A REVIEW OF VLSI STRUCTURES FOR THE IMPLEMENTATION OF FORMANT SPEECH SYNTHESISERS(1)

C.D. Summerfield(*)

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

ABSTRACT - This paper reviews the VLSI structures for the implementation of formant speech synthesisers. It concentrates on the application of bit-serial arithmetic structures for the implementation of signal processing functions.

INTRODUCTION

The parallel formant speech synthesiser has been shown (Holmes(1983) and Clark, Summerfield & Mannell(1986)) to produce superior speech quality and intelligibility. Up to now real-time implementations of parallel formant synthesisers have been constructed using general purpose signal processing devices (Quarmby & Holmes(1984) and Summerfield & Clark(1986)). The major difficulty with these designs is that a number of compromises have to be made to achieve real-time performance. In the Quarmby & Holmes(1984) synthesiser a number of the less important parameters are fixed, whereas in the Summerfield & Clark(1986) designs, considerable extra circuitry was included to supplement the synthesis calculation to maintain flexibility. In this paper a number of VLSI structure are reviewed that allow a highly flexible synthesiser design similar to that described by Clark, Summerfield & Mannell(1986) to be implemented as a single VLSI device. Throughout the design, bit-serial arithmetic structures based on the VLSI primitives used in the FIRST silicon compiler and described by Denyer & Renshaw(1985) have been utilised. This technology is ideally suited to speech synthesis applications. It allows extremely high internal data resolution to be realised within compact VLSI structures. However, the high data resolutions have a consequential processing time penalty, but, as will be described in this paper, the processing bandwidth of modern VLSI technologies is far in excess of that required for real-time speech synthesis and, thus, can be ignored in this application.

The following discussion will concentrate on the VLSI implementation of the formant channel filters and a scheme for generating the resonance filter coefficients from the formant frequency and bandwidth data. The compact structures produced by using a bit-serial approach allows all the synthesis function to be implemented on a single chip.

FORMANT CHANNEL FILTERS

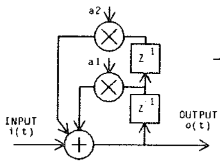
Central to the design of the parallel formant speech synthesiser is the implementation of the formant channel filters. A bit-serial approach is used exclusively to implement this section of the VLSI synthesiser. There are four circuits in the channel filter design, the input excitation function mixer and gain circuits, the resonance filter circuits, the fixed formant skirt filters and the output accumulator circuit.

The mixer and gain have been combined in the VLSI implementation to minimise the VLSI primitive component count. In this circuit 2 serial multipliers are used to control the gain of the voiced and fricative excitation functions independently. The output of the multipliers is connected to an adder primitive to combine the excitation waveforms. In this arrangement it is necessary to provide the absolute gain of both the voiced and fricative components in the waveform.

The formant filter calculation is performed on the bit-serial data provided by the mixer/gain circuit. The formant resonance filter is realised as a second-order direct form 2 recursive digital filter as shown in figure 1a. The output sample values of this filter are calculated using the equation:

$$o(t) = i(t) + o(t-\tau)a1 + o(t-2\tau)a2 \qquad (1)$$

Where o(t) is the output sample value and is calculated from the sum of the input sample value i(t) and the sum of the product terms, $o(t-\tau)a1$ and $o(t-2\tau)a2$. a1 and a2 are the resonance filter coefficients.



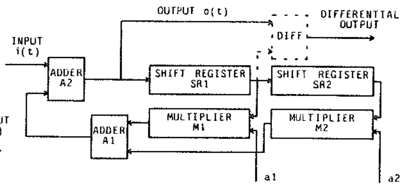


Figure 1a. Second-Order Filter. Figure 1b. Bit-Serial VLSI Structure.

The second-order filter is implemented in a serial arithmetic structure using 2 shift register, 2 multipliers and 2 adder primitives connected as shown in figure 1b. In this structure the 2 multipliers, M1 and M2, perform the product calculation $o(t-\tau)a1$ and $o(t-2\tau)a2$, respectively. As the structure is clocked the serial data in SR1 is applied to the input port of the serial multiplier M1. At the same time, the serial representation of the coefficient value a1 is applied to the coefficient port of the multiplier to produce the first product term at the output. A second multiplier M2 is clocked concurrently with M1 to produce the second product from the serial representation of $o(t-2\tau)$ stored in SR2 and the coefficient a2. The clocking also shifts the serial value in shift register SR1 to SR2 for the following calculation. The product terms at the output of M1 and M2 are applied to a serial adder structure A1 to form the partial summation. This is combined with the input value i(t) in a second adder primitive A2 to complete the filter calculation. The recursive path is formed by connecting the output of the second adder circuit to the input of the first shift register SR1. It is necessary in the filter calculation to apply some form of non-linear truncation of the results before it is applied to the shift register structure. In the present design rounding is applied before truncation to minimise numerical errors in the filter calculation.

In this structure the data is represented in a bit serial form using a 2's complement format. All calculations are performed LSB first and sign extensions are automatically applied by the arithmetic structures. It is important at the outset to consider the resolution of the data representation in the filter calculation as this determines, to some extent, the size of the VLSI structure and, to a greater degree, the speed of the computation. To maintain good signal-to-quantisation noise ratio in the output synthetic speech it is necessary to use a long data word-length within the filter structure. In the present design 24-bit data representation is specified for the filter calculation. This specification exceeds the signal-to-quantisation noise performance from synthesiser designs based on general purpose signal processing devices.

By rearranging the primitives in this structure to form a non-recursive circuit, an anti-resonance filter may be produced. The same data resolution specification are applied for good signal-to-noise performance.

The skirt filters for formant channels F2, F3, F4 and F5 are simple differentiators. These can be implemented by introducing an extra subtractor primitive in the formant filter structure. It is convenient to include the subtractor here as the previous filter output value $(o(t-\tau))$ is available from the shift register. The skirt filter for F1 is a fixed second order section containing poles and zeros and is implemented separately from the resonance filter structure. Work is progressing to minimise the filter size by approximating the coefficient values as combinations of power of 2 components. This enables the multiplying operation to be substituted by smaller adder and subtractor primitives.

The results of the formant channel calculations are accumulated in a single shift register. Formant channel results are either added to, or subtracted from, the shift register value by multiplexing an adder or subtractor primitive at the input to the accumulator register.

Even with the extended data resolution, the clock rate for real-time speech synthesis for this structure is extremely modest for modern VLSI technologies. Therefore, it is practical to multiplex the formant filter structure (and the mixer/gain structure) for all formant channel filter calculations (F1 to F5 and Fn) in the synthesiser design and still produce real-time operation with relatively modest clock rates. Apart from the multiplexing logic to direct the input and output serial data, the only additional circuitry required is an extension to the shift-registers SR1 and SR2 to accommodate the increase in data storage requirements for each formant channel (Jackson, Kaiser & McDonald(1968)).

COEFFICIENT GENERATION

The coefficient values a1 and a2 used in the formant resonance filter calculation are derived from the formant frequency and bandwidth specification using the equations:

$$a1 = 2R\cos(\theta) \qquad (2a)$$
$$a2 = -R^2 \qquad (2b)$$

R is the radial distance from the origin (on the z-plane) and is derived from the formant bandwidth specification (β) by $R=\exp(-\pi\beta\tau)$. The vector angle (on the z-plane) θ is derived from formant frequency (φ) using the hertz-radians relation $\theta=2\pi\varphi\tau$. In both these equations τ represents the synthesis sampling interval ($100\mu S$).

The formant filter coefficient are derived from the input formant frequency and bandwidth specifications using 2 Programmable Logic Arrays (PLA's) and 2 serial multipliers as shown in figure 2 and has the capacity to generate all resonance filter coefficients within every sample period. PLA1 determines the z-plane radial distance (R) from the bandwidth specification (β) by providing the mapping $R=\exp(-\pi\beta\tau)$. The cosine term in equation 2a is derived from the input frequency specification (φ) by using a second PLA (PLA2) to perform the mapping $\cos(\theta)$. Two serial multiplier primitives are used to derive the coefficient values from the output of these arrays. The value of a1/2 is determined by computing the product $R\cos(\theta)$ from the data output of PLA1 and PLA2. This can be doubled by bit-shifting (shift register scaling) to form the coefficient value a1. The negative result of a2 ($-a2$) is produced by squaring the output value from PLA1 using a second serial multiplier. The negative result of this may be cancelled in the formant filter calculation by substituting a subtractor primitive in place of A1 in the formant filter structure.

The absolute size of the PLA's are determined by the frequency and bandwidth resolution required. However, the size of PLA2 can be halved by using the relation $\cos(\theta)=-\cos(\pi-\theta)$. A half size PLA is constructed to cover the range $\theta = 0$ to $\pi/2$ radians. The correct result for frequencies above this range is achieved by reflecting the frequency value about $\pi/2$ radians and taking the negative result at the output of the PLA.

Two exclusive OR gate buffers (XOR1 and XOR2) controlled by the MSB of the frequence address bus are used to perform this operation.

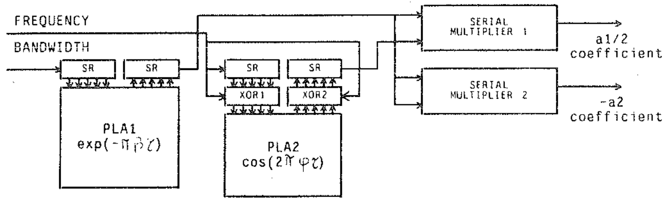


Figure 2. Coefficient Generator VLSI Implementation.

SOURCE FUNCTION GENERATOR

The strategy for producing voiced and fricative excitation functions in the VLSI speech synthesiser is similar to that used in the real-time hardware implementation described by Summerfield and Clark (1986).

The pitch period is controlled by a 16-bit programmable counter circuit. This counter initiates the glottal volume/velocity pulse generation at the start of each pitch period. In the VLSI structure a stylised glottal volume/velocity function is stored in a PLA structure. This is addressed by an accumulator which is incremented by glottal pulse rise and fall slope factors. An overflow condition from the accumulator is used to reset the glottal pulse generation and it is not re-initiated until the start of the next pitch period.

Four glottal waveform values are produced every sample period. A simple down-sampling filter with coefficient values based on powers of 2 is used to combine the glottal function values to produce a single glottal volume/velocity function value. The radiation characteristic is introduced at this point by differentiating the full glottal volume/velocity function.

Fricative noise is generated by a serial random bit sequence generator. This consists of a serial shift register with a number of exclusive OR feedback loops. A serial multiplier is used to produce voiced fricative excitation by modulating the random bit sequence with the glottal volume/velocity function. Spectral shaping of the fricative signal is provided by a fixed second-order section based on power of 2 coefficients.

SYNTHESISER CONTROL

Synthesiser design contains an internal RAM for storage of all the synthesis control parameters. At the start of each calculation the RAM is accessed by the synthesis structures and all the tract function parameters (formant frequencies, bandwidths and gains) are updated. The source function parameters (pitch period, glottal pulse rise and fall factors and the fricative voicing index) are read from the internal RAM at the start of every pitch period. This allows any synthesis control parameter to be updated as often as required. Two output clocks are provided for external circuit timing, the sample rate clock (10KHz) and a pitch period clock. These are used to synchronise the external controlling circuitry and provide controls for various synthesis data updating strategies, including pitch synchronous, frame synchronous or continuous updating of interpolated synthesis data.

COMMENTS AND CONCLUSIONS

It is estimated that these VLSI structures have 8 times the processing bandwidth required to produce real-time synthetic speech. This may be utilised in several ways. The design may be reduced to a multiplexed structure containing single set of arithmetic primitives. This would produce an extremely compact single channel speech synthesiser design. Conversely, the multiplexing arrangements may be extended to produce a multi-channel high performance formant speech synthesiser device.

NOTES

(1) This work forms part of the author's candidature for the degree of Doctor of Philosophy at the Department of Electrical Engineering, University of Sydney.

REFERENCES

CLARK, J.E. SUMMERFIELD, C.D. & MANNELL, R.H.(1986) "A High performance Parallel Formant Speech Synthesiser", (this conference).

DENYER, P. & RENSHAW, D.(1985) "VLSI Signal Processing: A Bit-Serial Approach", (Addison-Wesley).

HOLMES, J.N.(1983) "Formant Synthesizers - cascade or parallel?", Speech Communication, 2, pp 251-273.

JACKSON, L.B., KAISER, J.F. & McDONALD H.S.(1968) "An Approach to the Implementation of Digital Filters" IEEE Trans Vol AU-16, No3, pp 413-421

QUARMBY, D.J. & HOLMES J.N.(1984) "Implementation of a parallel-formant synthesiser using a single-chip programmable signal processor", IEE Proc. Vol 131 Pt F No. 6, pp 563-569.

SUMMERFIELD, C.D. & CLARK, J.E.(1986) "Implementation of a High Performance Speech Synthesiser", (this conference).