

The Forbes Filterbank Manual

David Forbes
16-December-2005

1. Introduction.....	3
2. Specifications.....	4
3. System Topology.....	5
3.1. IF Processors.....	5
3.2. Down-Converter Chain.....	6
3.3. Channel Filters.....	8
3.4. System Timing.....	8
4. System Components.....	9
4.1 IF processors.....	9
4.2 Filter Crate.....	11
4.3 Digital Interface.....	21
5. Filter Alignment.....	25
6. Installation.....	28
6.1. Cabling.....	28
7. Operation.....	31
7.1. Startup sequence.....	31
7.2. Configuration.....	31
7.3. Calibration.....	32
7.4. Taking Data.....	32
7.5. Shutdown sequence.....	32
8. Maintenance.....	34
8.1. Scheduled maintenance.....	34
8.2. Unscheduled Maintenance.....	34
9. Performance.....	37
9.1 Overview.....	37
9.2. Stability.....	37
10. Hardware info (pinouts, backplane signal names).....	39
10.1. Converter card keying.....	39
10.2. Filter crate backplane.....	41
10.3. Filter cards.....	42
10.4. PC interface parallel port cable.....	43
10.5. Video cable.....	43
10.6. Digital crate backplane.....	45
11. Driver Software.....	46
11.1. Parallel port.....	46
11.2. EPP card.....	48
11.3. Int card.....	50
11.4. Timing card.....	55
11.5. Filter crate control.....	57
11.6. Converter cards.....	60
11.7. IFctl cards.....	63
11.8. Taking Data.....	67
12. Application Software.....	68
12.1. Introduction.....	68
12.2. Driver.....	69
12.3. Data structure.....	70
12.4. fbcal_utils.....	72
12.5. ffbllib.....	76
12.6. Configuration.....	81
12.7. Blanking Considerations.....	82
12.8. Initialization.....	83

1. Introduction

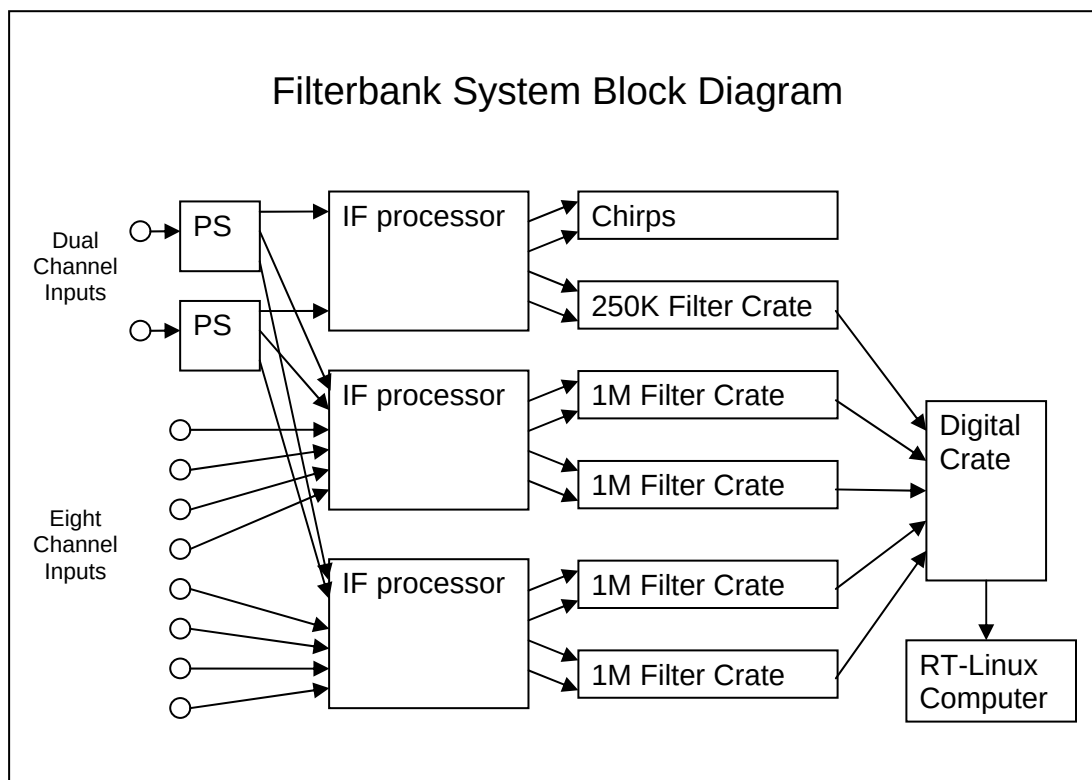
This document describes the filter bank system for the Heinrich Hertz Submillimeter Telescope. This instrument provides the flexibility and performance to match the new 7 channel DesertStar receiver system. The design philosophy is to maximize performance, convenience of operation and ease of repair while minimizing cost.

The first part of the manual presents a detailed description of the filterbank design philosophy and hardware implementation. Each module is discussed, along with design decisions and the reasoning behind each decision. The next part of the manual covers the installation, operation and maintenance of the filterbank system. The last part of the manual covers the software, both at the driver level and the application level.

Included in this manual are block diagrams of the entire system, each subsystem and each card. Schematic diagrams, component data sheets and circuit board layouts are contained in separate binders.

Features

- * 2048 channels of 1 MHz filters in eight steerable sections
- * 512 channels of 250 KHz filters in two steerable sections
- * Flexible input steering - 1, 2 or 8 inputs
- * High performance - 40dB image rejection within the 2GHz band
- * Modular construction
- * Compact size - 2560 channels in three racks
- * Easy maintenance - all circuitry on operator-accessible plug-in cards
- * Computer-controlled gain/zero calibration eliminates manual adjustment



2. Specifications

Input frequency range: 4.0 - 6.0 GHz
Input signal level: -20 dBm/GHz +/-10 dB
IF input ports: One 2GHz, two 1GHz or eight 256 MHz
Physical layout: Eurocard crates with plug-in units and cards
Conversion type: Four downconversions with high-side injection
Image rejection: 40dB minimum

One MHz filters
Number of channels: 2048
Channel spacing: 1 MHz
Filter bandwidth: 1.0 MHz at 3dB down

250 KHz filters
Number of channels: 512
Channel spacing: 250 KHz
Filter bandwidth: 0.25 MHz at 3dB down

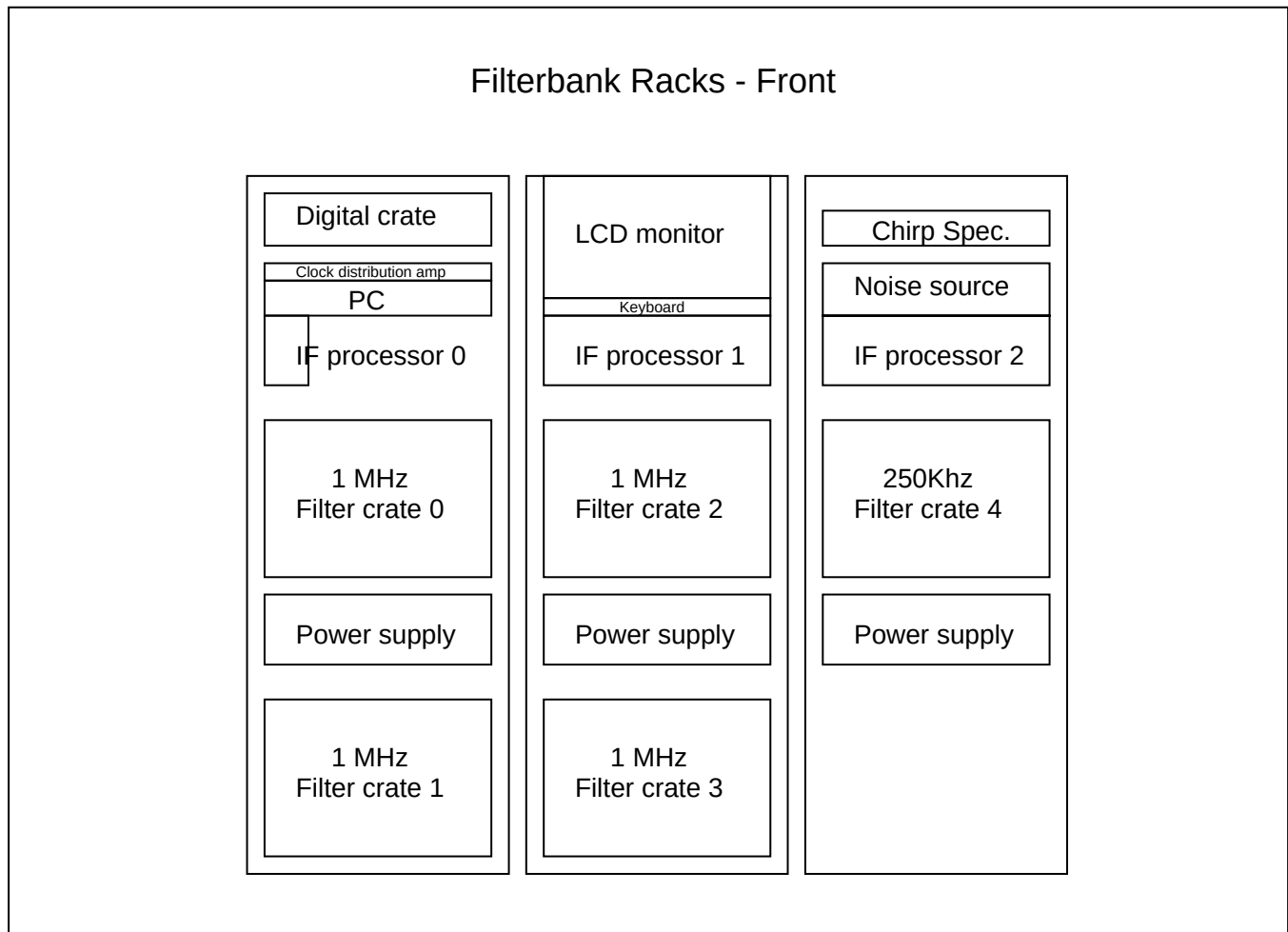
Channel filters: 2 section LC, tunable
Filter response: Butterworth
Power detector: Square law using AD835 multiplier IC

Analog integrator: 25 millisecond integration time
Readout scan downtime: 615 microseconds per scan
A/D Converters: One 14-bit converter per 512 channels
A/D Conversion rate: 1.0 microsecond per channel
Digital integrator: 24 bits wide, adds 2 to 1024 frames of 14 bit data
Gain/offset correction: Part of A/D converter circuit, computer-controlled

PC Interface: Parallel port in EPP mode
PC data rate: 0.6 MByte/sec; up to 40 2560-chan datasets/second

3. System Topology

The system consists of three IF processors, five filter crates and a digital interface to a PC. These are described in more detail below. A 4.5-5.5 GHz test signal generator and a 10 MHz clock distribution amp are also included in the system.



3.1. IF Processors

The complete filterbank system has three IF processors. Each of these converts four IF inputs at 5 GHz to four outputs at 2 GHz.

Two of the IF processors are assigned to the eight 1MHz filterbanks. The third IF processor is assigned to the two 250KHz filterbanks and the two chirp transform spectrometers (CTS).

The two 1MHz filterbank IF processors have the following routing options for the eight 256-channel filter banks that they feed:

- * One 512 MHz wide input from DesertStar pixel 1 and six 256 MHz wide inputs from the other 6 DesertStar pixels

- * Two parallel 1 GHz wide inputs from the JT or SIS receiver IF channels A and B
- * One 2GHz input from the JT receiver IF channels A or B (under development)

The third IF processor has the following options:

Filter A,B to IF A or B in series 128 MHz wide or to IF A and B in parallel 64 MHz

CTS A,B to IF A or B in series 350 MHz wide or to IF A and B in parallel 200 MHz

An input termination switch on each IF processor box allows zero level calibration. A comb/noise generator input is also provided. The comb/noise generator is a separate unit with three outputs centered at 5 GHz.

Each of the four converters in each IF processor may be individually steered to any frequency range within the 4.0-6.0 GHz band. The frequency granularity is 1.0 MHz.

3.2. Down-Converter Chain

Since the input bandwidth is so wide, the first downconversion must be to about 2 GHz to keep the LOs out of the passband and to permit easy image filtering. The frequency range chosen for the first converter output is 1800-2056 MHz.

The next task is to convert and split this 2GHz band down to the filter card input requirement of 20-36 MHz. This requires three more downconversions to meet the above image rejection requirements.

The four downconversion frequencies for the 1 MHz filters are:

3976-6024 MHz to 1800-2056 MHz
to 422-550 MHz
to 110-174 MHz
to 20-36 MHz

The four downconversion frequencies for the 250 kHz filters are:

4936-5064 MHz to 1896-1960 MHz
to 506-538 MHz
to 90-106 MHz
to 5-9 MHz

The LO frequencies have been chosen so that as few IF or LO frequencies appear in the image frequency bands of the converters as possible. The only LOs to potentially cause trouble are the two high fourth converter values. The physical arrangement of the filter cards has been optimized to keep unwanted LO signals distant from the cards that could receive them.

The second harmonic of the second LO falls within the range of the input signals. The system design is such that this LO signal has several levels of isolation from the input signals.

A detailed chart of filter conversion frequencies is given below:

2048 Channel 1MHz Filterbank										Units: MHz
	Input		LO		IF		Image		Step	
First	3976	4232	6032	1800	2056	7832	8088	256		
Conv	4232	4488	6288	1800	2056	8088	8344	256		
	4488	4744	6544	1800	2056	8344	8600	256		
	4744	5000	6800	1800	2056	8600	8856	256		
	5000	5256	7056	1800	2056	8856	9112	256		
	5256	5512	7312	1800	2056	9112	9368	256		
	5512	5768	7568	1800	2056	9368	9624	256		
	5768	6024	7824	1800	2056	9624	9880	256		
Second	1800	1928	2350	422	550	2772	2900	128		
Conv	1928	2056	2478	422	550	2900	3028	128		
Third	422	486	596	110	174	706	770	64		
Conv	486	550	660	110	174	770	834	64		
Fourth	110	126	146	20	36	166	182	16		
Conv	126	142	162	20	36	182	198	16		
	142	158	178	20	36	198	214	16		
	158	174	194	20	36	214	230	16		

256 Channel 250KHz Filterbank										Units: MHz
	Input		LO		IF		Image		Step	
First	4936	5000	6896	1896	1960	8792	8856	64		
Conv	5000	5064	6960	1896	1960	8856	8920	64		
Second	1896	1928	2434	506	538	2940	2972	32		
Conv	1928	1960	2466	506	538	2972	3004	32		
Third	506	522	612	90	106	702	718	16		
Conv	522	538	628	90	106	718	734	16		
Fourth	90	94	99	5	9	104	108	4		
Conv	94	98	103	5	9	108	112	4		
	98	102	107	5	9	112	116	4		
	102	106	111	5	9	116	120	4		

The second and third conversions each reduce the bandwidth by a factor of two. Thus, each stage of the conversion requires twice the number of converters for the same total channel count. This arrangement was chosen to minimize the number of different frequencies of filters and LOs needed. The result is that only two types of converter and filter cards are used.

The input frequency range for each of the second through fourth converters is selected by controlling the LO frequency based on the card's slot number. This works since the output frequency range is the same for both cards.

The image filters are of the elliptic bandpass type. The first and second converter filters are standard SMA types. The last two converter filters are elliptic lumped LC filters in surface-mount packages, made by KR Electronics.

3.3. Channel Filters

The channel filters are contained on filter cards. Each filter card has two separate 16-channel filter groups. The filters are arranged this way to allow higher packing density; the two filter groups are fed from different IF inputs but are at the same relative IF frequency.

The channel filters are Butterworth bandpass type made of two capacitively-coupled resonators. See the filter card description below for more information.

3.4. System Timing

The entire system is driven at a 25.615 millisecond frame rate. The frame time is divided in to two portions: integration and readout.

The integration time is 25.000 milliseconds and is the 'quiet' time of the system. The digital crate doesn't generate any signal transitions during this time.

The readout time is when the digital crate scans the integrators of the filter crates, but the integrators are placed into hold mode (disconnected from the square-law detectors) so that any digital noise entering the RF chain will not be integrated. The readout time is 513 microseconds - one microsecond per channel plus one microsecond of pipeline delay.

The integration capacitors are discharged (dumped) after readout. This time is currently set to 100 microseconds. It may be possible to shorten this time if needed.

All of the above times are programmable with the exception of the readout rate, which is fixed in the controller hardware at 1.0 microsecond per channel.

The telescope Blanking signal is used to interrupt the integration and read out the analog integration if the observing phase ends during the 25 millisecond integration period. The telescope Sync signal controls the digital integration of the resulting data into the proper bin: on-source or reference.

Furthermore, the Local Blanking inputs to the several Int cards are used to disable digital integration of the data being collected during a frame. These signals are typically driven by the various receiver lock signals from the receivers. Data integrity is achieved by stopping the integration as soon as the error condition is reported.

4. System Components

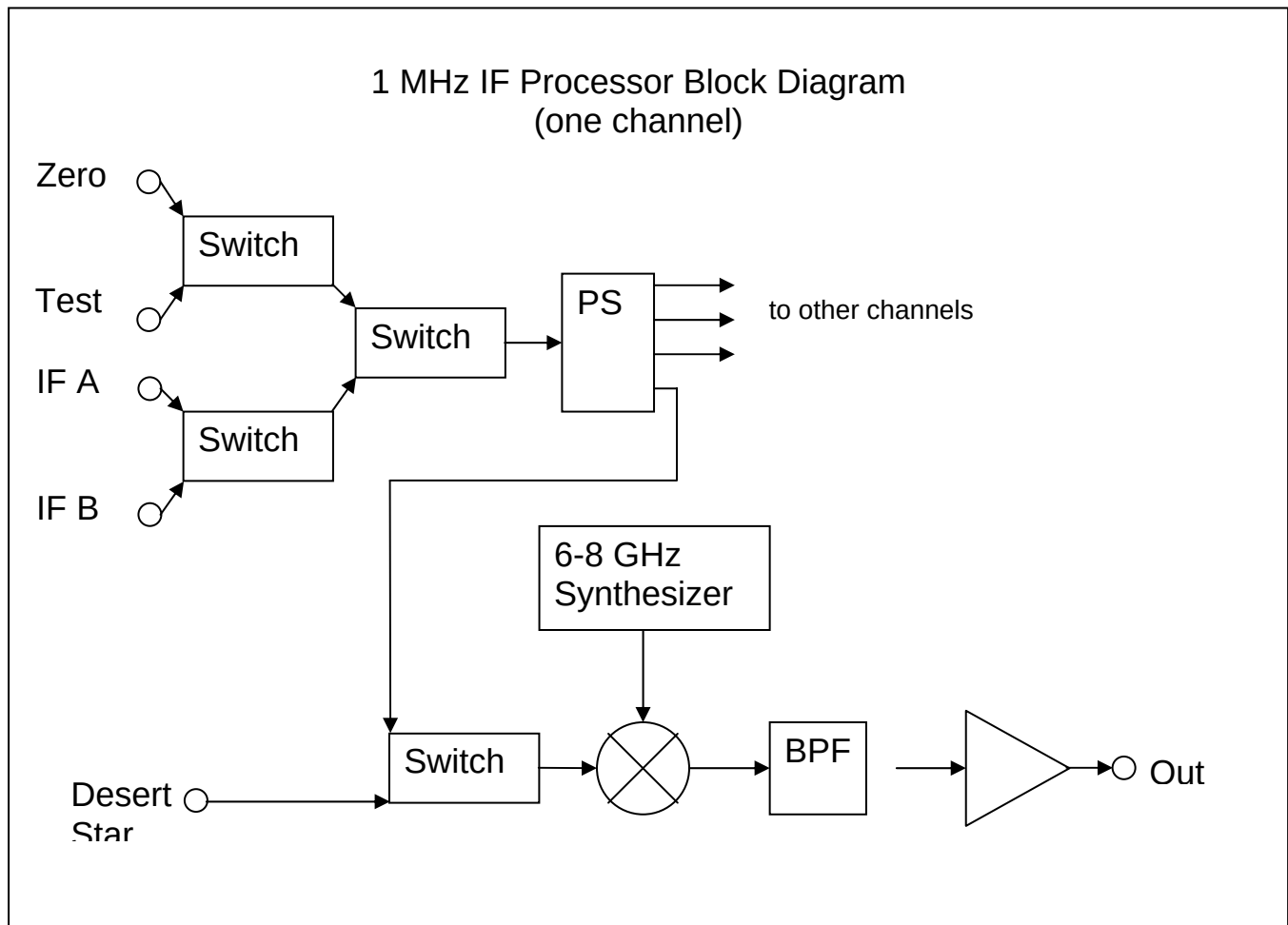
The system is comprised of three major sections:

- * IF Processors
- * Filter Crates
- * Digital Interface

These are described in detail below.

4.1 IF processors

Each of the three IF processors divide the incoming 4-6 GHz IF signals into four 256-MHz segments. These segments are fed to two filter crates.



The IF processors are built of SMA connectorized microwave components mounted in an aluminum box and are of conventional design. A computer interface is used to control the LO synthesizers, relays and LEDs. There are no front-panel controls other than a power switch. Front-panel LEDs show power supply, LO lock and signal routing status.

The IF processor inputs consist of the IF A and IF B signals from the JT receiver, plus four inputs from the DesertStar seven-beam receiver (only on the 1 MHz processors). A mode switch selects between DesertStar and JT. A second switch set selects between IF A, IF B, noise/comb or no signal (50 ohm termination). The 250K processor allows each section to select A or B; the 1MHz processor only allows per-box selection.

The four first converters are located in the IF processor. Each one converts a 256 MHz wide segment of the input signal to the 1.80-2.06 GHz range. The four LOs in the IF processor are agile to allow feeding any of the filter crates with any portion of the 4.0-6.0 GHz band. These are 6-8 GHz PLL YIG oscillators with digital control inputs driven by the Digital Interface.

The third IF processor feeds two 256-channel 250KHz filters and two Chirp spectrometers (Digital CTS). It has a 1.8-2.2 GHz output band. Each of its four converters may be independently steered to use any portion of either IF A or IF B.

4.1.1 Mechanical

The IF processor resides in a Bud RM-14123 4U x 17" deep rackmount box. The covers are TBC-14262 (ventilated).

The components are all mounted on a 1/4" thick aluminum baseplate that divides the box into a 2U upper RF section and a 2U lower power/logic section. All RF parts are bolted to the baseplate, with the semi-rigid coax cables floating above the parts.

The baseplate is ventilated with a large quantity of 1/2" holes. The rack fan draws air in the ventilated bottom cover, through the baseplate, out the top cover and out the top of the rack via the baffle plate above the IF processor box.

The RF connectors are grouped by function on the rear panel. N connectors are used for the six 4-6GHz IF inputs. SMA connectors are used for the four IF outputs and the test signal input. A BNC connector feeds the 10 MHz reference input. A set of four LO monitor SMA jacks on the front panel allow verification of LO stability and frequency. Red alarm LEDs alert the operator of a PLL unlock condition. A set of green front-panel LEDs indicates the current IF source selection.

Front panel indicators consist of the following:

- Power switch (1/2" toggle)
- Four LO monitor outputs (SMA)
- Three DC power supply monitor LEDs (green)
- Four LO out-of-lock LEDs (red)
- IF source LEDs (green) for each IF output or group:
 - 1MHz: 5 LEDs per box: DesertStar, IF A, IF B, Zero, Test
 - 250K/CTS: 2 LEDs per output: IF A, IF B
 - 250K/CTS: 3 LEDs per box: IF B, Zero, Test

4.1.2 Power Supplies

The IF processor uses 15V, 5V and 28V power supplies. 28V current is up to 0.6 Amps. 15V current is up to 1.3 Amps. 5V current is about 0.5 Amps. These are all open-frame linear supplies of frame size B, mounted to the bottom of the RF baseplate.

It was discovered that the magnetic field of a power supply transformer modulates nearby LO synthesizers, so one power supply had to be relocated away from the synthesizers. It had originally been mounted directly below channel 1's LO module, and was relocated to the rear of the box. The symptom was a 60 Hz frequency modulation of the LO signal with a 10 KHz bandwidth.

4.1.3. Computer interface

The IF processor connects to the digital crate's I/O card via an IEEE1284-C cable. The data bus is 8 bit parallel with strobe and 3 address lines which select one of eight ports. The ports provide control of the four synthesizers and 8 relays, 6 of which are currently defined. Additional ports provide control of up to 16 LEDs, eight switches or two attenuators, and an 8-way power detector module to be built if needed.

Synthesizer control signals consist of four PLL serial data streams (Clock, Data, Enable). There are eight TTL relay control signal pairs for latching relays.

Readback signals consist of four PLL lock detect bits and eight relay indicator pairs.

Sixteen LEDs are provided for. These indicate PLL lock and IF source selection.

4.1.4. RF stuff

The IF processor contains four downconverters, each with an agile synthesizer. An output amplifier provides 25 dB of gain to overcome the mixer and pad losses, which add up to about 20 dB.

A bandpass filter on the IF side of the mixer filters the converted signal to 1750-2250 MHz. This is sufficient to cover the 1800-2056 filterbank and the 2000-2200 MHz chirp spectrometer IF bands.

4.1.5. Temperature stabilization

During testing of the filterbank stability, it was discovered that the IF amplifiers have a temperature gain variation of about one percent per degree C. This gain variation was contributing an unknown amount to the Allan variance. It was decided to eliminate the temperature changes by mounting the IF amplifiers to a separate aluminum rail with a heater and temperature controller. The rail is held to approximately 38 degrees C by a proportional temperature controller using an RTD sensor to monitor the rail temperature. The FBTEMP controller card has onboard trimpot adjustments for temperature and loop gain.

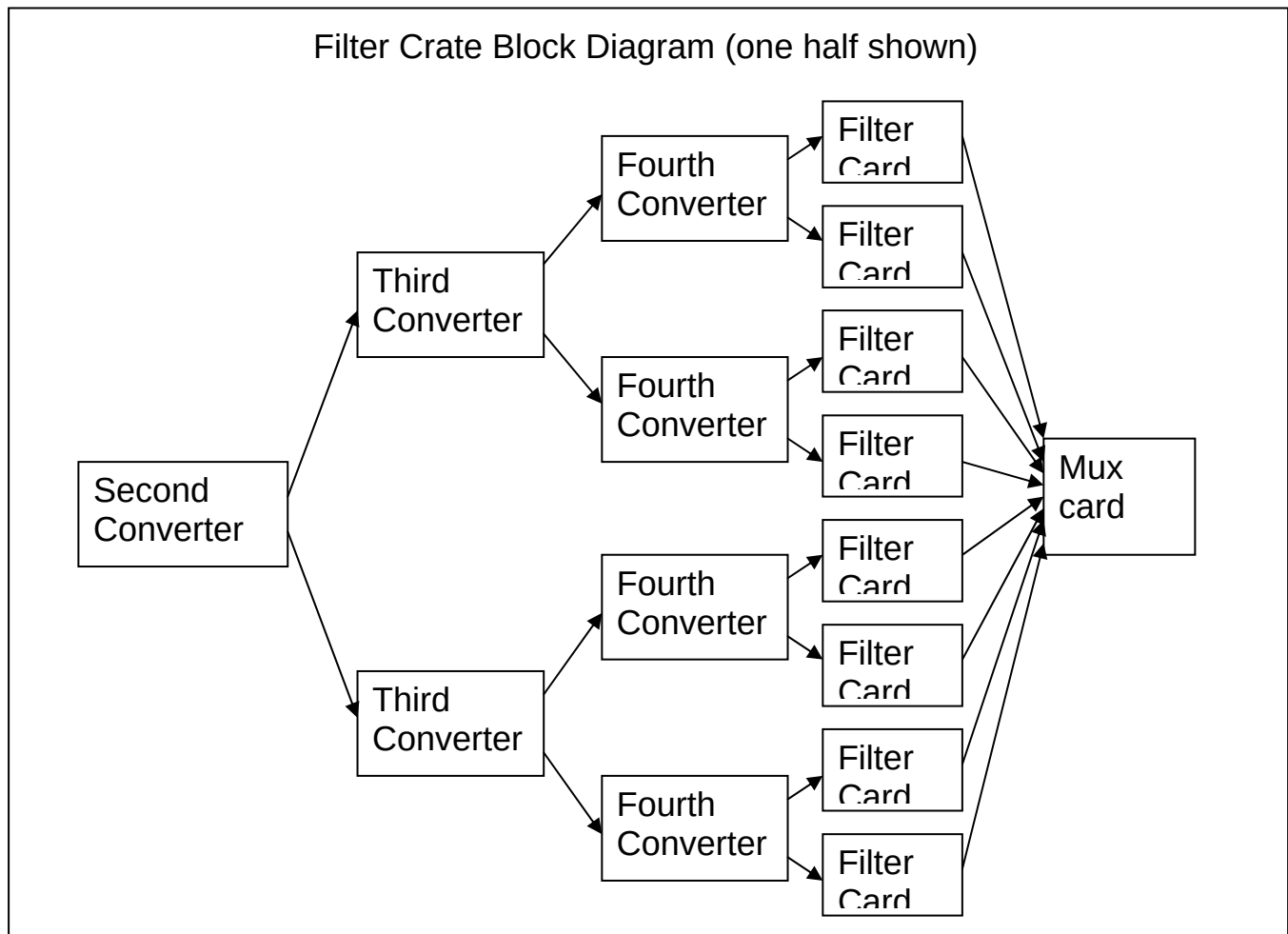
The stabilization is good to approximately 0.1 degree C over the operating range of the computer room.

A pair of 100 ohm RTD sensors (one on the baseplate, one on the heated rail) are connected to a 10-pin round connector on the rear panel for temperature monitoring purposes.

4.2 Filter Crate

Each filter crate processes 512 MHz of spectrum divided into two 256 MHz input bands. Four filter crates are used in the complete 2 GHz system.

The filter crate performs the second, third and fourth down-conversions of the IF signal as well as detection and multiplexing of the DC spectral output. Two IF inputs and one serial analog output per crate are provided.



The filter crate is filled with 3U and 6U Eurocards. These cards are:

2 3U FBCON2	Second Converters
4 3U FBCON3	Third Converters
8 3U FBCON4	Fourth Converters
16 6U FBFIL32	Filter cards
1 6U FBMUX	Multiplexer card

The 3U crate is located above the 6U crate in the rack.

The 3U crate and the top half of the 6U crate share an IF backplane. This backplane carries the IF signals in stripline between the converter cards and to the filter cards.

The bottom half of the 6U crate has a video backplane on the filter card P2 connectors which distributes power and multiplexer addresses, and feeds the analog filter output signals to the mux card.

Two write-only serial data busses are provided in the IF backplane: one for loading the PLLs and attenuators on the converter cards, the other for loading the zero DACs on the filter cards.

Each serial bus is based on the SPI standard with an added address mode for card and register selection. Four physical SPI busses are used to reduce signal loading and trace length - left and right copies of the converter and DAC busses. The serial busses are driven by the FBMUX card.

A programmable-function Error signal in each serial bus allows the host computer to read the status of the converter cards. This feature is used both for card detection at startup time and for PLL lock alarm reporting.

The 10 MHz station clock is distributed on the IF backplane via a terminated stripline. It is used as the PLL reference frequency.

Slot ID codes are built into the backplanes to provide geographic board addressing. An ID signal QMHz specifies the filter crate's resolution.

4.2.1 Mechanical

The physical arrangement of the filter crates is important in that it affects the ease of maintenance and repair, as well as the thermal stability of the system.

The Green Bank filter banks are solidly built, but their ease of repair (at least of the IF processing circuits) leaves something to be desired. This is because the IF amplifiers are mounted in boxes inside the filter crate, so the filter crate must be removed from the rack and disassembled to get at the IF circuitry.

To improve repair access, it was decided to package the filter bank electronics on Eurocards accessible from the front of the racks. The Eurocard standard has evolved to include cardcages with RF shielding strips. These are available from Schroff or APW in a suitable array of sizes.

RF shielding of the circuit on the Eurocards is accomplished with prefabricated surface-mount PC board shield units, of the type used in cell phones. These may not be needed, but footprints on the boards are provided to accommodate them if needed. Furthermore, the PC boards are designed with four layers, allowing the RF signals to cross the boards in stripline, which provides excellent signal shielding. The IF backplane is designed the same way.

The available DIN 41612 connectors of type M provide up to eight coaxial connectors plus 24 signal pins, which is sufficient for the RF signal routing requirements of the filter crate cards. The coaxial connector inserts are available in a PC-mount version suitable for building the IF backplane.

The RF circuitry on the filter crate cards is built entirely of surface-mount components. The parts include mixers, amps, splitters etc. from Mini-Circuits, Hittite and other mainstream RF component manufacturers. To facilitate repair, only components large enough to be handled with tweezers and soldered with a soldering iron (under a microscope!) are used.

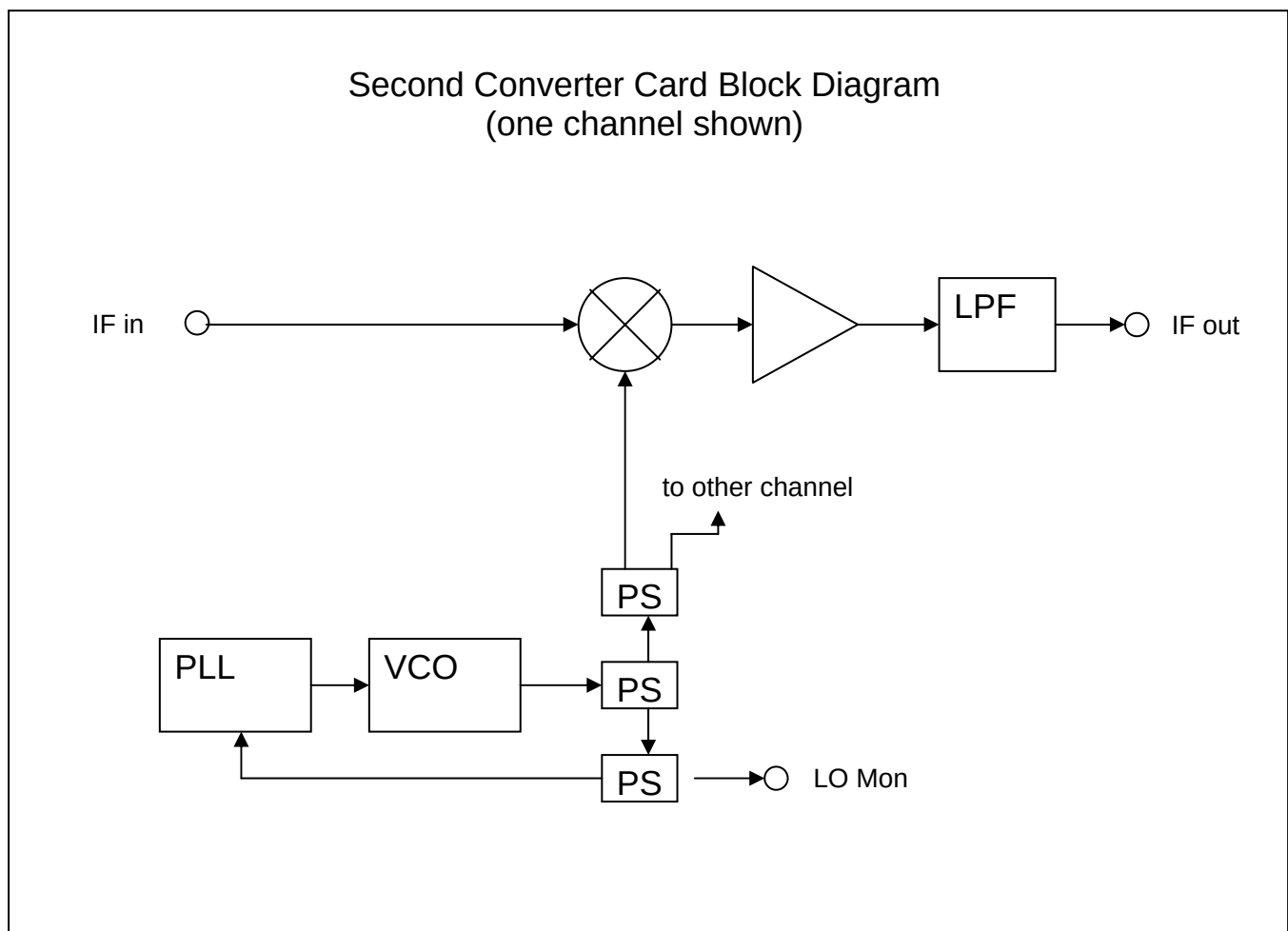
In the interest of conserving rack space in the control room, it is desired to make the equipment as compact as possible without compromising either performance or serviceability. An analysis of cardcage requirements leads to the choice of a 512 channel filter crate comprised of a 6U cardcage for the 16 filter cards and multiplexer card plus a 3U cardcage for the converter cards.

The cardcages are mounted in the front of the racks, allowing front access to all cards. The rack height is 9U for 512 channels. A pair of 1U air baffles must be mounted above and below the crate for a total installed height of 11U.

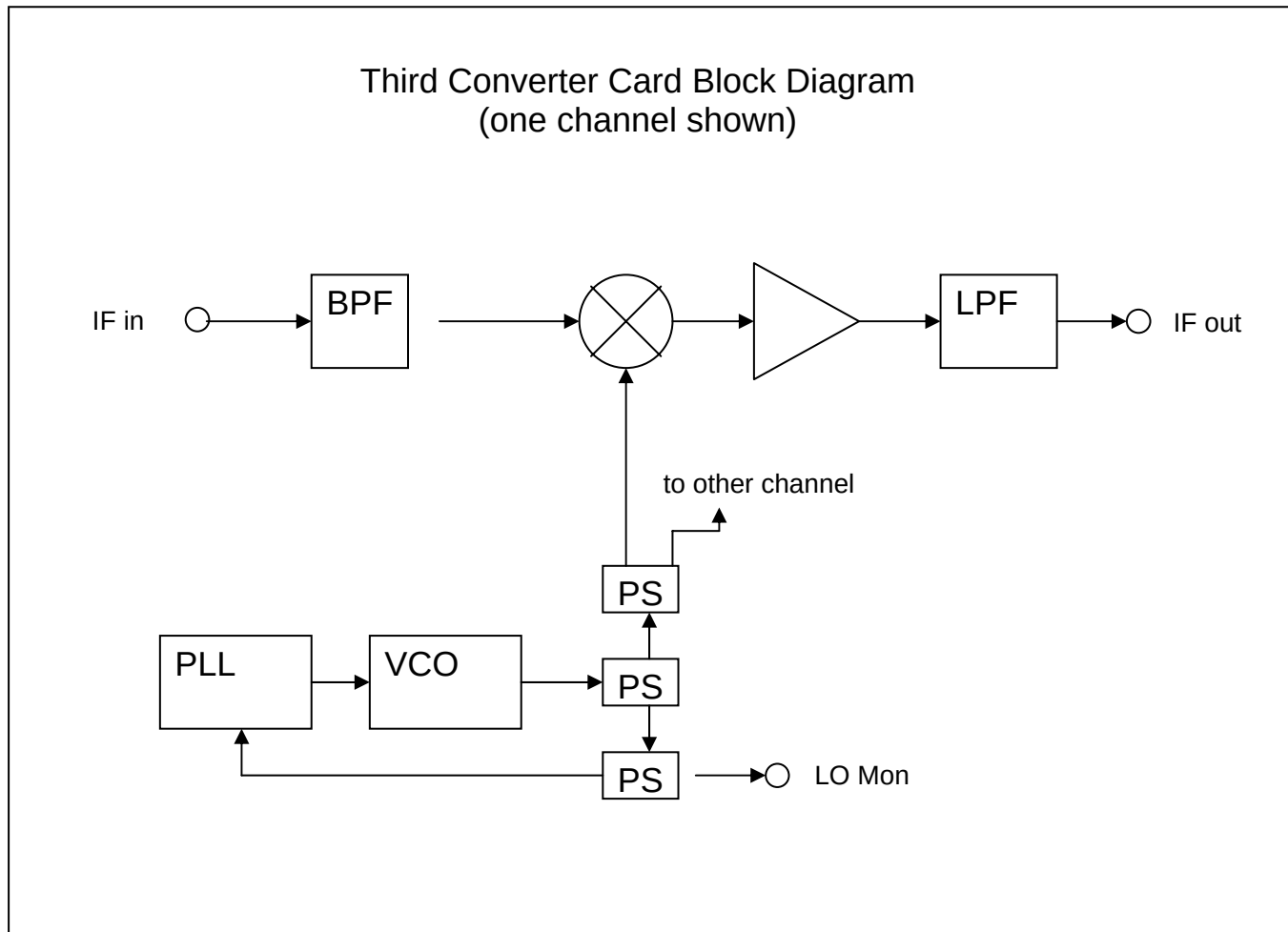
4.2.2 Converter cards

There are three types of Converter cards: Second Converters, Third Converters and Fourth Converters. All three types of converter cards are 3U Eurocards with coaxial backplane connectors for the IF signals. There are two styles of Fourth Converter cards, depending on the filter resolution.

The Second Converter cards FBCON2 accept two IF inputs in the 2GHz range and produce four outputs in the 500 MHz range. The Second Converter card has no onboard bandpass filters, depending instead on filters located in the IF processor. This card has a single LO in the 2.4 GHz range. The LO frequency is determined by which slot the card is plugged into.



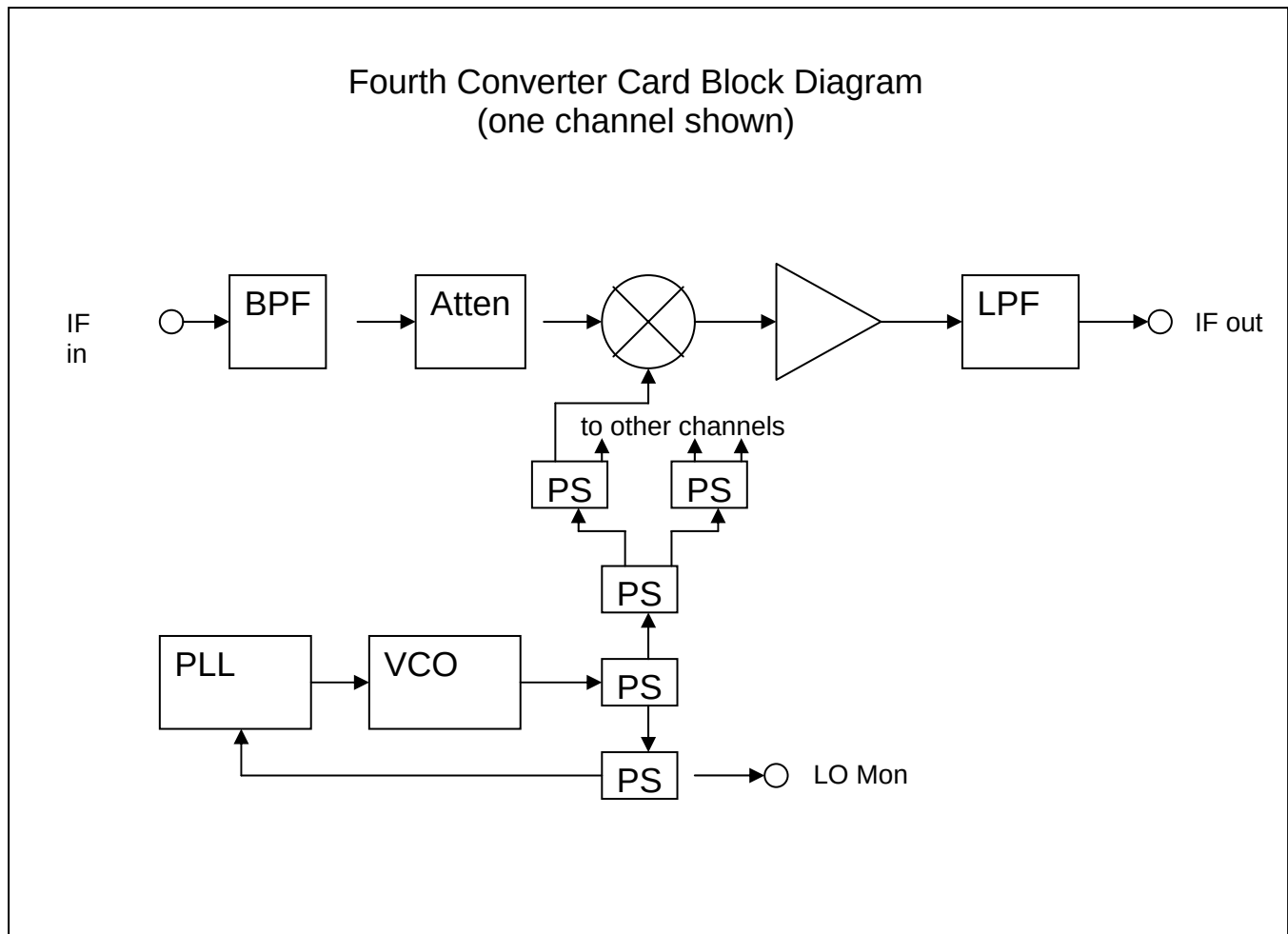
The Third Converter cards FBCON3 each convert two IF inputs in the 500 MHz range to four outputs in the 140 MHz range. The IF inputs are filtered with bandpass filters which are built in two different frequency ranges. Otherwise these cards are quite similar to the Second Converter cards in operation if not appearance.



There are two versions of the FBCON3 cards: FBCON3-1 which operates at 422-486 MHz, and FBCON3-2 which operates at 486-550 MHz. The different parts are the bandpass filters on the inputs. Only the FBCON3-2 card is used for the 250K filters, whose combined input frequency range is 506-538 MHz.

The IF Backplane contains splitters to produce double copies of the Third IF signals before passing to the Fourth Converter cards. This was done to save the complexity and expense of adding splitter cards, since the Third converter cards would otherwise have needed ten coax connectors apiece.

The 1 MHz Fourth Converter cards FBCON4 accept four inputs from a 16 MHz subset of the 110-174 MHz range and convert each down to a 28 MHz center frequency. Each input has a bandpass filter and a programmable attenuator to achieve computerized gain equalization of the IF system.



There are four different versions, each with a different input filter center frequency and LO frequency. They are: FBCON4-1 at 110-126 MHz, FBCON4-2 at 126-142 MHz, FBCON4-3 at 142-158 MHz, and FBCON4-4 at 158-174 MHz. Again, each has its own bandpass filter frequency. The LO low-pass filter cutoff frequency is also board-dependent.

The 250 kHz Fourth Converter cards FBCON4-25x accept four inputs from a 4 MHz subset of the 90-106 MHz range and convert each down to a 7 MHz center frequency. The four different versions are: FBCON4-251 at 90-94 MHz, FBCON4-252 at 94-98 MHz, FBCON4-253 at 98-102 MHz, and FBCON4-254 at 102-106 MHz.

Each stage in the downconversion process is characterized by having two or more adjacent frequency bands. The input frequency range selected on the converter and LO cards by means of a slot ID code. The host computer sets each card's PLL to its required LO frequency, as determined by its slot number, at system startup time. An error signal allows the computer to verify that all the correct cards are installed and that the PLLs are locked on frequency.

All types of converter cards have one onboard LO apiece, which is described below.

4.2.3 Local Oscillators

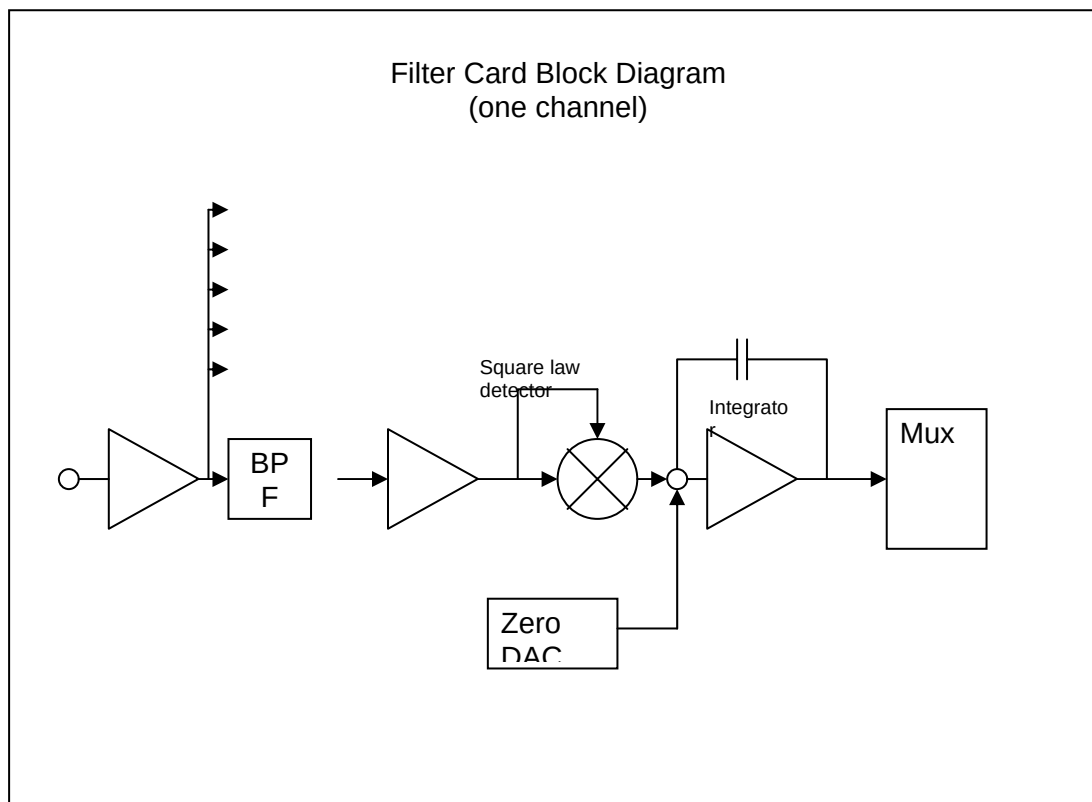
The second through fourth conversions happen at reasonable frequencies for PC board construction. The local oscillators are built with Analog Devices PLL chips and Z-Comm or Modco half-inch square VCO modules.

The PLL reference frequency of 10.000 MHz is obtained from the station clock (rubidium oscillator or hydrogen maser) and distributed on the IF backplane in an end-terminated 50 ohm line. The converter cards tap into this line with 1K ohm resistors. The backplane expects a reference clock level of approximately +10dBm.

An interesting problem was discovered with the lower-frequency VCOs made by Z-Comm as used on the 250 kHz FBCON4 cards. The tuning port is not compatible with a passive PLL loop filter at V_t below 2.0V. The problem is that the RF from the oscillator is rectified by the varactor tuning diodes and coupled into the tuning port as a DC current, up to about 70 microamperes. This required retuning all the FBCON4-25x VCOs by adding a 5.0 pF capacitor in parallel with the varactor diodes on a spare SMT footprint.

4.2.4 FBFIL32 Filter cards

Previous filter bank designs have put the integrators and multiplexer in a separate card cage from the filter cards. This makes for a lot of cabling and extra hardware to get all those analog signals from one crate to the other. The approach taken here is to put all this circuitry on the filter cards, which dramatically reduces the size of the system without impacting performance.



The Filter cards are 6U Eurocards which contain two filter drivers, 32 channel bandpass filters, 32 square-law detectors, 32 analog integrators and an analog

multiplexer chain. Each filter card is essentially two separate 16-channel filter circuits.

4.2.4.1 Filter drivers

Each 1 MHz filter card has two 16-channel filter sets that operate at 20 to 36 MHz. The 250 KHz filter cards operate at 5 to 9 MHz.

The bandpass filters require drive of about -30dBm per channel for a total power of about -17dBm. The channel filter distribution lines are each driven by an LT1395 in cascade with an OPA685 high-speed, low-distortion op amp. The filter distribution lines are AC-coupled to the line (which is biased to -4.0V) and end-terminated in 100 ohms.

The filters are each driven by a transistor driver whose base is directly coupled from the distribution line and whose collector circuit is the first resonator of the bandpass filter. This stage provides about 15-20dB of gain and matches the line impedance of 100 ohms to the filter impedance of 1K ohms.

4.2.4.2 Channel Filters

The channel filters are two section LC bandpass filters of Butterworth response. These are based on the NRAO Green Bank filters from the 1970s. This old filter bank design is well-engineered and has proven itself in decades of service.

Each filter consists of two capacitively-coupled parallel resonating sections. These resonators have a constant impedance of about 1000 ohms, which means that the Q increases with center frequency. A fixed resistor is added to each resonator to set the Q to the desired value. The end result is consistent filter response across the band with minimal tuning effort. The Q adjustment resistor values were selected empirically after the first filter card was built and sweep-tested. The filter shapes are matched between the 16 filter frequencies to about 1% of output level shape.

An LT1395 op-amp after each filter boosts the signal into the square law detector. This amplifier was added to reduce system noise after the first baby filterbank test revealed excessive noise from the square law detectors. Another benefit of this amplifier is that with reduced gain, the square law detector zero drift over temperature is greatly reduced. This has eliminated the need for frequent zero adjustment.

4.2.4.3 Detectors

The square-law detectors are Analog Devices AD835 multiplier chips connected as squarers. The inputs are wired to produce a negative product term so that the integrator will produce a positive final result for the A/D converter.

The detector output is filtered with a 10uS time constant to remove the RF component. This time constant is not critical, since the observed signal (a spectral line) is a tone and therefore has no high-frequency component in the detected video signal.

4.2.4.4 Integrators

Each channel's power signal is integrated over a 25 millisecond period by an analog integrator. The integrators are each made with an LF353 op amp, a polypropylene (PPS) capacitor and a dual analog switch for integrate and charge-dump control. A voltage follower op-amp isolates the integrator from charge injection caused by multiplexer switching.

The 25 millisecond period is selected to provide a short enough integration time to not require extreme care in the design of the integrator, yet long enough time to provide good temporal efficiency over 512 channels. The efficiency is about 98% with a 1 megapixel/second readout rate.

4.2.4.5 Offset Correction DACs

The square law detectors have substantial zero offset, too large to be corrected by fixed-value trimmer resistors. To fix this, one DAC per channel is provided to set the offset voltage of the square law detectors.

A further DC offset is added via a range-setting resistor per channel (R4_xx) to bring the DAC control voltage within range. This resistor value is changed at board checkout time if needed, after the PLD is programmed.

The fbcal program prints a report of all R4 resistors that require changing. The boards are assembled with 33K ohm R4_xx resistors. If a channel reports that R4 needs to be increased in value, its R4 is changed to a 47K ohm resistor. If it needs to be increased again, a 100K resistor is used. If a further increase is needed, the resistor is removed.

Each channel requires a unique offset voltage which is determined at calibration time. The DACs are loaded from the video backplane by a serial command stream. The reference voltage for the DACs is generated by a REF191 bandgap reference chip.

The fbcal program determines the required DAC setting by measuring each channel's DAC gain with two different DAC settings, measuring the channel's offset ADC result, and extrapolating the required DAC setting from these two numbers.

4.2.4.6 Multiplexer

Each filter crate has a total of 512 channels of analog data to pass to the A/D converter in sequence. The analog multiplexing function occurs in two stages. There is a set of multiplexers to select one of 32 channels on the filter card, and a separate FBMUX card selects one of the 16 filter cards as the analog data source.

After the analog integrators have accumulated data, they are disconnected from the detectors to prevent digital noise from the multiplexer control circuitry from getting into the analog data. Then the multiplexer is scanned through all 512 channels at a rate of one channel per microsecond. The address bit wiring of the filter cards is arranged so that the scanning occurs in straight RF input frequency order. After the data has been sent to the digital crate, the DUMP pulse resets the integrators. The standard DUMP time is 100 microseconds, but it could be shorter.

The analog multiplexer on the FBFIL32 card consists of five 8-way multiplexer chips feeding a sixth chip. The fifth multiplexer chip has four extra inputs which are used for reference levels of zero and one volts, which are presently unused. The sixth chip selects analog data from one of the first five chips.

A buffer amplifier drives the analog signal down the backplane over a video pin which is wired on the backplane in a star pattern. This is a 100 ohm end-terminated transmission line. A multiplexer on the mux card selects the desired video signal.

4.2.5 FBBP Video backplane

The video backplane selects the filter channel on each filter card and collects the multiplexed video signals from all filter cards. It also feeds the INT and DUMP signals to the filter cards.

4.2.6 FBMUX Multiplexer card

The Multiplexer card presents the analog integrated filter signals to the digital interface box via the video cable. It also receives the differential frame sync and data clock pulses to step the analog multiplexers through the channel sequence.

The functions provided by the control interface are as follows:

- * Load card register
- * Load channel register
- * Increment channel/card numbers
- * Write to SPI busses

A multiplexer selects one of sixteen video signals from the filter cards. This signal is converted to differential via a Maxim MAX4142 video line driver. A rear panel connector sends the video signal through the video cable to the digital crate.

The Multiplexer card receives the RS-422 scan clock and scan-enable signals from the integrator card and converts them to an incrementing channel number. The channel number is bit-swapped and driven to the FBBP card and channel busses. A Xilinx CoolRunner CPLD performs the interface functions.

The multiplexer card distributes the SPI serial bus signals to the converter and filter cards. The SPI interface is an 8-bit register with two sets of four-bit SPI control signals: Clock, Enable, Data and Adr. Adr selects address mode, used to load a register address.

An error readback signal allows the computer to poll the filter crate for wrong or missing cards, as well as such errors as converter PLL out-of-lock.

4.2.7 Video cable

The video cable is a standard IEEE1284 C-C printer cable which has a high-density 36-pin connector on each end and 18 twisted pairs of wire in a shielded round cable. It is available off-the-shelf in lengths from 3 to 30 feet for a reasonable price (\$50). The cable plug is secured to the panel receptacle by means of thumb-release latches.

Signaling is TTL for the data bus, RS-422 differential for the control strobes and differential analog for the video signal.

4.2.8 Power Supplies

Regulated open-frame linear power supplies are used for the filter crate. Each 512 channel filter crate uses about 150 Watts of power. The power consumption has been measured with a few cards and extrapolated as follows.

+5V Analog 12.8A
-5V Analog 14.4A
+5V RF 2.6A
+5V Digital negligible

Two 5V at 18 Amp supplies provide analog +/-5V to the filter cards. A 5V at 3 amp supply powers the digital circuitry, and a separate 5V at 3 Amp supply powers the converter RF circuits.

The power supplies had been mounted in the rear of the filter crates, but this posed problems in both overheating and filter stability. As a result, the filter crate power supplies have been relocated to a separate box.

One power supply box feeds both filter crates in a rack. It has four 5V 18A supplies and four 5V 3A supplies, as well as three muffin fans to cool the supplies. A separate power switch controls power to each filter crate. The fans are always powered up when the power supply box is connected to its AC source.

4.2.9 Cooling

The filter crates have been tested on the mountain in their original configuration with the power supplies in the rear of each filter crate. The power supplies run hot in the thin air, with 55C measured on the supply frame. Also, the Allan variance testing revealed that the room air circulating through the filter crates causes short-term instability. The Allan variance time was doubled to 25 seconds by isolating the room air from the filter crate air and recirculating the filter crate air to cool the boards.

The filter crates require stable air temperature to provide good signal level stability. They also require moving air to cool the filter cards, which generate a fair bit of heat. The solution is to circulate air inside the filter crate yet isolate that air from the room air.

A fan plate is installed in the rear portion of the filter crate. Two 115VAC muffin fans blow air downward towards a 1U box with an open top, which forms a U-turn and directs the air up through the filter cards and converter cards. A second 1U box with an open bottom, mounted above the filter crate, directs the air back down to the fans again. The heat is taken from the filter cards and spread to the front, sides and rear of the filter crate, where it is transferred to the room air.

4.3 Digital Interface

The digital interface is a 3U Eurocard crate that holds the A/D converters, timing generator and PC parallel interface cards. It is based on the AOS interface designed by Dave Ashby, but with changes to reduce noise and provide increased flexibility.

The digital crate contains the following cards:

4 3U Integrator cards

1	3U	Parallel port controller
1	3U	Timing generator
2-3	3U	IF crate controller

These are described in detail below.

4.3.1 FBBP Digital Backplane

The digital crate's backplane is the same PC board design as the filter crate's digital backplane. It has 17 slots total. The center slot is 4HP wide and holds an FBEPP interface card - this slot is not numbered. The other 16 slots hold Int, Timing or I/O cards, slots 0-7 to the left of the center card and slots 8-15 to the right.

The digital backplane carries host data bus, system timing signals, board decoding signals and power to the cards in the digital crate.

The backplane is designed with low noise paramount. There are many ground pins spread among the signal pins to reduce crosstalk. The digital signaling is done synchronous to the 10 MHz clock. The board select signals, one per slot, are derived from the EPP port's address write function and change infrequently.

The 10 MHz bus clock signal is distributed from the FBEPP card which resides in the center slot between slots 7 and 8.

4.3.2 FBINT Integrator card

The FBINT Integrator card connects the filter crate to the computer. Four of these input cards are required to support 2048 channels of filterbanks. It is possible to place up to eight Integrator cards in the system for a maximum system size of 4096 channels.

The Integrator card receives the video signal from the filter crate multiplexer card over an analog twisted pair. It sends the scan reset, scan clock, INT and DUMP timing signals to the filter crate over RS-422 differential pairs.

The Integrator card provides a gain/zero normalization system. Each channel has a gain number and a zero number associated with it which are stored in the Integrator card in digital form and applied to the A/D converter as it converts the analog filter data. The result is the ability to equalize the gain of all channels and flatten the baseline to within a few counts of the A/D converter.

The A/D converter is 14 bits wide and operates at one channel per microsecond. A 12 bit multiplying D/A converter is placed in series with the analog input to allow the gain of the A/D converter to be controlled digitally for gain calibration of each filter channel. This eliminates the gain pots on the filter cards.

The gain and zero correction circuitry may be disabled under software control to allow a gain/zero error map to be generated. This map provides the ability to check the system performance - a marginal or failing component will cause changes in the map from the previous test run. The system may be set up to automatically check for this type of problem before every observing run.

The A/D converter's digital output enters a field-programmable gate array (FPGA) which is programmed as an integrator. The integrator sums successive frames of data

for all 512 channels. It stores and subtracts a separate zero correction value for each channel, thereby eliminating zero pots on the filter cards.

The FPGA has internal SRAM that stores two successive datasets of integrated data as well as the gain and zero values for each channel. The total SRAM size is 40K bits, which is divided across 512 channels as 16 bits each of gain and zero plus two frames, 24 bits wide, of integrated data.

The digital integrated data is scanned out of the SRAM onto the digital backplane to the PC over the parallel port. This data reading activity may occur simultaneously with the A/D converter working.

Two BNC connectors on the front panel allow input of TTL level signals, and are earmarked for Local Blanking signals. Each local blanking signal has its own enable bit, and all are wire-ORed on the digital backplane to be used by the Timing card. Alternatively, the BNC connectors can be used as output control bits or general-purpose input bits.

4.3.3 FBEPP Parallel port controller

The parallel port controller card provides the EPP parallel port interface from the digital crate to the PC. A programmable logic device converts the EPP parallel port protocol to the synchronous bus structure of the digital crate. This consists of resyncing the strobes to the 10 MHz bus clock, capturing the port address from the PC and decoding the board select signals from this address.

A 36 pin high-density IEEE1284-C connector on the front panel connects to the host PC. A BNC connector receives the station 10 MHz clock which is used for the filterbank system timing.

This card may be upgraded in the future if necessary to modern PC interface standards such as FireWire or USB or whatever Intel comes up with. The required interface consists of an 8-bit address register, an 8-bit data register, and the ability to read or write data as an ordered series of bytes to/from the specified address.

4.3.4 FBINT Timing generator

This card generates the system timing signals from the telescope Blank and Sync signals. It provides integration time and observing phase data to the host PC. Two BNC connectors on the front panel receive blank and sync signals from the telescope.

The Timing card is made from the same PCB as the Integrator card, but the A/D and D/A converters and 36-pin connector are not stuffed. Instead, the two BNC connectors are stuffed.

Details of the timing generation system follow.

The system can be in one of four modes: Int, Scan, Dump or Idle. Int mode is when the filter cards are integrating power in their analog integrators. Scan mode is when the analog power information is read sequentially from the filter cards to the Int cards. dump mode is when the analog integrators are reset. Idle mode is when the system is not taking data, typically because no observation is in progress.

The Sync and Blank signals from the telescope control the timing card. Blank informs that the subreflector is not in position. Sync informs that the subreflector is in phase 0, which is on-source for the SMT. Sync is used to reset the data bin number to 0 in the digital integrators. Blank is used to interrupt the taking of data and force the data transfer to the computer.

Local blanking is used to prevent bad data from getting to the computer. It consists of two inputs to each of the Int cards. Any of the local blank inputs may be independently enabled. Each Local blanking input has individually programmable polarity.

When a Local blanking input is enabled, it is monitored during Int time. If Local blank goes true for at least two microseconds at any time during the Int period, then the subsequent Scan period is skipped and the analog data are discarded in the ensuing Dump phase.

4.3.5 FBINT IF crate controller

This is a partially stuffed FBINT card with different firmware. Its purpose is to send configuration commands to the IF processor via a parallel interface cable.

The front panel connector drives an 8-bit parallel command bus to the IF processor. The bus provides access to eight 8-bit registers in the FBIFC card in the IF processor. These registers control the relays, synthesizers and LEDs in the IF processor box.

The analog video signal pair in the cable is reserved for an optional power detector signal.

Two IF crate controllers are needed for the 2048 channel filterbank. A third IF crate controller is needed for the 250 KHz filters and Chirp IF processor box.

4.3.6 Power Supplies

Regulated linear power supplies are used for the digital crate. These are mounted to Eurocard panels to form plug-in units. Separate +/-5V analog and +5V digital supplies are provided. (We'd buy Eurocard power supplies, but they are no longer made!)

4.3.7 Mechanical

The digital crate is a 3U high, 320mm deep Eurocard box. The digital backplane is mounted in the rear-facing direction so that the cards plug into the rear of the crate. The power supplies plug into the front of the crate. This is done so that the cables emanating from the cards' front panels will be inside the equipment rack.

5. Filter Alignment

The converter cards require no alignment.

The filter cards have two coils and one capacitor for each filter channel. Each of these filters must be aligned before the cards can be used. Fortunately, this task has been automated as much as possible short of building a robot tuner.

Filter alignment is done with a program running on the control PC which draws a real-time filter response curve on its screen with a template for adjustment limits. The PC also controls the RF test signal generator. The technician (me!) then must simply adjust the three filter tuning slugs until the curve matches the template, then press Enter to move on to the next channel.

Instead of buying a GPIB-programmable signal generator, a practical test generator has been made from two spare Fourth converter cards, wired to allow tuning over the necessary range under control of the host PC. The LO is not leveled, but this is not a problem for aligning the filters as the filter bandwidth is only 5%.

A triple tuning tool has been made to speed the alignment task. It is a metal frame which slides over one set of filter coils and holds three tuning blades in precise alignment.

The DAC zero resistors must also be selected for each channel. The default value of 33K ohms works for about 80% of the channels. The others need 47K, 100K or no resistor. The fbcsl program generates a list of which resistors on which cards need to be changed, for one crate (up to 16 cards) at a time.

The time to align a card is estimated at 20 minutes for filters, and another 10 minutes for the DAC zero resistors. Actual experience makes it closer to an hour per card total, although there have been corrosion problems with the tuning capacitors that made some cards take longer. The bad capacitors have since been replaced.

Alignment procedure:

The filter card alignment procedure has four steps.

First, the Xilinx CPLD chip must be programmed. This is done one card at a time. Plug in the Xilinx programming dongle to the card. Turn on power. Run the Xilinx IMPACT program. Do what it says. Power down the card, swap it out, power up, and program again as needed. No need to stop or restart IMPACT for each card -- it gracefully handles the card swapping.

Next, a batch of cards is placed in a filter crate. Power is applied, the cards are allowed to temperature-stabilize for 10 minutes, then the program fbcsl is run. fbcsl produces a list of all zero DACs that are out of range. Print the list and use it as a guide to replace the R4_xx resistors from 33K to 47K for all channels that need R4 increased, or 22K resistors for channels that need R4 decreased (none observed so far).

Replacement of resistors goes fastest with the Metcal Talon hot tweezers. No flux needed - just take off the old parts, put on the new and heat.

The cards are tested again with fbcsl. Any channels that need R4 increased again get a 100K resistor.

The cards are tested a third time with fbcals. Any channels that still need R4 increased require the removal of the 100K resistor. Not likely.

The next phase of alignment is tuning the channel filters. A special program, tuning tool and test setup have been created to speed this task.

The test setup to use sweep requires the insertion of two special FBCON4-B cards into the converter cardcage. These are the Sweep Test Oscillator and the Sweep Converter. They are S/N ST0 and SC respectively. The SC card goes in the leftmost converter slot, the FBCON4-4 slot. The ST0 card goes in the third converter slot from the left, the FBCON3-1 slot.

The filter card being aligned goes in the leftmost filter slot. All slots to the right of it up to the Mux card must be empty to fit your hand into the cardcage to adjust the coils and capacitors.

The alignment program is called sweep. It starts by displaying the response of the first channel in an X window via the spectro program. The display has the following features:

- * Horizontal grid lines showing 0%, 50% and 100% response
- * Vertical grid lines showing low and high frequency bounds of this channel and the two adjacent channels
- * Purple desired-response curve
- * Green actual-response curve
- * Numeric display of channel number in upper left corner
- * Floating display of left and right channel bound difference from 50% in units of .01%

When the sweep program is run, it initializes the filter crate, then displays channel 0's (top channel on the inside row) response. The display will typically be offset in frequency and wider than desired, with a dip in the middle of the passband. This is a result of the factory settings on the coils (slug flush to top of coil form) and capacitor (maximum capacitance).

The 1 MHz controls behave as follows: Turning a coil slug clockwise will increase the frequency. Turning the capacitor clockwise will narrow the passband. The passband width is biased, so that the lower cutoff frequency changes 10% as much as the upper cutoff frequency with the bandwidth adjustment.

The triple tuning tool is good for the first pass at tuning. It speeds up the process by holding three alignment wands in place, which allows all three controls to be turned in sequence without having to repeatedly put a wand into each coil or capacitor and get the blade lined up with the slot.

The tuning procedure for a channel is to first adjust the coils to get the frequency close, then adjust the capacitor to get the bandwidth close, then adjust coils alternately to get the top flat and the frequency correct, then trim the bandwidth with the capacitor. Since the controls interact, some experience is needed to get fast at it. But 2500 channels provides plenty of experience.

The first pass with the tuning tool isn't going to be entirely accurate due to its stray capacitance, so no need to get it perfect now.

The flattening of the top is affected by the relative tuning of the two coils. You will see a pattern emerge if you play with both coil controls one at a time.

turning a single coil will both shift the center frequency of the peak and tilt the top. The trick is to adjust both coils so that the tilt reaches horizontal just as the tuning gets centered.

Another thing to observe is that when the bandwidth is wider than desired, about 80% of the excess passband width is on the high (right) end of the peak. So when adjusting the frequency, aim for the right if the bandwidth is too wide.

After the channel is aligned reasonably well (error within +/- 500 counts), press ENTER while focused on the sweep text window, and the next channel down the card will be displayed.

Some coils must be turned clockwise several revolutions to get the slug in the right place. These ones will not display a response bump at all when first selected. The coils in question are yellow and green.

After all coils and capacitors have been tuned approximately with the triple tuning tool, run sweep again and check the tuning, touching up each channel as needed with individual wands. The yellow Spectrol model 8 tool fits the capacitors, and the blue/green ceramic 1.3mm blade tool fits the coils. The error display number want to be within +/-100 counts, and the top of the response peak should be flat.

6. Installation

The filterbank system is installed at the telescope, but parts may need to be removed for repair. The following describes the physical arrangement of the system, cabling, etc.

Each of the three filterbank racks contains an IF processor, one or two filter crates, a filter power supply, and various other equipment.

The left rack contains the digital crate, a 10 MHz station clock distribution amplifier and the PC. The center rack contains the video monitor and keyboard. The right rack contains the noise source, the chirp spectrometer(s) and empty space for future expansion.

The center rack contains two extra items in the rear area. An IF distribution panel at the top feeds the IF A and IF B signals to the IF processors in all three racks. A DIN rail-mounted National Instruments FieldPoint data acquisition system is connected to RTD temperature sensors in all three racks and reports their temperatures over a serial cable to the PC.

Each filter crate has air baffle boxes both above and below the crate. These must be in place for the cooling system to work properly.

6.1. Cabling

The system cabling is complex but straightforward. There are several classes of cables, each of whose connections are described below.

6.1.1. Power

Each box receives AC power from a power strip mounted in the rear of each rack. The strips are always on. Each is fed from a different phase of the UPS-fed AC and are plugged into outlets in the crawlspace above the computer room ceiling. The three loads have been balanced to keep the UPS running efficiently. The fan in the top of each rack remains powered at all times.

6.1.2. Ethernet

The 100BaseT Ethernet cable from the building switchbox connects to the PC's Ethernet port.

6.1.3. Parallel

There are two classes of parallel cables: The IEEE1284 A-C cable (DB-25 to HD-36) connects the PC to the EPP card in the center of the digital crate. A bunch of IEEE1284 C-C cables (HD-36 both ends) connect each other card in the digital crate to its IF processor or filter crate. The Int cards (card 0 is at the leftmost slot) each connect to a filter crate; the IFctl cards (card 0 is just to the right of the center slot) each connect to an IF processor. The first filter crate is the top left unit, second is bottom left, third is top middle, etc. The first IF processor is in the left rack. The Timing card in slot 7 (just to the left of the center slot) has no parallel cable connection.

6.1.4. Station clock

The 10 MHz station clock is received from the clock rack in the receiver room and fed to a distribution amplifier in the left rack. From there, 10 MHz clock signals go to each box in the system except the power supplies. The digital crate receives its clock on the BNC connector on the EPP card in the center slot. Each other box receives 10 MHz on its only rear-panel BNC connector.

6.1.5. Blank/Sync/LBlank

The telescope makes Blank, Sync and a half-dozen LBlank signals. The Sync signal is the bottom BNC on the timing card in slot 7. Blank is on the top BNC connector of that card. The first LBlank signal, LBlank0, is on the bottom of the leftmost Int card, LBlank1 on the top connector of that card, LBlank2 on the bottom of the next Int card, etc.

The blanking signal wiring is as follows:

Sambus	Cable	Signal	Description
=====	=====	=====	=====
9	Red/W	Sync	Inverted sync
21	Blue	Blank	Subreflector blanking
0	Black	LBlank0	On Source
7	Brown	LBlank1	Focus
28	Red	LBlank2	JT Gunn Lock
29	Org	LBlank3	345 Gunn Lock
30	Yel	LBlank4	490 Gunn Lock
[27]	---	LBlank5	[unused Rx lock]

6.1.6. IF

The eight DesertStar IF signals come from the DesertStar receiver in the right receiver room, through the crawlspace and into the left two racks, four into each rack. IF signals 1-4 go to the four leftmost N connectors on the rear panel of the IF processor in the left rack, and 5-8 go to the IF processor in the center rack.

The two 5 GHz IF signals A and B come from the AOS rack to the IF distribution panel at the top rear of the center rack. From there, IF A and B go to the rightmost two N connectors on the rear panels of the three IF processors.

Each IF processor has four IF outputs on SMA connectors, one below each of the DesertStar inputs. These are connected in pairs to the two SMA IF inputs on each of the two filter crates, 1-2 going to the top filter crate and 3-4 to the bottom filter crate.

The right rack has one filter crate and one or two chirp spectrometers. The first chirp spectrometer receives IF 3.

6.1.7. Test source

The test source in the right rack has three SMA noise/comb outputs. Each of these connects to the Test input SMA connector to the left of the 10 MHz BNC input connector on the rear panel of each IF processor.

6.1.8. Filter crate DC power

Each filter crate has a big round military connector for DC power. The top filter crate connects to the right connector on the rear of the power supply, while the bottom filter crate connects to the left connector on the rear of the power supply.

6.1.9. Temperature sensor

Each box has a small round military connector that connects to one or two RTD temperature sensors inside the box. The wiring of these to the FieldPoint RTD system is as follows:

1,2	Left rack IF processor
3,4	Center rack IF processor
5,6	Left rack power supply
7,8	Center rack power supply
9,10	Top left rack filter crate
11,12	Bottom left rack filter crate
13,14	Top center rack filter crate
15,16	bottom center rack filter crate

This wiring scheme is subject to change when the right rack sensors are wired in to the new channels 17-24.

7. Operation

The filterbank system is highly automated, so it is rather simple to use. All IF switching, leveling, calibration, etc. tasks are performed by the computer system. The operator only has to keep the power on, the fans running and the filters clean.

7.1. Startup sequence

The racks, filter crates and power supply boxes all have cooling fans that run continuously. These units may have been unplugged for summer shutdown. If so, plug them into the power strips at the nearest available outlets. It would be prudent to clean the power supply fan filters before powering on the power supply units.

Plug in the power cord to the clock distribution amplifier, located above the PC. The unit's front panel LEDs should all light green.

Plug in the power cord to the FieldPoint temperature monitoring system inside the center rack. Its LEDs should start flashing.

The PC is the first item to turn on. It has a rocker switch inside the front panel, which is guarded with a key. If the LCD monitor is not turned on, do that at this time. Its power switch is the rightmost button on the bottom right corner. The PC will take a couple minutes to boot Linux. It will display a login prompt when finished booting.

The digital crate is the next item to turn on. Its power switch is located inside the leftmost rack, accessible from the rear, near the top of the rack at the right side as viewed from the rear of the rack. Flip the switch up to power it on. Check that the three green LEDs on the front panel of the digital crate are lit, indicating that DC power is good.

Turn on the three IF processors next. Each has a front panel power switch that must be flipped up. Check that the three green power LEDs are lit. The four red "LO Unlocked" LEDs on each IF processor should flicker on when power is first turned on, then go out within a second or two.

Finally, turn on the five filter crates. Each has a switch on the power supply. The left side switch controls power to the filter crate above the supply, and the right hand switch controls power to the crate below the supply. Do not turn on the rightmost switch - it does not feed any filter crates so is wasted power.

Let the filterbank system warm up for two hours before attempting to take data. The system has long thermal time constants to improve the stability, so it takes a while before it's usable.

After the temperature has stabilized, the filterbank control programs may be started. Start the ffb_accum and ffb_control tasks on the main Rambo control screen. It will take about a minute for the startup process to finish.

7.2. Configuration

The backend screen allows selection of the one MHz filterbanks to be in the following modes:

One 2GHz wide FFBA
Two 1 GHz wide FFBA, FFBB
Four 512 MHz wide FFBA-FFBD
Eight 256 MHz wide FFBA-FFBH

The 250KHz may be set to the following modes:

One 128 MHz wide FB2A
Two 64 MHz wide FB2A, FB2B

Each may also be disabled.

7.3. Calibration

The zero and gain values for each channel must be determined from a calibration run before taking real data. The calibration run switches the input to no signal for zero and to hot and cold loads for gain determination.

Two or three iterations of gain and zero calibration may be needed to initialize the system after power-on. The zero adjustment has been seen to have temperature dependence, but stabilizes after an hour of operation.

7.4. Taking Data

7.5. Shutdown sequence

The filterbank uses a lot of power, so it is sensible to shut it off if the telescope will not be used for a week or more. The majority of power use is in the filter crates, so they could be turned off for shorter times of disuse to reduce the system's power consumption.

First, stop the ffb_accum and ffb_control tasks on the main Rambo control screen. Turn off the five filter crates. Each has a switch on the power supply. The left side switch controls power to the filter crate above the supply, and the right hand switch controls power to the crate below the supply. The right rack's power supply only uses the upper (left switch) section.

Turn off the three IF processors next. Each has a switch on the front panel; down is off.

The digital crate is the next item to turn off. Its power switch is located inside the leftmost rack, accessible from the rear, near the top of the rack at the right side as viewed from the rear of the rack.

Shut down the PC next. First, pull open the keyboard drawer below the LCD monitor and login as root, then type the command "shutdown -h now". Wait a minute for the shutdown procedure to complete as shown on the LCD screen, then turn off the power switch. The PC, located in the left rack, has two rocker switches inside the front panel, which is guarded with a key. Push the rocker switch with the power symbol on it once to turn off the computer.

Unplug the power cord to the clock distribution amplifier, located above the PC. Unplug the power cord to the FieldPoint temperature monitoring system inside the center rack.

The racks, filter crates and power supply boxes all have cooling fans that run continuously. Unplug these units from the power strips so that all fans stop turning.

8. Maintenance

8.1. Scheduled maintenance

Clean the filters on the front of each power supply box once a month. They are to be cleaned by unsnapping the plastic cover, pulling the filter media away from the fan, taking it to the sink, rinsing in cold water, squeezing dry then blotting dry with paper towels, reinserting over the fan and reinstalling the plastic cover frame.

8.2. Unscheduled Maintenance

Power supply

Each power supply is capable of powering two filter crates. The right rack power supply only has one crate to power; it is the spare unit in case a power supply in one of the other units fails.

To remove a power supply for service:

Turn off the power switch.

Wait for a minute for the box to cool down.

Clear a spot 2 feet wide on the workbench to hold the supply.

Unplug the power cord.

Unplug the large round connectors to the filter crates.

Unplug the small round connector to the temperature monitor.

Unscrew the two black rackmounting screws at each end of the front panel.

Get a second person to help - the power supply weighs about 70 lbs.

If the supply will not come out easily, then remove the filler panel below it.

Pull the supply out of the rack partway, one person on each side.

Hold the supply from underneath and remove it the rest of the way.

Carry it to the workbench.

The power supply units are standard 5V open-frame linear supplies.

Installation is the reverse of removal. The top cover is held on by six sheet metal screws threaded into soft aluminum - do not overtighten them.

IF processor

The relays can go bad. A bad relay requires desoldering, which requires removal of the IF processor unit from the rack.

Filter cards

The filter cards are each a 6U Eurocard with components on both sides and two DIN connectors. The P1 DIN connector has two RF coaxial inserts that can cause trouble when inserting the card.

When inserting a filter card, press the card into the slot by pushing on the center of the panel. If it doesn't seat all the way, then try again at a slightly different angle. Don't force the card - it will go in with patience.

Converter cards

The use of plug-in cards greatly reduces the maintenance burden by making all the active circuitry easily replaceable. The question then is how to tell the operator which card to replace based on a certain symptom.

A look-up chart relates channel group failures to cards. Since the converter cards each affect a certain number of channels, the size of the failed group indicates where to look for the failure.

Chan	Chan	Chan	Chan	Chan	CON2	CON3	CON4	Filter
0	512	1024	1536	2048	0A	0A	0A	0A
16	528	1040	1552	2064	0A	0A	1A	2A
32	544	1056	1568	2080	0A	0A	2A	4A
48	560	1072	1584	2096	0A	0A	3A	6A
64	576	1088	1600	2112	0A	1A	0B	0B
80	592	1104	1616	2128	0A	1A	1B	2B
96	608	1120	1632	2144	0A	1A	2B	4B
112	624	1136	1648	2160	0A	1A	3B	6B
128	640	1152	1664	2176	1A	2A	4A	8A
144	656	1168	1680	2192	1A	2A	5A	10A
160	672	1184	1696	2208	1A	2A	6A	12A
176	688	1200	1712	2224	1A	2A	7A	14A
192	704	1216	1728	2240	1A	3A	4B	8B
208	720	1232	1744	2256	1A	3A	5B	10B
224	736	1248	1760	2272	1A	3A	6B	12B
240	752	1264	1776	2288	1A	3A	7B	14B
256	768	1280	1792	2304	0B	0B	0C	1A
272	784	1296	1808	2320	0B	0B	1C	3A
288	800	1312	1824	2336	0B	0B	2C	5A
304	816	1328	1840	2352	0B	0B	3C	7A
320	832	1344	1856	2368	0B	1B	0D	1B
336	848	1360	1872	2384	0B	1B	1D	3B
352	864	1376	1888	2400	0B	1B	2D	5B
368	880	1392	1904	2416	0B	1B	3D	7B
384	896	1408	1920	2432	1B	2B	4C	9A
400	912	1424	1936	2448	1B	2B	5C	11A
416	928	1440	1952	2464	1B	2B	6C	13A
432	944	1456	1968	2480	1B	2B	7C	15A
448	960	1472	1984	2496	1B	3B	4D	9B
464	976	1488	2000	2512	1B	3B	5D	11B
480	992	1504	2016	2528	1B	3B	6D	13B
496	1008	1520	2032	2544	1B	3B	7D	15B

The channel number groups each include the next 16 channels. The CONx card numbers are identified above by the picture of the slots as shown below.

CON4 CON4	CON3 CON2 CON3	CON4 CON4	CON4 CON4	CON3 CON2 CON3	CON4 CON4
0 1	0 0 1	2 3	4 4	2 1 3	6 7

The filter card slots are 0 on the left, 15 on the right, in ascending order. The FBMUX card is in the center slot between the two groups of eight filter cards.

The chart above is broken down into groups of channels, each of which are affected by a single card. The size of the failure in channels is related to cards according to the following list:

block	what to suspect
=====	=====
512	FBINT card, FBMUX card, filter crate power or cables
256	IF processor section or IF cables to filter crate
128	FBCON2 card channel or cables from splitters to backplane
64	FBCON3 card channel
16	FBCON4 card channel or FBFIL32 card half
1	FBFIL32 card channel

Each converter card has several channels: the FBCON2 and FBCON3 have two IF channels, and the FBCON4 has four. The filter cards have two sections of 16 channels each.

To determine if a filter card or an FBCON4 cards is at fault for a 16-channel failure, swap two filter cards to see if the problem moves. If so, it was the filter card. If not, swap the two FBCON4 cards of the same type to see if the problem moves.

One desirable feature of maintainability is providing the ability to easily replace cards in the system without going through a lengthy rebooting process. This is achieved by making the power to each crate independently switched and designing the interface ports so that either end can be powered down without causing latchup or other undesired behavior.

The software has been written to allow the system to gracefully recover from a power cycle on any filter crate or IF processor. The initialization is performed by stopping and restarting the ffb_control and ffb_accum programs from the telescope control computer.

9. Performance

9.1 Overview

The filterbank has been evaluated for filter bandwidth, linearity and stability. The results follow, with an analysis of the results.

9.2. Stability

Observers desire stable backends to provide sky-limited stability of observations. The sky is assumed to be stable in the 1mm band for about 10 seconds maximum. however, on-the-fly mapping and position switching may be done with up to 30 seconds between reference scans.

The Allan variance test is done by observing with longer and longer Dicke switching time, plotting the RMS deviation of the signal. The plot assumes a V shape with a rounded bottom. The bottom of the left leg of the V is the interesting part.

The theoretical reduction in noise with increasing integration time is $1/\sqrt{t}$. System instability causes the signal to drift over time, which shows up as increased noise. Over a long enough period, the system drift overwhelms the noise being observed.

We chose an arbitrary definition for Allan variance time as the time at which one has to integrate for 10% longer than $1/\sqrt{t}$ to obtain the equivalent noise reduction.

The filterbank has been tested with a stabilized 5 GHz noise source to measure its Allan variance time, which is a measure of the maximum observation time before instability degrades the reciprocity of integration over time.

The filterbank system has also been fitted with a 16 channel National Instruments FieldPoint temperature measurement system with Linux logging software. This system is able to measure the temperature of any one of 16 RTD sensors to 0.01C resolution once every two seconds.

The testing area was the office in room 172 of Steward Observatory building. The air source for this room is the central HVAC system. The temperature logging equipment showed that the room temperature dropped by about 3 degrees C at 11PM every night and rebounded at 6AM every morning. This change inadvertently provided us with a temperature step function with which to evaluate the stability of the filterbank system.

A set of CLASS macros has been written to allow easy evaluation of the Allan variance of the filterbank system over time and channels. The first plot from a set of data is shown as Allan variance time versus channel number. This gives a quick indication both of the stability of the system and of which channels or groups of channels are working best and worst.

Another useful plot is that of RMS signal level versus time, averaged over some group of channels. This shows the drift of each part of the system. These plots can be overlaid on temperature plots taken from the temperature logging system to show the temperature dependence of the signal level.

The result of this testing on the initial system configuration showed us that the noise generator was responsible for half of the signal's temperature dependence, about 2% per degree C. The 2 GHz amplifiers in the IF processors were responsible for another 1% per degree C, while the filter cards and downconverters are responsible for one last percent per degree C.

The time constant of the signal level changes is a function of the packaging. The noise source and IF processors exhibit a slow response to air temperature changes, on the order of 20 minutes time constant.

The filter crates, on the other hand, showed a much faster response to room air temperature changes. It is a bit difficult to extract the time constant from the signal vs temperature plots, but it is apparent that there is a component of the signal level that responds nearly instantaneously to room air changes. This was demonstrated by opening the back door to a filter rack and looking at the resultant signal level plot.

The above rapid signal level change is not good for stability during an observation. It is desired to keep the time constant of all signal-affecting drift to be on the order of minutes, not seconds.

To this end, a method to increase the time constant of the signal level from the filter crates has been established and implemented. The problem was that the room air was used directly to cool the cards, so any change in the room air temperature was immediately reflected in the signal level. The only way to remove this dependence was to isolate the air moving across the filter cards from the room air.

The filter crates have been reworked to remove the power supplies to a separate box and to circulate air through the cards but not directly to the room. The heat is removed by transferring to the sheet metal of the filter crate and from there to the room air. This approach has been implemented on the filterbank system, and it resulted in a doubling of the average Allan variance time from about 12 seconds to about 25 seconds.

10. Hardware info (pinouts, backplane signal names)

10.1. Converter card keying

The FBCONx converter cards use both mechanical and electrical keying to ensure the right card is in the right slot. The mechanical keying uses the IEEE 1101.10 keying blocks installed behind the lower panel handle and in the bottom card guide in the filter crate.

Each card has two or three keying blocks installed. The rotation of the keying blocks is a function of card type and frequency. The keying code per the IEEE specification is as follows:

Looking into crate:

Chamber	Key
=====	===
D=left	1=left
E=middle	2=top
F=right	3=right
round hole	4=bottom

On rear of card front panel:

Chamber	Key
=====	===
round peg	1=left
F=left	2=bottom
E=middle	3=right
D=right	4=top

A table of keying for each type of card (- means no key installed):

Card	D	E	F
====	=	=	=
CON2	3	2	2
CON3-1	3	3	2
CON3-2	3	3	3
CON4-1	3	4	2
CON4-2	3	4	3
CON4-3	3	4	4
CON4-4	3	4	1
CON4-Q1	3	1	2
CON4-Q2	3	1	3
CON4-Q3	3	1	4
CON4-Q4	3	1	1
FIL32	2	2	-
FIL32-Q	2	3	-
MUX	2	1	-
EPP	4	1	1
INT	4	2	2

TIM	4	3	2
IFCTL	4	4	2

10.2. Filter crate backplane

The filter crate video backplane pinout is shown below.

pin	A	B	C
1	Pos5VD	Pos5VD	Pos5VD
2	Gnd	Gnd	Gnd
3	-	Gnd	-
4	-	Gnd	-
5	-	Gnd	-
6	Gnd	-	Gnd
7	-	Gnd	-
8	-	Gnd	Chan0
9	-	Gnd	Chan1
10	-	Gnd	Chan2
11	-	Gnd	Chan3
12	-	Gnd	Chan4
13	-	Gnd	Chan5
14	-	Gnd	-
15	-	Gnd	Qmhz
16	-	Gnd	Dump
17	-	Gnd	Int
18	Gnd	Gnd	Gnd
19	Vid0/Gnd	Gnd	Vid15/Gnd
20	Vid1/Gnd	Gnd	Vid14/Gnd
21	Vid2/Gnd	Gnd	Vid13/Gnd
22	Vid3/Gnd	Gnd	Vid12/Gnd
23	Vid4/Gnd	Gnd	Vid11/Gnd
24	Vid5/Gnd	Gnd	Vid10/Gnd
25	Vid6/Gnd	Gnd	Vid9/Gnd
26	Vid7/Gnd	Gnd	Vid8/Gnd
27	Gnd	Vid	Gnd
28	Neg12V	Gnd	Pos12V
29	Gnd	Gnd	Gnd
30	Neg5VA	Neg5VA	Neg5VA
31	Gnd	Gnd	Gnd
32	Pos5VA	Pos5VA	Pos5VA

Signals Chan0-5 select the channel on the filter card.

Vid0-Vid15 are the analog video signals to the Multiplexer card, while Vid is the video output from each filter card.

Int and Dump when high enable the integration and dumping, respectively, of the channel integrators.

The filter crate backplane has some keying info also. The signal qmhz (active low), formerly known as sp3 (spare #3) and on pin C15, is now to be tied to Gnd to define a 250 kHz resolution filter crate, or left floating to define a 1 MHz filter crate.

This signal is passed through the Mux card (with inversion) to the converter cards, which use it to compare their ID code to that of the filter crate to be sure

they're in the right place. The red LED is lit if the card is in an incompatible slot.

10.3. Filter cards

The order of the filter channels is scrambled to keep adjacent frequency channels physically separate. Here is the descrambling code:

chan	designator
0	_1
1	_5
2	_9
3	_13
4	_3
5	_7
6	_11
7	_15
8	_2
9	_6
10	_10
11	_14
12	_4
13	_8
14	_12
15	_16

```
designator = ((chan & 8) >> 3) | ((chan & 4) >> 1) | ((chan & 3) << 2);
```

10.4. PC interface parallel port cable

The host PC interface uses a standard EPP style IEEE-1284 parallel port cable with a type C connector. The standard EPP signals are used. The pinout:

FBEPP driving IEEE1284-C
Cable

pin	name	name	pin	source
1	Wait	Gnd	19	out
2	--	Gnd	20	--
3	Int	Gnd	21	out
4	--	Gnd	22	--
5	--	Gnd	23	--
6	D0	Gnd	24	bi-dir
7	D1	Gnd	25	bi-dir
8	D2	Gnd	26	bi-dir
9	D3	Gnd	27	bi-dir
10	D4	Gnd	28	bi-dir
11	D5	Gnd	29	bi-dir
12	D6	Gnd	30	bi-dir
13	D7	Gnd	31	bi-dir
14	Init	Gnd	32	in
15	Write	Gnd	33	in
16	AS	Gnd	34	in
17	DS	Gnd	35	in
18	--	--	36	--

Note that the spare signals are not used.

Wait is the acknowledge signal from the FBEPP to the host.

Int is the interrupt from the FBEPP to the host. It pulses low for a couple microseconds to tell the host that a new frame of data is available.

D0-D7 are the 8-bit bidirectional data bus. The direction is ostensibly controlled by the Write line, although some PC interfaces fail to play by this rule. Thus the data bus is driven with some interesting timing - see the FBEPP firmware for details.

Init is an active-low reset signal from the PC to the FBEPP card.

Write is the direction control line from the PC. It is high for reading, low for writing.

AS is the address strobe, indicating when low that the EPP register address is present on the data bus. It is interlocked with Wait, and is used only with write cycles in this application.

DS is the data strobe, indicating when low that data may be read or written. It is interlocked with Wait.

10.5. Video cable

The video cable connects the filter crate to the FBINT card. It has 18 twisted pairs, some of which are differential.

The video cable pinout is shown below.

pin	name	name	pin	source
1	Error	Gnd	19	filter
2	Video+	Video-	20	filter
3	A0	Gnd	21	host
4	A1	Gnd	22	host
5	A2	Gnd	23	host
6	D0	Gnd	24	bi-dir
7	D1	Gnd	25	bi-dir
8	D2	Gnd	26	bi-dir
9	D3	Gnd	27	bi-dir
10	D4	Gnd	28	bi-dir
11	D5	Gnd	29	bi-dir
12	D6	Gnd	30	bi-dir
13	D7	Gnd	31	bi-dir
14	Int+	Int-	32	host
15	Dump+	Dump-	33	host
16	Write+	Write-	34	host
17	DS+	DS-	35	host
18	HostHi	FilHi	36	both

The Error signal is TTL and is an inclusive OR of the SErr signals from the converter and filter card busses. This signal is used during configuration to detect missing or wrong cards, and during operation to detect a PLL out-of-lock condition.

The Video signal pair is analog differential, with signal levels of +/-0V for minimum signal and +/-2.5V for maximum signal.

The D0-D7 signals are the TTL data bus. This bus contains 0 when scanning and data when not scanning. It is used to reset the channel number to 0 at the start of the scan; 0 coincidentally is the first channel number.

The A0-2 signals are TTL and select which port to access. These signals are used to select the various registers on the Multiplexer card.

The Int+/- and Dump+/- pairs are RS-422. They enable the integration and charge dump functions of the channel integrators, respectively, when asserted.

The Write+/- pair is RS-422. This signal controls the direction of the interface. It is normally asserted since writing is the normal mode of operation.

The DS+/- pair is RS-422. An assertion edge on DS writes new data into the appropriate register on the multiplexer card or increments the channel as required by the addressed register.

HostHi is driven high by the integrator card and sensed by the multiplexer card. It tells the multiplexer card to ignore the cable pins when the digital crate power is off.

FilHi is driven high by the multiplexer card and sensed by the integrator card. It indicates that the filter crate is powered up and connected properly.

10.6. Digital crate backplane

The digital backplane pin list is shown below.

pin	A	B	C
1	Pos5VD	Pos5VD	Pos5VD
2	Gnd	Gnd	Gnd
3	Clock/Gnd	Gnd	Clock/Gnd
4	Clock/Gnd	Gnd	Clock/Gnd
5	Clock/Gnd	Gnd	Clock/Gnd
6	Gnd	Clock	Gnd
7	Write	Gnd	DS
8	A0	Gnd	A1
9	A2	Gnd	A3
10	sp1	Gnd	sp2
11	D0	Gnd	D1
12	D2	Gnd	D3
13	D4	Gnd	D5
14	D6	Gnd	D7
15	sp3	Gnd	sp4
16	Scan	Gnd	Dump
17	Flip	Gnd	Int
18	Gnd	Gnd	Gnd
19	Sel10/Gnd	Gnd	Sel15/Gnd
20	Sel11/Gnd	Gnd	Sel14/Gnd
21	Sel12/Gnd	Gnd	Sel13/Gnd
22	Sel13/Gnd	Gnd	Sel12/Gnd
23	Sel14/Gnd	Gnd	Sel11/Gnd
24	Sel15/Gnd	Gnd	Sel10/Gnd
25	Sel16/Gnd	Gnd	Sel9/Gnd
26	Sel17/Gnd	Gnd	Sel8/Gnd
27	Gnd	Sel	Gnd
28	Neg12V	Gnd	Pos12V
29	Gnd	Gnd	Gnd
30	Neg5VA	Neg5VA	Neg5VA
31	Gnd	Gnd	Gnd
32	Pos5VA	Pos5VA	Pos5VA

11. Driver Software

The following section describes the programming of the filterbank system from a driver perspective. This information is provided for reference, as the ffb driver for Linux is already written.

11.1. Parallel port

The digital crate communicates with the PC via a standard parallel port. There are three different types of parallel port protocols in existence: SPP, EPP and ECP. The BIOS on the PC has to be configured to match the EPP protocol used by the filterbank system.

The required BIOS settings are as follows:

Base addr: 0x378
Mode: EPP ver 1.9
Interrupt: Yes, periodic

The interface cable from the PC to the EPP card is an IEEE1284 A-C style. The type C connector is a fine-pitch 36 pin connector with a pinout optimized for high speed operation. The type A connector is the standard DB-25 connector found on nearly every PC ever manufactured.

The digital crate's parallel port connector resides on the EPP card. This card buffers the parallel port signals, resyncs them to the 10 MHz clock, and generates synchronous data cycles on the digital backplane.

The standard EPP parallel port consists of eight data lines, an address strobe /AS, a data strobe /DS, a /WRITE line, a WAIT line, an IRQ line, and a /RESET line. This format dictates the sequence of access cycles used to communicate with the filterbank system.

The port behaves much like a multiplexed microprocessor bus such as the 8085. The /AS signal indicates that a port address is present on the data lines. The /DS signal indicates that data may be read or written. /WRITE specifies whether a read or write cycle is taking place. WAIT is used to handshake with the host computer - it's normally low, but is brought high to acknowledge /DS or /AS falling, and is brought low to acknowledge /AS or /DS rising.

Since a parallel port has only 8 data lines, larger data words are split up by hardware on each end into multiple 8-bit transfer cycles. The super I/O chip in the PC automatically does this by virtue of its four-byte data port. The Timing and Int cards do the data funneling work in the digital crate. Access of any size (8, 16, 24 or 32 bits) may be performed.

The low-level EPP access sequence consists of writing an address, then reading or writing one or more bytes of data.

Backplane addressing scheme

Since each of the cards may respond to the EPP port accesses, a tri-state bus is used. Each card in the digital crate has a 16-byte address range. There may be up

to 16 cards in the digital crate. The cards are geographically addressed, 16 bytes of address space per card. Thus, card 0 is at 0x00, card 1 at 0x10, etc.

Each type of card has a different ID code returned by its ID register. The Linux ffb device driver header file ffb.h defines which card is supposed to be installed in which slot. The driver uses this information when initializing the system to detect and report missing or wrongly-installed cards.

The cards are assigned locations in the cardcage according to type:

slot	type
0-6	Int
7	Timing
mid	EPP
8-11	Ifctl

11.2. EPP card

This card converts the EPP parallel port on the front panel connector to decoded addresses and synchronous read and write cycles on the digital backplane. This simplifies the circuitry on the other cards and keeps bus noise down. a Xilinx CPLD contains the logic.

A front panel BNC connector receives the 10 MHz station clock which is converted to TTL level and distributed to all other cards on the digital backplane in a star network. The input clock is expected to be a sine wave of approximately +10dBm level, although a lower level will work.

An onboard oscillator was used for prototype/test systems when no 10 MHz station clock is available. The oscillator has since been removed from the cards to reduce RFI.

The backplane has the following signals:

Clock	10 MHz clock for data transfer timing and device clocks
Select0-15	16 separate board select lines, one per slot, active low
Address0-3	4 bit address bus for register select within each board
Write	Read when high, write when low
Strobe	Low-going pulse, one clock wide, signals data transfer
Data	8 bit bidirectional data bus

All bussed backplane signals except Clock are terminated to 3V at each end of the backplane with 330/470 ohm pullup/pulldown resistor networks.

The bus is synchronous. Signals are sampled on the rising edge of Clock. The bus clock distribution is designed so that all cards receive Clock at about the same time. This is done by routing Clock on the backplane in a branching tree rather than the usual bussed format. Trust me, I've done this before.

The EPP address forms the board select from bits 7-4 and the address bus from bits 3-0. This allows for up to 16 cards, each of which may have up to sixteen registers. An on-chip EPP address register returns the address when read.

The address bus and select lines are updated only upon an EPP address write cycle, so they are stable for several cycles before and after each data transfer. The previous board select is negated several clock cycles before the next is asserted. This means that board select may be used as a buffer enable for the Ack line.

Board select is also negated whenever the address is written, so it may be used to reset the data pointer in the multi-byte data transfer state machines.

Read/Write is low to write, high to read. It is used to select the transfer direction. It also is gated with BdSel to enable the data bus drivers.

Strobe is one clock wide and does the jobs of writing the write data and incrementing the byte pointer and word pointer (if RAM). It does not cause the data to become available; the read data are available two clocks after BdSel and Write are valid. Write the data or update the read data in the clock cycle following that in which DS is true.

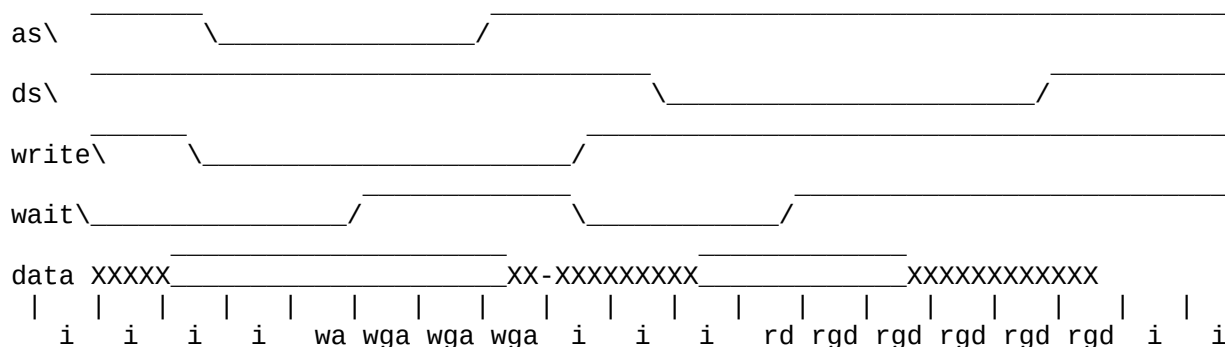
The data bus is driven by the selected board continuously if in read mode. In write mode, the data bus is driven by the host interface. Wait for one clock cycle after Write rises before enabling data to prevent bus contention.

The data transfers are EPP style; that is, they are 8-bit sequential byte transfers. Each card performs word-to-byte funneling if needed. The ADC cards may use 3-byte data words if desired.

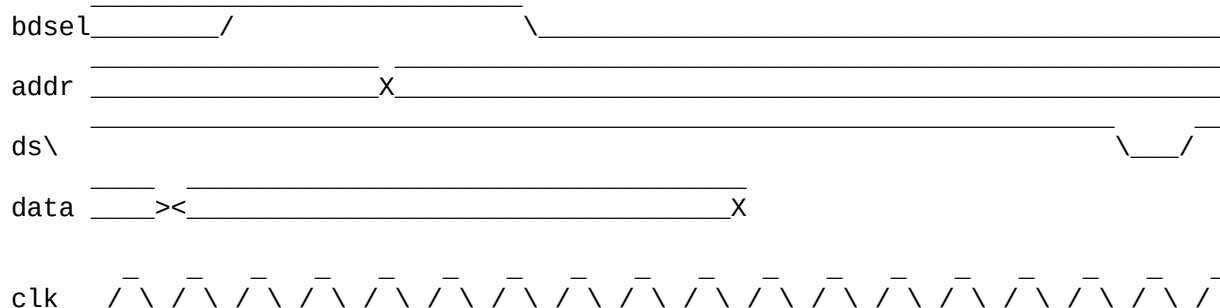
An empty slot returns the bus quiescent data value (0xFF). It is up to the PC software to read each card's ID register to ensure it's correct before using the data.

Read cycle

EPP:



Backplane:



11.3. Int card

The Int card performs the tasks of controlling the filter crate, scanning through the channels, converting the incoming video signal to digital form, integrating the digital data over several analog integration periods, and buffering and presenting the data to the host computer as requested.

The Int card contains a programmable gate array (FPGA) with integral RAM to store the datasets and correction values per channel. The FPGA contains control and status registers as well.

RAM

The Xilinx FPGA contains 40kbits of SRAM, which is split up into three sections:

- * Two 512 x 24 bit data buffers
- * A 512 x 12 bit zero correction table
- * A 512 x 14 bit gain correction table

A single address pointer is shared for all three memory sections. The byte pointer is implied; the address pointer is a 16 bit register that may be read or written. The addressing is done by channel number and may contain values from 0..511 inclusive. The top seven bits of this register are not implemented, but forward-looking programming practice requests that they be loaded with zeroes at all times.

The LSB is read or written first; MSB is last. This is in accordance with the x86 processor family's little-endian byte ordering.

The RAM byte pointer provides a means of reading or writing multiple bytes of data in a single RAM location. It is reset every time the EPP address is written. During a sequential access of a single RAM, it cycles through the bytes of each RAM word as needed per that RAM's width. The driver code performs the multiple byte reads/writes as needed.

There is no way to view the byte pointer; you must trust it to work correctly. To achieve this, do not allow multiple code threads to use the RAM access since that could allow one thread to break up access to a data word being read by a different thread.

Filter crate interface

This card also provides the interface to the filter crate's mux card. This interface is similar to the backplane bus except that it doesn't pass the 10 MHz clock. Instead, the rising edge of DS strobes the data into or out of the output card. This results in low noise transmission to the filter crate.

Registers 0xx8..0xxB in each Int and I/O card in the digital crate are reserved for use by the crate port. These are used by the filter crate for data scanning, offset DACs, PLL and attenuator control.

The crate port is controlled by the integrator when INT_CTL_SCAN is 1. Otherwise it's controlled by the host PC. When INT_CTL_SCAN is 1, crate addr and data are set to 0.

NOTE: The above limitation means that the filter crate control port is INACCESSIBLE while the filterbank is active! To change a filter crate setting, it is necessary to first disable scanning on the Int card temporarily.

Register set:

regname	addr	size	wr?	value
=====	=====	=====	=====	=====
INT_ID	0xx0	8 bit	R	0x08
INT_STAT	0xx1	8 bit	R	
INT_CTL	0xx2	8 bit	R/W	
INT_PTR	0xx3	16 bit	R/W	
INT_SUM3	0xx4	24 bit	R/W	contents of sum RAM in 24 bits
INT_SUM2	0xx5	16 bit	R/W	contents of sum RAM in 16 bits
INT_ZERO	0xx6	16 bit	R/W	contents of zero RAM
INT_GAIN	0xx7	16 bit	R/W	contents of gain RAM

The following registers exist on the filter crate output card:

INT_CARD	0xx8	8 bit	W	load/read filter card number
INT_CHAN	0xx9	8 bit	W	load/read filter channel number
INT_INCR	0xxA	8 bit	W	inc filter channel (ignores data)
INT_CFG	0xxB	8 bit	W	Load config register with serial data

The following provide manual control of the ADC and DAC:

INT_ADC	0xxC	16 bit	R	ADC data with continuous conversion
INT_DAC	0xxD	16 bit	R/W	DAC data for manual control
BNC_CTL	0xxE	8 bit	R/W	BNC connector control

Detailed register descriptions

INT_ID 0xx0 8 bit R 0x08

This read-only register contains the board type code 0x08 which indicates an Int card. It is used at initialization time to verify that the correct type of card is installed in each slot.

INT_STAT 0xx1 8 bit R

This 8-bit read-only register contains the board status. Its four active bits are defined below:

bit	name	description
===	=====	=====
0	filt_on	1 = filter crate is powered up
4	blank_in	1 = blank BNC input is high (5V)
5	sync_in	1 = sync BNC input is high (5V)
7	filt_err	1 = filter crate reports an error

Bit 0 indicates that the filter crate has power. Bit 7 indicates an error condition on one or more cards in the filter crate. The error control registers on the cards may be used to localize the error.

Bits 4 and 5 report the status of the two BNC input bits. These may be used for local blanking or for general-purpose digital input bits as specified by the BNC_CTL register.

INT_CTL 0xx2 8 bit R/W

This 8-bit read/write register contains the board control bits as defined below.

bit	name	description
===	=====	=====
0	INT_CTL_SCAN	enables scanning and conversion process
1	INT_CTL_GAIN	enables gain control; DAC set to INT_DAC if 0
2	INT_CTL_ZERO	enables zero correction; no zero correction if 0
3	INT_CTL_CONV	ADC manual conversion control
4-7	-	not defined; set to 0 for upward compatibility

INT_PTR 0xx3 16 bit R/W

This 16-bit read/write register is the RAM address pointer. Only 9 bits are used; all 16 bits of the register exist but the top 7 bits do not control anything.

INT_PTR is used to specify which channel in the RAM is being accessed. This register increments automatically when the last byte (as defined by the width of the RAM data register) of a RAM location is accessed.

INT_SUM3 0xx4 24 bit R/W contents of sum RAM in 24 bits

This read-write register allows access to the filter channel data in 24-bit mode. The first access to this register after setting INT_PTR reads or writes the LSB of data for that channel; the next access reads or writes the middle byte, and the third access reads or writes the MSB of data. The following access reads or writes the LSB of the subsequent channel, etc.

INT_SUM2 0xx5 16 bit R/W contents of sum RAM in 16 bits

This read-write register allows access to the filter channel data in 16-bit mode. The first access to this register after setting INT_PTR reads or writes the LSB of data for that channel, the next access reads or writes the middle byte of data. The following access reads or writes the LSB of the subsequent channel, etc. The top data byte is not available from this register.

INT_ZERO 0xx6 16 bit R/W contents of zero RAM

This read-write register allows access to the zero correction data. The first access to this register after setting INT_PTR reads or writes the LSB of data for that channel, the next access reads or writes the MSB of data. The following access reads or writes the LSB of the subsequent channel, etc.

INT_GAIN 0xx7 16 bit R/W contents of gain RAM

This read-write register allows access to the gain correction data. The first access to this register after setting INT_PTR reads or writes the LSB of data for that channel, the next access reads or writes the MSB of data. The following access reads or writes the LSB of the subsequent channel, etc.

The following registers exist on the filter crate Mux card:

INT_CARD 0xx8 8 bit W load/read filter card number

This write-only register loads the filter crate with the number of the card to be read from next. Only the bottom 4 bits of this register are used; the top 4 bits do nothing.

INT_CHAN 0xx9 8 bit W load/read filter channel number

This write-only register loads the filter crate with the channel number to be read from next. Only the bottom 6 bits of this register are used; the top 2 bits do nothing. Note that there are 32 channels on a card; bit 5 selects the test channels (total power, reference levels) when 1.

INT_INCR 0xxA 8 bit W inc filter channel (ignores data)

This write-only register causes the channel number to increment. It is not normally used manually; the scanning logic uses this register to step through the 512 channels when automatically reading the filter data in Scan mode.

The increment performed does not ever set bit 5 of the filter channel number, so the test channels cannot be read using this function.

INT_CFG 0xxB 8 bit W Load config register with serial data

This write-only register writes to the configuration register in the Mux card to allow loading of the various DACs, PLLs and attenuators in the filter crate. A complete description of the configuration function is provided in chapter 8.

INT_ADC 0xxC 16 bit R ADC data with continuous conversion

This read-only register allows the A/D converter to be read for test purposes.

INT_DAC 0xxD 16 bit R/W DAC data for manual control

This read-write register allows the DAC to be set to a constant value when not using the zero correction function.

BNC_CTL 0xxE 8 bit R/W BNC connector control

This register is used to control the function of the two BNC connectors. These connectors may drive signals to the outside world, or they may be used for local blanking purposes. Each connector has a direction control bit, a polarity bit, an output level bit and an enable bit.

bit	name	description
===	====	=====
0	BLANK_DIR_OUT	set blank BNC to output mode when 1
1	SYNC_DIR_OUT	set sync BNC to output mode when 1
2	BLANK_LBLANK_EN	enables lblank from blank BNC when 1
3	SYNC_LBLANK_EN	enables lblank from sync BNC when 1
4	BLANK_LBLANK_ACT	sets blank BNC polarity to lblank-when-high when 1
5	SYNC_LBLANK_ACT	sets sync BNC polarity to lblank-when-high when 1
6	BLANK_LEVEL_OUT	In output mode, drive blank BNC to 5V when 1
7	SYNC_LEVEL_OUT	In output mode, drive sync BNC to 5V when 1

Output mode is useful for controlling external devices such as the comb generator.

The Lblank inputs are used to discard data when the receiver is out of lock, the telescope is out of focus, etc. LBlank tells the timing generator to halt the integration, send the analog data collected so far, and suspend operation until LBlank is rescinded.

11.4. Timing card

The Timing card (physically just a differently-stuffed Int card) controls the timing of the filterbank system. It runs from the 10 MHz clock provided by the EPP card and generates the analog integration timing signals Int, Scan, Dump and Flip. These signals control the analog and digital integrators.

All of the times written to registers are in units of integer microseconds. For example, 25.0 milliseconds has the integer value 25000.

The Blank signal from the telescope informs that the telescope is not producing a usable antenna signal when high. The rising edge of Blank is used to disable the analog integrators and start the readout of the digital integrators to the host PC. Blank is typically high for about 30 milliseconds.

The Sync input indicates the observing phase on the falling edge of Blank. True means phase 0, false means go to the next phase. If only two phases, then Sync indicates directly (but inverted) which phase you're in. The observing phase is contained in the TIM_PHASE register for use by the driver program.

A typical subreflector chopping rate is 2 hertz, which is four phases per second for a blank period of 250 ms with an unblanked time of 210 ms. Fastest is 4 hertz which is 85 ms unblanked time.

If the observations do not require observing phases, then Blank must never be asserted.

The LBlank backplane signal is wire-ORed and asserted by one or more INT cards. It means that the current integration is bad, so should be aborted and the data transmitted. Then the timing state machine waits for LBlank to be rescinded and continues. blank during this time will advance the observation phase as usual.

The maximum analog integration period is TIM_INT, nominally 25 milliseconds. It will be shorter if Blank rises in the middle of a frame.

The actual integration time of a dataset is totalized in TIM_INTTIM for data reduction purposes.

The dataset is transferred to the PC at two times:

- 1) just after scan following rising edge of Blank or LBlank, and
- 2) just after scan in which analog frame count = max_frames_reg.

The analog integration period is divided into the following states:

State	Period	Description
====	=====	=====
Int	25.0 ms	Integrate detected signals in analog integrators
Scan	513 us	Scan the filter channels and coadd each ADC value to RAM
Flip	1 us	Switch to other bank if needed
Dump	100 us	Discharge the integrator capacitors
Blank	ext ctl	If blanked, wait patiently for blanking time to end

Scan time is the number of filter channels plus one.

Flip (bank switch) causes an EPP interrupt to the PC host so that it may read in the dataset. It also causes the digital integrator sums to be reset to 0.

Register set:

regname	addr	size	wr?	value
=====	=====	=====	=====	=====
TIM_ID	0x80	8 bit	R0	0x0C
TIM_CTL	0x82	8 bit	R/W	0x00
TIM_INTTIM	0x84	32 bit	R0	
TIM_PHASE	0x85	8 bit	R0	
TIM_INT	0x88	16 bit	R/W	integration period in microseconds
TIM_SCAN	0x89	16 bit	R/W	data readout scan period in microseconds
TIM_DUMP	0x8A	16 bit	R/W	integrator discharge period in microseconds
TIM_FIM	0x8B	16 bit	R/W	max analog frames integrated per dataset

TIM_CTL bit definitions:

bit	name	description
===	=====	=====
0	TIM_CTL_INT	manual int output: 1=int, used only when TIM_CTL_AUTO = 0
1	TIM_CTL_SCAN	manual scan output: 1=scan, used only when TIM_CTL_AUTO = 0
2	TIM_CTL_DUMP	manual dump output: 1=dump, used only when TIM_CTL_AUTO = 0
3	TIM_CTL_FLIP	manual scan output: 1=scan, used only when TIM_CTL_AUTO = 0
4-6	-	not defined; set to 0 for upward compatibility
7	TIM_CTL_AUTO	enables timing generator when 1; otherwise use bits above

D) Interrupt

The EPP interrupt line is used to send an interrupt pulse to the computer once per read bank. This interrupt specifies when the driver code is to read the next frame of data from the FPGA.

The interrupt pulse is high for one microsecond and corresponds to the Flip signal on the backplane. This signal causes the Int card to swap the two data buffers so that the one just filled may be read, and the one just read may be filled with new data.

11.5. Filter crate control

The filter crate requires configuration to set the converter LO frequencies, attenuators and filter zero settings. This configuration is done via two serial links accessible through the Int card.

The filter crate is controlled by four write-only registers on its Int card. These registers allow configuration of the filter and downconverter cards' various programmable bits. They also provide some control of the filter data scanning circuitry.

The filter crate bus has an error readback bit to allow configuration testing and PLL error detection. This bit is bit 7 of the Stat register of the fbint card. Reading 0 is OK; 1 is error.

The converter and filter cards provide control over the error bit for error localization. Each card has an error control register to allow the error bit to be disabled, forced true, or set to indicate bad-slot or PLL out-of-lock errors.

Scanning Sequence

The sequence of the filter channels with respect to filter cards is scrambled in order to accommodate the IF downconverter frequency map. The result is that the FBMUX card scans the card and channel numbers in a scrambled way to provide linear frequency scanning of channels. The bit scrambling is as follows, where filt() is the filter channel, Chan() is the channel on a filter card, and Card() is which filter card:

```

Chan(0) = filt(0)
Chan(1) = filt(1)
Chan(2) = filt(2)
Chan(3) = filt(3)
Card(1) = filt(4)
Card(2) = filt(5)
Chan(4) = filt(6)
Card(3) = filt(7)
Card(0) = filt(8)
Chan(5) = filt(9) (test channels; not used)

```

The filter crate is controlled over a parallel cable whose pinout is specially defined to provide a low noise data path for the control signals as well as for the analog filter signal. As a result, the control method is a bit unusual.

The Int card has four registers assigned to the filter crate. These registers are write-only and provide a means to initialize the converter PLLs and filter card DACs, and to select the filter channel being sampled.

NOTE: This set of registers may only be accessed by the PC when Scan mode is disabled (TIM_CTL_AUTO is zero). When Scan is 1, the automatic sequencing logic on the Int card scans all 512 channels once per integration/scan period.

Filter crate registers:

adr	name	description
===	====	=====

0	Card	loads card number (4 bits)
1	Chan	loads channel number (6 bits)
2	Incr	increments channel/card number (no data)
3	Cfg	loads configuration word shown below

The first three registers, Card, Chan and Scan, are used by the filter data scanning logic to read out the 512 filter channels in sequence. The Card and Chan registers are used to reset the Mux card to point to the first channel of the first card. Thereafter, the Scan register is written once per channel to increment the channel number being read. The Card number increments after 32 channels on a card have been scanned out.

The fourth register, Cfg, allows loading of DACs, PLLs and attenuators using the bit-banging method of serial port access. This register provides access to two different serial data busses - one for the filter cards, the other for the converter cards. Each bus is controlled by four bits of the register as shown:

Cfg register bits:

bit	name	description
===	====	=====
0	DSclk	DAC serial clock
1	DSEn	DAC serial enable
2	DSdat	DAC serial data
3	DSadr	DAC serial address mode
4	ISclk	IF serial clock
5	ISEn	IF serial enable
6	ISdat	IF serial data
7	ISadr	IF serial address mode

The filter backplane is fed from the DAC serial bus consisting of DSclk, DSEn, DSdat and DSadr, and the converter cards are fed from the IF serial bus consisting of ISclk, ISEn, ISdat and ISadr.

Each serial bus is an SPI bus with an additional mode bit to allow serial addressing of devices on the bus. The xSadr bit selects address mode when 1 or data mode when 0.

When address mode is selected, the serial bus will accept an 8-bit address which selects one of 256 serial devices on the bus. These are split into 16 devices per card, with up to 16 cards on a bus. (The serial busses are physically duplicated to place only 8 cards on a set of serial control lines, but that is hidden from the user.) The slot is the 4 LSBs of the address, and the specific device on the card is selected by the 4 LSBs.

Each device on the serial bus is written to by issuing a sequence of commands. The sequence is to set address mode by setting xSadr, write an 8-bit address (MSB first) using xSclk and xSdat, load the address by pulsing xSEn, then switching to data mode by clearing xSadr and clocking serial data to the device MSB first with xSclk and xSdat, then using xSEn to load the data into the device. The details of using xSEn are different for different devices, so those are explained in the text below.

The filter card address bits are arranged in a scrambled order to make the filter channels numbers come out straight in spite of the convoluted converter arrangement. The address bit translation from filter channel number to DAC

addresses is shown below, where filt() is the filter channel, Dac() is an octal DAC chip, DacCh() is the DAC in a DAC chip, and Card() is which filter card:

```
Dac(0)   = filt(0)
Dac(1)   = filt(1)
DacCh(1) = filt(2)
DacCh(0) = filt(3)
Card(1)   = filt(4)
Card(2)   = filt(5)
DacCh(2) = !filt(6)
Card(3)   = filt(7)
Card(0)   = filt(8)
```

Based on the above and the register table for a filter card, the DAC Address map is:

```
ssss1000    DAC 0
ssss1001    DAC 1
ssss1010    DAC 2
ssss1011    DAC 3
ssss1100    DAC TP
7548HL10    <- channel number (H is always hi, L is always low)
```

Channels within DAC chip:

```
 $\overline{623}$     <- channel number
```

Translation of card select bits based on channel number:

card bit	chan bit
0	8
1	4
2	5
3	7

Each DAC is loaded by clocking 11 bits to it while it is selected. Selecting a DAC is done by setting DSEn to 1, writing 11 bits, then setting DSEn to 0. Each bit is written by loading the data into the DSDat bit, then setting DSClk to 1, then setting DSClk to 0.

11.6. Converter cards

Each converter card has a local oscillator controlled by an Analog Devices ADF4116/8 PLL chip. The PLL chip must be programmed before the converter is usable.

The converter card PLL addressing and frequency chart is as follows:

addr	1MHz		250kHz	
	card	freq	card	freq
====	====	====	====	====
0x02	CON4-4	194.0	CON4-Q4	111.0
0x12	CON4-3	178.0	CON4-Q3	107.0
0x22	CON4-2	162.0	CON4-Q2	103.0
0x32	CON4-1	146.0	CON4-Q1	99.0
0x42	CON3-1	596.0	CON3-2	612.0
0x52	CON3-2	660.0	CON3-2	628.0
0x62	CON2	2478.0	CON2	2466.0
0x82	CON4-4	194.0	CON4-Q4	111.0
0x92	CON4-3	178.0	CON4-Q3	107.0
0xa2	CON4-2	162.0	CON4-Q2	103.0
0xb2	CON4-1	146.0	CON4-Q1	99.0
0xc2	CON3-1	596.0	CON3-2	612.0
0xd2	CON3-2	660.0	CON3-2	628.0
0xe2	CON2	2350.0	CON2	2434.0

The PLL chip is programmed by writing three bytes to each of three PLL registers. The PLL register select bits are contained within one of the three bytes. The PLL registers are described in the ADF4116/4118 data sheet. The details of PLL programming are contained in the ffbllib.c source file.

Each PLL is loaded by sending three bytes of eight bits each to it, MSB first. Each bit is written by loading the data MSB first into the ISDat bit, then setting ISClk to 1, then setting ISClk to 0. Repeat the above sequence for three bytes, then set ISEn to 1 then to 0 to load the PLL register. The two LSBs of the three-byte word specify which PLL register will receive the data.

Converter Attenuators

Each Fourth Converter has a set of attenuators in addition to a PLL. The attenuators are used to control the gain of each group of 16 channels in 1 dB steps from 0 to 31 dB of attenuation.

Fourth Converter Address map:

ssss0000	--	00 is safe
ssss0001	Error	
ssss0010	PLL	
ssss0011	--	
ssss0100	Att A	
ssss0101	Att B	
ssss0110	Att C	
ssss0111	Att D	
ssss1xxx	--	FF is safe

Each attenuator takes an 8-bit binary number, of which the 5 LSBs specify the attenuation in integer dB to apply to its group of 16 channels. The range is 0 (no attenuation) to 31 (31dB attenuation). The three high bits are ignored.

Error Register

The filter and converter cards each have an error register to allow the system to determine the source of any errors. The types of errors provided for currently are:

- * Missing card
- * Wrong card in slot
- * Filter crate resolution ID (configuration, not an error)
- * PLL out-of-lock (converters only)

The FB_ERROR bit on each int card (STAT bit 7) allows the software to read one error bit per crate. The error bit is a wired-OR function of all error bits from individual cards in the crate, so any single error will indicate a bad crate. The error control registers on each converter card allow us to learn which card is at fault via a process of elimination.

The error control register has the following bits:

bit	name	description	
===	====	=====	
0	Err_Force	1=Force ISerr error	0>manual error off
1	Err_Slot	1=slot error causes ISerr	0=slot error off
2	Err_PLL	1=PLL error causes ISerr	0=PLL error off
3	LED_force	1=Force red LED on	0>manual LED off
4	--	unused, set to 0	
5	LED_PLL	1=PLL error causes red LED	0=PLL LED off
6	Err_Qmhz	1=250K ID causes ISerr	0=ID error off
7	--	unused, set to 0	

The basic method of use is to first clear all error enable bits, then activate one bit on one board at a time to check the board's status. As each board in the system is interrogated, an inventory of good, bad and missing cards is created.

The Qmhz bit allows detection of the filter crate's resolution. This information is read back from the analog backplane where it is hard-wired according to the type of filter cards used in that crate. It may be read from any of the FBCON4 cards' error registers.

Once the configuration is checked, the converter boards are set to monitor the PLL lock. Any out-of-lock PLL will cause the error line to be activated, which will alert the driver software to a problem.

The converter cards themselves have red and green LEDs to indicate error conditions. The red ERROR LED on each converter card is lit whenever the card is inserted into an incompatible slot. It may also be lit when the PLL is out of lock if bit 5 is set to 1. The green LED is lit if no enabled error condition is true.

The error registers are configured by hardware to report no errors at power-on time. They are reprogrammed by the driver to report slot errors while doing the board inventory, then after the PLLs are initialized they are set to report PLL errors while running.

Board inventory

The converter cards exist in several different flavors, so it was considered necessary to provide a means for the computer to locate any mis-installed or non-functioning converter cards and alert the operator to the problem. This is done by inventory software that executes when the FFB control program is started.

Each filter crate is initialized in sequence. Before the PLLs are loaded with their frequencies, the converter cards are all checked for by the following algorithm.

The algorithm for doing a board inventory is:

Set all error control to CON_ERR_NONE.

For each board:

 Set error to CON_ERR_FORCE and check for error.

 Set error to CON_ERR_NONE and check for no error.

 Set error to CON_ERR_SLOT and check for no error.

 Set error to CON_ERR_QMHZ and save error status as crate resolution.

As the crate is scanned, a list of card errors is built. The wrong-slot cards are reported in the log file.

The following error codes are returned from the slot scanner:

```
/* slot scanner error codes */
#define SLOT_ERR_OK_1M          0 /* Correct card in slot, 1M resolution */
#define SLOT_ERR_OK_250K       1 /* Correct card, 250k resolution */
#define SLOT_ERR_STUCK         -1 /* Error bus stuck on */
#define SLOT_ERR_NOCARD        -2 /* No card in slot */
#define SLOT_ERR_BADCARD       -3 /* Wrong card in slot */
#define SLOT_ERR_NOLOCK        -4 /* PLL out of lock */
#define SLOT_ERR_UNUSED        -5 /* non-existent slot */
```

PLL errors

Now each board needs its PLL programmed. after the PLL is programmed and allowed a second to obtain lock, its error control is set to CON_ERR_PLL | CON_LED_PLL to monitor PLL lock.

11.7. IFctl cards

The IFctl card is a specially stuffed version of an Int card that provides control over an IF processor. The IFctl card communicates with a single IF processor over a parallel cable, in the same way that the Int card communicates with a filter crate. The IF processor has a corresponding I/O card (FBIFC) that receives commands and returns status to the IFctl card.

The following devices are controlled:

LED	1-16	front panel LEDs (not synth lock LEDs)
Synth	1-4	four LO synthesizers, Micro Lambda MLSL-0608
Relay	1-8	up to eight IF steering relays
Switch	1-8	room for 8 switches (not currently used)

An optional RF detector connector provides capability to measure power levels or voltages. This feature uses the ADC on the IFctl card, but is not currently implemented.

The address map of the IFctl card is:

adr	write	read	description
---	-----	----	-----
0	sdet	--	selects detector with bits 0-2
1	wrel	rrel	sets/reads indicators of relays 1-8 on bits 0-7
2	wled0	--	writes to LEDs 1-8 on bits 0-7
3	wled1	--	writes to LEDs 9-16 on bits 0-7
4	wsyns	rsyns	synth select / lock status
5	wsynd	rsynd	writes data byte to selected synth(s), reads Busy status on bit 7
6	wsdir	rsdir	dir of switch 1-8 on bits 0-7: 1=outputs, 0=inputs
7	wsdat	rsdat	read/write data to/from switch 1-8 on bits 0-7

wsyns/rsyns register:

bit	name	R/W	description
---	----	---	-----
0	sel1	RW	1 => synth 1 selected to receive data
1	sel2	RW	1 => synth 2 selected to receive data
2	sel3	RW	1 => synth 3 selected to receive data
3	sel4	RW	1 => synth 4 selected to receive data
4	lock1	R	1 = synth 1 is locked
5	lock2	R	1 = synth 2 is locked
6	lock3	R	1 = synth 3 is locked
7	lock4	R	1 = synth 4 is locked

The synth state machine on the fbifc card writes the byte of data (slowly) to the selected synth(s). The timing is 500 microseconds between edges, a bad speed for Linux, so the timing is done in hardware. The four select lines allow the same data to go to all synths if desired, or just one.

Data are sent by the byte, MSB first. Select the synth(s), write a byte of data, wait for busy false, then repeat for each byte. Finally, deselect all synths.

The LED wiring of the IF boxes is as follows:

1 MHz box

name	port	bit	description
=====	=====	=====	=====
LED5	LED0	0x10	test
LED6	LED0	0x20	zero
LED9	LED1	0x01	IF A
LED10	LED1	0x02	IF B
LED11	LED1	0x04	DS
LEDS_1M_ZERO	0x0020		
LEDS_1M_TEST	0x0010		
LEDS_1M_IF_A	0x0100		
LEDS_1M_IF_B	0x0200		
LEDS_1M_DS	0x0400		

250K/Chirp box

name	port	bit	description
=====	=====	=====	=====
LED5	LED0	0x10	test
LED6	LED0	0x20	zero
LED9	LED1	0x01	IF A FB 1
LED10	LED1	0x02	IF A FB 2
LED11	LED1	0x04	IF A Chirp 1
LED12	LED1	0x08	IF A Chirp 2
LED13	LED1	0x10	IF B FB 1
LED14	LED1	0x20	IF B FB 2
LED15	LED1	0x40	IF B Chirp 1
LED16	LED1	0x80	IF B Chirp 2

LED1-4 are not used by software - they are hardwired to display the synth PLL lock status.

wrel/rrel register:

This register controls the relays that select the signal source. It has one bit per relay to set the source, and one bit per relay to read back what the relay thinks it is set to.

The relays are of the pulse latching type with mechanical auto-turn-off, so they require a cycle on startup to get synchronized to the computer.

The readback is done through the indicator contacts on the relay, so it shows the position the armature is in.

The sources are:

DS1-4	DesertStar - four pixels per IF processor
IF A	JT receiver channel A
IF B	JT receiver channel B
Test	noise/comb source
Zero	no signal (for cal.)

The relay wiring of the 1 MHz IF boxes is as follows:

relay	bit	sig	0	1	
=====	====	====	=	=	
RY1	0x01	IF1	DS	IFABTZ	Selects DesertStar or IFA/IFB/test/zero
RY2	0x02	IF2	DS	IFABTZ	
RY3	0x04	IF3	DS	IFABTZ	
RY4	0x08	IF4	DS	IFABTZ	
RY5	0x10	TZ	test	zero	selects between test and zero
RY6	0x20	IFABTZ	IFAB	TZ	selects either IF A/B or test/zero
RY7	0x40	IFAB	IFB	IFA	selects IF A or B

Note that selection of A/B/test/zero inputs must be done as a group; it's not possible to feed IF A to two IF outputs while the other two look at the test signal.

The relay wiring of the 250KHz/Chirp IF boxes is as follows:

relay	bit	sig	0	1
=====	====	====	=	=
RY1	0x01	IF1	IFA	IFBTZ
RY2	0x02	IF2	IFA	IFBTZ
RY3	0x04	IF3	IFA	IFBTZ
RY4	0x08	IF4	IFA	IFBTZ
RY5	0x10	TZ	test	zero
RY6	0x20	IFBTZ	IFB	TZ

There is more flexibility here in that IF A/B selection can occur separately for each of four IF outputs.

The enable_if / disable_if functions remember which IF A/B/DS was selected so that it can put things back when exiting zero mode.

10-28-05 LED & Relay control software

The goal is to allow easy application control of the LEDs and relays for selecting the IF sources in both the 1 MHz and 250k IF processors. These two processors have different control abilities - the 250k allows selection of IF A vs. B on a per-input basis, while the 1M have a single selection that steers all four inputs to the same IF channel.

The 250K box is set up with the first two channels as the 250K filters, and the last two channels as the two CTS instruments, of which only one currently exists. The first channel is defined by bit 0, the last by bit 3 of the source select code, IF A = 0, IF B = 1.

Defines for the possible inputs for each type of IF processor: five possible inputs for the 1M IF processor, and 64 for the 250K box:

```
#define SOURCE_1M_IF_A      0
#define SOURCE_1M_IF_B      1
#define SOURCE_1M_DSTAR     2
```

The 250K rack has four bits to select source of each of 4 spectrometers:

```
#define SOURCE_FBQ1_A      0x00
#define SOURCE_FBQ1_B      0x01
```

```
#define SOURCE_FBQ2_A    0x00
#define SOURCE_FBQ2_B    0x02
#define SOURCE_CTS1_A    0x00
#define SOURCE_CTS1_B    0x04
#define SOURCE_CTS2_A    0x00
#define SOURCE_CTS2_B    0x08
```

Common control codes for test/zero:

```
#define SOURCE_TEST      0x10
#define SOURCE_ZERO      0x20
```

These allow the relay and LED code to set their output bits via some arrays.

11.8. Taking Data

The filterbank computer uses the ffb driver program to take data and control the filterbank hardware. This program is rather involved, but basically provides four functions:

- open
- close
- ioctl
- read

The open and close functions work like any Linux device. Open does basic work to create a file descriptor, but does no interaction with the hardware. This is done to allow the PC to reboot without requiring the digital crate to be powered up.

The close function simply relinquishes the file descriptor.

The ioctl function provides all of the filterbank control functions. since it's difficult to pass arrays through ioctl, a single 32 bit word is used for parameters. The needed parameters are bit-shifted to all fit in the 32 bits provided. Status or readback data is returned by the ioctl function.

The read function causes the filterbank to wait for a digital integration time, then returns a structure with the data array and some status and operating information. This function will hang if the filterbank system is not working correctly.

12. Application Software

12.1. Introduction

The Filterbank captures spectral line data from one to eight incoming 5000 MHz IF input signals up to 2GHz wide, and converts the data to usable digital form, passing it through an Ethernet interface to an external data logging computer.

The PC is a Pentium computer running the Linux operating system with the RTLinux real-time extension. The PC reads groups of spectral data from the integrator into a RAM data buffer using a device driver called /dev/ffb. This device driver talks through the PC's parallel port in EPP mode. It contains special functions to control the filterbank.

The filterbank requires some software to make it work. The user interface has not been well-defined in the past, owing to its roots in the half-baked AOS control software. That is being remedied with the ffb system.

Software structure

The software consists of the following files:

ffb.c	Linux filterbank device driver program
ffb.h	driver headers, used by all
ffblib.c	filterbank application library routines
ffblib.h	application library prototypes
fbcal_utils.c	initialization and calibration routines
fbcal_utils.h	initialization and calibration prototypes
filter_init.c	low-level init routines (used only by ffb.c)
filter_init.h	low-level init prototypes (used only by ffb.c)
fbcal.c	simple initialization and calibration program
spec.c	simple spectrum display application
allan.c	simple Allan variance application
sweep.c	channel bandpass filter alignment program (for maintenance use)

12.2. Driver

The filterbank driver /dev/fffb is a character device driver that permits multiple users. It provides four functions: `fopen()`, `fclose()`, `read()` and `ioctl()`.

```
device_fp = fopen(device_name, "r+");
```

The `read` function is presently blocking. It is recommended to use a timeout when calling `read()` so that it will return with an error code if the filterbank is not sending data, for example.

Each call to `read()` returns one integrated frame of data for each active filterbank crate, along with a timestamp and operational parameters.

12.3. Data structure

Each read function call returns a structure containing the spectral data as well as the timestamp and the filterbank interface register set.

Refer to the file ffb.h for the data structure variable names.

The data structure ffb_data_t is described below.

```
typedef struct FFB_DATA {
    struct timespec tms;           /* Time stamp */
    hrtime_t rtm;                 /* Read time in ns */
    tim_reg_t tim;               /* the timing registers */
    int raw[INTS*MAX_DATA_LENGTH]; /* The raw data */
} ffb_data_t;
```

tms is a standard RTLinux timespec indicating when the interrupt was received for this frame. This corresponds to the end of the integration period.

rtm is the length of time in nanoseconds it took to read the data into the driver.

tim is the set of registers from the timing board. Read about these in the filterbank interface hardware manual; the structure is shown below.

```
typedef struct {
    unsigned short int_tim;      /* integration time in us of this dataset */
    unsigned short ob_phase;     /* observation phase 0..n */
    unsigned short int_per;      /* integration period per analog scan */
    unsigned short scan_per;     /* data scan period (channels/crate + 1) */
    unsigned short dump_per;     /* analog integrator reset time in us */
    unsigned short frame_max;    /* max # of analog scans per dataset */
} tim_reg_t;
```

raw is an array which contains the spectral data. NCHANNELS defines the size of the filterbank in channels. Presently NCHANNELS is 2048, but eventually it will be up to 4096. INTS is the max number of filter crates (8) but presently only 3 are used.

Note that MAX_DATA_LENGTH is longer than the 512 word dataset by 64 words. This is provided as a safety margin for the ffb driver, which writes extra data (a fixed length with a maximum of 64 words) to the array. This needs fixing!!!

The raw data is only spectral data; the AOS interface stores other stuff here. The filterbank writes the other (timing) info in the timing register set.

Filterbank structure

The filterbank system consists of a digital crate and several filter crates. Each filter crate has two 256-channel sections, and connects to the digital crate with a single integrator card. Currently, four filter crates with 512 channels of 1 MHz resolution filters exist. A fifth filter crate with 512 channels of 250KHz resolution filters is under construction.

An IF processor is provided for each rack of 1024 channels of filters. The third IF processor feeds the future 250KHz filter crate as well as two chirp transform spectrometers (CTS).

An emulation of the telescope's blank and sync signals is generated by the "fake telescope" test unit. This allows the blanking feature to be tested when the telescope is not operational, and was built for use downtown. When on the mountain, the real telescope is typically used to generate these signals via the Smbus.

The entire filterbank is addressed by channel number 0..2047. The ffblib routines convert channel number to the hardware addresses as needed.

Different control functions operate on different channel multiples. These are:

Zero DACs	1 channel	channel
Attenuators	16 channels	card
Center freq	256 channels	section
Data on/off	512 channels	crate
IF on/off	1024 channels	rack

12.4. fbcal_utils

I have defined an interface consisting of a library of subroutines that control the system. These may be broken up into the following divisions:

System inventory
 Initialization
 IF control
 Calibration
 Data acquisition

fbcal.c calls various routines in fbcal_utils.c to do its good work.

There are two different tasks: initialization and calibration.

```
printf("Initializing timing card.\n");
/* load timing board registers with proper values */
set_timing(ffb, TIM_SLOT, INT_USEC, (CRATE_CHANS + 1), DUMP_USEC, FIM);
select_source(ffb, 0, RELAYS_1M_TEST);
select_source(ffb, 1, RELAYS_1M_TEST);

/* initialize the filter crates */
for (crate = 0; crate <INTS; crate++) {
    status = ffb_init(ffb, crate);
    if (status != 0)
        exit(1);
}
```

The IF processors each contain four downconverters that need to be set to the IF frequency that each filterbank section will be monitoring. A function call is provided to set one or more of these LO synthesizers. Here's a typical usage that sets up the synthesizers to a 1024 MHz wide band centered at 5000 MHz.

```
/* initialize the IF controllers */
for (ifc=0; ifc<2; ifc++) {
    for (synth=0; synth<4; synth++) {
        freq = 1928 + (5000 - 384) + (synth * 256);
        printf("Initializing synth %d on IFC %d to %d MHz\n", synth, ifc, freq);
        load_synth(ffb, (IFCTL_SLOT + ifc), freq, (1 << synth));
    }
}
```

```
/* *** Square law detector zero calibration *** */
ffb_zero_all(ffb);
```

```
/* *** Attenuator calibration *** */
ffb_atten_cal_all(ffb);
```

```
/* *** Gain normalization *** */
ffb_normalize_gain_all(ffb);
```

```

/* *** zero offset correction *** */
ffbb_zero_finish_all(ffbb, INT_USEC);

fclose(device_fp);

return 0;
}

```

/* fbcal.c */

```

/*
This code initializes and calibrates the filterbank.
The filterbank device is opened and loaded with operating parameters.
The attenuators on the converters are set to a reasonable level.
The detector zero levels are set for each channel.
The A/D converter gain is normalized across all channels.
The zero level for each channel is measured and nulled out.

```

fbcal_utils.c

This file contains two functions to initialize the filterbank and four functions to calibrate the system. Initialization need be done only after powering up any part of the system. Calibration should be done whenever the input selection or frequency are changed.

Initialization:

char * ffbb_detect(int fp)

This function tests for the presence of all hardware that is specified in ffbb.h. It checks to see that the digital crate is connected and powered up. Then it looks for the timing card and for the right number of INT cards. Any error results in a string being returned identifying the error encountered. Success returns NULL.

int ffbb_init(int fp, int crate)

This function initializes a single INT card. It stops any data transfer. then it loads the PLLs in the converter cards to the correct frequencies as listed in ffblib.h. The automatic gain control and zeroing are disabled. The BNC connectors are set to inputs to drive LBLank but disabled.

/* Do zero calibration for one filter crate */

int ffbb_zero_all(int fp)

```

/* *** Attenuator calibration *** */
int ffbb_atten_cal_all(int fp)
{
    /* This code enables the IF and looks at each group of 16 channels

```

```

    to find the highest signal. It then adjusts the attenuator for that
    group to meet the desired top input level. */

/* set_atten and get_atten must be called with data stopped!*/

/* first, set to max attenuation for guaranteed low levels */

/* *** Gain normalization *** */
int ffb_normalize_gain_all(int fp)
{
    /* This is done by reading the difference between IF on and IF off of each
       channel. The channel's gain correction DAC is then set inversely to
       the measured gain to normalize them all. */

/* *** zero offset correction *** */
int ffb_zero_finish_all(int fp, int integ_time)
{
    /* This is done after the gain correction in order to get accurate zero
       measurement, as the zero level is changed by the gain correction.
       Zero setting is scaled by a fudge factor equal to the integration time.*/

```

System inventory

The code is compiled for a fixed number of channels. The constants in ffb.h define the system size. Therefore, all the code must be recompiled when a chunk of filters is added or removed.

The ffb driver does not know how many channels are installed; it must be told via the variable ffb_dev.channels so that it knows how many channels to scan.

System inventory is the task of checking that all required cards are installed and functioning properly. Library functions are provided to ease the task.

The first task is checking all the timing, int and ifctl cards for presence.

The timing slot is in TIM_SLOT which is currently 7.
 The int cards start at slot 0 and go up from there by 2^{INT_SHIFT}.
 The ifctl cards start at IFCTL_SLOT and go up from there.

Filterbank initialization consists of setting up the timing parameters for the integrators, loading the attenuators with their initial values, and performing a zero/gain calibration.

Calibration consists of four steps:

1. Zeroing each channel's square law detector,
2. Setting the attenuators for a usable signal level,
3. Setting the gain correction factor for each channel, and
4. Setting the zero correction value for each channel.

Data acquisition consists of reading data from the filters repeatedly. The system appears to be stable enough that periodic internal calibration is not needed.

Initialization is done with the `ffb_init` function call.

The PLL initialization is done with the `init_all_plls` function call.

Calibration is currently done by the `fbcal` program. It is set up to use the test signal and on/off control of the IF processors.

The filter card DAC zero-setting procedure is done in `fbcal`.

Zero offset and gain correction

The filterbank system provides a means of calibrating the offset and gain. There are two different zeroing mechanisms and two different gain setting mechanisms.

Each channel has an offset value to zero the A/D converter and a gain multiplier setting to set the A/D gain. These two sets of values are stored in the A/D card. These settings possess long-term stability after the system has reached operating temperature.

In addition, each channel has a DAC on the filter card to zero the square-law detector output. This must be done once after the system has reached stable operating temperature. The system is stable enough that periodic zeroing of the filter card DACs should not be necessary.

Each group of 16 channels has an IF attenuator setting to allow control of the IF gain with 1dB resolution. This provides a means to roughly level the sensitivity of the entire system over the input frequency range.

The calibration routine provided in the program `fbcal.c` is sufficient to calibrate gain and offset of the filterbank system. The fake telescope is used to switch the IF on and off to perform the calibration.

12.5. ffbllib

The ffb driver provides a set of ioctl calls that are made easy-to-use by the application library ffbllib.c. These calls do useful work such as setting the timing, controlling the IF processors, setting attenuation levels and daking calibration data.

The set of library functions available in ffbllib.c include:

```
void disable_if(int fd, int rack);
void enable_if(int fd, int rack);
void select_source(int fd, int rack, int source);

void set_gain_manual(int fd, int crate, int gain);
void disable_gain(int fd, int crate);
void enable_gain(int fd, int crate);
void disable_zero(int fd, int crate);
void enable_zero(int fd, int crate);

void clear_fifo(int fd);
int  stop_data(int fd, int crate);
int  start_data(int fd, int crate);
int  read_one(int fd, int data[NCHANNELS], ffb_data_t data_frame);

void load_dac(int fd, int chan, int value);
void load_dacs(int fd, int crate, int value);
void set_atten(int fd, int chan, int level);
int  get_atten(int fd, int chan);

void set_timing(int fd, int slot, int int_period, int scan_period,
                int dump_period, int frame_max);
void get_timing(int fd, int slot, int int_period, int scan_period,
                int dump_period, int frame_max);

void init_all_plls(int fd, int crate);
void test_all_cons(int fd, int crate, int slot_codes[N_CONVS]);
void set_all_lock_dets(int fd, int crate);

void load_synth(int fd, int slot, int frequency, int enables);
void set_relays(int fd, int slot, int value);
int  get_relays(int fd, int slot);
void set_leds(int fd, int slot, int value);
int  get_leds(int fd, int slot);
int  get_switch(int fd, int slot, int device);

void init_lblank(int fd, int crate);
void set_lblank_en(int fd, int value);
void set_lblank_pol(int fd, int value);
int  get_lblank_en(int fd);
int  get_lblank_pol(int fd);
```

The common parameters are:

fd file descriptor for /dev/ffb

crate filter crate number (0..7)
rack rack number (0..3), corresponding to IF processor
chan filter channel number (0..4095)
slot slot number of the card
int_period time period for integration in microseconds
scan_period time period for reading out data (set to NCHANNELS+1 for now)
dump_period time period for discharging the integration capacitors in microseconds
frame_max maximum number of frames to accumulate per dataset

Each function is described below.

void disable_if(int fd, int rack);

Disables the IF signal from reaching the filterbank. This is used for zero calibration. This function controls the IF processor. It also saves the selected source so that enable_if can restore it properly.

void enable_if(int fd, int rack);

Enables the IF signal to reach the filterbank. Must be done to set up the IF processor for normal operation.

void select_source(int fd, int rack, int source);

Sets the IF processor in rack to look at the IF source as defined in ffb.h. Possible sources are DesertStar, IF A or B, test and zero.

void disable_gain(int fd, int crate);

Disables the gain normalization hardware. Used for calibration only.

void enable_gain(int fd, int crate);

Enables gain normalization. This is the normal operating mode.

void disable_zero(int fd, int crate);

Disables the zero correction logic. Used for calibration only.

void enable_zero(int fd, int crate);

Enables the zero correction logic. Normal state of system.

void clear_fifo(int fd);

Wipes out all stale data from the driver program's data buffer. Call this to get fresh data on the next read.

int stop_data(int fd, int crate);

Causes this crate to stop collecting data. Does not clear FIFO. Returns success flag for debugging since there have been reports of this call not working.

int start_data(int fd, int crate);

Causes this crate to start collecting data. Does not clear FIFO. Returns success flag for debugging since there have been reports of this call not working.

int read_one(int fd, int data[NCHANNELS], ffb_data_t data_frame);

Reads a few frames of data and normalizes it to 1e6. Used for calibration.

void load_dac(int fd, int chan, int value);

Loads one square law detector zero correction DAC. Used in calibration.

void load_dacs(int fd, int value);

Loads NCHANNELS worth of square law detector zero correction DACs with the same value. Used for calibration.

void set_atten(int fd, int chan, int level);

Sets the attenuator that affects channels chan..chan+15 to level dB of attenuation. This function only works after a stop_data call!

int get_atten(int fd, int chan);

Reads back the attenuator that affects channels chan..chan+15. This function only works after a stop_data call!

**void set_timing(int fd, int slot, int int_period, int scan_period,
int dump_period, int frame_max);**

sets the system timing to the values specified (in microseconds). Used for initialization and reconfiguration of the system.

**void get_timing(int fd, int slot, int int_period, int scan_period,
int dump_period, int frame_max);**

reads back the system timing parameters. Useful in conjunction with set_timing to change one parameter at a time.

void set_gain_manual(int fd, int crate, int gain);

sets the gain correction logic to manual mode with the gain specified (0..4095).

void init_all_plls(int fd, int crate);

loads all PLLs in a crate with the appropriate frequencies. Not for users.

```
void test_all_cons(int fd, int crate, int slot_codes[N_CONVS]);
```

tests all converter cards for presence. Not for users.

```
void set_all_lock_dets(int fd, int crate);
```

Sets the lock detect error indicators on each converter card in this crate, so that lock failures are reported both on the front panel and in the error bit of the Int card.

```
void load_synth(int fd, int slot, int frequency, int enables);
```

Loads one to four synthesizers with LO frequency specified in integer megahertz. The LO frequency specified must be equal to the desired IF frequency (in the range 4000-6000 MHz) plus 1928 MHz, which is the center frequency of the filter section. for example, to set the IF frequency to 5000 MHz, specify 6928 for frequency.

enables is a four-bit code where bit 0 is the first filter section, bit 1 is the second, etc. Setting enables to 0xf loads all four synthesizers in a rack with the same frequency.

slot = rack + IFCTLS.

```
void set_relays(int fd, int slot, int value);
```

Sets all relays in an IF processor to the 8-bit value specified. This is an internal function used by select_source; don't use it externally.

```
int get_relays(int fd, int slot);
```

Returns the relay settings previously written to an IF processor. This is an internal function used by select_source; don't use it externally.

```
void set_leds(int fd, int slot, int value);
```

Sets the LED array to the specified 16 bit value. 1 is lit, 0 is dark.

```
int get_leds(int fd, int slot);
```

Reads back the 16 bit value of the LEDs in the IF processor.

```
int get_switch(int fd, int slot, int device);
```

Reads back a single switch from the IF processor. There are currently no switches implemented, so this function has no purpose at this time.

```
void init_lblank(int fd, int crate);
```

Sets up both BNC connectors on this crate's INT card as inputs to the LBlank function. Done by the ffb_init routine.

```
void set_lblank_en(int fd, int value);
```

Sets all of the LBlank enables (up to 16) to value. The argument value has one bit per input. Bit 0 is LBlank 0 which is the SYNC BNC connector of the leftmost INT card, bit 1 is BLANK on that card, bit 2 is the SYNC connector of the second INT card, etc.

A bit set to 0 disables (ignores) its LBlank input, while a bit set to 1 allows that LBlank input to halt data-taking.

void set_lblank_pol(int fd, int value);

Sets the polarity of all of the LBlank signals. 1 causes blanking when high, 0 causes blanking when low. Bits are numbered as above.

int get_lblank_en(int fd);

Reads back the current LBlank enable settings for all bits. See set_lblank_en for format.

int get_lblank_pol(int fd);

Reads back the current LBlank polarity settings for all bits. See set_lblank_pol for format.

12.6. Configuration

The system is designed to be auto-configuring. However, there are some things that need to be done at initialization time. These are taken care of in fbcal.

Specifically, the integration and dump times must be set, as well as FIM (the maximum number of frames per dataset).

The IF processor requires setting to the desired IF distribution mode. A function call is provided to do this.

12.7. Blanking Considerations

The filterbank is disabled when global blanking is in effect. This allows it to start integrating again immediately after blanking ends. This is nicer than the AOS, which must keep its CCD clocking out continuously.

Two local blanking signals enter on each of the INT cards (four cards for a 2048 channel system). They are individually optionally inverted, enabled, and ORed together to make a common Local Blank signal that drives the timing card.

The timing card then uses this signal to pause and stop taking data while blanked. The data from the frame that got blanked is discarded!?!!!

Let's think about this:

The data taken so far is valid or not?

Data taking resumes in the same frame after Local Blank becomes false again.

Local blank takes priority over global blank.

12.8. Initialization

The filter cards and converter cards have programmable parts that require initialization before they will work properly. The filter cards have zero correction DACs and the converter cards have PLLs and attenuators.

The fbcsl program does the required initialization.

Filter card DACs

Each channel has a zero setting DAC to set the square law detector to zero output at zero signal. The zero setting must be done AFTER the system has reached operating temperature, as it is temperature dependent.

The zero setting routine must not set the zero point to exactly zero but instead should set the filter's output to a few hundred counts (out of 16384) with zero signal. This gives some 'wiggle room' to accommodate drift in the zero set point.

Advanced Info

The EPP port intrinsically communicates with a register space of 256 8-bit registers. This register space has been divided up into 16 cards of 16 registers each.

There are three types of cards in the filterbank digital crate:

Timing
Integrator
I/O

The Timing card controls the filterbank data acquisition timing. It contains registers to start and stop signal integration and to set the integration and discharge (dump) periods. It also provides a running total of the integration time seen by the filter card analog integrators.

The Integrator card connects to 512 channels of filterbank. It controls the filterbank operating parameters and buffers the data stream. Its register set contains control registers, data buffers, correction RAM, and access to the detector offset on the filter cards and the attenuator and PLL controls on the IF downconverter cards.

The I/O card is as yet undefined. It will provide control of the IF front-end processor when that part is designed and built.

ioctl

The ioctl commands provide a means of reading the driver parameters, setting the offset and gain correction values and reading or writing the EPP registers directly.

This call resets the filterbank interface to its power-up condition.

```
ioctl(fd, FFB_IOC_RESET);
```

This call clears the data FIFO so that fresh data will be returned.

```
ioctl(fd, FFB_IOC_FIFO_CLEAR);
```

These calls set/read the EPP port address for direct register access.

```
ioctl(fd, FFB_IOC_EPP_ADDRESS_WR, val);  
ioctl(fd, FFB_IOC_EPP_ADDRESS_RD, &val);
```

These calls read/write one 8 bit EPP data register.

```
ioctl(fd, FFB_IOC_EPP_DATA8_WR, val);  
ioctl(fd, FFB_IOC_EPP_DATA8_RD, &val);
```

These calls read/write one 16 bit EPP data register.

```
ioctl(fd, FFB_IOC_EPP_DATA16_WR, val);  
ioctl(fd, FFB_IOC_EPP_DATA16_RD, &val);
```

These calls read/write one 24 bit EPP data register.

```
ioctl(fd, FFB_IOC_EPP_DATA24_WR, val);  
ioctl(fd, FFB_IOC_EPP_DATA24_RD, &val);
```

These calls read/write one 32 bit EPP data register.

```
ioctl(fd, FFB_IOC_EPP_DATA32_WR, val);  
ioctl(fd, FFB_IOC_EPP_DATA32_RD, &val);
```

This call returns a structure containing a variety of ffb device info.

```
ioctl(fd, FFB_IOC_DRIVER_DATA_RD, &dev);
```

dev is a structure of type ffb_dev_t

This call enables or disables an FFB crate from taking data.

```
ioctl(fd, FFB_IOC_ENAB_SET, crate << 16 + enable);
```

crate is an int with value of 0..7, each crate is 512 channels.
enable is an int with value of 1 for enabled or 0 for disabled.

This call reads the enable state an filterbank card.

```
result = ioctl(fd, FFB_IOC_ENAB_GET, crate << 16);
```

crate is an int with value of 0, 1, 2.

result is an int with value of 1 for enabled or 0 for disabled.