In car audio

S. Cecchi, L. Palestini, P. Peretti, A. Primavera, F. Piazza, F. Capman, S. Thabuteau, C. Levy, J.-F. Bonastre, A. Lattanzi, E. Ciavattini, F. Bettarelli, R. Toppi, E. Capucci, F. Ferrandi, M. Lattuada, C. Pilato, D. Sciuto, W. Luk, J.G. De Figueiredo Coutinho

S. Cecchi, L. Palestini, P. Peretti, A. Primavera, F. Piazza, F. Capman, S. Thabuteau, C. Levy, J.-F. Bonastre, A. Lattanzi, E. Ciavattini, F. Bettarelli, R. Toppi, E. Capucci, F. Ferrandi, M. Lattuada, C. Pilato, D. Sciuto, W. Luk, and J.G. De Figueiredo Coutinho. *Hardware/Software Co-design for Heterogeneous Multi-core Platforms: The hArtes Toolchain*, chapter In car audio, pages 155–192. Springer Netherlands, 2012

5
In Car Audio

Stefania Cecchi, Lorenzo Palestini, Paolo Peretti, Andrea Primavera, Francesco Piazza,
DIBET - Universitá Politecnica delle Marche,
Francois Capman, Simon Thabuteau,
Thales Communications
Christophe Levy, Jean-Francois Bonastre,
Université d'Avignon et des Pays de Vaucluse
Ariano Lattanzi, Emanuele Ciavattini, Ferruccio Bettarelli,
Leaff Engineering,
Romolo Toppi , Emiliano Capucci,
Faital
Fabrizio Ferrandi, Marco Lattuada, Christian Pilato, Donatella Sciuto,
Politecnico di Milano,
Wayne Luk, Jose Gabriel de Figueiredo Coutinho,
Imperial College,

This chapter presents implementations of advanced in Car Audio Applications. The system is composed by three main different applications regarding the In Car listening and communication experience. Starting from a high level description of the algorithms, several implementations on different levels of hardware abstraction are presented, along with empirical results on both the design process undergone and the performance results achieved.

5.1 Introduction

In the last decade Car infotainment (i.e. the combination of information with entertainment features) systems have attracted many efforts by industrial research because car market is sensible to the introduction of innovative services for drivers and passengers. The need for an Advanced Car Infotainment System (ACIS) has been recently emerging, able to handle issues left open by CIS systems already on the market, and to overcome their limitations. Due to traffic congestion and growing distance from home to workplace, people spend more and more time in car, that hence becomes an appealing place to do many common activities such as listening to music and news, phone calling and doing many typical office tasks. Narrowing our focus to audio, the key role of the CIS is that it lets the driver concentrate on the road and at the same time it manages many different processing functions such as high quality music playback, hands-free communication, voice commands, speaker recognition, etc. Moreover, from the signal processing point of view, the wide band nature of the audio signal adds complexity while the requested quality calls for high precision signal processing, making in-car audio processing a very open research and development field. Therefore new architectures are needed to overcome the nowadays limitations. In this context, one of the main objective of the European hArtes Project (Holistic Approach to Reconfigurable real Time Embedded Systems) is to develop an ACIS, capable to meet market requirements. It is achieved with a multichannel input (microphone array) and output (speaker array) platform managed by NU-Tech framework [2] and implemented on a real car named hArtes CarLab. In particular such system has been adopted in the hArtes project [1, 2] as a proof-of-concept in order to test and assess the project methodologies. In fact the main goal of the hArtes project is to provide, for the first time, a tool chain capable of optimal, automatic and rapid design of embedded systems from high-level descriptions, targeting a combination of embedded processors, digital signal processing and reconfigurable hardware. Three main different applications have been developed for the ACIS within the hArtes project:

- Enhanced In-Car Listening Experience: It is necessary to develop audio algorithms to improve the perceived audio quality in cars, making the listening environment more pleasant, taking into account specific features of the car cabin.

- Advanced In-Car Communication: The advanced in-car communication scenarios are based on speech and audio signal processing in order to enhance hands-free telephony, in-car cabin communication and automatic speech recognition for an efficient user interface.

- In-Car Speaker and Speech Recognition: The automatic speech and speaker recognition modules aim to provide an improved man-machine interface (MMI) for user authentication and command and control of software applications.

Section 5.2 introduces the state of the art of the three main applications. Section 5.3 is focused on the description of the audio algorithms implemented to improve the audio reproduction and to manage the communication and interaction features. Section 5.4 describes the hArtes toolchain and how has been applied for the implementation of the selected algorithms. Section 5.5 reports the experimental results for the PC-based prototype and for the final embedded platform prototype.

5.2 State of the art for In Car Audio

In the following, the state of the art for each of the three main field of application will be reported.

5.2.1 Enhanced In-Car Listening Experience

Despite of the advent of consumer DSP applied to sound reproduction enhancement, few applications have been developed for the in-car listening experience especially due to characteristics of automobile environment which is not an ideal listening environment. The automobile is a well-known small noisy environment with several negative influences on the spectral, spatial and temporal attributes of the reproduced sound field [35]. Specially, depending on the absorbing or reflecting interior materials, the position of loudspeakers and the shape of the car cabin, the reflected sounds attenuate or amplify the direct sound from the loudspeakers [36]. One of the most comprehensive work found in literature, regarding a complete automotive audio system is presented in [37]. The embedded system comprises different processing units: a parametric equalizer is used to correct irregularities in the frequency response due to car cabin characteristics; an adjustable delay is needed in order to equalize arrival times due to loudspeakers position; a dynamic range control is considered for compressing and amplifying the reproduced material taking into account the measured background noise level; a surround processor to artificially recreate a more appealing listening environment. Some of the previous aspects have been extensively studied singularly in other works. The equalization task has been investigated thoroughly: in [38] some fixed equalization algorithms based on different inversion approaches is presented together with a surround processor to remove and add unwanted/wanted reverberation components. With the advent of multichannel audio content, different schemes have been considered to create a compelling surround experience. In [39] several audio technologies that support reproduction of high quality multichannel audio in the automotive environment are illustrated. Although these technologies allow a superior listening experience, different open problems remain such as off-axis listening position of the passengers. In [40] a valid solution to improve the surround imaging is presented: it is based on comb filtering designed taking into account inter-loudspeaker differential phase between two listening positions. A digital audio system for real time applications is here proposed. Its aims are substantially two: to develop a complete set of audio algorithms improving the perceived audio quality in order to make the listening experience more pleasant taking into account some specific features of the car cabin; to have a modular and reconfigurable system that allows to seamlessly add, remove and manage functionalities considering, for the first time, a PC based application.

5.2.2 Advanced In-Car Communication

Monophonic Echo Cancellation

Following the formalization of the adaptive complex LMS (Least Mean Square) in [43], the use of fast convolution methods for the derivation of FDAF (Frequency Domain Adaptive Filtering) have been proposed in order to reduce the overall complexity of the adaptive filter. These methods are either based on OverLap-and-Add (OLA) or OverLap-and-Save (OLS) method. The OLS method is more generally used since it has a direct and intuitive interpretation. A first implementation of an adaptive filter in the frequency-domain was proposed in [45]. An exact implementation of the LMS (Fast LMS) was proposed in [46] with the calculation of a constrained gradient. A sub-optimal version with the calculation of the unconstrained version was derived in [49] for further reducing the complexity, and an approximation of the constrained gradient was proposed in [51] using a time-domain cosine window. The application of FDAF to acoustic echo cancellation of speech signals have to solve additional problems due to the speech signal properties (non-stationary coloured signal) and due to the acoustical path properties (non-stationary acoustical channel with long impulse responses). The frequency-domain implementation of block gradient algorithm exhibits better performances than its time-domain counterpart since it is possible to perform independent step-size parameter normalisation for each frequency bin, acting as a pre-whitening process. The major advance for its application to acoustic echo cancellation was to partition the adaptive filter in order to identify long impulse responses as implemented in the MDF (Multi-Delay Filter) algorithm in [53]. In this approach it is possible to choose an arbitrary FFT size for filtering whatever the size of the impulse response to be identified. The processing delay is by consequent also significantly reduced for transmission applications. Further improvements in terms of performances have been achieved in the GMDF (Generalised Multi Delay Filter), [55], [50], using overlapped input data leading to an increased updating rate. The residual output signal is regenerated using a WOLA (Weighted OverLap-and-Add) process. More recently, an improved version of the MDF filter has been described in [59], the Extended MDF (EMDF) which takes into account the correlation between each input blocks resulting from the filter partitioning. One of the advantages of the frequency-domain implementation is also to implement globally optimised solutions of the adaptive filter for echo cancellation and speech enhancement as in [58], [44], [47] and [48]. The overall complexity of the combined AEC with the speech enhancement algorithm can further be reduced (common FFT) and enhanced performances can be achieved using noise-reduced error signal for adaptation.

Stereophonic Echo Cancellation

When generalizing the acoustic echo cancellation to the multi-channel case, one has to deal with the cross-channel correlation. This problem has been described in [52]. Most of the studies in the field of acoustic echo cancellation carried out in the nineties were devoted to the improvements of sub-optimal multi-channel algorithms, trying to achieve a good compromise between complexity and performances, [54], [56]. More recently, a more optimal derivation of the MDF filter was proposed in [60], where the EMDF filter is generalised to the multi-channel case leading to a quasi-optimal frequency-domain implementation taking into consideration both the correlation between sub-blocks resulting from the partitioning process and the cross-channel correlations. This solution is targeting the identification of very long impulse responses with high-quality multi-channel audio reproduction. However the gain in performances has not been proven to be significant for the limited stereophonic case in a car environment for which impulse responses are shorter than in rooms. Results showing the combination of a sub-optimal FDAF with an ASR system in a car environment have been given in [57].

In-Car Cabin Communication

Here the main objective is to improve the communication between passengers inside a car or vehicle. The difficulty of in-car cabin communication result from various factors: noisy environmental conditions, location and orientation of passengers, lack of visual feedback between passengers This is particularly true with the emergence of van vehicles on the market. A critical scenario is for passengers located at the rear of the car while listening to the driver in noisy conditions. This topic has been recently addressed in the literature, and is also part of working groups at standardisation bodies. The basic principle is to pick-up passengers voice with one or several microphones and to reinforce the speech level through loudspeakers. The main challenges are: to avoid instability of the system due to the acoustical coupling, to avoid reinforcing the noisy environment, and to keep the processing delay below an annoyance level. In [65], a characterisation of the transfer functions depending on the microphone position inside a vehicle is performed, and a basic system with only two microphones and two loudspeakers is described, using two acoustic echo cancellers. The authors also proposed to include a feedback cancellation based on linear prediction. In [64,66,67], the proposed system is composed of an acoustic echo canceller to remove the echo signal on the microphone which pick-up the passenger s voice, follower by an echo suppression filter for removing the residual echo and a noise suppressor in order to avoid reinforcing the environmental noise. A speech reinforcement of up to 20 dB is claimed. Further improvements of the proposed system are given in [68–71]. In [61–63], the authors proposed similar systems but also performed some subjective speech quality evaluation showing that the developed system was preferred to a standard configuration at 88.6 % (rejection of 4.3 %) at 130 km/h on highway, and at 50.7 % (rejection of 19.7 %) at 0 km/h with the vehicle parked closed to a highway. These results demonstrate the interest of such system in realistic conditions. Additional intelligibility tests also have shown a 50 % error reduction at 130 km/h on highway. If the components of such systems are basically related to acoustic echo cancellation and noise reduction, the major challenges in this particular application are related to stability issues, processing delay and speech signal distortion resulting from the acoustical cross-coupling.

5.2.3 In-Car Speaker and Speech Recognition

Automatic speech recognition

Automatic speech recognition systems have become a mandatory part of modern man-machine interface (MMI) applications, such as the automotive convenience, navigational and guidance system scenarios, in addition to many other examples such as voice driven service portals and speech driven applications in modern offices.

Modern architectures for automatic speech recognition are mostly software architectures generating a sequence of word hypotheses from an acoustic signal. The dominant technology employs hidden Markov model (HMMs). This technology recognises speech by estimating the likelihood of each phoneme at contiguous, small regions (frames) of the speech signal. Each word in a vocabulary list is specified in terms of its component phonemes. A search procedure is used to determine the sequence of phonemes with the highest likelihood. Modern automatic speech recognition algorithms use monophone or triphone statistical HMMs, the so called acoustic models, based on Bayesian probabilistic and classification theory. Language models, which give the probability of word bigrams or trigrams, are defined using large text corpora. The information provided with the language model is advantageous especially for continuous automatic speech recognition and can substantially increase the performance of the recognition system. This explains the intrinsic difficulty of a speech-tophoneme decoding, where only the acoustic information is available. The simplest ASR mode is the recognition of isolated words as is the case for the in-car hArtes application. Whole-word, monophone or triphone acoustic HMM models can be applied. Realistic word error rates are in the order of less than 5% with vocabularies of approximately 50 words. This ASR mode is usually applied for command driven tasks and is that best suited to the requirements of the hArtes project.

Automatic speaker recognition

The goal of a speaker authentication system is to decide whether a given speech utterance has been pronounced by a claimed client or by an impostor. Most state-of-the-art systems are based on a statistical framework. In this framework, one first needs a probabilistic model of anybodys voice, often called a world model and trained on a large collection of voice recordings of several people. From this generic model, a more specific, client-dependent model is then derived using adaptation techniques, using data from a particular client. One can then estimate the ratio of the likelihood of the data corresponding to some access with respect to the model of the claimed client identity, with the likelihood of the same data with respect to the world model, and accept or reject the access if the likelihood ratio is higher or lower than a given threshold, selected in order to optimise either a low rejection rate, a low acceptance rate, or some combination of both.

Different scenarios can take place, mainly text dependent and text independent speaker authentication

- In the context of text independent speaker authentication systems, where the trained client model would in theory be independent of the precise sentence pronounced by the client, the most used class of models is the Gaussian Mixture Model (GMM) with diagonal covariance matrix, adapted from a world model using Maximum A Posteriori (MAP) adaptation techniques.

- In text dependent speaker authentication, the system associates a sentence with each client speaker. This enables the use of the expected lexical content of the sentence for better modelling and robustness against replay attacks. Furthermore, in the more complex text prompted scenario, the machine prompts the client for a different sentence at every access, which should be even more robust to replay attacks, since the expected sentence for a given access in randomly chosen. On the other hand, models known to efficiently use this lexical information, such as Hidden Markov Models (HMMs), need more resources (in space and time, during enrolment and test) than text independent models.

- A mixed approach, which takes advantages of text-independent and text-dependent systems. In this approach, called GDW (Gaussian Dynamic Warping), the time structural information coming from the pronounced sentences are seen as a constraint of a text-independent system, allowing a balanced training of speaker specific information and text-dependent information. This characteristic allows text prompted speaker authentication systems to be built with a very light enrolment phase, by sharing the large part of the model parameters between the speakers. The main drawback of speaker authentication systems based on time structural information consists in the need of a specific acoustic model for each temporal segment of the sentence. Despite the fact that a large part of the acoustic parameters is shared between the users, the corresponding memory and computation resources are not negligible in the framework of embedded systems. A possible solution consists in building an unique model, to compute the transformation of this model for each of the temporal segments, and to store only the transformation parameters.

5.3 Algorithms description

A detailed description of the selected algorithms will be reported for each of the three fields of application.

5.3.1 Enhanced In-Car Listening Experience

In order to achieve an improvement of the perceived sound, two different tasks can be identified: the former tries to modify the overall behavior of the system, while the latter operates on each loudspeakers channel (Fig. 5.1). Regarding the first objective, a module composed by a digital crossover network in order to split the audio signal in different bands is employed. Then a parametric equalizer for each band is requested to compensate sound coloration due to several resonances in the small car environment. Each parametric (IIR) filter can produce a phase distortion: a phase processor is hence considered in order to correct the phase distortion and to enable adjustments among channels. The second objective comprises audio equalization with two different application scenarios: in the fixed case, the equalization will be done in sub-bands without considering the environment changes and for each frequency band it would be possible to modify the gain; the adaptive scenario provides for an adaptive equalization with reference to some car environment modifications (e.g. number of passengers).
In the following each constituent unit of the system will be described.

Crossover Network

In order to avoid distortions and drivers damages it is necessary to develop a Crossover Network able to split the audio spectrum into desired bands. This network should satisfy some well known design objectives, as expressed in [7]. A linear phase mixed FIR/IIR crossover, satisfying such design requirements, based on a tree structure is employed [7]. The lower branches of the tree are obtained by IIR filtering in order to reduce the overall delay with respect to FIR filtering that would have required thousands of taps. IIR filters are derived from all-pass filters designed with linear phase constraint. A FIR tree is grafted on the upmost branch of the IIR one, obtaining the higher frequency channels, whose filters require less coefficients, and preserving linear phase and ensuring an overall low system delay.

Parametric Equalizer

As pointed out in [6], a Parametric Equalizer is a useful tool in order to correct frequency responses and also to amplify low frequency components masked by the background noise. The parametric equalizer is designed through high-order Butterworth or Chebyshev analog prototype filters [8], generalizing the conventional biquadratic designs and providing flatter passbands and sharper band edges.

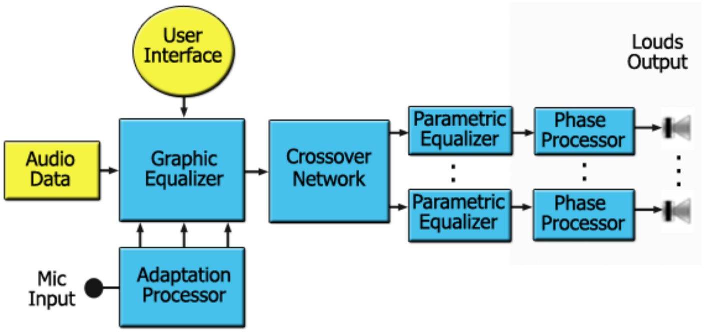


Figure 5.1: Enhanced Listening Experience Algorithms

Phase Processor

In a vehicle cabin the loudspeaker location possibilities are rather restricted. Off-center listening position is inevitable. The delay difference should be compensated through digital delay to equalize the sound arrival times from loudspeakers to the listening position. In this context a Fractional Delay can be used to achieve a better sound alignment. Among the various approaches reported in literature, we chose to resort to a computationally efficient variable fractional delay approach, based on coefficients polynomial fitting [9]. As well as this, a unit capable of rotating the phase spectrum by a certain angle could be introduced, as a mean of phase equalization. This feature has been realized through an all-pass filter with phase specification designed with LSEE approach [10].

Equalizer

Equalization is implemented to enhance tone quality and modify frequency response. Equalizers are used to compensate for speaker placement, listening room characteristics (e.g., in automotive application, to have low frequencies emphasized in the presence of background noise), and to tailor to personal taste (e.g. with relation to particular kinds of music). This compensation is accomplished by cutting or boosting, that is, attenuating or amplifying a range of frequencies. A graphic equalizer is a high-fidelity audio control that allows the user to see graphically and control individually a number of different frequency bands in a stereophonic system. The solution we have adopted is based on an FFT approach ensuring linear phase and low computational cost due to fast FIR filtering implemented in the frequency domain [12]. With respect to efficiency, it is possible to design very efficient high quality digital filters using multirate structures or by using well known frequency domain techniques [12]. Symmetric FIR filters are usually considered for digital equalization because they are inherently linear phase and thus eliminate group delay distortions. An efficient FIR filtering implementation is achieved by using frequency domain techniques as the overlap and add (OLA) method, splitting the entire audio spectrum in octave bands. A proper window, whose bandwidth increases with increasing center frequency, has been used to modulate the filter transitions and the resulting smoothness of the equivalent filter [12]. Starting from the cabin impulse response, a set of fixed curves has been derived to compensate for some spectral characteristics of the car environment. The algorithm can be extended to consider a feedback signal from a microphone near the passenger position to which each gain should be adapted. As fixed and adaptive equalization substantially share the overall architecture, only the adaptive approach will be described. A clear advantage of the proposed solution is that the fixed scheme can be easily turned to an adaptive one, necessary in realistic environment.

5.3.2 Advanced In-Car Communication

In this part the different speech and signal processing components required for improved communication and interaction through an Automatic Speech Recognition (ASR) system are described. The corresponding objectives are the following: hands-free function for in-car mobile telephony, stereophonic echo cancellation for ASR barge-in capability while using the in-car audio system, and speech enhancement for communication and signal pre-processing for ASR. In the following, the corresponding selected algorithms are described, and some results of the integration process in the CarLab using NU-Tech software environment are given.

Monophonic Echo Cancellation

This feature is required for the realization of the hands-free function in order to cancel the acoustic echo signal resulting from the far-end speech emitted on the in-car loudspeakers. Monophonic echo cancellation can also be used to provide barge-in capability to Automatic Speech Recognition (ASR) system when using speech synthesis based Human Machine Interface. The proposed solution should not only handle narrow-band speech but also wide-band speech, in order to cope with the standardised Wide-Band Adaptive Multi-Rate speech codec (AMR-WB) for mobile telephony. For performance and complexity reasons, a Frequency-Domain Adaptive Filter (FDAF) has been implemented. It includes a partitioning structure of the adaptive filter in order to vary independently the length of the identified impulse response and the size of the FFT (Fast Fourier Transform), as introduced in [13]. Additionally, overlapped input data process is used to both improve tracking capability and reduce the algorithmic delay, as described in [14]. The residual output signal is then regenerated using WOLA (Weighted OverLap-and-Add).

Stereophonic Echo Cancellation

Extended FDAF algorithm to the stereophonic case has been also considered in order to cancel the acoustic echo signals coming from the in-car embedded loudspeakers when operating the ASR system while the radio is turned on.

When generalizing the acoustic echo cancellation to the multi-channel case, one has to deal with the cross-channel correlation. This problem has been described in [15]. A quasi-optimal frequency-domain implementation taking into consideration both the correlation between sub-blocks resulting from the partitioning process and the cross-channel correlations has been proposed in [16]. This algorithm is particularly well-suited for long impulse responses. Due to shorter impulse responses encountered in the car environment no significant gain is expected. A simple sub-optimal extension of the monophonic FADF has been implemented, using the global residual signal for the adaptation of each adaptive filter.

Single-Channel Speech Enhancement

Noisy speech signals picked-up in a car environment have been one of the major motivations for the development of single-channel speech enhancement techniques either for improved speech quality for phone communication or ASR applications. Methods proposed in [17] and [18] are considered as state-of-the-art approaches especially regarding the naturalness of the restored speech signal. These algorithms implement respectively an MMSE (Minimum Mean Squared Error) estimator of the short-term spectral and log-spectral magnitude of speech. The spectral components of the clean signal and of the noise process are assumed to be statistically independent Gaussian random variables. The log-MMSE Ephraim & Mallah algorithm has been implemented together with a continuous noise estimation algorithm as described in [19]. This algorithm has been designed to deal with non-stationary environments, and remove the need for explicit voice activity detection.

Multi-Channel Speech Enhancement

The use of several microphones for speech enhancement has been also studied for in-car applications. In general, the different microphones signals are first time-aligned (beam-forming), and the resulting signal is then further processed either using adaptive filtering techniques or adaptive post-filtering. For cost and complexity reasons, a four microphone array has been studied using cardioid microphone (AKG C400 BL) with 5 cm spacing. The different channels are continuously synchronized using appropriate time delays and summed together to form the output signal of the beam-former. The array is steered in different possible directions and the selected direction is the one which maximized the output power of the steered beam-former, known as the Steered Response Power (SRP) criterion. Improved performance is obtained by using the Phase Transform (PHAT) as a spectral weighting function. An evaluation of the resulting SRP-PHAT criterion can be found in [20]. Different post-filters have been implemented to further reduce non-localized noise interferences at the output of a beam-former. The first commonly used post-filter is known as Zelinski post-filter, as described in [21], which is an adaptive Wiener post-filter applied to the output of the beam-former. Various improvements have also been considered as proposed in [22], [23], [24] and [25].

5.3.3 In-Car Speaker and Speech Recognition

The automatic speech and speaker identification modules provide an improved man-machine interface (MMI). Several functionalities could be based on these modules, in order to improve the conviviality of the user interface. A speech recognition system provides hands-free operations well suited for various in-car devices and services such as: radio equipment, multimedia management, CD player, video player and communications systems (mobile phones, PDA, GPS, etc.). Isolated/connected word recognition is a simpler task than continuous speech recognition (which allows to integrate a speech recognition system into an ACIS). Finding a command word in a continuous audio stream is one of the hardest speech recognition tasks, so we decide to use a push-when-talk approach (driver should press a button before saying the command to recognize). Automatic speaker recognition provides user authentication (driver) for secured access to multimedia services and/or specific user profiles (seat settings, favorite radio station, in-car temperature, etc.). For embedded speech/speaker recognition, the highlights are usually the memory and computational constraints. But realistic embedded applications are also linked to several scenario constraints, giving short speech utterances, few training data and very noisy environments. Also, such systems should work with degraded speech quality. Here, two applications are targeted: isolated/connected word recognition (for command and control) and speaker identification (to load driver parameters). The ASR module has been developed using Hidden Markov Models (HMMs) [26] and is based upon LIASTOK [28]. The development is combined with the speech and audio modules for speech pre-processing. Referring to Fig. 5.2, we begin with a Gaussian Mixture Model (GMM) which represents the whole acoustic space and then derive from it the necessary HMM-state probability density functions to represent each phoneme. This approach to model acoustic space was fully presented in [27]. A word is so composed of a number of sequential phonemes. As the recognizer is phoneme-based, it allows a lexicon easily modifiable. This is deemed to sit particularly well with the hArtes paradigm. It will allow to adjust the vocabulary, according

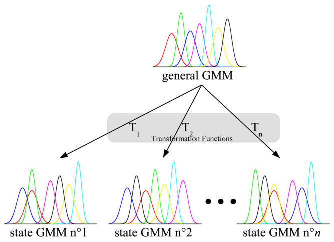


Figure 5.2: General approach for speech recognition

to modifications made to the ACIS or as new applications or functionalities are introduced. In addition, the lexicon can contain any number of different phonetic transcriptions per vocabulary item thus assisting recognition with different pronunciations and accents. Moreover this approach used for acoustic modeling allows easy adaptation to a speaker or to a new environment [28]. The remainder of the decoding algorithm is essentially a traditional Viterbi algorithm [29]. Likelihoods are calculated on the fly with highly efficient and sophisticated likelihood calculations. Given an audio recording the likelihood of the observed data given each model is calculated and that which produces the greatest likelihood defines the decoded or recognized word. The goal of a speaker authentication system is to decide whether a given speech utterance was pronounced by the claimed speaker or by an impostor. Most state-of-the-art systems are based on a statistical framework (Fig. 5.3). In it, one first needs a probabilistic model of anybody's voice, often called Universal Background Model (UBM or world model) and trained on a large collection of voice recordings of several people. From this generic model, a more specific, client-dependent model is then derived using adaptation techniques, using data from

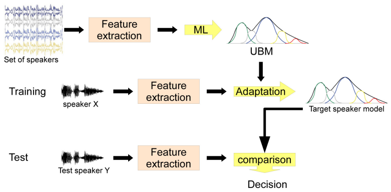


Figure 5.3: General approach for speaker recognition

a particular client. Our work is based on ALIZE [30] toolkit and dedicated to such embedded applications. To save both memory and computation time, only a few part of UBM components are adapted and saved (other mean parameters, weight parameters and variance parameters are taken from the UBM model) like in [31] or [32]. One can then estimate the ratio of the likelihood of the data corresponding to some access with respect to the model of the claimed client identity, with the likelihood of the same data with respect to the world model, and accept or reject the access if the likelihood ratio is higher or lower than a given threshold.

5.4 Applying the hArtes Toolchain

A complete description of the hArtes toolchain application on the selected algorithms follows.

5.4.1 GAETool Implementation

The hArtes Graphical Algorithms Exploration (GAE) tool is the highest level of the hArtes toolchain based on the NU-Tech Framework [2, 3] (Fig. 5.5). It can be seen as a development environment for the interaction of DSP algorithms; the user can program his own C/C++ objects separately and then consider them as blocks, namely NUTS (NU-Tech Satellites), to be plugged-in within the available graphical design framework. Each NUTS is fully configurable to provide the needed number of inputs and outputs. Therefore GAE tool has two basic functionalities:

1. Assist the designers to instrument and possibly improve the input algorithm at the highest level of abstraction so that feedback concerning the numerical and other high-level algorithmic properties can be easily obtained.

2. Translate the input algorithms described in different formats and languages into a single internal description common for the tools to be employed further on. In the hArtes tool-chain, this internal description will be done in ANSI C language.

The GAE-tool accepts existing (e.g. NU-Tech library by Leaff) or new functional blocks written in C as input description. Integration of existing functional blocks allows component reuse: project-to-project and system-to-system. The GAE-tool also supports reconfigurable computing in the very initial phase allowing different descriptions of the same part of the application to let hardware/software consequent partitioning explore different implementations with respect to different reconfigurable architectures. Besides the C based description of the input application, outputs of the GAE tool contain directives provided by the user and profiling information collected during algorithm development. Possible user directives concern parallelism, bit width sizing, computing type (e.g. systematic vs. control) and all the parameters important to optimize the partitioning / mapping of the application on the available hardware. Possible profiling information collected by the GAE tool is the processor usage per single functional block with respect to the whole network of functional blocks and the list of possible bottlenecks. Exploiting NU-Tech plug-in architecture, every modules of the applications have been developed as C NUTS with associated settings window, useful to change internal parameters. Fig. 5.5 shows the GAEtool integration of single channel speech enhancement.

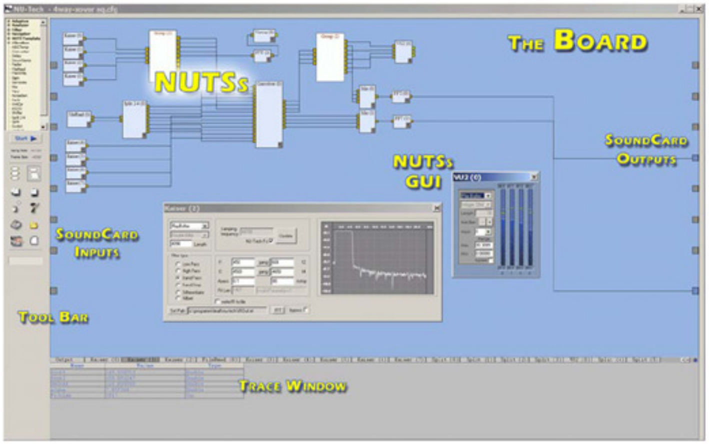


Figure 5.4: NU-Tech Framework

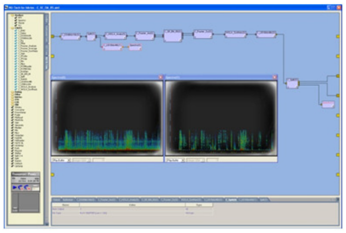


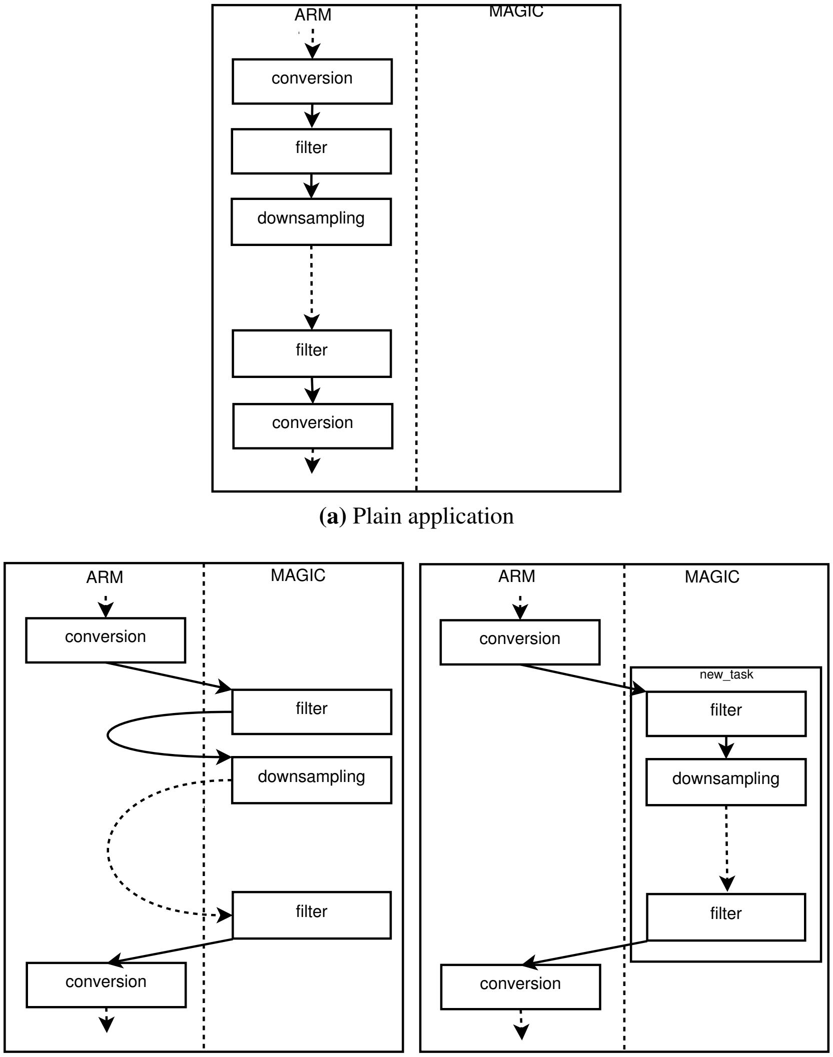
Figure 5.5: NU-Tech integration of single-channel speech enhancement

5.4.2 Task Partitioning

Zebu tool has been applied to a reduced version of the Advanced In-Car Communication application. The reduced version consists of the pre-processing and the post-processing processes required by the Stationary Noise Filter. The main kernel of the application is characterized by a sequence of transformations applied to the sound samples: Fourier transforms, filters, Weighted Overlap-and-Add, downsampling and upsampling.
The application has been partitioned and mapped by Zebu for the execution on the Atmel's DIOPSIS board and the performance of the resulting versions of the application are measured directly on the board.
Figure 5.6(a) shows the plain execution of the kernel, without exploiting the DSP processor. Then, Figure 5.6(b) shows the effects of applying only the mapping, that assigns most of the functions for the execution on the DSP. In this case, we obtained a speed-up equals to 4.3x. However, the data exchanged among the different functions are still allocated on the main memory since it is the master processor which at each function invocation has to pass and read back the data processed by the single function.
Finally, if Zebu performs also task transformations, such as sequential partitioning, it partitions the kernel of the application by grouping together the functions with highest data exchanging into a new task function `new_task`, as shown in Figure 5.6(c). This task is then mapped on the DSP, so the functionalities mapped on the DSP are actually the same of the previous case. However, since these functionalities are now grouped in a single task, the data exchanged among the functions are localized and do not need to be moved anymore inside and outside the DSP memory at each function call. As a result, this partitioning transformation increases the speed-up from 4.3x to 5.8x.

5.4.3 Task Mapping

In this section, we report the results achieved with the hArmonic tool (Section 2.7.6) on the Audio Enhancement and Stationary Noise Filter applications using the automatic mapping approach.
Table 5.1 summarizes the main results. The total number of tasks measures the number of calls in an application, where each invoked function can be executed on a particular processing element. Note that the number of tasks is not necessarily an indication of the size of the application since each task can be of a different size granularity, however the number of tasks can affect the time to produce a mapping solution. The number of tasks mapped to DSP measures the fraction of the application that is executed outside the main processing el-



(a) Plain application

(b) Mapped application

(c) Partitioned and Mapped application

Figure 5.6: Effects of partitioning and mapping on the kernel application.

ement (ARM). The time to compute a solution corresponds to the total time required by the hArmonic tool to process each application, and the speedup corresponds to the ratio between the execution time of the application running on the ARM and the performance of the same application using the mapping solution derived by the hArmonic tool.

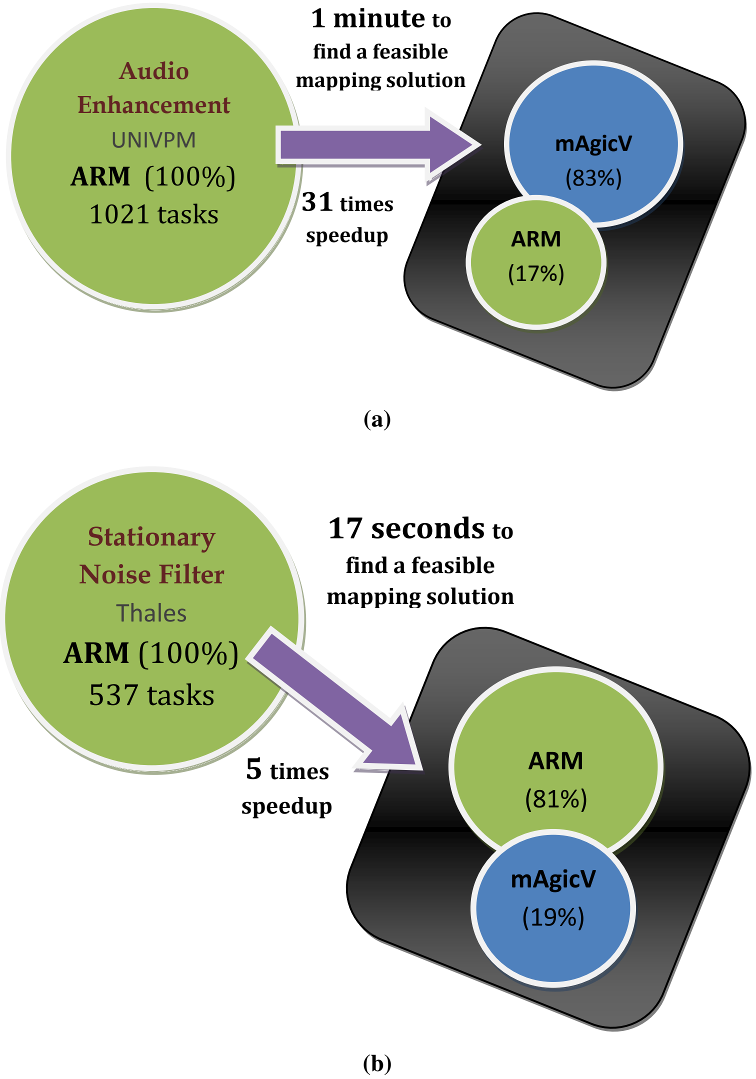
In addition to the Audio Enhancement application, Table 5.1 also presents the

Table 5.1: Evaluation of the automatic mapping approach on the Audio Enhancement application and related kernels (Xfir, PEQ, FracShift, Octave), and the Stationary Noise Filter. The speedup corresponds to the ratio between the performance of the application running on the ARM and the mapping solution produced by the hArmonic tool.

hArtes Application	**Total Number of Tasks**	**Number of Tasks Mapped to DSP**	**Time to compute solution (sec)**	**Speed up**
Xfir	313	250	8	9.8
PEQ	87	39	3	9.1
FracShift	79	13	24	40.7
Octave	201	137	6	93.3
Audio Enhancement	1021	854	59	31.6
Stationary Noise Filter	537	100	17	5.2

results of four of its kernels (Xfir, PEQ, FracShift, Octave) which are executed and evaluated independently. The number of tasks for the audio enhancement kernels is relatively small (from 79 to 313) and for three of the kernels, hArmonic produces an optimized mapping solution in less than 10 seconds, with speedups ranging from 9 times to 93 times.

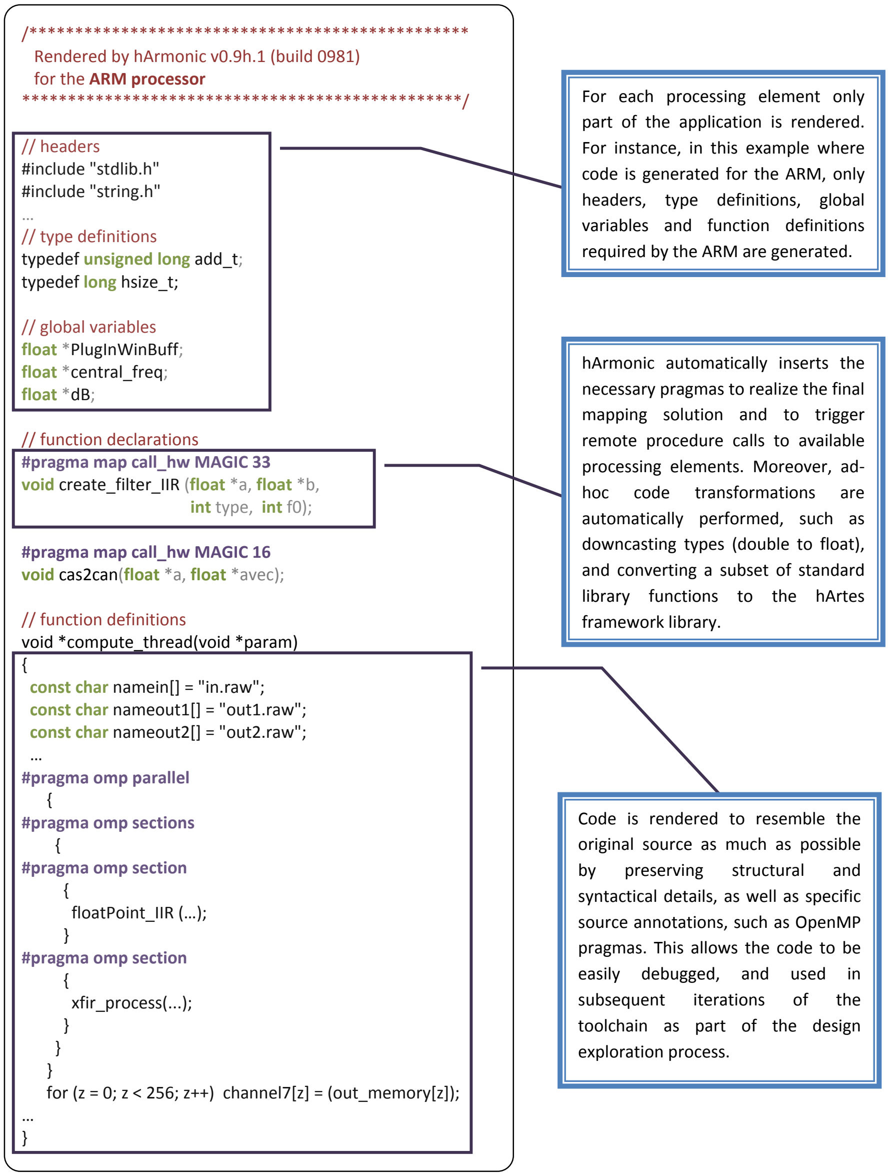
The Audio Enhancement application has 1021 tasks and is the largest application in terms of the number of tasks evaluated by the hArmonic tool (Fig. 5.7 (a)). The hArmonic tool takes 61 seconds to produce an optimized solution where 83% of the application is executed on the mAgicV DSP, and exibits an acceleration performing 31 times faster. On the other hand, the Stationary Noise Filter has 537 tasks; hArmonic takes 17 seconds to produce a mapping solution that results in a 5 times speedup. The DSP coverage in this case is only 19%, however the task granularity of this application is understood to be on average larger than the Audio Enhancement application. Another difference between both applications is the use of DSPLib (part of the hArtes standard library), which in the case of the Stationary Noise filter is not used and therefore the mapping solution does not yield a performance improvement as large as that for the Audio Enhancement. However, since the Stationary Noise filter does not rely on the hArtes standard library, it allows the application to be compiled and run on other platforms, such as the PC. Note that the magnitude of the acceleration on both applications is influenced by many factors: the use of optimized libraries, manual management of the memory footprint, the quality of the synthesis toolbox (backend compilers such as hgcc and targetcc) and the hArtes hardware platform.



(a)

(b)

Figure 5.7: Main results of the hArmonic tool in combination with the synthesis toolbox and the hArtes hardware platform: **(a)** Audio Enhancement application and **(b)** Stationary Noise Filter.



```
/**************************************************
  Rendered by hArmonic v0.9h.1 (build 0981)
  for the ARM processor
**************************************************/

// headers
#include "stdlib.h"
#include "string.h"
...
// type definitions
typedef unsigned long add_t;
typedef long hsize_t;

// global variables
float *PlugInWinBuff;
float *central_freq;
float *dB;

// function declarations
#pragma map call_hw MAGIC 33
void create_filter_IIR (float *a, float *b,
                        int type,  int f0);

#pragma map call_hw MAGIC 16
void cas2can(float *a, float *avec);

// function definitions
void *compute_thread(void *param)
{
  const char namein[] = "in.raw";
  const char nameout1[] = "out1.raw";
  const char nameout2[] = "out2.raw";
  ...
#pragma omp parallel
   {
#pragma omp sections
     {
#pragma omp section
       {
         floatPoint_IIR (...);
       }
#pragma omp section
       {
         xfir_process(...);
       }
     }
   }
   for (z = 0; z < 256; z++)  channel7[z] = (out_memory[z]);
...
}
```

For each processing element only part of the application is rendered. For instance, in this example where code is generated for the ARM, only headers, type definitions, global variables and function definitions required by the ARM are generated.

hArmonic automatically inserts the necessary pragmas to realize the final mapping solution and to trigger remote procedure calls to available processing elements. Moreover, ad-hoc code transformations are automatically performed, such as downcasting types (double to float), and converting a subset of standard library functions to the hArtes framework library.

Code is rendered to resemble the original source as much as possible by preserving structural and syntactical details, as well as specific source annotations, such as OpenMP pragmas. This allows the code to be easily debugged, and used in subsequent iterations of the toolchain as part of the design exploration process.

Figure 5.8: Main features of the code generated by the hArmonic tool for each back-end compiler in the synthesis toolbox. In this figure, we focus on the code generated for the ARM processor.

The results are achieved using the hArmonic tool in combination with the synthesis box of the hArtes toolchain on a Core 2 Duo machine with 4GB RAM and running on the hArtes hardware platform. The synthesis box is responsible for compiling the source targetting each processing element and then linking them to form one binary. The hArmonic tool generates the source-code for each processing element in the system based on the mapping selection process. In particular, each source-code generated corresponds to part of the application and hArmonic computes the minimal subset (headers, types, variables and functions) required to successfully complete the compilation process for each part of the application. Fig. 5.8 illustrates the main features of the code generated for the ARM.

5.5 Experimental Results

Starting from a detailed description of the ACIS architecture, firstly a first evaluation of the algorithms functionality will be reported using a PC-based framework, then the final implementation on the hArtes embedded platform will described.

5.5.1 Car Lab Prototype

The selected vehicle for the demonstrator is a Mercedes R320 CDi V6 Sport car, which has been chosen because of space needed by the Carputer hardware and audio system. The car has been equipped with up to 30 loudspeakers and 24 microphones located inside the car according to algorithms development requirements Fig. 5.9.
In particular ad hoc loudspeaker set has been designed and produced. The resulting audio reproduction system outperforms current production systems in terms of performance and allows also the exploration of a large variety of audio processing algorithms (CarLab requirement). During algorithm exploration, the system can be freely downgraded and reconfigured, switching on and off different audio channels. The system consists of a full three-way stereo subsystem for the first row of seats (6 loudspeakers in 2 sets of woofer, midrange and tweeter), a full three-way stereo subsystem for the second row of seats (other 6 loudspeakers) and a two-way subsystem (4 loudspeakers in 2 sets of mid-woofer, tweeter) for the third row (Fig.5.10 and Fig. 5.11). One high performance subwoofer, located in the trunk, helps the subsystems to reproduce the lower audio frequencies. In this way, a complete set of loudspeakers able to cover with time coherence the whole audio frequency range is installed

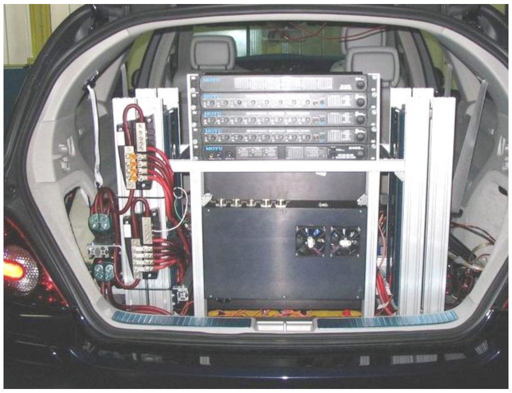


Figure 5.9: PowerServer final setup in the car trunk.

near each passenger. A suitable set of 9 specific loudspeakers has been developed for communication applications. It covers the full speech frequency range and has been mounted on the roof-top with almost one loudspeaker very close to each passenger ears, ensuring the transmission of a clean and high intensity sound with a relatively low absolute pressure level thus improving the signal to noise ratio and minimizing the signal energy returning into the microphones. Special effort has been devoted to contain loudspeaker nonlinear behaviors to avoid undesired effects. At high pressure level the system should remain weakly nonlinear in order to maintain signal quality and allow hArtes applications to perform correctly.

The ACIS system has to control each single loudspeaker channel for implementing high-end audio/video algorithms. Therefore every single loudspeaker has a dedicated high quality automotive power amplifier. Professional ASIO soundcards have been used to manage all the system I/Os. Off-the shelf components have been chosen to minimize the overall development process and focus on the demonstration of hArtes approach for applications development. Weight, size and power consumption considerations have been overcome by

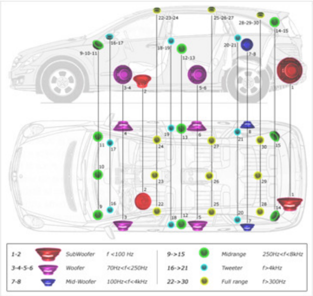


Figure 5.10: Loudspeakers position

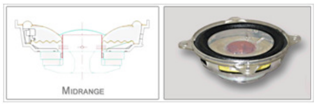


Figure 5.11: Example of a midrange loudspeaker

performances issues.

5.5.2 PC-based In-Car application

As previously stated, a working ACIS system has been implemented to show the capabilities of the hArtes platform. However, in order to develop a proof-of-concept, a two step strategy has been employed: first a Carputer-based CIS has been developed and installed in a test vehicle as a test bench for the in-car multichannel audio system and the in-car audio algorithms, then the final ACIS proof of concept has been designed using the entire hartes platform.
In this part, we give an overview of the hArtes integration process, through the use of NU-Tech software environment tool for demonstrating the different algorithmic solutions on a PC-based reference platform. The selected algorithms have been developed independently using standard ANSI C-language and libraries and split in elementary software components. These components are homogeneous and have been specifically structured for an efficient reusability. Some optimizations have been achieved using specific IPP (Intel Performance Primitives) library, which is well-suited for the realization of a reference real-time demonstrator. Since the developed algorithms mainly rely on FFT and IFFT calculations, we provide some performances figures for 1024-points Direct and Inverse Complex Fast Fourier Transforms in Table 1.

Table 5.2: Average Execution Time of Direct and Inverse 1024-pts Complex FFT For Different Optimization Levels

	C-Code without compiler optimization (reference)		C-Code with compiler optimization		IPP-Code with compiler optimization	
1024-pts D-FFT	444993 cycles	0.1859 ms	173656 cycles	0.0725 ms	17203 cycles	0.0072 ms
Gain	1.00		2.56		25.87	
1024-pts I-FFT	450652 cycles	0.1882 ms	179449 cycles	0.075 ms	16446 cycles	0.0069 ms
Gain	1.00 (reference)		2.51		27.40	

Enhanced In-Car Listening Experience

Tests have been performed in order to evaluate the performances of the algorithms as NU-Tech plug-in, considering the car environment with the overall audio system. Two different validation sessions have been considered:

after having verified the algorithms functionalities by synthetic simulations, the proposed algorithms have been tested inside the hArtes CarLab reproducing audio material inside the car cockpit. Some preliminary subjective listening tests have also been conducted to assess the perceived audio quality. For the first part, each algorithm has been separately validated defining an elicited criterion to verify its correct functionality: for the sake of brevity, the entire validation of this part will not be illustrated (for more details see [4]). Concerning in-car validation, only test sessions relative to the adaptive equalizer will be reported, due to its greater significance w.r.t static algorithms. A microphone (AKG 417L) has been positioned on the roof near the driver seat and 12 loudspeakers were driven by a crossover network (Fig. 5.10, Loudspeakers 1, 3, 4, 5, 6, 9, 10, 11, 16, 17, 19, 18, 12, 13). A preliminary recording session considering white noise as input was performed. The target equalization curve was a flat one in order to better underline the equalization capability. As shown in Fig. 5.11, the frequency response of the microphone signal with the equalizer turned on presents a flatter performance and dips and peaks are attenuated w.r.t. the non-equalized signal. The corresponding equalizer at a given time is depicted in Fig. 5.13 together with the measured frequency response prototype: weights at high frequencies are thresholded

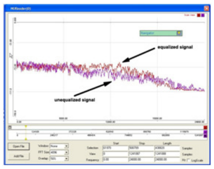


Figure 5.12: Effect of adaptive stereo equalization on car

to prevent excessive gains. Informal listening tests have been conducted by reproducing audio material to evaluate the perceptive effect of the overall system (Fig. 5.1): the listeners were asked to sit into the car at the driver position and to use a suitable SW interface available on the dashboard to modify the algorithm parameters and to change the audio source. Preliminary results seem to confirm the validity of the proposed approach since all subjects involved have reported positive comments and impressions on the global perceived sound image. Further details will be reported in future publications.

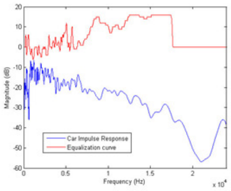


Figure 5.13: Smoothed car impulse response prototype and equalization weights during adaptive equalization

Advanced In-Car Communication

Monophonic and stereophonic echo cancellation algorithms have also been integrated in NU-Tech environment. Various experiments have been carried out with some real in-car recordings for the validation of the implemented algorithms. The Signal to Noise Ration (SNR) improvement obtained with the single-channel speech enhancement is on average between 10 dB and 14 dB. The result of the stereophonic echo cancellation for a stereo music sequence is

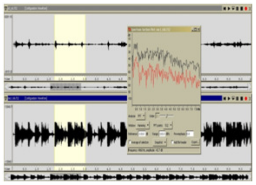


Figure 5.14: Stereophonic Echo Cancellation of a music sequence

displayed on Fig. 5.14. The averaged Echo Return Loss Enhancement (ERLE) is between 25 to 30 dB depending on the music sequence, when using a filter length of 512 to 1024 taps at 16 kHz sampling frequency. Since the stereophonic echo cancellation is intended to be used as a pre-processing stage to Automatic Speech Recognition (ASR), no post-filter has been implemented in order to avoid distortion on near-end speech, as it can be seen on Fig. 5.14. For the monophonic echo canceller, a frequency-domain post-filter has been included in order to reduce the residual echo when used for hands-free telephony. The Impulse Response (IR) measured in the car at a microphone located on the mirror, has been considered less than 50 ms, when all the loudspeakers of the car were active, which could be considered as the worst configuration. The result of this criterion using one single frame of speech uttered at the driver's location is depicted on Fig. 5.15. When used in the final integration, the SRP-PHAT criterion is estimated on a reduced set of candidate locations in order to significantly reduce the overall complexity. Assuming that the speaker's location is known in advance, we can limit the search using some hypothesis on the angle and radial ranges. This is illustrated on Fig. 5.16.

In-Car Speaker and Speech Recognition

Three libraries have been developed for the In Car Speaker and Speech recognition application: one for speech parameterization (which is common for speech and speaker recognition applications and aims to extract relevant information from speech material), one for speech recognition and one for speaker authentication. The two first libraries have been developed using C-ANSI language while speaker authentication modules used C++ language. Both languages allow a great integration with the NU-Tech Framework.
Speaker Verification algorithms have been used for user authentication to the CarLab system using a push-to-talk approach (the user has to say his user name and password) while Speech Recognition algorithms have been used both for

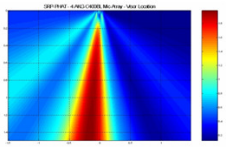


Figure 5.15: Finally, the multi-channel speech enhancement module has been implemented in NU-Tech, using the SRP-PHAT criterion for on-line speaker localization.

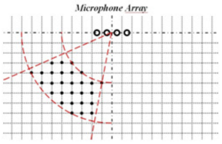


Figure 5.16: Limited search using region-constrained estimation of SRP-PHAT criterion.

the authentication system (user password recognition) and for the Audio Playback Module where a list of 21 commands have to be recognized during music playback (Play, Stop, Pause, Volume Up, Exit, etc...). Recording sessions inside the car under a greater range of conditions - including real running conditions - have been performed in order to acquire 19 different users (15 male and 4 female) and significantly improve the acoustic model adopted. All algorithms used for speech and speaker recognition systems are tested and evaluated during several evaluation campaigns. For speech recognition, in similar acoustic conditions some tests on French digit recognition allow an accuracy rate higher than 97% (performing around 2300 tests). On a voice command task, with the same kind of audio records, the accuracy rate is around 95% (on 11136 tests). A complete presentation of these results could be found in [33]. In both cases, the audio data are recorded in several cars with A/C turn on/off, radio turn on/off, widows opening or not, etc. Speaker recognition algorithms are also tested and evaluated with participation on the international campaign of speaker recognition: NIST-SRE [34]. The LIA results obtained during this campaign allows to be in the TOP10 of the participants.

5.5.3 Embedded Platform In-Car application

As described in Sec.5.5.2, after the first step of developing a Carputer-based prototype for the validation of the system functionalities, the hartes hardware has been integrated within the already developed system. Fig.5.17 shows how the hartes platform has been integrated; a specific software is available on a PC for loading code on the platform and implementing the application interface with the system, based on NU-Tech platform [2]. The connection between the hartes board and the audio system (i.e. professional soundcards) has been realized considering the ADAT protocol, while a TCP-IP connection has been used to manage the algorithm parameters through a graphical interface PC-based.

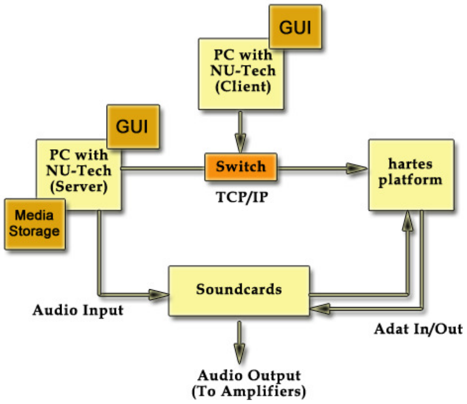


Figure 5.17: Scheme of the Power Server System considering the hartes platform

Starting from the applications, four different approaches has been evaluated through the toolchain:

- Direct approach, considering a manual implementation on the board just to have a benchmark; two different C projects have been developed, one for ARM processor and one for mAgicV. While ARM is used to control the user interface, mAgicV is used to perform the DSP operations, to configure the audio interface, and to control the input/output audio flow.
- No mapping approach; the application is entirely compiled for the ARM processor, providing a working reference and allowing the evaluation of the basic performances for the application porting on DEB.
- Manual mapping approach (i.e. martes approach), considering the toolchain with a manual partitioning and mapping of each part of the algorithm considering a specific hardware;in this way, it is possible to obtain the maximum performances of the application with respect to the system.
- Automatic mapping and partitioning (i.e. DSE approach), considering the overall functionality of the toolchain (i.e. automatic task partitioning and mapping). The hArtes tools find automatically the optimal partitioning/mapping, in order to evaluate performances with a very low human effort.

Performances of each algorithm will be considered in terms of workload and memory usage taking into account the Diopsis evaluation board (i.e. DEB) and the entire hArtes platform(i.e. hHp). In the case of workload requirement, it has defined as follows

$$Workload = \frac{T_a}{T_s}\% \tag{5.1}$$

where T_a is the time required to perform the entire algorithm within a single frame while T_s is time required for realtime application for the same frame size, i.e. considering the frequency sampling of the algorithm and the clock cycles.
For the sake of brevity, just the results of some algorithms of the entire In-Car application will be reported in this section.

Enhanced In-Car Listening Experience

Using the entire toolchain, as described in Sec.5.4, the whole Audio Enhancement application has been developed and uploaded in the hArtes Platform. A

similar configuration of the PC based prototype has been considered; also in this case, the final configuration has defined by two Equalizers, four Crossover Networks of three channels, and twelve Phase Processor to cover four audio front composed by a tweeter, a midrange and a woofer (i.e. twelve total channels). Then a low-pass filtering and a high-pass filtering have been considered respectively for the central channel and for one subwoofer.
Regarding the hardware mapping of each part of the entire algorithm, two different solution can be presented. Taking into account a manual mapping in the case of the direct implementation and the martes approach, the overall application has been divided on each part of the platform as depicted in Fig.5.18. Considering the automatic mapping, each part of the entire application is mainly mapped as in the martes approach, except for some vectors allocations and I/O processing that are mapped in the ARM processor and for the IIR filtering that is mapped in the DSP.
For the workload analysis, Eq.5.1 has been considered deriving T_s for a sampling frequency of $48kHz$, a frame size of 256 sampling and a clock cycle of $100Mhz$. At each frame a min and max values of the time required (T_a) is derived: the final parameter is calculated considering a temporal mean of each value. A logarithmic scale has been considered to better underline the results. Obviously, having good performances is equivalent to have a workload less than 100%, in order to have a real time application.

Considering Tab.5.3, it is possible to assess the performances of the entire toolchain in terms of workload, where No-mapping approach refers to the entire application running on ARM, overlooking other processors, and DSE refers to the automatic approach. The final results are very similar to those obtained considering a manual implementation both for the martes and for DSE approach (Fig.5.19). This is due to a good decision mapping applied by the user since the multichannel application lends itself to an easy partitioning. Moreover, considering the no-mapping approach that describes the natural way to write a code, it is clear a great advantage considering the toolchain: this confirms the validity of the methodology.
A great advantage could be found considering also concurrent applications; in this case, taking into account the martes approach, IIR filtering applications performed over the FPGA have been parallelized with respect to the DSP elaboration. As we can see from Fig.5.20, it is clear a great reduction of the overall workload of about 15%. Considering separately the performances of the three processor ARM, DSP and FPGA, each workload and memory requirement can be shown. Fig.5.21a and Fig.5.21b demonstrate that the DSP

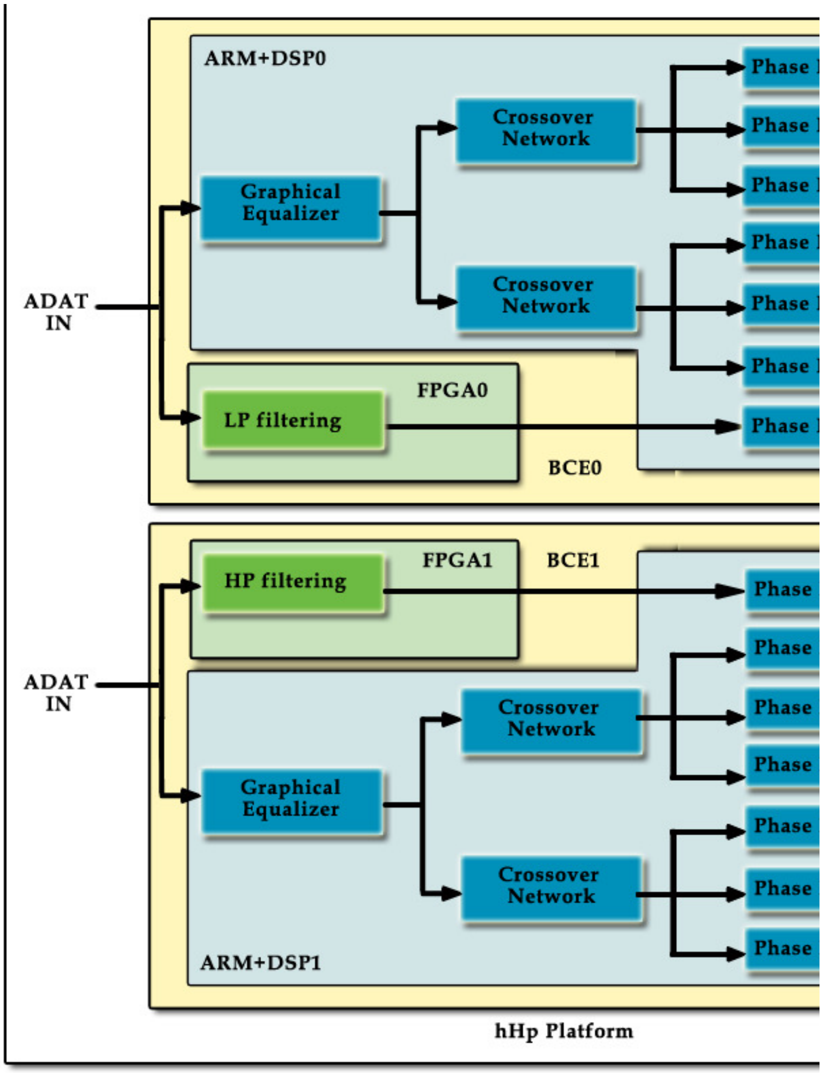


Figure 5.18: Scheme of the overall Audio Enhancement In-Car application developed for the hHp platform.

performs most intensive operation requiring the greater workload while the ARM processor requires more memory due especially to the I/O management. FPGA processor shows less performances since the algorithm performs is less intensive than the others.

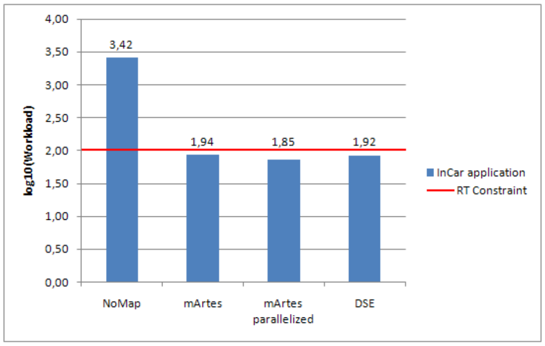


Figure 5.19: In Car application performances considering different approaches

Therefore the validity of the toolchain has been assessed and the following remarks can be done comparing it with the manual implementation:

- The manual implementation requires a deep knowledge of the embedded platform (i.e. audio input/output management, memory allocation/communication (Dynamic Memory Acces) and parallelization using thread programming) and the implementation performances are strictly related to the subjective capability. Using the hartes toolchain, all these aspects can be avoided.

- On the other hands, the toolchain provides an automatic mapping of the application in case of lack of developer experience but it could be considered as a proof where a first assumption of manual mapping is done.

All these aspects imply great advantages in terms of development time: start-

Table 5.3: Performances of the martes and DSE approach in terms of workload for each algorithm with relation to the direct implementation

	Workload			
Algorithms	Direct Implementation	No-mapping	mArtes	DSE
Graphic Equalizer	7,30	873,6	7,52	7,69
Crossover Network	7,48	76,8	8,07	8,30
Phase Processor	5,76	323,5	5,94	6,06
IIR Filter	-	31	16,25	10,7

ing from an idea implemented and tested on a PC using ANSI C language, it is possible to realize directly an embedded application representing a first prototype and then reducing the time to market.

Advanced In-Car Communication

The advanced In-Car set of applications has been first integrated in the CarLab demonstrator in order to verify and improve their behavior in real environment. This work has been done through the creation of a set of 26 NU-Tech CNUTS allowing the execution of the algorithms on any PC computer equipped with low latency audio soundcard. The stationary noise filter, the echo canceller and the beam former were successfully integrated and easily combined together to provide enhanced In-Car experience. The processing power required for the most complete applications is really important and being able to use them on embedded systems was really challenging. The CarLab multi-core computer was able to handle our most complete solution and the challenge of hArtes project was to be able to execute the exact same application on the provided hardware without having to completely redesign the application to fit with the hardware constraints. The hArtes hardware selected for the Advanced In-Car Communication is the DEB that is based on the ATMEL Diopsis, low power, heterogeneous System on Chip providing two different processors: one General Purpose Processor and one Digital Signal Processor. The mAgic DSP, is a Very Long Instruction Word class processor which provide a symmetrical architecture optimized for Fourier space calculation in float representation. This architecture is dedicated to achieve complex algebra by providing two data path each composed with a pipeline of 5 operators. The biggest advantage of such processor is its impressive efficiency to calculate Fast Fourier Transform requiring a really low amount of energy. It is one of the architecture on the current DSP market which require the less number of cycles to process it.

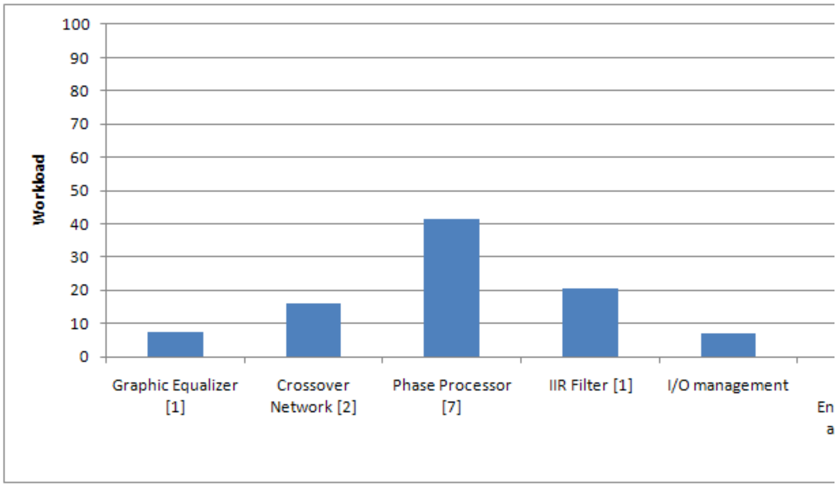
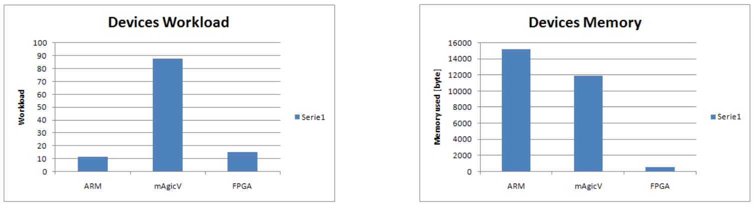


Figure 5.20: Workload of the overall Audio Enhancement system considering a parallelization of the applications

The properties of such processor allowed us to select one of our high performance application as a candidate to be executable in real time. We selected the Stationary Noise Filter in the GSM configuration and demonstrated its implementation using all the compilation techniques proposed by the hArtes tool chain. The required processing power of that only application is high enough



(a) (b)

Figure 5.21: (a) Devices workload and (b) Devices memory requirements for the Audio Enhancement system in case of parallelized applications

to consume most of the available power on this architecture so we decided to keep it as small as possible so we have a chance to make it working in real time on the hardware. One of the constraint that we wanted to keep all along the project is the portability of the source code, so we decided not to use the hand optimized libraries of the DSP to do signal analysis. This decision, allowed us to maintain our ability to modify and test the code on a PC, using for example the high level algorithm exploration tools like the NU-Tech software but made us waste the resources offered by the architecture. This trade off between development easiness and final efficiency is a real concern in THALES business areas, because the development cost of a software represent a significant amount in the cost of a product. The implementation we chose then allowed us to be able to migrate easily part of the software from the PC to the GPP and finally to the DSP. This way of handling software on this kind of heterogeneous hardware is one of the most efficient because we are able to design an algorithm on the PC, quickly validate its usability on the embedded GPP using the full feature operating system it provides and then finally use the DSP to handle the final application and reach the Real-Time performance. The ATMEL Diopsis heterogeneous hardware introduced some limitations in that portability because of two technical issue that we encountered. The ARM926EJ-S is a 32 bit processor used in a little endian mode that do have access to internal and external memories without any kind of restriction in term of data alignment or data size. This great connectivity is unfortunately not true for the mAgic DSP. The mAgic is a 40bit processor with 16000 words of 40 bit memory available. The first technical issue is the small amount of memory that we are able to use without having to manually use the DMA controller to access the external memory. The second limitation is the data alignment inside of the 40bit memory. Numbers with float representation do exploit the 40bits while numbers with integer representation only exploit 32bits. This means that data transmitted from the GPP to the DSP have to be converted in order to be in the appropriate format and imply that we can not transmit at the same time float and integer data because the hardware doesn't have a way to know which part of the transmitted data is to be converted or not. These two limitations made us completely rewrite the source code of the application in order to split the code into several modules. The low amount of available data memory made us split the calculation and send it to the DSP in several times. The goal was to send a small enough amount of data to the DSP, make it process them and get the result back. Doing that for each part of the algorithm and making the GPP control that process allowed us to be able to overcome the memory size limitation. Furthermore, in order to be able to send the full amount of data required by a module in one request, we needed to send one homogeneous block of data containing numbers all using the same representation (float or int) in order to avoid the data representation conversion problem. This was achieved by using only float in the whole application, even for integers numbers, and by taking care of the floating point operators to make them approximate the values correctly. After identifying these limitations and understood how deep was their impact on the tool chain, we wrote some good practices and showed to all partners an easy way to handle the two problems so they could speed up their developments. In the end we managed to obtain a manual mapping of the application split into modules. Each modules data are sent one by one to the DSP and processed one after another. One of the first big success of the hArtes tool chain was that it provided a way to select manually which functions of the source code we wanted to be executed on the DSP by the means of a C language extension: the pragmas. The tool chain then processed both GPP and DSP source code in order to automatically take care of the Remote Procedure Call implemented on the DSP. The tool chain automatically replaced the function call on the GPP by a synchronization primitive that do send the data to the DSP, launch the appropriate function and wait for the result. On the DSP side, the processor waits for data and for procedure number to be able to work. Once that first integration phase was done, we took time to refine the algorithm in order to reach the real-time performance. This step required us to redesign some parts of the algorithm in order to reduce the required processing power by replacing some implementations in the algorithms by more efficient ones. The next integration work was to compare the manual mapping application against the automatically mapped one in order to validate that the hArmonic tool was able to find an appropriate repartition source code between GPP and DSP. At the end of the project, the hArmonic tool succeeded to reproduce the exact same mapping without having as an input any kind of information on how the application was architectured or designed. The only resolution of the dependencies between the various parts of the source code made the tool decide a coherent mapping really close to the one we did manually. This implementation allowed us to reach a really high performance, close to the one we obtain using manual mapping. The Ephraim and Mallah Stationary Noise Filter has been implemented using several techniques allowing the comparison of each of them in terms of implementation and performance.

The above graph shows the execution time of 3 versions of the application considering three successfull compilation techniques. The vertical axis is time expressed in sec and each graph represents the statistical repartition of the execution time of the filter. The vertical line shows the min/max values and the box delimits the 25%/75% quartiles. The Real-Time constraint is located at 16000sec so if the worst case execution times (the highest location of the vertical line) is below it, the application is fast enough. The first implementation we made was using no mapping at all. All the source code of the application was executed on the GPP. This solution was the most easy to obtain because the GPP do provide a complete Linux operating system, with a lot of libraries and debugging tools. It was a necessary step in our integration work because it allowed us to evaluate the processing power of the global architecture. The ARM GPP is not designed to process some floating point calculation and is required to emulate this kind of operations by the mean of a software library thus the resulting performance is really poor. This step allowed us to check that our implementation wasn't more than 10 times slower than the wanted performance so the speedup we obtain using the DSP has a chance to reach the real-time performance. The Fig. 5.22 shows that the GPP mapping of the latest version of the algorithm is 4 times slower than the real time. Another interesting point is the repartition of the execution times. The vertical bar delimits the maximum and the minimum execution time of the algorithm on the GPP. The Linux Operating system is clearly subject to interrupts and the in-

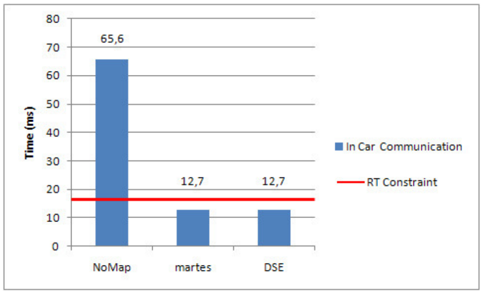


Figure 5.22: In-Car communication performances in terms of time elapseds

determinism of it is really high so the execution time fluctuates between 58ms and 88ms. The second implementation using manual mapping is the most efficient one. The average execution time is 12.7ms while the required execution time is 16ms. This final implementation is below the Real-Time performance and allowed us to obtain some nice demonstrators directly using the audio input/output of the hardware. One interesting benefits of this implementation is the repartition of the execution time. The DSP operating system is much more determinist than the GPP one and we obtain a much more reliable behaviour. The third implementation using the automatic mapping proves by its results that the automatic mapping is correct. The tools successfully chose the appropriate mapping and the resulting application is as performant as the manual mapped one. The average execution time is 12.7ms. The big advantage of hArtes toolchain is to provide an efficient way to explore the mapping that we can put in place on the hardware to make an application running and also to divide automatically the code between the processing elements. This new generation of compilers will probably be the basis of the tools that we will be looking for in order to build software for the heterogeneous hardware and even for the manycore architecures which are already on the market and they will probably reach the low power market very soon.

5.6 Conclusion

Starting from the development of a CIS Carputer-based test to test and validate the multichannel audio system and algorithms inside the cars, a development of the final ACIS proof-of-concept has been carried out. Starting from the CNUTS implementation of the algorithms, different approaches have been realized to test within the hArtes toolchain: a direct implementation to be used as the final target for the toolchain optimization tools; no-mapping approach considering the entire application running on ARM (GPP option), providing a working reference and allowing the evaluation of the basic performances for the application porting on DEB; a manual mapping (mArtes approach) where each function of the algorithms is manually divided among processors as a term of comparison for the toolchain partitioning tools; an automatic mapping/partitioning (DSE approach) to test the functionality of the entire toolchain (optimization and partitioning). Several tests have been done in order to evaluate the performance achievement considering different toolchain versions and different hardware platforms in terms of memory allocation and workload requirements. Good results were achieved with the toolchain comparing it with the reference examples. Considering the martes approach, it allows having the same results of the direct implementation introducing a good achievement in terms of development time required to realize an embedded application since a deep knowledge of the platform is not required. It is clear that there is a great advantage also considering the DSE approach comparing it with the direct implementation and No-mapping approach respectively; the automatic approach (DSE) is the first step of the implementation suggesting a mapping solution of the target application allowing users to very quickly override some decisions and get better performance. Some adjustments of the toolchain is certainly requested, moreover this approach represents a first step on the development of advanced compilers to easy generate complex solutions. As a matter of fact, the toolchain functionality allows to implement an application without a deep knowledge of the final embedded platform, reducing considerably the development time.

Bibliography

[1] S. Cecchi, A. Primavera, F. Piazza, F. Bettarelli, E. Ciavattini, R. Toppi, J.G.F. Coutinho, W. Luk, C. Pilato, F. Ferrandi, V. Sima and K. Bertels, The hArtes CarLab: A New Approach to Advanced Algorithms Development for Automotive Audio, Proc. of the 129th Audio Engineering Society Convention, Nov. 2010.

[2] A. Lattanzi, F. Bettarelli and S. Cecchi NU-Tech: the entry tool of the hArtes toolchain for algorithms design Proc. of the 124th Audio Engineering Society Convention, May 2008.

[3] NUTS Software Development Kit 2.0 (Rev 1.1) http://www.nu-tech-dsp.com

[4] F. Piazza, S. Cecchi, L. Palestini, P. Peretti, F. Bettarelli, A. Lattanzi, E. Moretti, E. Ciavattini, Demonstrating hArtes project approach through an Advanced Car Information System, ISVCS Int. Symposium on Vehicular Computing Systems - Jul. 22-24, 2008 - Trinity College Dublin, Ireland.

[5] S. Cecchi, L. Palestini, P. Peretti, E. Moretti, F. Piazza, A. Lattanzi, F. Bettarelli, Advanced Audio Algorithms for a Real Automotive Digital Audio System, Proc. of the 125th Audio Engineering Society Convention, Oct. 2008.

[6] J. Kontro, A. Koski, J. Sjoberg and M.Vaananen, Digital Car Audio System, IEEE Transactions on Consumer Electronics, Vol.39,No3, Aug. 1993.

[7] L. Palestini, P. Peretti, S. Cecchi, F. Piazza, A. Lattanzi, and F. Bettarelli, Linear Phase Mixed FIR/IIR Crossover Networks: Design and Real-Time Implementation, Proc. of the 123th Audio Engineering Society Convention, Oct. 2007.

[8] J. S. Orfanidis, High-order digital parametric equalizer design, J. Audio Eng. Society, vol. 53, no. 11, Nov.2005.

[9] H. Zhao and J. Yu, A simple and efficient design of variable fractional delay FIR filters, IEEE Transaction on Circuits and Systems II: express briefs, vol. 53, no. 2, Feb. 2006.

[10] M. Lang and T. Laakso, Simple and robust method for the design of allpass filters using least-squares phase error criterion, IEEE Transaction on Circuits and Systems II: Analog and Digital Signal Processing, vol. 41, no. 1, pp. 40-48, Jan. 1994.

[11] H. Schopp and H. Hetzel, A linear phase 512 band graphic equalizer using the fast fourier transform, Proc. of the 96th Audio Engineering Society Convention, Jan. 1994.

[12] A. J. S. Ferreira and A. Leite, An Improved Adaptive Room Equalization in the Frequency Domain, Proc. of the 118th Audio Engineering Society Convention, May 2005.

[13] J-S. Soo, K.K. Pang, Multi-Delay Block Frequency Domain Adaptive Filter, IEEE Transaction on Acoustics, Speech and Signal Processing, Feb 1990.

[14] E.Moulines, O.A. Amrane, Y. Grenier, The Generalised Multi-Delay Adaptive Filter: Structure and Convergence Analysis, IEEE Transaction on Signal Processing, Vol. 43, pp 14-28, January 1995.

[15] M.M.Sondhi,D.R. Morgan, J.L. Hall, Stereophonic Acoustic Echo Cancellation - An Overview of the Fundamental Problem, IEEE Signal Processing Letters, 1995.

[16] H.Buchner, J.Benesty,W. Kellermann, Generalised Multi-channel Frequency-Domain Adaptive Filtering: Efficient Realization and Application to Hands-Free Speech Communication, Signal Processing, Vol. 85, N3, pp 549-570, March 2005.

[17] Y. Ephraim, D. Malah, Speech Enhancement using a Minimum Mean-Square Error Short-Time Spectral Amplitude Estimator, IEEE Transaction on Acoustics, Speech and Audio Signal Processing, Vol. 32, N6, pp 1109-1121, 1984.

[18] Y.Ephraim, D. Malah, Speech Enhancement using a Minimum Mean-Square Error Log-Spectral Amplitude Estimator, IEEE Transaction on Acoustics, Speech and Audio Signal Processing, Vol. 33, N2, pp 443-445, 1985.

[19] S. Rangachari, A Noise Estimation Algorithm for Highly Non-Stationary Environments, In Speech Communications, Vol. 48, pp. 220-231, 2006.

[20] J. H. DiBiase, A High Accuracy, Low Latency Technique for Talker Localization in Reverberant Environments Using Microphone Arrays, PhD Thesis, Brown University, Rhode Island, May 2000.

[21] R. Zelinski, A Microphone Array with Adaptive Post-Filtering for Noise Reduction in Reverberant Rooms, Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 2578-2581, April 1988.

[22] J. Meyer, K.U. Simmer, Multi-Channel Speech Enhancement in a Car Environment using Wiener Filtering and Spectral Subtraction, Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 21-24, 1997.

[23] C. Marro, Y. Mahieux, K.U. Simmer, Analysis of Noise Reduction and Dereverberation Techniques based on Microphone Arrays with Post-Filtering, IEEE Transaction on Speech and Audio Processing, Vol. 6, N3, pp 240-259, May 1998.

[24] J. Bitzer, K.U. Simmer, K.D. Kammeyer, Multi-Microphone Noise Reduction by Post-Filter and Super-directive Beam-former, Proceedings of International Workshop on Acoustic, Echo and Noise Control, pp. 100-103, 1999.

[25] I. A. McCowan, H. Bourlard, Microphone Array Post-Filter based on Noise Field Coherence, IEEE Transaction on Speech and Audio Processing, Vol. 11, pp 709-716, 2003.

[26] F. Jelinek, Continuous speech recognition by statistical methods, Proceedings of the IEEE, Vol. 64-4, pp. 532-556, 1976

[27] LIASTOK software webpage,http://lia.univ-avignon.fr/fileadmin/documents/Users/Intranet/chercheurs/linares/index.html

[28] C. Levy, G. Linares, P. Nocera, J.F. Bonastre, Embedded mobile phone digit-recognition, in Chapter 7 of Advances for In-Vehicle and Mobile Systems, 2007

[29] A. Viterbi, Error bounds for convolutional codes and an asymptotically optimum decoding algorithm, IEEE Transaction on Information Theory, Vol. 2-13, pp. 260-269, 1967

[30] ALIZE software, http://mistral.univ-avignon.fr/

[31] J.F. Bonastre, P. Morin, J.C. Junqua, Gaussian dynamic warping (GDW) method applied to text-dependent speaker detection and verification, Eurospeech, pp.2013-2016, 2003

[32] A. Larcher, J.F. Bonastre, J.S.D. Mason, Reinforced Temporal Structure Information For Embedded Utterance-Based, Interspeech, 2008

[33] C. Levy, Modles acoustiques compacts pour les systmes embarqus, Phd Thesis, 2006

[34] NIST-SRE website, http://www.nist.gov/speech/tests/sre/2008/index.html

[35] N. House, Aspects of the Vehicle Listening Environment, Proc. of the 87th Audio Engineering Society Convention, Oct 1989.

[36] R. Shivley, Automotive Audio Design (A Tutorial), Proc. of the 109th Audio Engineering Society Convention, Sep 2000.

[37] J. S. Jarmo Kontro, Ari Koski and M. Vaananen, Digital Car Audio System, IEEE Transactions on Consumer Electronics, vol. 39, no. 3, Aug. 1993.

[38] A. Farina and E. Ugolotti, Spatial Equalization of Sound Systems in Cars, Proc. of 15th AES Conference Audio, Acoustics & Small Spaces, Oct. 1998.

[39] B. Crockett, M. Smithers, and E. Benjamin, Next Generation Automotive Research and Technologies, Proc. of the 120th Audio Engineering Society Convention, May 2006.

[40] M. Smithers, Improved Stereo Imaging in Automobiles, Proc. of the 123rd Audio Engineering Society Convention, Oct. 2007.

[41] S. P. Lipshitz and J. Vanderkooy, A family of linear-phase crossover networks of high slope derived by time delay, J. Audio Eng. Society, vol. 31, no. 1/2, pp. 2/20, Feb. 1983.

[42] P. Regalia and S. Mitra, class of magnitude complementarity loudspeaker crossovers, IEEE Transaction on Acoustic, Speech, Signal Processing, vol. 35, pp. 1509/1516, Nov. 1987.

[43] B. Widrows, J.MC.Cool, M. Ball, The Complex LMS Algorithm, Proceedings of the IEEE, Vol.63-4, Apr. 1975.

[44] F. Capman, J. Boudy, P. Lockwood, Acoustic Echo Cancellation and Noise Reduction in the Frequency-Domain: A Global Optimization, Proceedings of European Signal Processing Conference (EUSIPCO), 1996.

[45] M. Dentino, J. McCool, B. Widrow, Adaptive Filtering in Frequency Domain, Proceedings of the IEEE, Vol. 66-12, December 1978.

[46] E.R. Ferrara, Fast Implementation of LMS Adaptive Filters, IEEE Transactions on Acoustics, Speech and Signal Processing, Aug. 1980.

[47] J. Lariviere, R. Goubran, GMDF for Noise Reduction and Echo Cancellation, IEEE Signal Processing Letters, Vol. 7, Issue 8, pp. 230-232, Aug. 2000.

[48] J. Lariviere, R. Goubran, Noise-reduced GMDF for Acoustic Echo Cancellation and Speech Recognition in Mobile Environments, in Vehicular Technology Conference, Vol. 6, pp. 2969-2972, 2000.

[49] D. Mansour, A.H. Gray, Unconstrained Frequency-Domain Adaptive Filter, IEEE Transactions on Acoustics, Speech and Signal Processing, Oct. 1982.

[50] E. Moulines, O.A. Amrane, Y. Grenier, The Generalised Multi-Delay Adaptive Filter: Structure and Convergence Analysis, IEEE Transactions on Signal Processing, Vol. 43, pp. 14-28, Jan. 1995.

[51] P.C.W. Sommen, Frequency-Domain Adaptive Filter with Efficient Window Function, Proceedings of ICC-86, Toronto, 1986.

[52] M.M. Sondhi, D.R. Morgan, J.L. Hall, Stereophonic Acoustic Echo Cancellation: An Overview of the Fundamental Problem, IEEE Signal Processing Letters, 1995.

[53] J-S. Soo, K.K. Pang, Multidelay Block Frequency Domain Adaptive Filter, IEEE Transactions on Acoustics, Speech and Signal Processing, Feb. 1990.

[54] F. Amand, J. Benesty, A. Gilloire, Y. Grenier, Multichannel Acoustic Echo Cancellation, Proceedings of International Workshop on Acoustic, Echo and Noise Control, Jun. 1993.

[55] O.A. Amrane, E. Moulines, Y. Grenier, Structure and Convergence Analysis of the Generalised Multi-Delay Adaptive Filter, Proceedings of EUSIPCO, August 1992.

[56] J. Benesty, F. Amand, A. Gilloire, Y. Grenier, Adaptive Filtering Algorithms for Stereophonic Acoustic Echo Cancellation, Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, 1995.

[57] F. Berthault, C. Glorion, F. Capman, J. Boudy, P. Lockwood, Stereophonic Acoustic Echo Cancellation and Application to Speech Recognition: some experimental results, Proceedings of International Workshop on Acoustic, Echo and Noise Control, 1997.

[58] J. Boudy, F. Capman, P. Lockwood, A Globally Optimised Frequency-Domain Acoustic Echo Canceller for Adverse Environment Applications, Proceedings of International Workshop on Acoustic, Echo and Noise Control, 1995

[59] H. Buchner, J. Benesty, W. Kellermann, An Extended Multidelay Filter: Fast Low-Delay Algorithms for Very High-Order Adaptive Systems, Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 5, pp 385-388, April 2003.

[60] H. Buchner, J. Benesty, W. Kellermann, Generalised Multi-channel Frequency-Domain Adaptive Filtering: Efficient Realization and Application to Hands-Free Speech Communication, in Signal Processing, Vol. 85, N.3, pp. 549-570, Mar. 2005.

[61] T. Haulick, Signal Processing and Performance Evaluation for In-Car Cabin Communication Systems, ITU-T Workshop on Standardization in Telecommunication for Motor Vehicles, Nov. 2003.

[62] T. Haulick, Speech Enhancement Methods for Car Applications, The Fully Networked Car, A Workshop on ICT in Vehicles, ITU-T Geneva, Mar. 2005.

[63] T. Haulick, Systems for Improvement of the Communication in Passenger Compartment, ETSI Workshop on Speech and Noise in Wideband Communication, Sophia Antipolis, May 22-23, 2007.

[64] E. Lleida, E. Masgrau, A. Ortega, Acoustic Echo Control and Noise Reduction for Cabin Car Communication, Proceedings of Eurospeech, Vol. 3, pp. 1585-1588, Sep.2001.

[65] K. Linhard, J. Freudenberger, Passenger In-Car Communication Enhancement, Proceedings of European Signal Processing Conference (EUSIPCO), 2004.

[66] A. Ortega, E. Lleida, E. Masgrau, DSP to Improve Oral Communications Inside Vehicles, Proceedings of European Signal Processing Conference (EUSIPCO), 2002.

[67] A. Ortega, E. Lleida, E. Masgrau, F. Gallego, Cabin Car Communication System to Improve Communications Inside a Car, Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 4, pp. 3836-3839, May 2002.

[68] A. Ortega, E. Lleida, E. Masgrau, Residual Echo Power Estimation for Speech Reinforcement Systems in Vehicles, Proceedings of Eurospeech, Sep. 2003.

[69] A. Ortega, E. Lleida, E. Masgrau, L. Buera, A. Miguel, Acoustic Feedback Cancellation in Speech Reinforcement Systems for Vehicles, Proceedings of Interspeech, 2005.

[70] A. Ortega, E. Lleida, E. Masgrau, Speech Reinforcement System for Car Cabin Communications, IEEE Transactions on Speech and Audio Processing, Vol. 13, N.5, pp. 917-929, Sep. 2005.

[71] A. Ortega, E. Lleida, E. Masgrau, L. Buera, A. Miguel, Stability Control in a Two-Channel Speech Reinforcement System for Vehicles, Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, 2006.