Report on the relative performance of different CPE devices for Nokia SDSL

Michael Sokolov

Harhan Engineering Co.
Open WAN Connectivity Project
http://ifctfvax.Harhan.ORG/OpenWAN/

ABSTRACT

This study sought to characterize the performance of Harhan's OSDCU acting as a CSU/DSU for Nokia SDSL/ATM and to compare it against other available CPE options for Nokia SDSL. The tests have been performed in a lab environment set up with a Nokia D50 DSLAM as a simulated service provider (SSP); the CPE devices tested with this SSP included the OSDCU, two Netopia routers and Paradyne iMarc 9783-C. Performance measurements (latency and throughput) have been obtained by doing ping and FTP tests from the simulated subscriber host to a server on the SSP side.

The results show that the highest Nokia SDSL speed tier which the OSDCU can handle is 1152 kbps for the FPGA-enabled version and 768 kbps for the sans-FPGA version. The throughput degradation in the FPGA-enabled version occurs in the downstream direction at 1152 kbps (94.838 kbyte/s measured); in the sans-FPGA version it occurs in the upstream direction at 768 kbps instead (67.769 kbyte/s measured); in every other case there is no statistically significant difference in the FTP throughput performance between different CPE devices.

While the addition of the FPGA reduces the latency slightly (by 2-5 ms for small packets and 10-15 ms for large packets), it is still worse than the Paradyne unit by 15-25 ms, and significantly worse than the Netopia routers. Of all CPE devices tested Netopia 4652 featured the lowest latency. These performance figures appear to be the limit of what the current OSDCU hardware can do, i.e., there is no immediately apparent way to reduce the latency to match the existing commercial CPE devices and to support all Nokia SDSL speed tiers using the current OSDCU hardware design; the most promising avenue toward that goal would be to use an MPC8xx PowerQUICC processor instead.

Introduction

In the course of our Open SDSL Connectivity Project (predecessor to the Open WAN Connectivity Project) Harhan has built a device consisting principally of an MC68302 microprocessor, the RS8973 SDSL transceiver and an EIA-530 synchronous serial DCE port; this device is called OSDCU. (The name is a historical artifact and stands for Open source SDSL Debug and Connectivity Unit.) Functionally this device acts as a CSU/DSU, interfacing SDSL to a third-party WAN router via EIA-530.

Owing to the existence of many different flavors of SDSL, the concept of a ''CSU/DSU for SDSL'' can in fact mean one of two very different functions:

1. For simple HDLC-based SDSL flavors such as Copper Mountain a CSU/DSU is a trivial device in which the bit stream from the SDSL transceiver IC is simply wired to the V.35 or EIA-530 DCE port.
2. If the SDSL flavor sends ATM cells to the subscriber (as is the case with Nokia), the previous approach of presenting the raw SDSL bit stream to a standard (i.e., HDLC-expecting) WAN router would not be particularly useful. Instead a *practically useful* CSU/DSU for SDSL/ATM would be a device that converts between ATM and HDLC on the fly, talking ATM to the DSLAM while presenting HDLC to the locally

attached WAN router.

A CSU/DSU implementing Approach 1 would have no measurable effect on network performance. If one is working with an SDSL flavor that sends HDLC to the subscriber, there is no fundamental performance-impacting difference between a monolithic CPE device that goes from SDSL all the way to Ethernet versus a two-piece solution consisting of a separate router and CSU/DSU: it does not make any difference whether the connection between the router's HDLC interface and the SDSL transceiver IC is a set of PCB traces or a V.35 cable. (The propagation delay through the cable and the EIA transceivers is well below anything measurable at the network level.) Although it is clear that different router designs may perform differently, evaluation and comparison of classic HDLC WAN routers is outside the scope of this study; the relevant point is that the CSU/DSU component makes no contribution to anything performance-impacting.

A markedly different situation exists in the case of SDSL/ATM such as Nokia. An ATM/HDLC converting DSU will necessarily add some latency: at the very minimum, each packet normally has to be received on one interface (ATM or HDLC) in its entirety before it can be forwarded to the other interface. The converter is effectively an extra hop in the network packet path, even though it doesn't show up in traceroute because it happens at Layer 2 rather than 3, and it adds latency just like an extra router would. And just like a router's added latency may be smaller or greater depending on the router's design, the same holds for an ATM/HDLC converting DSU.

Harhan's OSDCU can function as an ATM/HDLC converting DSU for the Nokia SDSL flavor, and when it is used in this operation mode, the specifics of the device's design can impact the resulting system performance. Therefore, we have felt it prudent to perform a set of experiments to characterize the performance of our OSDCU when used as a CSU/DSU for Nokia SDSL. In order to isolate those performance effects which are attributable to the OSDCU, we have designed a series of controlled experiments in which we have made head-to-head performance comparisons between our OSDCU and other existing CPE devices for Nokia SDSL.

An additional question of interest to us was the performance impact of the optional FPGA part on our OSDCU board. Our current OSDCU design allows the Nokia SDSL/ATM to HDLC conversion function to be performed in two different ways:

1. Using SCC1 (one of the 3 Serial Communication Controller cores in the MC68302 processor) to receive and transmit Nokia SDSL frames.
2. A hybrid approach using SCC1 in the Rx direction only, whereas SDSL Tx bit stream generation is moved into the EPF10K30A FPGA.

The SCC-only approach is aesthetically less elegant (using an SCC in the transparent mode to implement a telecom framer is stretching the MC68302 well past the range of applications for which it was designed to perform well), and the hybrid SCC/FPGA version was created in the hope of obtaining better performance. Thus our performance tests have had an additional objective of evaluating whether or not the FPGA has provided the expected improvement.

(Doing both Rx and Tx in the FPGA for Nokia SDSL is not possible on the current OSDCU hardware because there isn't enough internal RAM inside the EPF10K30A to hold the needed frame buffers; the current OSDCU board cannot accommodate external SRAM or a higher-density FPGA either.)

It was already known prior to this formal study that our OSDCU is unable to handle the highest speed tiers of Nokia SDSL. The common SDSL8 line card offers 5 data rate choices: 192, 384, 768, 1152 and 1536 kbps. The SCC-only version of the Nokia SDSL/ATM to HDLC Layer 2 converter on the OSDCU is unable to handle 1152 or 1536 kbps speeds: it experiences an SCC Tx underrun which then results in the SDSL link being torn down. The SCC/FPGA hybrid version fails similarly at 1536 kbps, except that the failure mode is an SCC Rx overrun instead. Thus the highest Nokia SDSL speed that one could even consider to be supported by the OSDCU before taking any performance considerations into the account is 768 kbps for the SCC-only version and 1152 kbps for the SCC/FPGA hybrid version. Characterizing the performance of the device at these ‘‘top’’ speeds was another goal of this study.

Materials and methods

All CPE devices under test were being connected to the same simulated service provider (SSP). The SSP consisted of a Nokia D50 ATM multiplexor (MCS and LCS chassis) simulating the CLEC and a Cisco 3640 router serving out ATM on a DS3, simulating the partner ISP. The DS3/ATM output from the C3640 was fed directly to the trunk input on the D50 MCS; as real networks will have additional ATM switches in between, we were testing an idealized case. Furthermore, there were no other active ATM connections through the D50 except the one under test. These idealizations further our objective of bringing to light whatever performance limits are imposed by CPE devices, not by network business practices (oversubscription etc) or any other factors.

The D50 ATM multiplexor (DSLAM) system had been put together as follows:

Node software version:	11.1.1
Trunk card type:	DS3T2
Interconnect between MCS and LCS chassis:	MLAT3/LSMT3
Line card:	SDSL8

The D50 had been configured via Craft Terminal, Nokia's GUI tool. All ATM QoS settings in Craft Terminal (which this hacker admits to not really understanding) had been left at the defaults. The only place where Craft Terminal required us to make a choice was the traffic descriptor selection when making the A-Z connections; predefined traffic descriptor #9 had been chosen.

The C3640 hardware and software configuration is shown in Listing 1 below.

```
TR3B>show version
Cisco Internetwork Operating System Software
IOS (tm) 3600 Software (C3640-IS-M), Version 12.2(19a), RELEASE SOFTWARE (fc2)
Copyright (c) 1986-2003 by cisco Systems, Inc.
Compiled Mon 29-Sep-03 23:45 by pwade
Image text-base: 0x60008930, data-base: 0x61134000

ROM: System Bootstrap, Version 11.1(19)AA, EARLY DEPLOYMENT RELEASE SOFTWARE (fc1)

TR3B uptime is 1 week, 5 days, 2 hours, 10 minutes
System returned to ROM by power-on
System image file is "slot0:c3640-is-mz.122-19a.bin"

cisco 3640 (R4700) processor (revision 0x00) with 58368K/7168K bytes of memory.
Processor board ID 10705470
R4700 CPU at 100Mhz, Implementation 33, Rev 1.0
Bridging software.
X.25 software, Version 3.0.0.
SuperLAT software (copyright 1990 by Meridian Technology Corp).
1 Ethernet/IEEE 802.3 interface(s)
1 FastEthernet/IEEE 802.3 interface(s)
2 Serial network interface(s)
1 ATM network interface(s)
DRAM configuration is 64 bits wide with parity disabled.
125K bytes of non-volatile configuration memory.
8192K bytes of processor board System flash (Read/Write)
16384K bytes of processor board PCMCIA Slot0 flash (Read/Write)

Configuration register is 0x2102
```

The DS3/ATM module was NM-1A-T3. The relevant snippet from the running configuration is shown in Listing 2 below:

```
interface ATM2/0
 ip address 172.23.3.1 255.255.255.0
 atm clock INTERNAL
 atm scrambling cell-payload
 no atm ilmi-keepalive
```

```
 pvc 0/41
  protocol ip 172.23.3.41
  ubr 384
  encapsulation aal5snap
 !
 pvc 0/42
  protocol ip 172.23.3.42
  ubr 384
  encapsulation aal5snap
 !
!
ip route 172.23.41.0 255.255.255.0 172.23.3.41 permanent
ip route 172.23.42.0 255.255.255.0 172.23.3.42 permanent
```

The ATM PVCs were being served out in the routed RFC 1483 encapsulation. 172.23.3.*x* addresses were being assigned to these routed (non-Ethernet) WAN interfaces, while 172.23.*x*.*y* were being made available for a LAN behind the CPE router under test.

The C3640's FastEthernet interface was connected to Harhan's house network. There is an x86 Linux server on that network which was used as a test target; this server was separated from the C3640 only by a single 100 Mbps Ethernet switch. The performance tests were conducted by plugging a laptop (also Linux/x86) into the LAN behind the CPE router under test (i.e., simulating the subscriber host), then doing ping and FTP tests from that laptop to the abovementioned test server on the SSP side. The connection between the laptop and the CPE router under test was always 10BaseT, a straight-through cable with no additional hubs or switches.

The objective of the ping tests was to observe any differences in latency (packet round trip time) between different CPE devices connected to the same SDSL service from the SSP; the objective of the FTP tests was to observe any differences in the effective data transfer throughput at the user level. Because packet rtt is usually quite different between small and large packets, both of which are important in the real world, two ping tests have been performed: one with standard small ping packets and the other with large packets close to the 1500 byte maximum. The test ping commands were:

Small packets: ping −n −c 50 $server_ip
Large packets: ping −n −c 50 −s 1400 $server_ip

The −n option was used to prevent the possibility of reverse DNS query packets skewing the results; each ping test was run for 50 packets with 1 s intervals between packets.

The FTP tests consisted of connecting via FTP from the laptop to the test server, then downloading a test file, then uploading that test file back. The test file was an MP3 of length 3395210 bytes. The transfer time duration as reported by the FTP client with 0.1 s resolution was the recorded result of each test. The throughput number was then computed by dividing bytes/seconds and reported in kbyte/s; the 'k' SI prefix here means 1000, not 1024. (The throughput number reported by the FTP client was not used because it's too rounded and unclear on the 1000 vs 1024 distinction.)

The following CPE devices have been tested:

CPE Device	Notes
Netopia R7200	Motherboard firmware version 4.7.2 SDSL/ATM wanlet fw v1.0.30
Netopia 4652	Firmware version 5.4.1
Harhan OSDCU	Both N1L2CS and L2CNF1
Paradyne iMarc 9783-C	

The Netopia routers have been included as a control case because they are a ‘‘standard’’ CPE choice used by most ‘‘normal’’ SDSL users, hence they make a reasonable baseline against which other CPE options should be compared in terms of performance. (The other ‘‘standard’’ CPE choice for Nokia SDSL, namely Flowpoint/EN/Siemens routers, have not been included in this study because they are so extremely difficult to configure.)

Whereas the Netopia routers go all the way from SDSL/ATM to Ethernet, our OSDCU converts SDSL/ATM to HDLC instead, hence performance comparison requires that it be tested together with some HDLC-to-Ethernet router, making a two-piece CPE solution. A Cisco 1601 router has been used for that function in this series of experiments, with the following hardware and software configuration:

```
Router#show version
Cisco Internetwork Operating System Software
IOS (tm) 1600 Software (C1600-NY-L), Version 12.2(26c), RELEASE SOFTWARE (fc1)
Copyright (c) 1986-2007 by cisco Systems, Inc.
Compiled Mon 30-Jul-07 16:47 by ccai
Image text-base: 0x0803FBFC, data-base: 0x02005000

ROM: System Bootstrap, Version 11.1(7)AX [kuong (7)AX], EARLY DEPLOYMENT RELEASE SOFTWARE (fc2)
ROM: 1600 Software (C1600-BOOT-R), Version 11.1(7)AX, EARLY DEPLOYMENT RELEASE SOFTWARE (fc2)

Router uptime is 6 hours, 3 minutes
System returned to ROM by power-on
System image file is "flash:c1600-ny-l.122-26c.bin"

cisco 1601 (68360) processor (revision C) with 16384K/2048K bytes of memory.
Processor board ID 04169326, with hardware revision 00000000
Bridging software.
X.25 software, Version 3.0.0.
1 Ethernet/IEEE 802.3 interface(s)
1 Serial(sync/async) network interface(s)
System/IO memory with parity disabled
2048K bytes of DRAM onboard 16384K bytes of DRAM on SIMM
System running from FLASH
7K bytes of non-volatile configuration memory.
8192K bytes of processor board PCMCIA flash (Read ONLY)

Configuration register is 0x2102

Router#show config
Using 736 out of 7506 bytes
!
version 12.2
service timestamps debug uptime
service timestamps log uptime
no service password-encryption
service udp-small-servers
service tcp-small-servers
!
hostname Router
!
boot system flash flash:c1600-ny-l.122-26c.bin
!
ip subnet-zero
!
!
!
!
interface Ethernet0
 ip address 172.23.41.1 255.255.255.0
```

```
 no ip route-cache
 keepalive 300
!
interface Serial0
 no ip address
 encapsulation frame-relay IETF
 no ip route-cache
 no keepalive
 no fair-queue
!
interface Serial0.1 point-to-point
 ip address 172.23.3.41 255.255.255.0
 no ip route-cache
 frame-relay interface-dlci 16
!
ip classless
ip route 0.0.0.0 0.0.0.0 Serial0.1 permanent
no ip http server
!
no cdp run
!
line con 0
 exec-timeout 0 0
line vty 0 4
 login
!
end

Router#
```

Because it is known from theory that a CPE solution involving an ATM-to-HDLC Layer 2 converter cannot fairly compete with a native ATM router on the metric of latency, it was prudent to compare our OSDCU not only against ‘‘standard’’ Netopia routers, but also against some pre-existing device from another vendor that performs the same function of converting Nokia SDSL/ATM to V.35/HDLC. Such devices are highly obscure by nature, and to our knowledge the only existing one that supports Nokia’s SDSL/2B1Q is Paradyne iMarc 9783-C. We have thus added it to the set of CPE devices to be tested, used with exactly the same C1601 router as the OSDCU under test.

The specific mode of ATM-to-HDLC Layer 2 conversion for this series of experiments has been FRF.8, in both OSDCU and Paradyne iMarc experiments. With this conversion mode the ATM encapsulation served by the SSP remains the same as when using ‘‘standard’’ Netopia CPE (routed RFC 1483), and the C1601 router saw Frame Relay (routed RFC 1490). The C1601 configuration shown above (which has FR LMI aka keepalive disabled) was used with the OSDCU; the configuration used with the 9783-C was the same with one exception: ANSI LMI (DTE role) had to be enabled.

The CONFIG.TXT parameters on the OSDCU had been set as follows:

```
terminal_type=R
preact_type=AutoBaud
ccitt_114=BCLK
ccitt_115=/16
obey_dtr=no
```

OSDCU experiments had been performed using AutoBaud on the D50 for the sake of convenience. Because none of the ‘‘mainstream’’ CPE devices support AutoBaud when configured for the Nokia flavor, switching speeds during testing required reconfiguration on the CPE device in addition to the D50 and the C3640 ‘‘ubr’’ line. 4 speeds were tested: 384, 768, 1152 and 1536 kbps. These correspond to the 5 data rates offered by the

SDSL8 line card, with the exception of 192 kbps which wasn't tested because 384 kbps was deemed to be the lowest speed of practical interest.

Because the performance observed with the OSDCU and 9783-C devices is affected not only by the DSU but also by the C1601 router, an additional control was needed to characterize the performance of the C1601 serving as a CPE device by itself. In addition to the FastEthernet and ATM interfaces used in the main series of experiments, the C3640 router serving as our SSP has classic WIC-1T serial interfaces available. In the special control case the connection between the SSP and the C1601 CPE router was a direct synchronous serial bit pipe between the HDLC interfaces of the two Ciscos, rather than ATM delivered through the D50 as in all other experiments. The encapsulation was still ‘‘frame-relay IETF’’ with no keepalive, and the clocked bit pipe was provided by a pair of OSDCUs in the bit-transparent IFCTF SDSL mode. The bit rate was thus precisely controlled, and was set to 384, 768, 1152 and 1536 kbps for comparison with Nokia SDSL/ATM performance.

Results

Table 1: Minimum rtt for small packets, ms

SDSL speed	Netopia R7200	Netopia 4652	OSDCU N1L2CS	OSDCU L2CNF1	9783-C	C1601 control
384 kbps	9.896	9.676	36.944	32.096	18.728	11.267
768 kbps	7.079	6.102	24.351	22.103	13.536	9.229
1152 kbps	6.070	5.096	20.507	18.166	11.880	8.485
1536 kbps	5.425	4.157	———	16.272	11.005	8.122

Table 2: Average rtt for small packets, ms

SDSL speed	Netopia R7200	Netopia 4652	OSDCU N1L2CS	OSDCU L2CNF1	9783-C	C1601 control
384 kbps	10.922	10.586	42.152	37.665	19.486	14.226
768 kbps	7.585	6.641	29.238	26.487	17.113	10.241
1152 kbps	6.288	5.395	22.058	20.049	12.260	10.202
1536 kbps	5.620	4.670	———	20.913	12.425	9.245

Table 3: Maximum rtt for small packets, ms

SDSL speed	Netopia R7200	Netopia 4652	OSDCU N1L2CS	OSDCU L2CNF1	9783-C	C1601 control
384 kbps	12.280	11.668	56.800	54.370	20.330	43.335
768 kbps	8.633	7.380	48.181	46.375	37.108	28.283
1152 kbps	7.078	6.142	24.986	21.611	14.616	30.794
1536 kbps	6.620	5.566	———	37.894	33.559	29.002

Notes:

1. N1L2CS is the SCC-only version of the OSDCU operational code; L2CNF1 is the SCC/FPGA hybrid version.
2. 1536 kbps connection with N1L2CS had not been attempted at all given the unacceptably poor performance at 1152 kbps, as shown on the following pages.

Table 4: Minimum rtt for large packets, ms

SDSL speed	Netopia R7200	Netopia 4652	OSDCU N1L2CS	OSDCU L2CNF1	9783-C	C1601 control
384 kbps	87.145	78.083	148.711	135.961	115.356	72.968
768 kbps	53.798	43.503	91.975	80.315	65.641	42.368
1152 kbps	41.706	31.659	———	62.431	47.950	32.177
1536 kbps	35.374	25.394	———	54.893	39.028	27.036

Table 5: Average rtt for large packets, ms

SDSL speed	Netopia R7200	Netopia 4652	OSDCU N1L2CS	OSDCU L2CNF1	9783-C	C1601 control
384 kbps	87.794	78.941	157.223	142.781	116.177	77.573
768 kbps	54.094	44.358	96.137	86.884	67.078	44.035
1152 kbps	42.043	32.121	———	68.469	52.581	35.082
1536 kbps	35.721	25.742	———	61.167	40.661	31.513

Table 6: Maximum rtt for large packets, ms

SDSL speed	Netopia R7200	Netopia 4652	OSDCU N1L2CS	OSDCU L2CNF1	9783-C	C1601 control
384 kbps	88.423	79.915	185.437	163.102	120.068	112.872
768 kbps	55.389	44.894	117.521	153.565	109.159	60.519
1152 kbps	42.945	32.581	———	89.332	71.322	50.817
1536 kbps	36.035	26.230	———	80.307	53.201	50.070

Notes:

1. Attempting a large packet ping with N1L2CS at 1152 kbps instantly killed the DSU with an SCC Tx underrun.
2. 1536 kbps connection with N1L2CS had not been attempted at all.

Table 7: FTP download times, s

SDSL speed	Netopia R7200	Netopia 4652	OSDCU N1L2CS	OSDCU L2CNF1	9783-C	C1601 control
384 kbps	84.1	84.4	84.5	84.5	84.6	74.8
768 kbps	42.2	42.2	42.3	42.3	42.3	37.4
1152 kbps	28.2	28.1	———	35.8	28.2	24.9
1536 kbps	21.1	21.1	———	———	21.3	18.7

Table 8: FTP download throughput, kbyte/s

SDSL speed	Netopia R7200	Netopia 4652	OSDCU N1L2CS	OSDCU L2CNF1	9783-C	C1601 control
384 kbps	40.371	40.228	40.180	40.180	40.133	45.391
768 kbps	80.455	80.455	80.265	80.265	80.265	90.781
1152 kbps	120.398	120.826	———	94.838	120.398	136.354
1536 kbps	160.910	160.910	———	———	159.400	181.562

Notes:

The OSDCU was unable to handle the FTP download without crashing at 1152 kbps with N1L2CS and at 1536 kbps with L2CNF1.

Table 9: FTP upload times, s

SDSL speed	Netopia R7200	Netopia 4652	OSDCU N1L2CS	OSDCU L2CNF1	9783-C	C1601 control
384 kbps	82.1	81.9	82.1	82.7	82.1	73.3
768 kbps	41.4	41.0	50.1	41.2	41.4	36.6
1152 kbps	27.6	27.6	———	27.6	27.3	24.2
1536 kbps	20.7	20.7	———	20.7	20.7	18.2

Table 10: FTP upload throughput, kbyte/s

SDSL speed	Netopia R7200	Netopia 4652	OSDCU N1L2CS	OSDCU L2CNF1	9783-C	C1601 control
384 kbps	41.355	41.456	41.355	41.055	41.355	46.319
768 kbps	82.010	82.810	67.769	82.408	82.010	92.765
1152 kbps	123.015	123.015	———	123.015	124.367	140.298
1536 kbps	164.020	164.020	———	164.020	164.020	186.550

Notes:

Upload tests at 1152 and 1536 kbps had not been attempted with N1L2CS because the performance degradation from CPU starvation is already clearly evident at 768 kbps. N1L2CS running at 1152 kbps had also been observed to underrun in an apparently spontaneous manner, with no deliberate test activity in progress; that would have made FTP upload testing more difficult to perform.

Discussion

The first observation that can be made from the results of these experiments is that at least as long as one sticks to the lower speeds of Nokia SDSL which the OSDCU handles comfortably (192 and 384 kbps with both versions and also 768 kbps with L2CNF1), there is no statistically significant difference in the user-visible data transfer throughput between the OSDCU and other CPE devices for the same SDSL service. (The slight fluctuations in the numbers can be reasonably assumed to be measurement uncertainty and/or imperfect test repeatability.) In particular, there are no effects such as packet drops that would result in the OSDCU giving worse throughput than other available CPE devices.

Unfortunately however, the OSDCU fares significantly worse on the latency metric. Although it was expected that a two-piece CPE solution would not be able to compete in terms of latency with Netopia routers which terminate SDSL/ATM more directly, it was disappointing to see that even the improved SCC/FPGA hybrid version still adds more latency than the functionally equivalent, but offensively proprietary Paradyne iMarc 9783-C. While moving the SDSL Tx bit stream generation function from the SCC into autonomous logic (FPGA) did reduce the added latency as expected, the measured improvement is significantly less than we had hoped.

So far the best performers we've seen in terms of lowest latency are Netopia 4652 for the SDSL/ATM to Ethernet functionality and iMarc 9783-C for the SDSL/ATM to V.35/HDLC functionality. The latency numbers observed in the C1601 control case (direct HDLC connection to the SSP, bypassing ATM) are relatively high, and are easily blamed on the age and ''classicness'' of the C1601. The maximum rtt numbers (Tables 3 and 6) show frequent high spikes in all test cases that involved the C1601; we assume that these spikes result from some internal happenings in the C1601.

The SCC-only version of the Nokia SDSL/ATM to HDLC converter on the OSDCU is only able to keep up with Nokia SDSL speeds up to 768 kbps, and even at that ''top'' speed it already shows a sign of performance degradation as seen in the FTP upload throughput numbers. Running at 768 kbps, N1L2CS keeps up fine with the downstream traffic, but is unable to fill the upstream direction of the pipe like other CPE devices do. The effect observed here is CPU cycle starvation on the MC68302: preparing bits for transmission on the SDSL side in N1L2CS involves a heavy amount of CPU-mediated processing, and the responsible code runs at the lowest relative priority. As the SDSL data rate goes up, the CPU work load increases, and at 768 kbps this low-priority code no longer gets enough cycles to do all of its work. Increasing the SDSL data rate even further to 1152 kbps causes the more critical interrupt handler to fail as well, making the DSU unusable.

With L2CNF1 the SDSL Tx direction is much more robust because the previously CPU-mediated CRC computations have been moved into autonomous logic, and the FPGA IRQ wiring provides a more robust interrupt nesting structure. The FTP upload throughput numbers prove that the upstream direction remains perfectly healthy and undegraded all the way up to 1536 kbps. However, the new limiting factor becomes the downstream Rx side.

L2CNF1 downstream Rx performance remains undegraded up to 768 kbps, but at 1152 kbps the measured FTP download performance is suddenly noticeably worse than that obtained with any other CPE device on the same SDSL service. How can this be, considering that the shape of the downstream traffic is determined by the SP side and that the synthetic HDLC link between the OSDCU and the C1601 was made to run at a higher rate so that no packet drops ought to ever occur at the DSU in the downstream direction? The answer appears to be some internal limits in the MC68302 IMP's CP block: what happens is that Rx packets are dropped as they are ready to be transmitted on the HDLC side (handled by SCC2), even though there is plenty of bit bandwidth available on that HDLC link. Going up to 1536 kbps causes SCC1 Rx to suffer an overrun.

Finally, the FTP throughput numbers from the C1601 control case, as compared to those established in the Nokia SDSL/ATM experiments, provide a clear illustration of the difference in bit efficiency between HDLC and ATM. The 1536 kbps case specifically illustrates the difference between a T1 line and an ''equivalent'' 1.5 Mbps SDSL service.

Conclusion

This series of experiments has made it clear that our OSDCU is unfortunately not viable as a CPE device for the higher speed tiers of Nokia SDSL. It does perform acceptably at the lower speeds, and exhibits no degradation in data transfer throughput compared to other available CPE devices, but the CPE-added latency is higher than that exhibited by any of the competitors.

It needs to be remembered that the OSDCU was originally intended to provide *proof of concept* connectivity to various SDSL flavors; supporting the highest speed tier of each flavor or providing competitive performance weren't and aren't required for this proof of concept. The OSDCU has actually exceeded the minimal requirements for the proof of concept in that it performs well enough for Harhan to have deployed it operationally on our own live SDSL connection at 384 kbps; the fact that it can't handle the higher speeds well is no one's fault. Our Nokia SDSL implementation approach was novel (different from anything observed in the examination of the insides of various pre-existing CPE products), and it wasn't obvious ahead of time whether it would perform well or not. We had proceeded with the hardware design and build nonetheless because there were other motivating reasons at the time.

But now we know that the OSDCU hardware design is effectively a dead end when it comes to SDSL/ATM. Raising the system clock frequency is not a promising prospect: we currently run at 25 MHz (all experiments in this study have been performed on a 25 MHz OSDCU board), and this frequency seems to be the reasonable maximum for a circuit of this sort to function reliably. The MC68302 processor is available in a 33 MHz speed grade, but only in a different package that would require massive changes to the PCB layout. That would be a massive outlay of effort for questionable gain. Replacing the EPF10K30A FPGA with a different part having enough internal RAM to implement both Rx and Tx for the Nokia flavor in autonomous logic would most likely make 1152 kbps perform at full throughput in both directions, but it is unlikely to provide the same flawless performance at 1536 kbps or to bring down the high added latency.

One needs to remember however, that as feeble as it may be, the OSDCU is the only device currently in existence that provides connectivity to Nokia SDSL in a fully open source manner, friendly to the high-IQ user, and it does provide acceptable performance at the lower speeds. Furthermore, this series of experiments has proven that it *is* worth building the next batch of OSDCU boards with the EPF10K30A FPGA populated: it adds 768 kbps to the repertoire of Nokia SDSL speeds with acceptable performance, and it does reduce the added latency somewhat, even if not by as much as we had hoped.

In the longer term the goal is to create two fully open source, elitist-friendly SDSL/ATM CPE devices: one a router with full IP termination, the other a converter to HDLC over V.35 or EIA-530 like the OSDCU and 9783-C devices. The former needs to perform no worse than Netopia 4652, the latter no worse than 9783-C. In the opinion of this hacker the way to achieve that goal with a high certainty of success would be to use a hardware design that is substantially similar to that of the products whose performance we seek to match. Both Netopia 4652 and iMarc 9783-C use PowerQUICC processors (different feature subsets of MPC860P), hence we reason that an open source hardware design based on Freescale MPC866 (the feature and performance superset of the entire MPC85x/86x line) ought to have the capability to perform as well as those existing commercial devices. We have plans for a modular MPC866-based WAN hacking platform, but they are not presently on a high-priority schedule.