A Data-Driven SoC System for Embedded Continuous Speech Recognition

Raymond Hoare, Kshitij Gupta, Jeffrey Schuster
University of Pittsburgh, Pittsburgh, PA, 15233
Phone: 412-624-4706
Email Addresses: {hoare, jws52, ksg3}@pitt.edu

Abstract

In this paper we present a SoC system able to perform Small-Vocabulary Automatic Speech Recognition (SVASR) based on Hidden-Markov Model (HMM) recognition techniques. Through in-depth analysis of the data-flow within the SPHINX 3 software [1], we create an efficient single-chip architecture tailored to the specific computational needs of a the system. By creating a token-passing scheme to control the work-load within the system the on-chip resources as well as the complexity of the global control required can be minimized, while the bandwidth usage can be maximized, creating a stream-lined FPGA architecture able to evaluate small vocabularies in real time.

Introduction

Many of today's state-of-the-art ASR systems rely on the use of HMM evaluation to calculate the probabilities associated with a set of phonetic units, based on inputs supplied from Gaussian probability evaluations [2]. These systems are able to achieve accuracy rates upwards of 95% on a dictionaries greater than 1,000 words, however, this accuracy comes at the expense of needing to evaluate of hundreds of thousands of Gaussian probabilities resulting in execution times up to 10x real-time [3]. The computationally intensive nature of this problem as well as the large amount of memory bandwidth required both make the use of embedded systems very difficult.

These types of systems can be broken down into four major components: the Feature Extractor (FE), the Acoustic Modeler (AM), the Phoneme Evaluator (PE), and the Word Modeler (WM), each presenting its own unique problems. FE involves decomposing the incoming speech in to its frequency content via either the Fast Fourier Transform of the Discrete Cosine Transform. These operations can be performed on most currently available DSP devices and will therefore not be considered within the scope of this research. The AM is the front-end of the recognition system, responsible for evaluating the inputs received from the DSP unit, generally Cepstral coefficients and their derivatives, with respect to a database of known Gaussian probabilities and producing a normalized set of scores for the individual sub-phonetic units, senones, represented in the database. The PE associates groups of senones into to HMMs representing the phonetic units allowable in the systems dictionary. The WM uses a tree-based structure to string phonemes together into words based on the sequences defined in the system dictionary.

The architecture presented in this paper utilizes shared memories between the stages to create three different task-specific blocks connected in a cascade fashion to achieve the highest amount of performance possible within each stage. Further, by implementing a token-passing scheme, similar to[4], between the PE and WM blocks, an architecture was created in which the active data in the system determines the scheduling of the systems work-load. Figure 1 shows the basic data-flow through the system with shared RAM blocks between each phase and FIFOs between the 2nd and 3rd stages to implement the token passing scheme.

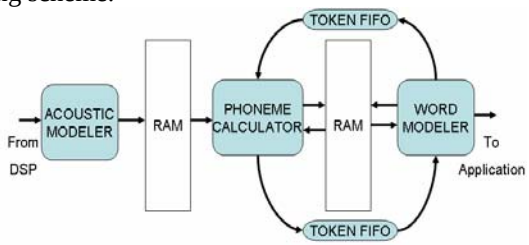

Figure 1: Block Diagram of Single-Chip ASR Architecture

Acoustic Modeler

AM is responsible for relating the data received from the feature extractor to the data found in the systems dictionary and can account for over 70% to 95% [5] of the computational effort in modern HMM-based ASR systems. Every senone, in the database is made-up of 8 components, each one representing a 39 dimensional multi-variant Gaussian probability distribution. The components of a senone are log-added [6] to one another to obtain the probability of having observed the given senone. Based on the 1000-word RM1 dictionary [7], our system utilizes 1935 senones, requiring over 2.5 Million Floating-Point operations to calculate scores for every senone. For any practical system, running at real-time, these calculations become the critical path and need to be done as efficiently as possible.

During analysis of the SPHINX 3 system it was observed that not all of the senones in the database actually needs to be evaluated all the time. Further, on average 2 of the 8 components of a given senone contribute well over 80% of the senone score, leading to the ability to save 75% of the computations per senone with negligible reduction in accuracy. To obtain this reduced set of components a SubVector-Quantization, algorithm was implemented with an F_{MAX} of 177MHz post place-and-route.

By performing an in-depth analysis of the calculations being performed it was also found that the computationally intensive floating-point Gaussian probability calculations found in the SPHINX 3 system could be replaced with fixed-point calculations while only introducing errors on the order of 10^{-4}. The ability to use fixed instead of floats, allowed for the design of a 22 stage fully pipelined acoustic modeler running at over 250MHz post place-and-route on a Vertix-4 SX35-10.

When the entire AM was synthesized it utilized only 8% of the available DSP blocks, 15% of the CLB slices, and 27% of the available block RAMs, with a post place-and-route speed of 125MHz.

Phoneme Evaluator

The core of the PE consists of a HMM evaluation based on a basic 3-state Bakis topology. By pipe-lining the necessary calculations along with the RAM reads / writes a simple 10-stage pipe can be derived using only combinational logic to perform the necessary mathematics that synthesizes at 200MHz on a Virtex-4 chip, pre-place-and-route.

PE is also responsible for pruning the active data in the system based on a relative-beam pruning methodology. The output scores for each HMM are compared to the beams and the active token is routed to its appropriate FIFO, either Dead or Valid, for processing by the Word Modeler. The pruning process only adds 4 additional stages to the pipe-line and the entire Phoneme Calculator was synthesized at 124MHz on the Virtex-4 and consumed only 8% of the CLB Slices and 9.4% of the Block RAMs.

Word Modeler

In modern ASR, WM uses tree-based searching techniques to seek out the most likely strings of phonemes, and subsequently words, observed over time. To maintain similarity to the SPHINX 3 software, this search was implemented using a flat-lexical tree search algorithm. Traversing this tree can be thought of as the evaluation of a link-list of link-lists, where each node in the tree defines the beginning of a link-list, and the number of possible branches from that node to its adjacent nodes defines the number of elements in the list. Since the phonetic structure of the words is known well in advance all of this information can be placed into ROMs before run-time, allowing the majority of the WM operations to be simple ROM reads, wherein the data read from one ROM becomes the address into the next.

While the WM is responsible for both cleaning up the data pruned out by the phoneme modeler, and processing the data that remains active, the dead token clean-up simply pulls a token from the Dead FIFO and overwrites the RAM contents found at the address specified by that token and therefore has an F_{MAX} equal to that of the target technology. The valid evaluation stage is significantly more complex and requires routines to evaluate the link-lists as well as determine the validity of the output of a given tree in the lexicon. Despite its added complexity, the valid evaluation block consumes less than 1% of the resources on a Virtex-4 with a pre-place-and-route F_{MAX} of 216MHz.

Conclusions

This research has proven the ability to implement HMM-based ASR on embedded systems for use in real-time speech recognition and transcription tasks. Through implementation of a token passing scheme the amount of work done during recognition is minimized resulting in the consumption of fewer resources, ideal for small-scale embedded systems.

Our final system performed recognition on a 17 word dictionary utilizing 451 nodes split between five different trees, and achieved a post-place-and-route speed of 112MHz on the Vertix-4. The system only consumed 24% of the slices available on the chip, but due to its heavy memory requirements, consumed 61% of the available Block RAMs. Table 1 summarizes the specifications for each of the stages of the design.

Table 1: Summary of Synthesis Results for Vertix-4 SX35-10

	F_{MAX} (MHz)	CLB Slices (%)	Block RAMs (%)
AM	125	15	27
PE	124	8	10
WM	216	1	0
System	112	25	61

References

[1] P. Placeway, *et al*, "The 1996 HUB-4 Sphinx-3 System", *Proc. DARPA Speech Recognition Workshop*, Feb. 1997.

[2] K.K. Agaram, S.W. Keckler, D. Burger, "Characterizing the SPHINX Speech Recognition System", University of Texas at Austin, Department of Computer Sciences, Technical Report TR2001-18, January 2001.

[3] M. Ravishankar, *et al*, "The 1999 CMU 10X Real Time Broadcast News Transcription System", *Proc. DARPA Workshop on Automatic Transcription of Broadcast News*, Washington DC, May 2000

[4] S.J. Young, et al. "Token Passing: a Simple Conceptual Model for Connected Speech Recognition Systems", Cambridge University Engineering Department. July 1989.

[5] J. Nouza, "Feature Selection Methods for Hidden Markov Model-Based Speech Recognition," *Proc. International. Conference on Pattern Recognition*, 1996, vol. 2, pp. 186.190.

[6] X. Li & J. Blimes, "Feature Pruning in Likelihood Evaluation of HMM-Based Speech Recognition", Unversity of Washington, 2003.

[7] Linguistic Data Consortium. University of Pennsylvania. 27 Oct. 2004<www.ldc.upenn.edu>