

# RF OVER ETHERNET FOR WIRELESS INFRASTRUCTURE

Gerald Britton, Byron Kubert, and John Chapin  
(Vanu, Inc., Cambridge, MA, USA) {gbritton,bkubert,jchapin}@vanu.com

## ABSTRACT

Existing interconnects between RF and processing hardware have been tightly integrated at a system bus level or have incorporated proprietary network hardware, usually tied to the particular data being transported. This paper describes a system taking advantage of commodity Gigabit Ethernet to provide a flexible sample interconnect. Using off the shelf hardware provides flexibility at low cost allowing a choice of network topologies and equipment vendors. Our soft real time solution deals well with Ethernet jitter and unreliability, which has traditionally prevented its use in real time systems. The solution is general in nature, allowing for a variety of data content in many applications, including baseband interconnects, distributed antennas and signal processing clusters. We have implemented RF over Ethernet for the Vanu AnyWave™ GSM cellular basestation and report measurements of its performance.

## 1. INTRODUCTION

The *radio front end* of a software radio device contains the analog electronics for both the transmit and receive paths. The *baseband processor* contains the digital electronics for the (reconfigurable) signal processing.

In traditional designs, the front end and baseband subsystems were coupled via an analog signal at a low intermediate frequency, such as the 70 MHz standard used in US DOD systems, or via a pair of analog signals representing the in-band and quadrature components of the radio signal. In these systems the A/D and D/A converters were built into the baseband unit. More recently, system designers have been integrating the A/D and D/A converters into the radio front end, so the link between the front end and the baseband unit is a digital *sample interconnect*.

Organizations such as OBSAI [1], CPRI [2], and VITA Working Group 49 [3] have proposed sample interconnect standards enabling a radio front end developed by one company to be easily mated to a baseband processor from another. Section 2 describes these proposals in more detail.

In this paper we introduce a simple method to exploit Gigabit Ethernet as an RF sample interconnect. Vanu, Inc. has implemented the baseband side of the interconnect, while front end vendor Protium Technologies [4] has implemented the radio front end side. Our sample interconnect is commercially deployed as part of the Vanu AnyWave™ cellular telephone infrastructure system, where

it supports GSM/GPRS operation, and is effectively supporting wider bandwidth waveforms in the laboratory.

Gigabit Ethernet has numerous advantages as a sample interconnect which are described in Section 3. Section 4 outlines the main challenges to be overcome. Section 5 describes our design and Section 6 reports its performance.

## 2. SURVEY OF SAMPLE INTERCONNECT STANDARDS

A sample interconnect consists of a link-level data connection and a protocol used to communicate over that connection.

**Link-level only interconnect standards:** The most widely-used link-level connection standard for RF sample interconnect is undoubtedly PCI, if one adds together all the cards sold in the PCI, Cardbus, and PMC form factors to support software radio, radar, sensor and similar applications. Each of these cards has a custom protocol layer requiring card-specific drivers in the OS of the baseband processor. PCI is high speed and very cheap, but the fragmentation of form factors and the requirement for card-specific drivers limits inter-compatibility. Also, the lack of support for switching limits PCI to supporting a single point-to-point baseband-front end link, which rules out certain advanced applications and makes fault tolerance more difficult.

The other standard digital interfaces available on COTS processor boards have all undoubtedly been used in one project or another to support sample interconnects. The most important for our discussion of Ethernet is the use of USB 2.0 in the Universal Software Radio Peripheral (USRP) [5] associated with the GNU Radio open source software radio project [6]. USB2 is cheap, it runs at 480 megabits/second and is switchable, making it roughly comparable to Gigabit Ethernet.

For higher end systems, RapidIO [7], RaceWay [8], and Race++ are well-established link-level standards frequently used as sample interconnects in VME form factor systems. These standards scale to multiple gigabits per second with very low latency, are switchable, and appear to software in the baseband processor as a traditional peripheral bus, minimizing software overheads. However, the limited size of the market for these products means that exploiting these standards is expensive, both due to the high cost of COTS boards supporting them and due to the specialized chips, OS

software and design knowledge required to build these standards into new products.

**Protocol only interconnect standards:** The VITA (VMEbus International Trade Association) Working Group 49 proposal [3] is a protocol intended to support baseband/front end interoperability irrespective of the link-level connection. As of this writing the group has developed a detailed draft specification that is still evolving. The current draft defines a sophisticated protocol supporting a wide range of data formats and timing mechanisms.

In our view the implementation complexity and processing costs associated with a sophisticated protocol like this rule it out for systems that in operation will only ever support one or two data formats. Furthermore, the radio front end will frequently be implemented entirely in hardware, making protocol complexity particularly expensive. We believe it is important to investigate protocol alternatives that are extremely simple and low-overhead yet meet the needs of the majority of communication applications and systems.

**Full interconnect standards:** The OBSAI (Open Base Station Architecture Initiative) [1] Reference Point 3 specification is a sample interconnect standard intended to support a range of cellular telephony waveforms. The most recent version available publicly on the web, version 1.0, has protocol provisions for UMTS (3G WCDMA), GSM/EDGE, and CDMA 2000. It is easy to see how it could be extended for other waveforms. The link-level specification supports switching and has considerable design sophistication related to keeping the radio front ends and baseband processors synchronized to a single master clock.

The CPRI (Common Public Radio Interface) [2] specification has a much narrower goal than OBSAI. It focuses on the sample interconnect for UMTS (3G WCDMA) base stations. The authors of CPRI have argued that it can be extended to other waveforms but this is not well reflected in the current standards document. Like OBSAI RP3, CPRI provides time synchronization support.

Our concern with OBSAI and CPRI is simply the low manufacturing volumes and corresponding high costs and design challenges that are expected from any software radio-specific link-level interconnect. A full interconnect standard dedicated to software radio should only be considered if the engineering requirements cannot be met through use of a software radio specific protocol running over an existing widespread link-level interconnect.

### 3. BENEFITS OF USING ETHERNET

Ethernet has significant advantages as an RF sample interconnect.

*Fast:* Gigabit Ethernet is fast enough to handle roughly 20 MHz of instantaneous spectrum (sampled at Nyquist

with up to 16 bits digitizer resolution), which is wide enough for a large number of useful software radio applications.

*Easy upgrade path to higher speed:* 10-gigabit Ethernet is coming soon. Upgrading to exploit it will be largely transparent to baseband processor and front end implementations.

*Widely available:* For systems that can use an off-the-shelf board as the baseband processor, Gigabit Ethernet is provided on the motherboard of most servers and desktops, and is available in many laptops and embedded devices. For custom-built baseband processors and radio front ends, many microcontrollers and equivalent parts offer Gigabit Ethernet MACs on the die, while IP cores can readily be licensed for any FPGA.

*Cheap:* The low cost of working with Ethernet comes not just from mass-market chips and IP cores, switches and cables, but also from the built-in support in all operating systems, excellent diagnostic tools, and similar factors that dramatically reduce engineering cost. The cheap cost also makes it efficient to add multiple links to a radio front end, increasing the available bandwidth as well as allowing it to easily communicate with multiple baseband processors. This flexibility in network layout can also be extended to support failover or clustered processing for fault tolerance.

*Scalable range:* Gigabit Ethernet runs efficiently for a wide range of system designs. It can be a point-to-point link over cheap copper cabling, from a few inches to a few hundred feet long, without a switch. It can be point-to-point over longer distances over fiber. It also can be bridged over arbitrary distances and aggregated and switched to support arbitrarily large or fault-tolerant systems.

*High compatibility:* A radio front end using a Gigabit Ethernet link can be plugged into any of a wide range of baseband processors, and vice versa, regardless of form factor or operating system.

### 4. CHALLENGES OF USING ETHERNET

Some challenges must be considered, and if necessary, overcome in order to exploit Ethernet as an RF sample interconnect.

*Latency and jitter:* Most engineers intuitively feel that the latency and jitter of Ethernet are too high for a sample interconnect. This intuition is incorrect. Older versions of Ethernet had high latency and jitter due to collisions in a shared medium; Gigabit Ethernet uses dedicated links and has no collisions. Current networks experience latency and jitter due to queuing delays in switches. The RF sample interconnect can be point-to-point, without a switch, and if a switch is used, the necessary resources can be reserved in advance.

*Link overhead of packetization:* Ethernet has a relatively short packet length, approximately 1.5 kilobytes maximum for standard packets. Header transmission and

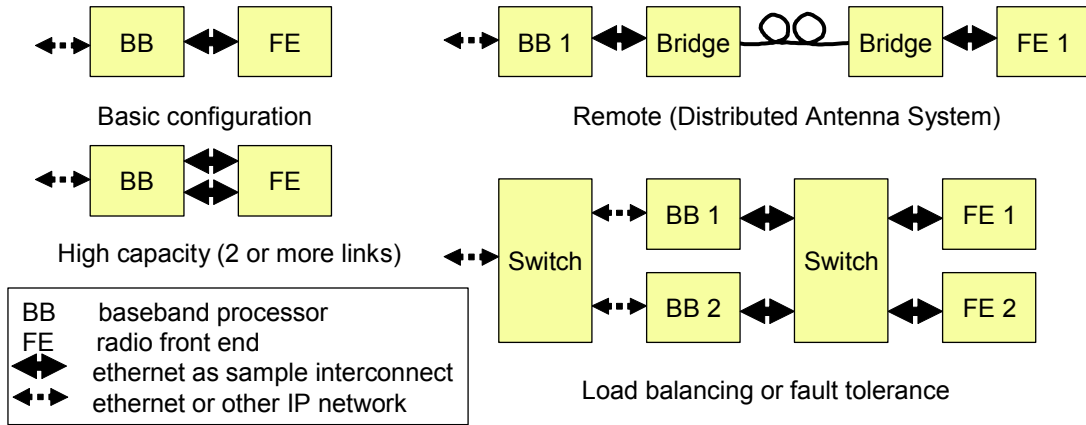


Figure 1. Example configurations.

inter-packet guard times reduce link capacity. We have found the overheads are tolerable, especially since we do not use IP headers, and the packetization significantly simplifies the challenge of multiplexing multiple streams onto the link. Using Jumbo packets can further reduce overhead at the expense of higher packetization latency.

*Packet loss:* Ethernet links are highly reliable. Essentially all packet loss in an Ethernet network is due to queue buffer overruns in switches. Buffer overruns can be largely eliminated in an embedded network like a sample interconnect by exploiting an end-to-end flow control mechanism in the Ethernet standard. Bit error can never be entirely eliminated, so some errors will occur. However, these will be detected by the Ethernet checksums causing entire packets to be dropped. Our implementation is designed to cope with an occasional dropout in the data stream with little or no impact on the overall system.

*Protocol processing costs in GPPs:* While Ethernet can be highly efficient to implement in a FPGA or microcontroller, in a GPP the protocol processing cost of receiving an Ethernet packet can be significant. We have found the processor cost is tolerable under Linux, given the high benefits and cost reductions elsewhere in the system.

*No support for clock distribution:* Ethernet is an asynchronous network. This is necessary for large-scale and fault-tolerant system implementations. However it requires a change in design philosophy, since traditional radio designs assume a common clock is distributed to all subsystems. In Vanu cellular telephone infrastructure systems, where the front ends are widely separated, each front end includes a GPS time and frequency reference. In other applications where the radio front ends are colocated, the front ends share a common clock through analog distribution. The additional cost of running a separate wire for clock distribution is minimal compared to the benefits of using Ethernet for the sample interconnect.

## 5. THE VANU ETHERNET SAMPLE INTERCONNECT

This section describes our RF sample interconnect at a high level.

**Raw Ethernet:** We chose to implement the protocol directly on top of Ethernet rather than on top of IP. This significantly reduces protocol overhead, both on the wire and in receive processing. It makes it much simpler to do an all-hardware implementation in the radio front end. It does not appear to limit the capability of the system: a bare Ethernet network without IP still supports switching, end-to-end flow control, and remote management. We see no immediate need for layer-3 routing in these tightly coupled embedded networks.

**Dedicated Ethernet:** We assume the Ethernet links and switches forming the sample interconnect are dedicated to the sample interconnect function. This reduces both the probability of data loss and the software overheads in the baseband processor. Figure 1 shows some sample configurations.

Many COTS boards have multiple Gigabit Ethernet ports so other ports are available for connection to a general IP or backhaul network. In some form factors such as ATCA, dual Ethernet switches and independent connections to the baseband processor are integrated into a single hardware unit, facilitating this architecture.

**Packet format:** Figure 2 shows the packet format. Following the Ethernet packet header, the packet consists of a two byte channel number with flags embedded in the high bits, a four byte timestamp, and an arbitrary length payload (up to the maximum Ethernet packet size). One channel number is reserved as a control channel; the others are data channels. Each data channel can transmit or receive samples at a rate different from the other channels.

Packets tagged with the control channel number follow a higher layer protocol, not described in this paper. Packets tagged with a data channel number have a homogenous

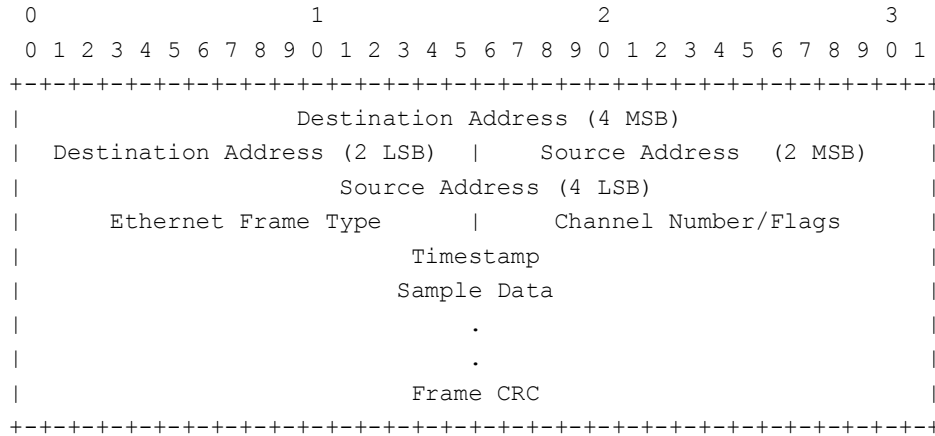


Figure 2. Format of a raw 802.3 Ethernet packet for RF sample interconnect

payload consisting of some number of 32-bit complex samples, each containing 16-bit real and imaginary components. This leads to the following link capacity for some common wireless standards assuming a 4x oversampling rate. (The per-packet overhead used in this computation includes not only the 24 bytes of protocol information shown in Figure 2 but also the 12-byte gap and 8-byte preamble required by the Ethernet PHY.)

Table 1. Ethernet link capacity with Vanu protocol

	GSM	CDMA	WCDMA
Msamples/sec at 4x oversampling	1.08	4.9152	14.746
Packets/sec @ 256 samples/pkt	4,219	19,200	57,602
Mbits/sec including overhead	36	164	492
Channels per Ethernet link	27	6	2

If one end of the link is a COTS GPP server as in the Vanu systems, we target 800 Mbits/sec rather than 1 GBit/sec, so the number of channels supported is 80% of the above. This target will increase in next generation faster systems.

**Timestamp semantics:** The timestamp on a data packet sent by the radio front end indicates the time at which the first sample of the packet was produced by the A/D converter for that channel. The timestamp on a data packet sent by the baseband processor to the radio front end indicates the time at which the first sample of the packet should go out through the D/A converter for that numbered channel. If a packet reaches the head of the transmit queue in the front end and the timestamp has already passed, it is assumed an error has occurred and the packet is discarded.

The timestamp semantics allow both sides to recover from occasional discarded packets due to bit errors on the wire or system load on the processing host. Both the baseband processor and the radio front end assume fill of zeroed samples for missing data packets.

The timestamp on a control packet is used to synchronize commands such as frequency hopping and telemetry such as RSSI values with the channel data.

## 6. PERFORMANCE AND EXPERIENCES

*Latency:* The transmit stream needs to be a certain number of samples ahead of the receive stream to account for the roundtrip latency in the RF sample interconnect. We measured this delay as 1.04 milliseconds in a configuration with 2.4576 megasamples per second RX, 1.2288 megasamples per second TX, 4 byte samples, and 256 samples in each Ethernet packet. The baseband processor was an HP Proliant DL380 server with dual 2.8 GHz Intel Xeon processors. The front end was a Protium Technologies device with a low-latency all-hardware Ethernet implementation.

We measured the latency with a test application that starts by pre-filling the TX buffer in the radio front end with some number of samples. Then the A/D and D/A clocks are started. Whenever the application receives samples from the front end, it transmits half that number of samples. Since the TX is half the rate of RX, this should be steady state. Transmitted samples are selected so that the waveform output by the front end is a sinusoid. This waveform is viewed on a spectrum analyzer. Any TX buffer underruns will appear on the spectrum analyzer as a distortion in the sinusoid (this is due to the behavior of the Protium front end in this condition). To determine the minimum amount of TX buffering, the pre-fill size was increased until a pure

sinusoid was observed. We found the minimum pre-fill size to be 1280 samples at 1.2288 MSPS.

This round trip latency would not be acceptable for an 802.11b access point, which requires several orders of magnitude faster response. Similarly, certain military applications such as IFF require extremely low round-trip latencies. Ethernet is not the right sample interconnect for applications like these. However, the round trip latency of 1.04 msec is small enough for most wireless standards.

*Processor overhead:* The baseband application running the above test consumed 19% of one of the 2.8 GHz Xeon processors, which is 10% of the dual-CPU machine's overall processor capacity. Overhead was measured with a "cycle soaker." The soaker application runs at a low process priority in parallel with the test application. The soaker increments an integer variable in a tight loop. Comparing the rate of incrementing when the test application is running to the rate when it is not gives an overall picture of the system load, including OS costs, created by running the Ethernet sample interconnect. Note that the baseband application read all the received RX samples.

We feel that spending 10% of the machine's processor capacity on sample I/O is an acceptable cost given the other benefits of using Ethernet as the RF sample interconnect. This cost is expected to decrease in future, faster machines.

*Packet loss:* We have never observed packet loss when the radio front end and the baseband processor were connected point to point. Connecting the units through a switch resulted in substantial packet loss in the TX direction on some switches and not on others. This was due to variations in buffer management in different switches when presented with a high packet load and occasional Ethernet-level flow control packets. The loss occurred in TX and not in RX because the receive buffers are much smaller in the Protium front end than in the Linux server. On one switch where we found the problem, the vendor was able to make a simple firmware upgrade that eliminated the packet loss.

*Engineering benefits:* Compared to other sample interconnects, such as PCI, we found Ethernet very easy to work with. We were able to avoid the cost and risk of driver development. Furthermore, it turned out to be quite valuable to directly observe sample data flow, non-intrusively, using standard Ethernet capture tools provided with the Linux operating system. This was a valuable probe point into the behavior of the software radio.

## 7. CONCLUSIONS

We have found Gigabit Ethernet to be an efficient and cost-effective RF sample interconnect. By selecting an extremely simple protocol, we enabled a straightforward all-hardware implementation in the radio front end and minimized software overheads in the baseband processor. The ability to

bridge Ethernet over long distances and use switches provides a cost-effective zero-hardware-development path to deploy a variety of advanced system designs for scalability, distribution, and fault tolerance. Ethernet also has the strong advantage that a 10x speed upgrade is beginning rollout. 10GbE will be affordable and available whenever endpoint processing capacity becomes fast enough to need more data than Gigabit Ethernet can supply.

We observed that latency, processor overhead and packet loss rates were all acceptable for commercial wireless infrastructure. USB 2.0 may be a good alternative for applications requiring lower latency than Ethernet can support. (For a lower bound latency analysis of a USB2 sample interconnect, see [9]). However USB lacks the upgrade path to higher speeds and sophisticated system architectures that gives Gigabit Ethernet long term viability. No other interconnect comes close to matching the combination of high performance and low cost of Ethernet.

We found that it is necessary to evaluate Ethernet switches carefully as only some have the right behavior to perform well in a sample interconnect.

The interconnect described in this paper has been commercially deployed and we expect to use it as the basis for many future system designs.

## ACKNOWLEDGEMENT

We are grateful to Mike Smutek and the Protium Technologies team for their contributions to the design and implementation of this sample interconnect.

## REFERENCES

- [1] <http://www.obsai.org>
- [2] <http://www.cpri.info>
- [3] <http://www.digitalif.org>
- [4] <http://www.protiumtechnologies.com>
- [5] [http://home.ettus.com/usrp/usrp\\_guide.html](http://home.ettus.com/usrp/usrp_guide.html)
- [6] <http://www.gnu.org/software/gnuradio>
- [7] <http://www.rapidio.org>
- [8] <http://ess.web.cern.ch/ESS/standards/Av5dot1.pdf>
- [9] <http://lists.gnu.org/archive/html/discuss-gnuradio/2005-02/msg00035.html>