DESIGN AND IMPLEMENTATION OF SPEECH RECOGNITION ON A SOFTCORE BASED FPGA

Hyunjin Lim, Kisun You and Wonyong Sung

School of Electrical Engineering
Seoul National University
Kwanak-gu, Seoul 151-742 Korea
Phone: +82-2-880-9372, Fax: +82-2-882-4656
Email: {hjlim, ksyou, wysung}@dsp.snu.ac.kr

ABSTRACT

A speech recognition algorithm is implemented on a Xilinx's MicroBlaze softcore based FPGA platform. When compared to programmable DSP or RISC processor based design, this approach not only can support large vocabulary speech recognition but also leads to a multi-channel system implementation utilizing an off-the-shelf FPGA device. The design space is explored in a few ways; firstly by configuring the datapath of the MicroBlaze softcore and secondly by adding a custom hardware block to speed up the emission probability computation. The hardware block designed for emission probability computation incorporates the BRAM where the reference data is accessed directly thereby reduces the overhead of transferring data from the processor to the hardware block.

1. INTRODUCTION

An FPGA based design has been acquiring a lot of attentions in the domain of hardware design, but not that of software based embedded system design [1]. As a result, most of the speech recognition algorithms have been implemented on programmable DSP (Digital Signal Processor) or RISC processor based architectures [2][3]. However, the pure software based approach is not efficient for high performance speech recognition applications mainly because it requires an excessive number of instruction cycles.

In this paper, the speech recognition algorithm is implemented on an FPGA platform containing a softcore, which supports program, especially C language, based application development. Obviously, it is needed to add special functional blocks for achieving the speed up close to or even beyond the capability of pure software based solution.

The softcore based solution for FPGA has been emerging recently, and Xilinx MicroBlaze and Altera Nios are of the popular examples [4]. The softcore based solution has several advantages over a pure hardware based design as well as the hard processor based design. Firstly, it supports a program based application development. Thus, a complex application such as speech recognition can be implemented quite quickly. Secondly, it is possible to equip special datapath for assembling a high performance instruction. Although this feature has yet some limitations, it is not difficult to add a hardware multiplier or even a floating-point unit to the softcore processor. Thirdly, the softcore features does not require any silicon area when it is not needed, while the hard processor based approach always consumes the area. It is also possible to implement multiple processors as long as the chip capacity is allowed. Finally, it is natural to add a hardware block for accelerating the execution of certain functional blocks. As a result, this approach seems very adequate not only for early time-to-market but also for high performance system design beyond the capability of a program based system design [4].

This paper is organized in the following manner. Section 2 briefly discusses the Xilinx's MicroBlaze softcore processor and the speech recognition algorithm. In section 3, the effects of configuring a datapath for feature extraction are explained. Section 4 gives the design and performance evaluation of the hardware based emission probability computation block. Finally in section 5, concluding remarks are made.

2. OVERVIEW OF SYSTEM AND ALGORITHM

Xilinx's ML402 evaluation board which combines XC4VSX35 of Virtex-4 FPGA family, provides an environment for designing a reconfigurable system based on the MicroBlaze softcore processor.

2.1 MicroBlaze Softcore Processor

MicroBlaze is a softcore processor based on a traditional 32-bit Harvard RISC architecture. Unlike any other off-the-shelf hard processors which are actually implemented in the

FPGA at a transistor level, the soft core is an IP block written in HDL and is implemented in the free resources of an FPGA [5]. Therefore it is configurable by choosing the components that would be included in the system. According to the type of applications, MicroBlaze processor can be configured in several ways to save power or area.

MicroBlaze processor provides four bus connections, namely the Local Memory Bus (LMB), the On-chip Peripheral Bus (OPB), the Fast Simplex Link (FSL) and the Xilinx Cachelink (XCL). The LMB is a dedicated bus for MicroBlaze and on-chip block RAM connection. The OPB is a CoreConnect IBM standard bus and gives the capability to connect variety of available IP blocks and peripherals. The FSL has a FIFO-based interface and it provides a connection between a custom hardware to assist particular application [6] [7]. Lastly the XCL interface provides a link between MicroBlaze processor and data and instruction caches.

2.2 Speech Recognition

Speech recognizer is a widely used application especially in the field of embedded systems such as PDA's and mobile phones. Due to a heavy arithmetic computation nature of the speech recognition, it has been implemented on a DSP or RISC-DSP systems. However, it takes a lot of consideration to implement the speech recognition using a low-cost embedded RISC CPU, especially in reducing arithmetic operations due to the lack of hardware support. A speaker independent continuous speech recognition employing context dependent HMM is used for the implementation. The recognition process mainly consists of three parts, feature extraction, emission probability computation and Viterbi beam search [8].

At the feature extraction stage, a 13th–order MFCC (Mel Frequency Cepstral Coefficients) and corresponding delta and acceleration coefficients are computed for each frame, which constitute a 39-dimensional feature vector. The most computation intensive part for MFCC computation is a 512-point FFT, which requires many add and multiply operations.

The emission probability computation employs the phoneme model that is only affected by its neighbor phonemes within the word boundary. Each phoneme model consists of three states and each state has transition and output probability densities modeled by the Gaussian mixture. The models were trained with TIMIT corpus, and parameter tying was applied in the training phase [9]. The total number of Gaussian mixtures in the trained models was about 4000.

In the emission probability computation stage, evaluation of the Gaussian mixture using the trained and feature vectors. The Gaussian log-probability for mixture n is given by,

$$\log(b^n(O_t)) = K + \sum_{k=1}^{39} (O_{tk} - \mu_k^{\,n})^2 \sigma_k^{\,n^{-2}}, \quad (1)$$

where O represents the feature vector and μ, σ are mean and variance of the mixture n respectively. The emission probability of the state with mixture N is given by,

$$\log(b^1(O_t) + b^2(O_t) + \cdots + b^N(O_t)). \quad (2)$$

In the Viterbi search, the beam pruning technique is adopted to reduce the search space so that the number of emission probability computation is reduced. The states with a low accumulated log-likelihood are pruned and are not used for the next emission probability computation. Although the pruning saves many operations, especially for a large size recognition task, it can lower the recognition accuracy.

Fig. 1 shows the cycle requirements for 100, 200, 400 and 800 word tasks simulated on an ARM7 RISC processor.

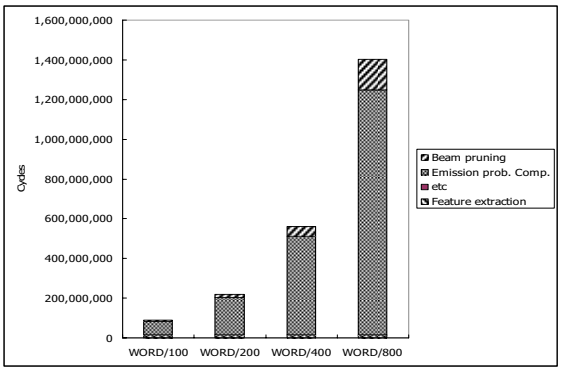

Figure 1. MIPS requirements for various recognitions tasks

3. DATAPATH SELECTION FOR FEATURE EXTRACTION

As it is shown in Fig. 1, the computation requirement for the feature extraction is constant, around 10 MIPS (Million Instructions Per Second) regardless of the recognition word size. The feature extraction part is implemented on a platform based on the MicroBlaze processor. Note that the basic MicroBlaze architecture does not contain a hardware multiplier, but it can accommodate a hardware multiplier by a simple configuration. The execution time of handling a single frame of input speech sample was measured. To show the effect of additional hardware block, the execution time is measured for the case of MicroBlaze with a hardware multiplier support and the case with the absence of hardware multiplier. Also the execution time on ARM7TDMI is measured for the general comparison purpose. Fig. 2 shows the execution time of handling a single frame of input speech sample on ARM7TDMI, MicroBlaze without hardware multiplier support and

MicroBlaze with an additional hardware multiplier. The ARM7TDMI is operating at 60 MHz and the MicroBlaze is working at 100 MHz.

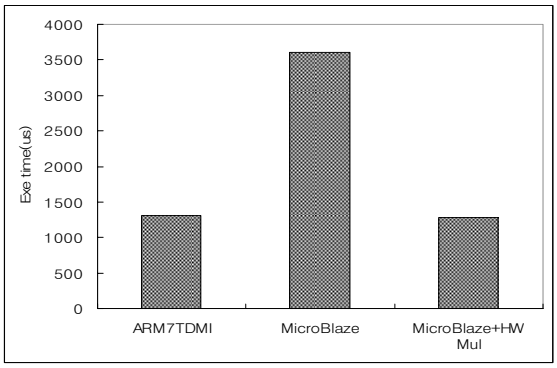

Figure 2. The execution time of the feature extraction part of speech recognition

As it can be seen from the figure, ARM7TDMI and the MicroBlaze system with a hardware multiplier support give nearly equal performance. The execution times were measure to be 1280.9us and 1307.183us for ARM7TDMI and MicroBlaze with hardware multiplier respectively. Note that the instruction efficiency of MicroBlaze is a little bit poorer than the ARM7TDMI because of the architectural restriction and compiler differences.

Also Fig 2 shows that there is quite a good chance of performance improvement just with the addition of a hardware multiplier. The execution time for MicroBlaze without the hardware multiplier support was measured to be 3605.3us. The addition of hardware multiplier brought a speedup of about 180%. With the absence of hardware multiplier, MicroBlaze computes multiplication with software library routines which is rather time consuming. Since the additional hardware resource for adding hardware multiplier is not very demanding as will be discussed later, it is logical to add a hardware multiplier in the system design.

4. HARDWARE BLOCK FOR EMISSION PROBABILITY COMPUTAITON

As shown in Fig. 1, the emission probability computation is increasing rapidly as the input task set increases. Thus, it is inevitable for a high performance system design to add a hardware block for a speedup. Note that the involved arithmetic operations for the emission probability computation as shown in Eq. (1) is relatively simple and can easily be implemented using pipelined hardware arithmetic units.

The custom hardware developed for computing the Gaussian log-probability for mixture *n* is illustrated in Fig. 3. This hardware block contains the datapath for the Gaussian log-probability computation. In addition, the feature vector storage is implemented. The feature vector is transferred to the hardware units for each frame once, so it does not consume many cycles. Note that the amount of the model data for triphone based speech recognition adopted in this paper is between 160 KB and 447 KB according to the size of recognition task. Thus, in this experiment the model data is stored at the external DDR. However, if the model data is stored at the internal memory, the overall design becomes simpler and the performance would be better.

In the implementation of 50 words task without any hardware block, the number of the core cycles for computing one model vector with the feature vector, which corresponds to the computation of Eq. (1) is 2920 clock cycles. With the architecture shown in Fig. 3, it is found that the core cycle is not much reduced and becomes 1349 clock cycles. Note that the delay of delivering the model data through the MicroBlaze and FSL is the major bottleneck for this architecture.

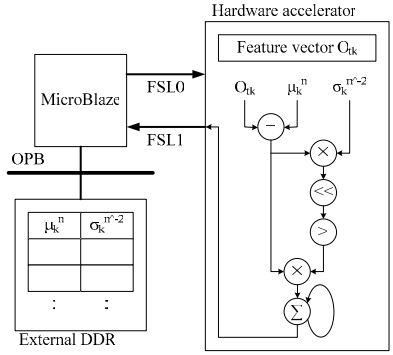

Figure 3. Hardware block for emission probability computation

The architecture shown in Fig. 4 is designed to load the model vectors directly from the internal BRAMs. Note that the access of model vector needs 39 32-bit data read cycles, but the access can utilize the fast read mode of the BRAM. The total required BRAM size to hold necessary parameters is about 160 KB.

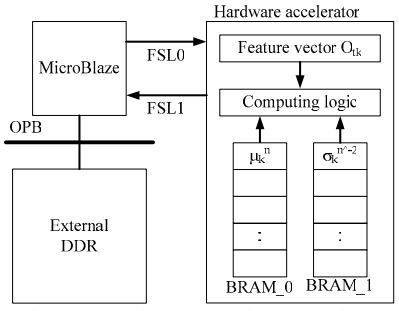

Figure 4. Hardware accelerator with internal BRAM

In addition, the data load can overlap with the computation. As a result, the number of the core cycles is reduced to 47

clock cycles. In this implementation, BRAM is connected to 32bit bus and needs 1 clock cycle in the fast read mode.

The above implementation results with the hardware block in Fig. 4 shows that the overall system becomes memory bound architecture. Thus, the performance can be improved much if the model data is stored in the internal BRAM. Fig. 5 shows the total execution cycles needed for the recognition for 50 words task. As it can be seen from the figure, the execution cycles for the emission probability computation has been reduced about 6.5 times which resulted in about 2 times reduction in the total execution cycles when the hardware accelerator with BRAM is adopted. As it was shown in the Fig. 1, the overhead of emission probability computation increases as the size of recognition task become larger. Therefore the effect of adopting the proposed architecture would yield further improvement.

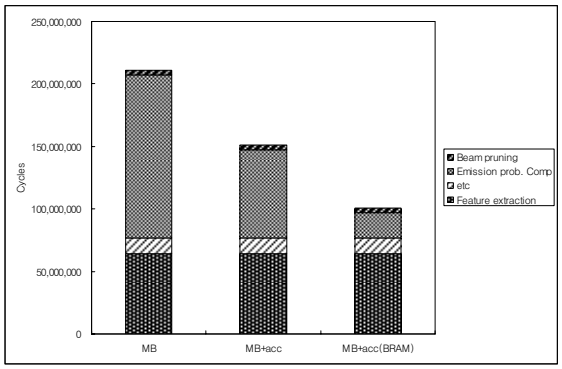

Figure 5. Total execution cycles of recognition on various implementations

The hardware resources for the implementation are summarized in Table 1. The results show that the increased hardware resource requirements are not excessive when compared to the basic MicroBlaze core. Note that the XC4VSX35 device, which is one of the Virtex-4 SX family, contains 15,360 slices, 30,720 slice FFs, 30,720 4-input LUTs, and 192 DSP-48 blocks. Thus, the hardware occupation ratio of each block is less than 10% of the XC4VSX35 device as shown at the parentheses in Table 1.

Table 1. Hardware resources.

	MicroBlaze w.o. MUL	MicroBlaze with MUL	Hardware Block
Slices	1117 (7%)	1130 (7%)	469 (3%)
Slice Flip Flops	725 (2%)	744 (2%)	778 (2%)
4 input LUTs.	1405 (4%)	1410 (4%)	169 (5%)
DSP-48s	0 (0%)	3 (1%)	2 (1%)

5. CONCLUDING REMARKS

This paper evaluated the design space of implementing speech recognition algorithm on the MicroBlaze softcore processor based FPGA. The computational requirements for speech recognition for different word size are analyzed, and optimized architecture design for real-time implementation is conducted. Especially, the hardware for accelerating the emission probability computation is designed, where it is shown the algorithm needs direct memory accesses.

The implementation results show that the FPGA softcore based system is not only programmable and flexible but also can execute large vocabulary speech recognition in real time. The synthesized results also show that more than 10 channels of speech recognition circuit can be implemented on an off-the-shelf FPGA device.

6. ACKNOWLEDGMENTS

This study was supported by the Brain Korea 21 Project of Korean Ministry of Education in 2005.

7. REFERENCES

[1] Xilinx: "Virtex-4 Family Overview," http://direct.xilinx.com/bvdocs/publications/ds112.pdf.

[2] Y. Gong and Y.-H. Kao, "Implementing a high accuracy speaker-independent continuous speech recognizer on a fixed-point DSP," *in Proc. ICASSP*, pp. 3686 – 3689, 2000,.

[3] S. Ryu, Y. Lee and W. Sung, "Implementation of Speech Recognition Algorithm for A 32-bit CPU Based Portable Device," *in Proc. ICCE*, pp. 240-241, 2002.

[4] R. Lysecky and F. Vahid, "A study of the speedups and competitiveness of FPGA soft processor cores using dynamic hardware/software partitioning," *in Proc. DATE*, pp. 18-23, 2005.

[5] Xilinx UG081: "MicroBlaze Processor Reference Guide," http://www.xilinx.com/ise/embedded/mb_ref_guide.pdf.

[6] M. Ouellette and D. Connors, "Analysis of Hardware Acceleration in Reconfigurable Embedded System," *in Proc. IPDPS*, pp. 168a-168a, 2005.

[7] Xilinx: XAPP529: "Connecting Customized IP to the MicroBlaze Soft Processor Using the Fast Simplex Link," http://www.xilinx.com/bvdocs/appnotes/xapp529.pdf.

[8] X. Huang, A. Acero and H. Hon, *Spoken Language Processing*, Prentice Hall, 2001.

[9] S. Young et. Al, *The HTK Book ver. 3.0*, Cambridge University, 2000.