

Computed Tomography (CT) Scan Image Reconstruction on the SRC-7

David Pointer
SRC Computers, Inc.
dpointer@srccomputers.com

Abstract

The computer algorithm used to reconstruct CT scanner images is very computationally intensive, which forces medical equipment manufacturers to strike a balance among image resolution, image generation time, and system cost when designing a new CT scanner. Cost-effective performance acceleration for image reconstruction would allow manufacturers to generate higher resolution CT images while maintaining the required balance in their system design. This paper describes the results of an early development effort on the SRC-7 MAPstation™ which yielded a 29x performance increase in CT image reconstruction.

1. Introduction

Computed Tomography (CT) scanning uses special x-ray equipment to generate multiple views of the inside of the body. These multiple x-ray views are reconstructed into cross-sectional images of the body using computer systems [1]. Radiologists use CT scan imagery to more easily diagnose medical problems such as some cancers, cardiovascular disease, trauma, and musculoskeletal disorders.

The computer algorithms used to reconstruct CT scanner images are very computationally intensive, which forces medical equipment manufacturers to strike a balance between image resolution, image generation time, and system cost when designing a new CT scanner. Higher image resolution provides more visual diagnostic details to the radiologist, but it takes longer to generate the images. It also requires more processing power, which adds significantly to the cost of the CT scan equipment. The current generation CT scan equipment is not the very best equipment current technology could provide; it is a balance of “good-enough” image resolution produced in a reasonable time at a cost hospitals can afford. Discussions with medical industry representatives indicate that the current result of this balancing act

yields 512x512 CT image resolution reconstructed in five minutes.

What would a radiologist see if they could get a thousand times more detail in a CT scan of a person’s heart? What if a high-resolution 3D CT scan could be generated in *real-time* for a surgeon at work, without needing several roomfuls of computers? What would hospital management say if they could obtain faster, higher-resolution CT scan equipment at the same or less cost than equipment available today?

This paper presents a single Altera Stratix II FPGA floating-point implementation of a 165 detector, 180 view reconstruction of a 1024x1024 pixel 2D image.

2. Prior Work

A simulation-based paper [2] describing a 32-bit integer backprojection FPGA implementation for a 512 detector 165 view CT system reconstructing a 256x256 pixel 2D image reports a 23x performance improvement. The same paper reports simulated performance improvements ranging from 11x to 44x for various ATI and nVidia GPUs for a floating-point implementation using the same CT and image parameters.

A hardware-based paper [3] describing a fixed-point FPGA implementation for a 1024 detector 1024 view CT system reconstructing a 512x512 pixel 2D image reports a 112x performance improvement.

3. CT scan image reconstruction

At the core of any CT scan image reconstruction is a computer algorithm called Filtered Backprojection (FBP) (Figure 1) [4]. Each of the hundreds of x-ray image data sets obtained by the CT scanner is filtered to prepare them for the backprojection step. Backprojection is nothing more than adding each filtered x-ray image data set’s contribution into each pixel of the final image reconstruction. Each x-ray view data set consists of hundreds of floating point numbers, and there are

hundreds of these data sets. In a high-resolution image, there are millions to tens of millions of pixels. It is easy to see why summing hundreds of large data sets into millions of pixels is a very time-intensive operation which only gets worse as the image resolution increases.

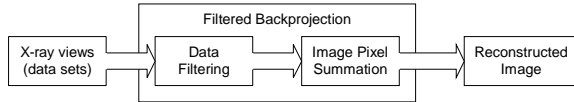


Figure 1 - CT Scan Image Reconstruction

Because of the need to balance image resolution, image generation time and cost in CT scan systems, much work has already been completed to make the FBP more computationally efficient. It turns out that the filtering step is extremely time-intensive if the x-ray view image data is left in its natural spatial coordinate system. However, if the image data sets are translated into the frequency domain, the filtering operation becomes trivial. Figure 2 shows the state of the art in computationally efficient FBP algorithm implementation [5]. The x-ray view data sets are translated into the frequency domain through the application of a Fast Fourier Transform (FFT) algorithm. The frequency domain data are passed through a convolution step, which is nothing more than a simple multiplication operation on all x-ray view data points. These data are then translated back into the spatial domain through the use of an inverse FFT operation. The final image pixel summations step remains unchanged.

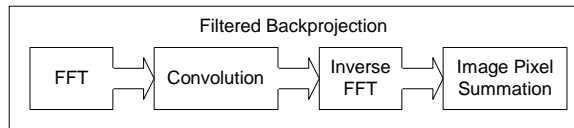


Figure 2 - Computationally efficient Filtered Backprojection

4. Image reconstruction implementation

SRC's IMPLICIT+EXPLICIT™ Architecture [6] is well suited to accelerating CT scan image reconstruction. In the simplest SRC-7 [7] system implementation (Figure 3), a microprocessor is paired with a Series H MAP® processor [8]. The system microprocessor provides data input and displays the final image using a commodity graphics card. The MAP processor contains an instantiation of the FBP algorithm. These two processors working together

achieve a 29x performance boost over the 3.0 gigahertz 64-bit Xeon microprocessor working alone.

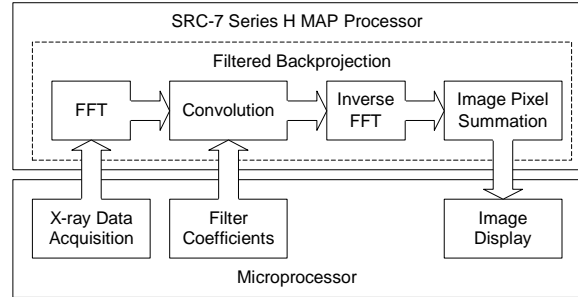


Figure 3 - CT image reconstruction on the SRC-7

4.1 CTsim application

A freely available open source application, Ctsim [9], was used as the target application for this implementation. CT scanner manufacturers fix all equipment parameters (detectors, geometries, number of scans, image resolution, etc.) at design time. Dr. Kevin Rosenberg M.D. wanted an environment in which he could vary any CT scan equipment parameter or algorithm then analyze the results. To this end, Dr. Rosenberg developed CTsim, a CT scanner simulator and analysis software package. CTsim uses the computationally efficient FBP shown in Figure 2. For high-performance FFT calculation, CTsim also uses the efficient fftw [10] library to perform forward and reverse FFT operations.

This implementation of image reconstruction on the SRC-7 replaces the FBP portion in CTsim with a MAP routine. All scanner view data obtained by CTsim are input to the MAP routine, all image data generated by the MAP routine are passed back to CTsim for display.

4.2 MAP FBP overview

This implementation of the MAP Filtered Backprojection function is shown in Figure 4. Data are obtained from CTsim and transferred to the MAP function during the initialization step. Then, for all views of an object: the projection data are filtered, eight duplicates of the filtered projection are built, and all pixels are reconstructed for this projection view, eight pixels at a time. After all projection views have been processed, the final reconstructed image data are transferred back to CTsim running on the CPU for visual display.

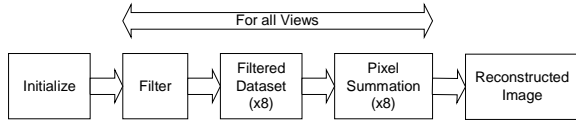


Figure 4 - MAP FBP implementation

4.3 MAP initialization

CTsim generates image data by simulating a multiple view scan of a standard CT phantom object, a Herman Head [11] (Figure 5). Unlike commercial CT equipment, scan parameters are programmable in CTsim. This implementation used 165 detectors, 180 object views, and one sample per detector. These settings generate 180 data sets of 165 floating point numbers each as projection data called a “sinogram”. (Figure 6). CTsim expands these data sets to 180 sets of 1024 words of complex floating point data. These data are transferred to the MAP routine as part of the MAP initialization.

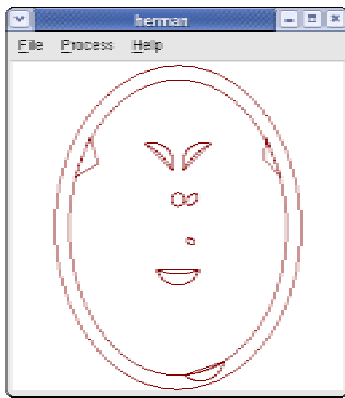


Figure 5 - Standard Herman Head phantom

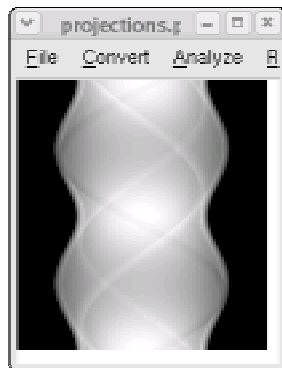


Figure 6 - Projections (sinogram)

A CT scanner obtains projections of an object by taking x-rays at multiple views around an object. In order to reconstruct a final image from these multiple views, the geometry for each projection’s data set must be taken into account during reconstruction. In this implementation CTsim generates 3 times 180 sets of floating point data representing the X, Y, and initial view angles for each view’s data set. These data are transferred to the MAP function as part of the initialization step.

For the filter step in FBP, CTsim generates a set of 1024 floating point filter coefficients. These coefficients are transferred to the MAP as part of the MAP initialization.

This FBP implementation in the MAP processor uses a programmable FFT (cfft_fp32) macro from SRC’s image processing library. Since this macro has a programmable point size, it requires an FFT twiddle table as input. This FFT twiddle table is generated for 1024 points and transferred to the MAP as part of the MAP initialization.

The measured time for this initialization step, including all data movement, was 3.70 milliseconds, or 2.2% of the MAP processor’s total execution time.

4.4 MAP filter

The filter implementation for the MAP routine is shown in Figure 7. All data sets are already stored in four banks of the MAP’s On-Board Memory (OBM). Filter coefficients and FFT twiddle factors are stored in FPGA internal memory arrays. The FFT macro (cfft_fp32) is instantiated once and used for both the forward and inverse FFT operation. Note that the FFT macro operates on four complex floating-point data words per FPGA clock.

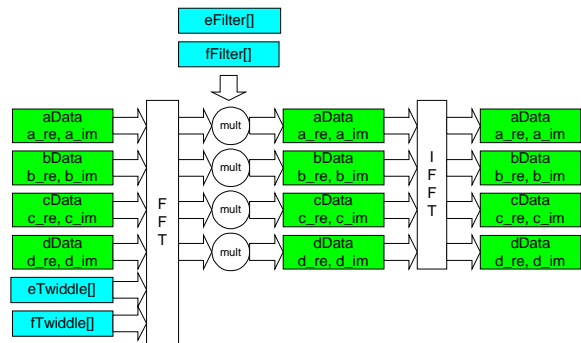


Figure 7 - MAP filter implementation

All 1024 complex floating-point elements of a single view’s data set are translated into the frequency domain by the forward FFT. As results emerge from the FFT, they are multiplied by the filter

coefficients. These filter coefficients were stored two across in two memory arrays during initialization so that this step can multiply 4 data elements in parallel. The results of the FFT-multiply operation are returned to OBM. After the entire single data set has been processed in this manner, the filtered data set is returned to the spatial domain via the inverse FFT and stored in OBM.

The measured time for a single data set's filtering operation was 20.4 microseconds. For all 180 data sets, the processing time is 3.63 milliseconds, or 2.1% of the MAP processor's total execution time.

4.5 MAP filtered dataset

In order for the pixel summation step to build eight pixels in parallel, eight copies of the filtered data set for the current view are built in FPGA internal memory in this step. This process consists of serializing the real part of filtered data across 4 OBM banks into 8 duplicate memory arrays. This step consumes 6.14 milliseconds (3.6%) for all 180 views.

4.6 MAP pixel summation

Pixel summation is actually a two-step operation. The first step is shown in Figure 8. For a given projection view (iView) and 8 pixel x,y positions (ix, iy+[0..7]), eight indices (detector position [0..7]) are calculated using the X, Y, and initial view angles (det_dx[], det_dy[], and detPosColStart[]) for each projection view's data set.

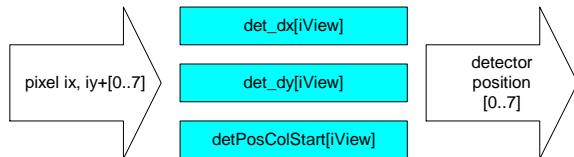


Figure 8 - MAP pixel summation step 1

Step 2 in pixel summation (Figure 9) uses these 8 detector position indices to select a filtered data element from each of the 8 duplicate filtered data sets described in section 4.5. These eight values are summed into eight pixels of the image stored in OBM.

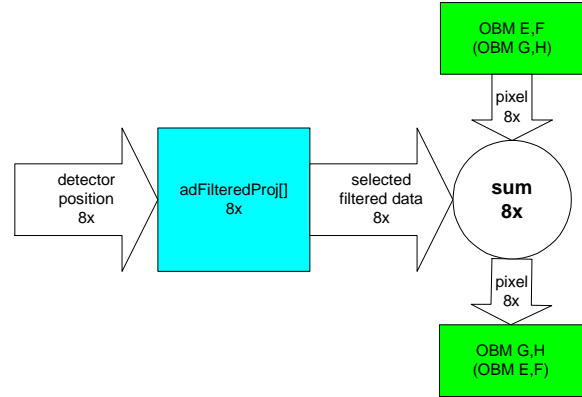


Figure 9 - MAP pixel summation step 2

Eight image pixels are accessed from two OBM banks per FPGA clock. This is possible because the Series H MAP supports two OBM read or write accesses per clock. One OBM bank is 64-bits wide, which is storage for two 32-bit floating point numbers. Accessing four 64-bit words from 2 OBM banks per clock yields eight floating point numbers per clock.

The image is initially stored in OBM banks E and F. However, a read-add-write operation from and to the same OBM bank would cause a loop slowdown and an unnecessary performance loss. To avoid this, OBM banks E and F are coupled with OBM banks G and H in a ping-pong fashion. That is, during an even-valued projection view count, this operation reads from OBM E and F, adds the selected filtered projection data to 8 pixels in parallel and writes the result to OBM G and H. During an odd-valued projection view count, the operation direction reverses: from OBM G and H to OBM E and F.

This MAP pixel summation step takes 157 milliseconds for all projection views, or almost 92% of the total MAP execution time.

4.7 MAP reconstructed image output

The final OBM pixel buffer is simply transferred back to the CPU allocated buffer for this image. This step consumes 0.58 milliseconds, or 0.3% of the MAP processor's total execution time. The final reconstructed image of the original Herman Head phantom (Figure 5) is shown in Figure 10.

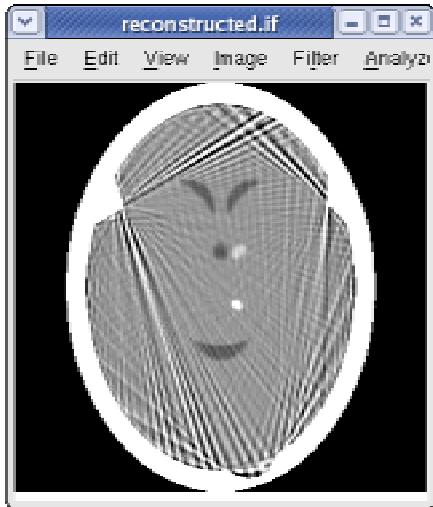


Figure 10 - Reconstructed Herman Head image

4.8 MAP FBP performance summary

This design was implemented for reconstructing a 1024x1024 image from 180 projection data sets. The measured MAP FBP performance summary, by MAP function section, is listed in Table 1. The bulk of the 172 millisecond execution time is spent in pixel summation, where each of 1,048,578 pixels have the results of 180 projection views summed into them.

Table 1 - MAP FBP performance summary

MAP FBP step	Time (ms)	% of total
Initialization	3.70	2.2
Filter	3.63	2.1
Filtered dataset	6.14	3.6
Pixel sum	157.40	91.8
Image transfer	0.58	0.3
total	171.45	100.0

4.9 MAP utilization summary

The Series H MAP has two Altera Stratix II 2S180 FPGAs available to a programmer. This implementation uses only one of these devices. A utilization summary for this design shows these results:

ALUTs: 66,416 / 143,520 (46 %)
 Registers: 92,818 / 143,520 (65 %)
 M512 rams: 211 / 930 (23 %)
 M4K rams: 704 / 768 (92 %)
 M-RAMs: 0 / 9 (0 %)
 DSP blocks: 408 / 768 (53 %)

The critical resource is the available M4K ram, of which 92% are used. This high utilization is due to the 8 duplicate filtered datasets, which is required in order to extract the 8x parallelism in pixel summation.

5. CPU FBP performance

The FBP portion of the original CTsim code (which uses the fftw library) was bracketed with gettimeofday() system calls, and an elapsed time was measured for a 180 view 1024x1024 CT image reconstruction. The elapsed CPU FBP execution time thus obtained was 5.057 seconds. The CPU used for this measurement was a 3.0 gigahertz 64-bit Xeon processor with a 2048 kilobyte cache. Given this, a microprocessor and a Series H MAP processor working together outperform a microprocessor working alone by a factor of 29x.

6. Conclusions

Reducing the 5 second CPU execution to 0.17 seconds for one image reconstruction is not significant. However, radiologists process more than one CT data set in a day. This 29x speedup indicates that radiologists could process 29 times more images in a day than they do now, with the additional bonus of significantly higher image resolution.

These results indicate that the SRC-7 eliminates the current limitations of image resolution and image generation time: two out of three of the trade-offs that concern medical equipment manufacturers. What about cost? Clustered computer systems typically operate at 15% CPU efficiency. This means that on average, any single CPU in a clustered system is doing useful work only 15% of the time an application is running. This means that 29 CPUs in a clustered system will not match the performance of a single SRC-7 system with its 29x performance advantage. Instead, it will require 193 clustered CPUs to achieve the same performance level. It is safe to say that a single SRC-7 system with 29x performance will cost significantly less than a system cluster of 193 CPUs.

Discussions with medical researchers indicate that this 29x performance improvement for a 180-view 1024x1024 CT image reconstruction is very promising from a proof-of-concept perspective. Future SRC-7 CT image reconstruction implementations will be concerned with application performance reconstructing higher resolution images on tomorrow's CT scan equipment.

7. Future directions

The measured performance results in section 4.8 indicate that any effort to accelerate FBP further on a MAP must concentrate on the pixel summation section, where almost 92% of the total MAP execution time is spent.

In section 4.9, the 92% utilization of the M4K ram resources to support 8x pixel summation parallelism suggest that further performance improvements will not be obtained by simply attempting to support 16x pixel summation. A re-examination of the loop iterators for the pixel summation reveals that this loop is determined by iterating over all pixels *and over all projection views*. A performance analysis indicates that a 54x performance improvement for a 180-view 1024x1024 image reconstruction may be obtained by performing only 4 pixel summations in parallel over 4 views' worth of filtered datasets. This means more performance may be obtained using fewer M4K ram resources. Work is currently underway to resolve the inherent data dependencies in this approach and implement this design.

Only one of the two Altera Stratix II 2S180 FPGAs was utilized for this design. Simply assigning half of the image reconstruction to each user logic device will likely yield a 56x performance improvement.

The index used to select filtered detector data to be summed into a pixel for each view is a function of fixed CT equipment parameters. This suggests these indices may be pre-computed and stored during system initialization, which would allow re-dedication of FPGA resources to additional parallel computational units. The only new inputs are patient scan data sets streaming from the CT equipment. One difficulty in this approach is the amount of memory required to store these pre-computed values. For 180 views and 1024x1024 pixels, roughly 1 gigabyte of storage with sufficient bandwidth to the FPGA to feed the computational engines in the MAP every FPGA clock is required. Work is underway at SRC Computers to understand the trade-offs in this approach and to evaluate potential performance improvements.

Real-time multi-modal (fMRI, CT, PET) medical image registration is currently of interest to the medical community. SRC is investigating the reconstruction techniques used across these imaging modalities with the intent to combine multiple sensor data processing with image registration.

8. References

- [1] Herman G.T., *Image Reconstruction from Projections: The Fundamentals of Computerized Tomography*, Academic Press, New York, 1980.
- [2] Xue, X, Cheryauka, A., and Tubbs, D., *Acceleration of Fluoro-CT Reconstruction for a Mobile C-Arm on GPU and FPGA Hardware: A Simulation Study*. Proceedings of SPIE -- Volume 6142, Medical Imaging 2006: Physics of Medical Imaging, Michael J. Flynn, Jiang Hsieh, Editors, 61424L, March 2, 2006.
- [3] Leeser, M., Coric, S., Miller, E., Yu, H., and Trepanier, M., *Parallel-beam backprojection: an FPGA implementation optimized for medical imaging*. J. VLSI Signal Process. Syst. 39, 3 (Mar. 2005), pp. 295-311.
- [4] Natterer F. and Wubbeling F., *Mathematical Methods in Image Reconstruction*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2001.
- [5] Bushberg, J.T., Seibert, J.A., Leidholt, E.M. and Boone, J.M., *The Essential Physics of Medical Imaging*. 2nd ed.. Lippincott Williams & Wilkins, Baltimore, 2001.
- [6] SRC Computers, Inc., IMPLICIT+EXPLICIT™ Architecture, MKT-030-01, Colorado Springs, CO, March 22, 2007.
- [7] SRC Computers, Inc., Introduction to the SRC-7 MAPstation™, MKT-036-01, Colorado Springs, CO, November 5, 2007.
- [8] SRC Computers, Inc., Series C-H MAP® Processor, http://www.srccomp.com/techpubs/docs/SRC_MAP_69226_BE.pdf, 2008.
- [9] Kevin M. Rosenberg, CTsim: The Open Source Computed Tomography Simulator, <http://www.ctsim.org>, version 4.5.3, 2006.
- [10] Frigo M. and Johnson S., The Fastest Fourier Transform in the West, <http://www.fftw.org>, 2006.
- [11] Shepp L. and Logan B., "The Fourier Reconstruction of a Head Section", *IEEE Transactions in Nuclear Science*, NS-21(6), 1974.