GSM HALF-RATE SPEECH CODEC: BIT EXACTNESS VERSUS DSP ARCHITECTURE

Tim Fingscheidt, Thomas Wiechers and Eckhard Delfs

e-mail: tim@ind.rwth-aachen.de
Institute of Communication Systems and Data Processing (IND)
Aachen University of Technology
52056 Aachen, Germany

ABSTRACT

In this paper the GSM Half-Rate Speech Codec is examined with respect to implementation complexity taking the bit exactness requirements as well as differences in DSP-architectures into consideration. While the codec specification [4] assumes some "ideal DSP" with a certain architecture, it turns out that the implementation on a real-world DSP could require about 50 MIPS. The computational load can be reduced by minor modifications of the DSP instruction set (appropriate saturation logic and barrel shifter) down to 25 MIPS. However, by exploiting detailed knowledge of the codec algorithm, a processor load of less than 25 MIPS can be achieved too, even if the DSP does not provide the required saturation logic. This is shown by way of example, using a single NECμPD77018 DSP for a full duplex real time implementation. The potential for complexity reduction is discussed.

1. INTRODUCTION

The *European Telecommunications Standards Institute* (ETSI) has specified the GSM Half-Rate Codec with a bit rate of 5.6 kbit/s in January 1995 [1]. The specification requires bit exactness which imposes strict constraints on an implementation using any fixed point DSP. To investigate the matching of the GSM Half-Rate Codec algorithm to a given DSP, the most frequently used "basic mathematical operations" in the algorithm are examined. Their complexities on different DSPs strongly depend on the architecture (saturation logic e.g.), but also vary with the range of the numbers they are working on. Section 3 deals with architectural aspects and bit exactness requirements regarding the very frequently used shift operations as well as the multiply-accumulate (MAC) instruction. In Section 4 an efficient input value dependent solution for shift routines is specified adapted to the GSM Half-Rate Codec. Furthermore, low complexity implementation possibilities of the MAC instruction in some time critical parts of the algorithm are stated for DSPs not providing the saturation requirements as specified by ETSI. To resume the methods of efficiency enhancement, a complexity estimate is done assuming different versions of a fictional DSP. Finally, in Section 5, a bit exact implementation using the NECμPD77018 DSP will be discussed as an example. Although this DSP differs from the "ideal" one, an efficient implementation can be obtained by a consequent realization of the previously discussed complexity reduction aspects.

This work has been supported by NEC Electronics Europe GmbH in Düsseldorf, Germany.

2. OVERVIEW TO THE GSM HALF-RATE ALGORITHM

In the following, a brief overview to the GSM Half-Rate Codec is given regarding Fig. 1. A complete algorithmic description can be found in [1], [7].

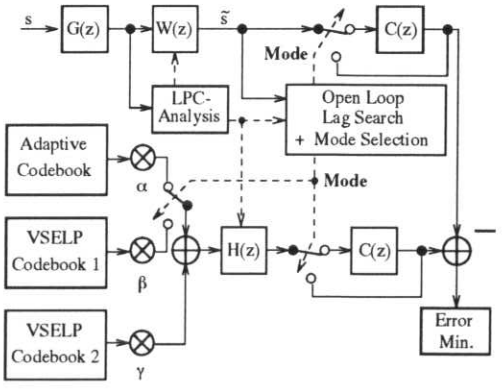


Figure 1: GSM Half-Rate Encoder Structure:
G(z) : High Pass, W(z) : Spectral Weighting Filter,
C(z) : Harmonic Weighting Filter,
H(z) = A(z) · W(z) : Cascade of the LPC Synthesis Filter A(z) and the Spectral Weighting Filter W(z)

The GSM Half-Rate Codec consists of an analysis-by-synthesis structure working in different modes. 10 LPC coefficients are computed from an analysis frame length of 170 highpass filtered samples and updated every 20 ms. The 170 analysis samples are the most recent ones in a ringbuffer containing 195 input speech samples. This is due to an algorithmic delay of 24.4 ms.

The analysis procedure [6] is an autocorrelation fixed point lattice technique (AFLAT) working on autocorrelation values evaluated by a covariance lattice algorithm (FLAT). After an optional interpolation of the filter coefficients and spectral weighting of the input signal by $W(z)$ an open loop search for "candidate" long term predictor lags is performed. For an efficient delta coding of the lags, on optimum frame lag trajectory is computed. A voiced/unvoiced decision is taken by determination of a 2 bit mode parameter: If the speech is decided to be "totally unvoiced", two VSELP codebook searches [7] each consisting of 7 basis vectors are performed over a subframe length of 5 ms minimizing the total squared error (speech weighting is done by $W(z)$). Otherwise, a closed loop fractional lag search is carried out based on the optimum frame lag trajectory found in the open loop search. Afterwards, one VSELP codebook search is done working with 9 basis vectors. In this case, the speech is additionally weighted by a harmonic noise weighting filter with transfer function $C(z)$.

For a real time implementation, the time critical parts of the codec algorithm are the AFLAT recursion, the open loop search for candidate lags as well as the subsequent codebook searches. They will be referenced in Section 4.

3. BASIC MATHEMATICAL OPERATIONS

The GSM Half-Rate Speech Codec algorithm is specified exclusively by performing "basic mathematical operations", in the following called "macros" [4]. They are called e.g. add() for the addition of two 16 bit words or L_mac()[1] for the multiplication of two 16 bit words and a final addition to a 32 bit number. Only in case of the following exceptions computations are not carried out by these basic operations:

- Address evaluation is expected to be done in parallel to other instructions (performed by the address generation unit of the DSP). So the usual address pointer increments are done directly without usage of an add() or sub() macro.
- Furthermore, there is no basic operation macro to perform a loop. This is due to the hardware loop capabilities of most DSPs that do not need an explicit software counter.

[1] The prefix L_ denotes the result of the basic operation being a 32 bit word.

All the other computations in the half rate algorithm make use of the macros specified bit precisely.

The structure of these macros is strongly correlated to the number of processor instructions a given DSP needs to perform a bit exact operation.

3.1 The Multiply-Accumulate Operation

One of the most important instructions in DSP applications is the MAC (multiply-accumulate) instruction represented by the L_mac() macro. On modern DSPs it requires a single instruction cycle to perform

$$Z = Y + x_1 \cdot x_2 \tag{1}$$

with small letters indicating 16 bit numbers in the range $-1 \leq x_1, x_2 \leq 1 - 2^{-15}$ and capital letters denoting 32 bit numbers in the range $-1 \leq Y, Z \leq 1 - 2^{-31}$. All numbers are assumed to be represented by the 2's complement. In general, the multiplication of two 16 bit values results in a 32 bit number (done by the L_mult() macro) except the case, that $x_1 = x_2 = -1$, because the result of $+1$ is principally not representable in the 32 bit number format. Accordingly, to ensure a valid 32 bit result, a saturation of the multplicator output has to be performed.

For comparison: The GSM Full-Rate Codec is designed in a way that this exception surely does not occur so saturation is not necessary after a multiplication with 32 bit output [5].

The L_mac() operation of the half rate codec is defined as L_mult() followed by L_add(). The addition of two 32 bit values (Y and $X = \text{saturate}(x_1 \cdot x_2)$) should again result in a 32 bit value Z. Sometimes, this is ensured by previous downshifting of X and Y, but in general this can only be achieved by saturation again.

It can be summarized that the MAC operation with input values covering their full range requires four instructions: multiplication, saturation, addition, saturation. So this is the specification for a MAC operation of DSPs ideally matched to the GSM Half-Rate Codec bit exactness requirement.

DSPs providing a saturation logic, i.e. an optional saturation after a certain set of instructions without requiring additional instruction cycles, have large advantages in terms of complexity:

- They need only one instruction if the MAC operation performs even the first saturation (the second one is performed by the saturation logic),
- they need two instructions (MULT and ADD) if the multiplication result during their MAC operation is not saturated. In general, these DSPs cannot use their MAC instruction for a half rate codec implementation.

DSPs lacking a saturation logic are mismatched to the basic operation definitions made for the half rate codec. They have to perform four instructions

instead of a single cycle L_mac() operation. To match the required 16 bit or 32 bit result number format of any basic mathematical operation, these DSPs have to provide an additional saturation instruction after a large set of operations like add(), sub(), mult(), neg(), abs(), etc.

Although the designers of the GSM Full-Rate Codec did not expect a DSP having a MAC operation, even DSPs without saturation logic but providing a MAC instruction may use it whenever an L_mult() is followed by an L_add(). This is due to the fact that in the full rate codec the L_mult() operation never results in an overflow.

Table 1 summarizes the DSP instructions to be performed for implementing the L_mac() operation in the half rate codec or the cascade of L_mult() and L_add() in the full rate codec, respectively. "SAT" denotes the processor instruction for saturation.

DSP Specification	GSM FR	GSM HR
Ideal DSP providing both saturations	MAC	MAC
DSP providing a saturation logic	MAC	MULT ADD
DSP providing no saturation logic	MAC, SAT	MULT, SAT ADD, SAT

Table 1: Instructions to perform a bit exact implementation of the L_mac() macro on different DSP architectures

An interesting result of the problems mentioned above is that "non-ideal" DSPs lacking the saturation after the multiplication of their MAC instruction which could use their extension bits for higher mathematical accuracy, are prohibited to do so for the requirements of bit exactness. This effect is also known from implementations of other bit exactly specified algorithms [8]. The mathematical macros as defined by ETSI are optimally matched to a DSP without any extension bits in its ALU registers but with automatical saturation of each overflow.

Possibilities to use the MAC instruction even in "non-ideal" DSPs are discussed in Section 4.

3.2 Shifting Operations

There are several basic mathematical operations performing shifts with optional rounding. They are defined in a very general manner: The L_shl() macro for example carries out a left shift with overflow control, i.e. a saturation if the sign bit would be at least one time inverted during the shifting process. If the "number of bits to shift left"- argument of L_shl() is negative, then an arithmetic right shift has to be performed. A general implementation of this shift routine on a modern DSP can require 5 or more instruction cycles. Nevertheless, in Section 4.2 it is shown that in practice the shifting macros can be reduced to one or two DSP instructions.

4. ALGORITHMIC COMPLEXITY REDUCTION ASPECTS

4.1 The Multiply-Accumulate Operation

In comparison to other macros, L_mac() and L_msu() (multiply-subtract) contribute most to the codec complexity. So it is of great interest for all implementations on "non-ideal" DSPs whether both saturations in these macros are necessary or not. In the following, parts of the algorithm are discussed where large complexity savings are possible. All numbers denoting savings of computational load are based on the assumption, that every instruction in the right column of Table 1 requires one instruction cycle.

As an example, in the AFLAT recursion the residual error of each vector from the reflection coefficient quantizers is computed. The products within MAC operations in the recursion loop are multiplications by $\hat{r}$ or $\hat{r}^2$ with $\hat{r}$ denoting a reflection coefficient from the (pre-) quantizer. Because the entries in the quantizer tables show that $\hat{r} \neq -1$, a saturation after such multiplications never is necessary and accordingly all DSPs may use their MAC operation. This leads to savings of about 0.7 MIPS on DSPs providing a saturation logic (one instruction instead of two) and 1.4 MIPS on DSPs lacking a saturation logic (max. two instructions instead of four).

Another interesting part of the algorithm is a single MAC operation in the open loop LTP lag search procedure. Here the autocorrelation sequence $C(k,m)$ is computed for every subframe m of length $N_s = 40$.

$$C(k,m) = \sum_{n=0}^{N_s-1} \tilde{s}(n+(m-1)N_s) \cdot \tilde{s}(n+(m-1)N_s-k) \tag{2}$$

with $18 \leq k \leq 144$. The weighted input speech $\tilde{s}(n)$ is scaled such that $C(k,m) < 1$, so no saturation is necessary anyway. The savings are about 1 MIPS and 3 MIPS (both saturations are unnecessary), respectively.

The codebook search algorithm [1] works in an efficient way evaluating $C_i^2 G_{best} > C_{best}^2 G_i$ for each codevector. Because the codevectors $\mathbf{u}_i$ consist of a (signed) sum of M basis vectors $\mathbf{v}_m$, the above values are given as

$$C_i = \frac{1}{2} \sum_{m=1}^{M} \theta_{im} R_m, \text{ with } R_m = 2\mathbf{p}^T \mathbf{q}'_m \tag{3}$$

with $\mathbf{p}^T$ being the transposed target signal vector and $\mathbf{q}'_m$ denoting the decorrelated zero state response of the synthesis filter cascade to $\mathbf{v}_m$. $\theta_{im} \in \{-1, +1\}$ denotes the sign of basis vector $\mathbf{v}_m$ in codevector $\mathbf{u}_i$.

The energy term is given by

$$G_i = \frac{1}{2}\sum_{j=2}^{M}\sum_{m=1}^{j-1}\theta_{im}\theta_{ij}D_{mj} + \frac{1}{4}\sum_{j=1}^{M}D_{jj}, \quad (4)$$

$$D_{mj} = 4\,\mathbf{q}_m'^T\mathbf{q}_j' \,. \quad (5)$$

The codebook search is done such that successive codevectors i, u differ only in one bit position μ. The corresponding crosscorrelation and energy terms can be computed using

$$C_u = C_i + \theta_{u\mu}R_\mu \quad (6)$$

$$G_u = G_i + \sum_{j=1}^{\mu-1}\theta_{uj}\theta_{u\mu}D_{j\mu} + \sum_{j=\mu+1}^{M}\theta_{uj}\theta_{u\mu}D_{\mu j} \quad (7)$$

The M MAC operations required for an update of equations (6),(7) can be done without any saturation because the R_m and D_{mj} values in (3) and (5) are scaled to prevent overflow during the codebook search. This scaling is due to a complexity saving of more than 0.5 MIPS (1.5 MIPS).

There are a lot of filter routines contributing a large amount of codec complexity by their L_mac() and L_msu() operations. It can be stated that a saturation after multiplication never is required, so all DSPs can use their MAC operation. Saturation after addition is still necessary.

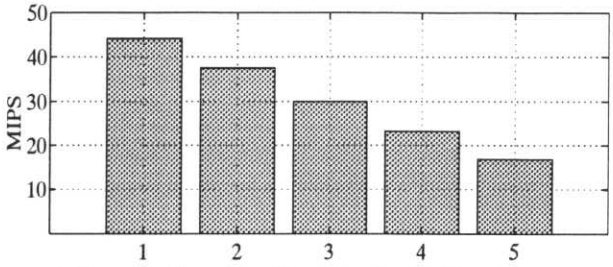
4.2 Shifting Operations

If the DSP has a barrel shifter, the shifting operation (without rounding) mostly will not require more than one or two processor instructions. This is due to the fact that on the one hand in many cases the shifts are right shifts by a positive constant or left shifts by a negative constant number of bits. Both cases are in fact an arithmetic right shift that normally can be executed during one instruction cycle. On the other hand, most of the shift operations by a variable, possibly negative number of bits can be simplified extremely: The FLAT algorithm for example works on the windowed covariance coefficients $\Phi'(i,k)$ evaluated from the highpass filtered input speech. Three matrices $\mathbf{F}, \mathbf{B}, \mathbf{C}$ (initially filled with $\Phi'(i,k)$) of dimension $N_P \times N_P$ with $N_P = 10$ being the LPC filter order are recursively updated. During the whole algorithm accessed matrix elements are left shifted by $l = m - c$ bits with $m \geq 0$ denoting the number of left shifts to normalize the maximum element of $\mathbf{F}, \mathbf{B}, \mathbf{C}$. This is done to guarantee a maximum of mathematical accuracy. The constant $c = 2$ preserves the following operations from overflow. It is evident, that l may be positive or negative depending on the input data $\Phi'(i,k)$. Because a left shift by m bits never results in an overflow (normalization!), the shift by l bits can be split into a left shift by m bits (saturation not necessary) followed by a right shift by c bits.

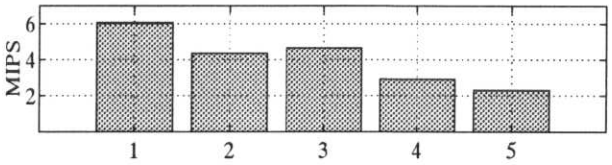
In spite of the very generally defined shift operations, it can be stated that a DSP providing a barrel shifter will need only one or two instructions to carry them out (except optional rounding).

4.3 Estimation of Codec Complexity

It is of interest to estimate the computational savings that can be obtained by the previous mentioned simplifications. To do so, a speech codec complexity estimation is done on the basis of the mathematical operations as defined by ETSI. In Fig. 2 worst case encoder and decoder complexities are depicted belonging to different versions (1 ... 5) of a fictional DSP. All versions provide a barrel shifter, a MAC instruction performed in one instruction cycle with no saturation possibility after multiplication and shifting instructions only working on a positive number of bit shifts.



a) Speech Encoder Complexity Estimate



b) Speech Decoder Complexity Estimate

Figure 2: The depicted bars show the following cases:

1. Fictional DSP without saturation logic: All basic mathematical operations are implemented in their general function as specified by ETSI.
2. DSP as 1), but the shifting functions are adapted to their arguments (see Section 4.2): They are estimated to a medium complexity of two cycles per shift.
3. DSP as 1), but the DSP provides a saturation logic (still no saturation after the multiplication of the MAC instruction!).
4. Both specifications of 2) and 3) together.
5. DSP as 4); additionally it is assumed that the saturation after the multiplication of a MAC instruction can be omitted (see the examples in Section 4.1).

Although the absolute values in Fig. 2 may be very sensitive to a certain DSP architecture and instruction set, the complexity differences between the de-

picted cases give a realistic impression of the computational savings gained by a saturation logic and some algorithmic know how. An implementation based on the mathematical operations in their general definition would require an amount of 50 MIPS. Relative to this value, a saturation logic saves about 15.7 MIPS. Introducing the simplifications discussed in this Section, a further reduction by 8.4 MIPS (shift routines) plus 7.0 MIPS (MAC instructions) is possible resulting in a complexity of about 20 MIPS.

It can be summarized that even a "non-ideal" DSP of the above assumed type is capable of performing the GSM Half-Rate Codec with about 20 MIPS if a saturation logic exists and the complexity reduction possibilities of shift and MAC routines are exploited.

5. A REAL TIME IMPLEMENTATION ON A SINGLE NECμPD77018 DSP

To investigate the matching properties of the GSM Half-Rate algorithm and a given DSP, we implemented the speech codec including muting techniques, the voice activity detection (VAD) [3], comfort noise generation, and discontinuous transmission (DTX) [2] on a DSP NECμPD77018. This is a modern 16 bit fixed point DSP having a 30 ns cycle time (33.3 MIPS) equipped with a barrel shifter, 8 general purpose 40 bit registers and a comfortable 32 bit instruction set, but without a saturation logic. It provides on chip data memory of 2 $\times$ 3 K RAM and 2 $\times$ 12 K ROM and an amount of 24 K ROM instruction words on chip.

The instruction set of this processor is partly significantly different from the "typical" macro instruction set as specified by ETSI. Although the lack of a saturation logic was shown to lead to an overhead of 15.7 MIPS, Tab. 2 demonstrates that a real time implementation using less than 25 MIPS is possible even without a saturation logic. This is due to the large number of general purpose registers and the comfortable 32 bit instruction set. The number of shifts to normalize e.g. can be evaluated very efficiently during one instruction cycle only. Additionally, the required RAM for scratch and static data does not exceed 3.7 K. This is much less than expected from [4] and indicates, that not all variables declared as static really need an exclusive memory space. Finally, there is enough space for performing the channel codec and other functions on the same chip also in real time.

		Speech Codec		
		ENC	DEC	Codec
Worst Case MIPS		21.25	3.33	24.6
Load on NECμPD77018		64.0 %	10.0 %	74.0 %
Data Memory	static RAM			2.31 K
	scratch RAM			1.36 K
	ROM			8.0 K
Program Memory				7.7 K

Table 2: Memory usage and processor load of the GSM Half-Rate Codec on the NECμPD77018 DSP

6. CONCLUSION

The GSM Half-Rate Codec algorithm requires a DSP providing a saturation after each operation. A detailed analysis of the speech codec algorithm leads to a complexity estimate of about 20 MIPS using such a DSP. However, the implementation on a real-world DSP e.g. without saturation logic could require about 50 MIPS. A bit exact implementation on a single NECμPD77018 processor is presented. It turns out that even DSPs lacking a saturation logic are able to perform the GSM Half-Rate Codec in real time requiring less than 25 MIPS, if detailed knowledge of the algorithm is exploited.

7. REFERENCES

[1] "European Digital Cellular Telecommunications System Half Rate Speech Part 2: Half Rate Speech Transcoding", *Recommendation GSM 06.20, ETSI/TC SMG*, January 1995.

[2] "European Digital Cellular Telecommunications System Half Rate Speech Part 5: Discontinuous Transmission (DTX) for Half Rate Speech Traffic Channels", *Recommendation GSM 06.41, ETSI/TC SMG*, January 1995.

[3] "European Digital Cellular Telecommunications System Half Rate Speech Part 6: Voice Activity Detector (VAD) for Half Rate Speech Traffic Channels", *Recommendation GSM 06.42, ETSI/TC SMG*, January 1995.

[4] "European Digital Cellular Telecommunications System Half Rate Speech Part 7: ANSI-C Code for the GSM Half Rate Speech Codec", *Recommendation GSM 06.06, ETSI/TC SMG*, April 1995.

[5] "GSM Full Rate Speech Transcoding", *Recommendation GSM 06.10, ETSI/TC SMG*, February 1992.

[6] A. Cumani, "On a Covariance–Lattice Algorithm for Linear Prediction", *ICASSP*, May 1982, pp.661–665.

[7] I.A. Gerson and M.A. Jasiuk, "A 5600 BPS VSELP Speech Coder Candidate for Half-Rate GSM", *EUROSPEECH*, Berlin, 1993, pp.253–256.

[8] M.T. Murphy and C.E.M. Cox, "A Real Time Implementation of the ITU G.728 LD-CELP Fixed Point Algorithm on the Motorola DSP56156", *Signal Processing VII, Proc. of EUSIPCO*, September 1994, pp.1617–1620.