Overall FS for

for

Multifunction Valve Pack Controller Card

April 2012
SMD Ref: SMDH MVP Overall FS.doc

ISSUE CHANGE REGISTER

File Name	Last Saved by
SMDH MVP Overall FS.doc	Keith Lawson

No:	Issue Date	Change Details	Issued to:
0	28 Feb 2007	Initial issue	SMDH
1	11 June 2013	Added sensor Input data	SMD
2			
3			
4			
5			
6			

CONTENTS

1. INTRODUCTION

This document specifies set up and configuration of the MFVP PCB. This document details MFVP address set up, analogue output conditioning and analogue input conditioning.

The MFVP contains the following functions:

- 15 A / B Proportional valve controls software controlled
- 10 Analogue inputs, which is jumper selectable to be 4-20mA or 0-10V inputs
- Transducer excitation supplies, software switchable in pairs. Transducer excitation 9 and 10 is switchable between 12V and 24V excitation.
- 1 Dedicated moisture input
- 1 Dedicated PT100 temperature input

There are two revisions of this PCB

1. **Lower Res** – AA0439

2. **High Res** – AD7057 – Features higher resolution outputs only.

The difference between the two is the OUTPUT resolution on the 12 servo drive functions.

2. DIMENSIONS AND HOOKUP

The following is an example of basic hookup showing pin designations. Power and communications are applied via J1. Separate power and control supplies are provided; the control will nominally run off 24v Regulated and the Field

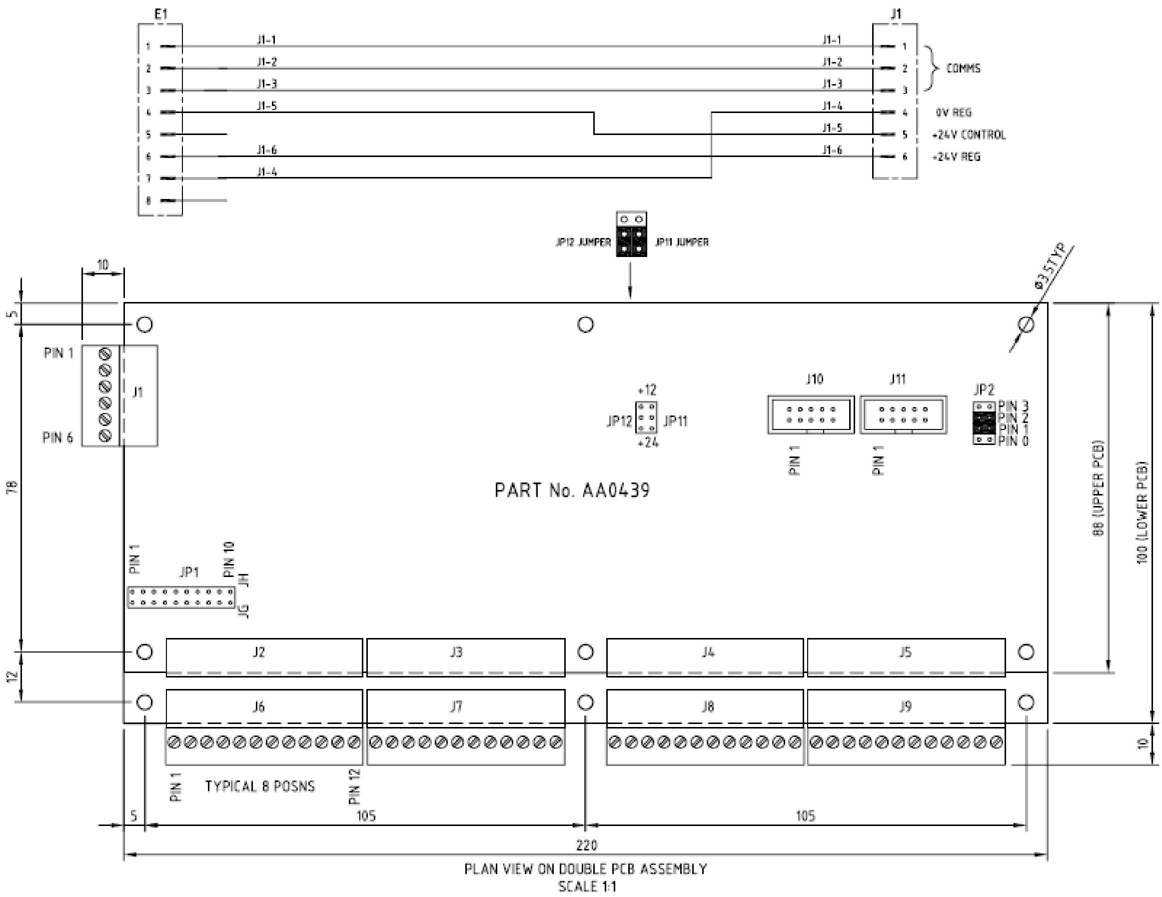


Figure 1 - MFVP PCB layout

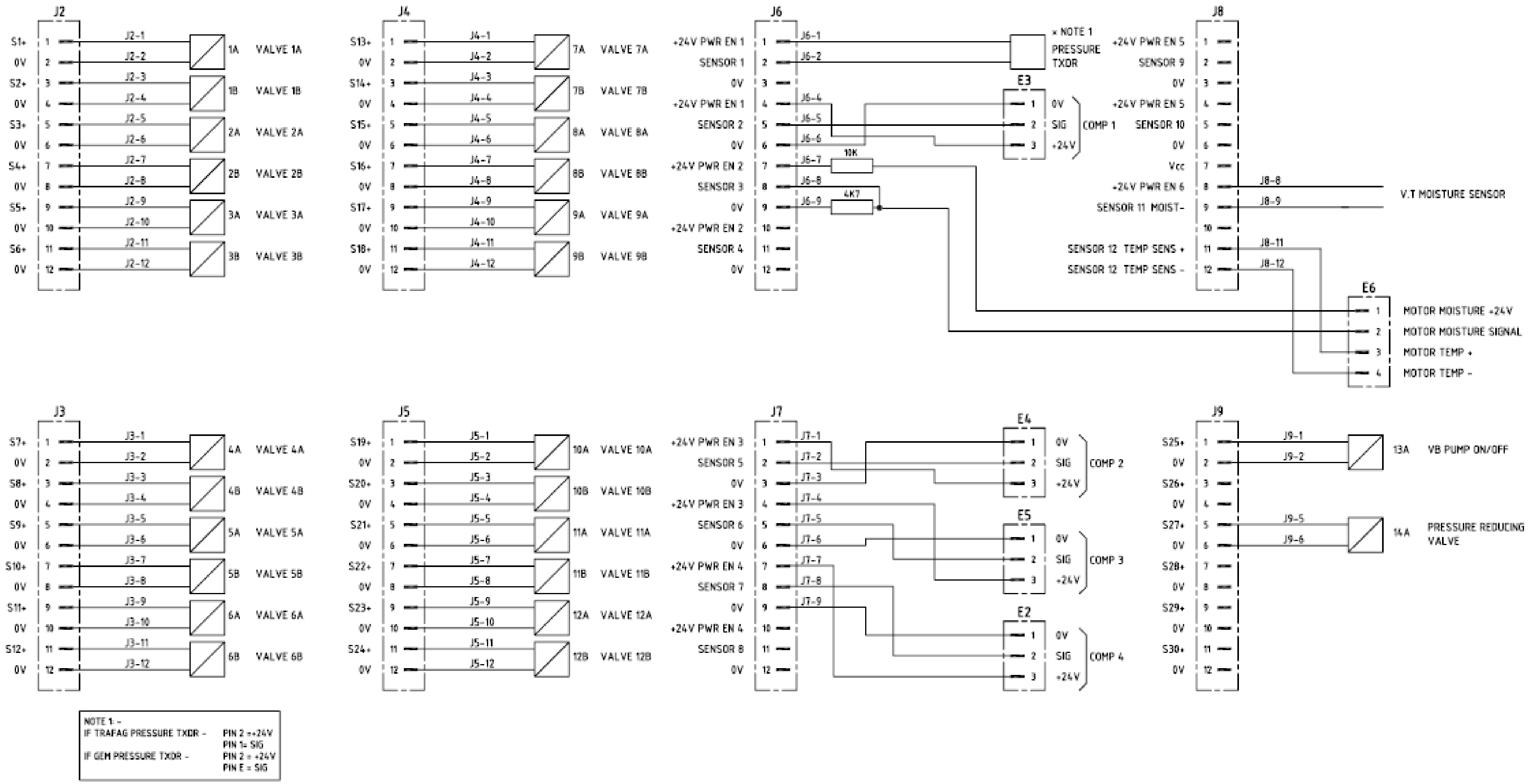


Figure 2 - Typical connector wiring

The connectors for the MFVP are split between the upper and lower PCB connector J1 – J5 are on the upper PCB, J6 – J9 are on the lower PCB. The function of each of the connectors is defined as follows

7.1 Connector J1 Configuration

Connector J1 supplies power and communication to the MFVP, the connector is detailed as follows.

Connector Number	Description
1	RS485_A
2	RS485_B
3	RS485_GND
4	0V
5	+24V Control Voltage
6	+24V Output Voltage

7.2 Connector J2 Configuration

Connector J2 is the solenoid drive for directional solenoids 1 -3

Connector Number	Description
1	Solenoid 1A +Supply
2	Solenoid 1A 0v
3	Solenoid 1B +Supply
4	Solenoid 1B 0v
5	Solenoid 2A +Supply
6	Solenoid 2A 0v
7	Solenoid 2B +Supply
8	Solenoid 2B 0v
9	Solenoid 3A +Supply
10	Solenoid 3A 0v
11	Solenoid 3B +Supply
12	Solenoid 3B 0v

7.3 Connector J3 Configuration

Connector J3 is the solenoid drive for directional solenoids 4 -6

Connector Number	Description
1	Solenoid 4A +Supply
2	Solenoid 4A 0v
3	Solenoid 4B +Supply
4	Solenoid 4B 0v
5	Solenoid 5A +Supply
6	Solenoid 5A 0v
7	Solenoid 5B +Supply
8	Solenoid 5B 0v
9	Solenoid 6A +Supply
10	Solenoid 6A 0v
11	Solenoid 6B +Supply
12	Solenoid 6B 0v

7.4 Connector J4 Configuration

Connector J4 is the solenoid drive for directional solenoids 7 -9

Connector Number	Description
1	Solenoid 7A +Supply
2	Solenoid 7A 0v
3	Solenoid 7B +Supply
4	Solenoid 7B 0v
5	Solenoid 8A +Supply
6	Solenoid 8A 0v
7	Solenoid 8B +Supply
8	Solenoid 8B 0v
9	Solenoid 9A +Supply
10	Solenoid 9A 0v
11	Solenoid 9B +Supply
12	Solenoid 9B 0v

7.5 Connector J5 Configuration

Connector J5 is the solenoid drive for directional solenoids 10 -12

Connector Number	Description
1	Solenoid 10A +Supply
2	Solenoid 10A 0v
3	Solenoid 10B +Supply
4	Solenoid 10B 0v
5	Solenoid 11A +Supply
6	Solenoid 11A 0v
7	Solenoid 11B +Supply
8	Solenoid 11B 0v
9	Solenoid 12A +Supply
10	Solenoid 12A 0v
11	Solenoid 12B +Supply
12	Solenoid 12B 0v

7.6 Connector J6 Configuration

Connector J6 contains the signal excitation and analogue inputs for sensors 1 - 4

Connector Number	Description
1	Sensor 1 supply 24V
2	Sensor 1 analogue input (0-10V / 4-20mA)
3	Sensor 1 0v
4	Sensor 2 supply 24V
5	Sensor 2 analogue input (0-10V / 4-20mA)
6	Sensor 2 0v
7	Sensor 3 supply 24V
8	Sensor 3 analogue input (0-10V / 4-20mA)
9	Sensor 3 0v
10	Sensor 4 supply 24V
11	Sensor 4 analogue input (0-10V / 4-20mA)
12	Sensor 4 0v

7.7 Connector J7 Configuration

Connector J7 contains the signal excitation and analogue inputs for sensors 5 - 8

Connector Number	Description
1	Sensor 5 supply 24V
2	Sensor 5 analogue input (0-10V / 4-20mA)
3	Sensor 5 0v
4	Sensor 6 supply 24V
5	Sensor 6 analogue input (0-10V / 4-20mA)
6	Sensor 6 0v
7	Sensor 7 supply 24V
8	Sensor 7 analogue input (0-10V / 4-20mA)
9	Sensor 7 0v
10	Sensor 8 supply 24V
11	Sensor 8 analogue input (0-10V / 4-20mA)
12	Sensor 8 0v

7.8 Connector J8 Configuration

Connector J8 contains the signal excitation and analogue inputs for sensors 9 – 10, moisture and temperature.

Connector Number	Description
1	Sensor 9 supply 12 / 24V (switchable)
2	Sensor 9 analogue input (0-10V / 4-20mA)
3	Sensor 9 0v
4	Sensor 10 supply 12 / 24V (switcahble)
5	Sensor 10 analogue input (0-10V / 4-20mA)
6	Sensor 10 0v
7	Not connected
8	Moisture sensor +ve supply
9	Moisture sensor signal
10	Not connected
11	Temp sensor +ve
12	Temp sensor -ve

7.9 Connector J9 Configuration

Connector J9 is the solenoid drive for directional solenoids 13 -15

Connector Number	Description
1	Solenoid 13A +Supply
2	Solenoid 13A 0v
3	Solenoid 13B +Supply
4	Solenoid 13B 0v
5	Solenoid 14A +Supply
6	Solenoid 14A 0v
7	Solenoid 14B +Supply
8	Solenoid 14B 0v
9	Solenoid 15A +Supply
10	Solenoid 15A 0v
11	Solenoid 15B +Supply
12	Solenoid 15B 0v

8 SOFTWARE INTERFACING

The following IO table controls the MVP LO RES PCB. **Note** – QNX software drivers normalised all sensor input / output values to 10000. This isn't done in the PLC software i.e if thei was a 10 bit DAC of 0-1023 raw in QNX this appears as 0-10K; in the PLC this isn't pre-scaled and the raw value is the actual raw value.

Assignment	**Type**	**Description**	**Suggested scaling**					
			QNX raw min	QNX raw max	PLC raw min	PLC raw max	scale min	scale max
J2-1,2 Valve 1A Enable	DO	See note 1	0	1			0	1
J2-3,4 Valve 1B Enable	DO		0	1			0	1
J2-5,6 Valve 2A Enable	DO		0	1			0	1
J2-7,8 Valve 2B Enable	DO		0	1			0	1
J2-9,10 Valve 3A Enable	DO		0	1			0	1
J2-11,12 Valve 3B Enable	DO		0	1			0	1
J3-1,2 Valve 4A Enable	DO		0	1			0	1
J3-3,4 Valve 4B Enable	DO		0	1			0	1
J3-5,6 Valve 5A Enable	DO		0	1			0	1
J3-7,8 Valve 5B Enable	DO		0	1			0	1
J3-9,10 Valve 6A Enable	DO		0	1			0	1
J3-11,12 Valve 6B Enable	DO		0	1			0	1
J4-1,2 Valve 7A Enable	DO		0	1			0	1
J4-3,4 Valve 7B Enable	DO		0	1			0	1
J4-5,6 Valve 8A Enable	DO		0	1			0	1
J4-7,8 Valve 8B Enable	DO		0	1			0	1
J4-9,10 Valve 9A Enable	DO		0	1			0	1
J4-11,12 Valve 9B Enable	DO		0	1			0	1
J5-1,2 Valve 10A Enable	DO		0	1			0	1
J5-3,4 Valve 10B Enable	DO		0	1			0	1
J5-5,6 Valve 11A Enable	DO		0	1			0	1
J5-7,8 Valve 11B Enable	DO		0	1			0	1
J5-9,10 Valve 12A Enable	DO		0	1			0	1
J5-11,12 Valve 12B Enable	DO		0	1			0	1
J9-1,2 Valve 13A Enable	DO		0	1			0	1
J9-3,4 Valve 13B Enable	DO		0	1			0	1
J9-5,6 Valve 14A Enable	DO		0	1			0	1
J9-7,8 Valve 14B Enable	DO		0	1			0	1
J9-9,10 Valve 15A Enable	DO		0	1			0	1
J9-11,12 Valve 15B Enable	DO		0	1			0	1
J6-1 & J6-4 PSU	DO	See note 2	0	1			0	1
J6-7 & J6-10 PSU	DO		0	1			0	1
J7-1 & J7-4 PSU	DO		0	1			0	1
J7-7 & J7-10 PSU	DO		0	1			0	1
J8-1 & J8-4 PSU	DO		0	1			0	1

J8-8 Moisture Sensor PSU	DO		0	1			0	1
J6-2 Signal	AI	See note 3	2000	10000	204.8	1024	0	400
J6-5 Signal	AI		1269	9900	129.9456	1013.76	100	0
J6-8 Signal	AI		1250	4920	128	503.808	0	100
J6-11 Signal	AI							
J7-2 Signal	AI							
J7-5 Signal	AI							
J7-8 Signal	AI							
J7-11 Signal	AI							
J8-2 Signal	AI							
J8-5 Signal	AI							
J8-9 Moisture Signal	AI		0	7600	0	778.24	0	100
J8-11 & 12 PT100 Signal	AI		4452	6151	455.8848	629.8624	0	100
PCB Temp	AI	see note 4	0	7000	0	716.8	0	100
PCB 24V	AI	see note 5	4000	7500	409.6	768	16	30
J2-1,2,3,4 Valve 1 AO	AO	See note 6	0	10000	0	100	0	100
J2-5,6,7,8 Valve 2 AO	AO							
J2-9,10,11,12 Valve 3 AO	AO							
J3-1,2,3,4 Valve 4 AO	AO							
J3-5,6,7,8 Valve 5 AO	AO							
J3-9,10,11,12 Valve 6 AO	AO							
J4-1,2,3,4 Valve 7 AO	AO							
J4-5,6,7,8 Valve 8 AO	AO							
J4-9,10,11,12 Valve 9 AO	AO							
J5-1,2,3,4 Valve 10 AO	AO							
J5-5,6,7,8 Valve 11 AO	AO							
J5-9,10,11,12 Valve 12 AO	AO							
Not Fitted Valve 13A&B Value	AO							
Not Fitted Valve 14A&B Value	AO							
Not Fitted Valve 15A&B Value	AO							

Note 1

Digi Outs only enable valve 1; there needs to be an analogue demand for the valve to do anything also. The output is 24v PWM to drive up to an amp. Output is varied using the raw channel of the associated AO channel. 0-10000 (QNX) or 0-100 (PLC) providing 0-1A output.

There are 15 valve drive ccts and they all behave the same way. Usually only 12 are used on most jobs though.

Typically these drive solenoid valves; if we need a hydraulic function to drive in 2 directions we fit 2 solenoids one for each direction and energise one DO at a time.

Note 2

These digital outputs are used for enabling sensors connected to these connectors.

Note 3

The next 10 analogue inputs can be scaled as 0-10V or 4-20ma. Jumper settings on the board decide this on JP1. If the link is in it's 4-20 else it's 0-10.

If it's 4-20ma set raw min/max to 2000 / 10000 (QNX) and Smin / Smax to 4 / 20 to read 4-20ma.

If it's 0-10 then 0-10000 (QNX) 0-1024 (PLC) relates to 0-10V

Note 4-5

This AI channel is embedded in the PCB and does not appear at an edge connector

Note 6

0-10000 (QNX) 0-100 (PLC) giving 0-100% drive. Of 1A and 1B. The appropriate DO must be energised for anything to happen on the hydraulics.

The Drive LEDs will only come on if there is a drive demand on the AO and DO.

For the higher resolution card this is 0-255 rather than 0-100 on the original cards; in the QNX the raw value has to be set to 0-25600 to get the full range. Note; the PCB only actually drives 85 – 230.

.

8.1 PCB temp scaling

The following suggested non linear scaling (amazingly) can be used for the PCB temperature.

ChannelName=PCB Temp	QNX Raw	PLC Raw	PCB Temp
Point1	870	89.088	-40
Point2	2023	207.1552	-20
Point3	3607	369.3568	0
Point4	4975	509.44	20
Point5	5845	598.528	40
Point6	6324	647.5776	60
Point7	6569	672.6656	80
Point8	6696	685.6704	100

For the PLC this can be fed into an equation ***Temp = 2E-06Raw^3 - 0.0016Raw^2 + 0.5764Raw - 81.69*** (as opposed to the traditional Temp x 10).

8.2 AA0439 output scaling.

The original MFVP PCB AA0439 has a effective resolution of 34 output values including Off and 100% On. Each output value 'step' will be the response to at least 2 input demand values. Its reaction to input demands is tabulated below:-

STEP	DEMAND / register val
1	0 to 9
2	10,11
3	12,13,14
4	15,16
5	17,18,19
6	20,21,22
7	23,24
8	25,26,27
9	28,29,30
10	31,32
11	33,34,35
12	36,37,38
13	39,40
14	41,42,43
15	44,45,46
16	47,48
17	49,50,51
18	52,53,54
19	55,56
20	57,58,59
21	60,61,62
22	63,64
23	65,66,67
24	68,69,70
25	71,72
26	73,74,75
27	76,77,78
28	79,80
29	81,82,83
30	84,85,86
31	87,88
32	89,90,91
33	92,93,94
34	95 to 100

As an added complication to be aware of; the absolute accuracy of the output to any demand can also have a large error associated with it.

step	Register val	measured drive(%)	Max error
2	10,11,12	11.4	1.4

6	20,21,22	19.2	-2.8
9	28,29,30	25.4	-4.6
13	39,40	34.3	-5.7
17	49,50,51	43.8	-7.2
21	60,61,62	54.2	-7.8
24	68,69,70	62.6	-7.4
28	79,80	74.7	-5.3
32	89,90,91	87.0	-4.0
34	95 to 100	100.0	0.0

8.3 Hi-res PCB output scaling

This is fairly linear over a reduced range of values.

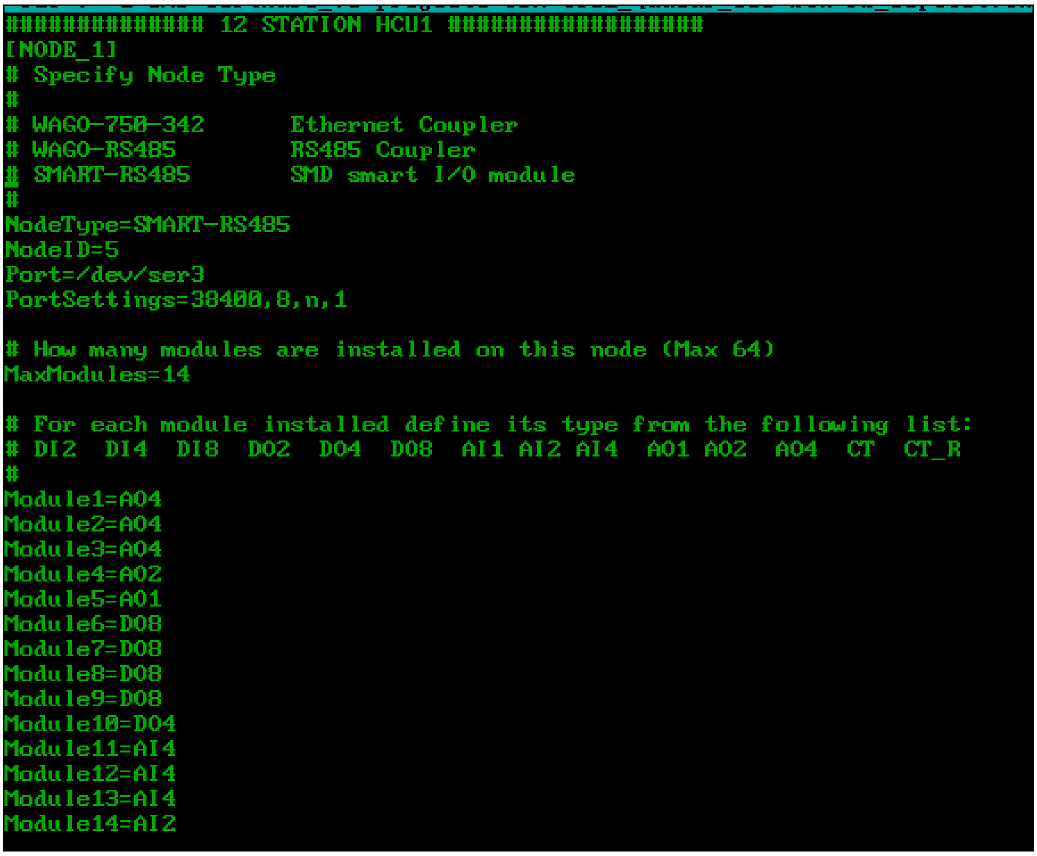
Register Value	Output voltage
85	0
90	0.1
100	3.4
120	6.1
140	9.1
160	12.5
180	16.3
200	20.5
220	20.5
230	21.5

Note 230-255 produces no change in output value.

8.4 Baud rate detection

These PCBs auto detect baud rate on power up from 9K6 – 19k2 – 38k4.

8.5 Standard IO allocation in QNX



```
############## 12 STATION HCU1 ##################
[NODE_1]
# Specify Node Type
#
# WAGO-750-342       Ethernet Coupler
# WAGO-RS485         RS485 Coupler
# SMART-RS485        SMD smart I/O module
#
NodeType=SMART-RS485
NodeID=5
Port=/dev/ser3
PortSettings=38400,8,n,1

# How many modules are installed on this node (Max 64)
MaxModules=14

# For each module installed define its type from the following list:
# DI2  DI4  DI8  DO2  DO4  DO8  AI1 AI2 AI4  AO1 AO2  AO4  CT  CT_R
#
Module1=AO4
Module2=AO4
Module3=AO4
Module4=AO2
Module5=AO1
Module6=DO8
Module7=DO8
Module8=DO8
Module9=DO8
Module10=DO4
Module11=AI4
Module12=AI4
Module13=AI4
Module14=AI2
```

Figure 3 - QNX setup for MVP board

9 MVP LED OPERATION

There are three LEDs on the PCB which indicate the following:

- **Red** LED indicates 24v power supplied to PCB.
- **Yellow** LED flashes to indicate the processor is running on the PCB.
- **Green** LED comes on to indicate comms is being received from the surface controller. N.B there is no guarantee the surface controller is receiving any RS485 data from this LED.

Red LED	On	Off	On	On
Yellow LED	Flashing	Off	Flashing	Off
Green LED	On	Off	Off	Off / On
	Board Ok; we do not know if the surface is receiving data from device though. Sometimes the board can receive comms with only one 485 wire connected.	No 24v power to board. Watchdog trip or no subsea supply (If HCU & TCU LEDs are OFF but Lamp dimmer PCB Power LED is on suspect it's the watchdog).	No Comms to board; suspect telemetry.	PCB firmware issue or processor frazzled; replace PCB.

Figure 4 - Basic PCB fault finding

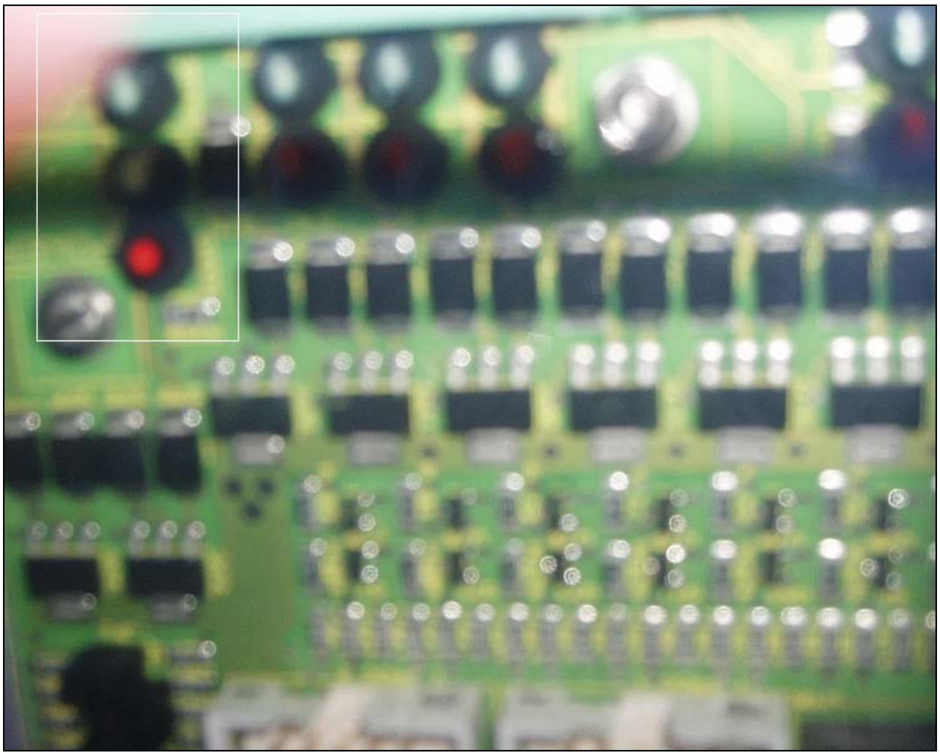


Figure 5 - HCU status LEDs (yellow flashes)

There are also a series of jumpers which control the PCB configuration. **NOTE** - These PCBs are supplied un-configured as spares so it is important that the existing configuration is noted when swapping out units.

There are two 24v Power supplies to this PCB; one powers the PCB control electronics and the other powers the solenoid drive circuits. It is normal for the second circuit to be dropped when the pod watchdog trips.

An additional series of LEDs gives the drive status of the proportional drives; RED LEDs show the 'A' direction and Green LEDs show the 'B' direction. N.B. The proportional flow must be set greater than 0% in the software for these LEDs to illuminate.

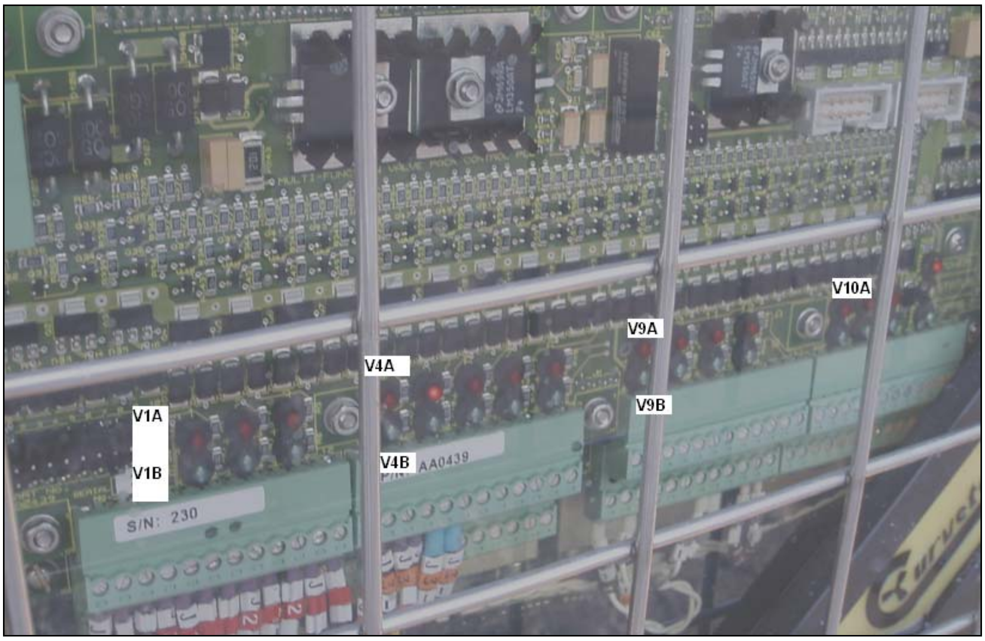


Figure 6 - TCU LED status - Showing 5A (thrusters null) on

10 MFVP JUMER CONFIGURATION

10.1 General

The MFVP consists of two PCB's with interconnections between the lower and upper board. All configuration jumpers are located on the upper board.

10.2 MFVP Board Layout

The location of the jumpers and connectors are shown in Figure 1. J1 to J5 are located on the upper board, J6-J9 are on the lower board.

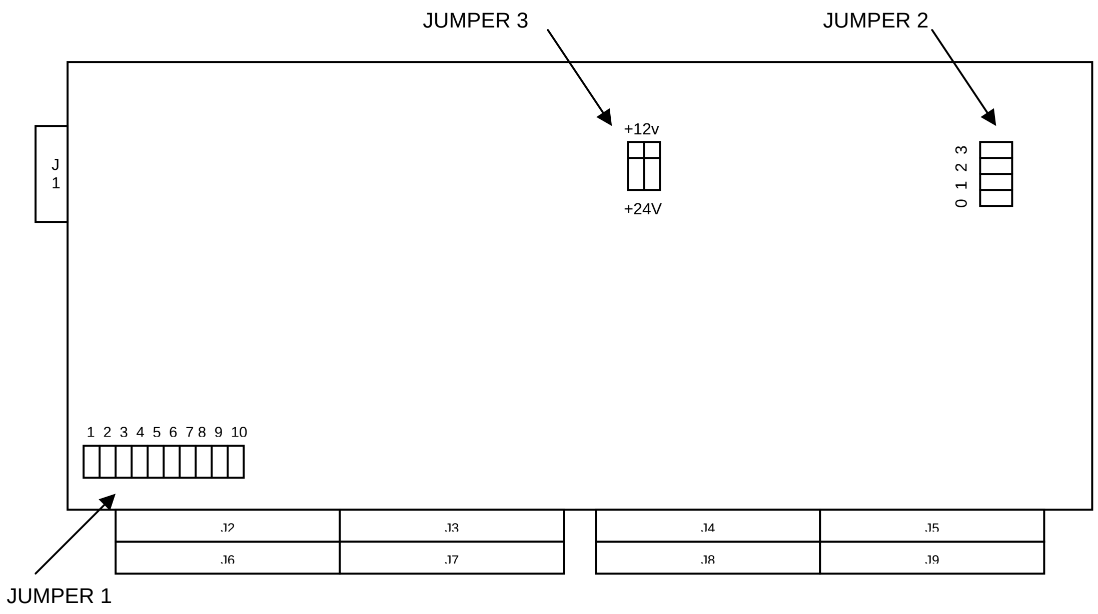


Figure 7 - MFVP Board Layout

10.3 Jumper 1 Configuration

Jumper 1 is used to configure the analogue inputs into the MFVP, the analogue inputs can be 4-20mA or 0-10V. There are 10 jumpers for the 10 analogue inputs which are numbered 1 – 10 from left to right, the jumpers are inserted vertically.

Jumper fitted – 4-20mA selected
Jumper not fitted – 0-10V selected

10.4 Jumper 2 Configuration

Jumper 2 is used to select the address of the board. The board has up to 16 address which can be selected using jumper 2. The addresses are configured as follows:

Address Required	0	1	2	3
0				
1	X			
2		X		
3	X	X		
4			X	
5	X		X	
6		X	X	
7	X	X	X	
8				X
9	X			X
10		X		X
11	X	X		X
12			X	X
13	X		X	X
14		X	X	X
15	X	X	X	X

Note: X indicates jumper is fitted, jumpers fitted horizontally.

10.5 Jumper 3 Configuration

Jumper 3 consists of two jumpers and is used to configure the transducer excitation voltage for analogue input 9 and 10 the second jumper is used to configure the moisture sensor.

The jumper is a three pin header, with the jumper in middle and bottom it sets the excitation as 24V with the jumper in middle and top it is 12V excitation.

The left hand jumper is for sensor 9 and 10 excitation, this jumper changes the excitation voltage for both sensor 9 and 10 connector J8-1 and J8-4 respectively.

The right hand jumper is for moisture transducer supply (water ingress), this is always left at 24V for SMD systems.

11 I/O INTERROGATION MESSAGE SPECIFICATIONS

11.1 General

The control system will send a message to the MVP. This message will contain the node number that the message is intended for, the command identifier, the output data that the control system wishes the MVP to write to its outputs, and a Cyclic Redundancy Code (CRC) used to check for the presence of errors in the received message.

The MVP consists of 2 independent processors, and both of these will receive the message, and after checking that the node address and CRC are correct, they will strip out the relevant output data.

Each processor will then reply in turn, replying with the node number, command identifier, analogue input data, and CRC. There will be a short delay between the end of the reply from the first processor and the start of the reply from the second processor.

The control system will read both replies and confirm the message integrity before accepting the input data and passing this to the rest of the control system.

11.2 I/O Interrogation Transmit Message Format

The transmit message to the MVP is as follows.

Byte Number	Description	Example
1	Node address, in the range 1 to 16	0x01
2	Command Identifier. Always 0x04	0x04
3	Reserved for future use	0x00
4	Number of output data bytes. For this configuration this will always be 21	0x15
5	Solenoid 1 and 2 output. In the range 0 to 100	0x00
6	Solenoid 3 and 4 output. In the range 0 to 100	0x64
7	Solenoid 5 and 6 output. In the range 0 to 100	0x00
8	Solenoid 7 and 8 output. In the range 0 to 100	0x64
9	Solenoid 9 and 10 output. In the range 0 to 100	0x00
10	Solenoid 11 and 12 output. In the range 0 to 100	0x64
11	Solenoid 13 and 14 output. In the range 0 to 100	0x00
12	Solenoid 15 and 16 output. In the range 0 to 100	0x00
13	Solenoid 17 and 18 output. In the range 0 to 100	0x00
14	Solenoid 19 and 20 output. In the range 0 to 100	0x00
15	Solenoid 21 and 22 output. In the range 0 to 100	0x32
16	Solenoid 23 and 24 output. In the range 0 to 100	0x32
17	Solenoid 25 and 26 output. In the range 0 to 100	0x32
18	Solenoid 27 and 28 output. In the range 0 to 100	0x32
19	Solenoid 29 and 30 output. In the range 0 to 100	0x32

20	Digital Output 1 MSB 9 to 16 Solenoid 9 to 16 enable. Bit 0 = Solenoid 9 enable Bit 1 = Solenoid 10 enable Bit 2 = Solenoid 11 enable Bit 3 = Solenoid 12 enable Bit 4 = Solenoid 13 enable Bit 5 = Solenoid 14 enable Bit 6 = Solenoid 15 enable Bit 7 = Solenoid 16 enable A bit value of 1 enables the solenoid output.	0x04
21	Digital Output 1 LSB 1 to 8 Solenoid 1 to 8 enable. Bit 0 = Solenoid 1 enable Bit 1 = Solenoid 2 enable Bit 2 = Solenoid 3 enable Bit 3 = Solenoid 4 enable Bit 4 = Solenoid 5 enable Bit 5 = Solenoid 6 enable Bit 6 = Solenoid 7 enable Bit 7 = Solenoid 8 enable A bit value of 1 enables the solenoid output	0x84
22	Digital Output 2 MSB 25 to 32 Solenoid 25 to 30 enable and sensor 1 to 4 PSU Bit 0 = Solenoid 25 enable Bit 1 = Solenoid 26 enable Bit 2 = Solenoid 27 enable Bit 3 = Solenoid 28 enable Bit 4 = Solenoid 29 enable Bit 5 = Solenoid 30 enable Bit 6 = Sensor 1 and 2 PSU control Bit 7 = Sensor 3 and 4 PSU control A bit value of 1 enables the solenoid output	0xEa
23	Digital Output 2 LSB 17 to 24 Solenoid 17 to 24 enable. Bit 0 = Solenoid 17 enable Bit 1 = Solenoid 18 enable Bit 2 = Solenoid 19 enable Bit 3 = Solenoid 20 enable Bit 4 = Solenoid 21 enable Bit 5 = Solenoid 22 enable Bit 6 = Solenoid 23 enable Bit 7 = Solenoid 24 enable A bit value of 1 enables the solenoid output	0x50

24	Digital Output 3 MSB 41 to 48 Not used	0x00
25	Digital Output 3 LSB 33 to 40 Sensor 4 to 12 control Bit 0 = Sensor 5 and 6 PSU control Bit 1 = Sensor 7 and 8 PSU control Bit 2 = Sensor 9 and 10 PSU control Bit 3 = Moisture sensor PSU control Bit 4 = Not used Bit 5 = Not used Bit 6 = Not used Bit 7 = Not used A bit value of 1 enables the PSU	0x0F
26	CRC MSB See appendix	0xD5
27	CRC LSB	0x60

11.3 I/O Interrogation Processor 1 Reply Format

The reply message from processor 1 will be in the following format

Byte Number	Description	Example
1	Node address, in the range of 1 to 16. Bit 4 will be set to indicate that this is the reply from processor 1	0x11
2	Command Identifier. Always 0x04	0x04
3	Reserved for future use	0x00
4	Number of digital input channels contained in this message. This will be zero in this case	0x00
5	Number of data bytes. For this application this will always be 16	0x10
6	Sensor 1 Analogue input MSB	0x03
7	Sensor 1 Analogue input LSB	0xFF
8	Sensor 2 Analogue input MSB	0x00
9	Sensor 2 Analogue input LSB	0x00
10	Sensor 3 Analogue input MSB	0x00
11	Sensor 3 Analogue input LSB	0x00
12	Sensor 4 Analogue input MSB	0x03
13	Sensor 4 Analogue input LSB	0xFF
14	Sensor 5 Analogue input MSB	0x00
15	Sensor 5 Analogue input LSB	0x00
16	Sensor 6 Analogue input MSB	0x00
17	Sensor 6 Analogue input LSB	0x00
18	Sensor 7 Analogue input MSB	0x03
19	Sensor 7 Analogue input LSB	0xFF
20	Sensor 8 Analogue input MSB	0x00
21	Sensor 8 Analogue input LSB	0x00
22	CRC MSB See appendix	0x5D
23	CRC LSB	0x92

Analogue inputs are 10 bit ADC's, and so will range from 0x0000 to 0x03FF.

If the received message format is incorrect, the processor takes no action, and will not send a reply

11.4 I/O Interrogation Processor 2 Reply Format

The reply message from processor 2 will be in the following format

Byte Number	Description	Example
1	Node address, in the range 1 to 16. Bit 5 will be set to indicate that this is the reply from processor 2	0x21
2	Command Identifier. Always 0x04	0x04
3	Reserved for future use	0x00
4	Number of digital input channels contained in this message. This will be zero in this case	0x00
5	Number of data bytes. For this application this will always be 12	0x0C
6	Sensor 9 Analogue input MSB	0x03
7	Sensor 9 Analogue input LSB	0xFF
8	Sensor 10 Analogue input MSB	0x00
9	Sensor 10 Analogue input LSB	0x00
10	Sensor 11 Analogue input MSB	0x00
11	Sensor 11 Analogue input LSB	0x00
12	Sensor 12 Analogue input MSB	0x03
13	Sensor 12 Analogue input LSB	0xFF
14	PCB Temperature Analogue input MSB	0x00
15	PCB Temperature Analogue input LSB	0x00
16	Supply 24V Analogue input MSB	0x00
17	Supply 24V Analogue input LSB	0x00
18	CRC MSB See appendix	0x50
19	CRC LSB	0x43

Analogue inputs are 10 bit ADC’s, and so will range from 0x0000 to 0x03FF.

If the received message format is incorrect, the processor takes no action, and will not send a reply

11.5 Baud Rate

The default baud rate of the RS485 bus will be 19200,8,n,1

If the board does not receive a valid message within 4 seconds of powering up, the MVP should start to auto search for a suitable baud rate, by cycling around baud rates of 9600, 38400 then back to 19200. It will wait 2 seconds at each baud rate before trying the next.

Once a message is correctly received the baud rate will be maintained for as long as the board remains powered.

11.6 TIMING

For normal operation, it is expected that processor 1 will reply 4ms after the last byte of the command message has been received. This will give time to allow the RS485 transceivers to settle into receive mode throughout the system.

Processor 2 will send its reply a further 4ms after the end of the first processor's reply message.

It may be necessary to modify this time if external equipment cannot meet this turn around delay. This can be achieved using the following command:

11.6.1 Delay Time Configuration Transmit Message Format

Byte Number	Description	Example
1	Node address, in the range 1 to 16.	0x01
2	Command Identifier. Always 0x05	0x05
3	Reserved for future use	0x00
4	Delay Time Processor 1 Value is in ms, in the range 1 to 255	0x01
5	Delay Time Processor 2 Value is in ms, in the range 1 to 255	0x01
6	CRC MSB See appendix	0xD9
7	CRC LSB	0x5C

11.6.2 Delay Time Configuration Processor 1 Reply Format

Byte Number	Description	Example
1	Node address, starting at node 1 to node 16. Bit 4 will be set to indicate that this the reply from processor 1	0x11
2	Command Identifier. Always 0x05	0x05
3	Reserved for future use	0x00
4	Software version 1 to 255	0x01
5	CRC MSB See appendix	0x22
6	CRC LSB	0x95

If the received message format is incorrect, the processor takes no action, and will not send a reply

11.6.3 Delay Time Configuration Processor 2 Reply Format

Byte Number	Description	Example
1	Node address, starting at node 1 to node 16. Bit 5 will be set to indicate that this the reply from processor 2	0x21
2	Command Identifier. Always 0x05	0x05
3	Reserved for future use	0x00
4	Software Version 1 to 255	0x01
5	CRC MSB See appendix	0x22
6	CRC LSB	0x9A

If the received message format is incorrect, the processor takes no action, and will not send a reply

11.7 Termination resistors

120 ohm termination resistors must be fitter to J1-1 and J1-2. For the 485 line to be terminated correctly particularly on 907 focal racks.

If this isn’t done you will get poor telemetry above 2km at higher baud rates.

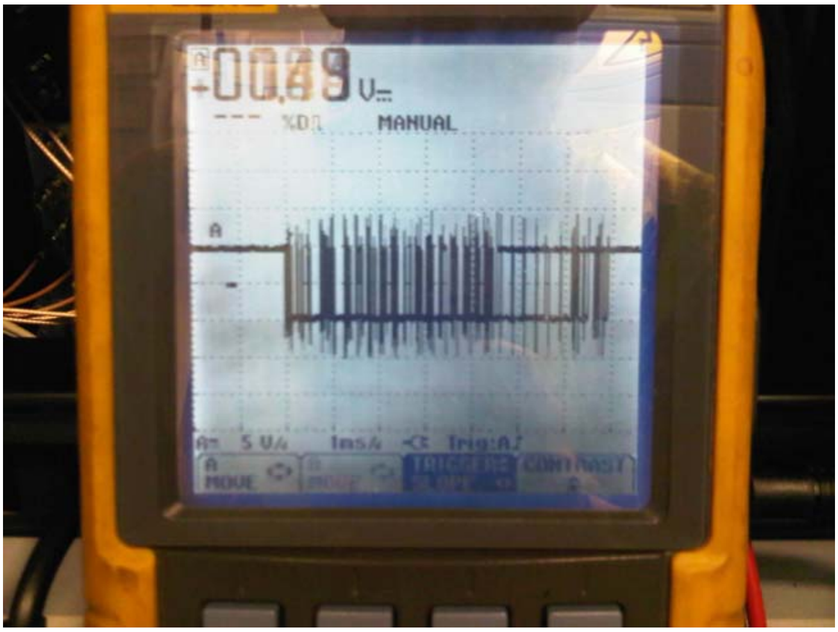


Figure 8 - Incorrectly terminated 485 line.

120R resistors also need adding to the anybus units as well between tx and rx.

11.8 APPENDIX A. CRC coding fragmEnt

The following shows a typical coding fragment used to calculate the 16 bit CRC

```
// Table of CRC values for high-order byte
static const unsigned char CRC_HiTable[] =
{
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40,
0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40,
0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40,
0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40
};

// Table of CRC values for low-order byte
static const unsigned char CRC_LoTable[] =
{
0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06,
0x07, 0xC7, 0x05, 0xC5, 0xC4, 0x04, 0xCC, 0x0C, 0x0D, 0xCD,
0x0F, 0xCF, 0xCE, 0x0E, 0x0A, 0xCA, 0xCB, 0x0B, 0xC9, 0x09,
0x08, 0xC8, 0xD8, 0x18, 0x19, 0xD9, 0x1B, 0xDB, 0xDA, 0x1A,
0x1E, 0xDE, 0xDF, 0x1F, 0xDD, 0x1D, 0x1C, 0xDC, 0x14, 0xD4,
0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2, 0x12, 0x13, 0xD3,
0x11, 0xD1, 0xD0, 0x10, 0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3,
0xF2, 0x32, 0x36, 0xF6, 0xF7, 0x37, 0xF5, 0x35, 0x34, 0xF4,
0x3C, 0xFC, 0xFD, 0x3D, 0xFF, 0x3F, 0x3E, 0xFE, 0xFA, 0x3A,
0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38, 0x28, 0xE8, 0xE9, 0x29,
0xEB, 0x2B, 0x2A, 0xEA, 0xEE, 0x2E, 0x2F, 0xEF, 0x2D, 0xED,
0xEC, 0x2C, 0xE4, 0x24, 0x25, 0xE5, 0x27, 0xE7, 0xE6, 0x26,
0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0, 0xA0, 0x60,
0x61, 0xA1, 0x63, 0xA3, 0xA2, 0x62, 0x66, 0xA6, 0xA7, 0x67,
0xA5, 0x65, 0x64, 0xA4, 0x6C, 0xAC, 0xAD, 0x6D, 0xAF, 0x6F,
0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB, 0x69, 0xA9, 0xA8, 0x68,
0x78, 0xB8, 0xB9, 0x79, 0xBB, 0x7B, 0x7A, 0xBA, 0xBE, 0x7E,
0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C, 0xB4, 0x74, 0x75, 0xB5,
0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71,
0x70, 0xB0, 0x50, 0x90, 0x91, 0x51, 0x93, 0x53, 0x52, 0x92,
```



```
0x96, 0x56, 0x57, 0x97, 0x55, 0x95, 0x94, 0x54, 0x9C, 0x5C,
0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E, 0x5A, 0x9A, 0x9B, 0x5B,
0x99, 0x59, 0x58, 0x98, 0x88, 0x48, 0x49, 0x89, 0x4B, 0x8B,
0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F, 0x8D, 0x4D, 0x4C, 0x8C,
0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42,
0x43, 0x83, 0x41, 0x81, 0x80, 0x40
};

static char         uchCRCHi     = 0xff;
static char         uchCRCLo     = 0xff;


/*---------------------------------------------------------*/
// CRC generation routine
static void crc16(char* msg, unsigned short usDataLen)
{
      unsigned uIndex;

      uchCRCHi = 0xff;
      uchCRCLo = 0xff;

      while (usDataLen--)
      {
            uIndex = uchCRCHi ^ *msg++;
            uchCRCHi = uchCRCLo ^ CRC_HiTable[uIndex];
            uchCRCLo = CRC_LoTable[uIndex];
      }
}


/*---------------------------------------------------------*/
// code fragment for calculating the CRC of an outgoing message

      ucObuf[0] = (char)Node[NodeID].nodeID; // Unit ID
      ucObuf[1] = FCN_READ_REGISTERS;
      ucObuf[2] = 0x00; //
      ucObuf[3] = 0x00; //
      ucObuf[4] = 0x00; //
      ucObuf[5] = regs; // number of registers to read

      crc16(ucObuf, 6);

      ucObuf[6] = uchCRCHi;
      ucObuf[7] = uchCRCLo;

      nSendChars = 8;   // number of bytes to send
```

```
/*-------------------------------------------------------*/
// code fragmant for checking the CRC of an incoming message

      int         crc_fail = FALSE, t_crc_hi, t_crc_lo;
      int         nRecvChars;

      nRecvChars = ReceiveData485(NodeID);

      if (nRecvChars <= 2)
      {
            crc_fail = TRUE;
      }
      else
      {
            t_crc_hi = ucIbuf[nRecvChars - 2];
            t_crc_lo = ucIbuf[nRecvChars - 1];
            crc16(ucIbuf, (nRecvChars - 2));

            if ((t_crc_hi != uchCRCHi) && (t_crc_lo != uchCRCLo))
            {
                  crc_fail = TRUE;
            }
      }
```

12 SENSOR INPUT SCALING

Iin	QNX raw	PLC Raw
0	0	0
1	478	49
2	956	98
3	1435	147
4	1913	196
5	2382	244
6	2861	293
7	3349	343
8	3827	392
9	4296	439
10	4775	488
11	5253	537
12	5732	586
13	6200	634
14	6679	683
15	7157	732
16	7616	779
17	8095	828
18	8573	877
19	9061	927
20	9530	975
20.9	9989	1022

Vin	QNX raw	PLC Raw
0	9	1
1	1025	105
2	2040	209
3	3056	313
4	4081	417
5	5107	522
6	6132	627
7	7157	732
8	8183	837
9	9188	940
9.8	9989	1022
10 (theoretical)	10217	1045