I2C Interfaces for Minerva RTC (MKII)

These are the first in a series of I2C interfaces designed to plug into the 9 pin D female connector fitted to current Minerva RTC (MKII) production. If your Minerva RTC doesn't have a connector, then connect 9 way IDC cable to a female 9 pin D IDC connector (see appendix I).

I2C stands for Inter-IC Communication, and the network was designed by Philips to reduce the number of connections between integrated circuits. All ICs that use the I2C bus are basically connected to three lines - a data line (SDA) and clock line (SCL) and ground line. Although there are a wide variety of circuits, all commands and data are carried on the single SDA line. The protocol for doing this is complex, but this is handled by the QL extension I2C_IO. Details of how this extension is used are in the Minerva manual. There has to be at least one master - the QL in this case. Philips specification allows more than one master, but the QL implementation allows one only - the QL itself.

We plan (time permitting) to write more software, particularly for using the analogue interface. These will include meter and oscilloscope emulators. The latter will be of fairly low frequency/resolution. Also the routines for x/y plotting of input from a waveform analyser will be very easy to write, using Superbasic graphics. These will be available to early purchasers for a nominal charge. We would be interested to see any software that is currently available.

Suggested applications:

- **Model train controllers** The 8 bit parallel port (half of the interface) should connect directly to TTL (low voltage logic input) commercial controllers.

- **Robot control** The parallel output can be used to control motors and switches (if solid state switches are connected -these are also fitted to the power driver board). The parallel input can be used for switch feedback. Analogue input can be used for position input via potentiometers. This is used by us (using the program HANOI_MIN_BAS on the Minerva utilities disk) to control a robot (Fischer Technik kit) which solves Towers of Hanoi. Mind you the plastic components have worn out and the robot is being remade in metal.

- **Mains switches** If solid state switches/appropriate relays/triacs are fitted to the parallel output, the these can be used for mains or low voltage switching. This is the power driver (see later in manual). A protoytpe was used to drive a flashing light display over Xmas 1993 in Port Isaac, Cornwall.

.

- **Process control** The power driver can be used for any switching application.

- **Temperature measurement** With a suitable probe, the analogue interface can be used for temperature measurement. This will need a resistor bridge to adjust the input scale, and a voltage reference if high stability is needed.
- **Liquid crystal displays** The parallel output can drive LCD displays directly.

I would very much welcome suggestions for further circuits and applications.

Getting started

The first thing to do is make a backup copy of the files on the utilities disk. You can use clone_bas on the disk for this, or just make a direct copy of all the files. Do not load or run any programs directly from the master.

Files on the disk are:

• Clone_bas	Makes backup copues of master
• i2c_bas	Source for Superbasic extensions
• i2c_obj	Extension code (compiled into resident procedures with the excellent Qliberator)
• 12c_io_bin	I2C driver extension (also on the Minerva disk)
• boot	Loads the extensions
• Updates_doc	Manual updates
• Scope4_bas	A simple 4 channel oscilloscope
• Wildgruber_zip	Some ideas
• Goodwin_zip	Simon Goodwin's analogue data

To load the superbasic extensions for accessing the I2C devices, use the following, preferably in your boot program, although Minerva allows this to be done any time:

a=RESPR(file_size):lbytes `flp1_i2c_obj',a:call a

or, with toolkit II: **LRESPR `flp1_i2c_obj'**

(see updates_doc for extensions file size)

The extensions are also provided as Basic source (i2c_bas). It is better to merge these basic extensions into your own basic program, as if there are errors calling external QLIB procedures, then QLIB will often trap these errors itself - not too helpful. You must of course, RESPR the `i2c_io_bin' extension provided with Minerva (also on the disk with these interfaces).

These interfaces are designed to mate together with 15 pin `D' connectors built into the case via the I2C bus. A lead can be made or supplied to allow interfaces to be positioned on top of others (not recommended for power driver).

A QL with Gold or Trump card expansions, for instance, could power at least 5 interfaces without problems. If more than this number are connected, then external DC supply should be supplied using the spare I2C socket (wiring details in the appendix).

If possible this supply should be around 4.5v, to avoid the risk of the supply being greater than the QL supply. Alternatively the QL 5v rail can be increased slightly, by putting a 1N4001 diode in series between the QL regulator ground (centre) and QL ground. This method is at your risk - only do it if you are SURE of what you are doing. Tony Tebby also recommended doing this to improve reliability of the network.

Also if you are connecting more than 5 interfaces, contact us for advice about possible changes to the pull_up resistors on the I2C data/clock lines (SCL/SDA). These extras resistors, if needed, are again fitted to the spare I2C socket.

Do not unplug or plug in the interfaces with the QL power on, as this can damage the QL and maybe the interfaces.

Where an interface has address switches, then port 1 address is set using switches 1 to 3, and port 2 is set with switches 4 to 6. This is a binary number, added to the 4 high bit base address of the device. (ON/down = 0, OFF/up = 1), giving 8 addresses per port.

000	001	010	011	100	101	110	111
0	1	2	3	4	5	6	7

eg The third address (base+2) is 1-DOWN, 2-UP, 3-DOWN, or 4-DOWN, 5-UP and 6-DOWN. In other words, the dip switches visually represent the binary number. All very elementary stuff if you are familiar with binary numbers!

ANALOGUE INTERFACE

See the appendix 1 for pinouts for the ports.

The addresses of the PCF8591P chip in the interface range from 72 to 79.

Analogue input

The four input lines read input voltages, and relate them to VREF and AGND. VREF/AGND can be set to VCC(5v) and GND(0v), which gives full resolution if the input is in that range. If however you wish to reduce the range to match an input device, then a resistance bridge can be used.

```
eg          R1          R2          R3
       |-----/\/\/---|---/\/\/\---|---/\/\/\------|    R1+R2+R3=10k
     gnd       agnd       vref            vcc
```

R1 = AGND * 2k
R2 = 2k * (VREF - AGND)
R3 = 10k - R1 - R2

For high accuracy, a voltage reference can be derived from a high precision reference (eg. MAPLIN YH39N for 1.2V). However for most purposes, the 5v reference will be accurate enough.

in$ = READ_ADC$(address%)

This function returns a 4 character string, each byte being the value of each input line, in the range 0 to 255. Byte 1 is input 1 etc. If address is out of range, then a single character NUL string is returned.

Address% can be in range 0-7 or 72-79.

eg for address 74:

```
10  a$=READ_ADC$(2)
20   IF a$='': PRINT `Address error'
```

ALWAYS check for the NUL string error return.

a$(1) = input 1, a$(2) = input 2, a$(3) = input 3, a$(4) = input 4

QDOS error returns (applies to all functions):

NOT FOUND - The function did not get an acknowledge from a device at that address.

FORMAT FAILED - The device was found, but there was some error.

These will have to be trapped separately, using QLIB or Minerva error trapping routines.

Analogue Output

The output voltage on the I2C analogue chip is in the range AGND to near VREF based on 0 to 255 data from the QL. See data sheet booklet for further information. Note: AGND and VREF need setting - normally tied to 0v and 5v respectively.

error_no% = WRITE_DAC%(address%,dac$)

This function writes dac$ to address%. Address rules as for READ_ADC$(). If address% or dac$ is out of range -15 is returned, otherwise 0. Maximum length of dac$ is approximately 32730.

eg: 10 if WRITE_DAC%(1,dac$) <> 0: print ``Address or value out of range'

This writes ASCII values of dac$ to device at address 1.

Machine code programmers will find that the vector II_DRIVE (Minerva manual I2C section), which accepts the same parameters as the basic extension, will provide more continuous data output. The maximum string length is limited only by ram, with only a slight glitch every 64k output.

PARALLEL INTERFACE

See appendix for pinouts.

Address range of each of the PCF8574A chips in the interface is 56 to 63.

This provides 16 lines, which can accept both input and generate output. As each port can be either input or output, the control program AND any input controllers must be `aware' of the affect on each. In most cases, if a port is used for input, it is easier not to use that for output, unless it is intended to control output directly in this way.

Parallel output

The output specs are listed on a separate booklet. They are relatively high current outputs, and can be used to drive LEDs/LCDs or solid state switches (eg L298) directly (see power driver later).

This is the most straightforward, and each bit of a byte of data acts on one pin.

For instance, `U', which is binary 01010101 produces the following:

P0 high, P1 low, P2 high, P3 low and so on.

error_no% = WRITE_PAR%(address%,value%)

where address% is 0-7 or 56-63, and value% is 0-255. A parameter error will return -15, otherwise 0.

QDOS errors as for analogue port.

Parallel input

Reference to `line' in the following is one i/o pin of the I2C chip. (See appendix for pinouts and pin descriptions)

This is the least straightforward to use. Each input line can be read individually, but must be initially HIGH. This is the case at power-on. If necessary the line read must be set to HIGH just before a read using WRITE_PAR% Once set, the line will remain high unless forced LOW by an outside input, or changed using WRITE_PAR%. The outside world can then set each of the 16 inputs low. This needs careful handling if an output is also connected to the line. In most cases, it will be more straightforward not to use a line for both input and output, but some layouts may be able to use this. For instance, an external device could be controlled both by the QL via output, and by an external device also connected to the same line.

The I2C chip will generate an interrupt (INT) if it detects a low (0v) `forced' on it by an input. This INT line is provided on the cabling, but will need special handling by the QL. The easiest way would be to use one of the spare input pins on Hermes (replacement 8049), and pull up the line to 5v with a 10k resistor. This pin would then have to be polled by software commands. In theory the QLs EXTINTL line could be used, with appropriate software and again pulling up to 5v with a 10k resistor - this will be left to the experienced user!

The easier method is to forget INT completely, and simply look at the inputs (poll them) periodically. In most real situations, the outside world will usually be much slower than the QL, even in uncompiled basic!

in% = READ_PAR%(address%)

where address% is 0-7 or 56 to 63.

ie in%=READ_PAR%(57) or in%=READ_PAR%(1) will read a byte from a parallel device at address 1 The result is in `in%'. If positive, then the appropriate bits will give the state of EVERY line. -15 return will be an invalid address. QDOS errors as for analogue interface.

POWER DRIVER INTERFACE

The circuit for the I2C part of this interface is identical to the parallel interface described earlier. One or both ports can be used in exactly the same way as described above. The pinouts for the input/logic output signals are the same.

Also included on the circuit is a dual full bridge driver. This acts as a high power switch, and uses an external power supply (see appendix for specs).

We supply two versions:

High current (up to 4a total - 7.5v to 46v) - L298 driver

Low current (up to 2a total - 7.5v to 35v) - L293 driver

This provides 8 high current outputs. If one motor is connected to each output with a common ground, then eight motors can be switched on uni-directionally. If four motors only are connected, then they can be reversed.

If only up to about 1.5 amps total current (if all motors are on) is expected, then the DC power connector (centre GND - 2.5mm) on the left hand side of the interface can be used (power will be connected to any other power driver interfaces connected). If more than this current is expected, then use the three power pins (6/7/8) and GND on the 15 pin I2C interface. Ground should be also connected to the metal sleeve on the `D' plug. Each power interface is individually fused - replace fuse with the appropriate fuse (low power - 2A. High power - 4A). IDC cable must NOT be used to connect two power drivers.

The interface has two full bridge drivers, each of which has two ports, each with two outlets. In addition to 4 inputs each, there are two enable lines for each device, which can be set from the QL or externally. These enable lines are inverted (ie 0 is enable) as the parallel I2C starts up with all pins HIGH.

eg if device 1 enable A is low, high/low on inputs 1 and 2 (bits 1/2) get converted to Motor supply/Ground respectively on the power outputs 1/2.

The function WRITE_PAR% is used to control power and TTL output. READ_PAR% is used for TTL input. (see parallel interface).

Function of bits is as follows:

bit	function		
0	TTL I/O or	CE1/CSA	enable 1 (for power select 1/2) - 0=enable
1	TTL I/O or	IN1	power select 1
2	TTL I/O or	IN2	power select 2
3	TTL I/O or	CE2/CSB	enable 2 (for power select 3/4) - 0=enable
4	TTL I/O or	IN3	power select 3
5	TTL I/O or	IN4	power select 4
6	TTL I/O		
7	TTL I/O		

A High (1) or LOW (0) on bits 1/2/4/5 will give Power voltage HIGH/Ground respectively on the power output pins. See appendix for complete pinouts.

Relay board

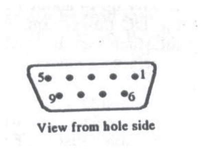
Inside the case are three connectors (NO - normally open. NC - normally closed. C - common) for each relay. Maximum suggested load current is 3 amps. If motors are driven, use a suppression capacitor across the motor terminals. The relays are suitable for up to 24vDC and 240V AC. Use the supplied cable ties inside the case on each wire to provide strain relief.

APPENDIX 1

Minerva RTC (MKII) I2C port pinouts

9 pin D socket

1	VBAT	This will be
6	VCC (+5V)	the order of wires
2	GND	on the IDC cable
7	SCL	
3	SDA	



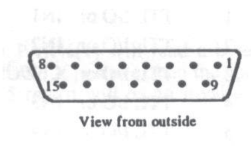
If you don't have such a connector fitted to your Minerva RTC (MKII), then connect about 6 inches of 9-way IDC (Insulation displacement connector) cable to a 9 pin IDC socket and solder the first 5 wires (starting at pin 1) to Minerva RTC (MKII) I2C output holes. Pin 1 (VBAT) is nearest the back of the board (next to the cylindrical clock crystal). Fold back and tape the remaining 4 wires to the cable. This cable is available from us.

I2C Interface connector wiring

15 pin IDC plug (interface to QL)

1	VBAT
9	VCC (5v)
2	GND
10	SCL
3	SDA
11	INT (flying lead at QL socket)
6/7/8	Power input (for power driver)

4/5 & 12-15 - reserved for future use.



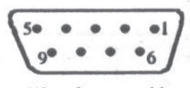
The 15 pin socket is used for interconnection between interfaces. Other interfaces can be connected directly side by side, or with ribbon cable and two IDC connectors. In this latter case, interfaces can sit on top of one another.

(NOT power driver)

Analogue input/output ports

9 pin D socket

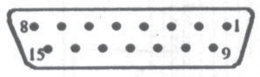
1	Input 1
2	Input 2
3	Input 3
4	Input 4
5	Output
6	VCC (+5v)
7	VREF - analogue voltage reference
8	AGND - analogue ground reference
9	GND



View from outside

Parallel input/output ports

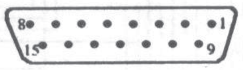
1	Input/output 1
2	Input/output 2
3	Input/output 3
4	Input/output 4
5	Input/output 5
6	Input/output 6
7	Input/output 7
8	Input/output 8
9-12	GND
13-15	VCC (+5v)



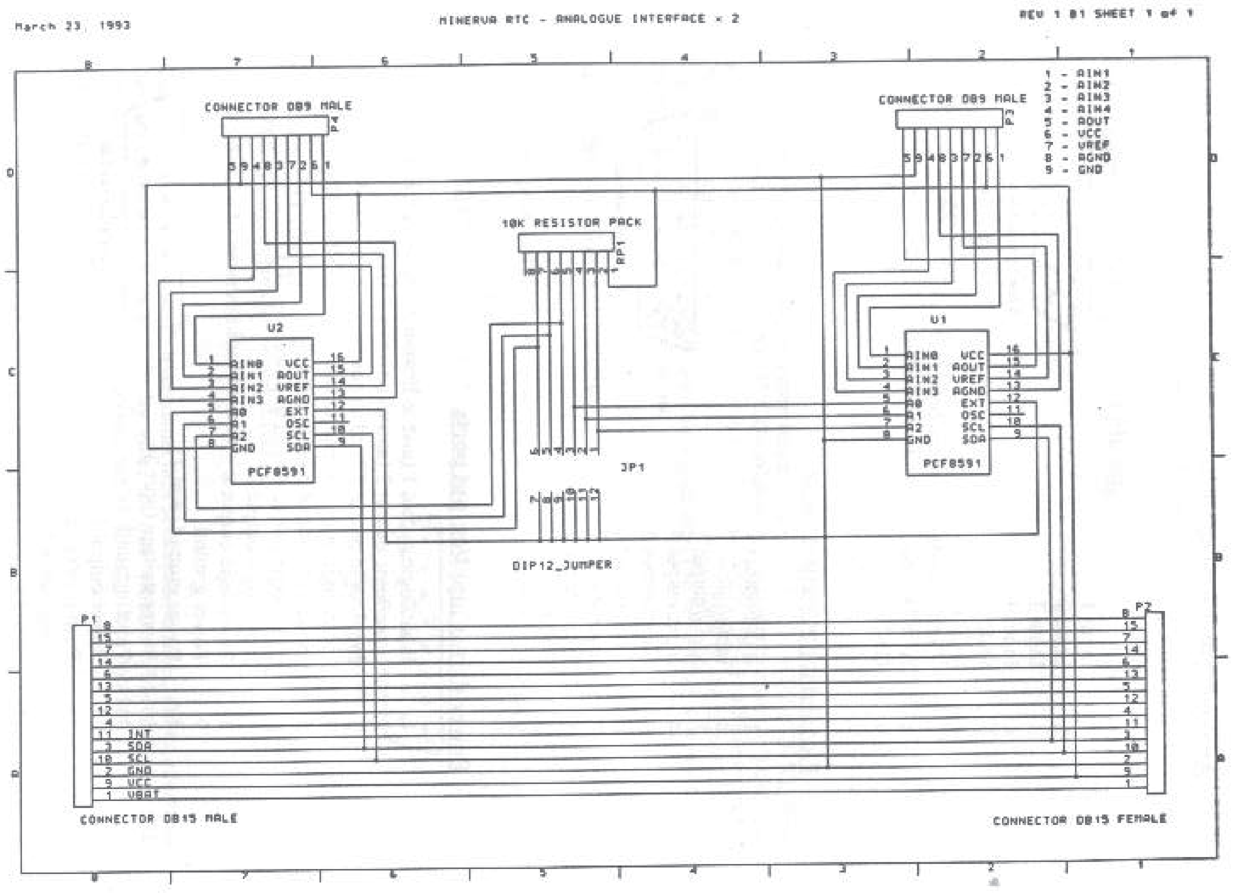
View from outside

Power driver input/output ports

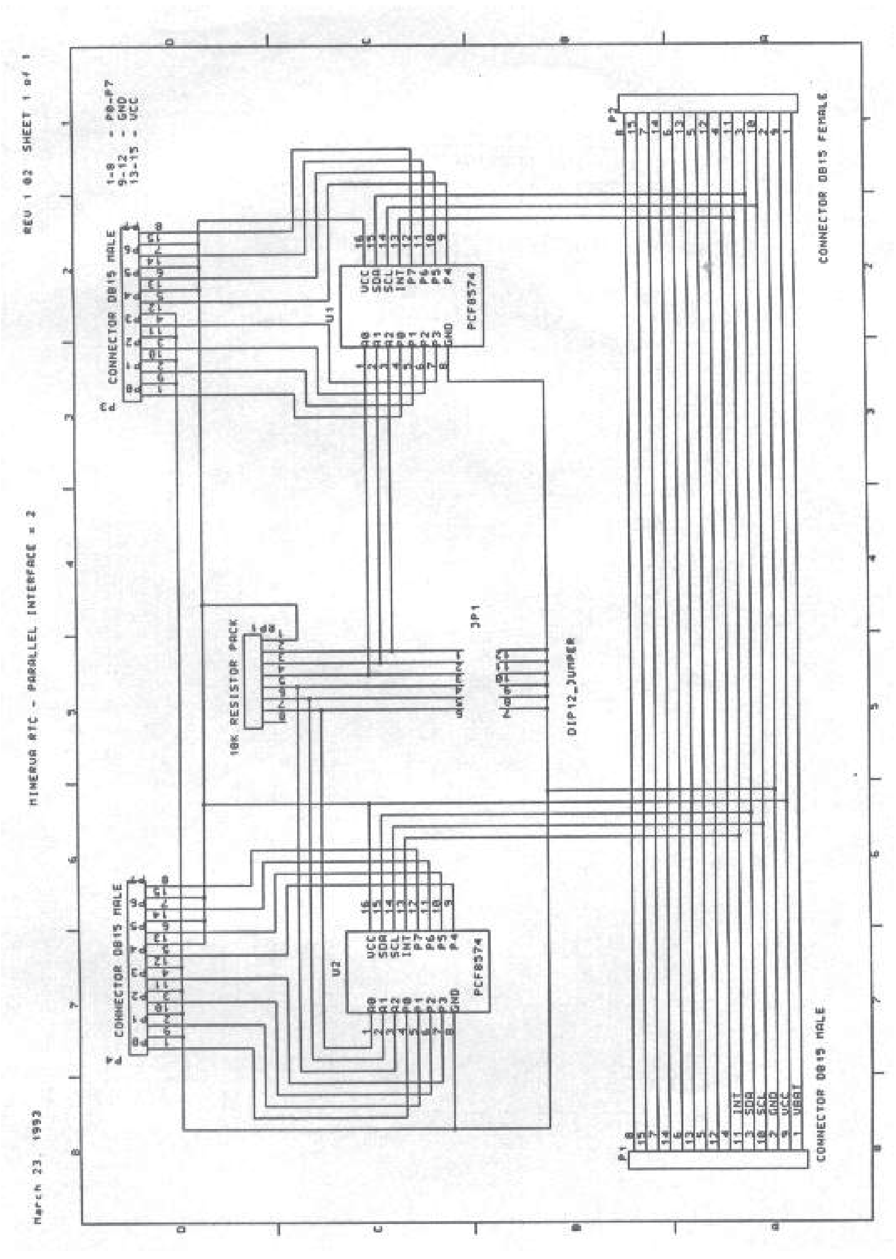
1	Input/logic output 1 (and to Enable A - CSA or CE1)
2	Input/logic output 2 (and to power out 1A)
3	Input/logic output 3 (and to power out 2A)
4	Input/logic output 4 (and to Enable B - CSB or CE2)
5	Input/logic output 5 (and to power out 1B)
6	Input/logic output 6 (and to power out 2B)
7	Input/logic output 7
8	Input/logic output 8
9	Power ground
10	Power ground (port 1)
10	Power voltage (port 2)
11	Power ground
12	Power output 1
13	Power output 2
14	Power output 3
15	Power output 4



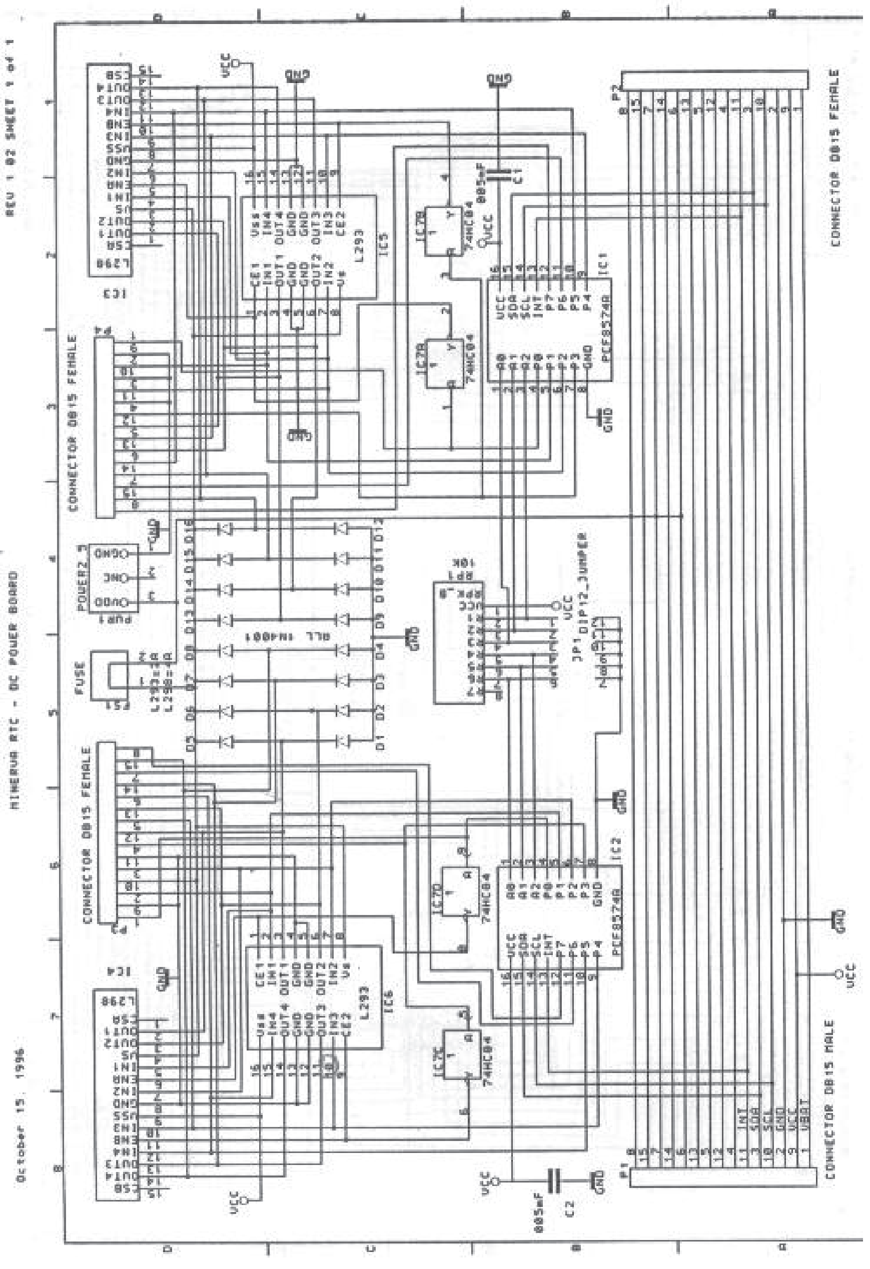
View from outside



March 23, 1993
MINERVA RTC - ANALOGUE INTERFACE × 2
REV 1 01 SHEET 1 of 1
CONNECTOR DB9 MALE
P4
CONNECTOR DB9 MALE
P3
1 - AIN1
2 - AIN2
3 - AIN3
4 - AIN4
5 - AOUT
6 - VCC
7 - VREF
8 - AGND
9 - GND
10K RESISTOR PACK
RP1
U2
U1
AIN0 AIN1 AIN2 AIN3 A0 A1 A2 GND
VCC AOUT VREF AGND EXT OSC SCL SDA
PCF8591
JP1
DIP12_JUMPER
P1
P2
INT
SDA
SCL
GND
VCC
VBAT
CONNECTOR DB15 MALE
CONNECTOR DB15 FEMALE



March 23, 1993
MINERVA RTC - PARALLEL INTERFACE = 2
REV 1 @2 SHEET 1 of 1
1-8 - P0-P7
9-12 - GND
13-15 - VCC
CONNECTOR DB15 MALE
P3
U1
PCF8574
VCC
SDA
SCL
INT
P7
P6
P5
P4
A0
A1
A2
P0
P1
P2
P3
GND
10K RESISTOR PACK
JP1
DIP12_JUMPER
CONNECTOR DB15 MALE
P4
U2
PCF8574
P2
CONNECTOR DB15 FEMALE
P1
INT
SDA
SCL
GND
VCC
VBAT
CONNECTOR DB15 MALE


MINERVA RTC - DC POWER BOARD
REV 1.02 SHEET 1 of 1
October 15, 1996
CONNECTOR DB15 FEMALE
CONNECTOR DB15 MALE
L298
L293
PCF8574A
74HC04
ALL 1N4001
DIP12_JUMPER
FUSE
POWER2_5
GND
VCC