A Real-time GSM Link Adaptation Hardware Demonstrator.

J Dunlop*, J Pons*, J Gozalvez*, and P. Atherton**

*Department of Electronic and Electrical Engineering, University of Strathclyde
204 George St, Glasgow G1-1XW, Scotland
Tel: +44 141 548 2381, Fax: +44 141 552 4968
Email: j.dunlop@eee.strath.ac.uk

**Motorola Semiconductors Product Sector
East Kilbride, Glasgow G75 0TG, Scotland

Abstract – Link Adaptation has been identified as a key technology for evolved GSM systems such as GPRS and EDGE. Previous work on Link Adaptation has proven the benefits of the technology in coverage and quality. This paper presents a Link Adaptation hardware implementation which allows real-time assessment of the perceived voice quality.

I. Introduction

Adaptive channel coding schemes, where a transmission code is chosen from a set of possible codes of varying robustness depending on the quality of the channel, have been widely reported in the literature [1]. The basis of Link Adaptation (LA) is to assess the channel conditions in order to select a transport mode from a set of possible implementations. An example of the potential of LA for GSM speech transmission was presented in [2] where the adaptation is achieved using the HR GSM speech codec with extended error protection over a TCH/FS channel when bad link conditions occur. The characteristic of such a codec is illustrated in figure 1.

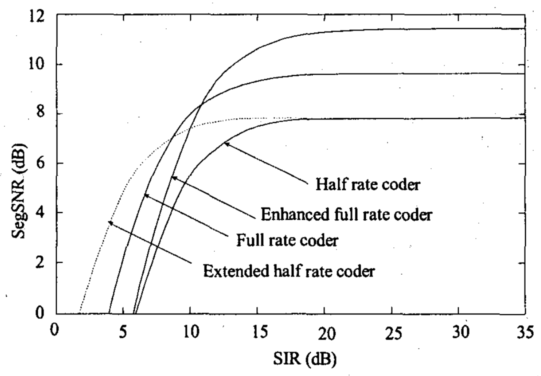

Figure 1 Performance of GSM codecs

The GSM recommendations specify quality thresholds for frame erasure rate (FER), residual bit error rate for class Ib bits (RBER Ib) and residual bit error rate for class I bits (RBER II). Figure 2 shows the results obtained for frame erasure rate for the Half Rate coder (GSM – HR) and the Extended Half Rate coder (Extended – HR) as a function of signal to interference ratio (SIR). It should be noted that the Extended – HR characteristic provides a margin of approximately 5 dB in SIR which may be used, for example, to achieve an extension in range [2].

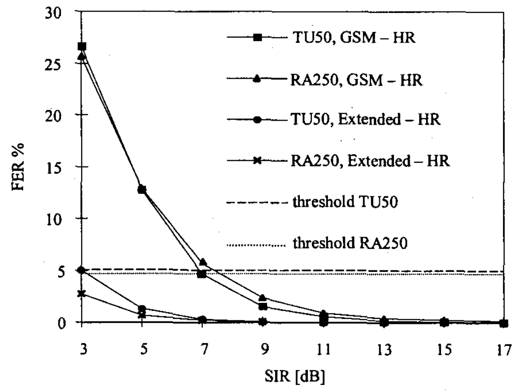

Figure 2 Frame erasure rate characteristic

The link adaptation mode selection process requires an accurate dynamic estimate of link quality. Performance assessment of link adaptation strategies has traditionally been based on SIR estimates to assess the channel quality. However, measurable channel metrics that can be used in the mode decision algorithm have been investigated in [4], and simulation results presented show that these metrics allow for fast and reliable mode decision with a very small computational cost.

This paper reports on further work which has led to the implementation of a real-time GSM link adaptation hardware demonstrator. The channel coding schemes proposed in [2] are utilised to implement Gross Rate link adaptation, and the mode selection process is performed using the metrics proposed in [4]. The system has been designed in a modular fashion to facilitate future integration of AMR codecs into the scheme, and can be adapted to assess performance of such codecs, in real time, using the same mode selection process tuned to specific codec characteristics. Further extensions to packet based systems are planned. The demonstrator is fully functional and it allows the user to assess the subjective speech quality enhancement achieved using LA, compared with the standard GSM speech channels under controlled dynamic variations in channel quality.

II. Hardware demonstrator structure

The emulator is composed of three main elements: a high speed PC and two Motorola DSP56307 Evaluation Module (EVM) boards. The requirements imposed by the real-time operation of the demonstrator has led to the use of a high spec PC (Pentium II 450 MHz, 512 Mbytes of RAM and an 18Gb hard disk). The DSP56307 was chosen because of its high performance (100 MIPS) and its suitability for wireless applications.

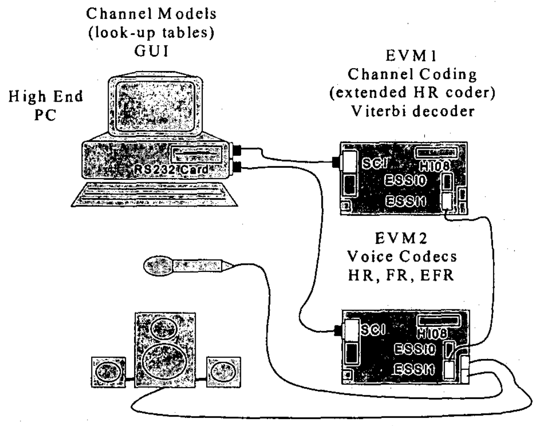

Figure 3 Structure of the real-time GSM hardware demonstrator

The structure of the emulator is depicted in figure 3. The emulation process is driven by variations in the CIR, which produce corresponding variations in the quality of the received bit stream. Error patterns are produced and transmitted to the first EVM which includes the channel coding and decoding, and carries out the Link Adaptation process. The second EVM implements the GSM speech codecs and the audio interface. Both DSPs work stand-alone.

The functionality, outlined in figure 4, and implementation trade-offs of the different elements of the emulator are detailed in the following paragraphs. The interfaces, used by the different elements of the emulator, to communicate are detailed in section IV.

Master EVM (EVM2)

The audio coding, speech coding and decoding functions are implemented in assembler in the Master EVM. Audio coding is performed by using a stereo audio codec with Delta-Sigma converters. The sampling frequency is 8 kHz and therefore 20 ms speech frames are produced. The timing introduced by the audio coding process is used to synchronise all the elements of the emulator hence the term *Master* EVM.

The three GSM standard speech codecs (Full Rate, Half Rate and Enhanced Full Rate) were provided by Motorola and were implemented in parallel in a single DSP. This parallel implementation of the speech codecs allows a seamless switch between the codecs without any loss in speech quality and demonstrates that the implementation of LA using actual technology is highly realistic.

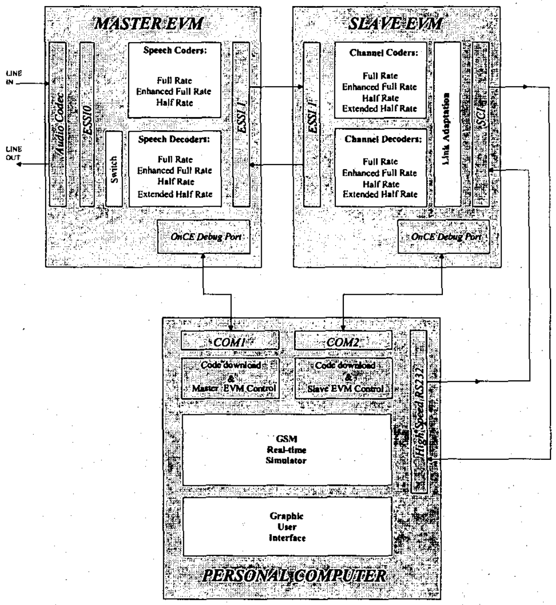

Figure 4 Functionality of the emulator components

The DSP56307 EVM includes external fast static RAM and flash memory for memory expansion. However, the assembler implementation of the Master EVM was highly optimised and the 64k words of internal memory was sufficient. The implementation used 14k Program RAM, 22k X data RAM, 20k Y Data RAM and 4k PROM, which led to the consumption of around 90% of the 100 MIPS available.

Slave EVM (EVM1)

The slave EVM includes the channel coding/decoding functions and the Link Adaptation mode selection algorithm. Four channel coding schemes are implemented in parallel: the three GSM standard modes (FR, HR and EFR) and the Extended Half Rate mode introduced for the purpose of Link Adaptation [2]. The Full Rate and Enhanced Full Rate traffic channels are implemented for subjective comparison. The Viterbi decoding algorithm is used for channel decoding.

The Link Adaptation mode selection algorithm implemented is based on the metrics described in [2]. It triggers changes in the selected LA mode on a frame by frame basis, according to link quality.

The implementation of the Slave EVM has been undertaken in C using the Tasking Embedded Development software and used around 40% of the internal DSP capabilities. The source code was written in ANSI C and thus was not specifically optimised for DSP implementation. Further code optimisation would significantly reduce the required DSP capabilities thus allowing the functions of the Master and Slave EVMs to be combined in a single DSP56307.

Personal Computer

The PC is responsible for downloading the code into the EVMs, the user interaction through a Graphic User Interface (GUI), and it includes the real-time GSM transmission chain simulator. The PC downloads the code into the EVMs through standard RS232 ports. The EVMs use an On-Chip Emulation (OnCE) module available in the EVMs for debugging and downloading purposes. The functionalities and different configuration capabilities provided by the GUI and emulator are presented in detail in section V.

The real-time capability of the emulator is made possible by producing a data base of error patterns produced by a GSM transmission chain. The error patterns are obtained from the output of the equaliser. The PC software selects the appropriate error patterns that will be sent through a high-speed RS232 port to the Slave EVM, according to the SIR experienced and the speed of the mobile. The SIR is monitored on a burst by burst basis so that the simulation process occurs in real-time. The link quality can therefore be modified at any point so that the PC acts as a fully functional simulator.

III. Information flow

The information flow within the emulator is illustrated in figure 5. The audio input is digitalised by the 16-bit stereo audio codec available in the EVMs. It produces 20 ms speech frames of 160 samples each.
The digital audio frame is then transformed into speech frames by the three GSM standard speech codecs. The outcome of the speech coding process is a 260-bit frame for the FR speech codec, a 244-bit frame for the EFR speech codec and a 112-bit frame for the HR speech codec. In order to add error protection capabilities to the speech frames, channel coding is then performed following the ETSI standard. The speech frames are then transformed into data blocks of 456 bits for the FR and EFR modes and 228 bits for the HR mode.

The Extended Half-Rate (EHR) mode is then introduced for implementing the LA algorithm. The channel coding scheme used transforms the 112-bit HR speech frame onto a block of 456 bits. The important error protection introduced by this code makes it a suitable candidate for environments with high interference conditions. The effect of the GSM environment is then introduced by the PC by means of error patterns. These error patterns are selected according to the speed of the mobile and the SIR experienced.

When simulating the GSM transmission chain the output of the channel coding is first interleaved and the received bit stream is de-interleaved before channel decoding. However, de-interleaving the error patterns and adding them to the output of the channel coder is equivalent. This means of implementation was adopted in the emulator to simplify timing issues.

Channel decoding is implemented using a Viterbi decoder. The error likelihood depends on the channel coding strategy. The recovered speech frames, that include the effect of the transmission channel, are then translated into digital audio data. An audio frame is produced for each mode and a switch drives the selected mode to the output of the audio codec.

IV. Communications Protocol

Due to the real-time nature of the emulator, a careful synchronisation and communication protocol between the different elements is required. Before describing the protocol, the different interfaces used between the EVMs and the PC are presented (see figure 4).

The PC is equipped with a high speed RS232 card in order to transmit the outcome of the GSM simulator to the Slave EVM. The modes selected by the user through the GUI and by the LA algorithm are also exchanged between the Slave EVM and PC using the serial link. The mode selected by the LA algorithm is sent to the PC so that it can be plotted in the GUI for illustrative purposes. The Slave EVM uses the Serial Communications Interface available in the EVM for this link. The transmission rate of the link is 307 kbps.
The Master and Slave EVMs communicate using one of the Enhanced Synchronous Serial Interfaces (ESSI) included in the EVMs. The ESSI works at a speed of 4.096 Mbit/second. This link is bi-directional and is used to exchange speech frames between the two EVMs. It is also used to signal to the Master EVM which mode has to be selected for the audio output. The ESSI implementation required considerable effort because of the complexity introduced by the tight timing and synchronisation requirements. The Master EVM uses the other available ESSI port for communication with the audio codec.

The Communications protocol defined in order to ensure that the speech frames are processed with the proper timing is detailed below and illustrated in figure 6. On power-up, the different interfaces in the two EVMs are initialised. Then, the PC starts the demonstration when indicated by the user to the GUI. A START message is sent to the Slave EVM through the serial link. The Slave EVM performs its set-up and signals the PC that it is ready using an acknowledgement message. The Slave EVM starts then processing the received speech frame from the Master EVM (in the initialisation case, this frame is a dummy one).

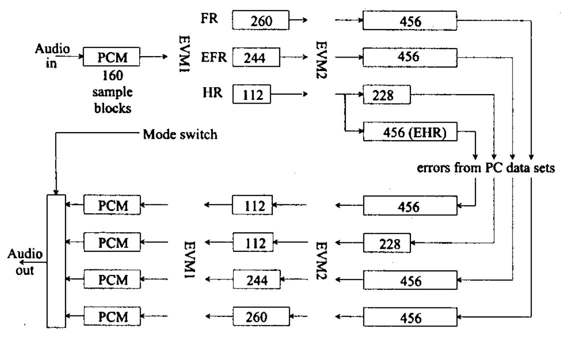

Figure 5 Information flow diagram

In parallel to this processing, the Slave EVM receives the error patterns from the PC. The PC starts this transmission when it received the acknowledgement from the Slave EVM. When the PC has finished transferring this data, it will wait to receive from the Slave EVM, the selected mode.

In the meantime, the Master EVM has also become operative. The first step of the Master EVM is to initialise the audio codec to start receiving speech frames. After initialising the audio codec, it processes the speech frame previously stored (in the initialisation phase, this frame is a dummy one). The Master EVM processing is done in parallel to the Slave EVM processing. When both EVMs have finished their processing, they will check the ESSI link to see if the other EVM has finished and is ready to exchange the processed frames. When both are ready, this exchange is accomplished.

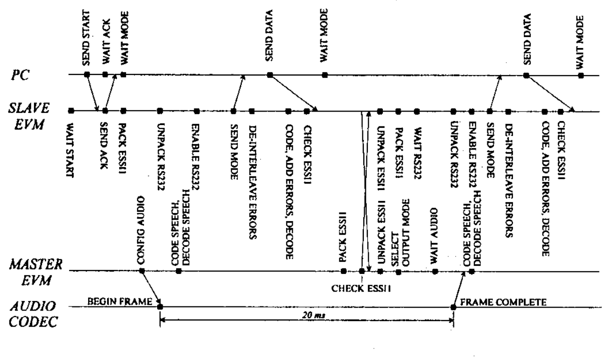

Figure 6 Communications protocol

After the frames have been exchanged through the ESSI, the Slave EVM resumes operation unpacking the data received from the PC and the Master EVM and signals to the PC the selected mode for the current speech frame. It then performs the channel coding, error insertion, channel decoding and the next speech frame mode selection. At the same time, the Master EVM unpacks the information received from the Slave EVM and waits for the audio codec to produce the next speech frame. When the next frame is available (each 20 ms), the Master EVM and audio codec exchange frames through the ESSI. The Master EVM then encodes the new audio frame and decodes the previous received frame from the Slave EVM. In the meantime, once the PC has received the selected mode from the Slave EVM, it starts sending the error patterns for the next frame. All the communications are synchronised by the timing introduced by the audio codec. It is therefore necessary for all processing and communication to be completed in less than 20 ms.

A very important feature of the DSP56307 that has been a key issue in the development of the emulator is the Direct Memory Access (DMA) controller. The DMA is an on-chip device that permits data transfer between memory and/or I/O in any combination without intervention of the program. The DMAs have been used by the EVMs so that they could exchange information through the different interfaces while processing the data, which allows the different elements of the emulator working in parallel. Without this feature the timing constraints introduced by the audio codec could not have been respected. All six DMAs available in the Master EVM were used for the communication with the Slave EVM and with the audio codec. The Slave EVM uses two DMAs for the communication with the Master EVM and one DMA to receive the error patterns from the PC while processing a frame at the same time.

V. Performance of the emulator

The demonstrator represents a real-time speech-to-speech dynamic GSM transmission chain emulator. The demonstration is driven by variations in the Signal to Interference Ratio as the user moves within a closed circuit within an urban environment.

Figure 7 represents the GUI of the demonstrator. It is composed of three plots and a status bar. The first plot displays the variation of SIR in dB as a function of time. The second plot displays the Bit Error Rate at the input of the channel decoder. The third one displays the mode selected by the Link Adaptation algorithm. The status bar includes the system configuration (speed, Base Station location, codec) and the current SIR value.

The demonstrator includes a set of examples of one minute each. Three Base Station locations have been considered within the urban circuit. Also, speeds for pedestrians (1 km/h and 5 km/h) and car terminals (10 km/h and 50 km/h) are included. The SIR can also be changed manually during simulation time. The user can also select the speech traffic channel to be used. The three GSM standard channels are available for comparison. When LA is activated the adaptation process selects the appropriate codec as shown in figure 1. An example of the configuration of the emulator through the GUI is shown in figure 8.

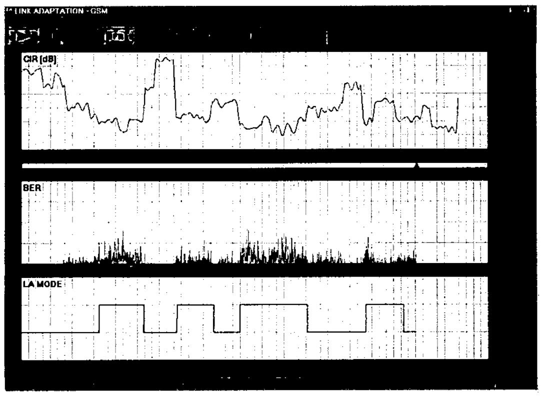

Figure 7 Main window of the LA demonstrator

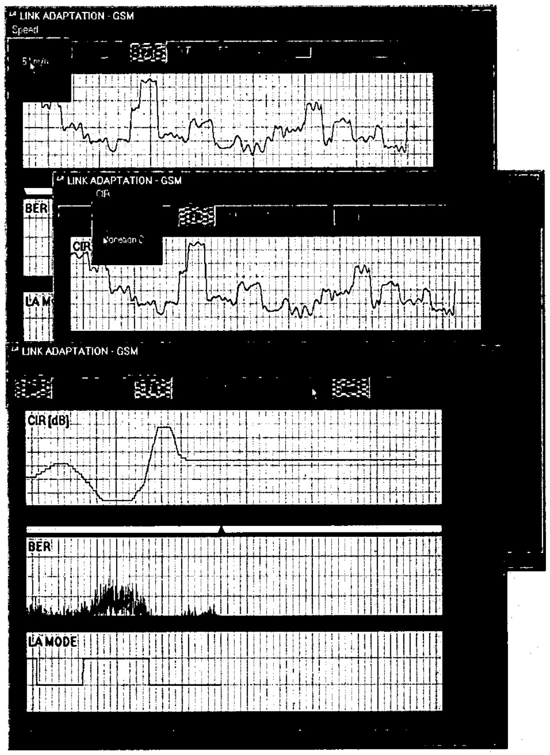

Figure 8 Configuration of the GUI

Figure 9 illustrates the use of LA to alleviate a "black spot situation" experienced between the 10 and 20 seconds markers. The statistics for that interval reveal that the FER is reduced from 3.3% to 0.4% thus producing a significant increase in perceived quality.

VI CONCLUSIONS

This paper has detailed the implementation of a real-time GSM Link Adaptation hardware demonstrator. The demonstrator is based on previous work on Gross Rate Link Adaptation. The demonstrator may be readily configured to evaluate the performance of AMR codecs under controlled dynamic conditions. Further work is being undertaken to evaluate the real time performance of link adaptation in GPRS and EDGE [3].

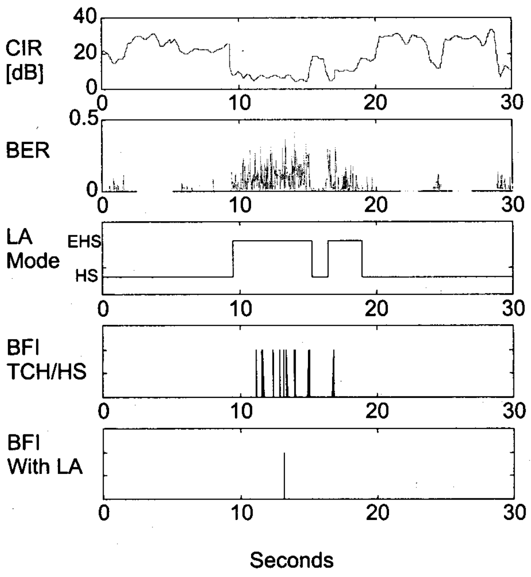

Figure 9 Performance example (50 km/h, BS location B)

VII ACKNOWLEDGEMENT

The authors acknowledge the support of Scottish Enterprise in this work and the valuable contribution of Mr Ian McCoull, of Scottish Enterprise, in terms of project management.

REFERENCES

[1] Dunlop, Irvine and Cosimini, "Implementation considerations for Gross Rate Link Adaptation" IEEE Vehicular Technology Conference, VTC-96, Atlanta 1996.

[2] Dunlop J and Pons J "Range Extension Based on Link Adaptation" Proceedings of the International Workshop on Wireless Communications, Chania, Greece, June 1999, pp 58-65

[3] Frodigh, Furuskar, Olofsson, Skold, "System performance of EDGE, a proposal for Enhanced Data Rates in Existing Digital Cellular Systems" Proc. IEEE VTC'98, Ottawa, pp. 1284-89.

[4] Pons, Dunlop, "Bit error based based link adaptation for GSM" Proc. PIMRC'98, Boston.