Real-Time Benchmarking

Presenter: Alexander Tessarolo, TI (a-tessarolo@ti.com)

Contributor: Ashwini Athalye, TI (ashwini.athalye@ti.com)

Benchmarking Dilemmas

- **MARKETING Likes The Single Number (DMIPS, CoreMARK,..)**
  - They also intermix or replace MHz with MIPS
  - MFLOPS, Equivalent MHz, are other measures used, but impart no real information about relevance to the customer problem

- **Customers Generally Are A Mixed Bag:**
  - Some Want To Know DMIPS, CoreMARKk, etc. Score
  - Some Will Supply Their Own Code To Benchmark & Want The Execution Time Of That Code
  - Some Will Take Our Internal Benchmarks As Reference

- **DMIPS Still Comes Up A LOT:**
  - We Have To Spend Time Educating Customers Why It Is A BAD Benchmark
  - Not All DMIPS Are Equal (what is legal and what isn't legal in the benchmark? is it an apples for apples comparison?)

- **Customers Generally Care About The Following:**
  - Can the processor/device do the job
  - Execution time and how much spare bandwidth is available for Safety, Security and other tasks
  - Headroom for entropy and future growth
  - Roadmap to future devices (how much performance will the next generation bring)
  - They generally make a platform decision that will last many product generations

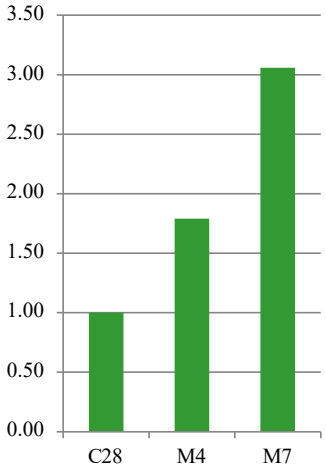
- **Challenges With Synthetic Benchmarks:**
  - Relevance to user application
  - Do not factor Signal Chain Performance (interrupts, peripheral access)
  - Cannot be broken up into smaller pieces to highlight strengths/weaknesses of a CPU
  - Memory System Impact (Cache vs TCM) Not Evaluated
  - Do Not Factor System Level Performance When Running Multiple Code Threads

A Single Benchmark or Single Score Cannot Convey The Reality & Complexity Of Embedded Systems Today & In The Future

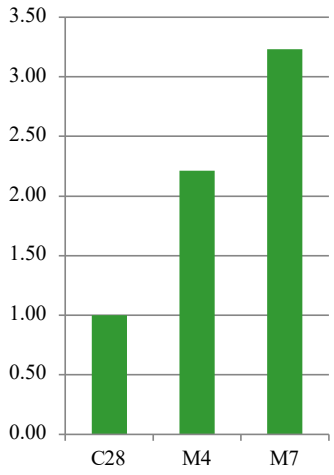
Synthetic Benchmarks vs "Real" Application Benchmarks

Synthetic Benchmarks

DMIPS/MHz
(normalized to C28)



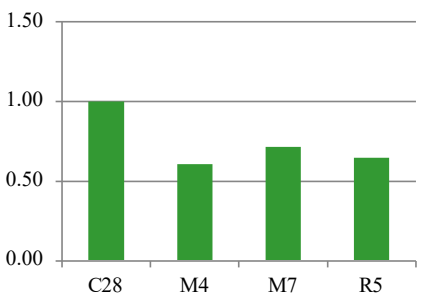
CoreMARK/MHz
(normalized to C28)



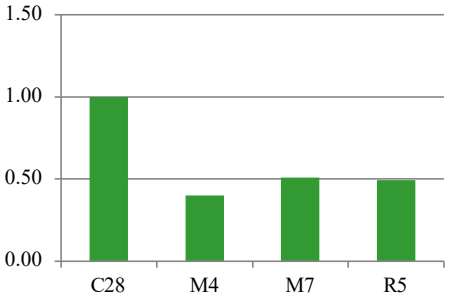
"Real" Application Benchmarks

Signal Chain Benchmarks

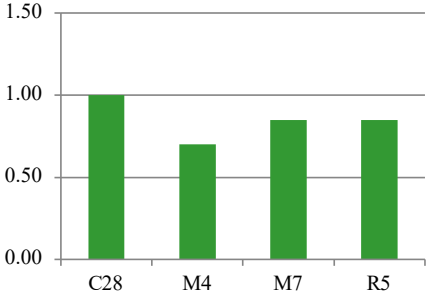
Motor Control
(e.g. AC Induction Motor)



Digital Power
(e.g. EV On Board Charger)

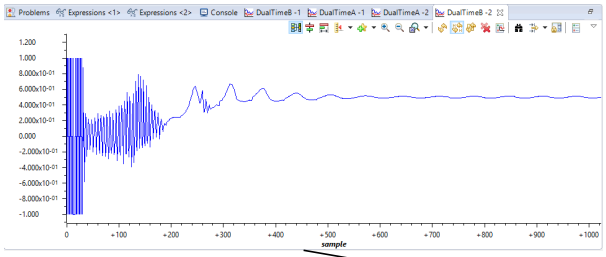


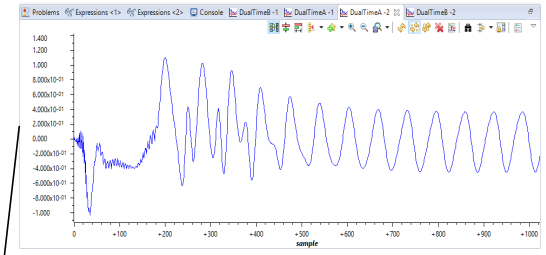
DSP Benchmark
(e.g. 512-RFFT, 32-bit Floating-Point for ARC detection, Solar Inverter)



Note: Performance Given As A Normalized Ratio To C28 CPU. Higher Number = Higher Performance

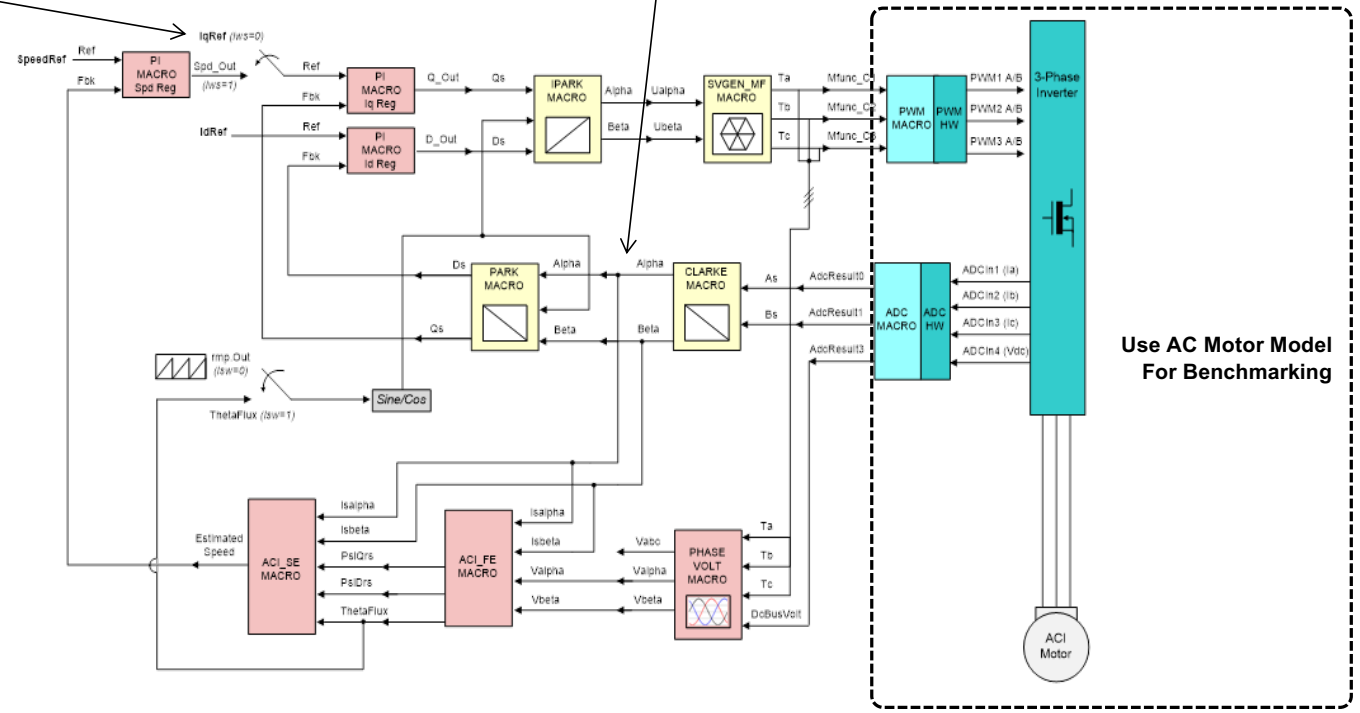
Example Benchmark: AC Induction Motor, Sensorless Algorithm



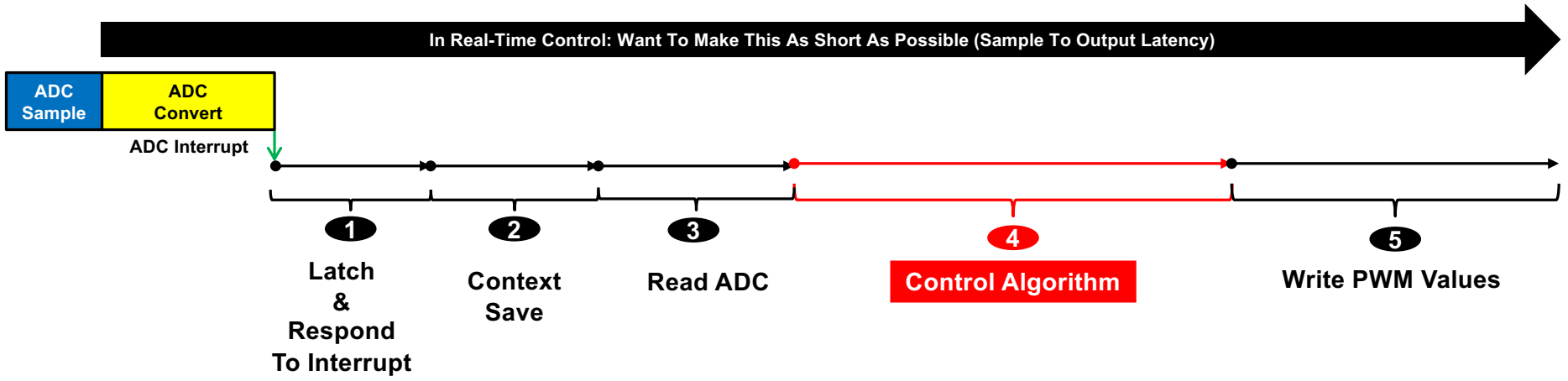


Current

Speed



Signal Chain Benchmarking



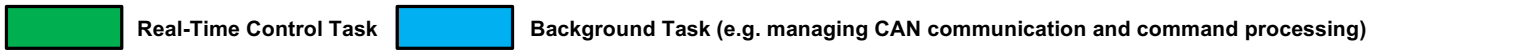
Motor Control Signal Chain Benchmark Example							
Device	**1**	**2**	**3**	**4**	**5**	**Total**	**Performance Gain**
C28 @100MHz	0.100 uS	0.380 uS	0.040 uS	4.790 uS (89% of total time)	0.080 uS	5.390 uS	**1.00 (ref)**
M4F @150MHz	0.123 uS	0.450 uS	0.080 uS	5.947 uS (89% of total time)	0.080 uS	6.680 uS	**0.81**

Digital Power Signal Chain Benchmark Example							
Device	**1**	**2**	**3**	**4**	**5**	**Total**	**Performance Gain**
C28 @200MHz	0.050 uS	0.100 uS	0.060 uS	0.880 uS (49% of total time)	0.705 uS (39%)	1.795 uS	**1.00 (ref)**
R5F @400MHz	0.125 uS	0.138 uS	0.313 uS	0.873 uS (48% of total time)	0.373 uS (20%)	1.820 uS	**0.99**
R5F @800MHz	0.063 uS	0.069 uS	0.156 uS	0.436 uS (41% of total time)	0.350 uS (33%)	1.074 uS	**1.67**

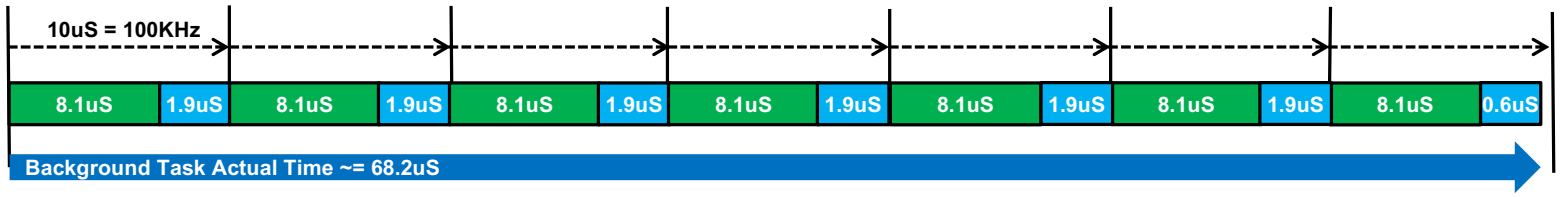
Desire: Real-Time Benchmarks That Factor I/O Processing & Interrupt Response

System Level Benchmarking - Mixing Real-Time Tasks With Background Tasks

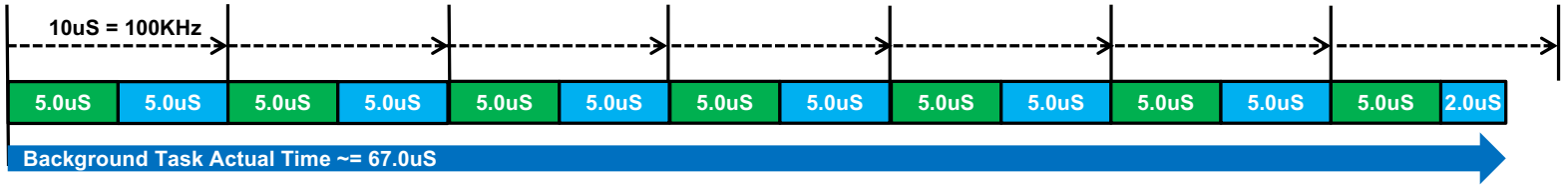
- **Background Task(s) Performance In Real-Time System Is Impacted By Real-Time Control Performance**



CPU-1 System
Real-Time Task = 8.1uS
Background Task = 12uS



CPU-2 System
Real-Time Task = 5.0 uS
Background Task = 32uS



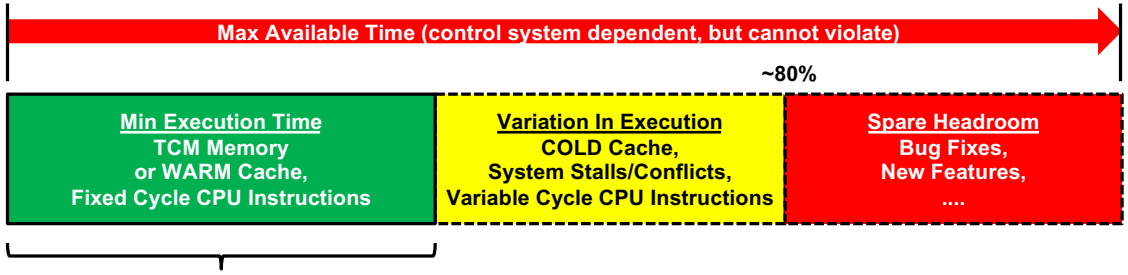
Note: Background Task Execution Time Shown Assumes It Is The Sole Task Executing.

Note: CPU-1 & CPU-2 Execute The Same Real-Time & Background Tasks:

- CPU-1 Is More Efficient At Background Tasks
- CPU-2 Is More Efficient At Real-Time Control Tasks
- Both Systems Have The Same Net Performance

Desire: Real-Time Benchmarks That Factor Mixing Of Real-Time & Non-Real Time Tasks

Cache Vs Tightly Coupled Memory (TCM) Systems Impact On DETERMINISM



A Deterministic System Can Run Closer To 80%, Hence Execute At Lower MHz

A Non-Deterministic System Needs To Run At Higher MHz To Make Sure It Doesn't Exceed 80% Of Allocated Time
(RULE Of THUMB: Cache Based Systems Need To Run > 2x MHz Relative To TCM Based Systems)

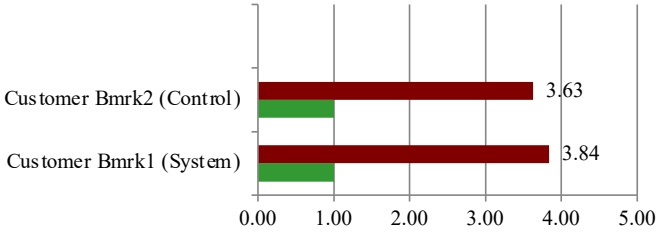
Example: Dhrystone Benchmark

Dhrystone	M7
DMIPS/MHz	2.27 (251 cycles per loop)

NOTE: The first time through the Dhrystone loop, the M7 takes more cycles as it builds up the caching mechanisms:

1st iteration	333 cycles	1.71 DMIPS/MHz
2nd iteration	271 cycles	1.89 DMIPS/MHz
3rd iteration	253 cycles	2.26 DMIPS/MHz
4th iteration	251 cycles	2.27 DMIPS/MHz (and all subsequent iterations)

Example: Customer Benchmark On Two Different Types Of Control Algorithms Using ARM-R5F CPU System



Note: To measure COLD cache performance, we clear (or reset) cache contents and measure cycles on very first iteration. Looping the same benchmark multiple times gives the WARM cache cycles. Numbers are then checked against running code from TCM memory.

Desire: Real-Time Benchmarks That Factor Memory System Impact & Hence Determinism Of Code Execution