IMPLEMENTATION OF A HIGH PERFORMANCE FORMANT SPEECH SYNTHESISER(1)

C.D. Summerfield (*) and J.E. Clark(**)

(*)Centre for Speech Technology Research
University of Edinburgh
and
Speech, Hearing and Language Research Centre
Macquarie University

(**)Speech, Hearing and Language Research Centre
Macquarie University

ABSTRACT - This paper describes the hardware structure and the real-time software developed to implement the parallel formant speech synthesiser described by Clark, Summerfield and Mannell(1986). Central to the speech synthesiser operation is a single TMS32010 signal processor microprocessor device which performs the synthesis calculation in real-time.

INTRODUCTION

For real-time synthetic speech production, it is necessary to perform the parallel formant speech synthesis calculation in less than 100μS. This is achieved by using a high speed signal processor device, the TMS32010, to perform the calculation. A separate Interface Processor has been incorporated to manage the complete synthesiser system and control communications between the host computer and the Signal Processor. A database facility has also been provided to supplement the synthesis calculation by supplying glottal source function sample values and formant filter coefficients.

The payoff of this added complexity is that an extremely flexible formant synthesis design can be implemented and achieve real-time performance. The synthesiser has a total of 33 control. These are designed to allow an extremely wide variation in the frequencies, bandwidths and gains of all formant filter in the synthesiser design. Shaping of the output speech spectrum is provided by a variable anti-resonance filter and a dynamic radiation correction filter. Controls are also provided to simulate a large variety of source function spectral and prosodic characteristics, including independent control over the glottal volume/velocity function rise and fall times, F1 bandwidth modulation, and F0 jitter. All synthesis controls parameters may be updated at the synthesis frame rate which is also programmable from 3mS to 31mS intervals.

HARDWARE DESCRIPTION

Figure 1 shows the block diagram of the synthesiser hardware.

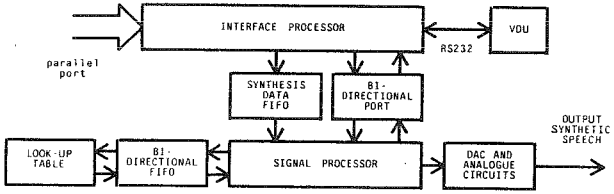

Figure 1. Block Diagram of the Real-Time Speech Synthesiser Hardware

The primary function of the Interface Processor is the control and management of data communications between the Signal Processor and the host computer. It is also responsible for managing the complete speech synthesiser system and contains a number of additional hardware facilities to allow it to carry out this task. A sufficiently large EPROM has been incorporated to allow a limited system monitor to be included in the Interface Processor. This enables the synthesiser to be disconnected from the host computer and used in a local mode for experimental speech sound synthesis. An asynchronous serial I/O port is provided for the connection of a VDU which can be used to manually enter synthesis parameters. A large RAM has also be included in the Interface Processor to accommodate asynchronous communication between the host computer and the Signal Processor. This has the capacity to hold up to 150 frames of synthesis control data. Synthesis data is down-loaded from the host computer at high speed through the parallel communication port. Communication between the Interface Processor and the Signal Processor is controlled through the bi-directional port and the synthesis data FIFO.

The heart of the Signal Processor hardware is the TMS32010 signal processor device. This device performs the synthesis calculation under the control of the Signal Processor ROM which contains the real-time synthesiser software. Auxiliary I/O circuits connect the signal processing device to the various communications channels in the synthesis hardware.

The look-up table database hardware provides the glottal function sample values and formant resonator filter coefficients required by the synthesis calculation. Four tables are stored in separate section of the look-up table EPROM memory and are selected on the condition of flags which accompany the look-up table data vectors. The glottal function values are stored as two sequences. These specify a stylised glottal volume/velocity function and a glottal area function, respectively. The volume/velocity function is derived from two polynomial functions recommended by Rosenberg(1971). These specify the rise and fall profiles of

the glottal volume/velocity function and are stored as single sequences containing 1024 values. A half sine function (from 0 to π radians) is used to specify the glottal area function which is similarly stored as a sequence of 1024 values. The formant filter coefficient values are stored as two data arrays. These are calculated from the formant resonance frequency and bandwidth using the following equations:

$$a1 = 2\exp(-\pi\beta\tau)\cos(2\pi\varphi\tau) \qquad (1)$$

$$a2 = -\exp(-2\pi\beta\tau) \qquad (2)$$

Where φ is the formant centre frequency, β is the 3db bandwidth and τ is the sampler interval (100μS). The a2 coefficients are stored in a single dimensional array which is accessed by a bandwidth vector only. The a1 coefficients are stored as a two dimensional array and are accessed by both bandwidth and frequency vectors. A register has been included in the hardware to temporarily store the bandwidth vector for accessing the a1 coefficient array. The a1 coefficient values are calculated for 256 linear bandwidth increments across the range 10Hz to 2.5KHz and for 512 linear frequency increments across the range 0Hz to 5KHz. The a2 coefficients are calculated for the same bandwidth values as for the a1 coefficients. The bandwidth and frequency graduations are approximately 10Hz and fall within the difference limens specified by Flanagan(1965).

The output analogue speech waveform is produced by a 16-bit digital-to-analogue converter (DAC). Anti-alias filtering is performed by a seventh order elliptic low-pass filter with a cut-off frequency of 4.7KHz and a final power amplifier is used to drive, either the internal speaker, or an external 600Ω +4dbm balanced output line.

SOFTWARE DESCRIPTION

There are two programmes which run concurrently in the synthesiser system: the Interface software, which resides in the Interface Processor, and and the Speech Synthesis software, which resides in the Signal Processor.

At power-on (or reset) the Interface software initialises the complete synthesis system and selects the mode of operation. If the self-test mode is selected the monitor will allocate default values for the synthesis calculation and load these into the Signal Processor for speech synthesis production. If the VDU data entry mode is selected default values are presented on the VDU and can be edited to produce the synthesis sound required. The non-real time communications mode allows down-loading of up to 150 groups of synthesis control data from the host computer. Once this is complete the synthesis operation is started by loading the data into the

Signal Processor. Continuous speech production mode is achieved by transferring synthesis data directly from the host computer to the Signal Processor via the Interface Processor.

The real-time speech synthesiser software is composed of three sections; synthesis initialisation; real-time synthesis calculation and the time-slice background task, as shown in figure 2.

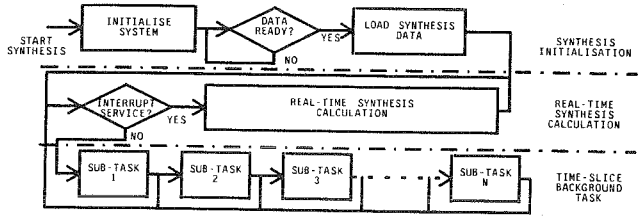

Figure 2. Speech Synthesis Software

The first section initialises the speech synthesis calculation. The programme is then locked until synthesis data becomes available from the Interface Processor which is indicated by the control word on the bi-directional port. When the data is ready, the software reads the control data form the FIFO into the background memory and sorts it before transferring it to the speech synthesis calculation.

Control is passed to the interrupt service routine which detects the occurrence of the 10KHz real-time clock and is responsible for timing of the synthesis calculation. If an interrupt has occurred the first synthetic speech sample value is calculated. This calculation uses the newly acquired synthesis control data and the glottal function and formant filter coefficient values from the look-up table database to determine the synthesis speech sample value. At the completion of this calculation, control returns to the interrupt service routine which provides timing for the complete synthesis operation.

If an interrupt has occurred the following synthesis calculation is immediately commenced. However, in most instances this does not happen and this allows time for the time-slice background task to be performed. This task acquires the synthesis data for the next synthesis frame using a series of time-sliced sub-tasks. Each sub-task performs a short independent portion of the data loading and sorting routine and are run in the interval between the end of the synthesis calculation and the next occurrence of the real-time clock. At the end of each sub-task, control is returned to the interrupt service routine which repeats the synthesis calculation after detection of the next interrupt. On subsequent passes the following sub-task in the time-slice background task sequence is performed. At the completion of this sequence, which normally takes

between 15 and 22 synthesis calculations, the control data for the following synthesis frame is ready for use and is stored in the background memory. At the end of the frame the control data is brought forward into the calculation and the background task sequence is re-initialised. This then commences to acquire the control data for the next frame. If, for some reason, the background task is incomplete at the end of the frame an error condition is flagged and the synthesis operation is halted.

In-line coding has been used to optimise the speed of the synthesis calculation as shown in figure 3.

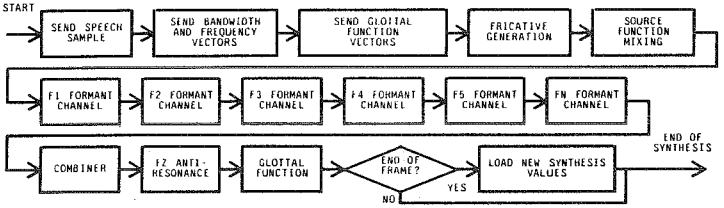

Figure 3. Real-Time Speech Synthesis Calculation

Initially, the speech sample values resulting from the last calculation is sent to the DAC. Following this the formant bandwidth and frequency vectors, and the glottal volume-velocity and area function vectors are sent to the bi-directional FIFO store. These are processed by the look-up table database and the resulting formant filter coefficients and glottal function sample values are returned.

The Signal Processor software proceeds to generate the fricative noise and the mixed excitation sample values for the six formant channel filters (F1 to F5 and FN). These calculations use the differential glottal volume/velocity sample value calculated during the previous cycle.

A formant channel filter calculation includes the calculation of the formant resonance filter and the accompanying formant channel fixed weighting, or skirt, filter. In formant channels F2, F3, F4 and F5 the skirt filters are simple differentiators and are included as part of the formant resonator filter calculation. The F1 formant skirt filter is a second order fixed filter section and is calculated using coefficient specified in the software. The nasal channel (FN) contains only a formant resonator calculation.

The formant resonator filter calculation involves evaluating the equation:

$$o(t) = i(t)a0 + o(t-\tau)a1 + o(t-2\tau)a2 \qquad (3)$$

a1 and a2 are the formant filter coefficient values supplied by the look-up table database and a0 is the filter gain. i(t) is the excitation sample

value (the results of the mixer calculations) and o(t) is the resulting resonance filter output sample value. $o(t-\tau)$ and $o(t-2\tau)$ are the previous two output sample values for the filter.

Once these calculations are complete the resulting six formant channel filter sample values are multiplied by the channel gain factors and combined to produce a single sample value. The anti-resonance filter calculation is then performed on the combined value to produce the final output speech sample value. The coefficients for this filter are derived from the equivalent formant resonance filter coefficients supplied by the look-up table database.

The glottal function values are read from the bi-directional FIFO and combined in a crude down-sampling filter to produce a differential glottal volume/velocity function which is used in the following synthesis calculation. The glottal area function value is also read from the bi-directional FIFO and used to modify the F1 formant bandwidth vector.

Before returning to the interrupt service routine a test is performed to check for the end of the synthesis frame. If the end-of-frame condition is detected the synthesis control data acquired by the background task is loaded into the synthesis calculation and the background task is re-initialised before returning to the interrupt service routine.

CONCLUSIONS

The extreme flexibility provided by this implementation of a parallel formant speech synthesiser enables exceptionally high quality and intelligibility synthetic speech to be produced in real-time.

NOTES

(1) This work was funded by Telecom Australia under contract 68250 and was conducted at Macquarie University.

REFERENCES

CLARK, J.E. SUMMERFIELD, C.D. & MANNELL R.H.(1986) "A High Performance Digital Hardware Synthesiser", (this conference).

FLANAGAN, J.L. (1965) "Speech Analysis, Synthesis and Perception", (Academic Press, New York).

ROSENBURG, A.E. (1971) "Effects of Glottal Pulse Shape on the Quality of Natural Vowels", J. Acoust. Soc. Am. 49, 583-590.