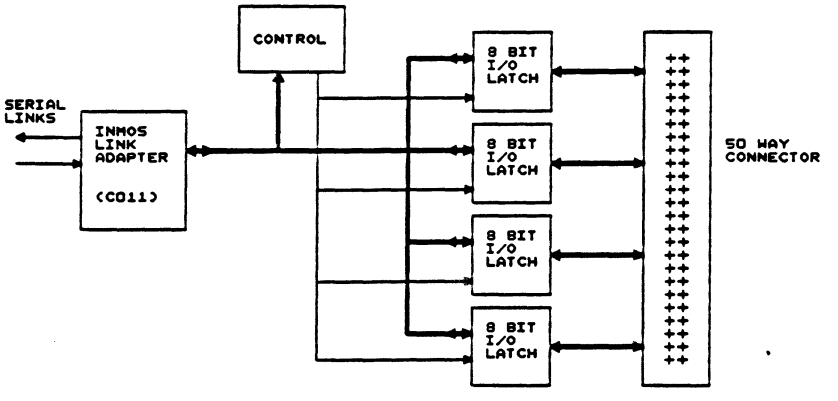


Figure 4.3.1 iot332 Block Diagram

As shown in Figure 4.3.1, the iot332 comprises four octal I/O devices, an Inmos link adapter and control logic.

The iot332 is configured, outputs updated and inputs read by sending or receiving a serial bit stream transmitted across the selectede TRAM link.

4.4 Input/Output Connections

Signals are connected to the module through a fifty way IDC connector. The connector pinouts are shown in Appendix B. Connector pinouts are compatible with the Arcom range of signal conditioning products.

The connector pinout and port arrangement is as follows...

port 0 MSB ..... LSB
data 7 ..... data 0

port 1 MSB ..... LSB
data 15 ..... data 8

port 2 MSB ..... LSB
data 23 ..... data 16

port 3 MSB ..... LSB
data 31 ..... data 24

4.5 Link Selection and Link Speed Selection

The serial link selection to the iot332 can be made to any one of the four links LKIN0,1,2,3 and LKOUT0,1,2,3 by the appropriate positioning of the jumper links LK2 and LK3. The positions of these links on the iot332 are shown in Figure 4.5.1. Take care to insert the jumpers correctly.

The link speed is selected using jumper link LK1 as shown in Figure 4.5.2.

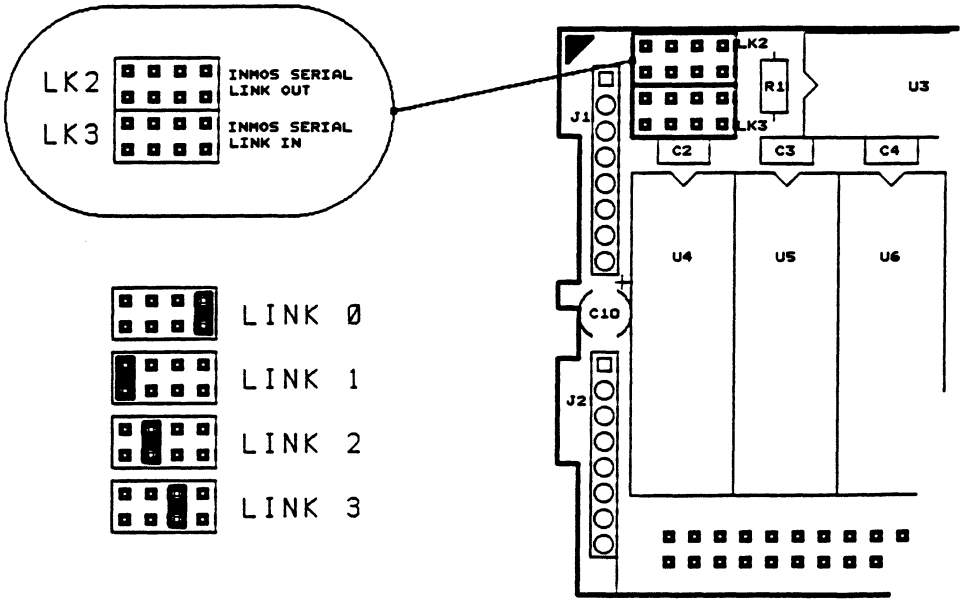


Figure 4.5.1 Link Selection

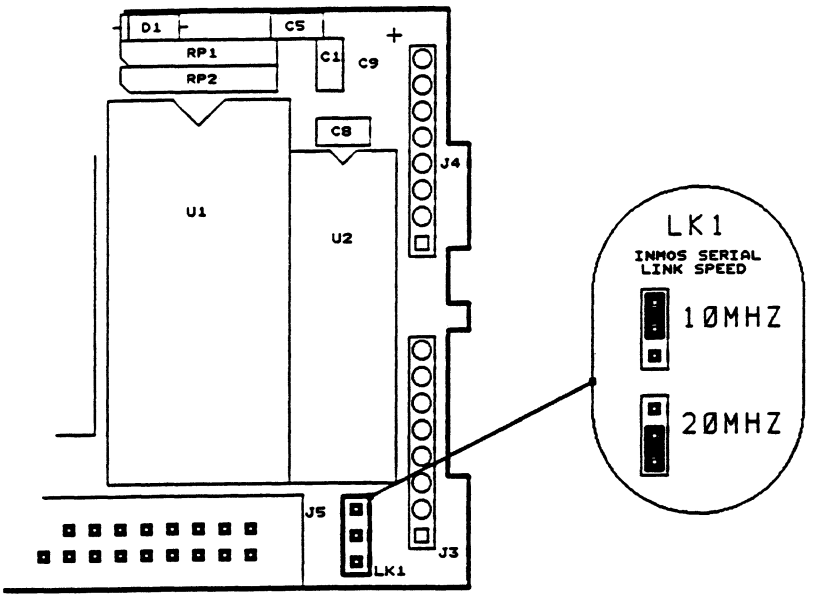


Figure 4.5.2. Link Speed Selection

4.6 Hardware Operation/Programming

The iot332 data acquisition module is provided with software support which allows the module to be used without detailed knowledge of its operation. However in the event that the supplied software is required to be modified, or new software written, this section explains the operation of the board and how to program it.

The iot332 is controlled by sending bytes of data across the Inmos serial links. There are four control instructions. Each is listed below along with its meaning.

Control instruction summary :-

SETIO : Set the ports to be input or output
READ : Read from a port
WRITE : Write to an output port
ID : Read the module identification byte

The control instructions and their function are described in detail in sections 4.6.1 through 4.6.4.

4.6.1 Set the iot332 Ports (SETIO)

This control instruction sets the ports to be inputs or outputs. The ports are numbered 0,1,2 and 3 with each port being independent of the others. The instruction contains two bytes, the first denotes the SETIO instruction, the second defines the port condition to be set. The format of the instruction is...

First Byte < 0 0 >
Second Byte < data > where data sets the port conditions as follows...

DATA (Hex)	PORT NUMBER 0	1	2	3
00	I	I	I	I
01	O	I	I	I
02	I	O	I	I
03	O	O	I	I
04	I	I	O	I
05	O	I	O	I
06	I	O	O	I
07	O	O	O	I
08	I	I	I	O
09	O	I	I	O
0A	I	O	I	O
0B	O	O	I	O
0C	I	I	O	O
0D	O	I	O	O
0E	I	O	O	O
0F	O	O	O	O

Where...
I = INPUT
O = OUTPUT

NOTE :-The power-on default condition is all ports set to inputs.

For example, to set the iot332 to operate with ports 0 and 1 as outputs and ports 2 and 3 as inputs, the following would be sent...

First Byte < 0 0 >
Second Byte < 0 3 >

4.6.2 **Read from a Port (READ)**

This instruction is used to read the data from the ports which are configured as inputs or to read the values which were last written to the ports configured as outputs. The instruction contains two bytes. The first byte denotes the READ instruction whilst the second defines which port is to be read. A single byte of data is returned for each read. A TTL high level is returned by the adt332 as a '1' whilst a low level is returned as a '0'. The format of the instruction is...

First Byte < 0 1 >
Second Byte < port > where ...port = 00 port 0
01 port 1
02 port 2
03 port 3

For example, to read the status of port 2 the following would be sent...

First Byte < 0 1 >
Second Byte < 0 2 >

The adt332 would then respond by sending a byte of data which related to the input levels on port 2.

4.6.3 **Write to an Output Port (WRITE)**

This instruction is used to write data to the ports which are configured as outputs. Data are written to each port independently of the others. The instruction contains two bytes. The first byte selects which port is to be written, the second contains the data to be output to the port. The format of the instruction is...

First Byte	< port >	where ...port	=	04 port 0
				05 port 1
				06 port 2
				07 port 3
Second Byte	< data >			

For example, to set port 3 such that the four MSB's are high and the four LSB's are low, the following would be sent...

First Byte	< 07 >
Second Byte	< F0 >

4.6.4 **Send the Module Identification Code (ID)**

This instruction is used to request the iot332 to send its module identification number. This is a single byte instruction. In response to this request the iot332 returns a value of 33H. The format of the instruction is...

Byte	< 0 F >

5.0 SOFTWARE SUPPORT

5.1 Introduction

The iot332 is supplied with support software on floppy disc, suitable for use on IBM PC or compatible machines. The software is intended to simplify the integration and use of the iot332 in your system. It does this by providing functions which control the operation of the module and which can be used in your own programs. The 'C' functions are simply included in your own applications software and allow the use of the iot332 without detailed knowledge of the hardware operation.

5.2 iot332 Software Functions

Supplied with the Sunnyside Systems iot332 module are the files :

IOT332.C - Contains the control functions required to set the iot332 ports.

IOT332.H - Contains prototypes and definitions used by iot332 software functions.

EX332.C - An example program using the iot332 functions.

EX332.CFG - Configuration file.

5.3 Configuring the Hardware and Software.

The software has been written in parallel 'C' to run on a Transputer. It is envisaged that the iot332 board will be linked to the Transputer via serial link 1. If this is not the case it is possible to change the link number in the configuration file.

Provided with the example software is a 3L 'C' configuration file. This file describes the hardware and software configuration of the system. In the example case, the hardware configuration consists of a single spt156 (Note 1) and a single iot332. These are linked by the standard Inmos link. Link 1 on the iot332 and link 2 on the spt156 are used. This connection is set with jumpers on the motherboard and the iot332 module.

The software configuration may seem a bit daunting at first. In the example there are three '**processes**' running on the system. The first is the 'AFSERVER' that is running on the host computer (PC). This waits for I/O requests from the spt156. The second is the example running on the spt156. This is the actual code we are interested in. The third, strictly speaking is not really a process at all, but must be described as such in the configuration file. It is 'running' on the iot332. In reality it is the actual hardware protocol that interfaces to the Inmos link. Fig. 5.3.1 shows a listing of the actual configuration file.

NOTE 1 :- The spt156 is a DSP TRAM manufactured by Sunnyside Systems which can act as the root Transputer in the computer system.

```
! EXAMPLE.CFG

processor host type=PC          ! The PC.
processor root                  ! The spt156.
processor iot332 type=PC        ! The iot332.
wire j1 root[0] host[0]         ! Join the PC and the spt156.
wire j2 root[2] iot332[1]       ! Join the iot332 and the spt156.


! Now describe the tasks that are on the system.

task ex332      ins=3 outs=3
task afserver   ins=1 outs=1
task adtserv    ins=2 outs=2
task filter     ins=2 outs=2 data=10k


! Now put the tasks onto the system.

place afserver host             ! The afserver is on the PC.
place ex332 root                ! The example is on the spt156.
place adtserv iot332            ! The protocol is on the iot332.
place filter root               !


! Now connect the tasks together through software channels.

connect ? filter[0] afserver[0]
connect ? afserver[0] filter[0]

connect ? filter[1] ex_332[1]
connect ? ex332[1] filter[1]

connect ? ex332[2] adtserv[0]
connect ? adtserv[0] ex_332[2]
```

Figure. 5.3.1 Configuration File

The line shown in boldface is the only line which has to be changed if a different hardware configuration is used. For example if the iot332 was connected to the spt156 through link 3 on the iot332 and link 3 on the spt156 then this line would become:

wire j2 root[3] iot332[3]

For more information on configuration files please refer to the 3L 'C' manual (Section 5.1).

If you experience any problems with the subject of configuration files please do not hesitate to contact your Supplier/Distributor or Sunnyside Systems Ltd.

5.4 iot332 Function Library

This section lists all of the iot332 software functions supplied by Sunnyside Systems Ltd in the IOT332.C file.

init_iot332 - Initialise the channels (serial links)
set_io_iot332 - Set the port I/O arrangement
read_block_iot332 - Read a byte from an input port
write_block_iot332 - Write a byte to an output port
iot332_id - Read the iot332 ID byte

A description of each of the above functions now follows...

FUNCTION : **set_io_iot332**

Syntax : void set_io_iot332(int data)

Description : This function specifies which of the ports are to be set as inputs and which as outputs.

Parameters : data: This parameter specifies the combination of inputs and outputs. Legal values are:

00 - All ports inputs.
01 - Port 0 output.
02 - Port 1 output.
03 - Port 0, 1 output.
04 - Port 2 output.
05 - Port 2, 0 output.
06 - Port 2, 1 output.
07 - Port 2, 1, 0 output.
08 - Port 3 output.
09 - Port 3, 0 output.
0A - Port 3, 1 output.
0B - Port 3, 1, 0 output.
0C - Port 3, 2 output.
0D - Port 3, 2, 0 output.
0E - Port 3, 2, 1 output.
0F - All ports outputs.

Return Value: None

Example : set_io_iot332(0);

This example sets all the ports as inputs.

FUNCTION : **read_block_iot332**

Syntax : char read_block_iot332(int block)

Description : This function reads one of four bytes (one of the input ports) from the iot332.

Parameters : block: The byte to read, where... 0 = port 0
1 = port 1
2 = port 2
3 = port 3

Return Value: The value of the byte read.

Example :
```
BYTE val;
set_io_iot332(0);
val = read_block_iot332(0);
```

This example reads port 0.

FUNCTION : **write_block_iot332**

Syntax : void write_block_iot332(int block, char data)

Description : This function writes one of four bytes to the iot332 (sets the state of the output ports).

Parameters : block: The port to write to, where... 0 = port 0
1 = port 1
2 = port 2
3 = port 3

data: The data to write.

Return Value: None.

Example :
```
set_io_iot332(0x0F);
write_block_iot332(0, 255);
```

This example sets all of the outputs on port 0 to be high.

FUNCTION : **iot332_id**

Syntax : char iot332_id(void)

Description : This function returns the id for the iot332.

Parameters : None.

Return Value: 0x33.

Example : char ID;
ID = iot332_id();

This example reads the iot332 identification byte.

FUNCTION : **init_iot332**

Syntax : void init_iot332(CHAN *in, CHAN *out)

Description : This function initialises the iot332 links.

Parameters : CHAN *in : A pointer to an input channel.
CHAN *out : A pointer to an output channel.

Return Value: None.

Example : init_iot332(in_ports[2], out_ports[2]);

6.0 TROUBLE SHOOTING

The following section gives a list of errors that may be encoutered when running the iot332 board and software. Beneath each problem is a list of the possible cause.

PC won't boot
The TRAM motherboard may be at the same address as another card in your PC. Try changing the motherboard base address.

Server terminated: cannot boot root Transputer
This message originates from the 3L software when it gets no response from the root Transputer. This could be for a number of reasons...

1. The motherboard card is not inserted securely into the PC expansion slot causing poor connections.

2. The root TRAM module is not plugged in correctly. Check that the orientation triangles are aligned.

bad INT32
This signifies there has been a communications protocol error which is usually caused by a loose connection. Check that the motherboard and all TRAM modules are inserted securely.

Other circumstances which could lead to problems, relating specifically to the iot332 hardware are...

1. Ensure that the Serial Output Code jumpers are in the correct position.

7.0 APPENDICES

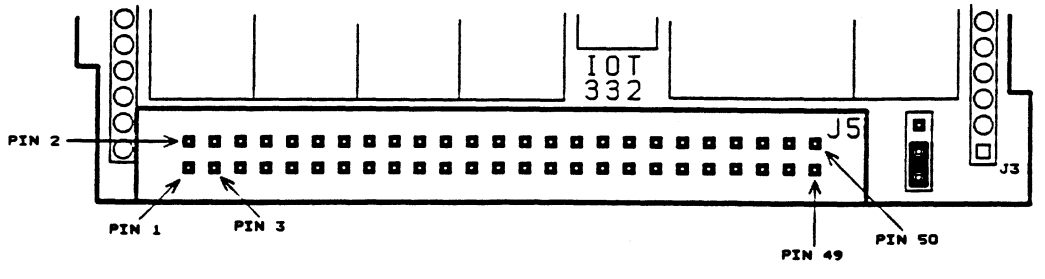
7.1 Appendix A : iot332 TRAM Connector Pinout

	Connector J1	Connector J2
Pin 1 :	LINK 2 OUT	nc
Pin 2 :	LINK 2 IN	nc
Pin 3 :	VCC (+5 Volt)	nc
Pin 4 :	LINK 1 OUT	nc
Pin 5 :	LINK 1 IN	nc
Pin 6 :	nc	GND (0 Volt)
Pin 7 :	nc	nc
Pin 8 :	CLOCK (5 MHZ}	nc

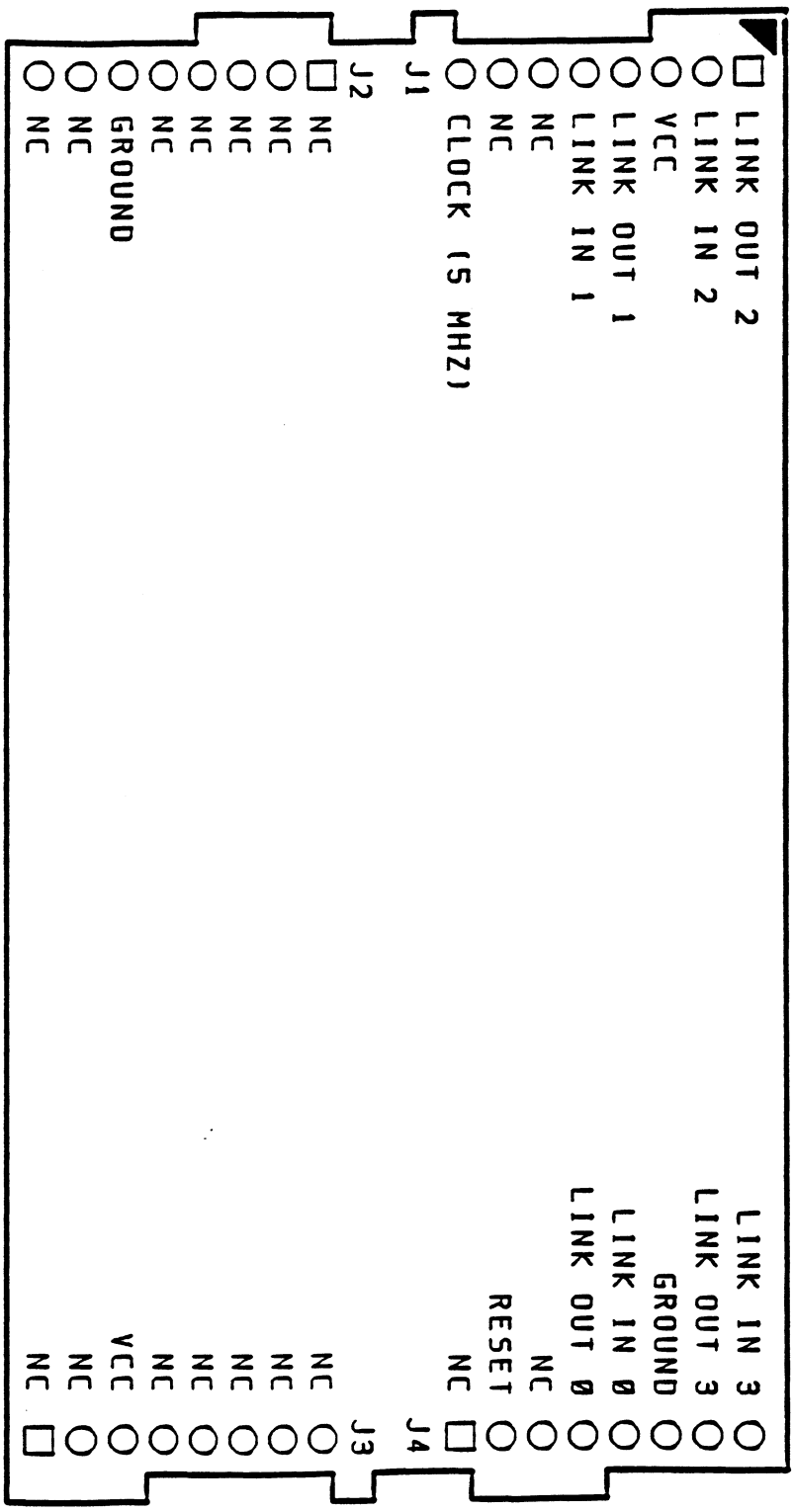
	Connector J3	Connector J4
Pin 1 :	nc	nc
Pin 2 :	nc	RESET
Pin 3 :	VCC (+5 Volt)	nc
Pin 4 :	nc	LINK 0 OUT
Pin 5 :	nc	LINK 0 IN
Pin 6 :	nc	GND (0 Volt)
Pin 7 :	nc	LINK 3 OUT
Pin 8 :	nc	LINK 3 IN

7.2 Appendix B : iot332 I/O Connector Pinout

	IDC CONNECTOR, J5			
	PIN NO.	SIGNAL	PIN NO.	SIGNAL
	PIN 1	GROUND	PIN 2	GROUND
PORT 0	PIN 3	DATA 0	PIN 4	DATA 1
PORT 0	PIN 5	DATA 2	PIN 6	DATA 3
PORT 0	PIN 7	DATA 4	PIN 8	DATA 5
PORT 0	PIN 9	DATA 6	PIN 10	DATA 7
	PIN 11	GROUND	PIN 12	N.C.
PORT 1	PIN 13	DATA 8	PIN 14	DATA 9
PORT 1	PIN 15	DATA 10	PIN 16	DATA 11
PORT 1	PIN 17	DATA 12	PIN 18	DATA 13
PORT 1	PIN 19	DATA 14	PIN 20	DATA 15
	PIN 21	GROUND	PIN 22	N.C.
PORT 2	PIN 23	DATA 16	PIN 24	DATA 17
PORT 2	PIN 25	DATA 18	PIN 26	DATA 19
PORT 2	PIN 27	DATA 20	PIN 28	DATA 21
PORT 2	PIN 29	DATA 22	PIN 30	DATA 23
	PIN 31	GROUND	PIN 32	N.C.
PORT 3	PIN 33	DATA 24	PIN 34	DATA 25
PORT 3	PIN 35	DATA 26	PIN 36	DATA 27
PORT 3	PIN 37	DATA 28	PIN 38	DATA 29
PORT 3	PIN 39	DATA 30	PIN 40	DATA 31
	PIN 41	N.C.	PIN 42	N.C.
	PIN 43	N.C.	PIN 44	N.C.
	PIN 45	N.C.	PIN 46	N.C.
	PIN 47	N.C.	PIN 48	N.C.
	PIN 49	+5 VOLTS	PIN 50	+5 VOLTS



7.3 Appendix C : iot332 Board Profile



8.0 REFERENCES

The following data books/manuals will be useful when using the iot332.

1. The Transputer Databook - Inmos
2. The Transputer Applications Notebook, Systems and Performance - Inmos
3. The Transputer Applications Notebook, Architecture and Software - Inmos

9.0 OCCAM DRIVER SUPPLEMENT

9.1 Installation

The Occam driver software is very simply installed by following the procedure outlined below. Should you experience any difficulty during the installation of the supplied software, please contact your Distributor or Sunnyside Systems Ltd.

9.1.1 Installing the Software (Occam 2 Toolset)

A batch file called **'install'** has been provided to simplify the process of loading the software onto the hard disc. The hard disc is assumed to be drive C:. Insert the floppy disc into drive A: and follow the steps listed below.

The commands shown in **bold** are the ones to be typed. For example if your floppy disk drive is drive A and your hard disk is drive C then type...

C>a:\occam\install a c

After a few seconds a message will appear on the screen and the software will be automatically transferred to the hard disc.

The install process will create a directory called \iot332 on the hard disc. All of the supplied Sunnyside Systems software will be stored in this directory. Upon successful installation of the iot332 software, the directory will contain the following file structure.

Directory :- C:\IOT332

Files :- read_332.me iot332.tco

Directory :- \IOT332\EXAMPLE

Files :- ex332 .occ iot332 .tco
ex332 .lnk make .bat
ex332 .pgm

Directory :- \IOT332\SOURCE

Files :- iot332 .occ iot332 .inc

9.2 Using the Library Routines

The software supplied with the iot332 contains routines and functions which help to make the module easy to use. With the aid of this software, it is possible to configure the I/O lines as inputs or outputs as required and to read/set TTL compatible levels on each input/output line.

By way of an example, this section uses the supplied software to configure the iot332 I/O lines, (16 as inputs and 16 as outputs). The status of each of the lines is displayed on the screen. The output lines are set by the program to behave like two eight bit counters. Note that unless the lines configured as inputs are driven, they will default to an undefined level. One option is to connect the inputs and outputs together. In this case the input and output data would track each other. The steps to be followed in order to create, assemble, compile, link and finally run the program are described below.

9.2.1 Running an Example Program (ex332.occ) [Occam 2 Toolset]

In this example we will use the software routines and functions supplied on the Sunnyside Systems disc to configure the I/O lines on the iot332, set each of the output lines to a pre-determined level and read the level on each of the input lines. During the example we will perform the following procedure...

1. Compile and link the Transputer software.
2. Download and run the Transputer software.
3. Display the data collected by the iot332.

Before attempting the example, ensure that the Sunnyside Systems iot332 board with a TRAM motherboard are installed within your PC and that the Sunnyside software is resident on your hard disc. If this is not the case then please refer to section 1.0, Installation. The Inmos D7205 'Occam 2' Toolset should also be installed on your hard disk and their directories included in the DOS 'path' command.

A TRAM which can act as the root processor must be installed on the motherboard. The iot332 is slave to the root Transputer, linked through link 2 on the root module and link 1 on the iot332. For details on how to configure the links on the mother board please consult the appropriate manual. For the iot332 link settings please refer to Section 4.5 of this manual.

STEP 1 : Changing to the EXAMPLE directory

Change your current directory to the iot332\example directory on your hard disc...

C>cd \iot332\example

List the directory contents (C>**dir**) and check that all of the following files are present...

iot332.inc
iot332.tco
ex332.occ
ex332.pgm
ex332.lnk
make.bat

The files listed above are duplicated in other iot332 directories on the hard disc so there need be no fear of irreparably corrupting files.

STEP 2 : Compiling the EXAMPLE program (ex332)

We must now compile the code that is to run on the Transputer. This is done using the Inmos 'Occam 2' compiler.

The 'Occam' source code to be run on the Transputer is contained within the file **ex332.occ.** The compile command will depend upon the type of Transputer on which the code will be executed, either T400 or T805. If you are unsure of the Transputer type fitted to the TRAM module in your system then look at the top of the device. The following will be printed on the top of the IC depending upon its type...

IMST400 : device is a T400
IMST805 : device is a T805

To compile the example program with a T400 transputer type the following...

C>oc ex332.occ /T400

To compile the example program with a T805 transputer type the following...

C>oc ex332.occ /T805

If you now list your directory contents there should be an additional file **ex332.tco.** If the compiler fails to run, check that the root Transputer module has been properly installed.

NOTE:- All the *.tco files on the installation disk have been compiled with the /T800 switch. If a T400 is fitted to the spt156 or root Transputer then it will be necessary to recompile them with the appropriate switch (/T400).

STEP 3 : Linking the EXAMPLE (ex332.occ) program

It is now necessary to link the example program to the rest of the Transputer code. Included within the directory is a make file to simplify the link and configure process, type...

make ex332 T400 : Make for T400 Transputer
make ex332 T805 : Make for T805 Transputer.

STEP 4 : Running the EXAMPLE (ex332.occ) program

Having compiled and linked the Transputer source code, we are now ready to download and run the example program.

To run the code we need to invoke the file server program. This is a program which runs on the host PC and downloads code to the root Transputer in the system. It will also handle any communications made by the root Transputer to the host computer, for example printing messages to the screen.

To run the code type the following...

C>iserver /sb ex332.btl

The status of the I/O lines is displayed on the screen. If the outputs are connected to the inputs then the data read back will track the data set on the output pins.

9.3 Software Support

9.3.1 Introduction

The iot332 is supplied with support software on floppy disc, suitable for use on IBM PC or compatible machines. The software is intended to simplify the integration and use of the iot332 in your system. It does this by providing procedures which control the operation of the module and which can be used in your own programs. The 'Occam' procedures are simply included in your own applications software and allow the use of the iot332 without detailed knowledge of the hardware operation.

9.3.2 iot332 Software Procedures

Supplied with the Sunnyside Systems iot332 module are the files :

IOT332.OCC - Contains the control functions required to set the iot332 ports.

IOT332.INC - Contains definitions used by iot332 software functions.

EX332.OCC - An example program using the iot332 functions.

EX332.PGM - Configuration file.

9.3.3 Configuring the Hardware and Software.

The software has been written in parallel 'Occam' to run on a Transputer. It is envisaged that the iot332 board will be linked to the Transputer via serial link 2. If this is not the case it is possible to change the link number in the configuration file.

9.3.4 iot332 Function Library

This section lists all of the iot332 software functions supplied by Sunnyside Systems Ltd in the IOT332.OCC file.

set.io.iot332 - Set the port I/O arrangement
read.block.iot332 - Read a byte from an input port
write.block.iot332 - Write a byte to an output port
iot332.id - Read the iot332 ID byte

A description of each of the above functions now follows...

FUNCTION : **set.io.iot332**

Syntax : PROC set.io.iot332(CHAN OF BYTE out, VAL BYTE data)

Description : This function specifies which of the ports are to be set as inputs and which as outputs.

Parameters : out: This is the software channel used to communicate with the 332.
data: This parameter specifies the combination of inputs and outputs. Legal values are:

00 - All ports inputs.
01 - Port 0 output.
02 - Port 1 output.
03 - Port 0, 1 output.
04 - Port 2 output.
05 - Port 2, 0 output.
06 - Port 2, 1 output.
07 - Port 2, 1, 0 output.
08 - Port 3 output.
09 - Port 3, 0 output.
0A - Port 3, 1 output.
0B - Port 3, 1, 0 output.
0C - Port 3, 2 output.
0D - Port 3, 2, 0 output.
0E - Port 3, 2, 1 output.
0F - All ports outputs.

Example : set.io.iot332(to.iot332, (BYTE #00))

This example sets all the ports as inputs.

FUNCTION : **read.block.iot332**

Syntax : PROC read.block.iot332(CHAN OF BYTE in, out, VAL BYTE block, BYTE data)

Description : This function reads one of four bytes (one of the input ports) from the iot332.

Parameters : in: This is the software input channel
out: This is the software ouput channel
block: The byte to read, where... 0 = port 0
1 = port 1
2 = port 2
3 = port 3
data: This will hold the value read

Example :
```
BYTE val:
SEQ
 set.io.iot332(to.iot332, (BYTE #00))
 read_block_iot332(from.iot332, to.iot332, (BYTE 0), val)
```

This example reads port 0.

FUNCTION : **write.block.iot332**

Syntax : PROC write.block.iot332(CHAN OF BYTE out, VAL BYTE block, VAL BYTE data)

Description : This function writes one of four bytes to the iot332 (sets the state of the output ports).

Parameters : out: The software output channel
block: The port to write to, where... 0 = port 0
1 = port 1
2 = port 2
3 = port 3

data: The data to write.

Example : set.io.iot332(to.iot332, (BYTE #0F))
write.block.iot332(to.iot332, (BYTE #00), (BYTE #FF))

This example sets all of the outputs on port 0 to be high.

FUNCTION : **iot332.id**

Syntax : PROC iot332.id(CHAN OF BYTE in, out, BYTE id)

Description : This function returns the id for the iot332.

Parameters : in: The software input channel
out: The software output channel
id: The id returned (#33)

Example : BYTE ID:
SEQ
iot332.id(from.iot332, to.iot332, ID)

This example reads the ID.