Software Defined Radio for Small Satellites

Eugene Grayver
Aerospace Corporation
2310 E. El Segundo Blvd.
El Segundo, CA 90245
eugene.grayver@aero.org

Andrew Chin
Aerospace Corporation

Jason Hsu
Aerospace Corporation

Stanislav Stanev
Aerospace Corporation

David Kun
Aerospace Corporation

Adam Parower
University of Michigan-Flint

Abstract— **Cubesats offer relatively lost-cost access to space. They are actively used for a wide range of scientific studies and technology development efforts. Small size, relatively low available power, and highly constrained cost have typically limited the radios on these satellites to low data rates. In this paper we present our first AeroCube 915 MHz software defined radio based on a Zynq processor and using Lime Micro transceiver developed to address the low-cost constraint while at the same time offering a state-of-the-art communications link. The radio supports a wide range of modulations and powerful error correction code capable of 10 Mbps. The coding and modulation can be adapted in real-time as the range from the ground station to the satellite changes to maximize downlink throughput.**

TABLE OF CONTENTS

1. INTRODUCTION

Development and launch of a satellite has traditionally been extraordinarily expensive. The high cost made developers necessarily risk-averse, impeding the introduction of new technologies for space. A class of small satellites collectively known as CubeSats was introduced to allow relatively low-cost access to space [1]. These satellites [1][3] are launched as a secondary payload and must fit into a well-defined size envelope. CubeSats are always (but not necessarily) launched into a low-earth orbit.

The first few generations of CubeSats (up until 2011) had relatively simple payloads and required a modest ground/space communications link. The latest generation is carrying megapixel cameras and other remote sensing instruments. It is now necessary that we maximize the amount of data that can be downlinked to the ground. The link characteristics (i.e. signal-to-noise ratio) changes as the satellite travels over the ground station. The radio can achieve greater throughput if it can adapt to these changes. Use of software defined radio (SDR) [4] to achieve high throughput has been suggested in a number of papers [5][6].

978-1-4799-5380-6/15/$31.00 ©2015 IEEE

In this paper we present a detailed design for the AeroCube SDR. The radio is based on state-of-the-art hardware for both digital signal processing and the RF subsystem. The flexibility of the SDR architecture will keep the radio design upgradable far into the future.

2. CUBESAT LINK AND CONOPS

A satellite is in Low Earth Orbit (LEO) if its altitude from the Earth surface is less than 2000 km. For the purpose of this paper, a LEO orbit is less than 1000 km average altitude – the Aerospace Corporation CubeSats (AeroCubes) use these orbits. The link consists of an omnidirectional antenna on the satellite with a gain of -10 dBi over a 90% sphere and a 23 dBi highly directional tracking dish antenna on the ground. Communications are maintained while the satellite is at least 10° above the horizon. The orbit for the target satellite provides 10-15 minutes of visibility from a ground station.

For a typical AeroCube pass, the received Signal to Noise Ratio (SNR) changes by over 20 dB as the satellite moves from the horizon to the zenith. The adaptive coding and modulation (ACM) technique tracks the link SNR and optimizes the data throughput with a combination of increased downlink symbol rate, higher forward error correction code rate, and higher order modulations (Figure 1). Aerospace's ACM technique supports a wide range of waveform parameters (Table 1) to maximize downlink throughput.

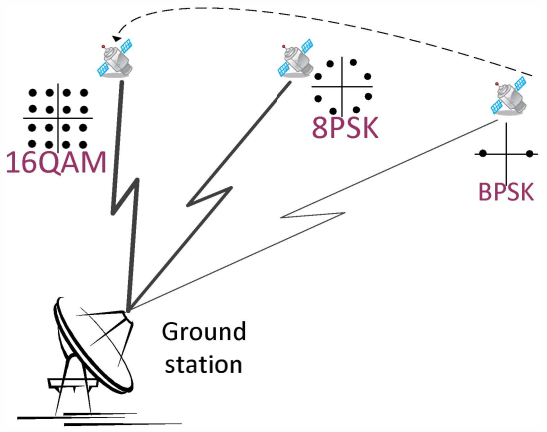

Figure 1: Conceptual ACM for space-to-ground link

Table 1: Waveform parameters

Parameter	Supported values
Symbol rate	1,000 – 1,000,000 symbols/s
Modulation	BPSK, QPSK, 8PSK
Pulse shaping	Root-raised-cosine
Error correction codes	Turbo (cdma2000) codec. Rate ¼, ½, block size 15 – 63 bytes.

Figure 2 is a simulation[1] of ACM performance for a notional satellite in an 800 km (500 mi) altitude circular orbit passing over a ground station. The SNR varies from -1 dB to 13 dB over the duration of a pass. The waveform changes for maximum throughput and the symbol rate is fixed to take full advantage of the available 1 MHz bandwidth. The ACM more than doubles the average throughput compared to fixed QPSK at ½ coding.

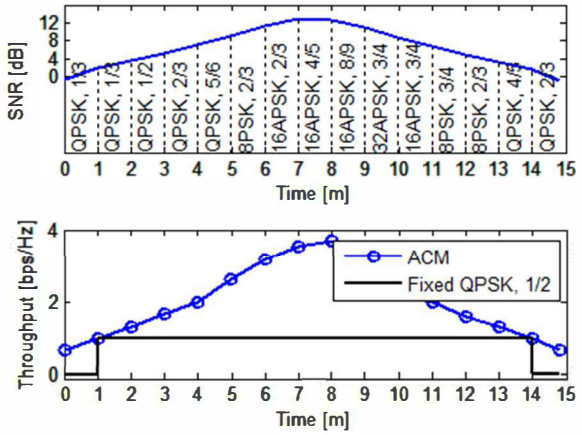

Figure 2: Simulated throughput results for a ground station pass with high SNR, 1 Msps

The radiation environment in the LEO orbit is relatively benign (i.e. < 5 KRad total dose in a year). Standard commercial off-the-shelf electronic components have a good chance of performing well in this environment.

The launch configuration of the SDR supports a subset of the waveforms in Figure 2, as listed in Table 2. The selected waveforms provide good coverage for low-SNR regime (3 dB increments) but do not provide coverage for the high SNR regime (11 dB step). The large jump is due to the selected error correction codec, which does not have code rates above ½. An on-orbit upgrade will allow higher code rates using 'non-standard' puncturing patterns.

[1] A 30 dBm transmitter with -9 dB antenna gain, 5 m (16 ft) ground dish antenna, 1 MHz bandwidth, 2.4 GHz RF carrier.

Table 2. Waveforms selected for initial capability

Mod	Symbol Rate [kSps]	Code Rate	Relative Bit Rate	Relative SNR	Delta SNR
BPSK	78.125	0.25	1	-26.54	
BPSK	156.25	0.25	2	-23.53	3.01
BPSK	312.5	0.25	4	-20.52	3.01
BPSK	625	0.25	8	-17.51	3.01
QPSK	625	0.25	16	-14.50	3.01
QPSK	625	0.5	32	-10.90	3.60
QPSK	625	1	64	0.00	10.90

3. DSP HARDWARE

The hardware for a SDR consists of two major subsystems: digital signal processing (DSP) and radio frequency (RF) as shown in Figure 3. The relatively low target symbol rate of 1 Msps makes an all-software implementation feasible. However, the radio can support much higher symbol rates in the future.

A field-programmable gate array (FPGA) performs the bulk of the DSP. The latest programmable system-on-a-chip from Xilinx was selected. The Zynq family [7] combines a FPGA with two powerful ARM processors. This chip has no flight heritage, but was selected by a NASA program for a next generation space computer [8]. A mid-range Zynq 7020 device was selected for the first AeroCube SDR radio. The key specifications are listed in Table 3. The chip is a 17 mm^2 ball-grid array.

Table 3. Key specifications for Zynq 7020

FPGA Resource	Available	Use [%]
Look up tables	53k	86
RAM	560 kB	56
Multipliers	220	74
CPU Resources	Available	Use [%]
Cores	2	50
Max frequency	866 MHz	25

The SDR signal processing firmware is legacy from earlier research and development into software defined radios. It was originally developed for an older Xilinx Virtex-II FPGA and ported to the Zynq. This firmware supports a wide range of modulations, including all the DVB-S2 constellations, as well as legacy waveforms such as FSK and GMSK. Only a small subset of the available features (Table 4) is used.

The radio can operate in either streaming or burst mode. For the transmitter, the two modes are essentially identical. The receiver resets all the tracking loops in burst mode, but can maintain loop state in streaming mode. Thus, the lower tracking loop bandwidth can be used in streaming mode since the loops have a long time to converge. The radio will operate in burst mode for the AeroCube application. The

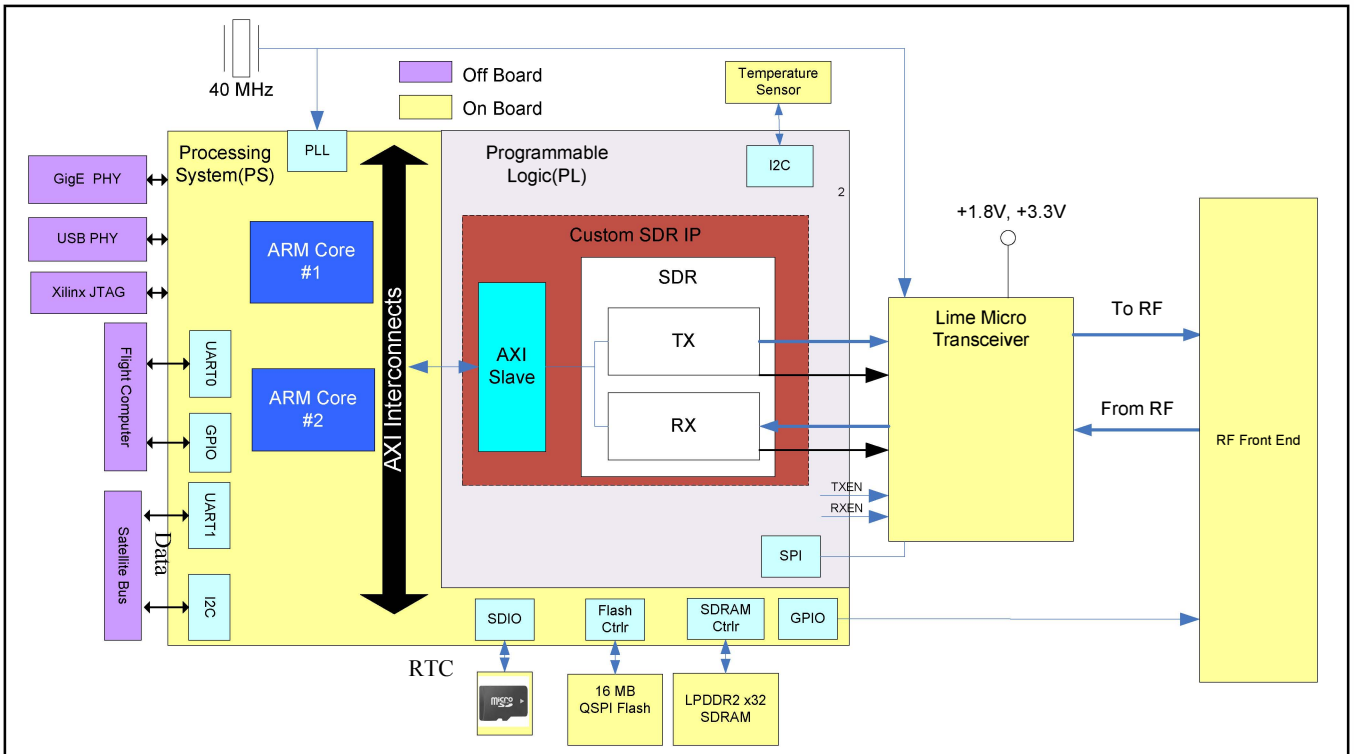

Figure 3. SDR hardware block diagram

relatively high Doppler is compensated (on the ground) by estimating the relative velocity based on the known orbit parameters. For simplicity, the correction is applied before a burst (packet) is transmitted.

A built-in BER tester is used to benchmark on-orbit downlink throughput performance. In that mode, the radio generates a pseudo-random sequence in the transmitter and checks the receiver output against that sequence[2].

The transmitter can be configured to output a single tone, or a chirp, while the receiver can be configured to capture raw (pre-D) samples for post-processing.

Table 4. Complete set of SDR capabilities

Supported Modulations	BPSK, QPSK, 8-PSK, 16,32,64-QAM User defined linear constellations with up to 64 points (6 bits/symbol) T/2 offset version of each modulation (e.g., OQPSK) Non-coherent detection of differentially encoded linear modulations: DBPSK, DQPSK, π/4-DQPSK, D8PSK, π/8-D8PSK Continuous phase modulations including GMSK, FSK
Spreading Modes	Unspread or Direct sequence spread spectrum
Spreading Codes	Maximum length, gold sequence, GPS C/A
Spreading Factor/PG	Spreading factors (SF) SF = {1, 16 – 2^{14}} Processing Gain (PG) = {0–42} dB
Symbol (Chip) Rates	100 symbols/sec – 10 Msymbols/sec
Data Rates	Data rate is a product of: spreading factor, symbol (chip) rate, modulation, and coding.
Pulse Shaping	Unshaped (rectangular pulse; β=0), Programmable shaping filter (e.g. RRC)
Forward Error Correction	• Reed-Solomon (regular or CCSDS) with symbol interleaver depths of 2 to 8 • Convolutional (r=1/2, 1/3) for constraint lengths of 7 and 9 • Turbo product code (up to 128×128) • Turbo 3GPP2 • LDPC (DVB-S2, JPL 8/9)
Security	AES encryption: 128, 192, or 256-bit keys
Additional Features	• Digital power amplifier linearizer • Adaptive receive constellation mapping

[2] Note that if the radio is the only (i.e. no backup) one available on the satellite, going into BER test mode will sever the link to ground. The results must be stored and reported the next time a link is established.

4. RF HARDWARE

The RF subsystem is based on a field-programmable RF IC from Lime Microsystems. The LMS6002D is a single-chip transceiver with key features listed in Table 5 [9]. The chip can operate in either full or half duplex, but is used in half-duplex mode for the target application. The AeroCube uses a fixed frequency of 915 MHz, and does not take advantage

of the full flexibility offered by the chip. However, a spectrum-sensing mode is possible using the wideband input of the LMS6002D. In this mode, the input bandpass filter is bypassed and the radio can tune anywhere in the supported frequency range.

Table 5. Key specifications for LMS6002D

Feature	Value	Units
Frequency range	0.3 – 3.8	GHz
Signal bandwidth	1.5 – 28	MHz
LNA noise figure @ 915	3.5	dB

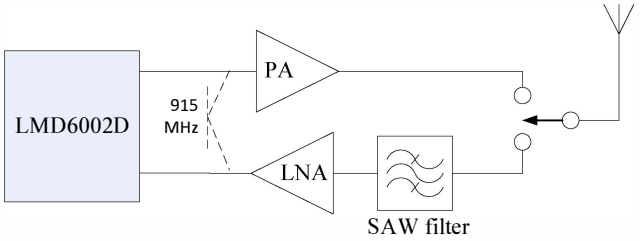

Figure 4. RF subsystem block diagram

The highly integrated Lime Micro RF IC requires extensive calibration to achieve acceptable performance. The chip has a number on internal DACs and gain stages to control LO leakage, IQ imbalance, etc. Calibration software runs on the ARM processor at boot-up and when temperature changes. An accurate temperature sensor is placed close to the RF chip to support temperature-dependent calibration. Some of the calibration functionality is internal to the chip and must simply be activated through the serial command interface. However, some calibration parameters (e.g. DC offset to control LO leakage) are measured using standard test equipment over a range of temperatures. A board-specific calibration file is then stored in on-board nonvolatile memory.

A power amplifier boosts output to the target 1W (30 dBm) level. An LNA boosts the input and sets the system noise figure.

5. Physical Characteristics

The SDR circuit board is a 3.0" square, fabricated on a standard substrate. All the components are surface mount and no issues with vibration are expected. The measured power consumption is 1.2W in receive mode and 2.5W when transmitting at 30 dBm.

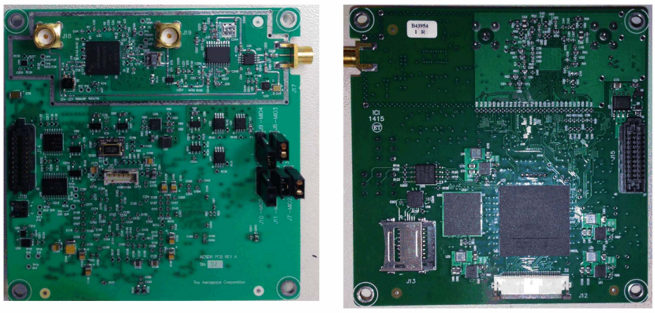

Figure 5. Photograph of the SDR board (RF on top, DSP on the back)

6. Peripherals

The ARM microprocessors provide an interface to a few standard peripherals and support ICs. The radio uses a serial port operating at 1 Mbps for data interface to the flight computer and other payloads. A second serial port is used for control and monitoring. This separation ensures that critical commands are not delayed by high-rate data, and that the high-rate data streams are guaranteed to be uninterrupted by commands.

A micro secure digital (SD) memory card is used for bulk nonvolatile storage. The card carrier is covered with epoxy before flight to reduce potential damage due to vibration. The micro SD memory card is only used for temporary storage since the data could be corrupted by radiation effects. For example, a new software/firmware upload is first saved on the µSD card. A 16 MB NOR flash is used for long-term storage of critical software and firmware. This flash is used to program the Zynq chip and contains the operating system for the ARM.

Wired Ethernet[3] and USB interfaces are on a separate board that is used only during development and debugging. A high-density ribbon cable links the development board to the SDR board.

7. Software

The embedded ARM microprocessor contains two cores although one core is currently disabled to reduce power consumption but can be enabled easily for additional processing needs.

The software consists of a Xilinx First-Stage-Boot Loader (FSBL), U-boot, Linux operating system, and SDR control software. The Xilinx FSBL and U-boot, a popular boot loader for embedded Linux are responsible for configuring low-level Zynq hardware and loading the FPGA part of the chip. It is also responsible for the fail-safe, on-orbit updates. The SDR is designed to be updated after launch by sending new software and/or firmware over the wireless link. However, it is possible for an upload to contain a bug that was not caught during testing. Failure of the radio

[3] Ethernet and USB connectors are bulky, heavy, and do not do well with vibration. However, an Ethernet interface makes software development significantly faster.

would end the entire mission and must be carefully guarded against [11]. To guarantee the SDR will always fall back to a known good condition should any failures due to uploaded software occur, two copies of the firmware including the FPGA configuration files are always stored in the Flash memory. A *golden* copy of the firmware is thoroughly tested before launch and is never allowed to be overwritten. The *updated* firmware will be saved in a different location in the Flash memory and can be overwritten many times. At every power-up, the bootloader checks the real-time clock to compute time since last successful communications session. In case the real-time clock fails, it also checks the number of power-ups since the last successful communications session. If either exceeds a specified maximum (e.g. 2 days, 100 bootups), the *golden* image is loaded from the Flash memory. Otherwise, the *updated* image is loaded. The main radio software records the time stamp of each successful link to the Flash memory. Repeated write to the same location in Flash memory can wear it out and cause that memory location to fail. Failure rate can be accelerated due to temperature and radiation. Instead of writing to the same memory location every time, a block of 1000 locations is allocated. The time stamp is stored to a new location (modulo 1000) every time and the boot-loader automatically retrieves the latest time stamp with the largest value. A block erasure to Flash is only needed after all 1000 locations have been used, effectively reducing the wear by a factor of 1000. A map of the Flash allocation is shown in Table 6.

Table 6. Map of the FLASH (all units in MB)

Contents	Offset	Size	
Boot Loader	0.0	0.1	
Kernel	1.0	1.5	golden
Device tree	2.6	0.1	
User software	2.6	1.0	
FPGA bit file	3.6	3.9	
Kernel	7.5	1.5	updated
Device tree	9.0	0.1	
User software	9.1	1.0	
FPGA bit file	10.1	3.9	
User mount	13.9	1.5	
System Log	15.4	0.6	

The SDR runs an embedded Linux operating system with BusyBox that provides basic Linux functionality in a small footprint. It is not a particularly lightweight solution as compared to a bare-metal or even a small RTOS. Linux was selected to expedite development because it provides a wealth of development tools and is familiar to most developers. The students at Calpoly San Luis Obispo have also used Linux for CubeSat software development with great success [10]. Finally, Linux (unlike bare-metal) provides a well-tested TCP/IP stack which will be used in the future.

The concept of operations for the radio is to wake up every 16 seconds, check if a ground station is attempting to establish a link, and power off if no link is established. Most of the time the satellite is not over any ground stations and the radio will turn-off quickly to save battery power. It is therefore important to boot up quickly. After significant effort to reduce the size of the Linux kernel, the boot-up time is about 3s from power-up to radio software processing incoming packets and then turn-off.

The SDR control software consists of two independent processes: a command processor and radio MAC. The command processor is responsible for: providing basic telemetry, on-orbit reprogramming, providing low-level access to FPGA registers, switching between data/BER modes. The radio MAC is described in the next section.

8. Radio MAC and Protocol Layer

The SDR firmware in the FPGA supports only the physical layer (i.e. bits to RF). The bits must be parsed into user data packets by the ARM. TCP/IP is an obvious choice for the higher levels of the protocol stack. Much consideration was given to the decision not to use TCP/IP in the first version of the SDR. We were concerned about the interaction between congestion control loops in TCP and the dynamic waveform adaptation in the SDR. Interaction between a dynamic wireless link and TCP has been studied extensively [12] and is usually safe. We were also concerned with the effect of the half-duplex link with non-negligible delay on the TCP throughput adaptation mechanisms. Finally, we were not committed to using Linux and a bare-metal implementation was considered. Given the lack of time to fully investigate the problems we chose to use a custom MAC that was developed for a previous version of the Aerospace AeroCube radio. The goal is to support TCP/IP in the next software update.

The custom MAC has been proven over two generations of AeroCubes, is relatively easy to describe and verify. The code did have to be ported from a legacy 8-bit microcontroller to the 32-bit ARM.

A quick summary of this MAC is presented here for developers interested in a very-lightweight solution. The overall design for the AeroCube Modem Software is that of a hierarchical state machine (HSM), a state machine design pattern that allows for nesting of states as well as special event handling on entry and exit of some states. The three major states are:

1. Establish a link. In this state the ground sends a 'Ping' packet to the satellite every second and waits for a response, 'Pong'. Once a response is received, both ground and satellite go to idle state and continue to exchange Ping/Pong packets.
2. Short data mode is entered when either side has a small amount of data to transmit (<512 bytes). In this mode each packet is acknowledged individually.
3. Long data mode is used when either side has a lot of data to send. In this mode, bursts of 16 packets

are sent at once and acknowledged. Long data mode is much more efficient than short data mode because the overhead of acknowledge packets is incurred 16 times less frequently. Adaptive coding and modulation is only enabled in this mode.

A fixed packet size and format is used to keep the code simple (and low risk). Each packet consists of a header, one FEC frame (if coded) and a 16b CRC. The entire MAC is only 5500 lines of C++ code, making it easy to analyze and verify.

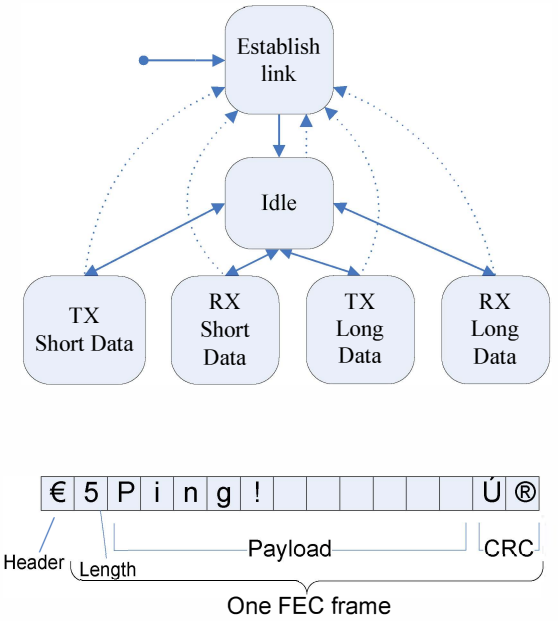

Figure 6. Packet structure used in Telemetry Mode and Short Data Mode

Adaptive coding and modulation is enabled in the long data mode. The FPGA firmware computes an estimate of the SNR for each received packet using the known preamble. The estimates are averaged over at least 16 packets. As the SNR increases, more efficient waveforms are enabled keep the link operating at close to capacity throughout the pass. The waveform definitions are binary files with register writes required to configure the FPGA. A table with supported waveforms is stored in a text file that contains key waveform information: minimum SNR, throughput, and the name of the waveform definition file. New waveforms can be added without recompiling the source code.
A spectrum analyzer capture of waveform adaptation in the long data mode is shown in Figure 7.

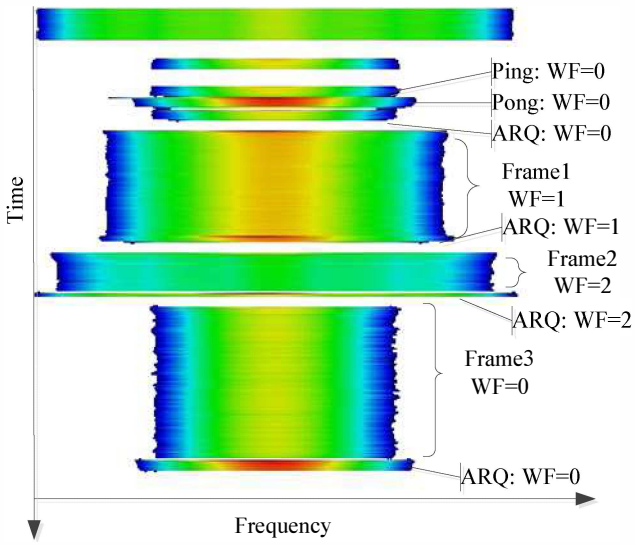

Figure 7. Spectrogram capture of waveform adaptation

9. Ka Band Upgrade

Our first AeroCube SDR will operate at a fixed frequency of 915 MHz. It has been fabricated and tested. We are planning to develop an SDR to support the Ka band at 26.1 GHz. Note that only the downlink is currently licensed by the FCC, but uplink in this band is also feasible. This section provides a summary of the *paper* study and no fabricated hardware exists at this time. The target symbol rate is 10 Msps or a bit rate of up to 40 Mbps using 12-4-APSK. A larger antenna will be required to achieve SNR sufficient to close the link at these higher-order modulations.

The baseband-to-IF hardware architecture and all the software remain the same. The signal is up/downconverted from an intermediate frequency of 2.5 GHz to the final 26.1 GHz frequency. The RF architecture is shown in Figure 8. The Ka-band component list is in Table 6
A Local Oscillator (LO) could not be found in the Ka-band frequency, so an X-band oscillator coupled with a frequency doubler and LO amplifier was used to generate the LO for the mixer.

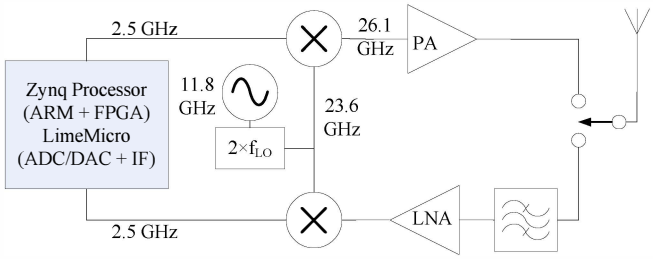

Figure 8. RF architecture for Ka band

Table 7. Components for Ka Band

Component	Part Number
LO	HMC807LP6CE
LO Driver	HMC449LC4
PA	HMC5445LS6
RF Switch	MASW-011036
LNA	HMC751LC4
Mixer	ML1-0832SM
Doubler	D-0612
Filter	3C60-8000/T20-O/O

Budgets for link gain and phase noise show that the design is feasible on a system level. The link budget has 1.4 dB margin for a net data rate of 10 Mbps using QPSK with the satellite at 10 degrees on the horizon. The LO phase noise is estimated to be low enough that it will not affect the bit error rate (BER). The necessary AeroCube DC transmit power is estimated at 8.8 W and the satellite receive DC power is estimated at 4.6 W. The LO and LO driver combine for 2.25 W of the aforementioned power consumption. It may be possible to find a lower power LO driver: however, the output power must be high enough to still drive the mixer after it goes through a splitter.

10. Conclusions and Future Work

A software defined radio will maximize the link throughput for the AeroCubes. It can be upgraded with more spectrally efficient waveforms and tested on orbit. The high power consumption is offset by the typically short 15 minute duration the radio is on when it is in view of a ground station. The satellite is scheduled to launch in 2015.

References

[1] Puig-Suari, J. and Turner, C. and Ahlgren, W. "*Development of the standard CubeSat deployer and a CubeSat class PicoSatellite*," IEEE Aerospace Conference, 2001

[2] Woellert, K., Ehrenfreund, P., Ricco, A. J., and Hertzfeld, H., "*CubeSats: Cost-effective science and technology platforms for emerging and developing nations*," Advances in Space Research, 47, 4 (2011), 663–684

[3] D. J. Barnhart, T. Vladimirova, and M. N. Sweeting, "Very-Small-Satellite Design for Distributed Space Missions," Journal of Spacecraft and Rockets, vol. 44, pp. 1294-1306, 2007/11/01 2007.

[4] E., Grayver, "Implementing Software Defined Radio," Springer, 2012

[5] Maheshwarappa, M. R., Bridges, C. P., "*Software Defined Radios for Small Satellites*," NASA/ESA Conference on Adaptive Hardware and Systems, 2014

[6] S. J. Olivieri, J. Aarestad, L. H. Pollard, A. M. Wyglinski, C. Kief, and R. S. Erwin, "*Modular FPGA-Based Software Defined Radio for CubeSats*," IEEE ICC - Selected Areas in Communications Symposium, 2012

[7] Xilinx, "*Zynq-7000 All Programmable SoC – Xilinx*", [online] www.xilinx.com/products/silicon-devices/soc/zynq-7000.html

[8] D. Rudolph, C. Wilson, J. Stewart, P. Gauvin, A. George, H. Lam, G. Crum, M. Wirthlin, A. Wilson, A. Stoddard, "*CSP: A Multifaceted Hybrid Architecture for Space Computing*," Small Satellite Conference, 2014

[9] Lime Microsystems, "LMS6002D is the world's first field programmable RF," [online] http://www.limemicro.com/products/field-programmable-rf-ics-lms6002d/

[10] G. Manyak, "*Fault Tolerant and Flexible CubeSat Software Architecture*," M.S. Thesis, California Polytechnic State University, 2011

[11] S. Fitzsimmons, "*Reliable Software Updates for On-orbit CubeSat Satellites*," M.S. Thesis, California Polytechnic State University, 2012

[12] G. Giambene1, S. Kota, "*Cross-layer protocol optimization for satellite communications networks: a survey*," International Journal of Satellite Communications and Networking, 2006

Biography

Dr. Eugene Grayver received a B.S. degree in electrical engineering from Caltech, and a Ph.D. degree from UCLA in 2000. He was one of the founding team members of a fabless semiconductor company working on low-power ASICs for multi-antenna 3G mobile receivers. In 2003 he joined The Aerospace Corporation, where he is currently working on flexible communications platforms. His research interests include reconfigurable implementations of digital signal processing algorithms, adaptive computing, low-power VLSI circuits for communications, and system design of wireless data communication systems. He is also participating in the software-defined radio community, trying to define a common configuration standard and determine optimal partitioning between software and hardware.

Andrew Chin was hired in 2006 as a summer intern during his undergrad studies at Harvey Mudd College. After completing his Masters in Electrical Engineering at Rutgers University, Andy was hired full time at the start of 2009. As a full-time employee, he continued his intern work by developing a next generation radio for the AeroCube

picosat program. The radio adaptively optimizes the data rate for maximum throughput. Andy is involved in most aspects of the picosat communication link, including the link budget, handshaking protocols, firmware programming, and RF characterization.

Jason Hsu received his M.S. degree in electrical engineering from UCLA in 2006. He has been with The Aerospace Corporation since 2004. He is mainly involved with the development of the modernized GPS user equipment and reconfigurable digital communication systems on field-programmable gate array (FPGA.) He was one of the key contributors to the Modernized Aerospace GPS Navigation Evaluation Test-bed (MAGNET), a compact multi-channel GPS receiverHe has also being supporting other DoD programs and IRAD projects, such as leading the development of a software cryptographic module for CubeSat and invention of Navigation Enabler for Single SV(NESS), a recorder-based GNSS constellation simulator which plays a critical role in proving backward compatibility of the GPS Block-III satellite

Stan Stanev received B.S and M.S degrees in electrical engineering from California State Polytechnic University in 2008. He joined Aerospace in June 2010 as a Member of Technical Staff. He is mainly involved with the development of software defined radio and reconfigurable digital communication systems on field-programmable gate arrays (FPGA.). He has implemented high speed Analog to Digital, Digital to Analog, Digital RF Tuner, 10Gbps network hardware and software interfaces. His expertise in embedded system development has helped advance the development of CubeSats.

David Kun received B.S. degrees in electrical engineering and computer science from University of Minnesota and an M.S degree from University of Illinois. Prior to coming to Aerospace, David worked at TRW Space and Electronics as a communication system engineer where he supported payload uplink jamming and payload-to-terminal timing and ranging performance and verification for the Advanced EHF program. He later became a design engineer where he designed, tested and integrated FPGA test drawers and fixtures for Advanced EHF, Orbital Express and ATP. In 2006, David joined Aerospace where he has developed and tested FPGA emulation testbeds and applied the software-defined radio technology for various NSS and NRO programs.

Adam Parower received a B.S. degree in Engineering and Economics from Harvey Mudd College in 2014. He is currently pursuing a teaching credential and M.A. in education. He plans to devote his life to teaching math and science to high school students in the city of Detroit.