NAS has produced a number of benchmarking tools which were developed for Intel for benchmarking IA processors. These include:

Range Compression Benchmarking Tool

The SAR 2D Benchmark implements multiple radar range compression operations thus providing a two dimensional result, implemented by multiple one dimensional FFTs. Range compression is performed by calculating a sequence of 1-d correlations of the transmitted radar waveform (typically an FM chirp) with the vector formed by the sampled radar returns. If the transmitted pulse contains m samples, and each radar return has N samples (where N > m), then each resulting range compressed pulse will contain N - m samples.

A demo of this tool is available here . You need to be registered to access this page.

Full SAR Processor Benchmarking Tool

The Full SAR Processor Benchmarking Tool was designed to scale efficiently across 1-24 cores, with the number of cores in the target system configurable at run-time. The tool also has a run-time option for continuous operation so that sophisticated performance analysis tools can be used to characterize its performance over a longer time frame than resolving a single image would require.The code is written in standard 'C' and contains a compile-time switch to link in either the Intel IPPs or MKL performance libraries.

The SAR processor implements the following algorithm requirements:

- Initialisation including range and azimuth filter preparation.
- Range compress input data with range compression filters.
- Corner turn the data by transposing the data.
- Azimuth compress input data with azimuth compression filters.

A demo of this tool is available here . You need to be registered to access this page.

SARMTI Benchmarking Tool

The SARMTI processor detects moving targets within radar data from a Synthetic Aperture Radar (SAR) surveillance system. Moving Target Indication (MTI) systems usually either detect high-speed targets or use a radar system with multiple phase centres to detect low speed moving targets. Most MTI radar systems do not operate with the same system parameters as SAR radar imaging systems. The radar is either in a SAR mode or an MTI mode. The MTI algorithms developed by us are designed to detect moving targets from a single-phase centre of a radar system operating in a SAR mode.

For Intel, we ported the SARMTI processor software suite to an Intel system made up of four 4-socket Caneland (Tigerton processors) each running at 1.86GHz giving access to 16 cores on one machine. When the first part of this work was carried out the processor was compiled with GNU gcc version 4.1.2 (http://gcc.gnu.org/) and used the FFTW library (http://www.fftw.org/) to perform Fast Fourier Transform (FFT) operations within the compression stage of the processor.

In order to speed up the processor and make use of the 16 cores available on the Intel platform we introduced threads into the code.

How fast the processor runs depends on the radar dataset that it is processing and on how many valid targets are in the data. The more targets that are detected in the stage 1 step then the more processing is required in stage 2 and 3. The SARMTI software package comes with the following four demonstration programs:

Demo Name.

Description.

run_sarmti_test1

Uniform background with no targets.

run_sarmti_test2

Uniformed background with 10 moving targets.

run_sarmti_test3

Structured background with no targets.

run_sarmti_test4

Structured background with 10 moving targets.

Table 1: The SARMTI demonstration programs.

The four demonstration programs were timed on the Caneland system.

The following table shows the total time for each demonstration program:

Demo Name.

Total Time in seconds.

run_sarmti_test1

85.412 secs

run_sarmti_test2

120.190 secs.

run_sarmti_test3

104.001 secs.

run_sarmti_test4

166.231 secs.

Table 2: Non-threaded code timings for demo programs.

The second demonstration program (run_sarmti_test2) is a typical example of executing the SARMTI processor where we have a number of targets to detect within the input complex data. The program run_sarmti_test2 runs in a total of 120 seconds on the Caneland System. Using the GNU profiler gprof we obtained a profile stating how long was being spent in each function of the code. The expensive functions (those that took more than 1% of the total time) could be split into the following four types:

- Target detection functions: this includes CfAR and detection filtering.
- FFTW library functions: forward and backward ffts.
- Compression functions: called in stage 1 and stage 2. These use FFTW.
- Data extraction routines: preparing input/output data.

The following tables show the above operation types and the expensive function calls associated with each type. The tables give an idea of how mush time is being spent within these operations.

Target Detection Function Calls.

Time %

Function name.

14.71

mti_cfar_rect

8.2

igamma

6.38

mti_create_detection_mask

1.19

GammaRegularized

30.48%

Total

Table 3: Serial target detection performance.

FFTW Function Calls.

Time %

Function name.

9.81

q1fv_8

8.35

n1fv_64

5.50

t3fv_8

4.96

n1_64

4.42

q1bv_8

4.37

t3fv_32

4.12

n1bv_64

3.47

n2fv_64

2.97

fftwf_cpy2d_pair_ci

2.84

t3bv_8

2.26

t3bv_32

53.07 %

Total

Table 4: Serial FFTW performance.

Other Data Compression Function Calls.

Time %

Function name.

7.98

mti_inter_ptr_correlate_fft

1.96

mti_cross_correlate_and_sum

1.28

mti_inter_ip_correlate_fft

11.22

Total.

Table 5: Serial compression routine performance.

Data Extraction Function Call.

Time %

Function Name.

2.04

mti_extract_window

2.04 %

Total

Table 6: Serial data extraction function performance.

 TOTALS.

Time %

Operation.

30.48

DETECTION.

53.07

FFTW.

11.22

OTHER COMPRESSION.

2.04

DATA EXTRACTION

96.81 %

Total.

Table 7: Overall serial processor performance.

The results show that 64 % of the time is spent compressing complex data and 30 % of the time is spent target detection processing. After data preparation and extraction all the rest of the processing functions are only taking up 3.19 % of the processing time.

A demo of the SARMTI Benchmarking Tool is available here . You need to be registered to access this page.