Diagnostic Breadboard Computer Control Manual LIGO-T0900579-v1

Patrick Kwee (patrick.kwee@aei.mpg.de)

November 17, 2009

Contents

1. Introduction

This manual describes how to perform diagnostic laser beam measurements with the computer control of the Diagnostic Breadboard (DBB). The field box connecting the DBB and the A/D and D/A cards, the Simulink model for the real time code generation, and the MEDM screens are described. Instructions for calibrating the field box and A/D and D/A cards and instructions for performing the laser beam measurements are given.

The optical setup and the analog electronic modules of the DBB are described in [3]. The measurement methods are described in the articles [4, 5].

1.1. Definitions and Acronyms

AA	autoalignment / alignment loops
CCD	charge-coupled device
DBB	diagnostic breadboard
DBID	diagnostic breadboard serial number
DWS	differential wavefront sensing
FRQ	frequency noise measurement
FSR	free spectral range
HV	high voltage
Lag	Laguerre Gaussian mode
LSD	linear spectral density
MSC	modescan measurement
PMC	pre-mode-cleaner
PNT1X	pointing measurement 1X
PNT1Y	pointing measurement 1Y
PNT2X	pointing measurement 2X
PNT2Y	pointing measurement 2Y
PZT	piezo-electric transducer
QPD	quadrant photodiode
RF	radio frequency
RIN	relative intensity noise
RINRF	relative intensity noise at radio frequencies
RPD	RIN photodiode
TEM	transversal electro magnetic mode, Hermite Gaussian mode
TF	transfer function
TIA	trans-impedance amplifier
TPD	photodiode in transmission of PMC

2. Field box

The field box is an electronic module placed in the electronic crate of the DBB and is the connection between the electronic signals of the DBB and the computer control system (Fig. 1). It also contains whitening and de-whitening filters for signal conditioning.

To control the DBB 24 A/D and 16 D/A channels are necessary. D-Sub 24 connectors are used to connect the field box to the computer control system. Since the computer control system is used for digital controlling the PMC length and the alignment PZTs, the analog electronic modules Controller 1 and Controller 2 are no longer necessary to operate the DBB.

The field box can be configured in two ways depending on the presence of these control modules. In case the two control modules are in the crate the switches on the PCB of the field box have to be in the 'ADD' position. In this configuration the signals from the computer system for the PZTs will be passed through the control modules to the HV Amplifier. If the control modules are absent the switches have to be in the 'CTRL' position. In this case the field box is driving the same lines on the crate bus which are normally driven by the control modules.

Never operate the two control modules (Controller 1 and Controller 2) and the Field Box (with PCB switches set to 'CTRL') at the same time. The modules will try to drive the same lines on the crate bus.

2.1. Cabling

The field box has to be mounted in the electronic crate of the DBB. Most signals are read and injected using the crate bus.

The three signals of the TPD photodiode have to be connected to the field box. Use Lemo 00 cables to make a connection between *board.tpd.0db* and *fbox.tpd.0db*, *board.tpd.40db* and *fbox.tpd.40db*, and *board.tpd.80db* and *fbox.tpd.80db*.

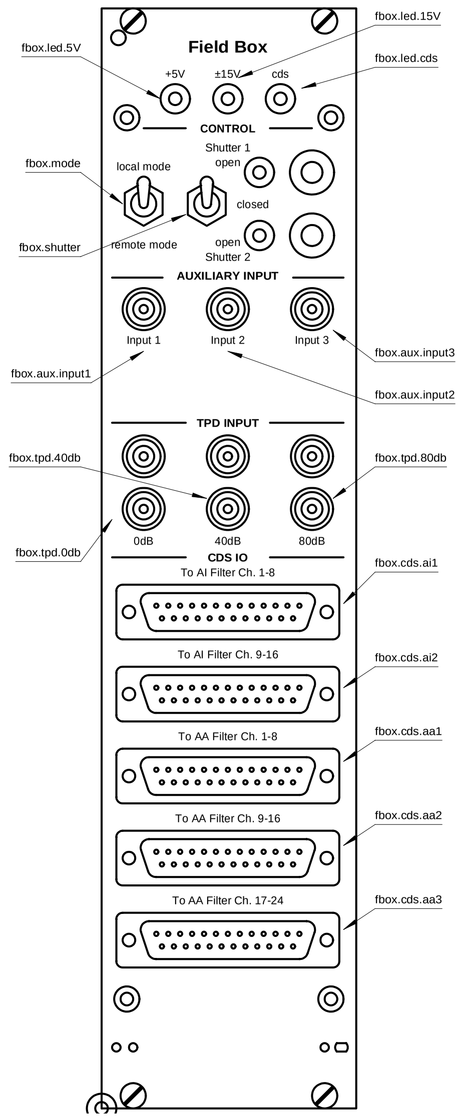
In case shutters are used for multiplexing laser beam for the DBB, two shutters (Thorlabs SH05) can be connected to the field box.

The field box has to connected to the anti aliasing and anti imaging filters of the computer system using five D-Sub 24 connectors. The channel assignment is described in the following subsection.

The field box has three auxiliary inputs which can be used for analog signals ranging from approx. -10 V to 10 V.

2.2. Channel assignment

The field box is connected via anti aliasing filters to 24 A/D converter card channels. Three DSUB24 connectors at the field box are used to interface the anti aliasing filters. The field box is connected via anti imaging filter to 16 D/A converter card channels. Two DSUB24 connectors are used at the field box. The channel assignment is given in Tab. 1, 2. To use the real time code model described in Sec. 3 the signals of the field box have to be connected to the A/D and D/A



Connector	Range	Type	Description
fbox.led.5v			Indicates a working ± 5 V power supply.
fbox.led.15v			Indicates a working ± 15 V power supply.
fbox.led.cds			Indicates an initialized computer system/cds.
fbox.mode			Selects between local and remote operation mode.
fbox.shutter			Controls the multiplexing shutters in front of the DBB.
fbox.aux.input1	-10 V. . . 10 V	in	Auxiliary input 1.
fbox.aux.input2	-10 V. . . 10 V	in	Auxiliary input 2.
fbox.aux.input3	-10 V. . . 10 V	in	Auxiliary input 3.
fbox.tpd.0db	-10 V. . . 10 V	in/out	Input and output of the TPD signals.
fbox.tpd.40db	-10 V. . . 10 V	in/out	Input and output of the TPD signals.
fbox.tpd.80db	-10 V. . . 10 V	in/out	Input and output of the TPD signals.
fbox.cds.ai1	-10 V. . . 10 V	in	Signals from anti imaging filters.
fbox.cds.ai2	-10 V. . . 10 V	in	Signals from anti imaging filters.
fbox.cds.aa1	-10 V. . . 10 V	out	Signals to anti aliasing filters.
fbox.cds.aa2	-10 V. . . 10 V	out	Signals to anti aliasing filters.
fbox.cds.aa3	-10 V. . . 10 V	out	Signals to anti aliasing filters.

Figure 1: Field box interface description.

To AA Ch. 1-8		
Pins	Name	A/D ch.
1-14	QPD1-QX	0
2-15	QPD1-QY	1
3-16	QPD1-QS	2
4-17	QPD2-QX	3
5-18	QPD2-QY	4
6-19	QPD2-QS	5
7-20	RIN-DC	6
8-21	TPD-0DB	7

To AA Ch. 9-16		
Pins	Name	A/D ch.
1-14	QPD1-DX	8
2-15	QPD1-DY	9
3-16	QPD-DS	10
4-17	QPD2-DX	11
5-18	QPD2-DY	12
6-19	PZTMON	13
7-20	HVMON	14
8-21	RINAC	15

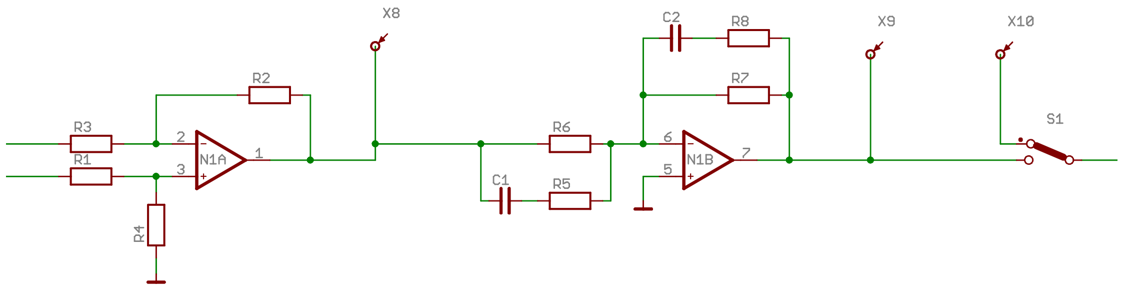
To AA Ch. 17-24		
Pins	Name	A/D ch.
1-14	TPD-40DB	16
2-15	TPD-80DB	17
3-16	SHUTTER-REQ	18
4-17	AUX-INPUT1	19
5-18	AUX-INPUT2	20
6-19	AUX-INPUT3	21
7-20	AUX-INPUT4	22
8-21	RMT-READBACK	23

Table 1: Channel assignment of DSUB24 connectors connected to the A/D card. The pins are given as 'signal pin - ground pin'.

To AI Ch. 1-8		
Pins	Name	D/A ch.
1-14	CTRL0	0
2-15	CTRL1X	1
3-16	CTRL1Y	2
4-17	CTRL2X	3
5-18	CTRL2Y	4
6-19	PHASE	5
7-20	LENS1	6
8-21	LENS2	7

To AI Ch. 9-16		
Pins	Name	D/A ch.
1-14	LENS-LATCH	8
2-15	SHUTTER-DBB	9
3-16	RMT	10
4-17	MODOFF	11
5-18	SCANMODE	12
6-19	SHUTTER-MULTIPLEX	13
7-20	FILTER-SWITCH1	14
8-21	FILTER-SWITCH2	15

Table 2: Channel assignment of DSUB24 connectors connected to the D/A card. The pins are given as 'signal pin - ground pin'.



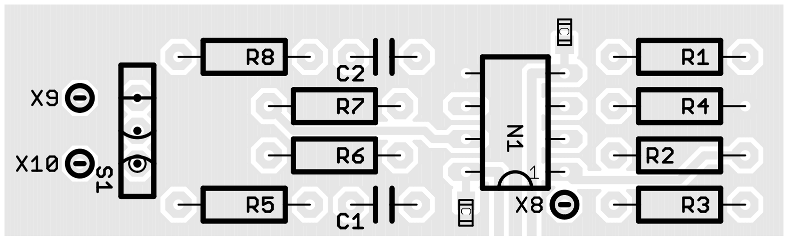


Figure 2: Schematic and board of the universal whitening and de-whitening filter of the field box.

channel number given in the tables. One A/D card with at least 24 channels and one D/A card with at least 16 channels are necessary.

2.3. Whitening and de-whitening filter

The field box contains whitening and de-whitening filter for several channels on a daughterboard. For a few uncritical channels the filters are fixed. The filter function for most other channels can be adjusted by replacing resistors and capacitors on the daughterboard board of the field box. Therefore universal filter modules consisting of a pseudo differential stage and the actual filter stage are used (Fig. 2). The transfer functions of the filter modules should be optimized for a specific laser system. Dependent on the noise of the laser system the gain and shape of the transfer function should be optimized to map the dynamic range of the A/D and D/A cards. Example filter module configurations are given in Tab. 3.

Channel	R1, R3	R2, R4	C1, R5, R6	C2, R7, R8	Compensation
`QPD1-DX` `QPD1-DY` `QPD2-DX` `QPD2-DY`	$10\,\mathrm{k\Omega}$	$10\,\mathrm{k\Omega}$	$(1\,\mathrm{nF} + 1\,\mathrm{k\Omega})\|100\,\mathrm{k\Omega}$	$100\,\mathrm{k\Omega}$	`zpk([15915], [157.6], 1, "n")`
`QPD2-DS`	$1\,\mathrm{k\Omega}$	$10\,\mathrm{k\Omega}$	$10\,\mathrm{k\Omega}$	$10\,\mathrm{k\Omega}$	`gain(0.1)`
`PZTMON`	$1\,\mathrm{k\Omega}$	$10\,\mathrm{k\Omega}$	$3.3\,\mathrm{k\Omega}$	$10\,\mathrm{k\Omega}$	`gain(0.303)`
`RIN-AC` `CTRL0`	$10\,\mathrm{k\Omega}$	$10\,\mathrm{k\Omega}$	$10\,\mathrm{k\Omega}$	$10\,\mathrm{k\Omega}$	`gain(1)`
`CTRL1X` `CTRL1Y` `CTRL2X` `CTRL2Y`	$10\,\mathrm{k\Omega}$	$10\,\mathrm{k\Omega}$	$10\,\mathrm{k\Omega}$	$150\,\mathrm{nF}\|10\,\mathrm{k\Omega}$	`zpk([106], [],1,"n")`

Table 3: Example whitening and de-whitening filters for the field box. Styroflex capacitors and 0.1% resistors should be used to provide long term stability. In order to compensate the hardware filters the parameters for the digital compensation filters are given.

3. Control

3.1. Real time code model

The DBB is controlled by the computer using a real time code which is generated from a Simulink model[1]. The real time code includes, among other things, the PMC length control loops and four alignment control loops, signal de-whitening of several photodiode signals, a software interlock to prevent PZT oscillation, a ramp generator for mode scans, an automatic lock acquisition for the PMC length control loop, and an automatic error signal calibration module. The real time code operates in one of six possible operation modes:

1. Interlock mode: This mode is activated if a hardware or software interlock occurs. All outputs to the DBB are set to default values, the shutter is closed, and the 1 MHz dither modulation is turned off. To reset the interlocks the system must be set to standby mode.
2. Standby mode: This mode is almost identical to the interlock mode: The outputs are set to default values, the shutter is closed, the lens movement is latched, and the dither lock modulation is turned off. Put the DBB into this mode if no measurement should be performed. This mode also resets a possible interlock.
3. Manual mode: The PMC length can be adjusted in this mode manually. The PMC length control loop is deactivated. This mode is primarily used to activate the prealignment loops with unlocked PMC.
4. Scan mode: The PMC length is scanned with a ramp signal in this mode. Mode scans can be performed in this operation mode. The dither modulation is turned off and the PMC PZT HV amplifier is set to the high bandwidth mode.
5. Lock mode: The PMC length control loop is closed including an automatic lock acquisition. The auto alignment loops can be closed if the PMC is resonant. This mode is used to perform frequency noise and pointing noise measurements.
6. Local mode: This mode indicates that the electronic modules are set by a switch at the field box to the local mode. In this mode the DBB electronics cannot be controlled by the computer.

The Simulink model is roughly described in following subsections. The function of the blocks used in the model are described in [1]. The source code for the function call blocks are given in the appendix.

3.1.1. Main model

The main model (Fig. 3) contains the A/D card, the D/A card, and several interconnected subsystems. In the following subsections the subsystems are described.

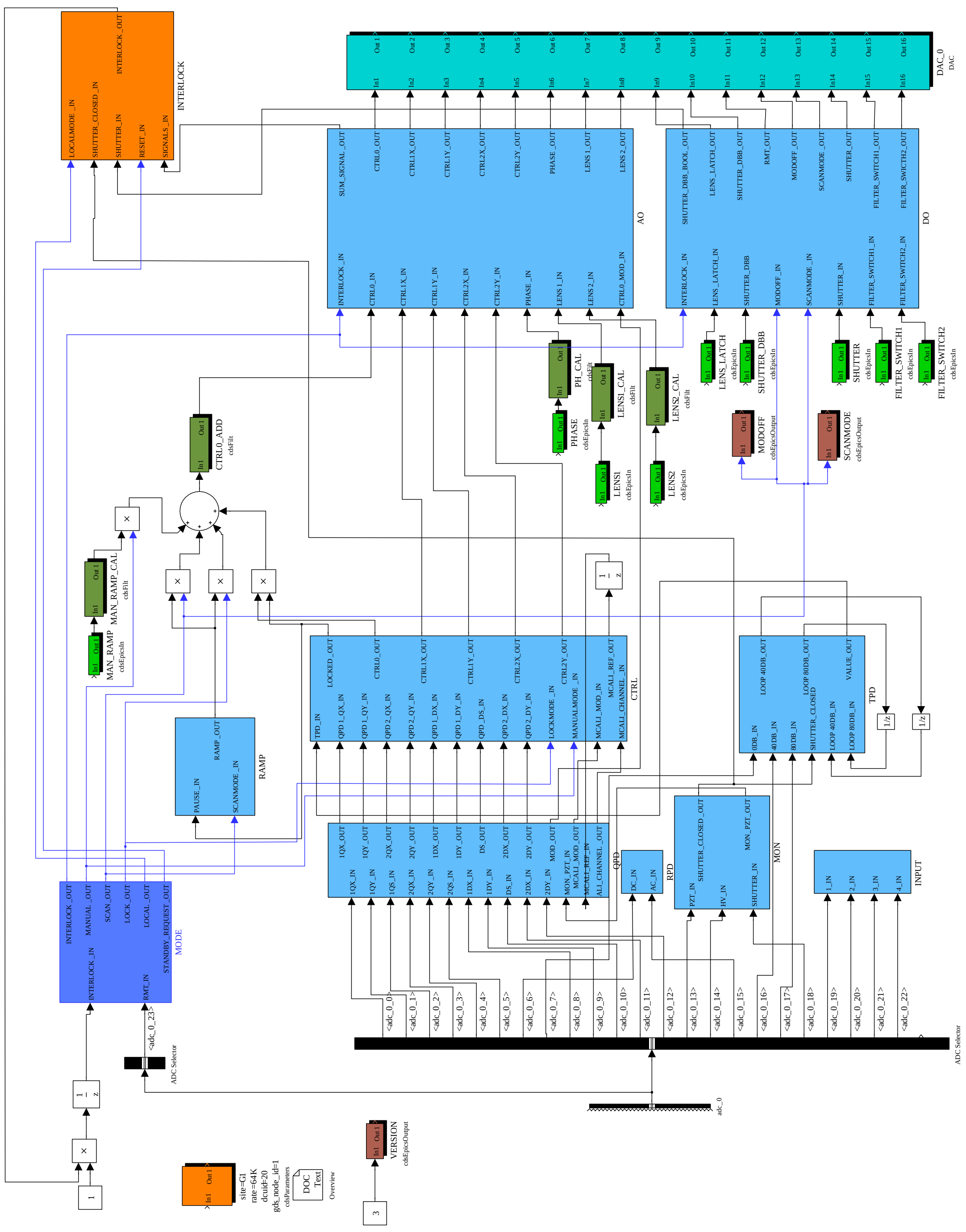


Figure 3: Main real time code model.

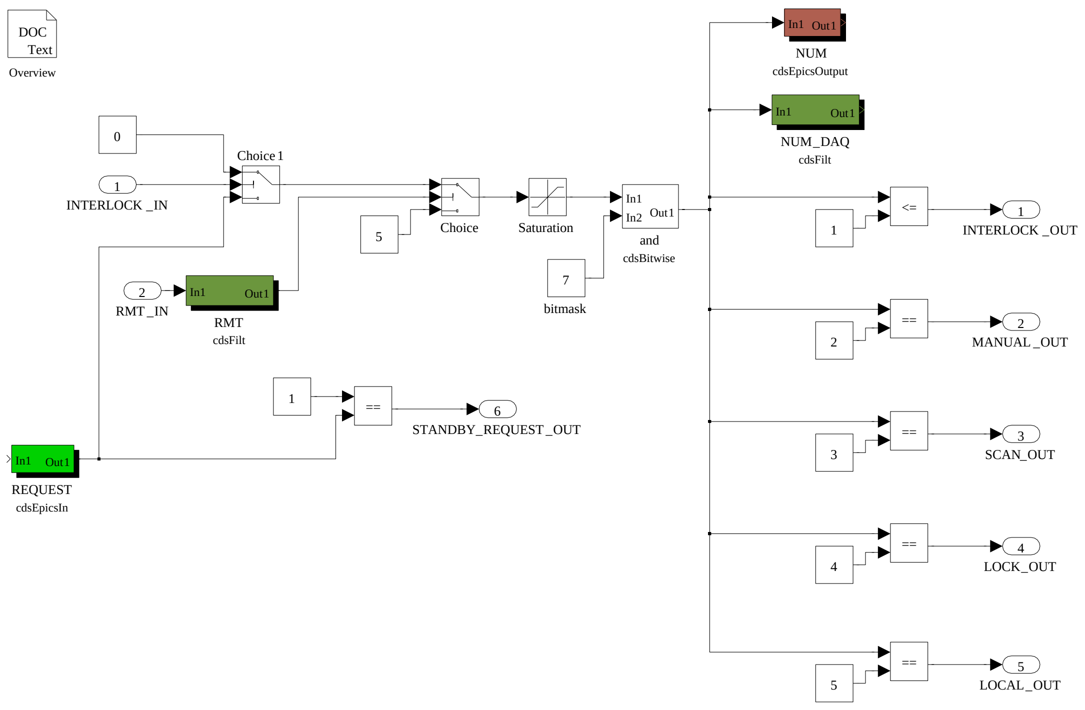


Figure 4: MODE subsystem.

3.1.2. MODE subsystem

The MODE subsystem (Fig. 4) is used to determine the current operation mode of the DBB. With the EPICS variable `DBB-MODE_REQUEST` a specific operation mode can be requested from the user. The actual current operation mode can be obtained from `DBB-MODE_NUM`. The requested mode can be overridden by an interlock or the local operation mode in case the local mode switch on the field box is turned on.

3.1.3. QPD subsystem

All signals from the quadrant photodiodes are processed in this subsystems (Fig. 5). The dc signals (all Q-signals) of the QPDs are connected to filter modules for the DAQ. The demodulated signals (all D-signals) are passed through filter modules and then to the next subsystems.

All demodulated signals, also called error signals, are connected to an automatic calibration software module implemented as C code functions `QPD_AUTOCALI.c` and `QPD_MULTICALI.c` . The automatic calibration injects a signal to the control signals of the loops above the unity gain frequency. The error signals are demodulated using this calibration signal and a signal from the PZTs is also demodulated as reference. A calibration factor is calculated and the D-signals are multiplied with this factor to get an calibrated error signal (all `_CAL` signals).

3.1.4. MON subsystem

All monitor signals from the DBB are passed into this subsystem (Fig. 6). The PZT and HZ monitor signals are passed through filter modules for calibration and the shutter read back signal is used to determine the status of the shutter.

3.1.5. RPD subsystem

The signals from the RPD photodiode used for relative power noise measurements are passed into this subsystem (Fig. 7). The signals are calibrated and pass de-whitening filters. A relative power signal is calculated and passed to a filter module for the DAQ (`DBB-RPD_REL_PWR`). Furthermore the photo current and the shot noise level is calculated via a C code function `RPD_CALC_SHOTNOISE.c`.

3.1.6. TPD subsystem

The signals from the TPD photodiode in transmission of the PMC are passed into this subsystem (Fig. 8). The three signals are combined to one signal with higher dynamic range with a C code function `TPD_SELECT.c` and passed into a filter module for the DAQ (`DBB-TPD_VALUE`). An automatic offset compensation is performed for the 40 dB and 80 dB signals. As soon as the shutter is closed the offset is measured and subtracted with a very slow integrator feedback loop.

3.1.7. INPUT subsystem

This subsystem (Fig. 9) contains the signals of the auxiliary inputs from the field box module.

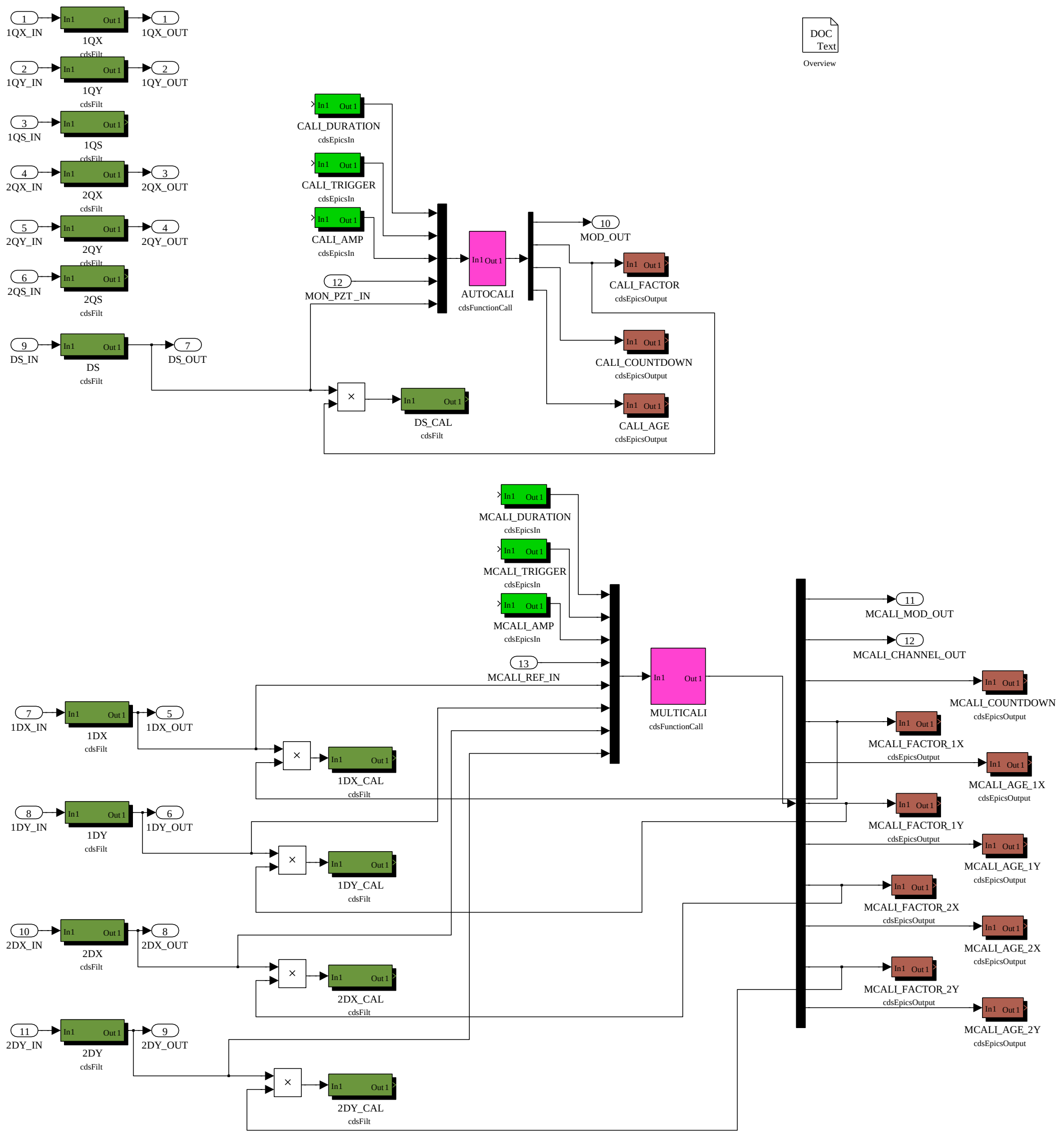


Figure 5: QPD subsystem.

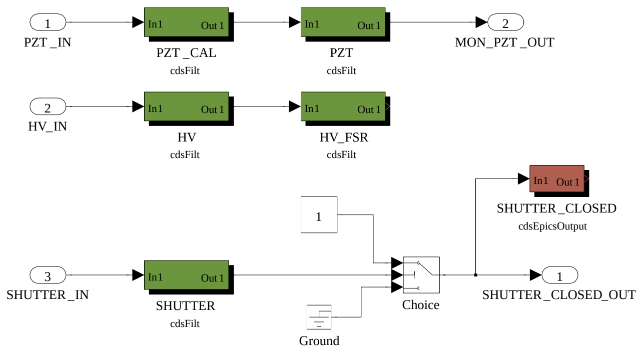


Figure 6: MON subsystem.

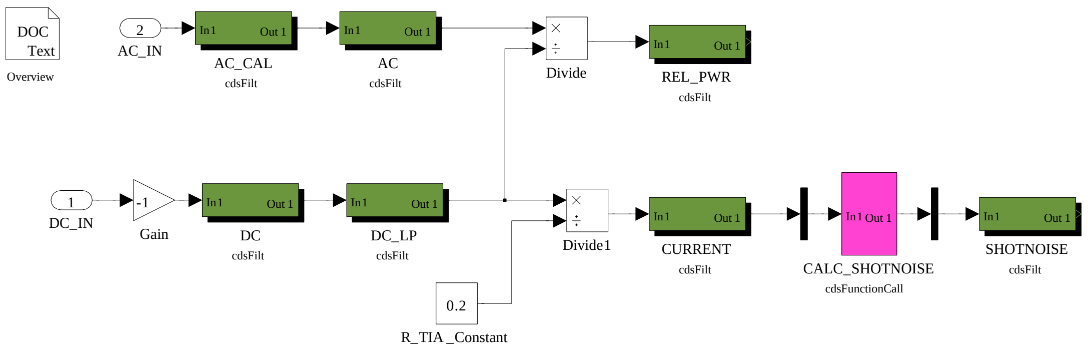


Figure 7: RPD subsystem.

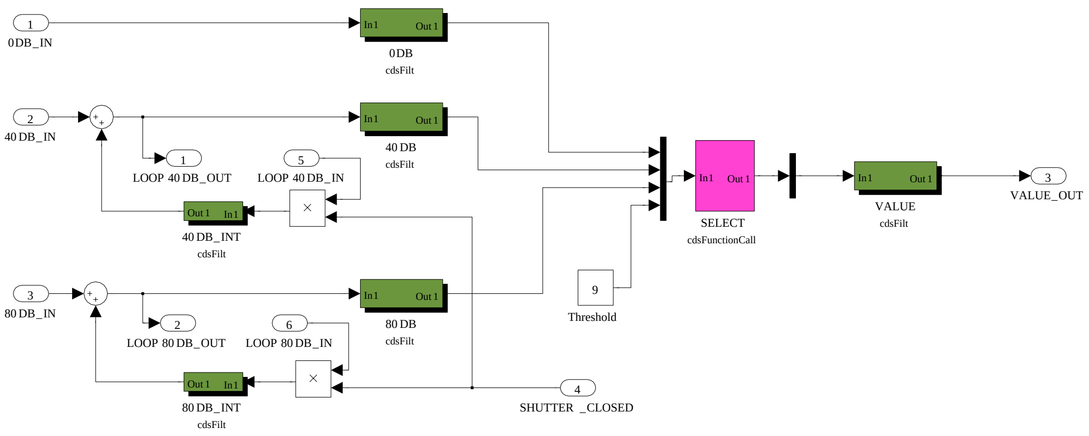


Figure 8: TPD subsystem.

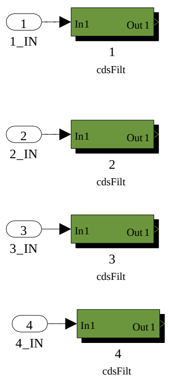


Figure 9: INPUT subsystem.

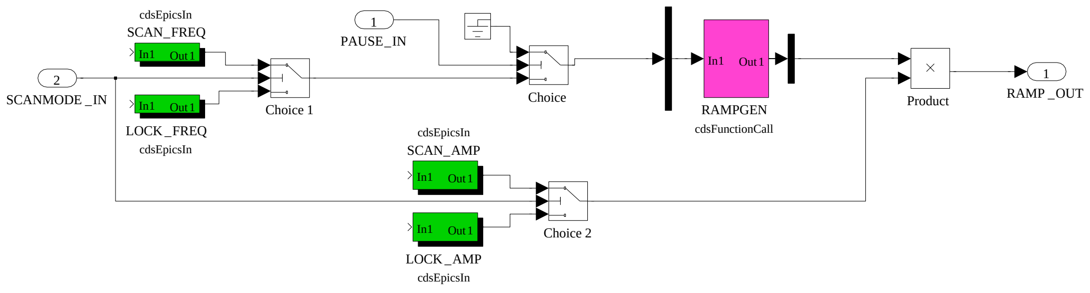


Figure 10: RAMP subsystem.

3.1.8. RAMP subsystem

This subsystem (Fig. 10) contains a ramp generator used for scanning the PMC in scan mode and for lock acquisition of the PMC length control loop. The amplitude and frequency can be set individually for the two ramp types. The ramp can be furthermore paused for the lock acquisition procedure. The C code function `RAMP_RAMPGEN.c` is used to generated the actual ramp.

3.1.9. CTRL subsystem

This subsystem (Fig. 11) contains the control loop filters for the five feedback loops and the lock acquisition logic. The PMC is considered resonant as soon as a threshold for the TPD signal is passed. If the PMC is resonant the PMC length control loop is closed if the DBB is in lock operation mode.

The four dc and the four error signals of the QPDs are mixed with two matrices. Depending on the operation mode and on the status of the pre- and auto-alignment the input signals are passed through filter modules for the control loops.

The control signals of the alignment PZTs are passed though filter modules for calibration and de-whitening. Furthermore signals for the automatic error signal calibration are injected

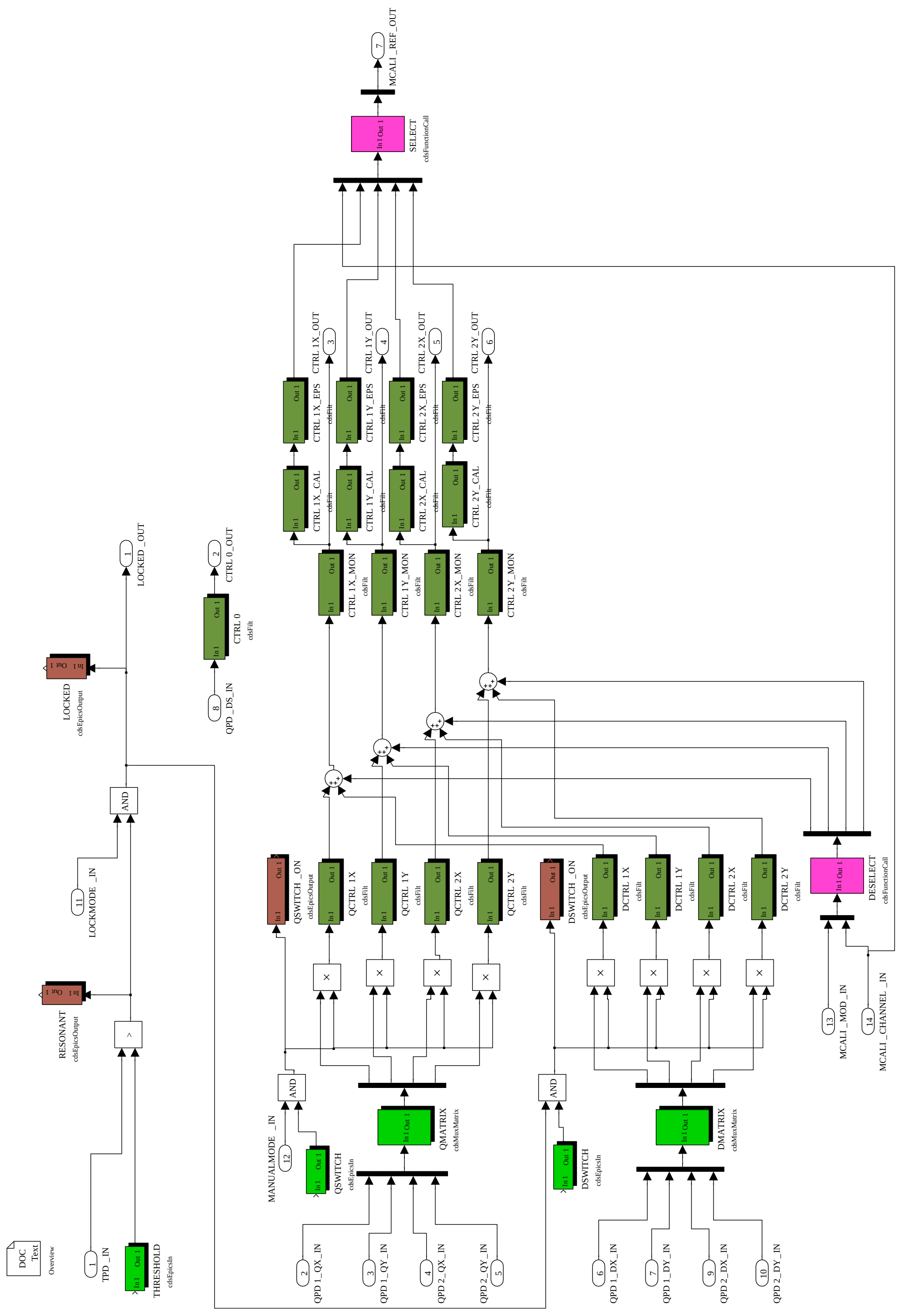


Figure 11: CTRL subsystem.

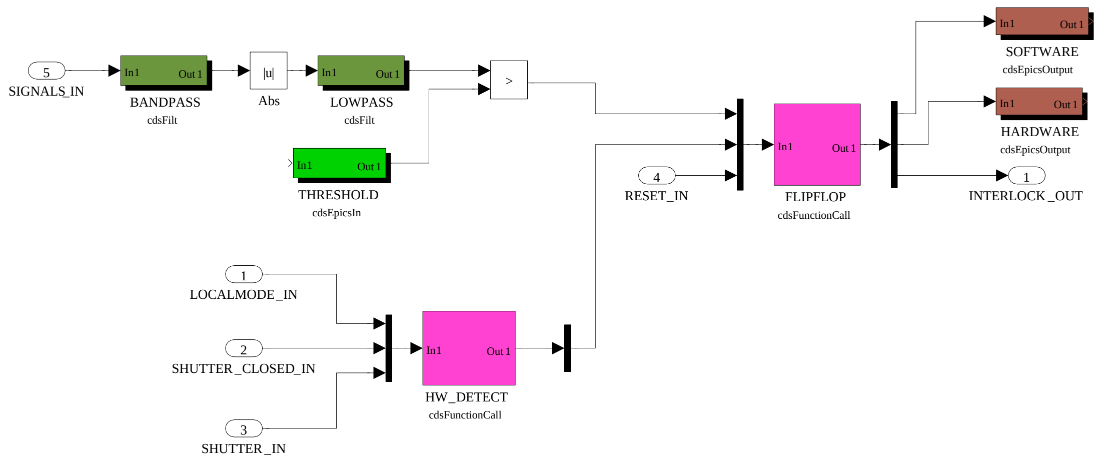


Figure 12: INTERLOCK subsystem.

and read back. C code functions `CTRL_SELECT.c` and `CTRL_DESELECT.c` are used for multiplexing the signals for the error signal calibration.

3.1.10. INTERLOCK subsystem

This subsystem (Fig. 12) contains the software interlock and the hardware interlock detector. A combined signal of all five control signals is passed through a bandpass filter, an absolute value block, and a low pass filter. The software interlock is triggered as soon as the signal is larger than a given threshold.

The interlock status of the DBB is saved with a dual flipflop block that is reset as soon as the system is put into standby mode.

The dual flipflop block is implemented as C code function `INTERLOCK_FLIPFLOP.c` . The C code function `INTERLOCK_HW_DETECT.c` contains the logic to detect a hardware interlock.

3.1.11. AO subsystem

All analog output signals are passed through this subsystem (Fig. 13) before they are passed to the D/A card. The combined signal for the PZT oscillation interlock is generated and the interlock shut down is implemented in this subsystem.

3.1.12. DO subsystem

All digital output signals are passed through this subsystem (Fig. 14) before they are passed to the D/A card. Since an analog output card is used the digital signals are gained to produce TTL compatible signal levels. The interlock is implemented in this module and generates appropiate default signals.

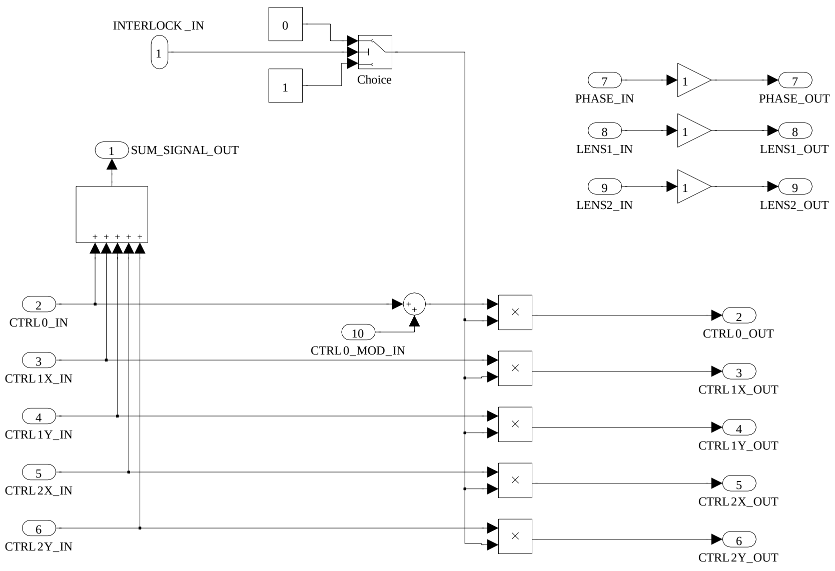


Figure 13: AO subsystem.

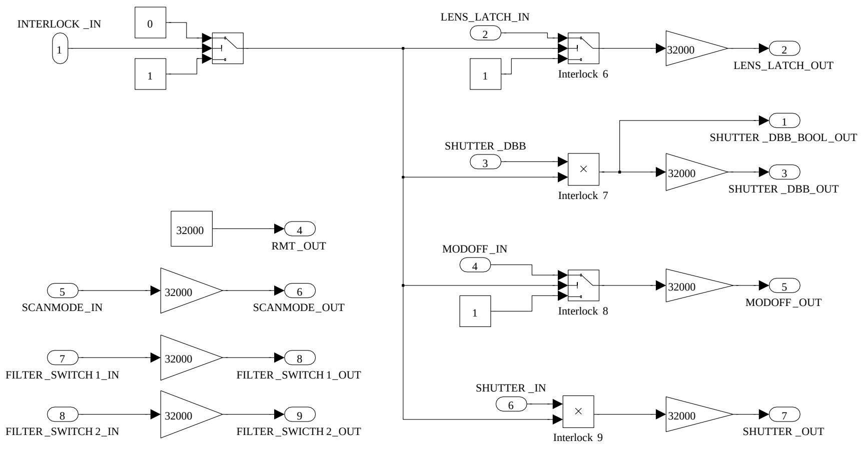
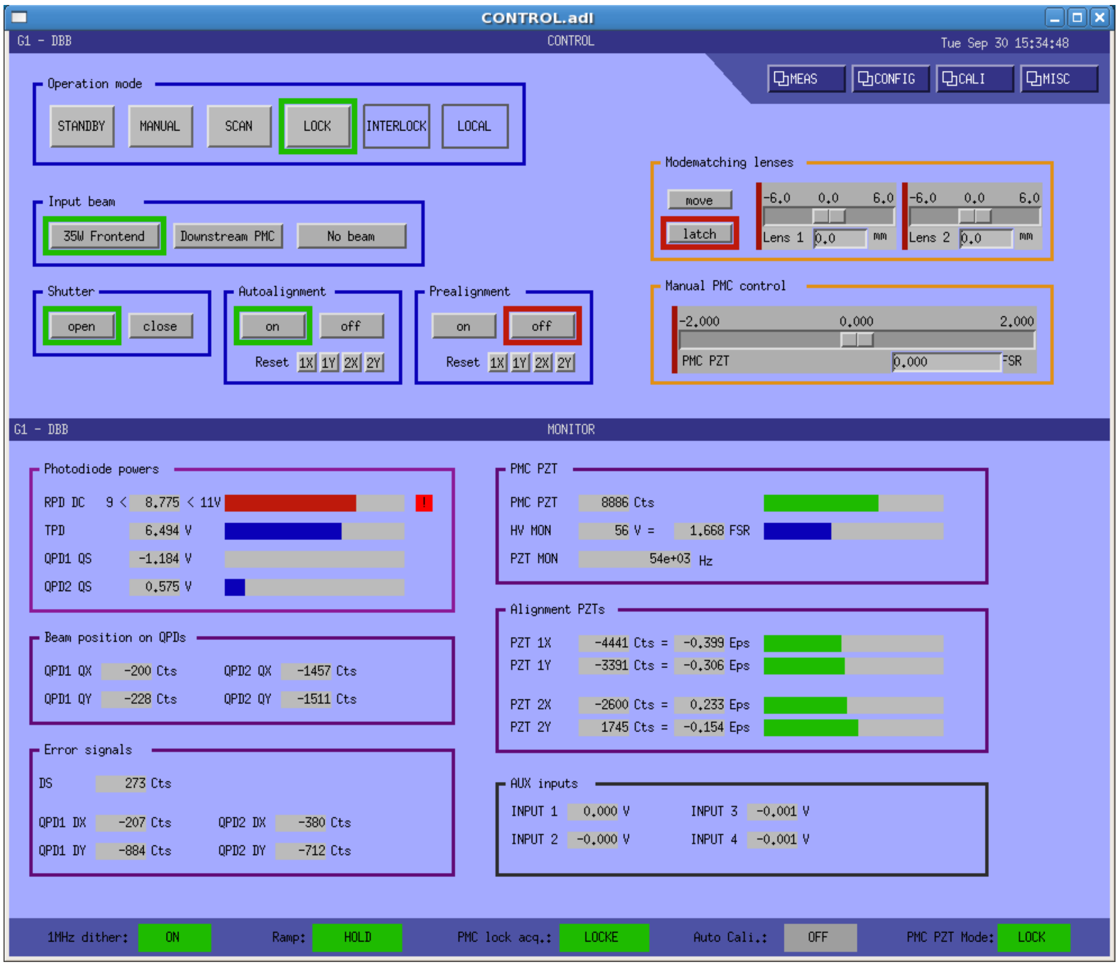


Figure 14: DO subsystem.



```
G1 - DBB                          CONTROL                     Tue Sep 30 15:34:48

Operation mode: LOCK
Input beam: 35W Frontend
Shutter: open
Autoalignment: on
Prealignment: off

Modematching lenses
  Lens 1  -6.0  0.0  6.0   0.0 mm
  Lens 2  -6.0  0.0  6.0   0.0 mm

Manual PMC control
  -2.000      0.000      2.000
  PMC PZT     0.000 FSR

G1 - DBB                          MONITOR

Photodiode powers
  RPD DC   9 <  8.775 < 11V
  TPD           6.494 V
  QPD1 QS      -1.184 V
  QPD2 QS       0.575 V

Beam position on QPDs
  QPD1 QX   -200 Cts      QPD2 QX   -1457 Cts
  QPD1 QY   -228 Cts      QPD2 QY   -1511 Cts

Error signals
  DS         273 Cts

  QPD1 DX   -207 Cts      QPD2 DX   -380 Cts
  QPD1 DY   -884 Cts      QPD2 DY   -712 Cts

PMC PZT
  PMC PZT     8886 Cts
  HV MON        56 V =    1.668 FSR
  PZT MON     54e+03 Hz

Alignment PZTs
  PZT 1X   -4441 Cts =  -0.399 Eps
  PZT 1Y   -3391 Cts =  -0.306 Eps

  PZT 2X   -2600 Cts =   0.233 Eps
  PZT 2Y    1745 Cts =  -0.154 Eps

AUX inputs
  INPUT 1   0.000 V      INPUT 3  -0.001 V
  INPUT 2  -0.000 V      INPUT 4  -0.001 V

1MHz dither: ON   Ramp: HOLD   PMC lock acq.: LOCKE   Auto Cali.: OFF   PMC PZT Mode: LOCK
```

Figure 15: Main control screen for the DBB.

3.2. Control screen

MEDM screens are used as user interface for the real time code [2]. The main control screen of the DBB is divided into two parts (Fig. 15). The upper parts contains all important control elements and the lower part monitors for the most important signals. The upper right corner contains menus for further screens which are described in the sections 4, 5, and 6 (Fig. 16).

In the control part of the screen the operation mode of the DBB (described in Sec. 3.1) and the input beam can be chosen. Furthermore the shutter on the DBB can be opened or closed. The prealignment and auto alignment control loops can be turned on or off and the integrators of the control loop filters can be reseted in case a control signal saturates. The prealignment loop can only be turned on in manual mode since the PMC have to be non-resonant. The autoalignment loop can only be turned on if the PMC is locked. Furthermore the screen contains controls for the positions of the two mode matching lenses and a control for the PMC PZT voltage which is activated only in manual mode.

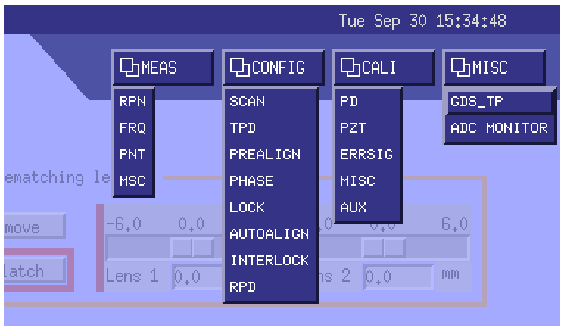
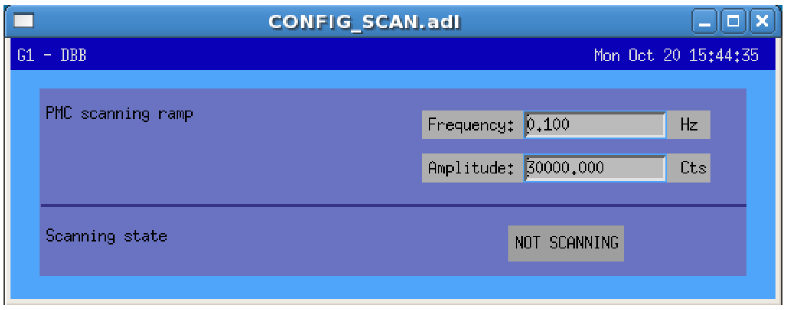


Figure 16: Menu of the main control screen.

The monitor part of the screen contains signals from the photodiodes and the control signals for the PZTs. A bar at the bottom of the screen shows important status information.


CONFIG_SCAN.adl

G1 - DBB   Mon Oct 20 15:44:35

PMC scanning ramp

Frequency: 0.100 Hz

Amplitude: 30000.000 Cts

Scanning state   NOT SCANNING

Figure 17: Configuration screen for the scanning mode.

4. Configuration

The configuration screens are used to enter the configuration of the real time code model including, e.g., the five feedback control loops, lock acquisition settings, and software interlock thresholds. The screens are accessible through the menu of the main control screen. The configuration is saved in several EPICS variables and the filter modules of the real time code. Therefore the values have to be saved before the the real time code is shut down e.g. with the BURT tool.

Most of the configuration settings needs to be adjusted only during the installation of the DBB. Typical configuration values and filter modules are given in the text.

4.1. Scanning configuration

The PMC PZT is scanned in the scan operation mode for performing mode scans. The amplitude and frequency of the ramp used can be adjusted in this screen (Fig. 17).

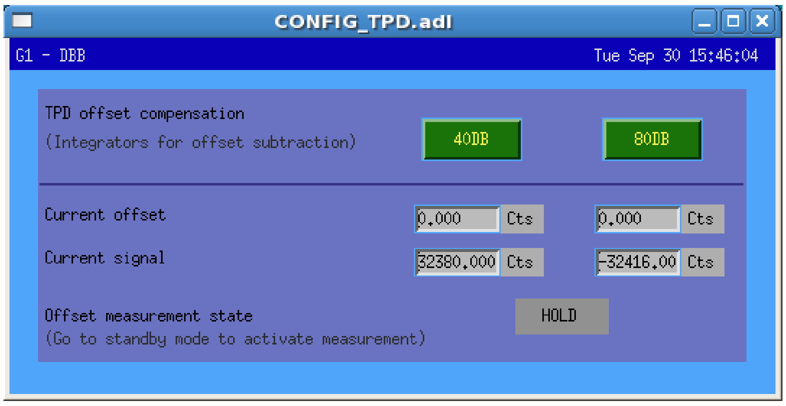
The non-linearity of the PMC PZT needs to be determined to calibrate mode scans. This non-linearity depends on the frequency and amplitude of the scanning ramp. The typical frequency is 1 Hz with full amplitude (meaning a full modulation of the HV monitor signal). In case the amplitude or frequency is changed the PMC PZT calibration polynom needs to be updated (see [3]).

4.2. TPD photodiode configuration

The electronic offsets of the 40 dB and 80 dB signals of the TPD photodiode are subtracted automatically. As soon as the shutter of the DBB is closed two slow feedback loops with integrators are closed to measure and subtract the offset.

The two filter modules that are linked in the configuration screen (Fig. 18) have to contain these integrators. The unity gain frequency of these feedback loops should be very low, e.g. 0.01 Hz. A typical filter module would be `zpk([],[0],0.01,"n")` .

The current offset which is subtracted from the input signal and the signal after this subtraction is shown. If the shutter is closed the average ’current signal’ should be zero.


CONFIG_TPD.adl

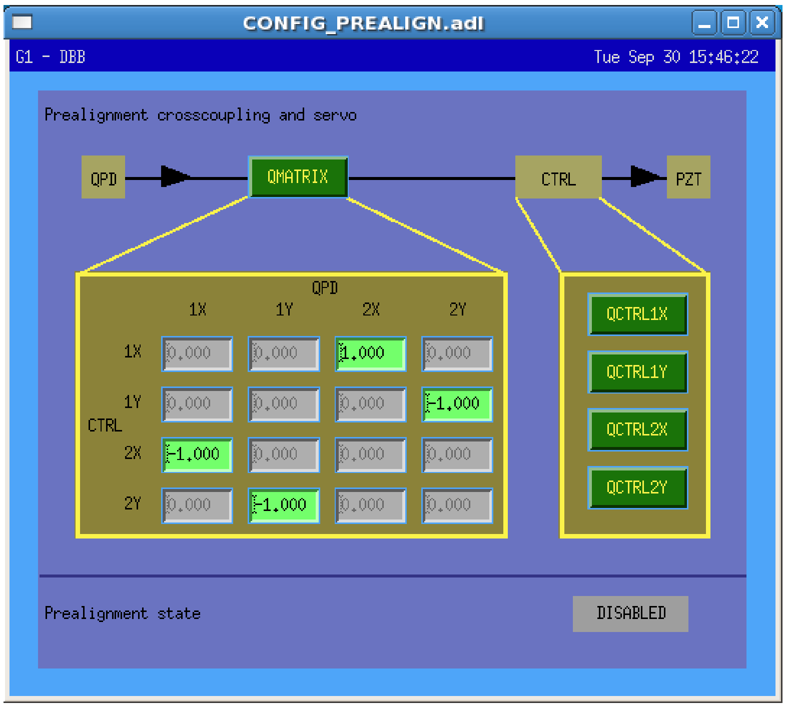
G1 - DBB   Tue Sep 30 15:46:04

TPD offset compensation
(Integrators for offset subtraction)

	40DB	80DB
Current offset	0.000 Cts	0.000 Cts
Current signal	32380.000 Cts	-32416.00 Cts

Offset measurement state   HOLD
(Go to standby mode to activate measurement)

Figure 18: Configuration screen for the TPD photodiode signals.


CONFIG_PREALIGN.adl

G1 - DBB   Tue Sep 30 15:46:22

Prealignment crosscoupling and servo

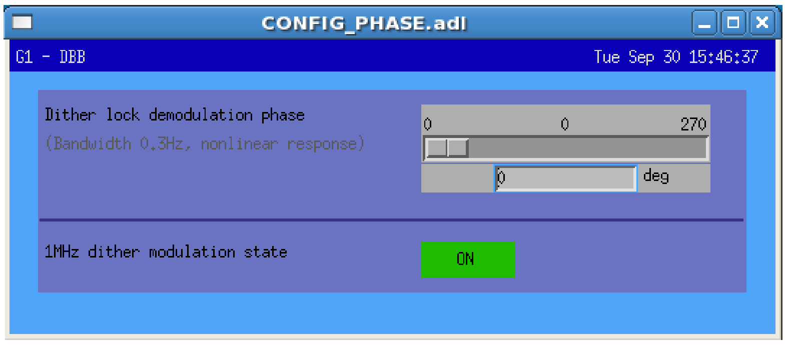
QPD → QMATRIX — CTRL → PZT

CTRL \ QPD	1X	1Y	2X	2Y
1X	0.000	0.000	1.000	0.000
1Y	0.000	0.000	0.000	-1.000
2X	-1.000	0.000	0.000	0.000
2Y	0.000	-1.000	0.000	0.000

QCTRL1X
QCTRL1Y
QCTRL2X
QCTRL2Y

Prealignment state   DISABLED

Figure 19: Configuration screen for the prealignment control loops.


CONFIG_PHASE.adl

G1 - DBB Tue Sep 30 15:46:37

Dither lock demodulation phase
(Bandwidth 0.3Hz, nonlinear response)

0 0 270

0 deg

1MHz dither modulation state ON

Figure 20: Configuration screen for the demodulation phase.

4.3. Prealignment configuration

The DC signals of the QPDs can be used to perform a prealignment of the incoming beam to the DBB without locking the PMC. The prealignment can only be turned on in 'Manual mode'.

The prealignment consists of four slow control loops (Fig. 19). The beam position on both quadrant photodiodes serve as error signals (QPD1-QX, QPD1-QY, QPD2-QX, QPD2-QY) and the alignment PZTs are used as actuators. The matrix is used to decouple the four loops. The four filter modules contain the control servo filter which should be real integrators such that the alignment is kept when the prealignment is turned off. The unity gain frequency should be rather low, e.g. 1 Hz. A typical filter module would be `zpk([],[0],1,"n")`

4.4. Demodulation phase configuration

The demodulation phase has to be adjusted if the cable lengths of the modulation signal or the quadrant photodiodes have changed. The phase shifter is non-linear and the phase shifting value is given only for a rough estimate in degree (Fig. 20). The phase should be adjusted by optimizing the error signal QPD-DS. In 'Manual mode' a ramp can be injected to the CTRL0-ADD filter module and the error signal in the QPD-DS channel can be optimized for maximal amplitude.

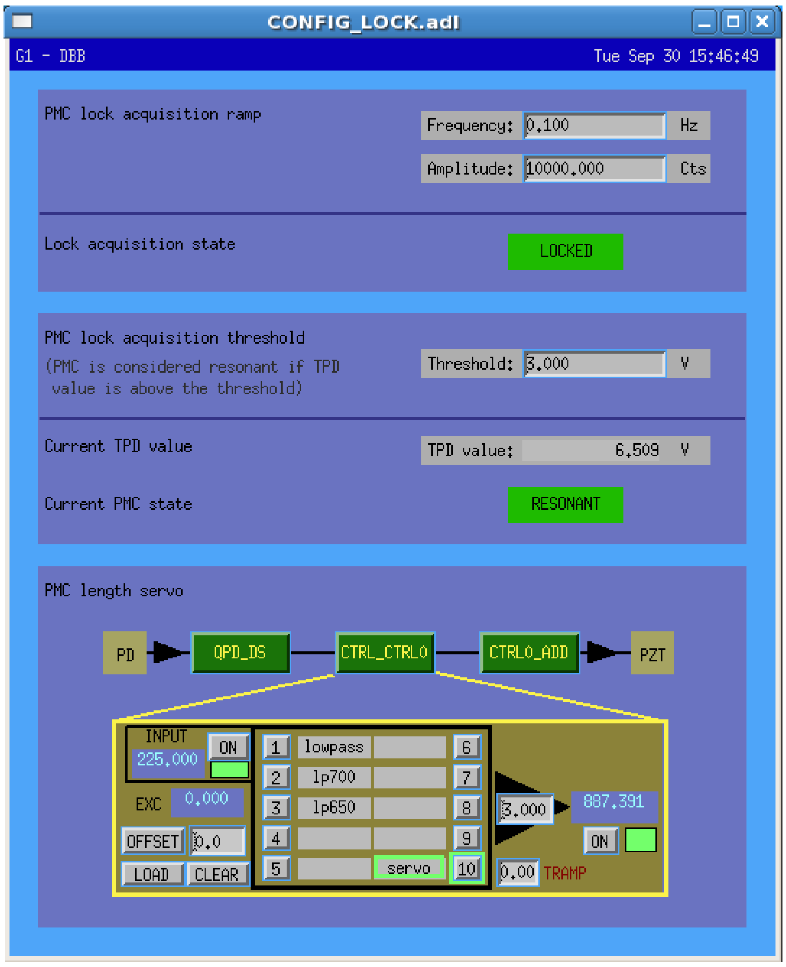
By shifting the phase it is possible to introduce a sign in the PMC length and the alignment control loop – be aware of this.

4.5. PMC length control configuration

The configuration screen (Fig. 21) is used to set the parameters for the lock acquisition of the PMC and the servo loop.

For lock acquisition the PMC length is scanned with a ramp signal. The frequency and amplitude of this signal can be adjusted in this screen. The amplitude should be chosen such that at least one FSR is scanned.

The photodiode TPD in transmission of the PMC is used to detect a resonance of the PMC. The PMC is considered resonant if the light power on the TPD photodiode is larger than the


CONFIG_LOCK.adl

G1 - DBB Tue Sep 30 15:46:49

PMC lock acquisition ramp

Frequency: 0.100 Hz

Amplitude: 10000.000 Cts

Lock acquisition state LOCKED

PMC lock acquisition threshold
(PMC is considered resonant if TPD
value is above the threshold)

Threshold: 3.000 V

Current TPD value TPD value: 6.509 V

Current PMC state RESONANT

PMC length servo

PD → QPD_DS — CTRL_CTRL0 — CTRL0_ADD → PZT

INPUT 225.000 ON

EXC 0.000

OFFSET 0.0

LOAD CLEAR

1 lowpass 6

2 lp700 7

3 lp650 8

4 9

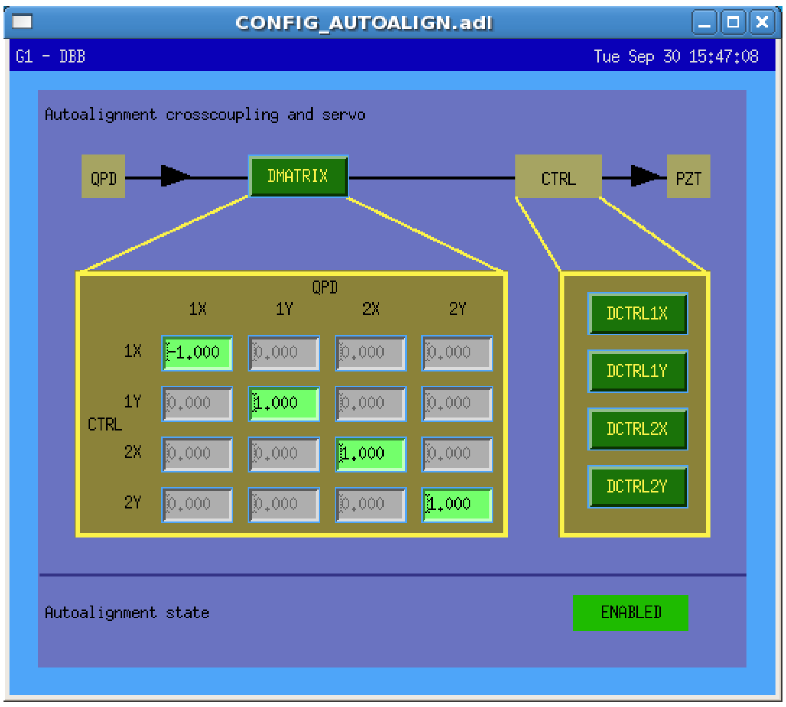
5 servo 10

3.000 887.391

ON

0.00 TRAMP

Figure 21: Configuration screen for the PMC length control loop.



CTRL \ QPD	1X	1Y	2X	2Y
1X	-1.000	0.000	0.000	0.000
1Y	0.000	1.000	0.000	0.000
2X	0.000	0.000	1.000	0.000
2Y	0.000	0.000	0.000	1.000

Figure 22: Configuration screen for the auto alignment control loops.

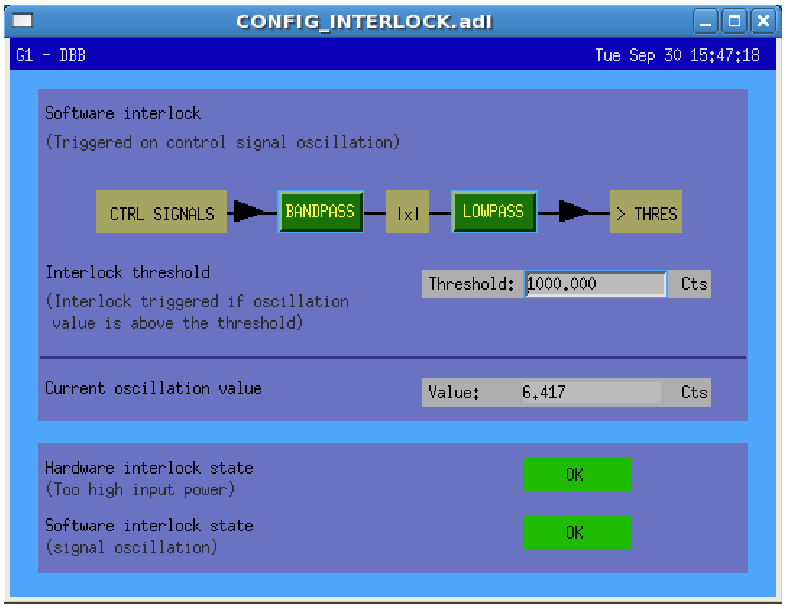
given threshold value. The lock acquisition ramp is stopped and the length control loop is closed as soon the PMC is resonant.

The filter module `CTRL_CTRL0` contains the control servo filter. The filter module must not contain a real integrator since the integrator cannot be reset automatically in case lock is lost. The control servo filter contains typically notches for the PMC PZT mechanical resonances. The unity gain frequency of the length control loop is typically 1 kHz. A typical control servo looks like: `zpk([30],[0.1],300,"n") zpk([100],[1],1,"n") notch(12487,4,30) notch(18894,4,30) notch(29628,2,30)`.

4.6. Auto alignment configuration

The DWS signals are used to align the incoming beam to the PMC while it is locked. The auto alignment can only be turned on if the PMC is locked.

The auto alignment consists of four slow control loops (Fig. 22). The DWS signals on both quadrant photodiodes serve as error signals (QPD1-DX, QPD1-DY, QPD2-DX, QPD2-DY) and the alignment PZTs are used as actuators. The matrix can be used to decouple the four alignment loops. The four filter modules contain the control servo filter which should be real integrators such that the alignment is kept when the auto alignment is turned off. The unity gain frequency should be rather low, e.g. 1 Hz. A typical filter module would be `zpk([],[0],0.1,"n")`.


CONFIG_INTERLOCK.adl

G1 - DBB Tue Sep 30 15:47:18

Software interlock
(Triggered on control signal oscillation)

Interlock threshold
(Interlock triggered if oscillation
value is above the threshold)

Threshold: 1000.000 Cts

Current oscillation value

Value: 6.417 Cts

Hardware interlock state
(Too high input power) OK

Software interlock state
(signal oscillation) OK

Figure 23: Configuration screen for the interlock.

4.7. Interlock configuration

The control signals of the five PZTs are monitored for oscillations. As soon as a oscillation is above a given threshold the DBB is set to the ’Interlock mode’. To reset the interlock the DBB has to be set to the ’Standby mode’. This software interlock is used to protect the PZTs from control loop oscillations.

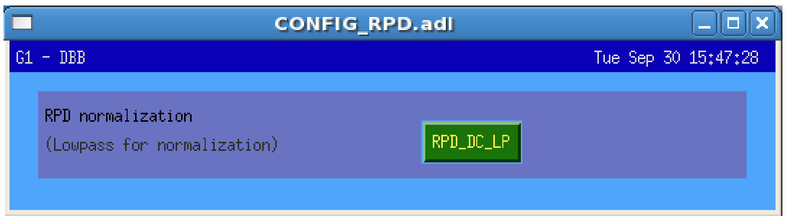
All five control signals are added and the sum signal is passed through a bandpass filter (Fig. 23). The bandpass filter should cut off all low frequency control signals but should pass all possible control loop oscillations. The absolute value of this signal is calculated and passed through a low pass filter afterwards. This signal is then used to trigger an interlock if a certain threshold is reached.

A typical filter for the bandpass is `butter("HighPass",4,50)` and for the low pass `butter("LowPass",4,0.3)`.

In principal it is possible to overwrite the interlock by disabling the bandpass or low pass filter.

4.8. RPD photodiode configuration

The DC signal of the RPD photodiode is used to normalize the AC signal such that a relative power signal is generated. The DC signal is passed through a low pass filter before the signal is used to normalize the AC signal. The corner frequency of this low pass should be rather low, e.g. 0.1 Hz (Fig. 24). A typical filter would be `butter("LowPass",4,0.1)`.



CONFIG_RPD.adl

G1 - DBB Tue Sep 30 15:47:28

RPD normalization
(Lowpass for normalization) RPD_DC_LP

Figure 24: Configuration screen for the RPD photodiode.

5. Calibration

The calibration screens are used to enter the calibration of the DBB and the computer interface, including the field box, the anti aliasing and imaging filters, and the A/D and D/A cards. The calibration is saved in several EPICS variables and the filter modules of the real time code. Therefore the values have to be saved before the the real time code is shut down e.g. with the BURT tool. The screens are accessible through the menu of the main control screen.

Most calibrations are long term stable and need only be adjusted if, e.g., the DBB is exchanged or the whitening filters inside the field box are changed.

To simplify the long term calibration procedure a dedicated electronics module is used (Fig. 25). This calibration module contains a stable voltage reference and a white noise generator. The white noise generator has a level of $100\,\mathrm{uVHz}^{-1/2} \pm 1\,\mathrm{dB}$ from 6 Hz to 30 kHz. These signals can be used for calibrating the DBB and the computer system. Furthermore the control signals for the alignment PZTs can be monitored and signals can be injected for the *pztmon* signal. A breakout box for the QPD connectors is integrated in the front of the module.

The calibration of the error signals depends on many parameters such as the input power, demodulation phase, or the beam quality. Therefore the calibration of the error signals should be performed right before a corresponding measurement. These calibrations are performed automatically by the real time code, but have to be triggered manually since calibration signals will be injected.

5.1. Photodiode calibration

This screen is used to enter the dc calibration for the four photodiodes (TPD, RPD, QPD1, QPD2) and to set de-whitening filters in order to compensate photodiode internal signal conditioning, field boxes and anti-aliasing filters (Fig. 26).

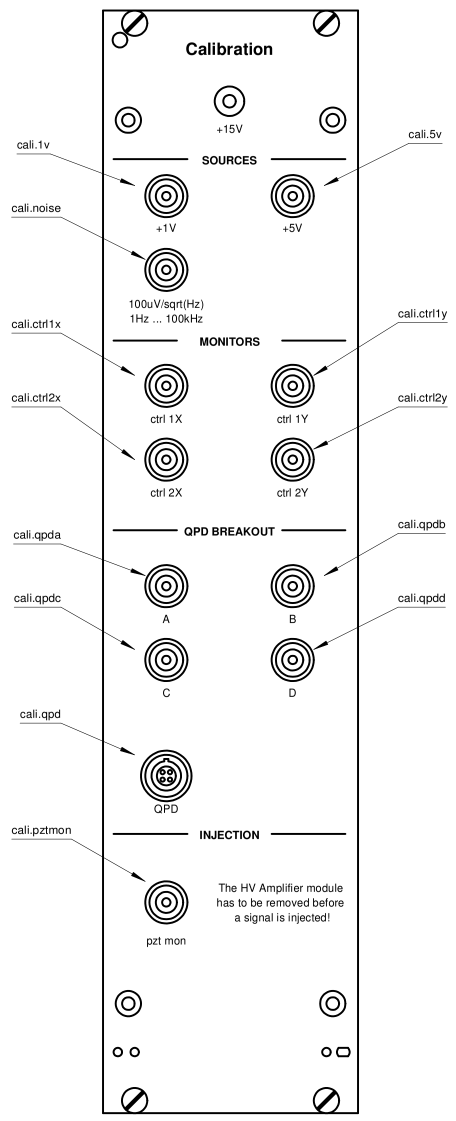
To calibrate the TPD photodiode signals, temporarily connect a voltage reference, e.g. 1 V, to the connectors *fbox.tpd.0db*, *fbox.tpd.40db*, and *fbox.tpd.80db*. Adjust the calibration factor (V/Cts) such that the right voltage is displayed.

The three filter modules `TPD_0DB` , `TPD_40DB` , and `TPD_80DB` should contain de-whitening filters of the field box and the anti aliasing filters. The calibration of these filter modules can be verified, e.g., by connecting a calibrated white noise source to the mentioned connectors and measuring the spectral densities of the signals `DBB-TPD_0DB` , `DBB-TPD_40DB` , and `DBB-TPD_80DB` . The photodiode internal signal conditioning, especially the 40 dB and 80 dB gain, must not be compensated.

To calibrate the RPD photodiode signals connect the voltage reference to the connectors *misc.rpd.dc.in* and *misc.rpd.ac.in*. Adjust the calibration factor (V/Cts) such that the right voltage is displayed.

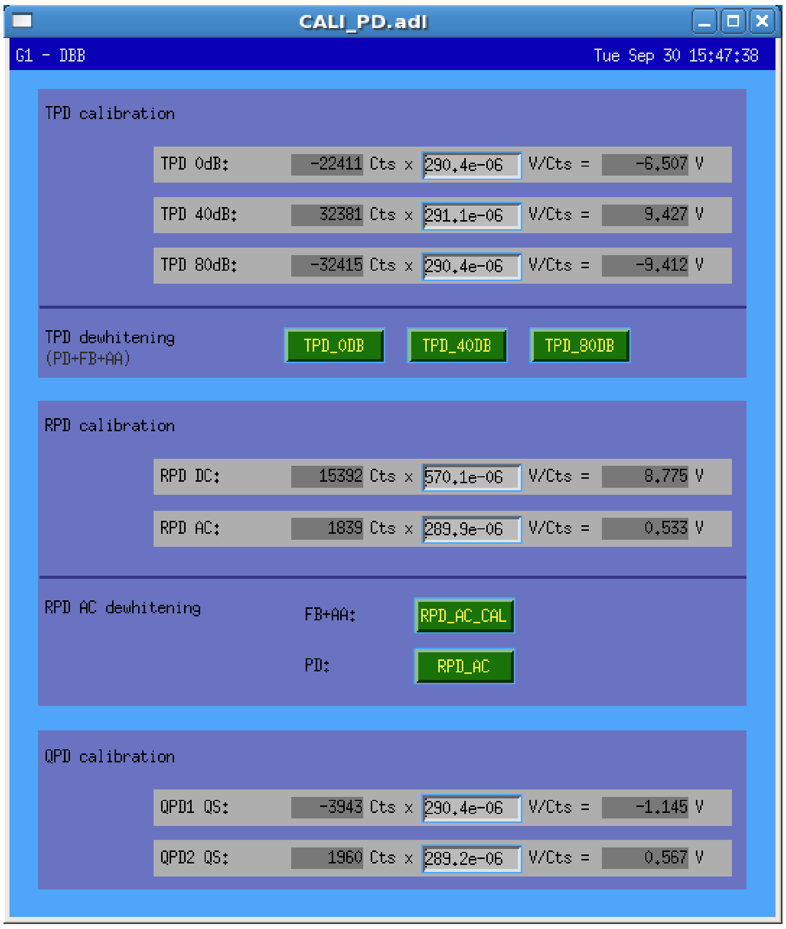
The filter module `RPD_AC_CAL` should contain the de-whitening filter for the field box and the anti aliasing filters. The calibration of this filter module can be verified by connecting a calibrated white noise source to the mentioned connectors and measuring the spectral density of the signal `DBB-RPD_AC_CAL` .

The filter module `RPD_AC` should contain the de-whitening filter of the photodiode internal signal conditioning for the ac signal path. The transfer function of the ac signal path is given as



Connector	Range	Type	Description
cali.1v	1 V	out	Voltage reference.
cali.5v	5 V	out	Voltage reference.
cali.noise		out	White noise source at a level of 100 $\mathrm{uVHz}^{-1/2}$.
cali.ctrl1x	-10 V. . .10 V	in	Monitor for CTRL1-X signal.
cali.ctrl1y	-10 V. . .10 V	in	Monitor for CTRL1-Y signal.
cali.ctrl2x	-10 V. . .10 V	in	Monitor for CTRL2-X signal.
cali.ctrl2y	-10 V. . .10 V	in	Monitor for CTRL2-Y signal.
cali.qpda		in/out	Signal for/of quadrant A.
cali.qpdb		in/out	Signal for/of quadrant B.
cali.qpdc		in/out	Signal for/of quadrant C.
cali.qpdd		in/out	Signal for/of quadrant D.
cali.qpd		in/out	High density connector.
cali.pztmon	-10 V. . .10 V	in	Connector to inject a signal to pztmon. HV Amplifier module has to be removed from the crate before!

Figure 25: Calibration module interface description.



TPD calibration

TPD 0dB:	-22411	Cts x	290.4e-06	V/Cts =	-6.507	V
TPD 40dB:	32381	Cts x	291.1e-06	V/Cts =	9.427	V
TPD 80dB:	-32415	Cts x	290.4e-06	V/Cts =	-9.412	V

TPD dewhitening
(PD+FB+AA)

RPD calibration

RPD DC:	15392	Cts x	570.1e-06	V/Cts =	8.775	V
RPD AC:	1839	Cts x	289.9e-06	V/Cts =	0.533	V

RPD AC dewhitening

FB+AA:

PD:

QPD calibration

QPD1 QS:	-3943	Cts x	290.4e-06	V/Cts =	-1.145	V
QPD2 QS:	1960	Cts x	289.2e-06	V/Cts =	0.567	V

Figure 26: Calibration screen for the photodiodes.

part of the DBB calibration data (`tf-rpd-ac.dat`). This transfer function has to be inverted for the de-whitening filter.

To calibrate the QPD photodiodes connect a voltage reference to the connectors *demod.qpd1.dc* and *demod.qpd2.dc* (using the QPD breakout in the calibration module). Adjust the calibration factor (V/Cts) such that the displayed voltage fits to the signal measured at *demod.qpd1.qs* and *demod.qpd2.qs*.

5.2. PZT calibration

This screen is used to enter the dc calibration for the PZT signals and the de-whitening of internal signal conditioning, field boxes and anti-aliasing filters (Fig. 27).

To calibrate the HV MON signal, measure the voltage at *hv.pmc.mon* and multiply this signal by 15 to get the real voltage at the PMC PZT. Adjust the calibration factor (V/Cts) to fit the voltage at the PMC PZT.

The calibration factor from voltage at the PMC PZT to FSR of the PMC can be calculated with the DBB calibration data (`pmc-cali.dat`).

The calibration of the manual control can be derived from the already calibrated HV MON signal. Adjust the calibration factor (Cts/FSR) such that the signal roughly fits to the HV MON signal (FSR).

To calibrate the PZT MON signal temporarily remove the HV Amplifier module and inject a voltage reference, e.g. 1 V, to the connector *fbox.pztmon*. Adjust the calibration factor (V/Cts) such that the right voltage is displayed.

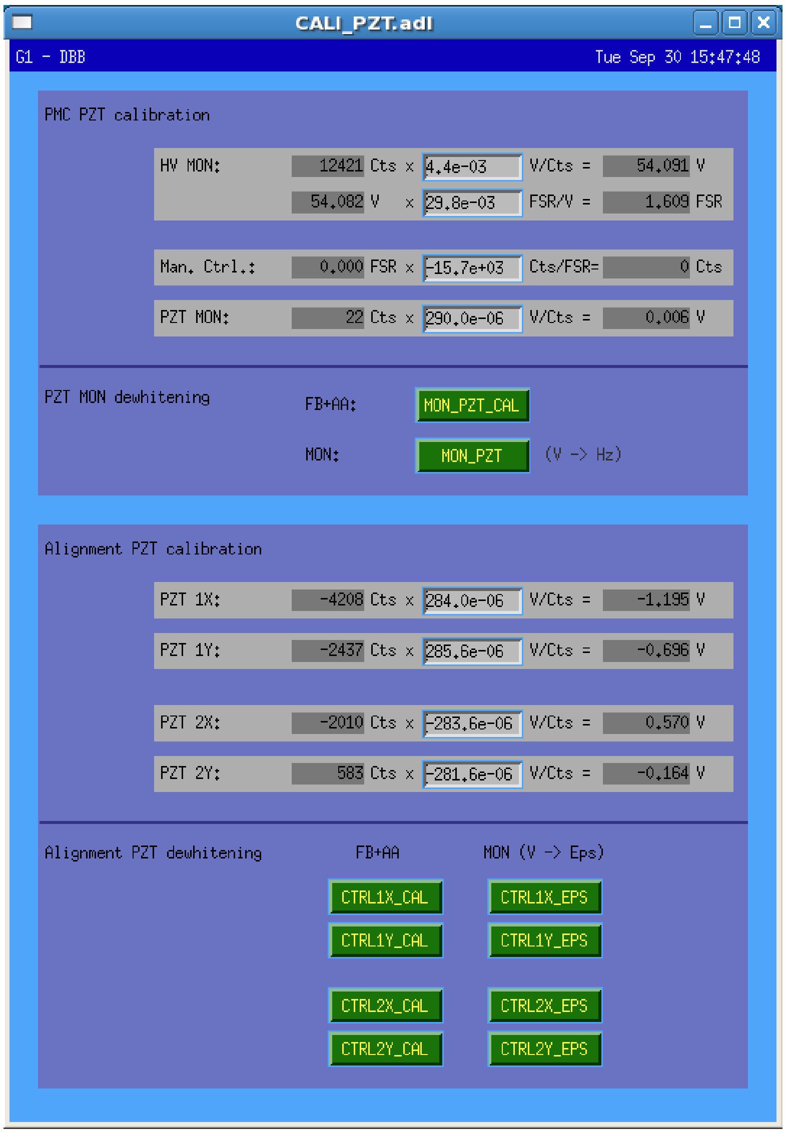
The filter module `MON_PZT_CAL` should contain the de-whitening filters for the field box and the anti aliasing filters. The calibration of this filter module can be verified by connecting a calibrated white noise source to the mentioned connector and measuring the spectral density of the signal `DBB-MON_PZT_CAL` .

The filter module `PZT_MON` contains the calibration of the PZT MON signal (Hz/V). The transfer function is given as part of the DBB calibration data (`tf-hv-pztmon.dat`).

The four alignment PZT are calibrated by adding some offset using the computer system and then measuring the actual output signal. In manual mode turn on the pre-alignment control loop and open the four filter modules `DBB-CTRL_QCTRL1X` , `DBB-CTRL_QCTRL1Y` , `DBB-CTRL_QCTRL2X` , and `DBB-CTRL_QCTRL2Y` . Disconnect the input signal in all four filter modules and add an offset to the input, e.g. 1000 Cts. Now measure the output voltage at the DBB calibration module and adjust the calibration factors such that the measured voltage matches the displayed one.

The filter modules `CTRL1X_CAL` , `CTRL1Y_CAL` , `CTRL2X_CAL` , and `CTRL2Y_CAL` should contain the de-whitening of the field box and the anti aliasing filters. This can be verified by injecting a white noise source using the computer system and measuring the output signals with the DBB calibration module.

The filter modules `CTRL1X_EPS` , `CTRL1Y_EPS` , `CTRL2X_EPS` , and `CTRL2Y_EPS` contain the calibration of the alignment PZTs. The filter modules have to contain the transfer functions given as part of the DBB calibration data (`tf-pnt1x.dat` , `tf-pnt1y.dat` , `tf-pnt2x.dat` , `tf-pnt2y.dat`).



```
G1 - DBB                                          Tue Sep 30 15:47:48

PMC PZT calibration

      HV MON:        12421 Cts x  4.4e-03     V/Cts =    54.091 V
                    54.082 V   x  29.8e-03    FSR/V =     1.609 FSR

      Man. Ctrl.:    0.000 FSR x  -15.7e+03   Cts/FSR=        0 Cts

      PZT MON:          22 Cts x  290.0e-06   V/Cts =     0.006 V

PZT MON dewhitening           FB+AA:   MON_PZT_CAL

                              MON:     MON_PZT       (V -> Hz)

Alignment PZT calibration

      PZT 1X:        -4208 Cts x  284.0e-06   V/Cts =    -1.195 V

      PZT 1Y:        -2437 Cts x  285.6e-06   V/Cts =    -0.696 V

      PZT 2X:        -2010 Cts x  -283.6e-06  V/Cts =     0.570 V

      PZT 2Y:          583 Cts x  -281.6e-06  V/Cts =    -0.164 V

Alignment PZT dewhitening       FB+AA        MON (V -> Eps)
```

Figure 27: Calibration screen for the PZTs.

5.3. Error signal calibration

The error signals for the PMC length control loop and the four auto alignment loops need to be calibrated right before a measurement of frequency or pointing noise. The calibration is automated. A calibration signal above the unity gain frequency of the control loops is injected to the PZT control signals and demodulated at the error signals. The calibrated PZT signals are used as reference to calibrate the error signals.

The duration and amplitude of the calibration signal can be adjusted in this screen (Fig. 28). The amplitude should be chosen larger than the measurement noise but should not saturate any signals. The calibration of the five error signals cannot be performed at the same time and must be started manually. The age of a calibration is recorded and is initialized to a very high value.

The filters `DS` , `1DX` , `1DY` , `2DX` , and `2DY` should contain the de-whitening filters for the field box and the anti aliasing filters. The calibration of these filter modules can be verified by connecting a broadband white noise source around 1 MHz to the connectors *demod.qpd1.ac* and *demod.qpd2.ac* and measuring the spectral density of the signals `DBB-QPD_DS` , `DBB-QPD_1DX` , `DBB-QPD_1DY` , `DBB-QPD_2DX` , `DBB-QPD_2DY` .

5.4. Miscellaneous calibration

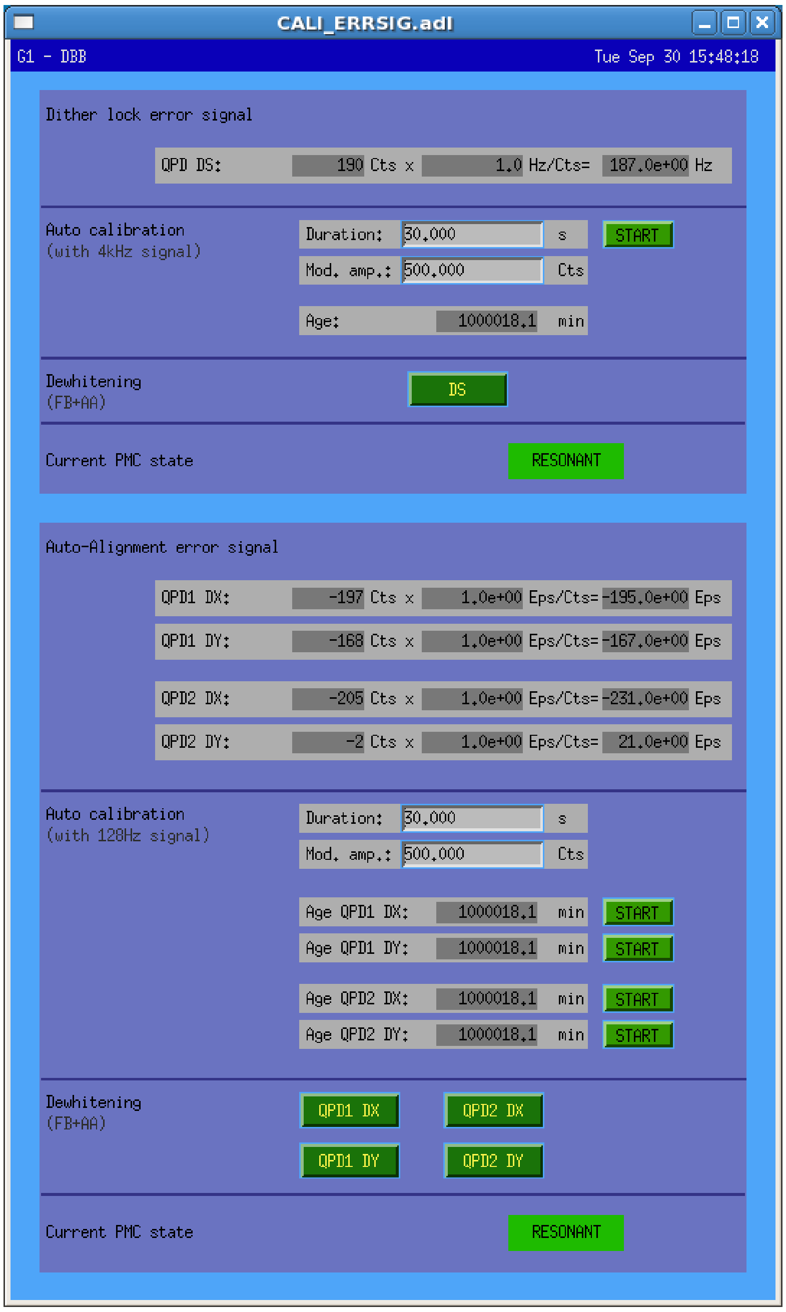
In this screen the calibration of the lens movement and the phase shifter can be adjusted (Fig. 29).

The lens position depends non-linear on the position signal. So the purpose of the calibration is to give a rough value how far a lens will be moved. The calibration needs to be determined by measuring the lens movement at the DBB manually.

The phase shifter depends as well non-linear on the phase shifting signal. So the purpose of the calibration is to give a rough orientation. The actual phase shift can be measured with a scope between the signals *demod.lomon* and *demod.modout*. Use this measurement to calibrate the signal.

5.5. Auxiliary calibration

The DBB field box has three auxiliary inputs. The calibration and de-whitening of these signals can be entered with this screen (Fig. 30).


CALI_ERRSIG.adl

G1 - DBB Tue Sep 30 15:48:18

Dither lock error signal

QPD DS: 190 Cts x 1.0 Hz/Cts= 187.0e+00 Hz

Auto calibration
(with 4kHz signal)

Duration: 30.000 s START
Mod. amp.: 500.000 Cts

Age: 1000018.1 min

Dewhitening
(FB+AA)

DS

Current PMC state RESONANT

Auto-Alignment error signal

QPD1 DX: -197 Cts x 1.0e+00 Eps/Cts= -195.0e+00 Eps
QPD1 DY: -168 Cts x 1.0e+00 Eps/Cts= -167.0e+00 Eps

QPD2 DX: -205 Cts x 1.0e+00 Eps/Cts= -231.0e+00 Eps
QPD2 DY: -2 Cts x 1.0e+00 Eps/Cts= 21.0e+00 Eps

Auto calibration
(with 128Hz signal)

Duration: 30.000 s
Mod. amp.: 500.000 Cts

Age QPD1 DX: 1000018.1 min START
Age QPD1 DY: 1000018.1 min START

Age QPD2 DX: 1000018.1 min START
Age QPD2 DY: 1000018.1 min START

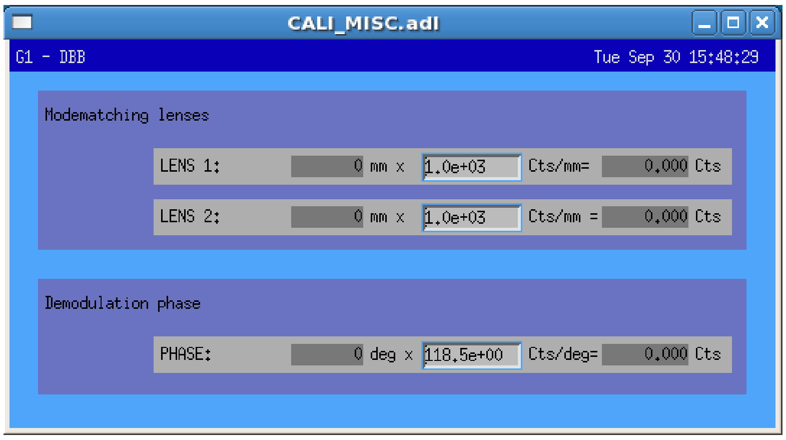
Dewhitening
(FB+AA)

QPD1 DX QPD2 DX

QPD1 DY QPD2 DY

Current PMC state RESONANT

Figure 28: Calibration screen for the error signals.


CALI_MISC.adl

G1 - DBB   Tue Sep 30 15:48:29

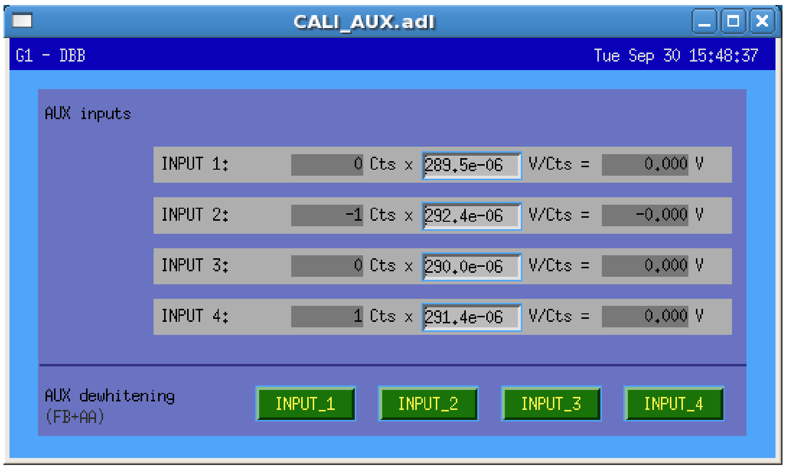
Modematching lenses

LENS 1:	0	mm x	1.0e+03	Cts/mm=	0.000	Cts
LENS 2:	0	mm x	1.0e+03	Cts/mm =	0.000	Cts

Demodulation phase

PHASE:	0	deg x	118.5e+00	Cts/deg=	0.000	Cts

Figure 29: Calibration screen for miscellaneous signals.


CALI_AUX.adl

G1 - DBB   Tue Sep 30 15:48:37

AUX inputs

INPUT 1:	0	Cts x	289.5e-06	V/Cts =	0.000	V
INPUT 2:	-1	Cts x	292.4e-06	V/Cts =	-0.000	V
INPUT 3:	0	Cts x	290.0e-06	V/Cts =	0.000	V
INPUT 4:	1	Cts x	291.4e-06	V/Cts =	0.000	V

AUX dewhitening (FB+AA)

Figure 30: Calibration screen for auxiliary signals.

Mode	Manual	Scan	Lock
Relative power noise (RPN)	•	•	•
Frequency noise (FRQ)			•
Pointing noise (PNT)			•
Beam quality (MSC)		•	

Table 4: Measurements dependent on operation mode.

6. Measurements

Dedicated screens are available to guide the several measurements that can be performed with the DBB. The screens are accessible through the menu of the main control screen. Detailed step by step instructions for performing the measurements in general are given in the instruction manual[3].

The screens are divided into three parts. The first part is a numbered check list indicating if all requirements for the measurement are fulfilled. Naturally only those requirements are listed which can be checked by the real time code.

The second part is a list of DAQ channels which contain the actual measurement data. The name of the corresponding filter modules are listed (the output of these filter modules is meant). The bandwidth and the unit of the channel are listed.

The third part are monitors corresponding to the measurement to give a rough view of the measurement data.

Depending on the current operation mode of the DBB several measurements can be performed in parallel (s. Tab. 4).

6.1. Relative power noise

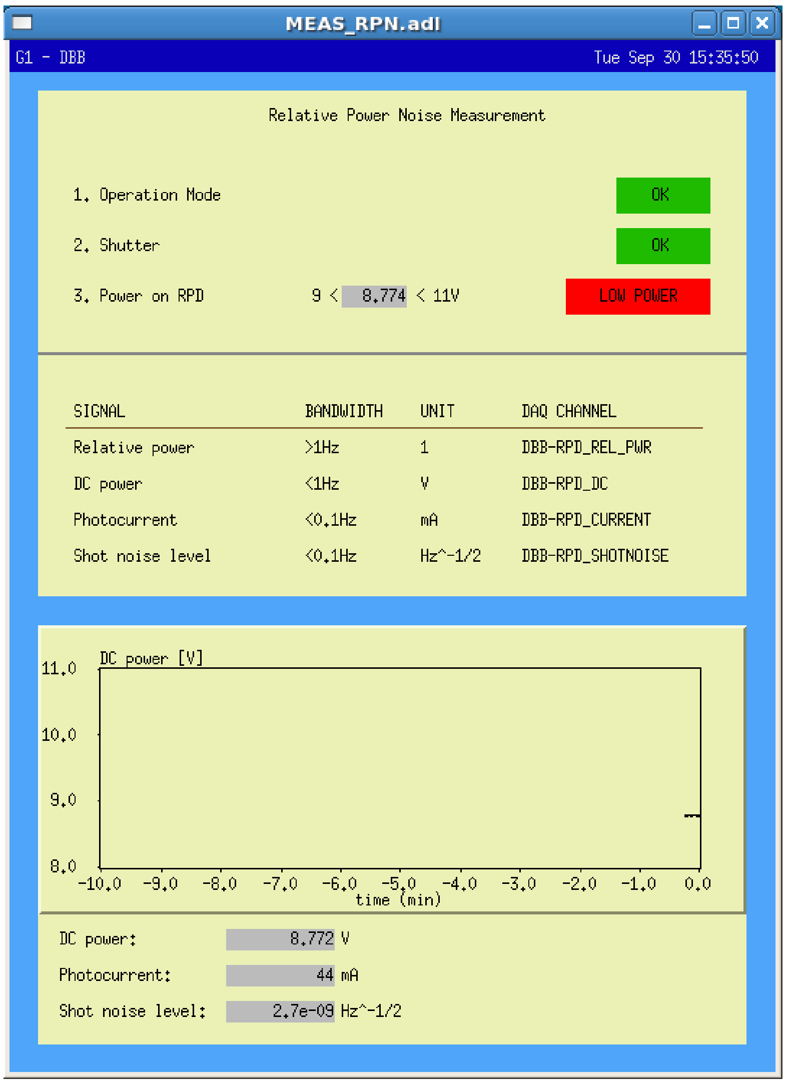
The real time code calibrates the photodiode signals and generates a time domain signal corresponding to the relative power (`DBB-RPD_REL_PWR`). This signal is already ac coupled at very low frequencies. This time signal should be used to calculate linear spectral densities of the relative power fluctuations.

For long term power fluctuations the dc signal of the photodiode should be used (`DBB-RPD_DC`).

Power noise measurements at radio frequencies cannot be performed with the computer control (see [3]).

6.2. Frequency noise

The error signal of the PMC length control loops need to calibrated before measuring the frequency noise. The real time code generates two calibrated time domain signals for the frequency noise: the error signal `DBB-QPD_DS_CAL` and the PZT signal `DBB-MON_PZT` . The linear spectral densities of these signals should cross at the unity gain frequency of the PMC length control loop around 1 kHz.



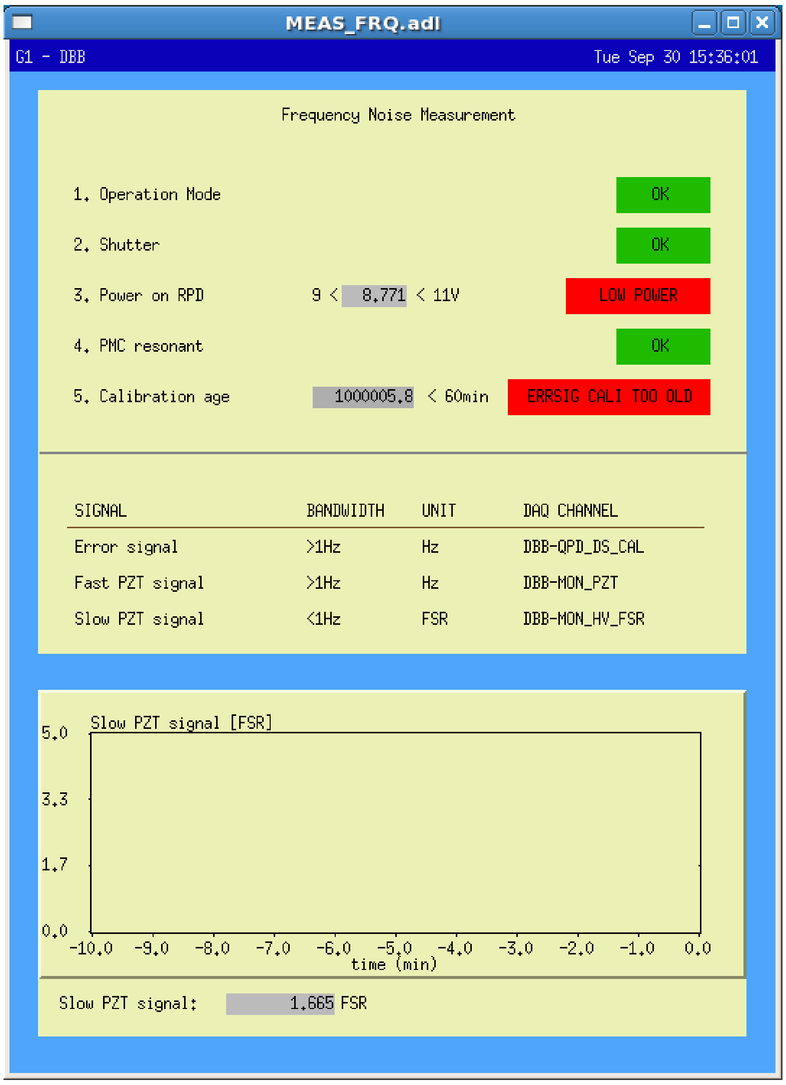
Relative Power Noise Measurement

1. Operation Mode		OK
2. Shutter		OK
3. Power on RPD	9 < 8.774 < 11V	LOW POWER

SIGNAL	BANDWIDTH	UNIT	DAQ CHANNEL
Relative power	>1Hz	1	DBB-RPD_REL_PWR
DC power	<1Hz	V	DBB-RPD_DC
Photocurrent	<0.1Hz	mA	DBB-RPD_CURRENT
Shot noise level	<0.1Hz	Hz^-1/2	DBB-RPD_SHOTNOISE

DC power:	8.772 V
Photocurrent:	44 mA
Shot noise level:	2.7e-09 Hz^-1/2

Figure 31: Checklist screen for relative power noise measurements.



MEAS_FRQ.adl

G1 - DBB   Tue Sep 30 15:36:01

Frequency Noise Measurement

1. Operation Mode		OK
2. Shutter		OK
3. Power on RPD	9 < 8.771 < 11V	LOW POWER
4. PMC resonant		OK
5. Calibration age	1000005.8 < 60min	ERRSIG CALI TOO OLD

SIGNAL	BANDWIDTH	UNIT	DAQ CHANNEL
Error signal	>1Hz	Hz	DBB-QPD_DS_CAL
Fast PZT signal	>1Hz	Hz	DBB-MON_PZT
Slow PZT signal	<1Hz	FSR	DBB-MON_HV_FSR

Slow PZT signal: 1.665 FSR

Figure 32: Checklist screen for frequency noise measurements.

The slow PZT signal can be measured as well but is probably dominated by environmental temperature fluctuations.

6.3. Pointing noise

The error signals of the alignment control loops need to calibrated before measuring the pointing noise. For each of the four degrees of freedom the real time code generates two calibrated time domain signals for the pointing noise: the error signal `DBB-QPD_?D?_CAL` and the PZT signal `DBB-CTRL_CTRL??_EPS`.

For measuring pointing noise above 1 Hz it is sufficient to evaluate the error signals since the alignment loops have a unity gain frequency of about 1 Hz. Accordingly it is often sufficient to evaluate the PZT signals for pointing fluctuations below 1 Hz

6.4. Beam quality

The beam quality is measured with the mode scan technique. The mode scans can be automatically analyzed by external programs (`modedetect.py` and `modeidentify.py`) using the transmitted power `DBB-TPD_VALUE` and the HV monitor signal `DBB-MON_HV`.

The two programs require an installation of python and scipy (tested with python 2.5.2 and scipy 0.6.0 on win32).

The input for the `modedetect.py` program is a text file with two columns. The first column must be the signal of the HV monitor signal and the second the transmitted power. Such an input file has to be generated, e.g. with ligoDV or DTT, in order to use this program. An example input data file might look like this:

```
57.9248 0.000771129
57.9205 0.000774619
57.9217 0.000775963
57.9195 0.000776704
...
```

The `modedetect.py` program will split the input file according to the ramp signals of the first column and will search for one FSR per ramp in the transmission signal. The scan of one FSR is then calibrated and higher order modes will be detected.

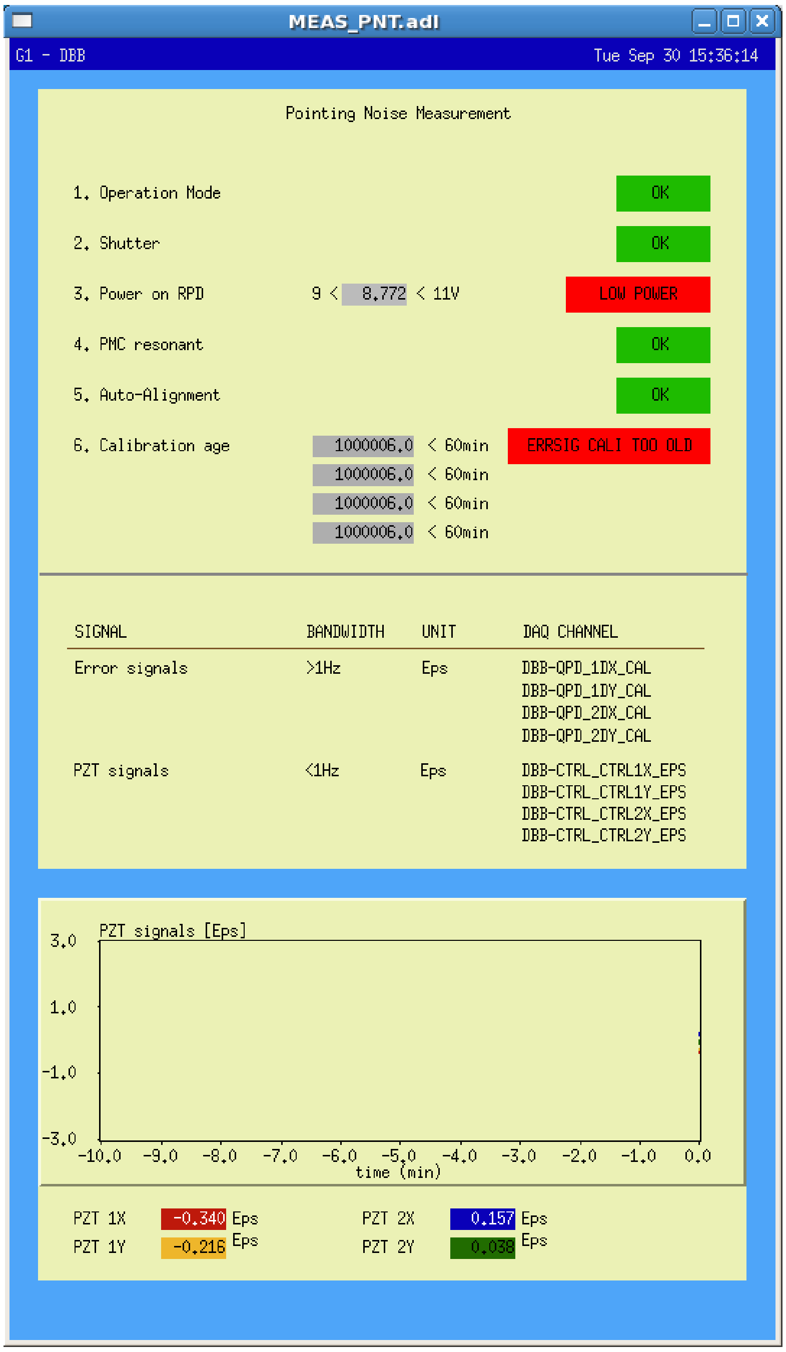
The output of the program is a list of detected modes and some additional information like the total power in higher order transversal modes. The higher modes are not identified by this program.

The program takes many optional arguments on the command line. A detailed list can be displayed with the following command:

```
$ python modedetect.py -?

usage: modedetect [options]
find mode peaks in a modescan by Patrick Kwee, (c) 2004, 2007, 2008
two column sgl or text data input expected at stdin with ramp and modescan.

  -w   write scan to file filename[,num of scan]
```



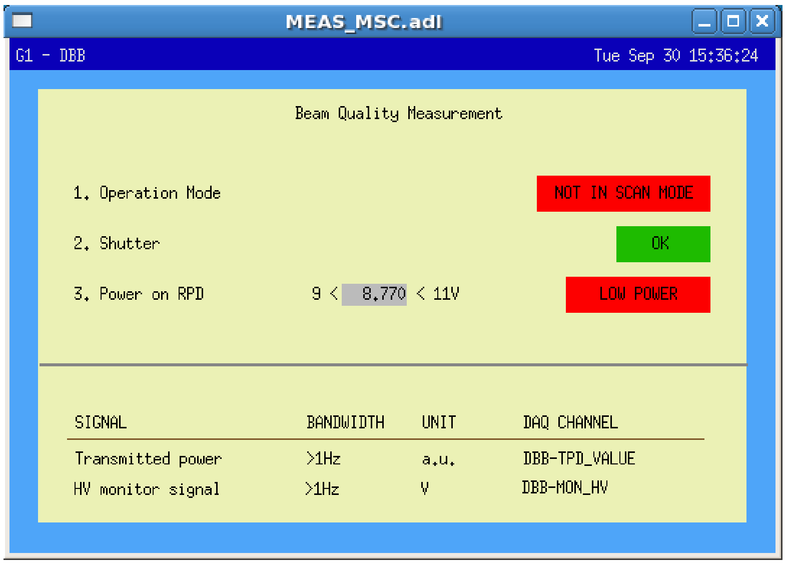
Pointing Noise Measurement

1. Operation Mode		OK
2. Shutter		OK
3. Power on RPD	9 < 8.772 < 11V	LOW POWER
4. PMC resonant		OK
5. Auto-Alignment		OK
6. Calibration age	1000006.0 < 60min	ERRSIG CALI TOO OLD
	1000006.0 < 60min	
	1000006.0 < 60min	
	1000006.0 < 60min	

SIGNAL	BANDWIDTH	UNIT	DAQ CHANNEL
Error signals	>1Hz	Eps	DBB-QPD_1DX_CAL DBB-QPD_1DY_CAL DBB-QPD_2DX_CAL DBB-QPD_2DY_CAL
PZT signals	<1Hz	Eps	DBB-CTRL_CTRL1X_EPS DBB-CTRL_CTRL1Y_EPS DBB-CTRL_CTRL2X_EPS DBB-CTRL_CTRL2Y_EPS

PZT 1X	-0.340	Eps	PZT 2X	0.157	Eps
PZT 1Y	-0.216	Eps	PZT 2Y	0.038	Eps

Figure 33: Checklist screen for pointing noise measurements.


MEAS_MSC.adl

G1 - DBB Tue Sep 30 15:36:24

Beam Quality Measurement

1. Operation Mode NOT IN SCAN MODE

2. Shutter OK

3. Power on RPD 9 < 8.770 < 11V LOW POWER

SIGNAL	BANDWIDTH	UNIT	DAQ CHANNEL
Transmitted power	>1Hz	a.u.	DBB-TPD_VALUE
HV monitor signal	>1Hz	V	DBB-MON_HV

Figure 34: Checklist screen for beam quality measurements.

```
  -u   format for written scan (default: nsua)
         r   raw ramp
         v   fitted ramp
         f   frequency
         n   normalized frequency
         m   raw scan
         s   normalized scan
         o   smoothed scan
         u   fundamental fit
         a   overall fit

  -s   number of ramp skips at front (default: 0)
  -n   number of ramps to analyse (default: -1)
  -v   increase verbosity
  -i   inspect range (default: 1e4)
  -r   reduce data by given factor (default: 1)
  -f   flip ramp and scan
  -c   pzt calibration file
  -q   pzt calibration function name (default: f)
  -d   up or down ramp (default: u)
  -o   filter cut off (default: 0.5)
  -t   how much percent to trim of the start/end of each ramp (default: 5)
  -x   interpret stdin data as text data and not as sgl data (default: sgl data)
```

To analyze a mode scan the following command can be used.

```
$ python modedetect.py -v -i 80e3 -c pmc-cali.py -x \
    -w scan.dat,2 < modescan.dat > ms.dat

--- Ramp: 1, Length: 81917 Samples ---
```

```
FSR: 59.0182V..87.3983V, 18171 Samples
FSR fit: Peak1 1.9064, Peak2 2.9031, Intensity 0.99, Finesse 349.70
Peaks found: 68
Higher mode power:   7.109%

--- Ramp: 2, Length: 81922 Samples ---
FSR: 58.6964V..87.0655V, 18164 Samples
FSR fit: Peak1 1.8954, Peak2 2.8914, Intensity 0.97, Finesse 357.62
Peaks found: 67
Higher mode power:   7.826%
Writing file 'scan.dat'

...

--- Summary ---
Number of Ramps: 7
Total Time: 45.2136s
Time per Ramp: 6.45909s
Ramp Length: 81919.4 +- 1.91663
Low PD signal: 0 +-0
High PD signal: 7.19888 +-0.0175301
FSR Length: 18169.3 +-3.73073 Samples
Calibration Deviation: -0.391669% +-0.0250295%
Finesse: 355.334 +-2.95274
Resolution: 51.1365 +-0.428772
Higher Mode Count: 67.2857 +-2.81396
Higher Mode Power: 7.41636 +-0.805611
```

Depending on the sample frequency and the ramp frequency the parameter `-i` has to be adjusted. The approximate number of samples per ramp must be given with the `-i` parameter. In this case a sampling frequency of 16 kHz and a ramp frequency of 0.1 Hz was used. Therefore one ramp consists of about $0.5 \cdot 16e3/0.1 = 80e3$ samples.

The calibration polynom for the PMC PZT has to be given with the `-c` argument.

The argument `-x` sets the input data format to text format. It is possible to use also a binary input format (see source code).

Individual scans can be saved to separate files with the `-w` switch. In this case the second scan is saved to the file `scan.dat`. The format of this file can be altered with the `-u` switch.

The input data is read from the text file `modescan.dat` and the results of the analysis are written to `ms.dat`. In this case the file `ms.dat` looks like:

```
# Ramp Number: 1
# Time: 3.21334s
# Ramp Length: 81917 Samples
# Ramp Signal: 11.033..138.959
# PD Signal: 0..7.17366
# FSR Length: 18171 Samples
# Resolution: 51.9612 Samples/linewidth
# Calibration Deviation: -0.33134%
# Fundamental Mode Ramp Position: 59.0182, 87.3983
# Finesse: 349.703
# Higher Mode Count: 68
# Higher Mode Power: 7.10944%
#
# [Normalized Frequency Position] [Relative Mode Power to Fundamental Mode] [Ramp Position]
7.469998e-003 7.729415e-003 5.923371e+001
1.859534e-002 2.298341e-004 5.955701e+001
2.902766e-002 1.549610e-004 5.986001e+001
3.645192e-002 3.566893e-004 6.007554e+001
...
```

The file `ms.dat` can now be used as input for the program `modeidentify.py`. This program tried to identify the higher transversal modes using their resonance frequency. The switches and arguments of the program can be displayed with:

```
$ python modeidentify.py -?

usage: modeidentify [options]
identify TEM modes in a modescan by Patrick Kwee, (c) 2004, 2007
output of modedetect is expected at stdin.

  -f   column of input data which contains normalized frequency (default: 0)
  -p   column of input data which contains relative power (default: 1)
  -g   roundtrip gouy phase in fractional FSR (default: 0.151)
  -o   start gouy phase optimization which start value
  -v   increase verbosity
```

The program is called e.g. in the following way:

```
$python modeidentify.py < ms.dat > mi.dat
```

The input data from the program `modedetect.py` is used as input and the results are saved in the file `mi.dat`. In this case the file `mi.dat` looks like:

```
# Ramp Number: 1
# Time: 3.21334s
# Ramp Length: 81917 Samples
# Ramp Signal: 11.033..138.959
# PD Signal: 0..7.17366
# FSR Length: 18171 Samples
# Resolution: 51.9612 Samples/linewidth
# Calibration Deviation: -0.33134%
# Fundamental Mode Ramp Position: 59.0182, 87.3983
# Finesse: 349.703
# Higher Mode Count: 68
# Higher Mode Power: 7.10944%
#
# Significant Modes: 19
# Highest significant mode: 59
# Average Deviation: 0.000883371
# Absolute average deviation: 0.00222695
# Gouy phase: 0.15101 +- 8.88561e-005
# Gouy phase optimization iterations: 1
```

Channel	Datarate	Unit
G1:DBB-MON_HV_FSR_OUT	1024 Hz	FSR
G1:DBB-MON_HV_OUT	65536 Hz	V
G1:DBB-MON_PZT_OUT	16384 Hz	Hz
G1:DBB-QPD_1DX_CAL_OUT	16384 Hz	eps
G1:DBB-QPD_1DY_CAL_OUT	16384 Hz	eps
G1:DBB-QPD_2DX_CAL_OUT	16384 Hz	eps
G1:DBB-QPD_2DY_CAL_OUT	16384 Hz	eps
G1:DBB-QPD_DS_CAL_OUT	16384 Hz	Hz
G1:DBB-CTRL_CTRL1X_EPS_OUT	16384 Hz	eps
G1:DBB-CTRL_CTRL1Y_EPS_OUT	16384 Hz	eps
G1:DBB-CTRL_CTRL2X_EPS_OUT	16384 Hz	eps
G1:DBB-CTRL_CTRL2Y_EPS_OUT	16384 Hz	eps
G1:DBB-RPD_CURRENT_OUT	1024 Hz	mA
G1:DBB-RPD_DC_OUT	1024 Hz	V
G1:DBB-RPD_REL_PWR_OUT	16384 Hz	1
G1:DBB-RPD_SHOTNOISE_OUT	1024 Hz	$\mathrm{Hz}^{-1/2}$
G1:DBB-TPD_VALUE_OUT	65536 Hz	V

Table 5: DAQ channel suggestion.

```
# Relative horizontal (X) misalignment: 0.0373149
# Relative vertical (Y) misalignment: 0.05153
# Relative mismodematching: 0.136361
#
# ... [Mode Order] [Odd l] [Deviation] [Significant]
#
# [Normalized Frequency Position] [Relative Mode Power to Fundamental Mode] [Ramp Position]
7.469998e-003 7.729415e-003 5.923371e+001  10 1 2.633574e-003 0
1.859534e-002 2.298341e-004 5.955701e+001  20 0 1.611804e-003 0
2.902766e-002 1.549610e-004 5.986001e+001  30 1 1.283056e-003 1
3.645192e-002 3.566893e-004 6.007554e+001  40 0 3.962367e-003 0
...
```

6.5. Data acquisition

A suggestion for the DAQ channels and the sampling rate which should be recorded by the frame builder is given in Tab. 5. The floating point values of the signals should not be interpreted as ’Counts’ but as the physical units given in the table.

A. Source code

```
void CTRL_DESELECT(double *in, int inSize, double *out, int outSize){
    double signal = in[0];
    int channel = in[1];
    int i;

    if ((channel<0) || (channel>=outSize)) channel = 0;

    for (i=0; i<outSize; i++) out[i] = 0;
    out[channel] = signal;
}


void CTRL_SELECT(double *in, int inSize, double *out, int outSize){
    int channel = in[0];
    if ((channel<0) || (channel>=inSize-1)) channel = -1;
    out[0] = in[1+channel];
}


void INTERLOCK_FLIPFLOP(double *in, int inSize, double *out, int outSize){
  int sw_set = in[0];
  int hw_set = in[1];
  int reset  = in[2];

  static int sw_state = 0;
  static int hw_state = 0;

  /* SW flip flop */
  if (reset) sw_state = 0;
  if (sw_set) sw_state = 1;

  /* HW flip flop */
  if (reset) hw_state = 0;
  if (hw_set) hw_state = 1;


  out[0] = sw_state;
  out[1] = hw_state;
  out[2] = sw_state || hw_state;
}


#define HW_DETECT_DELAY 0.999978847
/* for 32768 delayed samples, 0.5s */
/* HW_DETECT_DELAY = 10**(log(0.5) / number of delayed samples) */

void INTERLOCK_HW_DETECT(double *in, int inSize, double *out, int outSize){
  /* Detects a hardware interlock */

  int localmode = in[0];
  int shutter_closed = in[1];
  int shutter_open   = in[2];

  int newstate = !localmode && shutter_open && shutter_closed;
  static double state = 0.0;

  state = HW_DETECT_DELAY*state + (1-HW_DETECT_DELAY)*newstate;
  out[0] = state >= 0.5;
}


void QPD_AUTOCALI(double *in, int inSize, double *out, int outSize){

  /* 4kHz Modulation */
  static double modulation[16] = {0.0, 0.38268343236508978,
  0.70710678118654746, 0.92387953251128674, 1.0, 0.92387953251128674,
  0.70710678118654757, 0.38268343236508989, 1.2246467991473532e-16,
  -0.38268343236508967, -0.70710678118654746, -0.92387953251128652,
  -1.0, -0.92387953251128663, -0.70710678118654768, -0.38268343236509039};


  double integration_time = in[0];
  double trigger = in[1];
  double amplitude = in[2];

  double sig_ref = in[3];
  double sig_cali = in[4];

  static int countdown = 0;
  static double age = 1e6;

  static double sum_ref_p = 0.0;
  static double sum_ref_q = 0.0;

  static double sum_cali_p = 0.0;
  static double sum_cali_q = 0.0;

  static double cali_factor = 1.0;


  double norm, amp_ref, amp_cali;


  if (countdown > 0) {
      sum_ref_p += sig_ref * modulation[countdown % 16];
      sum_ref_q += sig_ref * modulation[(countdown+4) % 16];

      sum_cali_p += sig_cali * modulation[countdown % 16];
      sum_cali_q += sig_cali * modulation[(countdown+4) % 16];

      /* output modulation */
      out[0] = modulation[countdown % 16] * amplitude;
      countdown -= 1;

      if (countdown == 0) {
         norm = integration_time * FE_RATE;
         amp_ref = lsqrt( (sum_ref_p*sum_ref_p + sum_ref_q*sum_ref_q) /
            (norm*norm));
         amp_cali = lsqrt( (sum_cali_p*sum_cali_p + sum_cali_q*sum_cali_q) /
            (norm*norm));

         if (amp_cali < 1e-6) {
            cali_factor = 1.0;
         } else {
            cali_factor = amp_ref/amp_cali;
         }

         age = 0;
      }

      out[2] = (double)countdown / FE_RATE;

   } else {

      if (trigger>0) {
         countdown = integration_time * FE_RATE;
         sum_ref_p = 0.0;
         sum_ref_q = 0.0;
         sum_cali_p = 0.0;
         sum_cali_q = 0.0;
      }

      out[0] = 0.0;
      out[2] = 0.0;
   }

   out[1] = cali_factor;

   age += 1. / FE_RATE / 60;
   out[3] = age;
}
```

```
void QPD_MULTICALI(double *in, int inSize, double *out, int outSize){

  /* 128Hz Modulation */
  static double modulation[512] = {0.0, 0.012271538285719925, ...
  , -0.012271538285720572};


  /* input */

  double integration_time = in[0];
  double trigger = in[1];
  double amplitude = in[2];

  double sig_ref = in[3];
  double sig_cali;  /* in[4]...in[8] */

  /* output

    out[0] = modulation
    out[1] = channel num
    out[2] = countdown
    out[3..10] = factor, age

  */

  static int countdown = 0;
  static int channel = 0;

  static double sum_ref_p = 0.0;
  static double sum_ref_q = 0.0;

  static double sum_cali_p = 0.0;
  static double sum_cali_q = 0.0;

  static double age[4] = {1e6, 1e6, 1e6, 1e6};
  static double cali_factor[4] = {1.0, 1.0, 1.0, 1.0};


  double norm, amp_ref, amp_cali;
  int i;


  if (countdown > 0) {

    sum_ref_p += sig_ref * modulation[countdown % 512];
    sum_ref_q += sig_ref * modulation[(countdown+256) % 512];

    sig_cali = in[4+channel];
    sum_cali_p += sig_cali * modulation[countdown % 512];
    sum_cali_q += sig_cali * modulation[(countdown+256) % 512];

    /* output modulation */
    out[0] = modulation[countdown % 512] * amplitude;
    countdown -= 1;

    if (countdown == 0) {
      norm = integration_time * FE_RATE;
      amp_ref = lsqrt( (sum_ref_p*sum_ref_p + sum_ref_q*sum_ref_q) /
        (norm*norm));
      amp_cali = lsqrt( (sum_cali_p*sum_cali_p + sum_cali_q*sum_cali_q) /
        (norm*norm));

      if (amp_cali < 1e-6) {
        cali_factor[channel] = 1.0;
      } else {
        cali_factor[channel] = amp_ref/amp_cali;
      }

      age[channel] = 0;
    }

    out[2] = (double)countdown / FE_RATE;

  } else {

    if (trigger>0) {
      countdown = integration_time * FE_RATE;
      sum_ref_p = 0.0;
      sum_ref_q = 0.0;
      sum_cali_p = 0.0;
      sum_cali_q = 0.0;

      channel = trigger - 1;
      if ((channel<0) || (channel>3)) countdown = 0;
    }

    out[0] = 0.0;
    out[2] = 0.0;
  }

  out[1] = channel;

  for (i = 0; i<4; i++) {
    out[3+2*i] = cali_factor[i];

    age[i] += 1. / FE_RATE / 60;
    out[3+2*i+1] = age[i];
  }
```

```
}


void RAMP_RAMPGEN(double *in, int inSize, double *out, int outSize){
    //in[0]=frequency [Hz], ==0 means pause

    double dy;
    static int sign = 1;
    static double value = 0;

    dy=4*in[0]/FE_RATE;

    value += dy*sign;
    if (value>1) { sign=-sign; value = 1; }
    if (value<-1) { sign=-sign; value = -1; }

    out[0] = value;
}


void RPD_CALC_SHOTNOISE(double *in, int inSize, double *out, int outSize){
    double current = in[0];
    if (current<1e-6) current = 1;
    out[0] = 1sqrt(2*1.602e-19*1e3/current);
}


void TPD_SELECT(double *in, int inSize, double *out, int outSize){

  double in0db = in[0] < 0.0 ? - in[0] : in[0];
  double in40db = in[1] < 0.0 ? - in[1] : in[1];
  double in80db = in[2] < 0.0 ? - in[2] : in[2];
  double thres = in[3];

  if (in80db<thres) {
    out[0] = in80db/10000;
  } else if (in40db<thres) {
    out[0] = in40db/100;
  } else {
    out[0] = in0db;
  }

}
```

References

[1] Rolf Bork. AdvLigo CDS Realtime Code Generator (RCG) Application Developer's Guide. *Internal report*, 2008.

[2] Kenneth Evans. MEDM Reference Manual. 2006.

[3] Patrick Kwee. Diagnostic Breadboard Instruction Manual. *Internal report*, 2008.

[4] Patrick Kwee, Frank Seifert, Benno Willke, and Karsten Danzmann. Laser beam quality and pointing measurement with an optical resonator. *Rev. Sci. Instrum.*, 78:073103, 2007.

[5] Patrick Kwee and Benno Willke. Automatic laser beam characterization of monolithic Nd:YAG non-planar ring lasers. *Applied Optics*, ??:??, 2008.