15

SOC Design for Speech-to-Speech Translation

Shun-Chieh Lin[1], Jia-Ching Wang[2], Jhing-Fa Wang[3],
Fan-Min Li[4] and Jer-Hao Hsu[5]
[1]*Industrial Technology Research Institute*
[2]*National Central University*
[3, 4, 5]*National Cheng Kung University*

1. Introduction

In today's globalised world, information exchange between different languages is indispensable. Accordingly, speech-to-speech translation researches [1]-[6] grew to a leading-edge technology enabling multilingual human-to-human and human-to- machine interaction. In previous work, two different architectures are adopted by many researchers for speech-to-speech translation research [4],[15] – a conventional sequential architecture and a fully integrated architecture. The sequential architecture is composed of a speech recognition system followed by a linguistic (or non-linguistic) text-to-text translation system and a text-to-speech system [1],[2],[5],[16-20]. The integrated architecture combines speech feature models and translation models in a manner similar to that used for speech recognition. This integration brings about efficient translation by searching for an optimal word sequence of target language through the integrated network such as finite-state transducer [4],[21] and the others [22],[23].

According to these two architectures, there are two system implementations – client-server-based systems and stand-alone handheld systems. Nevertheless, a critical shortcoming of client-server based speech translation systems is that they should be built on the server computer. In other words, it is not available anytime or anywhere. An obvious solution would be to build portable stand-alone speech-to-speech translation handheld devices. To mention just a few: Isotani *et al.* [7] used a Pocket PC PDA with 206 MHz StrongARM/64 MB RAM to build a speech-to-speech translation system for the use in various situations while traveling. *Waibel et al.* [8] adopted a Pocket PC PDA with 400 MHz StrongARM/64 MB RAM to construct a speech-to-speech translation system for medical interviews. Watanabe *et al.* [9] developed a mobile device running on 400 MHz Pentium II-class processor/192 MB RAM that helps speech-to-speech translation in various situations during their travel abroad. However, these works show that real-time speech-to-speech translation with a resource-limited device is still a problem.

Table 1 shows a comparison among related speech-to-speech translation systems, including JANUS [1], Verbmobil [2], EUTRANS [4] etc. Clearly, the client-server based architecture can work in real time, but is not portable; while the stand-alone system is portable, but lacks the real-time performance. Therefore, a VLSI solution is presented by realizing the entire

speech-to-speech translation algorithm within a single chip. This SOC chip only requires a few peripheral components for complete operation, and is characterized by small size, low cost, real-time operation, and high reliability. The construction of this chip is accomplished in two main phases: the software simulation phase and the SOC design phase.

In the software simulation phase, the simulation is based on the multiple-translation spotting (MTS) method, a kind of integration of speech analysis and language translation [23]. The proposed multiple-translation spotting approach is directly from speech to speech without language models like other automatic speech recognition (ASR) approaches. With identifying speech features, translation primarily stays in the speech modality and does not go through a textual modality. The proposed MTS method not only retrieves the optimal multiple-translation spotting template, but also extracts the appropriate target patterns. With the extracted patterns, the target speech can be generated by a concatenation-based waveform segment synthesis method.

In the SOC design phase, besides a cost efficient programmable core used for system control and non-computation-intensive tasks, three specific hardware cores were designed to perform cepstrum extraction, template retrieval, target pattern extraction. Moreover, the A/D converter (ADC) and D/A converter (DAC) are also designed.

The rest of this paper is organised as follows. Section 2 gives the software simulation of the proposed speech-to-speech translation system. Section 3 discusses the SOC architecture for the MTS-based speech-to-speech translation system. Finally, a short conclusion is provided in Section 4.

System	Vocabulary Size	Response time	Specification
JANUS [1]	3,000~5,000	≤ 2 times real-time	PC-based platform (server-client)
Verbmobil [2]	≥10,000	4 times real-time	PC-based platform (server-client)
EUTRANS [4]	1,701(English) 2,459(Italian)	≤ 3 times real-time	PC-based platform (server-client)
ATR-MATRIX [5]	13,000	0.1 sec for TDMT	PC-based platform (server-client)
Isotani *et al*. [7]	20,000(English) 50,000(Japan)	Slower than V_R5500 processor	Pocket PC PDA (206 MHz StrongARM/64 MB RAM)
Speechalator [8]	–	≥2~3 sec	Pocket PC PDA (400 MHz StrongARM/64 MB RAM)
Watanabe *et al.* [9]	10000 (English) 50000 (Japan)	≥2~3 sec	Mobile PC (400 MHz Pentium II-class processor/192 MB RAM)

Table 1. A comparison among related speech-to-speech translation systems

		Source-language (English)	Target-language (Chinese)
TS	SL Input		
	I want a double room.	I want a single room with shower for tomorrow.	我 明天 要 一 間 有 淋浴 設備 的 單人 房
		⟨**I, want, a, room**⟩	⟨**我,要,一,間,房**⟩
	Is there a single room for tomorrow?	Is there a double room for tonight?	請 問 你們 今晚 有 一 間 雙人 房 嗎
		⟨**Is there, a, room, for**⟩	⟨**有,一,間,房,嗎**⟩
MTS	SL Input		
	I want a double room.	I want a ⟨single, double⟩ room with shower for ⟨tomorrow, tonight⟩.	我 ⟨明天,今晚⟩ 要 一 間 有 淋浴 設備 的 ⟨單人,雙人⟩ 房
		⟨**I, want, a, double, room**⟩	⟨**我,要,一,間,雙人,房**⟩
	Is there a single room for tomorrow?	Is there a ⟨single, double⟩ room for ⟨tomorrow, tonight⟩?	請 問 你們 ⟨明天,今晚⟩ 有 一 間 ⟨單人,雙人⟩ 房 嗎
		⟨**Is there, a, single, room, for, tomorrow**⟩	⟨**明天,有,一,間,單人,房,嗎**⟩

Table 2. Examples for TS and MTS.[1]

2. The Proposed Speech-to-Speech Translation System

2.1 System Overview

Multiple-translation spotting (MTS) is proposed as an improvement on the traditional translation spotting (TS) method [11], [12] and uses multiple-translation spotting templates derived from multiple pairs of translations for spotting all hypothesised target-language patterns. Table 2 lists some examples of TS and MTS. The source-language input "*I want a double room*" can only provide five target-language patterns <我,要,一,間,房> in traditional TS method but the proposed MTS provides all target-language patterns <我,要,一,間,雙人,房>. All target-language patterns <明天,有,一,間,單人,房,嗎> are also extracted by MTS while inputting the sentence "*Is there a single room for tomorrow.*" For each multiple-translation spotting template, hypothesised target patterns are generated in the MTS process.

Figure 1 shows an example of the MTS process for the direction of English-to-Chinese while inputting a speech. The multiple-translation spotting template of this example possesses eleven English patterns in a feature template. Based on the one-stage algorithm [13], when a speaker inputs an English speech "*Is a single room still available for tonight*", the proposed system can obtain the identified results – "*is*", "*a*", "*single room*", "*still*", "*available*", "*for*", and "*tonight.*" Following the identified results, the translations of "*is*↔嗎", "*a*↔一間," "*single room*↔單人房", and so on are extracted. A Mandarin Chinese speech pattern set that includes seven hypothesised speech patterns, "嗎", "一間", "單人房", "還", "有", and "今晚" is generated by the constructed waveform translations in the multiple-translation spotting template and represented as darker waveforms in Fig. 1.

For speech generation, after determining the optimal target speech sequence, these waveforms are rearranged with adequate overlapping portions to generate speech with the

[1] The gray tablets list the translation spotting results.

waveform similarity overlap and add (WSOLA) algorithm [24]. WSOLA introduces a tolerance on the desired time-warping function to ensure signal continuity at waveform segment joins. With a proper windows length and a timing tolerance, WSOLA usually produces high quality time-scaled speech [25],[26]. Therefore, the system can generate high phonetic/prosodic quality in the translated speech output. The advantages of this method are the small computational cost during the generation process and the high intelligibility of the generated speech. The following subsections further discuss the details of the kernel spotting algorithm within the speech translation.

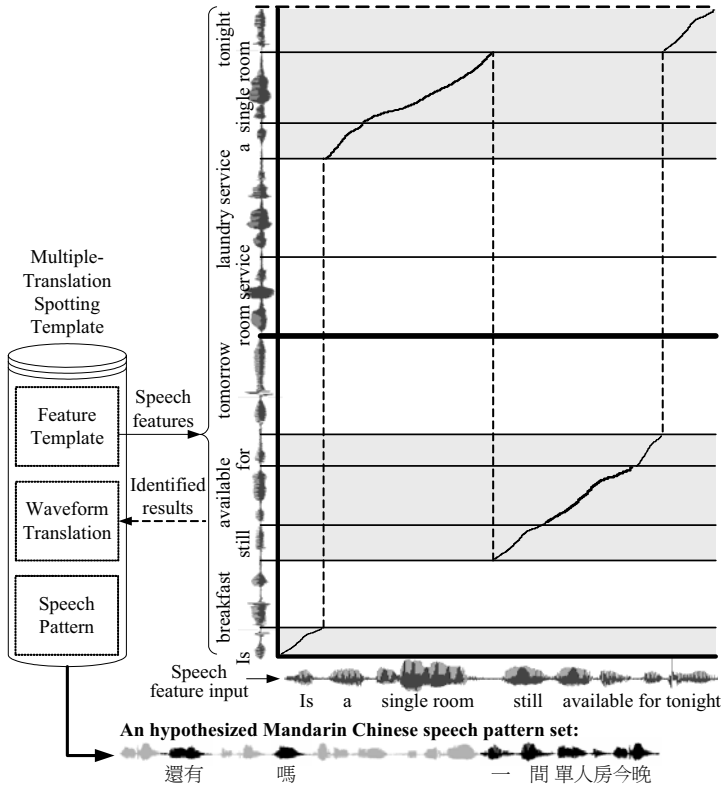

Fig. 1. An example of MTS process for the direction of English-to-Mandarin Chinese.

2.2 Multiple-Translation Spotting

To formulate the concept of MTS between input speech (X_1^L) and the v-th multiple-translation spotting template (r_v), we use the following notations: l represents the frame index within X_1^L, $1 \le l \le L$, j represents the translation pair $\langle s_j^v, t_j^v \rangle$ index within r_v, $1 \le j \le J$, and k represents the frame index within the j-th source pattern s_j^v and its mapped target pattern denoted by t_j^v, $1 \le k \le K_j$. Then for each input frame, the accumulated distortion $d_A(l,k,j)$ is defined by:

$$d_A(l,k,j) = d(l,k,j) + \min_{k-2\le m\le k}\left(d_A(l-1,m,j)\right), \tag{1}$$

for $2 \le k \le K_j$, $1 \le j \le J$, where $d(l,k,j)$ is the local distortion between the *l*-th frame of X_1^L and the *k*-th frame of the source pattern s_j^v. The recursion in (1) is carried out for the internal frames (i.e., $k \ge 2$) of each source pattern. At the pattern boundary, i.e., when $k = 1$, the recursion can be calculated as:

$$d_A(l,1,j) = d(l,1,j) + \min\left[\min_{1\le m\le J}\left(d_A(l-1,K_m,m)\right), d_A(l-1,1,j)\right]. \tag{2}$$

And the best path is determined by

$$d_G^v = \min_{1\le j\le J}\left[d_A\left(L,K_j,j\right)\right]. \tag{3}$$

After ranking all the templates, the hypothesized spotting template is decided from the Top *N* candidates with minimum distortion by

$$\hat{v} = \arg\max_{1\le v\le N}\sum_{j=1}^{J}\tau_j^v \tag{4}$$

According to the decided $\hat{v}$*-th* template from (4), the target patterns $\left\{t_j^{\hat{v}}\right\}_{j=1}^{J}$ can be obtained by $\left\{\tau_j^{\hat{v}}\right\}_{j=1}^{J}$. With the determining target speech patterns, these waveforms are rearranged with adequate overlapping portions to generate speech with the waveform similarity overlap and add (WSOLA) algorithm.

2.3 Software Simulation

The translation experiments were performed on both a PC-based platform and iPAQ PDAs. On the PC-based platform, the software simulations were done using Windows CE 3.0 on a Pentium® IV 1.8 GHz, 1 GB RAM, Windows® XP PC. On the COMPAQ iPAQ PDAs, the system was implemented on a 400 MHz Intel® XScale processor with 128 MB RAM. This work built a collection of English sentences and their Chinese translations that frequently appear in phrasebooks for foreign tourists. The phrasebooks are referred to [27] and [28]. Six sub-domains are used for collection including accommodation, restaurant, traffic, shopping, tourism, and asking for directions. Each sub-domain contains 174~251 translations.

Experiments were conducted between Mandarin Chinese and English. To evaluate the system performance, the 1,260 utterances of the training set used for constructing multiple-translation spotting templates were collected from one speaker (Sp1) and the 105 utterances of the test set were also collected from the same speaker (Sp1) for closed testing and two bilingual male speakers (Sp2 and Sp3) for open testing. First speaker's feature models (spotting templates) were used to proceed tests on the remaining two speakers. All speeches had an 8 kHz sampling rate with a precision of 16 bits on both the PCs and the PDAs. The speech feature analysis was performed using 10th-order linear prediction cepstral coefficients (LPCCs) using 32 ms frames with 8 ms overlap. The total memory requirement of the whole system at the startup was about 22.9 MB. The required work memory is about 1 MB, depending on the length of a speech input.

While hesitations or insertions are occurred, the proposed MTS approach can spot critical patterns as keyword spotting or partial matching and a target output can be properly generated by the spotted patterns with the decided spotting template. However, the proposed approach would be sensitive to disturbances including speaking rate, noise, speaker properties, and so on. For the effect of speaker properties for the proposed system,

For retrieving Top 5 templates, Table 3 shows that the spotting accuracy of Sp2 and Sp3 drops by 10 to 15 percent. A given spotting template is called a *match* when it obtains the same intention of the input speech. In addition, from the research presented in [31], the speaking rate had a significant effect on recognition accuracy and further adaptation methods of duration models for spotting templates are needed. Based on the proposed approach, when template or vocabulary size increases, the increasing spotting templates will lead to more speech feature vectors and hence more similarities will occur in speech spotting measurement, thus causing false spotting results and lowering spotting accuracy. By collecting more speech databases, the system can apply speaker-dependent or speaker-independent HMM to MTS for more robust speech translation. Speech translation performance also is degraded by noise. Related works to minimize the effects of the noise on the system performance are presented in [29],[30] and would be applied to the proposed system in the future.

To judge the generated translations from the matched templates, a subjective sentence error rate (SSER) in [3] is used to evaluate and classify the target generation results into three categories by three bilingual evaluators. Referring to the SSER evaluation method, *good* (*G*), *understandable* (*U*), *bad* (*B*) levels of translation quality are scaled to 1.0, 0.5, and 0.0, respectively. The understandable translation rate is calculated by

$$\text{rate} = \frac{1.0 \times (\text{no. of } G \text{ level}) + 0.5 \times (\text{no. of } U \text{ level})}{\text{no. of tests}} \times 100\% \tag{5}$$

The results revealed that our proposed approach roughly achieves 89% and 92% understandable translation rate from (5) for the Mandarin Chinese-to-English and the English-to-Mandarin Chinese translations, respectively.

			Template Size (spotting templates of Sp1)									
		Top5	360	460	560	660	760	860	960	1060	1160	1260
All	Sp1	E2C	0.96	0.93	0.93	0.9	0.86	0.83	0.83	0.8	0.76	0.73
		C2E	0.93	0.9	0.86	0.83	0.8	0.8	0.76	0.76	0.73	0.7
	Sp2	E2C	0.83	0.8	0.76	0.73	0.73	0.7	0.66	0.66	0.66	0.63
		C2E	0.8	0.76	0.73	0.73	0.7	0.66	0.63	0.63	0.63	0.6
	Sp3	E2C	0.76	0.76	0.73	0.7	0.7	0.66	0.66	0.63	0.6	0.6
		C2E	0.73	0.73	0.7	0.66	0.66	0.63	0.6	0.6	0.6	0.6

Table 3. Average spotting accuracy in multiple speaker testing.

3. SOC Design for the MTS-based Speech-to-Speech Translation System

The VLSI architecture for the speech-to-speech translation SOC was developed using the programmable application-specific technique. Besides a cost efficient programmable core [14], which is used for system control and other non computation-intensive tasks, three specific hardware cores are also designed for feature extraction, template retrieval, pattern extraction. Moreover, the A/D converter and D/A converter are also designed.

Figure 2 shows the overall block diagram of the VLSI architecture for the speech-to-speech translation SOC. This architecture mainly consists of a cepstrum extraction core, a template retrieval core, a pattern extraction core, a programmable core, and ADC/DAC. In the

following paragraph, we will discuss the three specific processing hardware cores and the ADC/DAC for our speech-to-speech translation system.

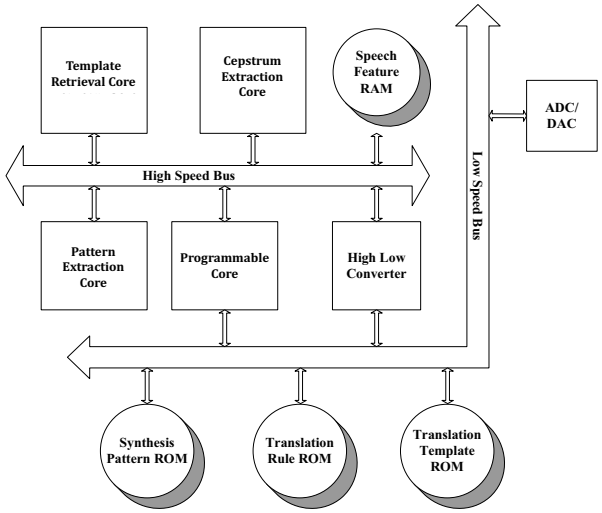

Fig. 2. Block diagram of the SOC architecture for the proposed speech-to-speech translation system.

3.1 The Design of the Cepstrum Extraction Core

Feature extraction is one of the most important issues in the field of speech recognition. In many speech recognition systems, the cepstral coefficients are treated as the feature parameters, derived by the Fourier Transform or from the LPC coefficients. We selecte the latter way as it yields high recognition rate and lower computational load.

3.1.1 Autocorrelation Analysis

The sampling rate of the input speech for the cepstrum extraction core is 8 KHz. The pre-emphasis filter for the digitized speech is defined by

$$H(z) = 1 - hz^{-1}, \quad 0.9 \le h \le 1.0 \cdot \tag{6}$$

Because we adopt fixed-point implementation, h is chosen as 15/16. The pre-emphasized speech is then blocked into N samples, with adjacent frames being separated by M samples. In other words, the adjacent frame begins M samples later than the previous frame and overlaps with it by N-M samples. In our design, M=192, the frame size N=256, and therefore the frame rate equals 31.25 frames per second. The next step is to window each frame so that the signal discontinuities at the beginning and the end of each frame are minimized. The Hamming window is used and has the form of

$$w(i) = \begin{cases} 0.54 - 0.46\cos(\frac{2i\pi}{N-1}), & 0 \le i \le N-1 \\ 0, & otherwise \end{cases} \tag{7}$$

For each frame of the windowed speech signal, autocorrelation analysis is performed by the following formula:

$$R(k) = \sum_{i=k}^{N-1} x(i-k)x(i), 0 \le k \le P' \tag{8}$$

where P is the order of the LPC analysis.

The detailed architecture for the autocorrelation analysis of overlapping frames from the digitized input speech can be seen in [32]. The architecture is based on a calculation procedure which is depicted in Fig. 3.

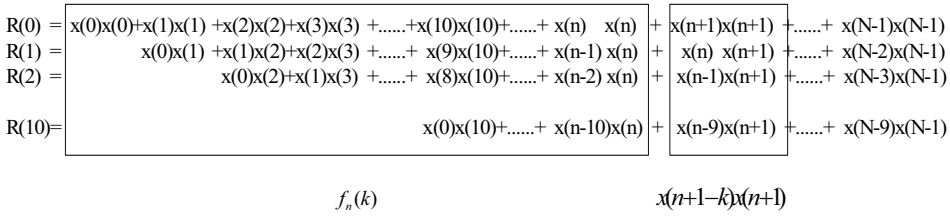

Fig. 3. An example illustrating the autocorrelation calculation procedure.

3.1.2 Linear Predictive Analysis

Of the various linear predictive analysis algorithms, we adopt the autocorrelation method because it leads to more stable results than the covariance method and has lower computational load than the lattice method [33]. For the autocorrelation method, the matrix equation for solving the LPC coefficients is in the form of

$$R(i) = \sum_{k=1}^{P} a_k R(|i-k|), \qquad 1 \le i \le P' \tag{9}$$

where $R(i)$ is the autocorrelation coefficient, P is the order of the LPC analysis, chosen as 10 in this paper.

The most efficient method known for solving this particular system of equations is the Levinson-Durbin recursion, which can be formulated as follows [33]:

$$E^{(0)} = R(0) \tag{10}$$

for $1 \le i \le P$

$$temp = R(i) - \sum_{j=1}^{i-1} a_j^{(i-1)} R(i-j) \tag{11}$$

$$K_i = temp / E^{(i-1)} \tag{12}$$

end for

$$a_i^{(i)} = K_i \tag{13}$$

$$a_j^{(i)} = a_j^{(i-1)} - K_i a_{i-j}^{(i-1)}, \qquad 1 \le j \le i-1 \cdot \tag{14}$$

$$E^{(i)} = (1 - K_i^2) E^{(i-1)}, \tag{15}$$

The above equations are solved recursively for $i = 1, 2, \cdots, P$ and the final solution is given as

$$a_j = a_j^{(P)}, \qquad 1 \le j \le P \cdot \tag{16}$$

Substituting (12) into (15), we obtain that

$$E^{(i)} = E^{(i-1)} - temp \cdot K^{(i)} \cdot \tag{17}$$

The architecture based on pipeline fashion for the Levinson-Durbin recursion is described in

Fig. 4. There are a small number of divisions needed for the LPC computation. However, it is not economical to prepare extra hardware for them. Instead, the division operations can be performed by prune-and-search [34] on the hardware without the use of an individual divider.

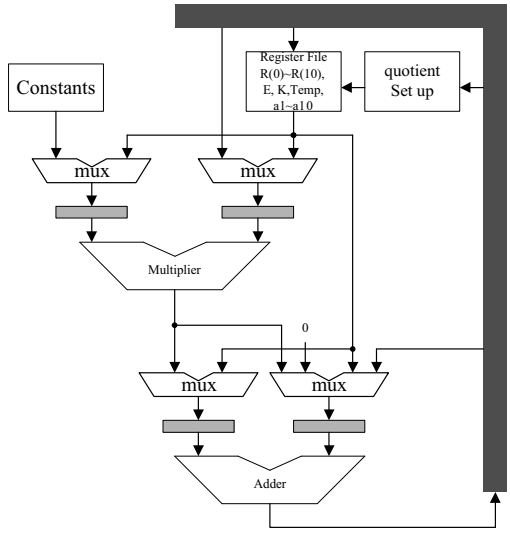

Fig. 4. The architecture for the LPC analysis.

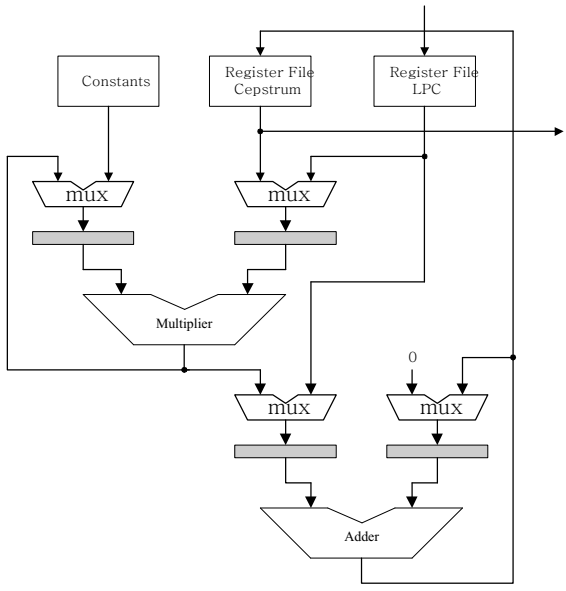

Fig. 5. The architecture for the conversion from LPC to cepstrum.

3.1.3 Conversion from LPC to Cepstrum

The cepstrum is computed from the linear prediction model. The recursive equations of the LPC based cepstrum are as follows [35]:

$$C_1 = -a_1, \tag{18a}$$

$$C_n = -a_n - \sum_{m=1}^{n-1}(1-m/n)a_m C_{n-m}, \qquad 1 \le n \le P, \tag{18b}$$

where C_n is the nth cepstral coefficient, a_n is the nth LPC coefficient, and P is the cepstrum order, which is set equal to the LPC order in this paper.

The memory requirement for the constant $(1-m/n)$ can be reduced as follows. There are n-1 constants for any n, so the total number of the various constants is 1+2+3+…+P-1=$P(P$-1)/2. By taking into consideration the pattern of the constant array, let $k_{m,n}$=(1-m/n), then we can show that

$k_{n\text{-}1\text{-}i,n}$=1- $k_{i,n}$, (19)

for i=1, 2 ,…,$\lceil n-1/2 \rceil$, and if n is even then $k_{n/2,n}$=1/2.

Equation (19) implies that $k_{i,n}$ is equal to the complement of $k_{n-1-i,n}$, therefore, the size of the constant array can be reduced to half. This algorithm is implemented by a store-and-accumulate technique and the architecture is described in Fig. 5. We need two storage elements to hold the LPC coefficients and the previous order cepstral coefficients. In addition, an accumulator is necessary for computing and totaling the main expression, (1-m/n)$a_n C_{n\text{-}m}$ to yield the cepstral coefficients defined in (18b).

The dedicated architecture for the autocorrelation analysis, the LPC analysis, and the conversion from LPC to LSP are individually designed. However, since the three procedures are performed sequentially, a resource-sharing technique,is performed so that the cepstrum extraction core will need only one multiplier and one adder.

3.2 The Design of the Template Retrieval Core

To exemplify our design concept, Figs. 6(a) and 6(b) display two dependence graphs (DGs) of the template retrieval. These dependence graphs are constructed according to (1) and (2), and can be regarded as the template retrieval space. The DG shown in Fig. 6(a) describes the case with 12 frames in the input speech (X_1^{12}). There are three source patterns ($s_1^{v1}, s_2^{v1}, s_3^{v1}$) in this template ($r_{v1}$), and the frame number of them is 7, 8, and 6, respectively. The DG shown in Fig. 6(b) has 8 frames in the input speech (X_1^8) and 2 patterns in the template (r_{v2}). The frame number of the two patterns (s_1^{v2}, s_2^{v2}) in this template is 6 and 7, respectively.

The different frame numbers within different input speeches cause the variation along the horizontal axis in the template retrieval space. The different pattern numbers within different templates and the different frame numbers within different patterns are responsible for the variation along the vertical axis in the template retrieval space. The architecture design for coping with the variation of the structure of the template retrieval space is described as follows. First, horizontal projection is performed in the original DGs (Fig. 6(a)) and the result is shown in. Fig. 6(c). This projection reduces the two-dimensional (2-D) template retrieval space into a one-dimensional (1-D) one and eliminates the variation resulting from different frame numbers within different input speeches. Each node in Fig. 6(c) requires a register to store the computational results for the distortion accumulation of the next frame. The nodes within one

column are divided into different blocks according to the patterns they belong to. Thus the DG in Fig. 6(c) consists of three blocks corresponding to three different patterns.

To further reduce the size of the DG, a vertical projection is performed in Fig. 6(c). This projection transforms the three-block DG into a one-block DG (see Fig. 6(e)). Because the frame numbers for the blocks are different, the largest frame number is taken as the frame number for the new one-block DG. To deal with the discrepancy of having different frame numbers within different patterns, we added some multiplexers. In addition, two extra registers are added in front of each node to replace the two eliminated blocks. Figure 6(e) is then modified as Fig. 6(g) to have a regular wire connection.

Figures 6(b), 6(d), 6(f) and 6(h) provide another example of the use of horizontal and vertical projections. One can find that the frame number and the register number for a certain node between the DGs in Figs. 6(g) and 6(h) are different. To combine the two DGs into a single one, the largest frame number and the largest register number between them are chosen, see Fig. 6(i). The added multiplexers in front of each node provide the selectivity among the one-block DGs.

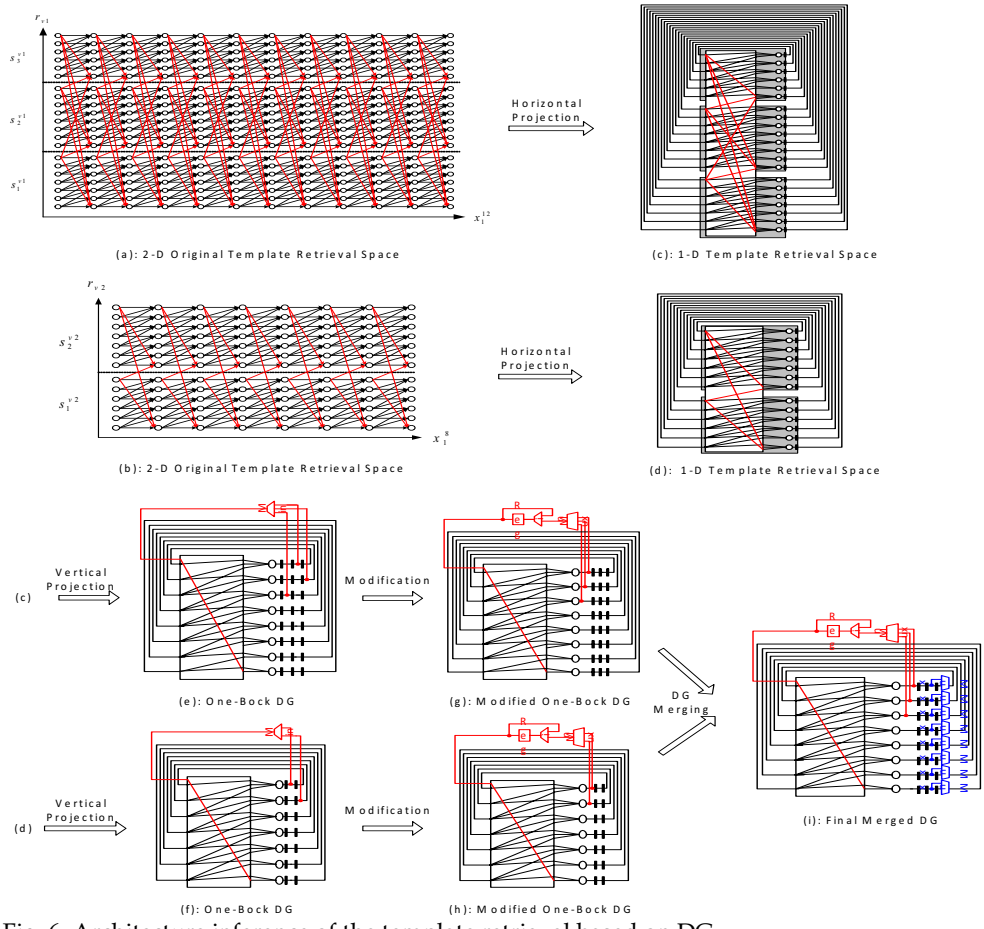

Fig. 6. Architecture inference of the template retrieval based on DG.

3.2.1 The Distortion Unit

Each node in the template retrieval DG associates with a local distortion. The local distortion between a frame of the template and a frame of the input speech is defined by

$$\text{Local distortion} = \sum_{i=0}^{P} |R_i - U_i|' \tag{20}$$

where R_i denotes the *i*-th cepstral coefficient in the *R*-th frame of the template, U_i represents the *i*-th cepstral coefficient in the *U*-th frame of the input speech, and *P* is the cepstrum order. The cepstral coefficients of the input speech and the templates are stored in RAM0 and ROM1, respectively. The distortion unit accesses the cepstral coefficients from the two memories, and then accumulates the distortion in register *R*.

3.2.2 The Processing Element

Each node in the DGs shown in Fig. 6(i) is implemented as a processing element. The architecture design of the PE is displayed in Fig. 7. After receiving all the accumulated distortions from the registers, the PE selects the minimum accumulated distortion, and then adds it to the local distortion $d(l,k,j)$. The result is the new accumulated distortion $d_A(l,k,j)$ associated with the current node. In addition to the accumulation process, the PE also generates the decision information that indicates the source node in a path transition. The use of the decision information is discussed together with the pattern extraction core later.

Based on the data path of the template retrieval core, a corresponding template retrieval controller is also developed. In addition to producing control signals, this controller also functions as a memory address generator for accessing the template speech features. The template retrieval core not only accumulates the minimum local distortion but also writes the decision information into the memory. The decision information for a node indicates its source node in a path transition. The best path can be obtained from the decision information, tracing it backward for extracting the hypothesised speech patterns. The design of the pattern extraction core is given in the next subsection.

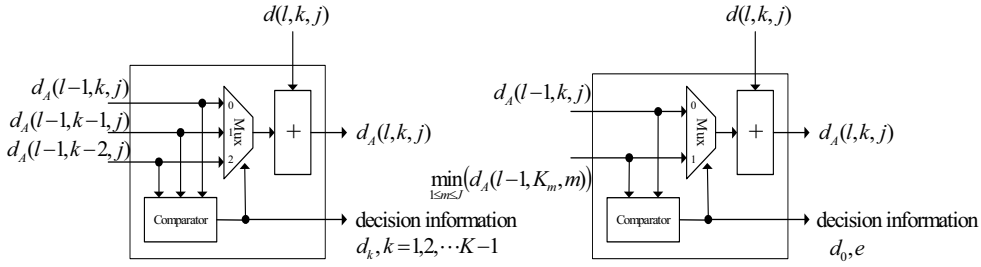

Fig. 7. Architecture of the processing element.

3.3 The Design of the Pattern Extraction Core

To extract the best pattern sequence, this work presents a block-node pattern extraction method. Figure 8 exemplifies the relationship between nodes and blocks. The block boundaries are the same as the pattern boundaries. This template retrieval space contains 24

blocks, numbered from 0 to 23. Blocks in the same row belong to the same pattern, and have the same number of nodes (frames). For example, blocks 2, 5, 8, …, 23 belong to the third pattern within this template. The pattern extraction method used in this work can be regarded as a two-level address decoding process. At the first level, the blocks are the addressing units. Each block consists of frame nodes, which become addressing units at the second level. If b denotes the block index and k denotes the local node index in a block, each node can also be referred to by the pair (b, k). For example, *node* 64 can be referred to as *node* (21, 1).

The decision word is generated for each block, and is denoted by

$$D \equiv \{d_0, d_1, d_2, ..., d_{K-1}, e\}, \tag{21}$$

where K is the largest number among all the block node ones, e is the pattern decision information from the first node of a block, and d_k, $0 \le k \le K-1$, is the decision information from the k-th node of a block. We use e to indicate the source pattern for an external (between-patterns) transition, and d_k to indicate the source node for an internal (within-patterns) transition. Assume that K_j denotes the number of nodes in the j-th pattern.

Pattern extraction starts from the end node, and recursively update the local node index and the block index to construct a best path in the template retrieval space. The block index b is updated by

$$b = \begin{cases} b - J, & \text{for internal } transition, \\ b - J + new\_e - old\_e, & \text{for external } transition, \end{cases} \tag{22}$$

where J is the number of patterns in this template, *new_e* is the current value of the decision information e, and *old_e* is the previous value of the decision information e.

As for the local node index k, it is updated by

$$k = \begin{cases} k, & \text{for internal transition with } d_k = 0, \\ k-1, & \text{for internal transition with } d_k = 1, \\ k-2, & \text{for internal transition with } d_k = 2, \\ K_e - 1, & \text{for external transition.} \end{cases} \tag{23}$$

Let us denote the best path end in the block by *end_block*, and its pattern by *end_pattern*. The flow chart of the algorithm is depicted in Fig. 9.

The block-node pattern extraction method is implemented as a finite state machine (FSM). Because template retrieval and pattern extraction are required for all templates, this design is based on a two-memory scheme. While the decision word is being written into RAM1, the back tracing is being performed in RAM2; and vice versa. This reduces the overall computation time of template retrieval and pattern extraction for all templates.

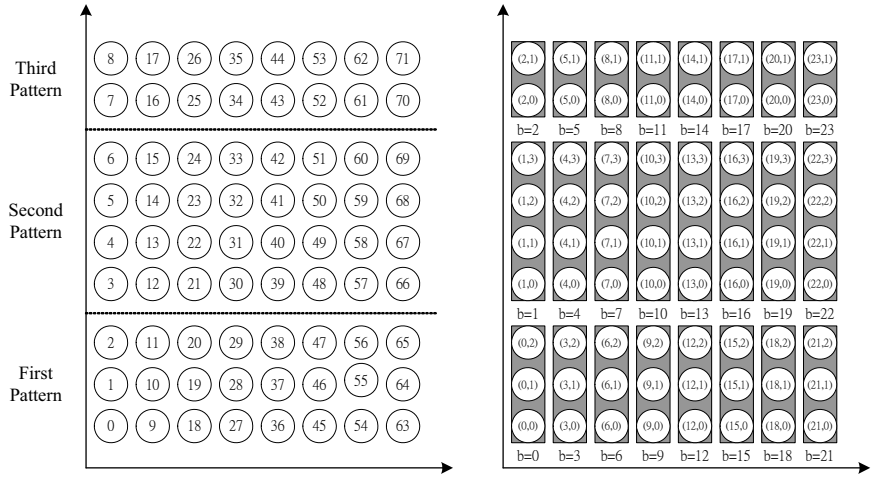

Fig. 8. Example illustrating the block-node addressing method.

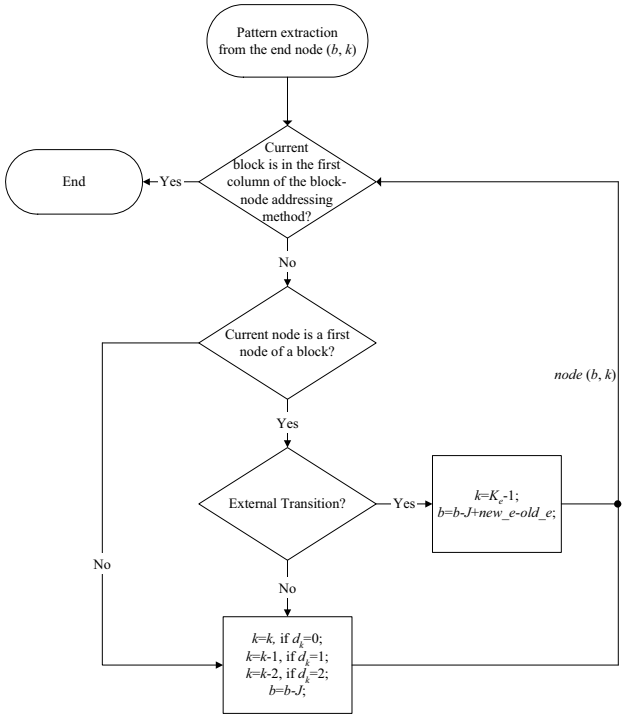

Fig. 9. Flow chart of the block-node pattern extraction.

3.4 The Design of the ADC/DAC

The last specific hardware core is ADC/DAC. In this part, we mainly present an efficient architecture to improve the conventional fully differential successive approximation ADC. The proposed architecture includes two optimal design solutions. First, in order to avoid the mismatch between positive and negative reference voltages, a single reference voltage (SRV) method that removes the negative reference voltage is developed by the distinct switched capacitor controlling algorithm. Figure 10 shows the implementation of the SRV algorithm in circuit. Second, a small area scheme for successive approximation register (SAR) is introduced. Comparing to the traditional successive approximation ADC, our proposed architecture is contributive to improve the area saving in SAR and the accuracy of ADC.

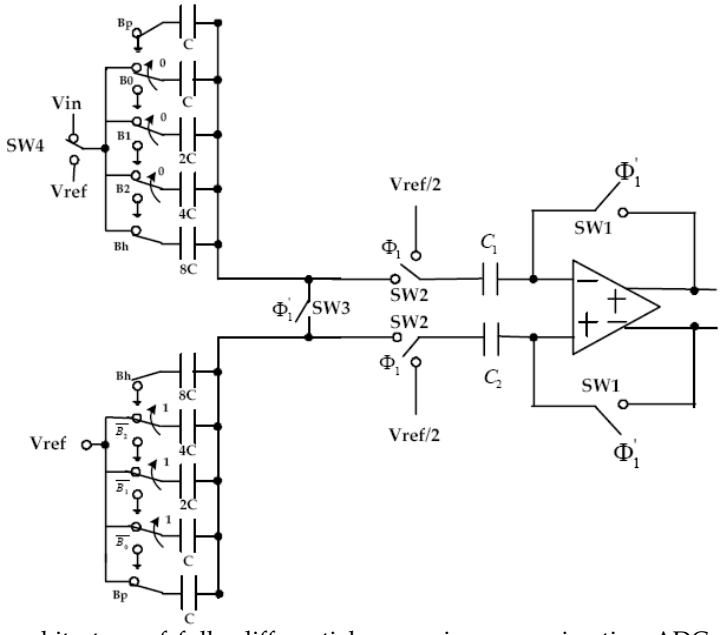

Fig. 10. The architecture of fully differential successive approximation ADC using single reference voltage.

To reduce the area for successive approximation register, this work presents a simplified non-redundant successive approximation register (SSAR). The basic architecture of the SSAR is a multiple input n-bit shift register, shown in Fig. 11. This multiple input register consists of a general D flip-flop and a multiplexer with two inputs. The function of SSAR is shown in Fig. 11. Suppose the "1" is the input token in Fig. 12. Whenever the SSAR is triggered, each register of SSAR must go through three modes: (1) when the token has not passed yet, the values of the registers are "0"; (1) when the token is staying at a certain register, its register value is changed to "1", and receives the result of comparator; (3) when the token has passed through, the value determined by the result of comparator is held until the whole conversion is done. The function can be implemented as below.

Therefore, when the value of k-th register of the SSAR is still "0", the Q_{k+1} selected by the multiplexer is connected to D_k. If the k-th register receives the token, the Q_k is changed to "1",

and the CMP would be selected. The result of comparator Z_k would be held in the k-th register until the whole conversion is done. The system can be implemented by the blocked CLK_k. The combination of all above implementations makes it possible to improve the accuracy and save the chip area.

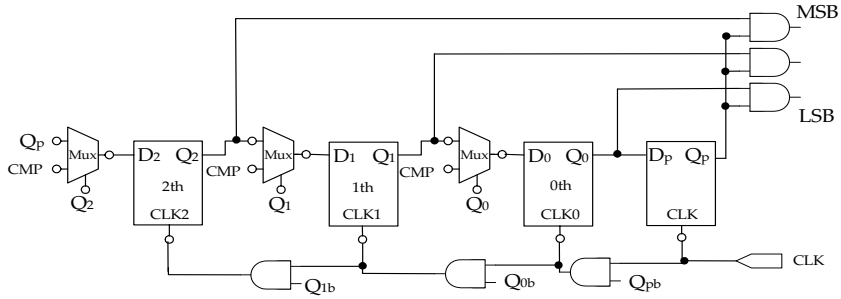

Fig. 11. The architecture of the simplified non-redundant SAR.

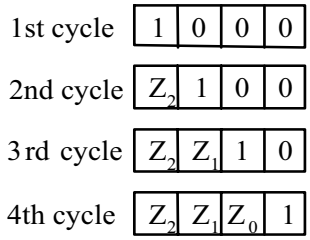

Fig. 12. The function of simplified non-redundant SAR.

4. Summary

Speech-to-speech machine translation is a prospective application of speech and language technology. This work presents an MTS based speech-to-speech translation system between Mandarin Chinese and English. The proposed MTS approach achieves about a 90% understandable translation rate on average. For a portable and real-time speech-to-speech translation system, this work also proposes the SOC realization. The architecture design is based on the semi-ASIC technique, which incorporates a cost efficient programmable core along with specific hardware accelerators: an LPC extraction core, a template retrieval core, and a pattern extraction core. Besides, the ADC/DAC are also included. A SRV-based fully differential successive approximation ADC with reduced SSAR is designed. These hardware cores construct a complete speech-to-speech translation SOC. The proposed SOC chip is the first one dedicated for speech-to-speech translation.

5. References

[1]A. Lavie *et al*., "JANUS III: speech-to-speech translation in multiple languages," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, Apr. 1997, pp. 99–102.

[2]W. Wahlster, *Verbmobil: Foundations of Speech-to-Speech Translation*. Berlin Heidelberg, New York: Springer-Verlag, 2000.

[3]H. Ney, S. Nießen, F. J. Och, H. Sawaf, C. Tillmann, and S. Vogel, "Algorithms for statistical translation of spoken language," *IEEE Trans. Speech and Audio Processing*, vol. 8, pp. 24-36, Jan. 2000.

[4]F. Casacuberta *et al*., "Speech-to-speech translation based on finite-state transducers," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing*, May 2001, pp. 613–616.

[5]F. Sugaya, T. Takezawa, A. Yokoo, and S. Yamamoto, "End-to-end evaluation in ATR-MATRIX: speech translation system between English and Japanese," in *Proc. 6th Eur. Conf. Speech Communication and Technology*, Sep. 1999, pp. 2431–2434.

[6]P. C. Ching and H. H. Chi, "ISIS: a trilingual conversational system with learning capabilities and combined interaction and delegation dialogs," in *Proc. National Conf. Man-Machine Speech Communications*, Nov. 2001, pp. 119–124.

[7]R. Isotani, K. Yamabana, S. Ando, K. Hanazawa, S. Ishikawa, T. Emori, H. Hattori, A. Okumura, and T. Watanabe, "An automatic speech translation system on PDAs for travel conversation," in *Proc. IEEE Int. Conf. Multimodal Interfaces*, Oct. 2002, pp. 211–216.

[8]A. Waibel, A. Badran, A. W. Black, R. Frederking, D. Gates, A. Lavie, L. Levin, K. Lenzo, L. M. Tomokiyo, J. Reichert, T. Schultz, D. Wallace, M. Woszczyna, and J. Zhang, "Speechalator: two-way speech-to-speech translation on a consumer PDA", in *Proc. European Conf. Speech Communication and Technology*, Sep. 2003, pp. 369–372.

[9]T. Watanabe, A. Okumura, S. Sakai, K. Yamabana, S. Doi, and K. Hanazawa, "An automatic interpretation system for travel conversation," in *Proc. Int. Conf. Spoken Language Processing*, Sep. 2000, pp. IV-444–IV-447.

[10]J. F. Wang, B. Z. Houg, and S. C. Lin, "A study for Chinese text to Taiwanese speech system," in *Proc. Int. Conf. Research on Computational Linguistics*, Aug. 1999, pp. 37–53.

[11]M. Simard, "Translation spotting for translation memories," in *Proc. HLT-NAACL Workshop on Building and Using Parallel Texts: Data Driven Machine Translation and Beyond*, May 2003, pp. 65–72.

[12]J. Véronis and P. Langlais, "Evaluation of parallel text alignment systems – the ARCADE project," in *Parallel Text Processing*, Dordrecht: Kluwer Academic, 2000, pp. 369–388.

[13]L. Rabiner and B. H. Juang, *Fundamentals of Speech Recognition*. Prentice-Hall, Inc., 1993.

[14]J. F. Wang, A. N. Suen, and C. K. Chieh, "A programmable application specific architecture for real-time speech recognition," in *Proc. of VLSI Design/CAD Symposium*, Aug. 1995, pp. 261–264.

[15]Y. Zhang, "Survey of current speech translation research," presented at Multilingual Speech-to-Speech Translation Seminar, Carnegie Mellon University, Pittsburgh, PA, 2003.

[16]Y. S. Lee and S. Roukos, "IBM Spoken Language Translation System Evaluation," in *Proc. INTERSPEECH2004 Workshop on Spoken Language Translation: Evaluation Campaign on Spoken Language Translation*, Oct. 2004, pp.39–46.

[17]L. Gu and Y. Q. Gao, "On Feature Selection in Maximum Entropy Approach to Statistical Concept-based Speech-to-Speech Translation," in *Proc. INTERSPEECH2004 Workshop on Spoken Language Translation: Evaluation Campaign on Spoken Language Translation*, Oct. 2004, pp.115–121.

[18]S. Nakamura, K. Markov, T. Jitsuhiro, J. S. Zhang, H. Yamamoto and G. Kikui, "Multi-Lingual Speech Recognition System for Speech-To-Speech Translation," in *Proc. INTERSPEECH2004 Workshop on Spoken Language Translation: Evaluation Campaign on Spoken Language Translation*, Oct. 2004, pp.146–154.

[19]K. Matsui, Y. Wakita, T. Konuma, K. Mizutani, M. Endo, and M. Murata, "An experimental multilingual speech translation system," in *Proc. ICMI-PUI*, 2001, pp. 1–4.

[20]S. Rossato, H. Blanchon, and L. Besacier, "Speech-to-speech translation system evaluation: Results for French for the Nespole! project first showcase," in *Proc. ICSLP*, 2002, pp. 1905–1908.

[21]F. Casacuberta, E. Vidal, and J. M. Vilar, "Architectures for speech-to-speech translation using finite-state models," in *Proc. ACL Workshop on Speech-to-Speech Translation: Algorithms and Systems*, 2002, pp. 39–44.

[22]H. Ney, "Speech translation: Coupling of recognition and translation," in *Proc. ICASSP*, 1999, pp. 517–520.

[23]J. F. Wang and S. C. Lin, and H.W. Yang, "Multiple-Translation Spotting for Mandarin-Taiwanese Speech-to-Speech Translation," *Int. Journal of Computational Linguistics and Chinese Language Processing*, vol.9, no.2, 2004, pp. 13-28.

[24]W. Verhelst and M. Roelands, "An overlap-add technique based on waveform similarity (WSOLA) for high quality time-scale modification of speech," in *Proc. ICASSP*, 1993, pp. 554–557.

[25]M. Demol, K. Struyve, W. Verhelst, H. Paulussen, P. Desmet, and P. Verhoeve, "Efficient non-uniform time-scaling of speech with WSOLA for CALL applications," presented at InSTIL/ICALL Symp. Computer Assisted Learning, Venice, Italy, 2004.

[26]H. G. Ilk and S. Tugac, "Channel and source considerations of a bit-rate reduction technique for a possible wireless communications system's performance enhancement," *IEEE Trans. Wireless Communications*, vol. 4, no. 1, pp. 93–99, Jan. 2005.

[27]E. T. Cornelius, *English 900*. Pace Group International Inc., 1999.

[28]B. E. Bagnell and M. Lee, *New Globe English Course on Travel*. New Globe Publishing Co., LTD, 1990.

[29]J. F. Wang and S. H. Chen, "Speech Enhancement Using Perception Wavelet Packet Decomposition and Teager Energy Operator", Journal of VLSI Signal Processing, Vol. 36 I: 2-3, pp. 125–139, Feb. 2004..

[30]J. F. Wang, C. H. Yang, and K. H. Chang, "Design of a Subspace Tracking Based Speech Enhancement System", in *Proc. IEEE TENCON 2004*, vol. 1, pp. 147–150..

[31]J. T. Chien and C. H. Huang, "Bayesian Learning of Speech Duration Models," *IEEE Trans. Speech and Audio Processing*, vol. 11 I:6, pp. 558–567, Nov. 2003.

[32]Jhing-Fa Wang, Jia-Ching Wang, Han-Chiang Chen, Tai-Lung Chen, Chin-Chan Chang, and Ming-Chi Shih, "Chip Design of Portable Speech Memopad Suitable for

Persons with Visual Disabilities," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 8, pp. 644-658, November 2002.

[33]J. Makhoul, "Linear prediction: a tutorial review," *Speech Analysis*, IEEE Press, New York, 1979.

[34]L. Y. Liu, J. F. Wang, J. Y. Lee, M. H. Sheu, and Y. L. Jeang, "An ASIC design for linear predictive coding of speech signals," in *Proc. Euro ASIC' 92*, 1992, pp.288-291.

[35]S. Furui, "Cepstral analysis technique for automatic speaker verification," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-29, no. 2, pp. 254-272, 1981.

VLSI

Edited by Zhongfeng Wang

ISBN 978-953-307-049-0
Hard cover, 456 pages
Publisher InTech
Published online 01, February, 2010
Published in print edition February, 2010

The process of Integrated Circuits (IC) started its era of VLSI (Very Large Scale Integration) in 1970's when thousands of transistors were integrated into one single chip. Nowadays we are able to integrate more than a billion transistors on a single chip. However, the term "VLSI" is still being used, though there was some effort to coin a new term ULSI (Ultra-Large Scale Integration) for fine distinctions many years ago. VLSI technology has brought tremendous benefits to our everyday life since its occurrence. VLSI circuits are used everywhere, real applications include microprocessors in a personal computer or workstation, chips in a graphic card, digital camera or camcorder, chips in a cell phone or a portable computing device, and embedded processors in an automobile, et al. VLSI covers many phases of design and fabrication of integrated circuits. For a commercial chip design, it involves system definition, VLSI architecture design and optimization, RTL (register transfer language) coding, (pre- and post-synthesis) simulation and verification, synthesis, place and route, timing analyses and timing closure, and multi-step semiconductor device fabrication including wafer processing, die preparation, IC packaging and testing, et al. As the process technology scales down, hundreds or even thousands of millions of transistors are integrated into one single chip. Hence, more and more complicated systems can be integrated into a single chip, the so-called System-on-chip (SoC), which brings to VLSI engineers ever increasingly challenges to master techniques in various phases of VLSI design. For modern SoC design, practical applications are usually speed hungry. For instance, Ethernet standard has evolved from 10Mbps to 10Gbps. Now the specification for 100Mbps Ethernet is on the way. On the other hand, with the popularity of wireless and portable computing devices, low power consumption has become extremely critical. To meet these contradicting requirements, VLSI designers have to perform optimizations at all levels of design. This book is intended to cover a wide range of VLSI design topics. The book can be roughly partitioned into four parts. Part I is mainly focused on algorithmic level and architectural level VLSI design and optimization for image and video signal processing systems. Part II addresses VLSI design optimizations for cryptography and error correction coding. Part III discusses general SoC design techniques as well as other application-specific VLSI design optimizations. The last part will cover generic nano-scale circuit-level design techniques.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821