FRONTGRADE

APPLICATION NOTE

UT699E/700

Benchmark Performance UT699E/700 LEON 3FT

11/20/2017
Version #: 1.0.1

Table 1: Cross Reference of Applicable Products

Product Name	Manufacturer Part Number	SMD #	Device Type	Internal PIC Number
UT700 LEON	UT700	5962-13238	Processor	WQ03
UT699E LEON	UT699E	5962-13237	Processor	WQ02

1.0 Overview

Although operation within a customer application is always the best measure of performance; software benchmarks provide a convenient way to compare processors to standard metrics. The amount of work done by a system in a period of time goes beyond just a benchmark score. It depends on many factors including peripherals, DMA that operates in the background that can steal cycles, memory waitstates, optimized libraries and compiler optimization level and more. For example, a Dhrystones benchmark can gain significantly better score with Loop optimization.

This paper provides several benchmarks (see section **3.0**) scores to show how the UT700 LEON processor performs on these benchmarks' tests. Frontgrade is not advocating any specific benchmark test software but advises our readers to visit the respective benchmark's websites to draw their own conclusion of which is most suitable for a particular application.

This figure **(Figure 1:1)** shows the UT700 LEON 3FT SPARC™ V8 Microprocessor functional block diagram. The UT699E is a subset of the UT700 (the former doesn't have SPI or 1553), however the results herein, are also applicable to this device.

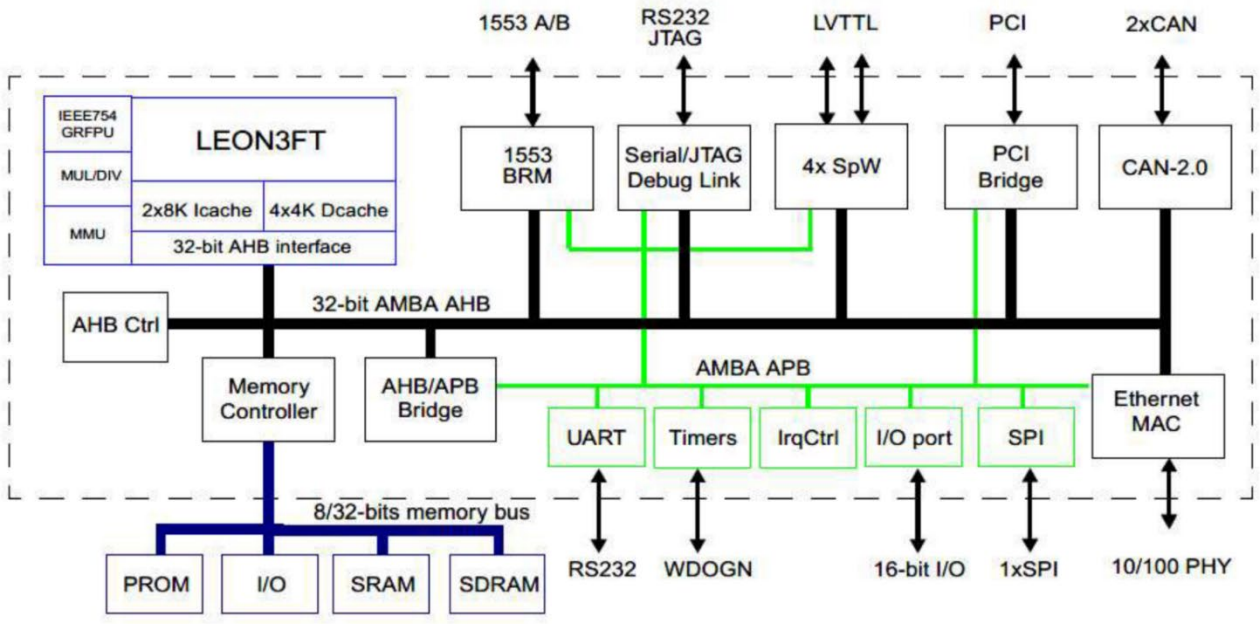


Figure 1:1: UT700 LEON 3FT SPARCÔ V8 Microprocessor

2.0 Benchmarks

In the following sections, Frontgrade provides several benchmarks plots and results. The benchmarks are tested using the LEON Evaluation Application Platform (LEAP) board. The LEAP board is manufactured and sold by Frontgrade as a development platform for our customer to jump-start development of their application code. The LEAP (Figure 2:1) board provides commercial MRAM and SDRAM for non-volatile and volatile memories, respectively; hence, all the subject benchmark tests are run on the SDRAM. For more information on LEAP, please see: frontgrade.com.

Figure 2:1: The LEAP Development Platform

The System and the Memory Bus clocks and the SDRAM waitstates are shown in **Table 2.**

Table 2: SDRAM Settings

	S1	S0	NODIV	memClk (MHz)	sysClk (MHz)	MCFG2 (SDRAM Settings)		
						DP	DF	DC
1	L	L	Up/Down	50/100	100	0/1	1/5	0/1
2	M	L	Up/Down	31/63	63	0/0	0/2	0/0
3	H	L	Up/Down	75/150	150	0/1	3/7	0/1
4	L	M	Up/Down	66/132	132	0/1	2/7	0/1
5	M	M	Up/Down	25/50	50	0/0	7/1	0/0
6	H	M	Up/Down	38/75	75	0/0	0/3	0/0
7	L	H	Up/Down	63/125	125	0/1	2/6	0/1
8	M	H	Up/Down	41/83	83	0/0	0/3	0/0
9	H	H	Up/Down	100/200	200	1/1	5/7	1/1

3.0 Test Conditions, Benchmarks Plots and Results

In this section, Frontgrade provides several benchmarks plots and results as follows:

- Dhrystones Benchmark
- Coremark Benchmark
- Flops20 Benchmark
- Stanford Benchmark
- Whetstone Benchmark

We used the GCC compiler tool chain for our benchmark's tests compilation with the following optimizations:

gcc version 4.4.2 (BCC 4.4.2 release 1.0.44)
sparc-elf-gcc -O3 –mv8 <infile> -o <outfile>

The UT700 is specified at a maximum system clock frequency of 166MHz. In the benchmark's tests, we deliberately tested it up to 200MHz. For all UT700 specification do refer to the UT700 datasheet.

3.1 Dhrystones Benchmark (Version 2.1)

Dhrystone is a synthetic computing benchmark program developed in 1984 by Reinhold P. Weicker intended to be representative of system (integer) programming. The Dhrystone grew to become representative of general processor (CPU) performance.

Dhrystone remains remarkably resilient as a simple benchmark, but its continuing value in establishing true performance is debatable. It is easy to use, well documented, fully self-contained, well understood, and can be made to work on almost any system. In particular, it has remained in broad use in the embedded computing world. Dhrystone remains in use 30 years after it was designed by Weicker, a longer life than most software.

The UT700 Dhrystone's results **(1.38 DMIPS/MHz)** and plots are as follows:

Table 3: Dhrystones Configurations

x=freq too high

S1	S0	NODIV	memClk (MHz)	SysClk (MHz)	DIMPS	DMIPS
L	L	Up/Down	50/100	100	86.7	138.1
M	L	Up/Down	31/63	63	53.7	85.4
H	L	Up/Down	75/150	150	129.9	XXX
L	M	Up/Down	66/132	132	114.4	179.8
M	M	Up/Down	25/50	50	41.0	67.7
H	M	Up/Down	38/74	74	65.9	101.5
L	H	Up/Down	62/124	124	109.3	168.9
M	H	Up/Down	41/82	82	71.2	113.7
H	H	Up/Down	100/200	200	185.1	XXX

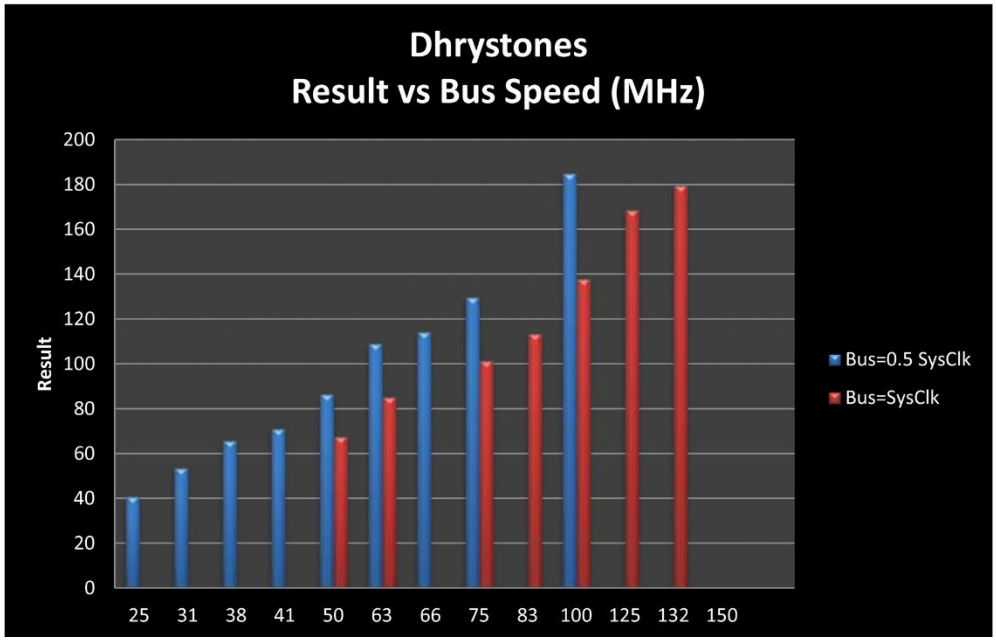


Figure 3:1: Dhrystones Benchmark

3.2 CoreMark Benchmark (Version 1.0)

CoreMark is a synthetic benchmark that measures the performance of just the central processing units (CPU), or core of a processor, used in embedded systems. It was developed in 2009 by Shay Gal-On at EEMBC and is intended to become an industry standard, replacing the antiquated Dhrystone benchmark. The code is written in C and contains implementations of the following algorithms: list processing (find and sort), matrix manipulation (common matrix operations), state machine (determine if an input stream contains valid numbers), and CRC.

The CRC algorithm serves a dual function; it provides a workload commonly seen in embedded applications and ensures correct operation of the CoreMark benchmark, essentially providing a self-checking mechanism. Specifically, to verify correct operation, a 16-bit CRC is performed on the data contained in elements of the linked-list.

To ensure compilers cannot pre-compute the results at compile time, every operation in the benchmark derives a value that is not available at compile time. Furthermore, all code used within the timed portion of the benchmark is part of the benchmark itself (no library calls).

The UT700 CoreMark's results and plots are as follows:

Table 4: CoreMark Configurations

x=freq too high

S1	S0	NODIV	memClk (MHz)	sysClk (MHz)	Iteration (/sec)	Iteration (/sec)
L	L	Up/Down	50/100	100	171.379	180.193
M	L	Up/Down	31/63	63	106.207	114.052
H	L	Up/Down	75/150	150	256.993	XXX
L	M	Up/Down	66/132	132	226.192	240.720
M	M	Up/Down	25/50	50	84.084	90.523
H	M	Up/Down	38/75	75	130.236	135.757
L	H	Up/Down	63/125	125	215.895	224.165
M	H	Up/Down	41/83	83	140.539	150.229
H	H	Up/Down	100/200	200	340.453	XXX

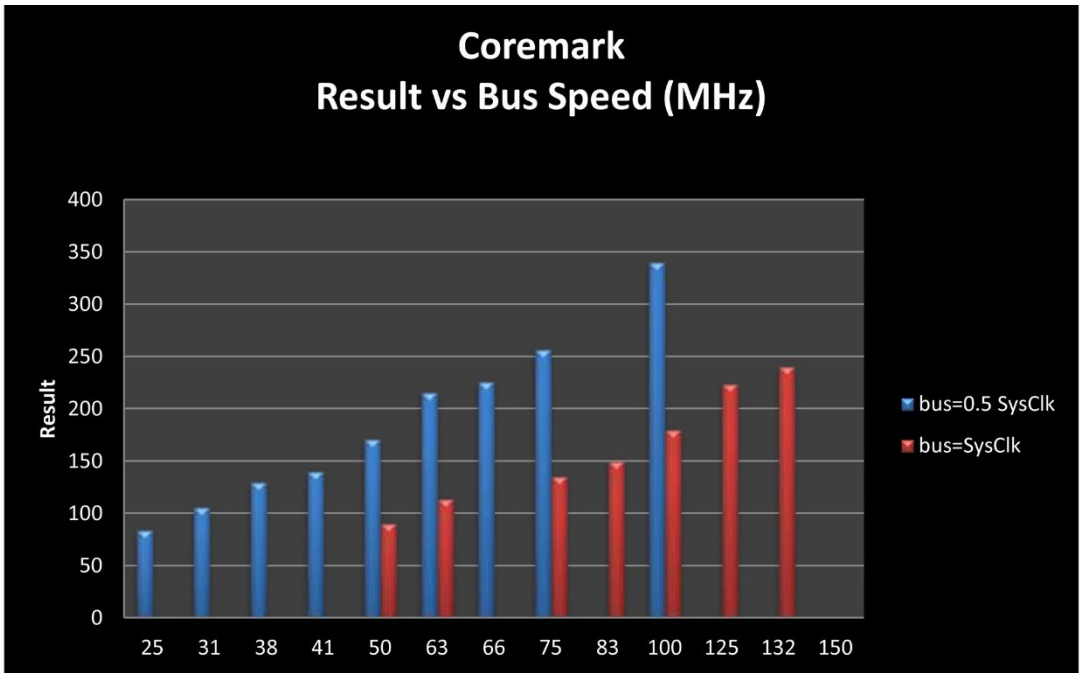


Figure 3:2: CoreMark Benchmark

3.3 Flops20 Benchmark (Version 2.0)

Flops20 is a C program which attempts to estimate your systems floating-point 'MFLOPS' rating for the FADD, FSUB, FMUL, and FDIV operations based on specific instruction mixes. The program provides an estimate of PEAK MFLOPS performance by making maximum use of register variables with minimal interaction with main memory. The execution loops are all small so that they will fit in any cache. Flops20 can be used along with Linpack and the Livermore kernels (which extensively exercises memory) to gain further insight into the limits of system performance. The flops20 execution modules include various percent weightings of FDIV's (from 0% to 25% FDIV's) so that the range of performance can be obtained when using FDIV's. FDIV's, being computationally more intensive than FADD's or FMUL's, can impact performance considerably on some systems.

The UT700 Flops20's results and plots are as follows:

Table 5: Flops20 Configurations

x=freq too high

S1	S0	NODIV	memClk (MHz)	sysClk (MHz)	MFLOPS	DIMPS(-)	DMIPS(+)
L	L	Up/Down	50/100	100	1	32.0000	32.0000
					2	21.6216	21.6216
					3	26.0947	26.0947
					4	28.7975	28.7975
M	L	Up/Down	31 /62	62	1	19.8355	20.1550
					2	13.4057	13.6220
					3	16.1790	16.4403
					4	17.8552	18.1433
H	L	Up/Down	75/150	150	1	47.9608	X
					2	32.4310	X
					3	39.1388	X
					4	43.1887	X
L	M	Up/Down	66/132	132	1	42.2228	42.8590
					2	28.5413	28.9722
					3	34.4441	34.9661
					4	38.0107	38.5874
M	M	Up/Down	25 /50	50	1	16.0000	16.0000
					2	10.8108	10.8127
					3	13.0474	13.0481
					4	14.3987	14.3987
H	M	Up/Down	37 /74	74	1	24.3204	24.3204
					2	16.4308	16.4302
					3	19.8311	19.8302
					4	21.8849	21.8849
L	H	Up/Down	62/124	124	1	40.3296	39.7136
					2	27.2432	26.8092
					3	32.8783	32.3579
					4	36.2821	35.7103
M	H	Up/Down	41/ 82	82	1	26.2378	26.2378
					2	17.7279	17.7279
					3	21.3968	21.3968
					4	23.6134	23.6134
H	H	Up/Down	100/200	200	1	64.0000	X
					2	43.2432	X
					3	52.1895	X
					4	57.5949	X

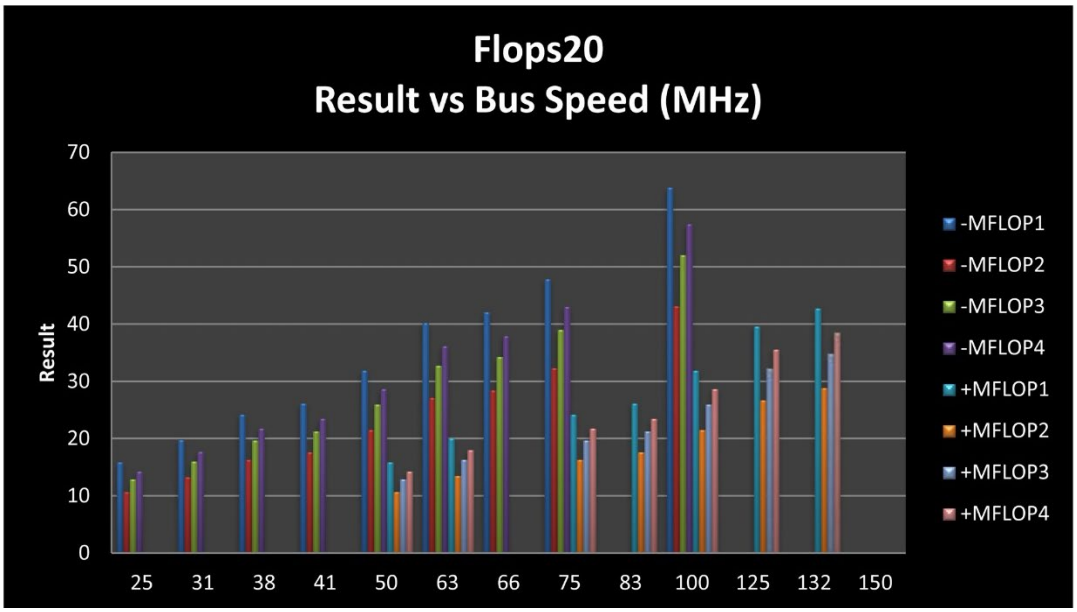


Figure 3:3: Flops20 Benchmark

3.4 Stanford Benchmark (Version: 4.2)

This benchmark suite is relatively short, both in program size and execution time. It requires no input, and prints the execution time for each program, using the system-dependent routine Getclock, to find out the current CPU time. It does a rudimentary check to make sure each program gets the correct output. These programs were gathered by John Hennessy and modified by Peter Nye.

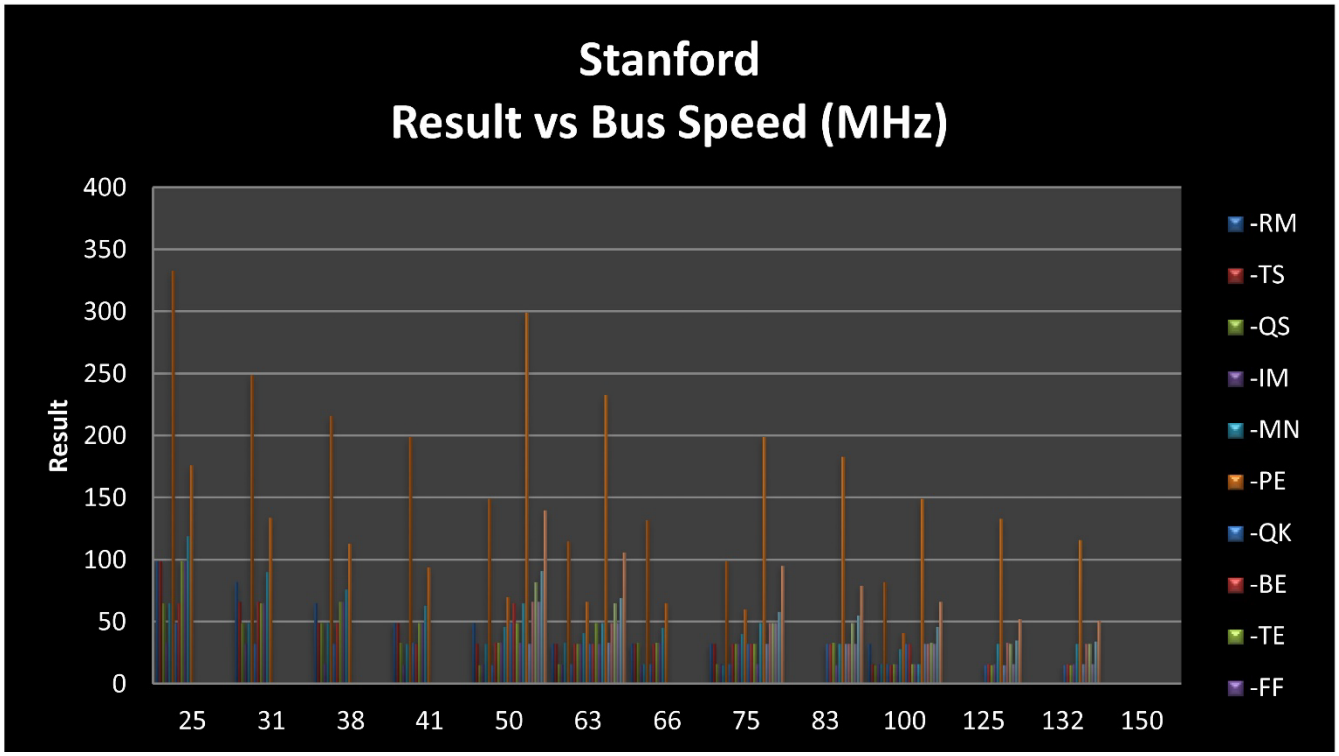
The UT700 Stanford's results and plots are as follows:

Table 6: Stanford Configurations

PM = Perm	TS = Towers	QS = Queens	+=NODIV (Down)
IM = Intmm	PE = Puzzle	QK =Quick	1 to 6: see Table 2
BE = Bubble	TE = Tree	FF = FFT	x=freq too high
NT = NFloat	FT = Float	—=NODIV (Up)	

	RM	TS	QS	IM	MN	PE	QK	BE	TE	FF	NT	FT
1U	50	33	16	17	33	150	16	34	34	34	47	71
1D	33	33	17	17	17	150	33	33	34	33	47	67
2U	83	67	50	33	50	250	33	67	66	67	91	135
2D	33	33	50	33	50	234	34	50	66	50	70	107
3U	33	33	17	17	16	100	17	33	33	34	41	61
3D	x	x	x	x	x	x	x	x	x	x	x	x
4U	33	33	34	16	17	133	17	33	34	33	46	66
4D	16	17	16	17	33	117	17	33	33	17	35	52
5U	100	100	66	34	66	334	50	66	100	84	120	177
5D	50	66	50	34	66	300	33	67	83	67	92	141
6U	66	50	50	17	50	217	33	50	67	50	77	114
6D	33	33	33	17	50	200	33	50	50	50	59	96

Figure 3:4: Stanford Benchmark



3.5 Whetstone Benchmark (Version 1996)

The Fortran Whetstone programs were the first general purpose benchmarks that set industry standards of computer system performance, primarily evaluates floating-point arithmetic performance and is available in the Fortran and C programming language. Whetstone programs also addressed the question of the efficiency of different programming languages, an important issue not covered by more contemporary standard benchmarks. Results are provided for computers produced during the 1960's to present day systems, including via different languages.

The Whetstone benchmark, a UK product, was based on work by Brian Wichmann of the National Physical Laboratory. It was developed by Harold Curnow of HM Treasury Technical Support Unit (TSU - later part of Central Computer and Telecommunications Agency or CCTA). This document was produced by Roy Longbottom (TSU/CCTA 1960 to 1993), who carried out further development.

The UT700 Whetstone's results and plots are as follows:

Table 7: Whetstone Configurations

U= NODIV (Up) D=+=NODIV (Down) 1 to 9: see Table 2 +=NODIV (Down)
x=freq too high

	MWIPS	Mflops1	Mflops2	Mflops3	Cosmops	Expmops	Fixpmops	Ifmops	Eqmops
1U	34.777	24.233	15.973	4.454	1.278	0.648	30.917	49.977	8.244
1D	38.615	24.234	16.799	5.118	1.364	0.701	33.271	49.991	9.625
2U	21.448	15.021	9.895	2.755	0.791	0.394	19.121	30.977	5.099
2D	24.942	15.264	10.546	3.392	0.864	0.442	21.443	31.483	6.290
3U	52.277	36.355	23.965	6.676	1.914	0.985	46.382	74.992	12.368
3D	x	x	x	x	x	x	x	x	x
4U	46.690	32.474	21.413	5.967	1.711	0.877	41.464	66.953	11.056
4D	52.973	32.473	22.433	7.160	1.824	0.960	44.269	66.978	13.345
5U	17.730	12.112	8.100	2.318	0.650	0.321	14.893	24.979	4.188
5D	19.766	12.112	8.367	2.697	0.685	0.347	17.020	24.982	4.995
6U	26.382	18.412	12.144	3.387	0.971	0.487	23.495	37.971	6.265
6D	29.746	18.173	12.551	4.044	1.028	0.529	25.512	37.485	7.488
7U	43.873	30.532	20.132	5.609	1.610	0.822	38.962	62.972	10.389
7D	48.734	30.050	20.789	6.630	1.686	0.884	41.335	61.999	11.550
8U	28.510	19.869	13.103	3.661	1.048	0.527	25.371	40.970	6.766
8D	32.936	20.111	13.894	4.469	1.138	0.588	28.254	41.478	8.284
9U	68.249	48.472	32.388	8.490	2.549	1.347	53.012	100.003	16.492
9D	x	x	x	x	x	x	x	x	x

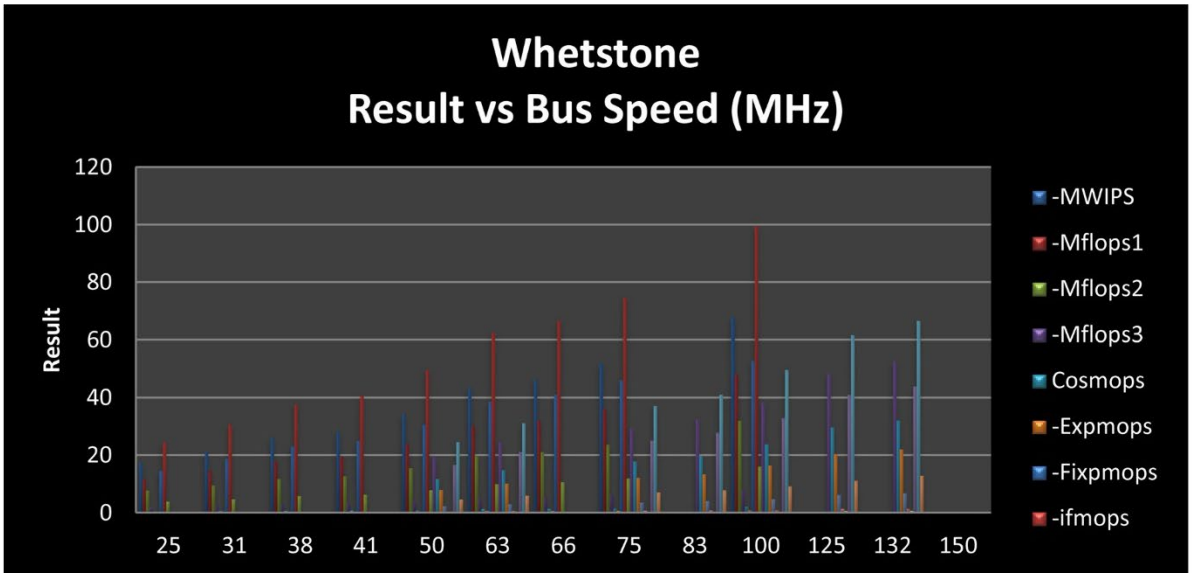


Figure 3:5: Whetstone Benchmark

-=NODIV (Up), +=NODIV (Down)

4.0 Summary and Conclusion

Frontgrade presented various benchmark results for the UT700 LEON processor and provided a detailed description of the methodology used to collect these results. Users should investigate the test methodology used before comparing benchmarks from multiple suppliers, as results can be strongly affected by compiler options, hardware options, type of memory, etc. Frontgrade is interested in any customer application performance data and will be pleased to discuss results with end users.

For more information about our UT700 LEON 3FT/SPARC™ V8 Microprocessor and other products please visit our website, frontgrade.com or email us at https://www.frontgrade.com/contact-us.

Revision History

Date	Revision #	Author	Change Description	Page #
11/10/2017	1.0.0	MTS	Initial Release	
11/20/2017	1.0.1	MTS	Update all Tables	