EURASIP Journal on Applied Signal Processing 2005:18, 3060–3068

SPAIDE: A Real-time Research Platform for the Clarion CII/90K Cochlear Implant

L. Van Immerseel

Medical Electronics Lab, University of Antwerp, Groenenborgerlaan 171, 2020 Antwerp, Belgium

Advanced Bionics European Research Lab, Groenenborgerlaan 171, 2020 Antwerp, Belgium
Email: luc.vanimmerseel@ua.ac.be

S. Peeters

Medical Electronics Lab, University of Antwerp, Groenenborgerlaan 171, 2020 Antwerp, Belgium

Advanced Bionics European Research Lab, Groenenborgerlaan 171, 2020 Antwerp, Belgium
Email: stefaan.peeters@ua.ac.be

P. Dykmans

Advanced Bionics European Research Lab, Groenenborgerlaan 171, 2020 Antwerp, Belgium
Email: dykmans@yucom.be

F. Vanpoucke

Advanced Bionics European Research Lab, Groenenborgerlaan 171, 2020 Antwerp, Belgium
Email: filiep.vanpoucke@advancedbionics.com

P. Bracke

Advanced Bionics European Research Lab, Groenenborgerlaan 171, 2020 Antwerp, Belgium
Email: peter.bracke@advancedbionics.com

Received 30 April 2004; Revised 19 November 2004

SPAIDE (*sound-processing algorithm integrated development environment*) is a real-time platform of Advanced Bionics Corporation (Sylmar, Calif, USA) to facilitate advanced research on sound-processing and electrical-stimulation strategies with the Clarion CII and 90K implants. The platform is meant for testing in the laboratory. SPAIDE is conceptually based on a clear separation of the sound-processing and stimulation strategies, and, in specific, on the distinction between sound-processing and stimulation channels and electrode contacts. The development environment has a user-friendly interface to specify sound-processing and stimulation strategies, and includes the possibility to simulate the electrical stimulation. SPAIDE allows for real-time sound capturing from file or audio input on PC, sound processing and application of the stimulation strategy, and streaming the results to the implant. The platform is able to cover a broad range of research applications; from noise reduction and mimicking of normal hearing, over complex (simultaneous) stimulation strategies, to psychophysics. The hardware setup consists of a personal computer, an interface board, and a speech processor. The software is both expandable and to a great extent reusable in other applications.

Keywords and phrases: research platform, sound processing, cochlear implant.

1. INTRODUCTION

The technical evolution in cochlear implant processing shows an ever-increasing complexity of both the hardware and software [1, 2]. This technological advance increases performance scores significantly but makes it difficult to implement and experiment with new sound-processing and stimulation strategies. Therefore, research tools that hide most of the complexity of implant hardware and communication protocols have been developed recently [3, 4, 5]. They allow streaming off-line processed data from PC to implant and support all stimulation features of the implant. However, off-line processing cannot support live input from a microphone. Furthermore, it is cumbersome when an experiment consists of comparing different processing strategies each with different parameter settings, more so when large

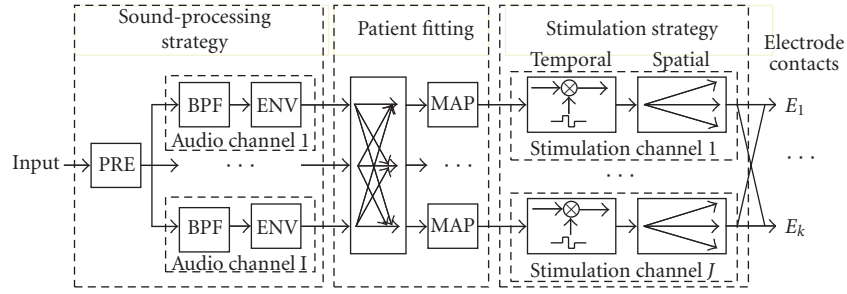

Figure 1: Relation between (typical) strategies, channels, and electrode contacts. PRE = pre-emphasis, BPF = bandpass filter, ENV = envelope extraction, MAP = mapping to current values.

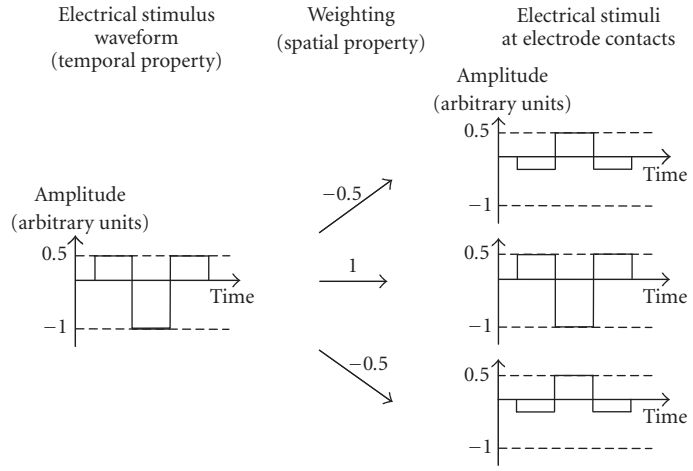

Figure 2: Illustration of temporal and spatial properties of a stimulation channel. A triphasic waveform (temporal property) is weighted (spatial property) and distributed over three electrode contacts to form a stimulation channel.

word and sentence databases are used for evaluating sound-processing or stimulation strategies.

SPAIDE (*sound-processing algorithm integrated development environment*) is a platform that makes all features of the Clarion CII and 90K cochlear implants [8] available for advanced research. It supports streaming off-line processed data and real-time processing on PC combined with streaming of the results to the implant. The platform supports live input and off-line processing of the test material for all possible test conditions is not necessary anymore.

The following section describes the basic concepts of SPAIDE and the terminology used throughout the paper. Then follow the key elements of the hardware and software. Next the specifications and benchmark results of the real-time processing are presented, and typical research applications with SPAIDE and the steps to set up an experiment are described. Finally the pros and cons of the platform and future developments are discussed.

2. BASIC CONCEPTS

The architecture and implementation of SPAIDE rely on the concept of channels. The platform makes a clear distinction between audio and stimulation channels, stimulation groups, and electrode contacts (Figure 1). The sound-processing strategy defines the number of audio channels and the different processing steps in each of these channels. A typical processing strategy in cochlear implants consists of pre-emphasis filtering, bandpass filtering, and envelope extraction [2]. The stimulation strategy specifies the number of stimulation channels, their stimulation sequence, and their temporal and spatial definitions. A stimulation channel is defined as a set of electrode contacts that simultaneously carry the same electrical stimulus waveform, though not necessarily with the same amplitude and sign (Figure 2). This general definition is possible because the CII/90K implant has 16 identical and independent current sources, one per electrode contact. The temporal definition of a channel describes the electrical stimulus waveform. The spatial definition of a channel specifies the weights with which the waveform is multiplied for the different electrode contacts in the channel (Figure 2). Some or all of the channels can be used more than once within a strategy, and an electrode contact can be part of different stimulation channels. All channels stimulated simultaneously constitute a stimulation group [6]. The waveforms of the different stimulation channels within a group may be different. Furthermore, one or more channels can be part of different stimulation groups. For instance, if channels

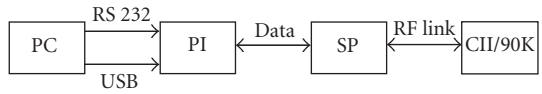

Figure 3: Hardware components of SPAIDE: personal computer (PC), programming interface (PI), speech processor (SP), and Clarion CII or 90K implant (CII/90K).

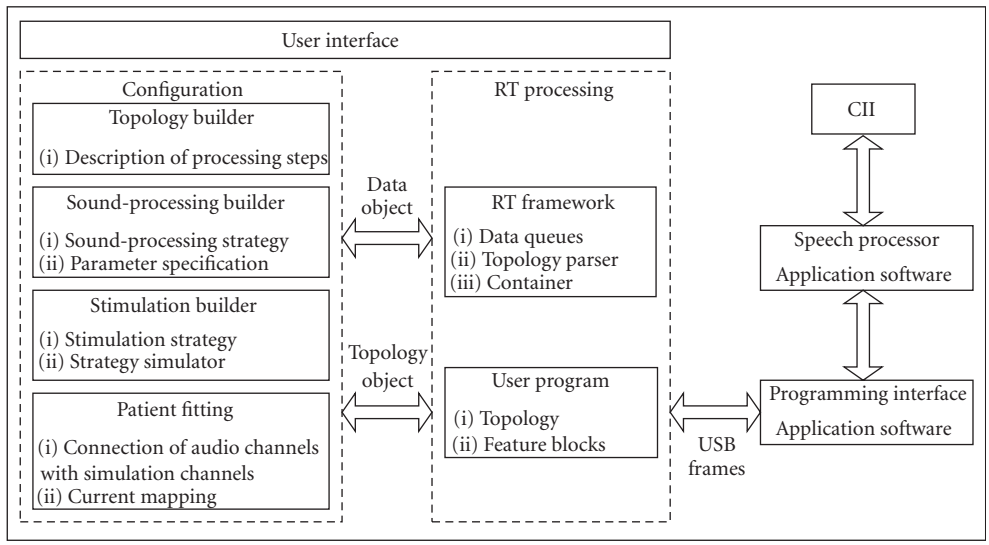

Figure 4: Software architecture of SPAIDE.

C1 and C2 form group G1 and channels C2 and C3 form group G2, then channel C2 is stimulated whenever group G1 or G2 is activated while channels C1 and C3 are stimulated only when group G1 or G2 is activated, respectively.

The sound-processing and stimulation strategies are specified independently of each other. Audio channels are connected to stimulation channels during patient fitting. Each stimulation-channel input is connected to one of the audio-channel outputs, mapped to current values by the stimulation-channel and patient-specific compression, and then multiplied with the stimulus waveform and the spatial weights to determine the current at the electrode contacts.

The stimulation strategy is independent of the input signal and sound processing. This is not a limitation imposed by the platform but due to the fact that the stimulation strategy is programmed in the CII/90K implant; it is programmed with a table that defines the shape and timing of the electrical waveforms generated by the current sources. As a consequence stimulation rates in the different channels are fixed and cannot be changed based on signal properties, for example, set to $1/F0$ with $F0$ the fundamental frequency.

3. HARDWARE

The hardware of SPAIDE consists of a personal computer (PC), a programming interface (PI), a speech processor (SP), and a Clarion CII or 90K implant (CII/90K) (Figure 3). Because the clinical programming interface (CPI), which is used during implant fitting in the clinical centre, does not provide a USB connection a SBC67 DSP board of Innovative Integration [7] is used. The only SP that currently supports the SPAIDE application is the portable speech processor (PSP) [8].

At boot time the PC downloads the application software to the PI through the RS232 link, the PI sends application software to the SP, and the SP configures the implant's registers. Once all hardware components are booted, the PC captures sound from file or audio input on PC, processes the signal in a custom way, and sends commands and data in packages through USB to the PI where data is buffered and commands are handled immediately. During stimulation the SP masters the timing by sending hardware interrupts to the PI whenever it needs to forward data to the implant. The PI thus sends the buffered data to the SP at the rhythm imposed by these hardware interrupts, and the SP transmits the data to the CII/90K. Finally, the CII/90K generates the electrical stimulation patterns. The SP continuously reads the implant status information and the PI monitors both the SP and implant status. Stimulation is stopped immediately on any error condition.

4. SOFTWARE

4.1. Overview

The software architecture of SPAIDE is shown in Figure 4. On the PC side the application consists of three major components. The user interface (UI) of SPAIDE allows for user interaction with the two other components, which are the

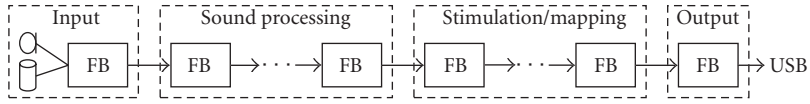

Figure 5: Typical processing chain of SPAIDE. The processing functions are implemented in feature blocks (FB).

configuration and the real-time (RT) processing. PC software runs on Windows XP platforms. Specific application software also runs in the programming interface and in the speech processor.

In order to run an experiment the different processing steps and their parameters must be configured in a so-called topology file. This description specifies the sequence and the parameters of all processing steps that are needed to implement the sound-processing and stimulation strategies. Different (graphical) user interfaces help to specify the experiment and fit the patient.

The processing chain consists of up to four larger components (Figure 5). The first component is the sound input, which reads data from file (e.g., WAVE audio) or captures sound on PC from its microphone or line input. The second component is the sound processing that can be completely user-defined within the processing capabilities of the PC. The third component is the application of stimulation strategy and patient fitting, and is custom within the limitations of the CII/90K. The fourth component is the output, which can be the USB driver for streaming data to the implant, sound output via loudspeaker or line output on PC, or output to file or MATLAB. Processing functions are implemented in feature blocks each of which implements one processing step, for example, a filter bank. Together, these feature blocks constitute an extendable collection of processing functions available to build a topology with.

RT processing is implemented in an RT framework. This framework requires that feature blocks implement a set of functions for initialization and processing, which is fulfilled automatically when the feature block is derived from the feature block class. Most software modules, from feature blocks to the stimulation-strategy builder used during configuration, are available as a Win32 dynamic link library (DLL) or as a static library. Therefore they can be reused in any application that can deal with these DLLs and libraries.

4.2. Configuration

The first step in the configuration (Figure 4) is the specification of the topology. The main task of the topology builder is to define the different processing steps (feature blocks), and to specify the names of the queues that interconnect them. The framework will automatically connect two feature blocks with corresponding input and output queue names. Once the topology is specified the parameters of the sound-processing strategy, for example, filter parameters, must be designed. These parameters can be in the topology file or the topology can contain links to the files with the parameters. Currently no UI is integrated in SPAIDE to specify the topology or to design sound-processing parameters. Other applications, for instance, MATLAB, should be used instead.

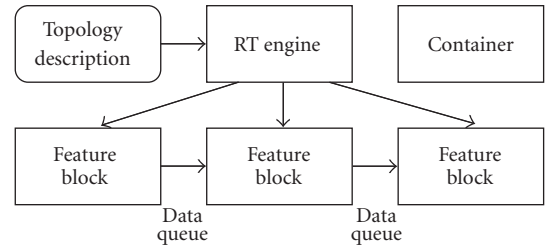

Figure 6: Example of a typical RT framework structure.

The stimulation builder window of SPAIDE specifies the temporal and spatial properties of stimulation channels and stimulation groups, and also specifies the grounding scheme. When one or both of the indifferent electrodes outside the cochlea the implant box or the ring electrode around the electrode array, are grounded this applies to all stimulation channels. The grounding of an electrode contact however is controlled dynamically, that is, the electrode contact can be grounded in one or more stimulation channels but can be an active contact in other stimulation channels. The specified stimulation strategy is converted into two tables. One table is used on the PC for timing the data/amplitude stream from PC to implant, the other is sent to the implant to control the shape and timing of the electrical waveforms generated by the current sources.

Patient fitting consists of the specification of patient-dependent parameters. It groups parameters that can be adapted to the patient. The most important parameters are the connection between audio and stimulation channels and the mapping functions in each of the stimulation channels. The mapping functions implemented in SPAIDE define the relation between the processed-audio amplitude and the current value (in μA) in each of the stimulation channels individually and are implemented as static compressions.

4.3. Real-time processing

The RT framework consists of several components (Figure 6) of which the core component is the RT engine that initializes and runs the topology. First the topology description is read and the feature blocks (processing functions) are connected through data queues. These queues have no specialized data type and are implemented as a byte buffer. Functions that use a queue are assumed to know the type of data their input queue is providing. The RT engine also creates a container object that is used to store data that is accessible by both the SPAIDE application and all components in the framework. After creating all components, the RT engine

initializes the processing with the parameters specified in the topology and sends the stimulation-strategy table to the implant. Once the whole processing chain and the implant are initialized, the engine starts its run-thread and sequentially executes the processing functions in the topology and realizes a continuous data flow from input to output. If needed a feature block can run in its own thread, which is useful for asynchronous processes like the USB transmission.

SPAIDE uses a frame-based paradigm to process the audio input in real time. The audio is first chopped in frames of approximately 50–100 milliseconds (see Section 5). These frames are processed through digital filters, mapping functions, and so forth, and samples are selected as specified by the stimulation strategy. These values are converted into stimulation currents according to the patient-dependent fitting parameters. To maximize the stimulation accuracy, SPAIDE automatically sets the current ranges in the implant to obtain the highest current resolution, and scales current values accordingly. Finally, the currents are organized into frame packets and transmitted over the USB link to the programming interface. During processing messages sent by SPAIDE, the RT framework, or feature blocks are logged in a window such that the experimenter is aware of the status of the platform. Changes to the fitting parameters are immediately used during RT processing as long as these changes do not require a reset of the implant. This allows for RT fitting with SPAIDE.

SPAIDE has a simulation mode in which there is no data stream to the implant. A simulator window displays the electrical waveforms, either at the stimulation-channel level or the electrode-contact level. This allows for verifying the whole configuration without the need for hardware.

4.4. Programming interface

The role of the application software in the PI is to handle the data stream from the USB link to the SP and the implant in a timely manner. It implements a FIFO in which data is written at the rhythm of the USB transmission, and from which data is read at the rhythm imposed by the hardware interrupts that are generated by the SP.

This FIFO buffer is necessary because processing on a PC running Windows shows jitter in the processing duration. This makes that temporarily no new data is sent by the PC to the PI. Without a buffer this would cause an underrun, that is, the PI has not enough data available to sustain the continuous stream to the implant. The length of the buffer is dimensioned such that the probability of an underrun is very low (see Section 5). If it occurs anyhow, then a frame holding zero amplitudes is inserted. In the case of overrun, that is, too much data is sent to the PI, the application on the PI can throw out frames. An overrun typically follows a series of underruns when the processing on PC is catching up its processing delay. Neither the insertion of zero amplitudes during underrun nor the deletion of frames during overrun can result in charge-unbalanced stimulation, thus guaranteeing patient safety.

Table 1: Average percentage use of time for CIS and HiResolution processing. Processing a 100-millisecond sound frame takes 20.4 milliseconds in case of CIS and 31.3 milliseconds in case of HiResolution.

CIS	HiResolution
20.4%	31.3%

4.5. Sound processor

The SP application software is a subset of the code base of the clinical SP, and includes only the functionality needed for forward telemetry to the implant. Other capabilities like audio capture from the SP, access to SP control settings, and back telemetry are currently not used but might be in the future.

5. RESULTS

Differences in speech perception scores of the HiResolution [8] strategy with SPAIDE and with the clinical processor have not been evaluated in a formal study. However, initial testing of the platform demonstrated a tendency of slightly lower scores with SPAIDE. This is probably a result of the accumulation of small implementation differences between SPAIDE and the clinical device. The sound processing in SPAIDE has to simulate the analogue front-end of the clinical processor and does not include the AGC. Furthermore, the mapping/compression in SPAIDE is similar but not identical to the one in the clinical device. Finally, due to some limitations in the current stimulation builder of SPAIDE (see Section 6), the stimulation strategy is not always identical to the clinically used strategy.

Table 1 shows benchmark results for the standard cochlear implant strategies used with the CII/90K, CIS [2], and HiResolution, as measured on a Pentium IV-1.7 GHz PC, with 512 MB RAM, and running Windows XP. Audio input is read in frames of 100 milliseconds from a WAV file with signals sampled at 44 100 Hz, and the processing uses double-precision floating-point values. The CIS processing chain consists of a 2nd-order IIR pre-emphasis filter, a 16-channel 6th-order IIR filter bank, half-wave rectification, 2nd-order lowpass IIR filters for envelope extraction, sample selection, compression, and USB transmission. The HiResolution processing chain consists of a full simulation of both the analogue and digital preprocessing stages in the SP programmed with the HiResolution strategy (without the AGC), a 16-channel 6th-order IIR filter bank, envelope extraction by rectification and averaging, compression, and USB transmission. In both cases the stimulation strategy is standard 16-channel CIS with a rate of 2900 pulses/s/channel. The longer processing time needed by the HiResolution strategy is due to the simulation of the analogue front-end. These timing results are only indicative because they also depend on PC hardware properties like bus speed, amount of cache memory, and so forth, but they show there is enough headroom for implementing more complex processing strategies.

An important aspect of stimulation in cochlear implants is that the timing of the electrical pulses is exact and that

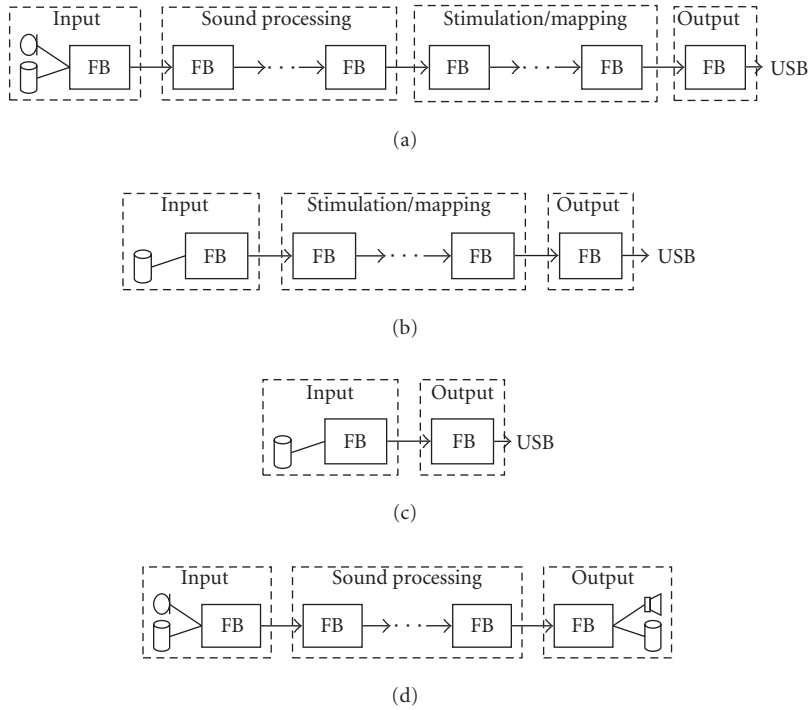

Figure 7: Different configurations of SPAIDE (FB denotes feature block).

the correct current values are delivered to the right electrode contacts. This asks for synchronization between the processing on PC and the stimulation in the CII/90K. All stimulation timing is controlled by the hardware interrupts generated by the SP with microsecond accuracy. Therefore it is independent of the exact timing of the software running under Microsoft Windows provided the PC is able to sustain the required data stream. As described before, the stimulation builder creates a timing table during configuration, which allows the PC to match the rate of data transmission from PC to PI to the rate at which the implant needs data. This minimizes the chance of data underrun provided the SPAIDE application can use PC resources as much as it needs. Unfortunately this is not always the case under Windows. An analysis has shown that Windows periodically calls processes that delay the processing, and thus the data transmission over USB, up to several milliseconds. Also some rare but much larger delays of several tens of milliseconds are found. Although it is possible to specify short processing frames and use the FIFO in the PI to buffer these processing delays, SPAIDE is typically used with processing frames of 50–100 milliseconds. That delay is needed to minimize or prevent underruns while using longer processing frames has also the advantage of a smaller processing overhead, that is, more efficient use of the PC's resources. The length of the FIFO in the PI is approximately 300 milliseconds. It is a value that gives extra headroom to buffer even larger processing delays due to UI interaction, for example, in the fitting window of SPAIDE, or due to other applications running at the same time as SPAIDE.

The overall latency from sound-processing input to electrical stimulation depends on the type of input. In case of file input and 100-milliseconds frames it is approximately 350 milliseconds, which is the sum of the processing duration (cf. Table 1), the USB transmission time, and the FIFO delay on the PI; in case of audio input this latency must be increased by the time to record the frame, that is, 100 milliseconds in this example.

Many safety measures are built in the platform to detect inconsistencies in the configuration and to prevent stimulation of the patient with too large or unbalanced currents. When a problem is encountered during configuration, the platform will not allow stimulation. Errors during processing will immediately result in a stimulation halt. The strategy builder always controls current balancing for each stimulation channel. The sum of currents should always be zero if no grounded contact is associated with the channel. If this condition is not met, the builder signals an error and refuses to build the strategy and extract the stimulation table for the implant.

As discussed before, the processing chain consists of up to four components (Figures 5 and 7a). This configuration is typical for research applications in the field of sound-processing or stimulation strategies that are not too complex such that real-time processing is feasible. It is the easiest configuration to evaluate strategies with sound or speech material stored as WAV files on hard disc. However, not all components are necessarily present in the configuration (Figures 7b, 7c, and 7d). Figure 7b shows a configuration where the input consists of audio-channel data that was generated and

saved to file earlier, for example, by SPAIDE (cf. Figure 7d) or by another application like MATLAB. The audio-channel data is connected to stimulation channels that are configured in SPAIDE, and the fitting parameters complete the configuration. This application is useful when the audio-processing complexity is too large for implementation in a real-time processing system on PC. Adaptive filter models of the cochlea used for a closer replication of the processing steps in the normal ear [9], for instance, are very likely to fall in this category. The setup is also applicable to experiments where audio data must meet specific long-term criteria which are difficult to guarantee in a real-time system, for example, noise addition to enhance stochastic resonance [10, 11]. Another category of applications that can use this configuration is found in studies of new stimulation strategies, where the input is used to modulate the electrical stimulation patterns and to generate special pulse patterns that allow evaluating spatial selectivity, temporal interaction, and so forh. The configuration in Figure 7c only consists of an input and an output component. It is a streaming application where the preprocessed data contains the amplitude values that must be transmitted to the implant. This is the preferred configuration when the experimenter wants maximum control over the stimulated currents, which is often the case in psychophysics. Finally, the configuration in Figure 7d is a pure sound-processing application where no data is streamed to the implant. This can be used to preprocess sound, for example, to study the application of simultaneous masking [12] in a free-field experiment with CI subjects.

The key to a successful experiment with SPAIDE is the preparation of topologies and sound-processing parameters, and the definition of stimulation strategies. To simplify this process for many experiments, SPAIDE comes with a set of topologies, parameters, and stimulation strategies that cover the CIS and HiResolution strategies for both streaming (Figures 7b and 7c) and real-time applications (Figures 7a and 7d). No further technical knowledge is needed to do experiments with SPAIDE. After preparation of the hardware setup and booting, the prepared topologies and stimulation strategies can be loaded. If necessary, the stimulation strategy can be fitted using the fitting screen of SPAIDE.

The software architecture, the separation of sound-processing and stimulation strategies, the implementation of processing steps in different modules (DLL, static lib), and the implementation of the RT framework as a separate module make it possible to reuse SPAIDE, or part of it, within a new Windows application that can deal with DLLs. Documentation and sample code are provided to support the development of new applications. The modularity also favours the expandability of SPAIDE through the addition of new feature blocks. A researcher can implement a custom processing function, for example, a specific filter bank or compression, as a new feature block which must be coded in C/C++.

SPAIDE is used in research on noise reduction and mimicking of normal hearing [12], and to evaluate new stimulation strategies [13]. Different research centres recently developed new C/C++, MATLAB, and Delphi applications that reuse SPAIDE functionality for psychophysical experiments and evaluation of new sound-processing and stimulation strategies.

6. DISCUSSION

In contrast with platforms that only can stream preprocessed data from file [3, 4, 5], SPAIDE is also able to simultaneously capture sound in real time, process this input immediately, and stream the results to the implant. Therefore SPAIDE is called a real-time system although it is not a hard real-time system as one would expect from a typical DSP platform. The reason is that all processing is done on a Windows platform that is not under full control of the application. This results in jitter in the processing duration, something that is accounted for in the application software of SPAIDE at the cost of an overall latency of 300–400 milliseconds. In many research applications this latency is much less disturbing than stimulation with zero currents due to an underrun. However, for applications that need live audio input from microphone this latency is too large when synchronization between visual cues for lip-reading and auditory perception is needed. Synchronization is mandatory when audio captured by the speech processor is used as a way to communicate with the patient. This mode is currently not available because only the USB downlink from PC to PI is used.

The platform is designed for use in a very broad range of research applications on sound-processing, stimulation strategies, and psychophysics. The platform however does not offer a ready-made solution for all possible research demands. Extending the possibilities with new processing functions (feature blocks) is one way to adapt the platform, but necessitates C/C++ programming knowledge. Another way is to use the exported functionality in an existing or new Windows application. This, of course, also demands programming skills but the application can be written in the preferred language like C/C++, C#, Visual Basic, MATLAB, and so forth.

Before experimenting with new sound-processing or stimulation strategies in patients, it is often required to first verify the whole processing chain from input to output. SPAIDE supports writing the data from queues, which interconnect the processing blocks, to files or to matrices in MATLAB. When SPAIDE is used in simulation mode, the current values and electrical waveforms can be verified. This allows verifying the processing up to the amplitude data transmitted to the programming interface and the temporal property of the stimulation channels, but not the currents delivered to the electrode contacts. These currents can only be monitored on an oscilloscope connected to the electrode load board of a reference implant in a box. A tool to analyse the RF signal to the implant [14] is no perfect alternative since the stimulation strategy, which delivers the amplitude data to the right electrodes in the cochlea, is programmed in the implant.

The current implementation of the stimulation builder supports up to 32 stimulation channels, each with its own temporal and spatial properties. The electrical waveform limited to a concatenation of 4 pulses and the duration of each of these pulses can be specified with a time resolution of 10.776

microseconds. The spatial scaling factor can only be specified in steps of 1/8, from −1 to +1. If the spatial scaling factor equals zero, the contact can be specified as active grounded or as floating (not grounded). Two simultaneously active stimulation channels can have common passive (grounded) contacts, but no common active electrode contacts. This would require current summation for the common contacts.

Further improvements to both the hardware and software of the platform are foreseen. Currently the programming interface is a specific DSP board and the platform only supports monaural experiments. The next generation (research) SP will have USB and will replace both the PI and SP. Furthermore, it will support binaural experiments. On the software side, the new stimulation builder will not have the constraints mentioned above and the USB uplink will be implemented such that microphone input from the SP can be used. Latencies from input to output will be reduced to obtain synchronization between visual cues for lip-reading and auditory perception. Furthermore, the specification of topologies and parameters will be integrated in the platform.

7. CONCLUSION

The SPAIDE platform is versatile and supports a broad range of research applications on sound-processing, stimulation strategies, and psychophysics. The separate configuration of sound-processing strategy, stimulation strategy, and patient-fitting parameters enhances the flexibility.

The open architecture offers two ways to further extend the platform's possibilities. Components of SPAIDE are reused in custom applications and research tools, or custom components (feature blocks) are added to SPAIDE. The platform is powerful because it supports the wide stimulation capabilities of the CII/90K implant. The real-time processing capability is limited by the available resources on PC, which not only depend on clock speed but also on available cache, Windows processes other than SPAIDE, amount of data transmitted over USB, and so forth.

Safety measures are incorporated to maximize patient safety, ranging from checking stimulation strategies to detection of configuration inconsistencies. The status of all components is continuously checked during processing and any unexpected event will immediately result in a stimulation halt. The processing runs under Windows, which results in jitter in the processing delay. However, buffering and synchronization systems in both SPAIDE and the application software in the programming interface minimize the chance of loosing data and of erroneous stimulation.

REFERENCES

[1] B. S. Wilson, "Cochlear implant technology," in *Cochlear Implants: Principles and Practices*, J. K. Niparko, Ed., pp. 109–127, Lippincott Williams & Wilkins, Philadelphia, Pa, USA, 2000.

[2] B. S. Wilson, "Strategies for representing speech information with cochlear implants," in *Cochlear Implants: Principles and Practices*, J. K. Niparko, Ed., pp. 129–170, Lippincott Williams & Wilkins, Philadelphia, Pa, USA, 2000.

[3] S. Peeters, L. Van Immerseel, D. Steenbrugge, and P. Dykmans, "A research platform for cochlear implants," in *Proc. Conference on Implantable Auditory Prostheses (CIAP '01)*, pp. 141–141, Asilomar, Calif, USA, August 2001.

[4] B. A. Swanson, C. A. Irwin, and M. Goorevich, "Nucleus implant communicator," in *Proc. Conference on Implantable Auditory Prostheses (CIAP '01)*, pp. 67–67, Asilomar, Calif, USA, August 2001.

[5] C. A. Irwin, B. A. Swanson, and M. Goorevich, "NIC, NMT and NIC stream research tools," in *Proc. Conference on Implantable Auditory Prostheses (CIAP '03)*, pp. 111–111, Asilomar, Calif, USA, August 2003.

[6] S. Peeters, E. Offeciers, and N. van Ruiten, "Implanted hearing prosthesis," United States Patent 6,355,064.

[7] http://www.innovative-dsp.com.

[8] http://www.bionicear.com.

[9] B. S. Wilson, D. T. Lawson, J. M. Müller, R. S. Tyler, and J. Kiefer, "Cochlear implants: some likely next steps," *Annual Review of Biomedical Engineering*, vol. 5, pp. 207–249, April 2003.

[10] R. P. Morse and E. F. Evans, "Additive noise can enhance temporal coding in a computational model of analogue cochlear implant stimulation," *Hearing Research*, vol. 133, no. 1-2, pp. 107–119, 1999.

[11] R. P. Morse and E. F. Evans, "Preferential and non-preferential transmission of formant information by an analogue cochlear implant using noise: the role of the nerve threshold," *Hearing Research*, vol. 133, no. 1-2, pp. 120–132, 1999.

[12] L. Van Immerseel and A. Goedegebure, "Use of psychoacoustic masking curves to improve speech perception in noise," in *Proc. 7th European Symposium on Paediatric Cochlear Implantation*, pp. 106–106, Geneva, Switzerland, May 2004.

[13] R. M. Bonnet, J. H. M. Frijns, S. Peeters, and J. J. Briaire, "Speech recognition with a cochlear implant using triphasic charge-balanced pulses," *Acta Otolaryngologica*, vol. 124, no. 4, pp. 371–375, 2004.

[14] W. K. Lai, H. Bögli, and N. Dillier, "A software tool for analyzing multichannel cochlear implant signals," *Ear and Hearing*, vol. 24, no. 5, pp. 380–391, 2003.

L.Van Immerseel received the M.S. degree in electrical engineering in 1985 and the Ph.D. degree in electrical engineering in 1993 both from the University of Ghent, Belgium. In 1994 he joined the Medical Electronics Laboratory of the University of Antwerp, Belgium. In 2000 he became a Coworker of the European Advanced Research Laboratory of Advanced Bionics, Antwerp, Belgium. His present research interests are signal processing, modeling of normal hearing, sound processing in cochlear implants, and modeling of neural responses to electrical stimulation. He is a Member of the Acoustical Society of America.

S. Peeters received the M.S. degree in electrical engineering in 1973 and the Ph.D. degree in electrical engineering in 1980 both from the Catholic University of Leuven, Belgium. In 1981 he became a postdoctoral Researcher at the ENT Department, University of Antwerp, Belgium, where he is currently heading the Medical Electronics Laboratory. In addition to his many academic achievements and consulting duties, he cofounded a company that commercialized the Laura cochlear

implant system. He is a Member of the European Society for Engineering and Medicine and of the Royal Flemish Academy of Sciences and Arts.

P. Dykmans received the M.S. degree in electronics engineering in 1988 from the KIHA Engineering College of Antwerp, Belgium. In 1990, he joined the Medical Electronics Research Group of the University of Antwerp. Since then, he has been working as Software Engineer for Philips, Cochlear and Advanced Bionics Cochlear Implant Divisions. His present interests are development of software for the .NET platform.

F. Vanpoucke received the M.S. degree in electrical engineering in 1991 and the Ph.D. degree in electrical engineering in 1995 both from the Catholic University of Leuven, Belgium. In 1995, he joined the Research Lab of Lernout & Hauspie where he became a Principal Research Engineer working on automatic speech recognition. In 2001, he joined the European Advanced Research Laboratory of Advanced Bionics, Antwerp, Belgium. His present research interests are signal processing for and physical modeling of cochlear implants. He is a Technical Committee Member of the Benelux IEEE Signal Processing Chapter.

P. Bracke received the M.S. degree in electronics engineering in 1998 from the Karel de Grote Engineering College of Antwerp, Belgium, and the M.S. degree in computer science in 2001 from the Catholic University of Leuven, Belgium. In 2001, he joined the European Advanced Research Laboratory of Advanced Bionics, Antwerp, Belgium. His present interests are PC and DSP software development.