

# A Digital Modem Card for a Multi-channel Satellite Communications Payload

Kobus Botha, Ewald van der Westhuizen and Gert-Jan van Rooyen

Department of Electrical and Electronic Engineering, University of Stellenbosch, Stellenbosch, 7600

Tel: 021-808-4315, Fax: 021-808-3951, Email: {kobus,ewald,gvrooyen}@dsp.sun.ac.za

**Abstract**—This paper describes the design of a digital modem card for use within a multi-channel communications payload for a low earth orbit microsatellite. The payload provides multiple channels enabling high-speed data transmission of messages to and from rural areas that fall outside normal terrestrial coverage. Designing a prototype was the goal of the first phase of the project and the modem card is a subcomponent of the entire payload. The modem card is based on an interface card from the SumbandilaSat project and consists of various ICs, a signal processor and interfaces to other boards. The design of the software modem and the digital design of the card combined with the simulation and testing to date show that a complete implementation is feasible.

## I. INTRODUCTION

The rapid growth of the telecommunications industry is a worldwide phenomenon with people and computers generating and transmitting more and more information each day. Despite this rapid growth, there are still areas in South Africa without communications coverage. People inhabit these rural areas, yet certain essential communication needs are often not met. For example, rural hospitals may find it very difficult to communicate with specialists or suppliers in urban areas. Low-cost, low-bandwidth satellite coverage can provide a valuable service in such circumstances. Furthermore, such a communications system can play a central role in the aggregation of data from distributed data capturing units, such as those recently deployed on a large scale around the world to monitor changing ocean temperatures.

The second locally built satellite, Sumbandila, has such a communications system on board [1]. However, the system communicates at under 4800 bps, which allows only fairly short text messages to be sent. The Department of Communications (DoC) of the South African government commissioned Stellenbosch University to design a high-speed, multi-channel communications payload for possible use on a future satellite. The requirement from the DoC was a high-speed, multi-channel communications system with low-cost ground stations that could be deployed throughout the country.

In this paper, the design of a modem card for the communications system is presented. The novel software-defined radio (SDR) modem implemented on a digital signal processor (DSP) and the digital design required to integrate the components are explained. Integration of all the components requires a robust data marshalling unit, the role of which is fulfilled by a field-programmable gate array (FPGA). The modem card provides 24 uplink-downlink channel pairs, each running at 19 200 bps,

providing a total system data rate of 460 800 bps in both directions, making transmission of files in the megabyte range via satellite possible.

An overview of the system is presented in section II, after which the design of the modem card (including intercomponent communication and grounding topology) is discussed in section III. Following this, section IV presents the design of a novel direct-conversion software modem, and section V illustrates the FPGA design, including the finite state machine design and the device configuration. The completed system and proposed system tests are presented in section VI with section VII providing a brief conclusion of the work done to date and a discussion of further work.

## II. SYSTEM OVERVIEW

A diagrammatic overview of the communications system is provided in figure 1. At the top left of the diagram the on-board computer (OBC) is illustrated, which runs the system software. Raw user data is managed by the OBC and sent to the DSP for modulation. The DSP sends modulated data in the form of samples to a digital-to-analogue converter (DAC) which links the analogue signals to the radio frequency (RF) stage. The DSP also demodulates incoming samples from an analogue-to-digital converter (ADC) and sends the raw data back to the OBC. The flow of data is described in section V.

Connected to the OBC is the modem card which routes data between the SDR modem, the OBC and the RF hardware. The purpose of the modem card is to manage the interfaces to the various components, to modulate user data into a form adequate for transmission over radio and to demodulate incoming analogue radio data back to digital user data. The signal interface card (SIC) is an add-on board that was originally designed for use with the experimental payload on the Sumbandila Satellite, and was used as a reference design during the current development.

The software modem is implemented in C on a DSP chip. The DSP chip used is a Freescale DSP56311 [9]. The RF interface to the modem card is through a dual-channel DAC and two single-channel ADCs. All data marshalling and routing is managed by an Actel ProAsicPlus FPGA.

## III. MODEM CARD DESIGN

Due to the practical nature of the project, off-the-shelf components were used wherever possible to ensure rapid design and

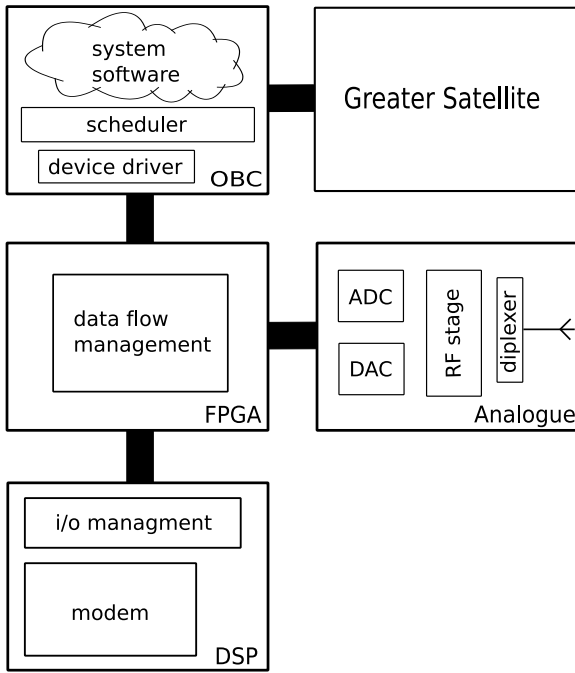


Fig. 1. System overview.

to avoid unnecessary debugging. The two design philosophies which were followed during design are:

- Let each component do what it is meant to.
- Simplicity.

“Let it do what it is meant to” means that tasks were assigned to the appropriate components. For the memory and storage intensive system software the OBC was used. DSP tasks were assigned to the DSP. Dataflow management, intercomponent data buffering and similar tasks were assigned to the FPGA. While it may have been possible to achieve the necessary data routing without an FPGA, it would involve complex logic schemes and hacks of unused pins on each device, making debugging an involved process and slowing down development.

The “Simplicity” philosophy caters for the finality of the project. When the payload is in orbit there is no way to reprogram the FPGA. System stability is critical as a flaw in the logic could render the entire payload non-functional.

#### A. Interfaces

Each interface is associated with at least one data path (described in section V). The modem card connects directly to the OBC, the DSP and the RF components.

1) *OBC Interface to the modem card:* The OBC, a Renesas SH4, is provided by SunSpace, and is considered an off-the-shelf component for this project. A CAN bus interface on the OBC connects it to the rest of the satellite, which allows telemetry commands to be sent and provides a maintenance backdoor into the system. The OBC has an expansion port on the address and data bus for interfacing to other components. The expansion port is memory-mapped inside the SH4, and any external devices are accessed as SRAM.

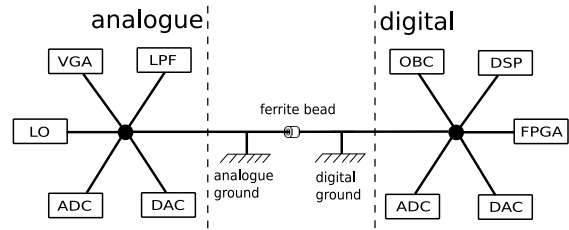


Fig. 2. Hybrid common reference point topology.

2) *DSP Interface to the modem card:* The DSP has a byte-wide interface called the host interface for interfacing to other subsystems [10]. Interrupt-driven transfers are used as the handshaking protocol for data transmission over the host interface. When the FPGA transmits or receives data over the host interface it causes an interrupt on the DSP. The DSP then jumps to the correct software routine for transferring the data between its internal buffers. Some qualities of the host interface that make it useful are:

- Choice of software polling, interrupts or DMA for data transfers.
- Host commands through which the host (FPGA) can issue commands to the DSP. This feature allows us to execute software routines on the DSP from the OBC, e.g. initialisation or reset routines.
- Host requests which provide the DSP with a set of signalling lines it can use to interrupt the FPGA.

3) *RF interface to the modem card:* The modem card interfaces with the analogue hardware by means of dual channel DACs and ADCs. I/Q mixing is used in a direct-conversion radio system. Both the DACs and ADCs have 16-bit quantisation resolution.

#### B. Grounding

All ground planes are connected to a common reference point. A hybrid grounding topology is used as defined in [7]. Star networks are used for the analogue and digital subsections. This means the ground pins of each device are connected to a central point. The two central points are then connected using a ferrite bead. The ground layout is depicted in figure 2.

### IV. DSP MODEM DESIGN

The function of the software modem is to act as a translator between digital information and radio frequency communication. The modem is implemented in the C language on a Freescale DSP56311.

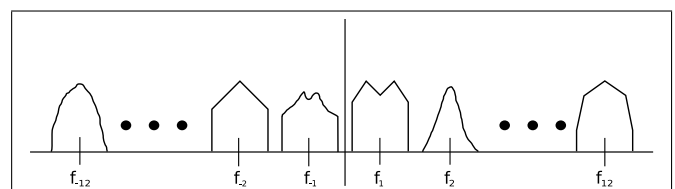


Fig. 3. Frequency spectrum of a multi-channel signal.

### A. Multi-channel modulation scheme

Phase shift keying (PSK) is a digital modulation technique that uses the phase characteristics of a carrier signal to convey data [2]. A change in the phase of the carrier signal indicates a change in symbol. Quadrature PSK (QPSK) uses a quadrature signal as carrier signal. Four phase positions in the quadrature signal represent four symbols. Each symbol represents two bits. Differential QPSK (DQPSK) uses the jump in phase position between the current and previous symbol to indicate the value of the symbol, rather than trying to determine the symbol value from the absolute phase of the message signal. Using DQPSK removes the need for the receiver to have a reference signal to determine the absolute phase of the received signal. QPSK also gives regular zero-crossing in the quadrature message signal with well randomized input data. The zero-crossings are used by the timing-error detector (TED) for symbol synchronisation.

The modem is designed as a direct-conversion (homodyne) transceiver [4]. The transmitted signal is constructed at baseband as a complex time-domain signal with twenty-four channels placed around DC, each channel with a bandwidth of 25 kHz (see figure 3). The radio frequency section mixes the baseband signal up to the carrier frequency. Each channel has a symbol rate of 9 600 symbols per second, equivalent to 19 200 bits per second.

The bandwidth of the system can be calculated from the number of channels and the bandwidth per channel:

$$B_{\text{sys}} = N \times B_{\text{ch}} = 24 \times 25 \cdot 10^3 = 600 \text{ kHz}$$

To represent sampled signals in the  $-300 \text{ kHz}$  to  $300 \text{ kHz}$  frequency band, a sampling frequency of at least twice the signal bandwidth is needed (Nyquist's criterion [2]). A sampling frequency at least three times larger than the highest frequency in the system gives enough excess bandwidth to allow for filter roll-off between channels. Therefore the minimum recommended sampling frequency ( $F'_e$ ) is:

$$F'_e = 3 \times 300 = 900 \text{ kHz}$$

The number of samples used to represent one symbol at the suggested sampling frequency is calculated as follows:

$$N'_{\text{sym}} = \frac{F'_e}{F_{\text{sym}}} = \frac{900\,000}{9\,600} = 93.75 \text{ samples per symbol}$$

For convenient digital signal processing, an integer value is chosen as the number of samples representing one symbol. Round  $N'_{\text{sym}}$  up to the nearest integer:

$$N_{\text{sym}} = \text{round}(N'_{\text{sym}}) = 94 \text{ samples per symbol}$$

Recalculate the required sample frequency for  $N_{\text{sym}} = 94$ :

$$F_e = F_{\text{sym}} \times N_{\text{sym}} = 9\,600 \times 94 = 902\,400 \text{ Hz}$$

### B. Modulator implementation

Bits are encoded to symbol values two bits at a time. The symbol values are used to generate a message signal pulse train. The message pulse train is sent through a raised-cosine filter to generate a message signal with smoothed phase transitions. The baseband carrier wave is generated by means of direct-digital

synthesis [3]. Each message signal is mixed to the respective baseband carrier frequency to form 24 modulated signals. The 24 modulated signals are scaled and summed to form a single baseband modulated signal ready for transmission. The samples of the baseband modulated signal are sent to the FPGA to be sent to the DACs.

### C. Demodulator Implementation

1) *Downmixing*: At the receiver each channel is mixed down to DC. A low-pass filter separates each channel from adjacent channels. Since each channel is mixed down to DC the same low-pass filter is used to separate the downmixed channels. The low-pass filter is implemented as a two-stage cascade fourth-order elliptic filter [5].

2) *Timing-error Detection*: Timing-error detection is done to synchronise the message signal with the intended symbol strobe positions. The timing-error detection algorithm by Gardner [6] is implemented. The advantage of this TED algorithm is that it is simple to implement and uses very little memory. A separate timing-error detector is implemented for each channel.

3) *Decoding*: The strobed signal values from the timing-error detector are used in the decoding process. The symbol jumps are determined from the change in the in-phase and quadrature components of the strobed message signal values. The previous symbol may change to any of four new possible positions. Each new possible position is at a 90 degree rotation from the next. The current strobed message signal values is compared with the four rotated possibilities of the previously strobed message signal values.

Let  $d_i(k)$  and  $d_q(k)$  be the current strobed in-phase and quadrature message signal values and  $d_i(k-1)$  and  $d_q(k-1)$  be the previous strobed in-phase and quadrature message signal values. The Euclidian distance between the current strobed signal values and each of the four rotated signal values of the previous strobed signal value is calculated as follows:

$$s_p = |d_i(k) - r_{ip}| + |d_q(k) - r_{qp}|$$

where  $p = 0, 1, 2, 3$  is each of the four rotated positions of the previous strobed message signal values. The possibility resulting in the minimum distance gives the most likely symbol jump. Each symbol jump is subsequently decoded to the two bits that the symbol jump represents. Decoding is done for each separate channel.

## V. FPGA DESIGN

The main function of the FPGA firmware is the routing of all the data between the various devices. A secondary function is the initialisation of the devices; the FPGA receives a reset signal from the OBC and then performs all necessary initialisation tasks. Finite state machines (FSMs) are used extensively in the implementation. The text will now discuss the types of data and their respective paths. Thereafter the state flow of the FSMs is discussed. Finally, the initialisation steps to achieve proper device configuration are outlined.

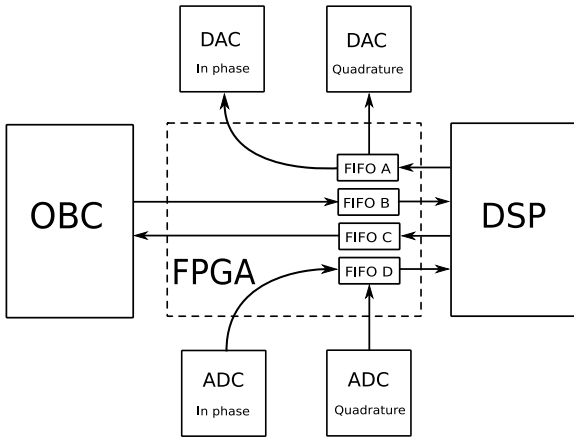


Fig. 4. System dataflow diagram.

### A. Dataflow management

Each component is associated with at least one type of data and this data flows in at least one direction (as seen from the FPGA's point of view). All data passes through the FPGA at least once. In short:

- SH4 – user data (e.g. text messages, images). Read and write.
- DACs – digital samples of modulated signal generated on DSP get sent to the DACs. Write only.
- ADCs – digital samples of modulated signal from the ADCs get sent to the DSP. Read only.
- DSP – digital samples and user data. Read and write.

Multiplexed tri-state buffers are used in cases where a single set of wires is used for both reading and writing. Each device runs at its own clock speed. Four asynchronous FIFOs are used to solve the problems presented by clock domain crossing. An asynchronous FIFO may be written to and read from at the same time at different clock speeds, i.e. the clocks do not need to be synchronous. FIFO underruns and overruns on the DACs and ADCs are ignored as they should not occur during normal operation – system-wide data rates are calculated and precisely adhered to. FIFO errors on the SH4 and DSP interfaces generate error signals that are passed up to the system software on the SH4 and ultimately are logged in a logfile which may be retrieved via the maintenance backdoor. Registers are used to connect the FIFOs to data busses where necessary. Figure 4 shows how the FIFOs connect to devices.

FIFO A is used to ensure data is clocked to the DACs at the constant sampling rate. FIFO B is used to buffer raw user data from the OBC for transmission to the DSP at a later stage. Similarly, FIFO C stores demodulated user data from the DSP for subsequent transmission to the OBC. Finally, FIFO D stores digital samples from the ADCs which are clocked by the FPGA at the constant sampling rate.

### B. State flow

Finite state machines (FSM) are synthesised on the FPGA in the form of registers as described in [8]. A FSM with  $n$  states is equivalent to  $2^n$  registers correctly wired. Five FSMs are implemented on the FPGA:

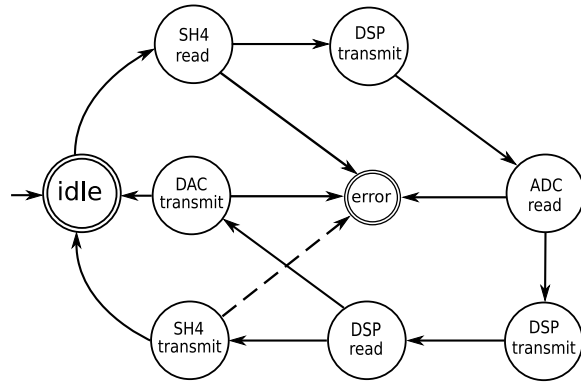


Fig. 5. Round-robin finite state machine (FSM).

- DAC FSM
- ADC FSM
- Expansion port FSM
- Host interface FSM
- Round-robin service FSM

The DAC FSM has a single goal: to clock digital samples from FIFO A to the DAC stage at the sampling rate. The digital samples are transmitted in pairs. One sample is the in-phase component and the other is the quadrature component of an I/Q sample pair. The FSM has three states, namely *idle*, *fifo\_read* and *dac\_write*. From the *idle* state, the FSM reads a pair of samples from FIFO A. In the next state it writes the samples to the DACs; then it returns to the *idle* state. Buffer underruns are not catered for. In the event of an underrun the previous sample pair is written to the DACs. Similarly, the ADC FSM's purpose is to clock the ADCs and store the sample pairs in FIFO D to be read at a later stage.

The expansion port FSM is a black box implementation to assist in transferring SH4 data. It conforms to the SRAM timings necessary to transfer data over the port. It has data lines, address lines and a read/write line as input, and outputs a "finished" signal to notify the parent FSM that the transfer over the interface is complete. In this way, data is read and written over the expansion port without knowledge of the timings that must be conformed to, because the FSM manages the timing. Similarly, the host interface FSM adheres to the timings for the host interface and implements a black box type read/write unit for the round-robin service FSM (described next).

The round-robin service FSM, depicted in figure 5, is the system scheduler of the FPGA. It ensures that each device is serviced routinely and also that no two devices try to write to the same bus at the same time. It begins in an *idle* state. It then notifies the expansion port FSM to read user data from the SH4 and place it in FIFO B. It then reads the next data in FIFO B and transmits it to the DSP using the host interface FSM. Next, the ADCs are read and samples transmitted to the DSP, after which data is read from the DSP. The data words received from the DSP are 24 bits wide, and the first byte is a control byte which is inspected to determine whether the data should be sent to the DAC or the SH4.

At any stage, if there is a FIFO overrun or underrun on FIFO B or C, an error signal is generated.

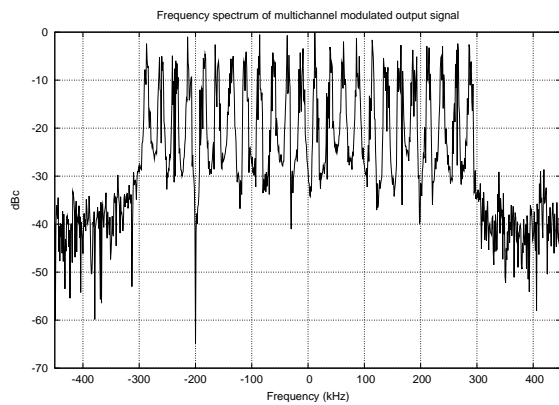


Fig. 6. Modulated signal frequency spectrum.

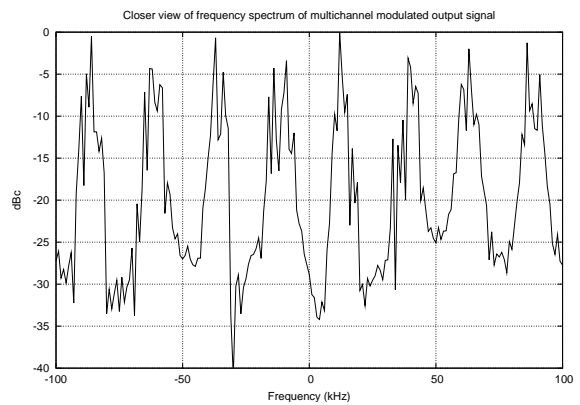


Fig. 7. Closer view of the frequency spectrum of a modulated output signal.

### C. Device configuration

After reset, the SH4 boots from an image stored on FLASH memory. The round-robin service FSM initialises the DSP and the DACs to known states. The DACs are configured by programming registers and the DSP is initialised using a host command which causes the DSP to jump to the initialisation routine. ADCs are configured with jumpers, so their configuration is always known.

## VI. SYSTEM TESTS

Completed and proposed tests are discussed next.

### A. Modem tests

The modulated multi-channel output signal was captured at the modulation output buffer of the DSP modem. A pseudo-random bitstream was used as input data to the modem. Figure 6 shows the magnitude plot of the frequency spectrum of the modulated signal. The modulated signal is a summation of all the channel output signals. The 24 individual channels are clearly visible in the plot with each channel centred around the expected channel center frequency. Figure 7 gives a closer view of a typical channel in the frequency domain.

An internal loopback test was performed with the DSP modem to verify that the modulated output signal is demodulated correctly. The data in the modulation output buffer in the DSP is moved directly into the demodulation input buffer. It is verified that the demodulated output equals the modulation input data.

A bit error ratio test was performed. The captured modulated output of the DSP modem is corrupted with Gaussian noise and demodulated. The demodulated data is compared with known correctly demodulated reference data and the ratio of corrupted bits is determined. A number of noisy modulated signals with known signal-to-noise-ratios (SNRs) are constructed by adding various noise levels to the uncorrupted modulated signal. Table I shows the percentage of bit errors and bit error probability for the corresponding SNR. The results are discussed in section VII.

TABLE I  
PERCENTAGE BIT ERRORS FOR VARIOUS SNRS.

SNR (dB)	% Bit errors	Bit error probability
15	0,081	$8,08 \cdot 10^{-4}$
10	1,389	$1,389 \cdot 10^{-2}$
3	15,938	$1,594 \cdot 10^{-1}$
0	27,370	$2,737 \cdot 10^{-1}$

### B. Interface tests

A data loopback test between the FPGA and the OBC was composed using C code on the OBC and a combination of FIFOs, registers, tristate buffers and multiplexers on the FPGA. Data in the form of 32 bit words are written to a register on the FPGA. A state machine on the FPGA then transfers the contents of the register into FIFO B. FIFO B is configured with a static trigger level of eighty percent. See Figure 4. When the level is reached i.e. when FIFO B is 80% full, another state machine transfers the contents from FIFO B into FIFO C. The OBC then reads the contents of FIFO C and compares the read values with the written values. No difference was found between the read values and the written values. The OBC-FPGA loopback test proves that the operation of the state machines, the FIFOs and the electrical interface between the SH4 and FPGA is reliable. Similarly, for the host interface between the FPGA and the DSP, a test was composed using assembly code on the DSP and digital circuitry on the FPGA. Data in the form of 24 bit words are transmitted from the DSP to the FPGA into FIFO A. A state machine then transfers the contents of FIFO A to FIFO D. Again, see Figure 4. The data is then read from FIFO D into the DSP memory where it is inspected. Glitch free data transfer was again observed indicating an error free interface.

### C. Further proposed tests

Other intercomponent communication tests that will be used to further verify the performance of the system include a loopback test that sends digital data from the FPGA to the DACs where the outputs of the DACs get looped back to the ADCs to be digitised. The loopback test between the FPGA, DACs and ADCs would serve as a verification that the designed data rate is reached and maintained and that data transmission is reliable. Data rate calculations on the expansion port need

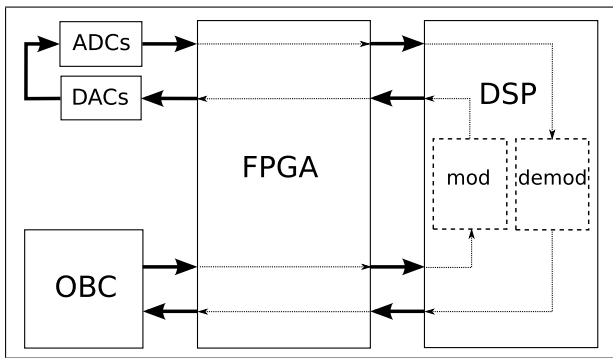


Fig. 8. System loopback test diagram.

to be done using the UNIX *time* command to verify that the interface can support the required data rate.

#### D. Full system loopback

The aforementioned tests will give the guarantee that all the component interfaces can be reliably integrated. The next step would be to do a full system loopback test. Figure 8 is a diagrammatic overview of a system loopback test with the radio frequency section omitted. The loopback occurs where the DAC output is fed back into the ADC input. The arrows indicate the direction of data flow. Messages sent by the OBC pass through the transmission and reception paths. Upon reception, each received message is compared to its transmitted counterpart.

## VII. CONCLUSIONS

The implementation of a software-defined radio modem for satellite use was presented as a design case study in this paper. The modem uses a direct-conversion architecture to minimise the sampling and processing rate of the digital hardware.

The software modem was successfully implemented on the Freescale DSP56311. Unit tests and an integrated test have confirmed the functionality of the modem. The modulated signal of the modem was captured and analysed and the frequency spectrum corresponds to simulation results. Demodulation also works as expected as is seen from the modem internal loopback test. The results from the bit error ratio test are poorer when compared to the ideal QPSK bit error probability figure in [2], which is mainly attributed to the increased quantisation error experienced by each channel when several channels are digitised together.

## REFERENCES

- [1] A. Cooke, *Rural E-mail System for the Sumbandila Satellite*, Masters thesis, Stellenbosch University, April 2007.
- [2] R.E. Ziemer and W.H. Tranter, *Principles of Communications: Systems, Modulation and Noise*, Fourth edition. New York: John Wiley & Sons, 1995.
- [3] J. Vankka, "Methods of Mapping from Phase to Sine Amplitude in Direct Digital Synthesis," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 44, no. 2, pp. 526–534, March 1997.
- [4] B. Razavi, "Design Considerations for Direct-Conversion Receivers," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 44, no. 6, pp. 428–435, June 1997.
- [5] M. Christensen and F. Taylor, "Fixed-Point IIR Filter Challenges," *EDN*, 9 November 2006.
- [6] F.M. Gardner, "A BPSK/QPSK Timing-Error Detector for Sampled Receivers," *IEEE Transactions on Communications*, vol. COM-34, pp. 423–429, May 1986.
- [7] T. Williams, *EMC for Product Designers*, Second edition. Oxford: Reed, 1996.
- [8] A. Rushton, *VHDL for Logic Synthesis*, Second edition. Chichester, West Sussex: Wiley, c1998.
- [9] Freescale/Motorola, *DSP56300 Family Manual*, Texas: 1994
- [10] Freescale/Motorola, *DSP56311 User Manual*, Texas: 1994