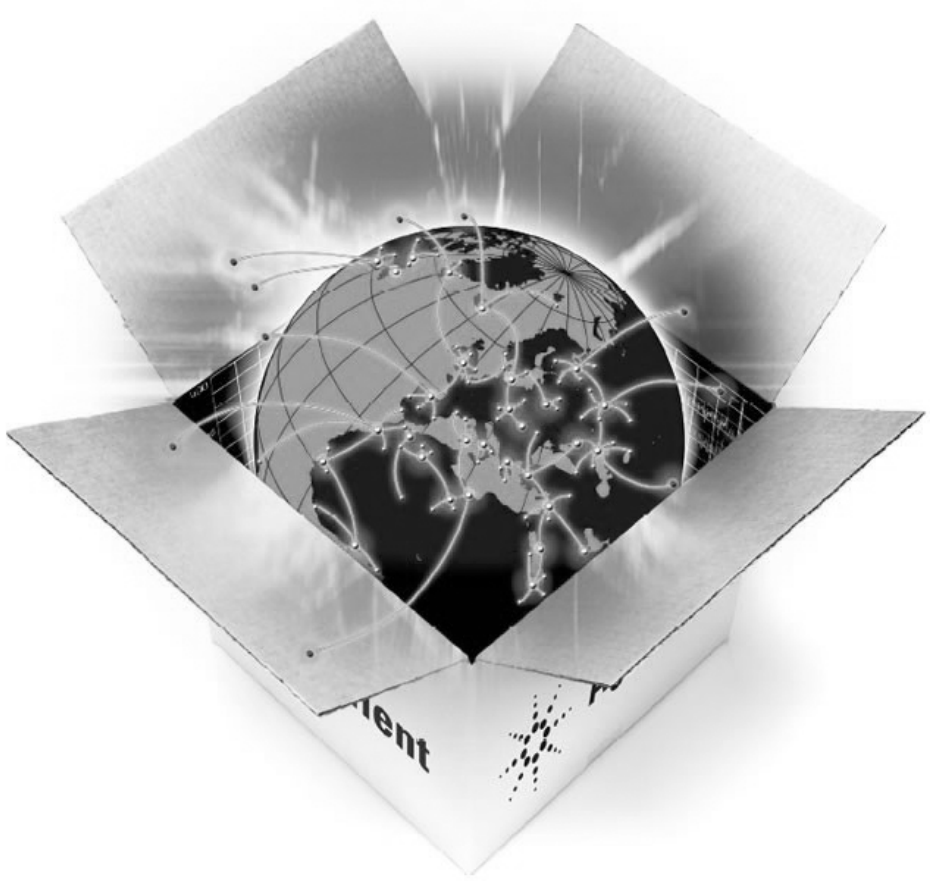
Agilent
Incremental Redundancy in EGPRS

Application Note



EGPRS uses incremental redundancy to achieve maximum efficiency over the air interface. This paper explains how incremental redundancy operates and discusses how it can be used to provide maximum throughput of data by changing coding and puncture schemes as necessary.

This paper illustrates methods for testing incremental redundancy and wireless device receiver performance. These methods allow users/developers to verify operation and accurately see how the device performs under differing air interface conditions in terms of general operation, data throughput, and memory organization.

Agilent Technologies

Table of Contents

Incremental Redundancy

Incremental redundancy is the system used by EGPRS to get maximum performance out of the available bandwidth. By using convolutional coding the system can add protection to the data. However, this coding adds correction data to the original data that carries no actual information (redundancy). This has the effect of reducing data throughput/user bandwidth.

How incremental redundancy works

Figure 1 shows the different states in EGPRS incremental redundancy. A data block is sent to the mobile. If there are no errors then the block will be passed up to the next layer of protocol for processing. However, if the block is received in error then the mobile will send an automatic repeat request (ARQ) to the base station. The base station will then retransmit the block using a different puncturing scheme. This block will be recombined with the first block increasing the amount of redundancy and giving the mobile/base station a better chance of being able to recover from the errors.

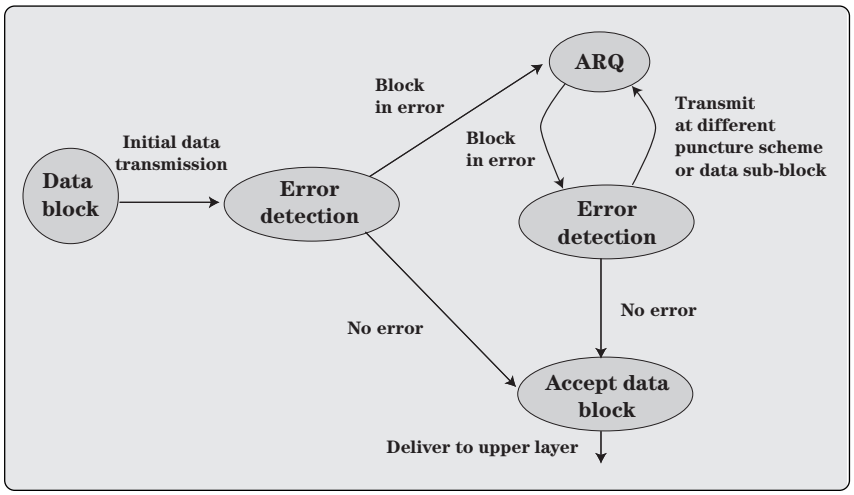


Figure 1. Incremental redundancy simplified state diagram.

To illustrate how incremental redundancy works, Figure 2 shows how original data is convolutionally encoded at a one third rate, meaning for every bit that goes into the coder, three come out. The data is then punctured into three puncture schemes: 1st Tx, 2nd Tx, and 3rd Tx.

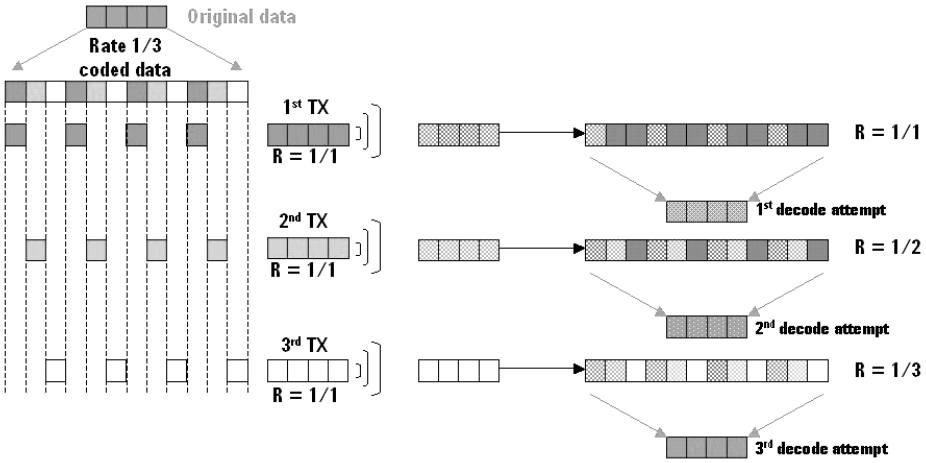


Figure 2. Example using MCS-9.

The data is first sent over the air (OTA) using one of the puncture schemes. In this demonstration the 1st Tx has a code rate of one. For every original data bit sent, there is one bit sent OTA. The demodulated data at the mobile is padded with data back to the size after convolutional encoding. The data used for padding, or bit stuffing, is not relevant since the original data is unknown at the receiving end (there is a 50 percent chance of getting the data correct). This is then viterbi decoded to give the four original bits with some probability of error. The first decode attempt is then checked against the cyclic redundancy check (CRC). If the block is in error it is stored and the automatic repeat request (ARQ) is sent to request a retransmission.

When the retransmission is sent, a different puncture scheme (2nd Tx) is used. To simpilfy this illustration, four bits will be sent OTA. When they arrive at the mobile, they will be recombined with the bits from the first transmission giving a code rate of one half. This means that for every original data bit there now have been two bits sent OTA. This gives the mobile a greater chance of correctly decoding the data. Once the data has been recombined and padded, it is viterbi decoded and then checked against the CRC. If the block is still in error, it is stored and the ARQ is sent to request a retransmission once more.

This time the retransmission is sent using the 3rd Tx puncture scheme. The four bits are sent OTA and are recombined at the mobile with the bits from the first and second transmissions. This generates an OTA code rate of one third, since for every original data bit there have been three data bits sent OTA. This maximizes the probability of correctly decoding the data. After viterbi decoding, if the block is still in error it is possible to retransmit using the same coding scheme and repeat the puncture schemes, or switch to a more robust coding scheme and increase the redundancy still further. Mobile coding schemes 1-4 (MCS 1-4) use GMSK modulation, which is less likely to incur errors during transmission.

Mobile coding schemes and families

As can be seen in Figure 3, the mobile coding schemes are arranged into families. Looking at Family A, MSC-9/6/3, the transmission starts in MCS-9 and two 74-octet blocks are sent. If either or both of these blocks are not received OTA successfully, even after being retransmitted using different puncture schemes, it is possible to send these blocks individually using coding scheme MCS-6. If this is still unsuccessful, then it is possible to send the 74-octet block over two 37-octet MCS-3 blocks. The number of retransmissions at each MCS will be determined by the network. This not only provides the ability to add redundancy by using different puncture schemes, it also allows dropping to a stronger code to perform the retransmission. Furthermore, since the air interface is volatile at best, it is unkind to modulation and can cause errors in the data stream. To help combat this problem EGPRS has two modulation types. The first type, MCS-5 through MCS-9, uses $\frac{3\pi}{8}$ 8 PSK. The second group, MCS-1 through MCS-4, uses Gaussian minimum shift keying (GMSK) modulation. The later type is much less prone to errors since it only has one bit per symbol as opposed to three bits per symbol in the same bandwidth.

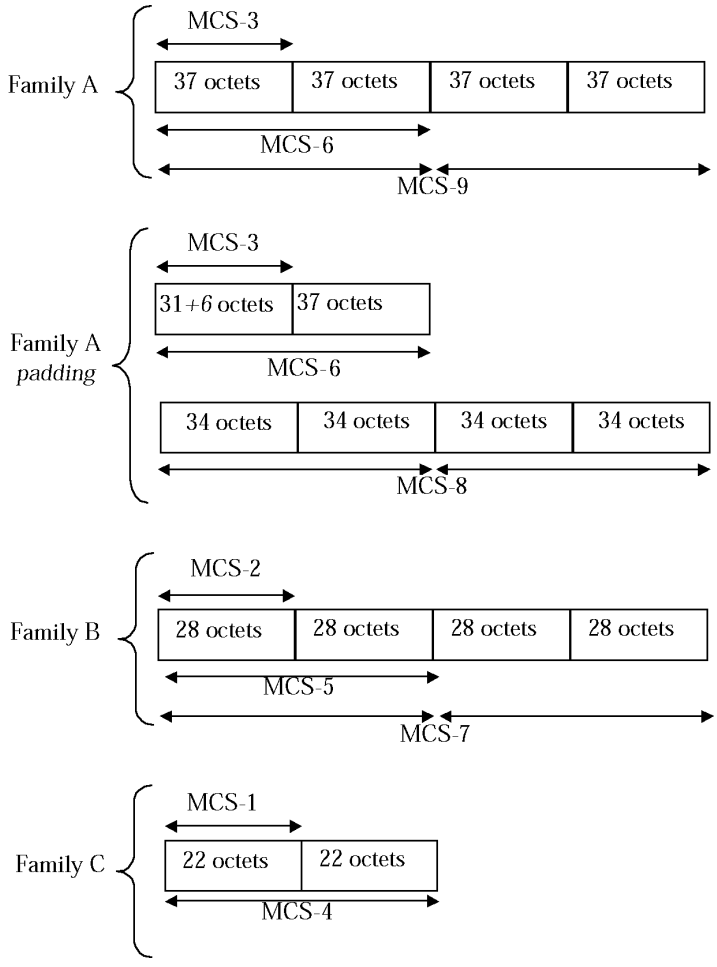


Figure 3. MCS families.

The special case amongst these families is Family A padding, where MCS-8 has a differing block size than the rest of its family members. In this case the block's lower coding schemes use padding to make the 34-octet blocks fit into their 37-octet block size.

The sections below illustrate how the coding schemes in Family A make up the blocks, coding, and puncturing used for transmission.

MCS-9 coding scheme

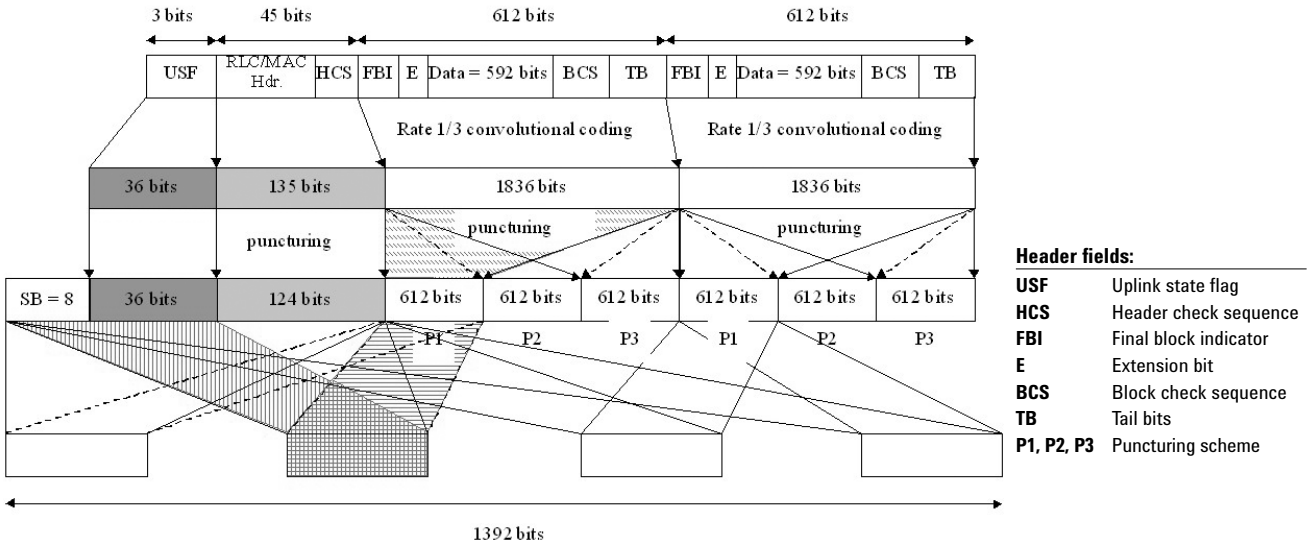


Figure 4. MCS-9 block coding and puncturing structure.

This diagram shows how two 612-bit blocks and header information are coded into four blocks, which will be transmitted over four frames. Notice that one third rate convolutional coding is being used; yet only one third of the coded bits are being sent, giving a code rate of one. As previously explained, the other bits from puncture schemes, P2 and P3, will not be transmitted unless P1 was not received correctly. However, sometimes a second puncture scheme maybe sent if only one of the MCS-9 blocks in the transmission is being used.

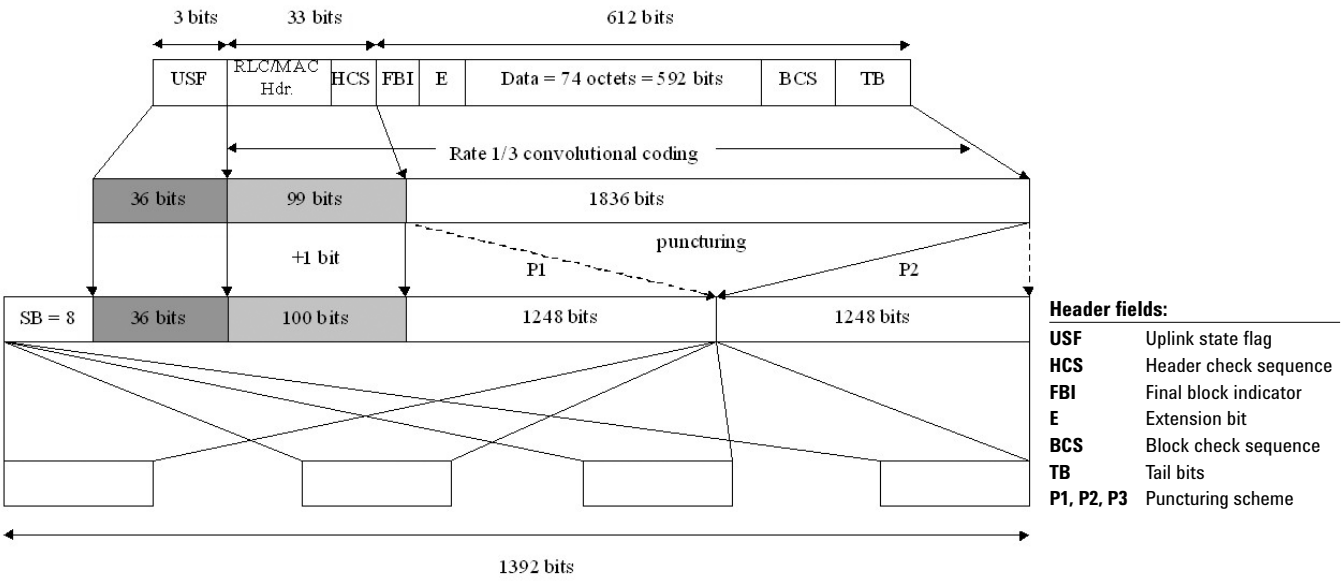


Figure 5. MCS-6 block coding and puncturing structure.

This figure shows the same information for coding scheme MCS-6. Now there is only one 612-bit block (this could be one of the MCS-9 blocks that was previously received in error). While it still uses a one third rate convolutional coding, MCS-6 has only two puncturing schemes so that the data transmitted OTA has more redundant data than MCS-9.

MCS-3 coding scheme

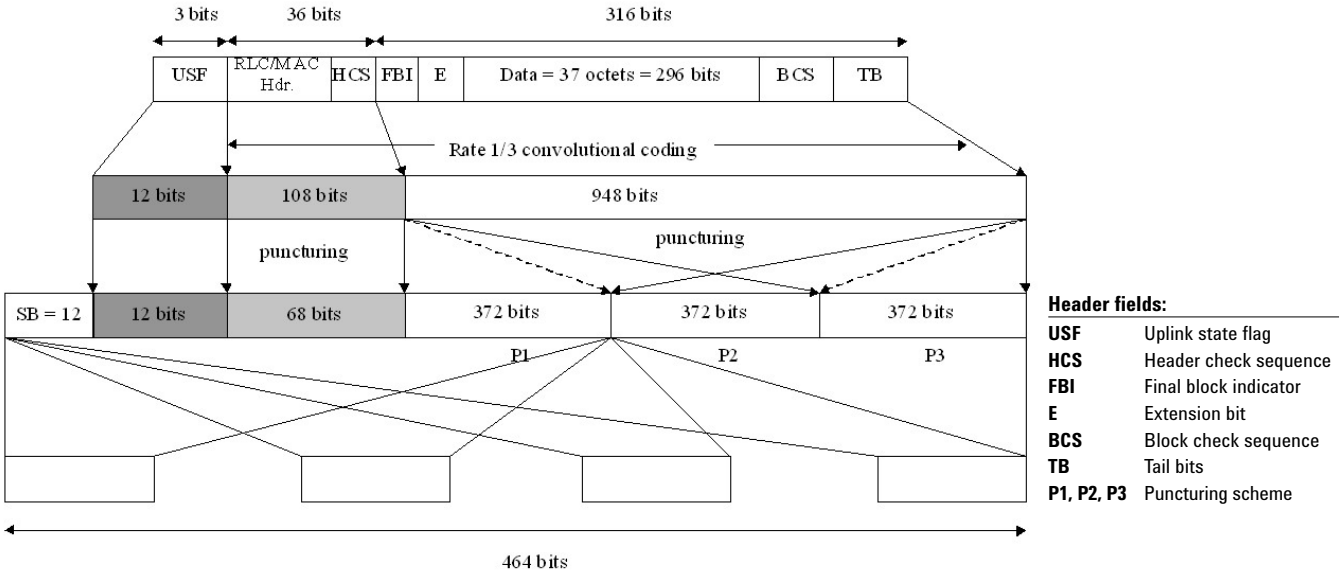


Figure 6. MCS-3 block coding and puncturing structure.

MCS-3 only has 316 data bits and three puncturing schemes, however this coding scheme now uses GMSK modulation. There is only one puncturing scheme sent OTA at a time. If this coding scheme is used to re-transmit an MCS-6 block, two MCS-3 blocks will need to be sent OTA.

Incremental redundancy, RLC windows, and full operation

Clearly, if every block of data was sent over the air and then either an acknowledgement (ack) or negative acknowledgement (nack) was sent back to ask for re-transmission, the system would not be very efficient. This brings in the concept of the radio link control window. Table 1 shows the window sizes relative to the number of timeslots.

Table 1. Allowed window sizes in EGPRS TBF mode for different multislot allocations. (3GPP TS 04.60 V8.7.0 (2000-11)).

Window size	Coding	Timeslots allocated (Multislot capability)							
		1	2	3	4	5	6	7	8
64	00000								
96	00001								
128	00010								
160	00011								
192	00100	Max							
224	00101								
256	00110		Max						
288	00111								
320	01000								
352	01001								
384	01010			Max					
416	01011								
448	01100								
480	01101								
512	01110				Max				
544	01111								
576	10000								
608	10001								
640	10010					Max			
672	10011								
704	10100								
736	10101								
768	10110						Max		
800	10111								
832	11000								
864	11001								
896	11010							Max	
928	11011								
960	11100								
992	11101								
1024	11110								Max
Reserved	11111	x	x	x	x	x	x	x	X

A simple way to think of this is as follows: the window represents the number of blocks that can be sent to the mobile without a nack. Blocks will be sent in sequence with a block sequence number (BSN) until the last block in the sequence or the end of the window. There will be a packet ack/nack sent back to the transmitter that contains a starting sequence number (SSN) and a received block bitmap (RBB). The BSN is then the SSN plus the bitmap position. From this equation the transmitter is able to determine which blocks were received with and without error. Blocks can then be re-transmitted. This re-transmission could be performed using the same, or different, puncture scheme, or even a different coding scheme. The first block that is in error then becomes the beginning of the RLC window, so it could be said that the window slides as blocks are received without error.

Testing Incremental Redundancy Performance

There are obviously two sides to the system where incremental redundancy can come into play, the uplink (transmission) and the downlink (reception). Both offer their own challenges and both have issues that need to be tested and addressed.

Testing downlink operation

There are two ways of simulating an environment where downlink incremental redundancy will come into play. Both methods require an EGPRS system simulator and base station emulator that supports incremental redundancy.

1. during a data transfer, reduce cell power to a point where errors start to enter the data stream and block errors cause re-transmission
2. using a system simulator capable of artificially introducing errors into the data stream. This will simulate a real transmission and blocks or bursts of errors will cause the mobile to request a re-transmission

The first of these methods, while being viable, will not provide any control over the re-transmissions and the ratio of errors in the data stream. The second method offers by far the best solution, especially if the system simulator offers good control over the errors which will be introduced. Ideally, there also needs to be some way of viewing data exchange on both the mobile and the network.

Once there is a method of introducing errors into the data stream, there also needs to be a way of controlling how incremental redundancy and mobile coding scheme switching work together to provide the best possible method for exercising the mobile with the maximum amount of control. There also needs to be a means of viewing the changes take effect. This can be done in two ways:

1. as a reduction of data throughput, or the result
2. by viewing the protocol and literally seeing the changed made to the coding and puncture schemes as they occur over the data transfer

Testing uplink operation

For the testing of uplink incremental redundancy operation, there are similar challenges. It is particularly difficult to introduce errors into the uplink data stream without altering the downlink path. To avoid this, the uplink signal must be moved outside of the acceptable operating range of the system simulator's input. Another, easier alternative is to artificially negatively acknowledge uplink blocks to force uplink retransmission. This way when the mobile receives the ack/nack bitmap from the system simulator, it will retransmit the negatively acknowledged blocks using a different puncture or coding scheme.

What the standards say

3GPP document 45.005 specifies the test for incremental redundancy in section 6.7 as follows:

Support for incremental redundancy reception is mandatory for all EGPRS-capable MSs. In incremental redundancy, RLC mode soft information from multiple, differently punctured, versions of an RLC data block may be used when decoding the RLC data block. This significantly increases the link performance. An EGPRS-capable MS shall, under the conditions stated in the below table, achieve a long-term throughput of 20 kbps per time slot (see note), measured between LLC and RLC/ MAC layer.

Required throughput	20,0 kbps per timeslot
Propagation conditions static, input level	–97,0 dBm
Modulation and coding scheme	MCS-9
Acknowledgements polling period	32 RLC data blocks
Roundtrip time	120 ms
Number of time slots	Maximum capability of the MS
Transmit window size	Maximum for the MS timeslot capability

Note: This corresponds to an equivalent block error rate of approximately 0.66 using the prescribed MCS-9

Strictly speaking this can be tested by using a system simulator that simply supports incremental redundancy. This approach is fine from a receiver conformance testing perspective; however it does not provide the developer with any indication of what, if anything, is wrong with the device, nor does it provide any insight into algorithmic errors.

Potential Issues for Mobile Device and Protocol Stack Developers

The major issues from a mobile perspective are memory and processor issues. The mobile must be able to store large quantities of data. Since the data will not be transmitted one block at a time, a mobile that handles eight slots must accept up to the maximum window size of 1024 blocks (see *Incremental redundancy, RLC windows, and full operation* on page 10.) This means that the mobile must be able to buffer this data and then potentially buffer it all twice more as the data is sent using different puncture schemes. The data can also be transmitted subsequently in different coding schemes from the same family and recombined, adding further complexity and increasing memory still further. However, the mobile can indicate to the network that it is out of memory by setting a flag in the packet downlink ack/nack message. If this flag is set then re-transmissions will only be performed in the same coding and puncture scheme, allowing mobile to stop buffering data.

Due to all of the viterbi decoding and buffering, the mobile has to work hard computationally. These calculations have to be performed real time if throughput is to be maintained. Control over the amount of uplink and downlink errors is essential if the developer wishes to verify that the mobile has enough computing power to keep up.

A Brief Overview of What the Agilent 8960 Has to Offer

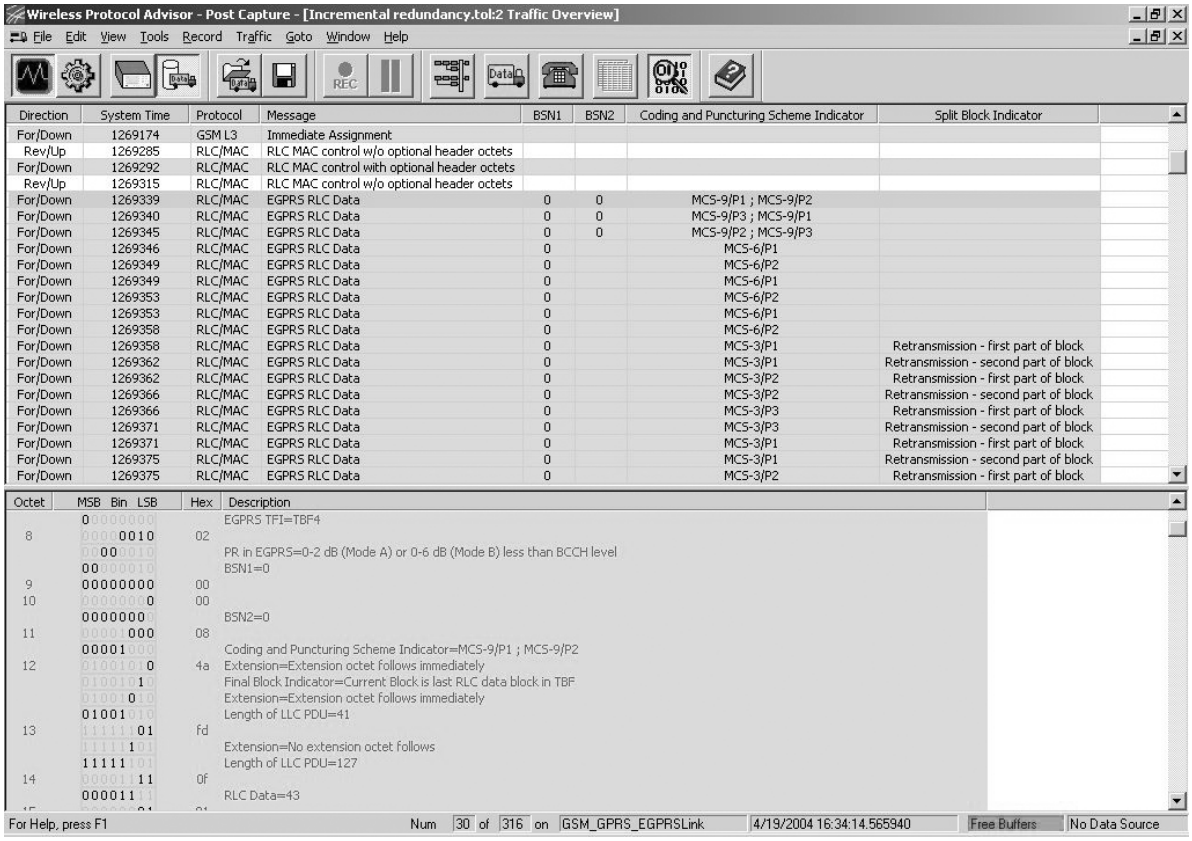
Agilent offers an EGPRS lab application for the 8960 wireless test set and Wireless Protocol Advisor software, which logs all OTA protocol. This configuration provides an extremely powerful analysis and testing tool for incremental redundancy. These instruments give the user independent control over all of the general link parameters, such as the coding scheme for uplink and downlink. In addition to this, incremental redundancy can be turned on and the 8960 will automatically switch puncture schemes for retransmission. Additionally, the user can elect to change coding schemes and use automatic MCS-switching based on the number of re-transmissions, which is settable by the user. Moreover, the 8960 allows segmentation of blocks and retransmission using MCS 1-4. This alone provides a powerful tool for the verification of incremental redundancy from a functional viewpoint.

The Agilent EGPRS lab application takes this much further. It provides functions for artificially controlling the uplink and downlink transmissions and retransmissions. For the uplink, the EGPRS lab application gives the user the ability to set a percentage value of uplink blocks to negatively acknowledge, thus forcing the mobile to retransmit these nacked blocks. For the downlink, even more control is provided. At Layer 1 the EGPRS lab application is able to provide the user with the ability to corrupt the downlink blocks on a burst-by-burst basis and allow the following parameters to be set:

Layer 1 corruption	on/off
Sequence length	1-2048
Blocks in sequence to corrupt	1-2048
Corruption applied to burst 1	on/off
Corruption applied to burst 2	on/off
Corruption applied to burst 3	on/off
Corruption applied to burst 4	on/off
First symbol to corrupt	0-147
Number of symbols to corrupt	1-148
Corruption pattern	all zero, all ones, invert

These parameters allow designers to control exactly how the corruption will be applied at layer one and to which RF burst this will be applied. Also, this method of corruption and the ability to choose the corruption pattern gives the user the ability to clearly see from the mobile unit's protocol logs, which bursts have been corrupted by putting a clear corruption pattern such as all ones or zeros into the data.

Coupled with this is the lab applications counterpart: the Wireless Protocol Advisor (WPA.) WPA supplies the ability to see all of the data exchange in real time with full decodes of the messages.



For example, using WPA triggers can be set up so that logging will only start when an MCS switch occurs. For details on the WPA please consult the Agilent Web site.

Agilent Technologies' Test and Measurement Support, Services, and Assistance
Agilent Technologies aims to maximize the value you receive, while minimizing your risk and problems. We strive to ensure that you get the test and measurement capabilities you paid for and obtain the support you need. Our extensive support resources and services can help you choose the right Agilent products for your applications and apply them successfully. Every instrument and system we sell has a global warranty. Support is available for at least five years beyond the production life of the product. Two concepts underlie Agilent's overall support policy: "Our Promise" and "Your Advantage."

Our Promise
Our Promise means your Agilent test and measurement equipment will meet its advertised performance and functionality. When you are choosing new equipment, we will help you with product information, including realistic performance specifications and practical recommendations from experienced test engineers. When you use Agilent equipment, we can verify that it works properly, help with product operation, and provide basic measurement assistance for the use of specified capabilities, at no extra cost upon request. Many self-help tools are available.

Your Advantage
Your Advantage means that Agilent offers a wide range of additional expert test and measurement services, which you can purchase according to your unique technical and business needs. Solve problems efficiently and gain a competitive edge by contracting with us for calibration, extra-cost upgrades, out-of-warranty repairs, and onsite education and training, as well as design, system integration, project management, and other professional engineering services. Experienced Agilent engineers and technicians worldwide can help you maximize your productivity, optimize the return on investment of your Agilent instruments and systems, and obtain dependable measurement accuracy for the life of those products.

Agilent Email Updates

www.agilent.com/find/emailupdates
Get the latest information on the products and applications you select.

Agilent T&M Software and Connectivity
Agilent's Test and Measurement software and connectivity products, solutions and developer network allows you to take time out of connecting your instruments to your computer with tools based on PC standards, so you can focus on your tasks, not on your connections. Visit **www.agilent.com/find/connectivity** for more information.

By internet, phone, or fax, get assistance with all your test & measurement needs

Phone or Fax
United States:
(tel) 800 829 4444
Canada:
(tel) 877 894 4414
(fax) 905 282 6495
China:
(tel) 800 810 0189
(fax) 800 820 2816
Europe:
(tel) (31 20) 547 2323
(fax) (31 20) 547 2390
Japan:
(tel) (81) 426 56 7832
(fax) (81) 426 56 7840

Korea:
(tel) (82 2) 2004 5004
(fax) (82 2) 2004 5115
Latin America:
(tel) (305) 269 7500
(fax) (305) 269 7599
Taiwan:
(tel) 0800 047 866
(fax) 0800 286 331
Other Asia Pacific Countries:
(tel) (65) 6375 8100
(fax) (65) 6836 0252
Email: tm_asia@agilent.com

Online Assistance:
www.agilent.com/find/assist

Product specifications and descriptions in this document subject to change without notice.

Printed in USA, May 18, 2004
5989-1156EN

Agilent Technologies