IRAM-COMP-064

Revision: 1
2/15/2011

Contact Author

Institut de RadioAstronomie Millimétrique

HEMT CAN Interface

Owner Francis Morel (morel@iram.fr)

Keywords: CAN, HEMT receiver

Approved by:	Date:	Signature:
A.Perrigouard	Feb10 2010	

Change Record

REVISION	DATE	AUTHOR	SECTION/PAGE AFFECTED	REMARKS
1	10Feb2010	F. Morel	Removed a lot and added some new functions to CANPVNG to drive the HEMT receiver.	LO1 driven with a box " LO Band2" as used for CANPdBNG and CANPVNG.
2	07Jul2010	F. Morel	Modified Amplis section due to I2C hardware mods	
3	03aug2010	F. Morel	Modified Cryostat/ Attenuators/Hot Load/Amplis and suppressed Vacuum due to I2C hardware doc update.	

Content

1 Introduction 4

2 Cryostat Temperature 4

2.1 Summary of control and Monitor Points 4
2.2 Control Points in Detail 4
2.3 Monitor Points in Detail...5

3 Hot Load Temperature 6

3.1 Summary of control and Monitor Points 7
3.2 Control Points in Detail 7
3.3 Monitor Points in Detail...7

4 LO1 8

4.1 Summary of Control and Monitor points..8
4.2 Control Points in Detail:..8
4.3 Monitor Points in Detail:..9
4.4 Receiver Motors – Control and monitor points: 11

7 LO2 12

7.1 Summary of Control and Monitor points:...12
7.2 Control points in detail 12
7.3 Monitor points in detail..12

8 IF Attenuators 13

8.1 Summary of Control and Monitor Points: 13
8.2 Control Points in Detail 13
8.3 Monitor Points in Detail...14

9 Power Supply operations...14

9.1 Summary of Control and Monitor Points 14
9.2 Control Points in Detail 15

9.3 Monitor Points in Detail 15

10 I2C Debug 15

10.1 Summary of Control and Monitor Points 16
10.2 Control Points in Detail 16
10.3 Monitor Points in Detail 16

11 Calibrations 18

11.1 Summary of Control and Monitor points 19
11.2 Control points in Detail 19
11.3 Monitor points in Detail 19

12 Amplifiers 20

12.1 Summary of Control and Monitor points 20
12.2 Control points in detail 21
12.3 Monitor points in detail 21

1 Introduction

The CAN bus in use for monitor and control the Pico Veleta instruments consists of the CAN 2.0B variant and a non-standard higher level protocol defined in the document IRAM-COMP-003 “PdB CAN Specification”.
The I2C bus is connected to a CAN controller used as a bridge, and the CAN control and monitor points derivate directly from the I2C functions.
A more complex set of commands, able of uninterruptible I2C access, was also defined as “convenience” functions. They use different CAN IDs, which are not mapped into the I2C address field.
Here after there is summary of the monitor and control points with their CAN IDs, data sizes and descriptions.

2 Cryostat Temperature

Originally the bus I2C is in use for monitoring and controlling the Cryostat Temperature. Yves Bortolotti has developed this interface. Get from him the applicable documentation.

2.1 Summary of control and Monitor Points

Name	CAN ID	Data Size	Description
SET_CRYO_CONTROL_REGISTER	00 0C 01 82	2	Set control register
SET_CRYO_BOX_TEMP_REGISTER	00 0C 01 90	1	Set MAX6633 register
GET_ CRYO_TEMPERATURE	00 0C 01 81	8	Get converted temperature data
GET_ CRYO_STATUS_REGISTER	00 0C 01 83	3	Get status register
GET_CRYO_BOX_TEMP	00 0C 01 91	3	Get MAX6633 temperature

2.2 Control Points in Detail

Name	SET_ CRYO_CONTROL_REGISTER	
CAN ID	00 0C 01 82	
Description	Set the order register and in particular indicate the number of channel to convert	

Data:	2 bytes
	Bytes[0,1] = bits[15-0]
	Bit[15] = Unused
	bits[14-9] = Command
	bits[8-0] = Parameter	

	Command:	Parameter
	0x00 to 0x07: Standby	Memory read start address. 0 to 511
	0x08: Set 1^{st} channel number to write	Channel number [0-7]
	0x09: Set write memory pointer	Memory write conversion address. 0 to 511
	0x0A: Set last channel number to write	Channel number [0-7]
	0x0B: Set number of samples per channel to write	Number-1 of samples/channel. 0 to 255
	0x10: Set 1^{st} channel number to read	Channel number [0-7]
	0x11: Set read memory pointer	Memory read start address. 0 to 511
	0x12: Set last channel number to read	Channel number [0-7]
	0x13: Set number of samples/channel to read	Number-1 of samples/channel. 0 to 255
	0x18 to 0x1F: Conversion start	Start conversion
	0x38 to 0x3F: Soft reset	Soft Reset

Default value at power on:

1st channel number	0
Last channel number	3
Number of samples per channel -1	0
Memory write conversion address	0
Memory read start address	0

Those values are the standard values for the operations at Plateau de Bure.

Operation:
When a conversion is started, the requested number of samples/channel of the given 1st channel are stored at the addresses starting from the value named “Memory write conversion address”. The conversions are stored in 2 bytes words at consecutive addresses. This conversion continues with the next channel up to the last channel and then stops. Each conversion takes 67.114 milliseconds to complete.
The “Memory read start address” is the memory starting address for reading the converted temperatures through the field bus. Although it is set independently of the “Memory write conversion address” it seems reasonable to set both to the same value for normal operations.

Name	SET_CRYO_BOX_TEMP_REGISTER
CAN ID	00 0C 01 90
Description	Set the MAXIM 6633 configuration register
Data	1 byte = 0x00: enabled. Default value at power on = 0x01: disabled

2.3 Monitor Points in Detail

Name	GET_ CRYO_TEMPERATURE
CAN ID	00 0C 01 81
Description	Get 4 channel values. After the execution the “Memory read start address” is incremented by 8 mod 256.
Data	8 bytes Bytes[0,1]: bits[15]: 1 = Invalid data, 0 = OK bits[14-12]: Channel number, 0 to 7 bits[11-00]: Channel unsigned value from 0x00 to 0xFFF Bytes[2,3]: bits[15]: 1 = Invalid data, 0 = OK bits[14-12]: Channel number, 0 to 7 bits[11-00]: Channel unsigned value from 0x00 to 0xFFF Bytes[4,5]: bits[15]: 1 = Invalid data, 0 = OK bits[14-12]: Channel number, 0 to 7 bits[11-00]: Channel unsigned value from 0x00 to 0xFFF Bytes[6,7]: bits[15]: 1 = Invalid data, 0 = OK bits[14-12]: Channel number, 0 to 7 bits[11-00]: Channel unsigned value from 0x00 to 0xFFF

Name	GET_ CRYO_STATUS_REGISTER
CAN ID	00 0C 01 83
Description	Get the status register which depends on the last message See SET_ CRYO_CONTROL_REGISTER

<table>
<tr><td>Data</td><td>3 bytes

Bytes[0,1] = bits[15-0]
bit[15] = Unit Busy
bits[14-9] = Status or last requested command
bits[8-0] = Parameter

<table>
<tr><td>Status or last requested command:</td><td>Parameter:</td></tr>
<tr><td>0x00 to 0x07: Standby</td><td>Memory read start address. 0 to 511</td></tr>
<tr><td>0x08: Get 1st channel number to write</td><td>Channel number [0-7]</td></tr>
<tr><td>0x09: Get write memory pointer</td><td>Memory write conversion address. 0 to 511</td></tr>
<tr><td>0x0A: Get last channel number to write</td><td>Channel number [0-7]</td></tr>
<tr><td>0x0B: Get number of samples per channel to write</td><td>Number-1 of samples/channel. 0 to 255</td></tr>
<tr><td>0x10: Get 1st channel number to read</td><td>Channel number [0-7]</td></tr>
<tr><td>0x11: Get read memory pointer</td><td>Memory read start address. 0 to 511</td></tr>
<tr><td>0x12: Get last channel number to read</td><td>Channel number [0-7]</td></tr>
<tr><td>0x13: Get number of samples/channel to read</td><td>Number-1 of samples/channel. 0 to 255</td></tr>
<tr><td>0x18 to 0x1F: Conversion start</td><td>Start conversion</td></tr>
<tr><td>0x38 to 0x3F: Soft reset</td><td>Soft Reset</td></tr>
</table>

Byte[2]: Error report
bit[2]=CAN error
bit[1]=I2C write error
bit[0]=I2C read error</td></tr>
</table>

Name	GET_CRYO_BOX_TEMP
CAN ID	00 0C 01 91
Description	Get the MAX 6633 read temperature

Data	3 bytes
	Bytes[0,1] = bits[15-0] (2's compliment)
	bits[15-3] = temperature
	bit[3], LSB = .0625deg Celsius .
	Byte[2]: Error report
	bit[2]=CAN error
	bit[1]=I2C write error
	bit[0]=I2C read error

Connection between temperatures and receiver band

Channel number	Name	Sensor Type	Sensor	Temp. Range (K)
0	Cryogenerator 77K	Platinum resistor	PT100	50 to 300
1	Plate 15K	Carbon resistor	H59	1.5 to 100
2	Load 15K	Carbon resistor	F54	1.5 to 100
3	Ampli	Carbon resistor	F53	1.5 to 100

3 Hot Load Temperature

Originally the bus I2C is in use for monitoring and controlling the Hot Load Temperature. Yves Bortolotti has developed this interface. Get from him the applicable documentation.

3.1 Summary of control and Monitor Points

Name	CAN ID	Data Size	Description
SET_HOT_LOAD1_DS620_REGISTER	00 0C 01 92	1	Set DS620-1 register
GET_HOT_LOAD1_DS620_TEMPERATURE	00 0C 01 93	3	Get hot load 1 temperature

Convenience monitor point:

GET_HOT_LOAD1_TEMPERATURE	00 0C 02 A0	3	Get hot load 1 temperature

3.2 Control Points in Detail

Name	SET_HOT_LOAD1_DS620_REGISTER
CAN ID	00 0C 01 92
Description	Set the DS620-1 configuration register
Data	1 byte = 0xAA to be able to read the hot load temperature. This value is incremented by each DS620 reading. As a consequence, this register has to be set to 0xAA each time the hot load temperature is monitored.

3.3 Monitor Points in Detail

Name	GET_HOT_LOAD1_DS620_TEMPERATURE
CAN ID	00 0C 01 93
Description	Get the hot load temperature (as far the configuration register is set to 0xAA).
Data	3 bytes Bytes[0,1] = bits[15-0] (2’s compliment) bit[0], lsb = 1/128 deg Celsius. Byte[2]: Error report bit[2]=CAN error bit[1]=I2C write error bit[0]=I2C read error

Convenience monitor point:

Name	GET_HOT_LOAD1_TEMPERATURE
CAN ID	00 0C 02 A0
Description	Get the hot load temperature. It is a convenient function which sets automatically the configuration register for reading the hot load temperature.
Data	3 bytes Bytes[0,1] = bits[15-0] (2’s compliment) bit[0], lsb = 1/128 deg Celcius. Byte[2]: Error report bit[2]=CAN error bit[1]=I2C write error bit[0]=I2C read error

4 LO1

4.1 Summary of Control and Monitor points

A LOB2 (Band 2) box is used for local oscillator 1

Name	CAN ID	Data size	Description
SET_LO1_COMMAND	02 00 01 10	2	LO1 command register
GET_LO1_COMMAND	02 00 01 20	3	LO1 command register
SET_LO1_HARM_MIXER_BIAS	02 04 01 10	2	LO1 harmonic mixer bias
SET_LO1_LOOP_GAIN	02 04 01 11	2	LO1 loop gain
SET_LO1_GUNN_BIAS	02 04 01 12	2	LO1 gunn bias
GET_LO1_HARM_MIXER_BIAS	02 04 01 20	2	LO1 harmonic mixer bias
GET_LO1_LOOP_GAIN	02 04 01 21	2	LO1 loop gain
GET_LO1_GUNN_BIAS	02 04 01 22	2	LO1 gunn bias
GET_LO1_STATUS	02 00 01 00	3	LO1 status register
GET_LO1_OFFSET_VOLTAGE	02 04 01 00	3	LO1 offset voltage
GET_LO1_PLL_IF_LEVEL	02 04 01 01	3	LO1 PLL OF level
GET_LO1_HARM_MIXER_CURRENT	02 04 01 02	3	LO1 harmonic mixer current

Receiver motors:

Name	CAN ID	Data Size	Description
SET_LO1_FREQ	02 10 01 01	2	See description below
SET_LO1_POWER_GUNN	02 14 01 01	2	See description below
SET_LO1_HARM_MIXER_POWER	02 18 01 01	2	See description below
SET_LO1_POWER1	02 1C 01 01	2	See description below
SET_LO1_POWER2	02 20 01 01	2	See description below
GET_LO1_FREQ	02 10 01 00	3	See description below
GET_LO1_POWER_GUNN	02 14 01 00	3	See description below
GET_LO1_HARM_MIXER_POWER	02 18 01 00	3	See description below
GET_LO1_POWER1	02 1C 01 00	3	See description below
GET_LO1_POWER2	02 20 01 00	3	See description below

4.2 Control Points in Detail:

Name	SET__LO1_COMMAND
CAN ID	02 00 01 10
Description	Set LO1 command register
Data	2 bytes Byte[0]: unused Byte[1] bit[7-4]: unused. bit[3]: sweep: 1:On, 0:off. bit[2]: loop: 1:Closed, 0:Open. bit[1]: deltaF: 1:+, 0:-. bit[0]: gunn: 1:On, 0:Off.

Name	SET_LO1_HARM_MIXER_BIAS

CAN ID	02 04 01 10
Description	Set LO1 harmonic mixer bias
Data	2 bytes 14 bits DAC Byte[0,1] = 0: 0V Byte[0,1] = 0x3FFF : 9.9998V

Name	SET_LO1_LOOP_GAIN
CAN ID	02 04 01 11
Description	Set LO1 loop gain
Data	2 bytes 14 bits DAC Byte[0,1] = 0: 0V Byte[0,1] = 0x3FFF : 9.9998V

Name	SET_LO1_GUNN_BIAS
CAN ID	02 04 01 12
Description	Set LO1 gunn bias
Data	2 bytes 14 bits DAC Byte[0,1] = 0: 0V Byte[0,1] = 0x3FFF : 9.9998V

4.3 Monitor Points in Detail:

Name	GET_LO1_STATUS
CAN ID	02 00 01 00
Description	Get LO1 status register
Data	3 bytes Byte[0]: unused Byte[1]: bit[7-4]: unused. bit[3]: sweep: 1:On, 0:off. bit[2]: loop: 1:Closed, 0:Open. bit[1]: deltaF: 1:+, 0:-. bit[0]: gunn: 1:On, 0:Off. Byte[2]: Error report: bit[2]: CAN error

Name	GET_LO1_COMMAND
CAN ID	02 00 01 20
Description	Get LO1 command register (reread last written command)
Data	3 bytes Byte[0]: unused Byte[1]: bit[7-4]: unused. bit[3]: sweep: 1:On, 0:off. bit[2]: loop: 1:Closed, 0:Open. bit[1]: deltaF: 1:+, 0:-. bit[0]: gunn: 1:On, 0:Off. Byte[2]: Error report bit[2]: CAN error

Name	GET_LO1_OFFSET_VOLTAGE
CAN ID	02 04 01 00

Description	Get LO1 offset voltage
Data	3 bytes 16 bits ADC Byte[0,1] = 0: 0V Byte[0,1] = 0xFFFF : 9.9998V Byte[2]: Error report bit[2]: CAN error

Name	GET_LO1_PLL_IF_LEVEL
CAN ID	02 04 01 01
Description	Get LO1 PLL IF level
Data	3 bytes 16 bits ADC Byte[0,1] = 0: 0V Byte[0,1] = 0xFFFF : 9.9998V Byte[2]: Error report bit[2]: CAN error

Name	GET_LO1_HARM_MIXER_CURRENT
CAN ID	02 04 01 02
Description	Get LO1 harmonic mixer current
Data	3 bytes 16 bits ADC Byte[0,1] = 0: 0V Byte[0,1] = 0xFFFF : 19.9997mA Byte[2]: Error report bit[2]: CAN error

Name	GET_LO1_HARM_MIXER_BIAS
CAN ID	02 04 01 20
Description	Get LO1 harmonic mixer bias request
Data	3 bytes Byte[0,1] = 0: 0V Byte[0,1] = 0x3FFF : 9.9998V Byte[2]: Error report bit[2]: CAN error

Name	GET_LO1_LOOP_GAIN
CAN ID	02 04 01 21
Description	Get LO1 loop gain request
Data	3 bytes Byte[0,1] = 0: 0V Byte[0,1] = 0x3FFF : 9.9998V Byte[2]: Error report bit[2]: CAN error

Name	GET_LO1_GUNN_BIAS
CAN ID	02 04 01 22
Description	Get LO1 gunn bias request
Data	3 bytes Byte[0,1] = 0: 0V Byte[0,1] = 0x3FFF : 9.9998V Byte[2]: Error report bit[2]: CAN error

4.4 Receiver Motors – Control and monitor points:

For details see document IRAM-COMP-008 “Receiver Motor control using CAN-Bus” written by Francis Morel.

Control points:

SET_LO1_FREQ (xx=10)
SET_LO1_POWER_GUNN(xx=14)
SET_LO1_HARM_MIXER_POWER(xx=18)
SET_LO1_POWER1(xx=1C)
SET_LO1_POWER2(xx=20)
	SET RPOS

xx =10,14,18,1C,20	02 xx 01 01
Description	Sets the Motor Requested Position

Data	2 bytes
	Data Bytes[0-1] = 12-bit unsigned value of Actual Position.
	Byte[0] = Position MSByte
	Byte[1] = Position LSByte.

STOP_MOTOR_xx	STOP
xx=10,14,18,1C,20	02 xx 01 03
Description	Stops the Motor.
Data	1 dummy byte

RESET_MOTOR_xx	RESET
xx=10,14,18,1C,20	02 xx 01 FF
Description	Resets the Motor. The motor will not move. This command has highest priority, and executes inside the CAN interrupt routine.
Data	1 dummy byte

Monitor points:

GET_LO1_FREQ (xx=10)
GET_LO1_POWER_GUNN(xx=14)
GET_LO1_HARM_MIXER_POWER(xx=18)
GET_LO1_POWER1(xx=1C)
GET_LO1_POWER2(xx=20)
	GET APOS

xx=10,14,18,1C,20	02 xx 01 00
Description	Reads the Motor Actual Position

Data	3 bytes
	Data Bytes[0-1] = 12-bit unsigned value of Actual Position.
	Byte[0] = Position MSByte
	Byte[1] = Position LSByte.
	Byte[2]:Error report
	bit[0] = CAN Warning.

GET_MOTORxx_STATUS	STS
xx=10,14,18,1C,20	02 xx 01 02
Description	Reads the Motor Status
Data	4 bytes Data Byte[0] = 1-bit Status Code *0x20: Board reset*

	0x10: Board stopped *0x8: Requested Position error (Bytes[1..2] = Requested Position)* *0x4: Position Aborted (Bytes[1..2] = Actual Position)* *0x2: Position Reached (Bytes[1..2] = Requested Position)* *0x1: Running (Bytes[1..2] = Actual Position)* Data Byte[1-2] = (Requested OR Actual) Position. Byte[1] = Position MSByte Byte[2] = Position LSByte. Error report in Byte[3]:: bit[0] = CAN Warning.

7 LO2

A 29 GHz DRO (Dielectric Resonator Oscillator) is used as 2nd Local oscillator. It can be switched On/Off , and status-monitored through I2C. Yves Bortolotti has developed this interface. Get from him the applicable documentation.

7.1 Summary of Control and Monitor points:

Name	CAN ID	Data Size	Description
SET_LO2_COMMAND	00 0C 01 A0	1	See description below
GET_LO2_STATUS	00 0C 01 A1	2	See description below

7.2 Control points in detail

Name	SET_LO2_COMMAND
CAN ID	00 0C 01 A0
Description	Sets the LO2 Command Register
Data	1 Byte Byte[0]: 0xF0 = Set LO2 ON 0xF1 = Set LO2 OFF

7.3 Monitor points in detail

Name	GET_LO2_STATUS
CAN ID	00 0C 01 A1
Description	Gets the LO2 Status Register
Data	2 Bytes Byte[0]:

	bits[7-6]: Always “1” bit[5]: 1 = LO2 locked 0 = LO2 unlocked bit[4]: 1 = LO2 ON 0 = LO2 OFF bits[3-1]: Always “1” bit[0]: Recopy of Command register bit[0] Byte[1]: Error report bit[2] = CAN error bit[1] = I2C write error bit[0] = I2C read error.

8 IF Attenuators

Yves Bortolotti has developed this interface. Get from him the applicable documentation.

8.1 Summary of Control and Monitor Points:

Name	CAN ID	Data size	Description
SET_V_ATTENUATOR_COMMAND	00 0C 01 A2	1	Set attenuator command register
SET_H_ATTENUATOR_COMMAND	00 0C 01 A4	1	Set attenuator command register
GET_V_ATTENUATOR_COMMAND	00 0C 01 A3	2	Get attenuator command register
GET_H_ATTENUATOR_COMMAND	00 0C 01 A5	2	Get attenuator command register

8.2 Control Points in Detail

Name	SET_POL_V_ATTENUATOR_COMMAND
CAN ID	00 0C 01 A2
Description	Set Vertical Polarity Attenuator command register. To change value: -First set attenuation to max (bits [7-0] = 0xC0). -Then set desired attenuation.
Data	1 Byte: Byte[0]: bit[7]: always to be set bit[6]: always to be set bit[5]: Pol V 16 dB attenuator (1=OFF, 0=ON) bit[4]: Pol V 8 dB attenuator (1=OFF, 0=ON) bit[3] : Pol V 4 dB attenuator (1=OFF, 0=ON) bit[2]: Pol V 2 dB attenuator (1=OFF, 0=ON) bit[1]: Pol V 1 dB attenuator (1=OFF, 0=ON) bit[0]: Pol V 0.5 dB attenuator (1=OFF, 0=ON)

Name	SET_POL_H_ATTENUATOR_COMMAND
CAN ID	00 0C 01 A4
Description	Set Vertical Polarity Attenuator command register
Data	1 byte:

	See SET_POL_V_ATTENUATOR_COMMAND

8.3 Monitor Points in Detail

Name	GET_POL_V_ATTENUATOR_REGISTER
CAN ID	00 0C 01 A3
Description	Get Vertical Polarity Attenuator register
Data	2 bytes: Byte[0]: bit[7]: always read as “1” bit[6]: always read as “1” bit[5]: Pol V 16 dB attenuator (1=OFF, 0=ON) bit[4]: Pol V 8 dB attenuator (1=OFF, 0=ON) bit[3] : Pol V 4 dB attenuator (1=OFF, 0=ON) bit[2]: Pol V 2 dB attenuator (1=OFF, 0=ON) bit[1]: Pol V 1 dB attenuator (1=OFF, 0=ON) bit[0]: Pol V 0.5 dB attenuator (1=OFF, 0=ON) Byte[1]: Error report bit[2]=CAN error bit[1]=I2C write error bit[0]=I2C read error.

Name	GET_POL_H_ATTENUATOR_REGISTER
CAN ID	00 0C 01 A5
Description	Get Horizontal Polarity Attenuator register
Data	1 byte: See GET_POL_H_ATTENUATOR_COMMAND

9 Power Supply operations

Originally the bus I2C is in use for monitoring and controlling some power supplies. Yves Bortolotti has developed this interface. Get from him the applicable documentation.
The I2C bus is connected to a CAN controller used as a bridge. The CAN control and monitor points derivate directly from the I2C functions.

9.1 Summary of Control and Monitor Points

Name	CAN ID	Data size	Description
SET_POWER_SUPPLY1_COMMAND	00 0C 01 48	1	Switch power supplies on/off
GET_POWER_SUPPLY1_STATUS	00 0C 01 49	2	Read power supplies commands and status
SET_POWER_SUPPLY2_COMMAND	00 0C 01 4A	1	Switch power supplies on/off
GET_POWER_SUPPLY2_STATUS	00 0C 01 4B	2	Read power supplies commands and status

9.2 Control Points in Detail

Name	SET_POWER_SUPPLY1_COMMAND
CAN ID	00 0C 01 48
Description	Switch on/off the power supplies which are under I2C control
Data	1 byte : bits[7-4] = Must be equal to 0xF bit[3] :1 = On, 0= Off. Command coil current and cryostat temperature module power supply bit[2] :1 = On, 0= Off. Command bias HEMT module power supply bit[1] :1 = On, 0= Off. Command bias junctions (5-8) module power supply bit[0] :1 = On, 0= Off. Command bias junctions (1-4) module power supply

Name	SET_POWER_SUPPLY2_COMMAND
CAN ID	00 0C 01 4A
Description	Switch on/off the power supplies which are under I2C control
Data	See SET_POWER_SUPPLY1_COMMAND

9.3 Monitor Points in Detail

Name	GET_POWER_SUPPLY1_STATUS
CAN ID	00 0C 01 49
Description	Get commands and status of the power supplies which are under I2C control
Data	2 bytes : Byte[0] bit[7] :0 = On, 1= Off. Coil current and cryostat temperature module power supply status bit[6] :0 = On, 1= Off. Bias HEMT module power supply status bit[5] :0 = On, 1= Off. Bias junctions (5-8) module power supply status bit[4] :0 = On, 1= Off. Bias junctions (1-4) module power supply status bit[3] :1 = On, 0= Off. Coil current and cryostat temperature power supply command bit[2] :1 = On, 0= Off. Bias HEMT module power supply command bit[1] :1 = On, 0= Off. Bias junctions (5-8) module power supply command bit[0] :1 = On, 0= Off. Bias junctions (1-4) module power supply command Byte[1]: Error report bit[2]=CAN error bit[1]=I2C write error bit[0]=I2C read error

Name	GET_POWER_SUPPLY2_STATUS
CAN ID	00 0C 01 4B
Description	Get commands and status of the power supplies which are under I2C control
Data	See GET_POWER_SUPPLY1_STATUS

At power on, the power supplies are requested to be on and, after, as a consequence, before any SET_POWER_SUPPLY_COMMAND CAN message, the message GET_POWER_SUPPLY_STATUS returns a byte with bit[7-4] equal to the status of the 4 power supplies and bit[3-0]=0xF.

10 I2C Debug

For debugging purposes**:**

- The I2C Controller status can be read.
- 2 special I2C commands are implemented. They will allow reading or writing up to 6 bytes at arbitrary I2C address.

10.1 Summary of Control and Monitor Points

Name	CAN ID	Data size	Description
GET_I2C_CONTROLLER_STATUS	00 0C 01 FC	2	Get I2C Controller status
DEBUG_I2C_WRITE	00 0C 02 E0	8	Write x bytes at I2C address y
DEBUG_I2C_READ	00 0C 02 E1	2 or 8	Read x bytes from I2C address y

10.2 Control Points in Detail

Name	DEBUG_I2C_WRITE
CAN ID	00 0C 02 E0
Description	Writes a list of "x" datas at specified I2C address "y"
Data	8 Bytes: Byte[0] = I2C address to be accessed (y) Byte[1] = Number of data bytes to be written at I2C address (x) Byte[2-7] = Data bytes to be written, unused (excess) bytes will be ignored.

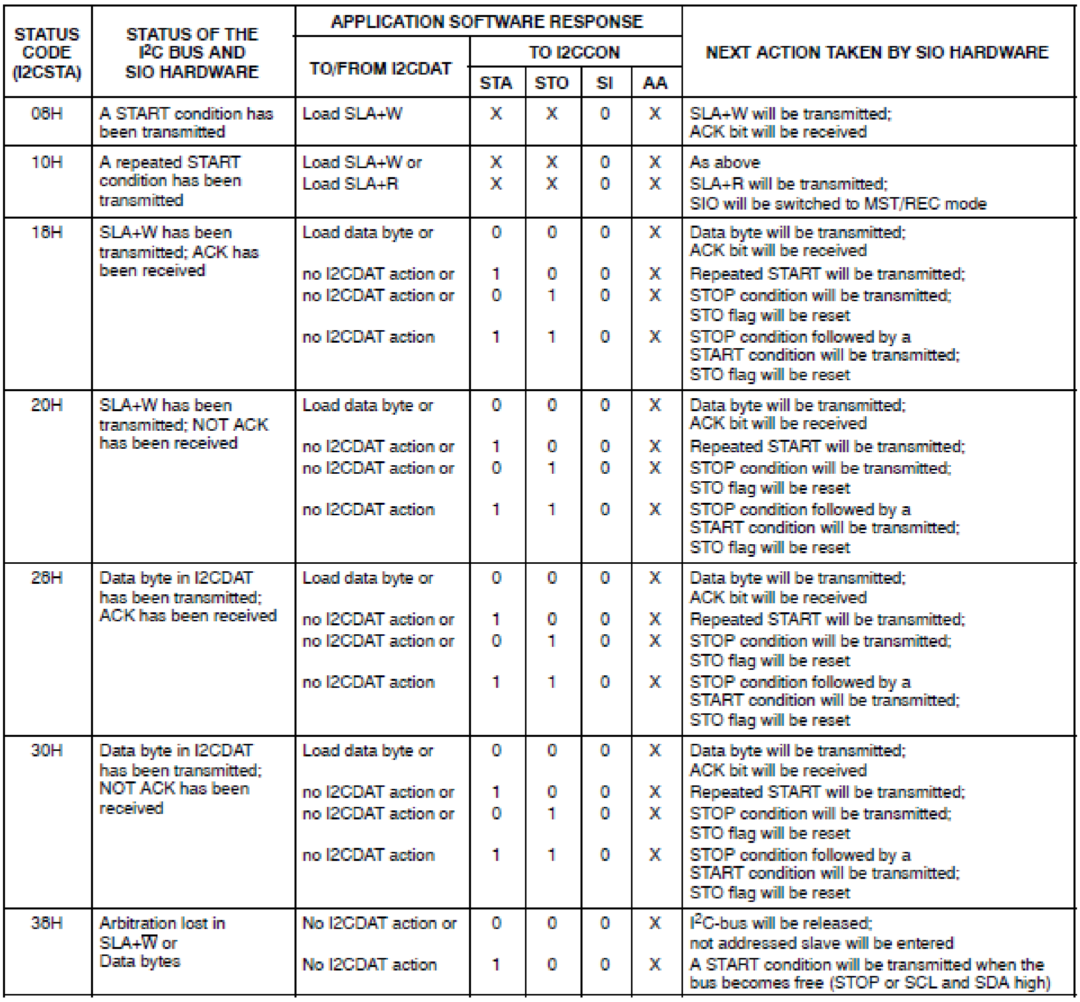
10.3 Monitor Points in Detail

Name	GET_I2C_CONTROLLER_STATUS
CAN ID	00 0C 01 FC
Description	Get the Status byte of the I2C Controller, as sampled after last transaction.

Data	2 Bytes:
	Byte[0]: PCA9654 Status (see below Status Codes)
	Byte[1]: Error report
	bit[2]=CAN error
	bit[1]=I2C write error
	bit[0]=I2C read error

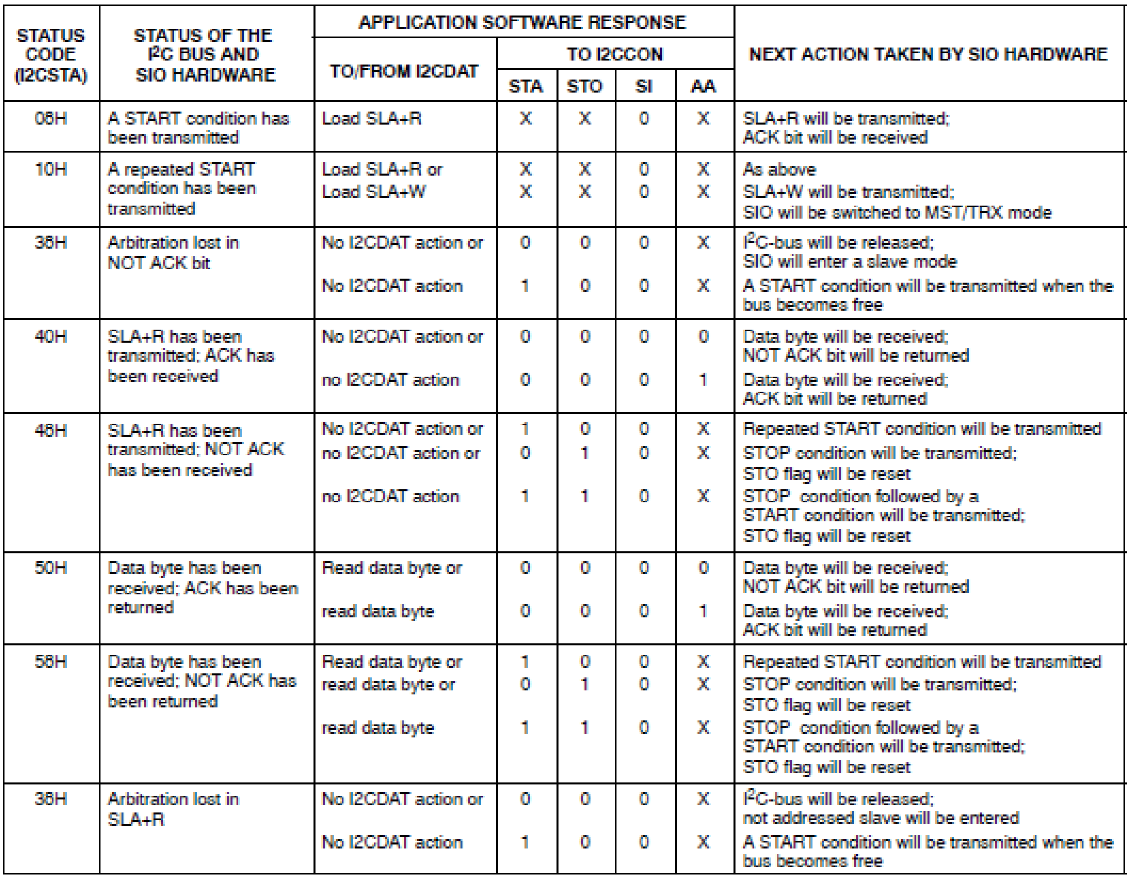
PCA 9654 Status Codes:

Table 2. Master Transmitter Mode



STATUS CODE (I2CSTA)	STATUS OF THE I^{2}C BUS AND SIO HARDWARE	APPLICATION SOFTWARE RESPONSE					NEXT ACTION TAKEN BY SIO HARDWARE
		TO/FROM I2CDAT	TO I2CCON				
			STA	STO	SI	AA	
08H	A START condition has been transmitted	Load SLA+W	X	X	0	X	SLA+W will be transmitted; ACK bit will be received
10H	A repeated START condition has been transmitted	Load SLA+W or	X	X	0	X	As above
		Load SLA+R	X	X	0	X	SLA+R will be transmitted; SIO will be switched to MST/REC mode
18H	SLA+W has been transmitted; ACK has been received	Load data byte or	0	0	0	X	Data byte will be transmitted; ACK bit will be received
		no I2CDAT action or	1	0	0	X	Repeated START will be transmitted;
		no I2CDAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset
		no I2CDAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset
20H	SLA+W has been transmitted; NOT ACK has been received	Load data byte or	0	0	0	X	Data byte will be transmitted; ACK bit will be received
		no I2CDAT action or	1	0	0	X	Repeated START will be transmitted;
		no I2CDAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset
		no I2CDAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset
28H	Data byte in I2CDAT has been transmitted; ACK has been received	Load data byte or	0	0	0	X	Data byte will be transmitted; ACK bit will be received
		no I2CDAT action or	1	0	0	X	Repeated START will be transmitted;
		no I2CDAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset
		no I2CDAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset
30H	Data byte in I2CDAT has been transmitted; NOT ACK has been received	Load data byte or	0	0	0	X	Data byte will be transmitted; ACK bit will be received
		no I2CDAT action or	1	0	0	X	Repeated START will be transmitted;
		no I2CDAT action or	0	1	0	X	STOP condition will be transmitted; STO flag will be reset
		no I2CDAT action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset
38H	Arbitration lost in SLA+$\overline{W}$ or Data bytes	No I2CDAT action or	0	0	0	X	I^{2}C-bus will be released; not addressed slave will be entered
		No I2CDAT action	1	0	0	X	A START condition will be transmitted when the bus becomes free (STOP or SCL and SDA high)

Table 3. Master Receiver Mode



STATUS CODE (I2CSTA)	STATUS OF THE I^2C BUS AND SIO HARDWARE	APPLICATION SOFTWARE RESPONSE					NEXT ACTION TAKEN BY SIO HARDWARE
		TO/FROM I2CDAT	TO I2CCON				
			STA	STO	SI	AA	
08H	A START condition has been transmitted	Load SLA+R	X	X	0	X	SLA+R will be transmitted; ACK bit will be received
10H	A repeated START condition has been transmitted	Load SLA+R or Load SLA+W	X X	X X	0 0	X X	As above SLA+W will be transmitted; SIO will be switched to MST/TRX mode
38H	Arbitration lost in NOT ACK bit	No I2CDAT action or No I2CDAT action	0 1	0 0	0 0	X X	I^2C-bus will be released; SIO will enter a slave mode A START condition will be transmitted when the bus becomes free
40H	SLA+R has been transmitted; ACK has been received	No I2CDAT action or no I2CDAT action	0 0	0 0	0 0	0 1	Data byte will be received; NOT ACK bit will be returned Data byte will be received; ACK bit will be returned
48H	SLA+R has been transmitted; NOT ACK has been received	No I2CDAT action or no I2CDAT action or no I2CDAT action	1 0 1	0 1 1	0 0 0	X X X	Repeated START condition will be transmitted STOP condition will be transmitted; STO flag will be reset STOP condition followed by a START condition will be transmitted; STO flag will be reset
50H	Data byte has been received; ACK has been returned	Read data byte or read data byte	0 0	0 0	0 0	0 1	Data byte will be received; NOT ACK bit will be returned Data byte will be received; ACK bit will be returned
58H	Data byte has been received; NOT ACK has been returned	Read data byte or read data byte or read data byte	1 0 1	0 1 1	0 0 0	X X X	Repeated START condition will be transmitted STOP condition will be transmitted; STO flag will be reset STOP condition followed by a START condition will be transmitted; STO flag will be reset
38H	Arbitration lost in SLA+R	No I2CDAT action or No I2CDAT action	0 1	0 0	0 0	X X	I^2C-bus will be released; not addressed slave will be entered A START condition will be transmitted when the bus becomes free

Name	DEBUG_I2C_READ
CAN ID	00 0C 02 E1
Description	Master requests a list of “x” datas at specified I2C address “y”
Data	2 Bytes: Byte[0] = I2C address to be accessed (y) Byte[1] = Number of data Bytes to be read (x)

Name	DEBUG_I2C_READ *(REPLY)*
CAN ID	00 0C 02 E1
Description	Slave replies a list of “x” data Bytes at specified I2C address “y”
Data	8 Bytes: Byte[0] = I2C accessed address (y) Byte[1] = Number of read data Bytes (x) Byte[2-7] = Read data bytes, unused bytes will be zeroed.

CAUTION: The Message “DEBUG_I2C_READ” does not respect the protocol defined for IRAM-CAN (see PdB CAN Specification, written by Alain Perrigouard) and should thus be used for debugging ONLY.

11 Calibrations

A 16-bit CANIO board is used to control/monitor hot lad, mirror MH5 and translation table. All three devices are pneumatic and have 2 stable positions only, switched through electrovanes. 3 outputs of the CANIO drive the 3 electrovanes, and 6 inputs monitor the 6 proximity sensors used as limit switches.

11.1 Summary of Control and Monitor points

Name	CAN ID	Data size	Description
SET_HEMT_CAL_COMMAND	01 0C 01 10	2	Set Calibration Command Register
GET_HEMT_CAL_COMMAND	01 0C 01 20	3	Get Calibration Command Register
GET_HEMT_CAL_STATUS	01 0C 01 00	3	Get Calibration Status Register

11.2 Control points in Detail

Name	SET_HEMT_CAL_COMMAND
CAN ID	01 0C 01 10
Description	Sets the Calibration Command Register 3 bits are used to drive the calibration interfaces
Data	2 Bytes Byte[0]: Data available, but unused Byte[1]: bit[7-3]: available, but unused bit[2] 0=Table OFF 1=Table ON bit[1] 0=Mirror OFF 1=Mirror ON bit[0] 0=Load OFF 1=Load ON

11.3 Monitor points in Detail

Name	GET_HEMT_CAL_COMMAND
CAN ID	01 0C 01 20
Description	Reads the Calibration Command register
Data	3 Bytes Bytes[0-1]:See SET_HEMT_CAL_COMMAND Byte[2]: Error report bit[2]=CAN error

Name	GET_HEMT_CAL_STATUS
CAN ID	01 0C 01 00
Description	Reads the Calibration Status register used to monitor the limit switches. A switch “ON” is read as “1”.
Data	2 Bytes Byte[0]: Data available but unused Byte[1]: Data bits[7-6] : available, but unused.

	bits[5-4] : Table limit switches 00= Table in undefined position 01 = Table in ON position 10 = Table in OFF position 11= Should never happen bits[3-2] : Mirror limit switches 00= Mirror in undefined position 01 = Mirror in ON position 10 = Mirror in OFF position 11= Should never happen bits[1-0] : Load limit switches 00= Load in undefined position 01 = Load in ON position 10 = Load in OFF position 11= Should never happen Byte[1]: Error report bit[2]=CAN error

12 Amplifiers

Originally the bus I2C is in use for monitoring and controlling some power supplies. Yves Bortolotti has developed this interface. Get from him the applicable documentation.
The amplifiers are controlled through a CAN-I2C interface.
There are 2 polarities: Horizontal (H) and Vertical (V). Each polarity uses 2 amplifiers. (A1 and A2).
Each amplifiers has 4 parameters which can be monitored.: ID, VD, VG1 and VG2. There are thus 16 parameters to monitor.

12.1 Summary of Control and Monitor points

Name	CAN ID	Data size	Description
Set_amplifiers_RAM_byte	00 0C 02 00	1 byte	Write 1 byte into amplifiers RAM for checkout
Get_amplifiers_RAM_byte	00 0C 02 10	2 bytes	Read byte from amplifiers RAM for checkout.
Set_all_amplifiers_init	00 0C 02 20	dummy	Init ALL amplifiers. TO BE DONE FIRST.
Set_amplifiers_power	00 0C 02 30 00 0C 02 31 00 0C 02 32 00 0C 02 33 00 0C 02 3B	1 byte	Turns OFF/ON amplifier(s)
Get_amplifiers_power_status	00 0C 02 40 00 0C 02 41 00 0C 02 42 00 0C 02 43	2 bytes	Read amplifier power status
Set_amplifiers_protection	00 0C 02 50 00 0C 02 51 00 0C 02 52 00 0C 02 53 00 0C 02 58	1 byte	(Un)protects amplifier(s)
Get_amplifiers_protection_status	00 0C 02 60	2 bytes	Read amplifier protection

	00 0C 02 61 00 0C 02 62 00 0C 02 63		status
Get_amplifier _"x"_corrections	00 0C 02 7x	5 bytes	To de defined
Get_Pol_V_channel_"y"_value	00 0C 02 8y	3 bytes	16-bit value
Get_Pol_H_channel_"y"_value	00 0C 02 9y	3 bytes	16-bit value

12.2 Control points in detail

Name	Set_amplifiers_RAM_byte
CAN ID	00 0C 02 00
Description	Sets a byte into volatile RAM of amplifiers.
Data	1 byte. This byte is cleared in case of power fail or shutdown.

Name	Set_all_amplifiers_init
CAN ID	00 0C 02 20
Description	Initialises all amplifiers. To be done before turning on and unprotecting the amplifiers.
Data	1 dummy byte

Name	Set_amplifiers_power
CAN ID	00 0C 02 30 Pol V amplifier1 00 0C 02 31 Pol V amplifier2 00 0C 02 32 Pol H amplifier1 00 0C 02 33 Pol H amplifier2 00 0C 02 3B All amplifiers
Description	Turns OFF/ON the power of one amplifier or all amplifiers at a time.
Data	1 Byte: byte[0]=0 turns OFF amplifiers, byte[0]= 1 turns ON amplifiers.

Name	Set_amplifiers_protection
CAN ID	00 0C 02 50 Pol V amplifier1 00 0C 02 51 Pol V amplifier2 00 0C 02 52 Pol H amplifier1 00 0C 02 53 Pol H amplifier2 00 0C 02 58 All amplifiers
Description	Turns OFF/ON the protection of one amplifier or all amplifiers at a time. To be done after init and power turn-on.
Data	1 Byte: byte [0]=0 unprotects amplifiers, byte [0]=1 protects amplifiers.

12.3 Monitor points in detail

Name	Get_amplifiers_RAM_byte
CAN ID	00 0C 02 10
Description	Reads RAM byte. It should be equal to last written value. If not, power has been cycled. The byte should be set again and amplifiers init **must** be done

Data	2 bytes:
	Byte[0]: Data
	Byte[1]: Error report
	bit[2]=CAN error
	bit[1]=I2C write error
	bit[0]=I2C read error

Name	Get_amplifier power_status
CAN ID	00 0C 02 40 Pol V amplifier 1

	00 0C 02 41 Pol V amplifier 2 00 0C 02 42 Pol H amplifier 1 00 0C 02 43 Pol H amplifier 2
Description	Gets the status (Power ON/OFF) of one amplifier.
Data	2 bytes: Byte[0]: Data bit[3]: Pol H amplifier 2 power bit[2]: Pol H amplifier 1 power bit[1]: Pol V amplifier 2 power bit[0]: Pol V amplifier 1 power (0 == power OFF, 1 == power ON) Byte[1]: Error report bit[2]=CAN error bit[1]=I2C write error bit[0]=I2C read error

Name	Get_amplifier protection_status
CAN ID	00 0C 02 60 Pol V amplifier 1 00 0C 02 61 Pol V amplifier 2 00 0C 02 62 Pol H amplifier 1 00 0C 02 63 Pol H amplifier 2
Description	Gets the status (protection ON/OFF) of one amplifier.
Data	2 bytes: Byte[0]: Data bit[7]: Pol H amplifier 2 protection bit[6]: Pol H amplifier 1 protection bit[5]: Pol V amplifier 2 protection bit[4]: Pol V amplifier 1 protection (0 == power OFF, 1 == power ON) Byte[1]: Error report bit[2]=CAN error bit[1]=I2C write error bit[0]=I2C read error

Name	Get_amplifier “x”_corrections
CAN ID	00 0C 02 7x with x = [0-15]
Description	To be defined

Data	5 bytes
	Byte[4]: Error report
	bit[2]=CAN error
	bit[1]=I2C write error
	bit[0]=I2C read error

Name	Get_Pol_V_channel_“y”_value
CAN ID	00 0C 02 8y with y = [0-15]
Description	Reads the value of Vertical polarity channel “y”.
Data	3 bytes: Byte[0]: Data MSB

“y”value	Channel name
[15-8]	unused
7	Polar V amplifier 2 VG2
6	Polar V amplifier 2VG1
5	Polar V amplifier 2 VD
4	Polar V amplifier 2 ID
3	Polar V amplifier 1 VG2
2	Polar V amplifier 1 VG1
1	Polar V amplifier 1 VD
0	Polar V amplifier 1 ID

	Byte[1]: Data LSB
	Bits[15-0] = raw data.
	2’s compliment signed data = raw data – 2^15 (offset Binary)
	Byte[2]: Error report
	bit[2]=CAN error
	bit[1]=I2C write error
	bit[0]=I2C read error

<table>
<tr><td>Name</td><td>Get_Pol_H_channel_“y”_value</td></tr>
<tr><td>CAN ID</td><td>00 0C 02 9y with y = [0-15]</td></tr>
<tr><td>Description</td><td>Reads the value of Horizontal polarity channel “y”.
<table>
<tr><td>“y”value</td><td>Channel name</td></tr>
<tr><td>[15-8]</td><td>unused</td></tr>
<tr><td>7</td><td>Polar H amplifier 2 VG2</td></tr>
<tr><td>6</td><td>Polar H amplifier 2 VG1</td></tr>
<tr><td>5</td><td>Polar H amplifier 2 VD</td></tr>
<tr><td>4</td><td>Polar H amplifier 2 ID</td></tr>
<tr><td>3</td><td>Polar H amplifier 1 VG2</td></tr>
<tr><td>2</td><td>Polar H amplifier 1 VG1</td></tr>
<tr><td>1</td><td>Polar H amplifier 1 VD</td></tr>
<tr><td>0</td><td>Polar H amplifier 1 ID</td></tr>
</table></td></tr>
<tr><td>Data</td><td>See Get_Pol_V_channel_“y”_value</td></tr>
</table>