EnergyAnalyzer:

Using Static WCET Analysis Techniques to Estimate the Energy Consumption of Embedded Applications

Simon Wegener[1], Kris K. Nikov[2], Jose Nunez-Yanez[3], Kerstin Eder[2]
[1]AbsInt Angewandte Informatik GmbH [2]University of Bristol [3]Linköping University

AbsInt

Acknowledgements

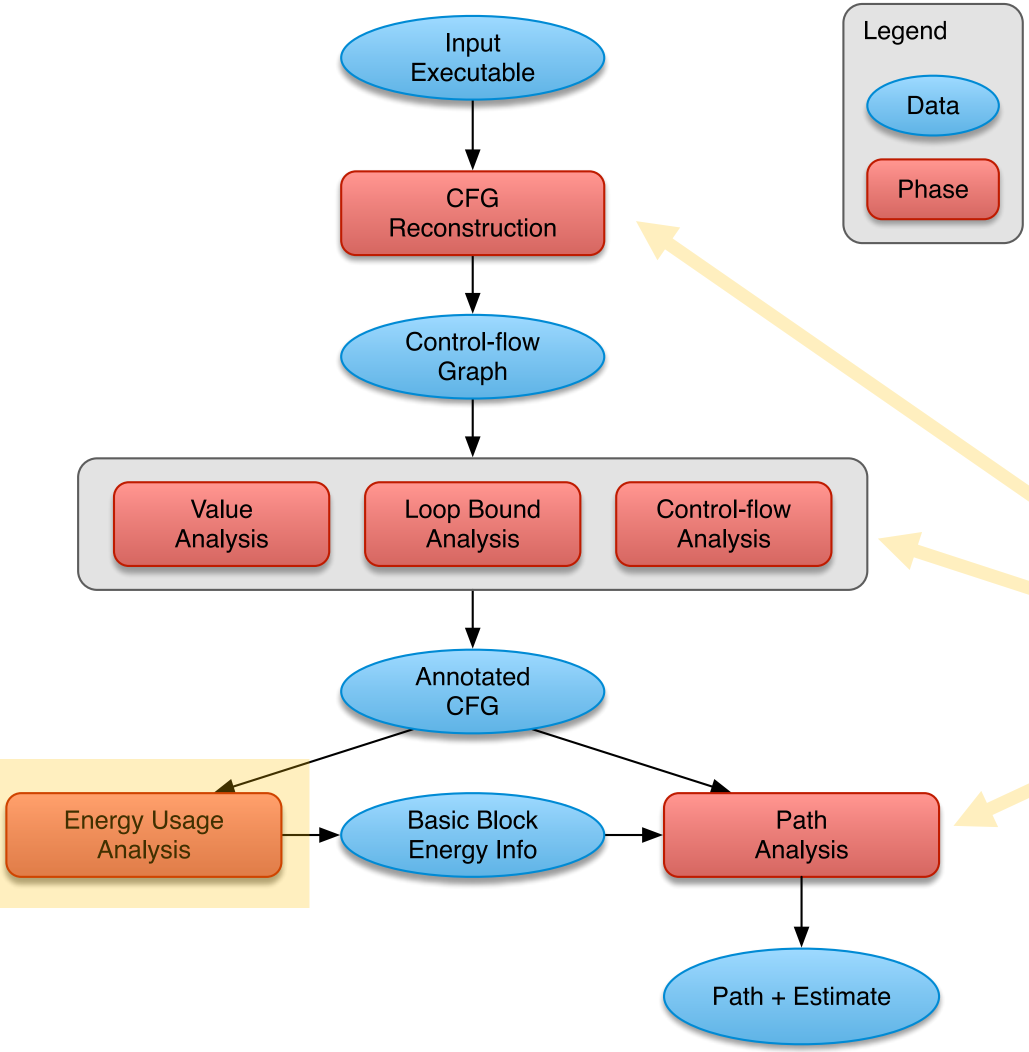
Funded by the Horizon 2020 Framework Programme of the European Union

- This work was supported by the H2020 project **TeamPlay**, Grant Agreement No. 779882.
- The authors like to thank:
  - **Marcos Martinez de Alejandro**, **Nikos Fragoulis**, **Ali Sahafi**, and **Vangelis Vassalos** for the work on the use cases
  - **Heiko Falk** and **Shashank Jadhav** for the integration of EnergyAnalyzer into WCC
  - Previous versions of the energy models have been generated by **Kyriakos Georgiou** and **Zbigniew Chamski**

Motivation

- Safety-critical systems must satisfy non-functional requirements
  - Computation must finish before a task reaches its deadline
  - Enough stack memory must be available for the system
- Solution: WCET/WCRT analysis, stack usage analysis
- What about energy-constrained systems?
- Idea: Apply techniques known from WCET analysis to statically estimate energy usage of embedded software

Tool Structure of EnergyAnalyzer



same as in aiT

Using energy models based on PMU counters

AbsInt

Performance Monitoring Units

- PMUs are hardware units that track various events during the execution of software
- Examples:
  - Cache hits
  - Cache misses
  - Memory accesses
  - Number of executed instructions
  - …
- We use the PMU counters as proxy for energy usage

AbsInt

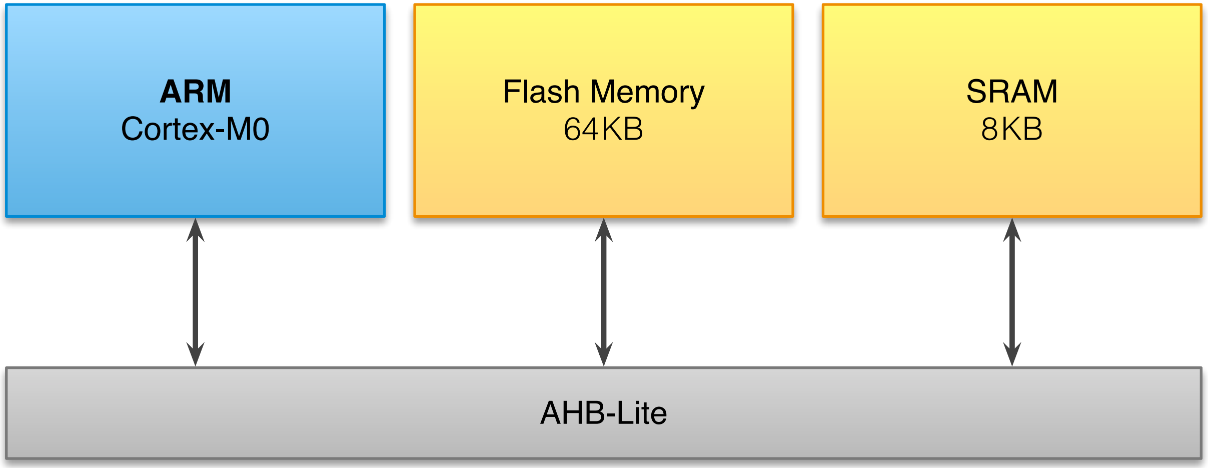
Energy Model Derivation

- Energy models are build using linear regression
  - We employed a Non-Negative Least Squares (NNLS) solver

- $E = \Sigma_x(\beta_x \times C_x) + \alpha$
  - where x are the various events tracked by PMU counters
  - C_x are the counter values
  - β_x are the coefficients of the model
  - α is the residual error term

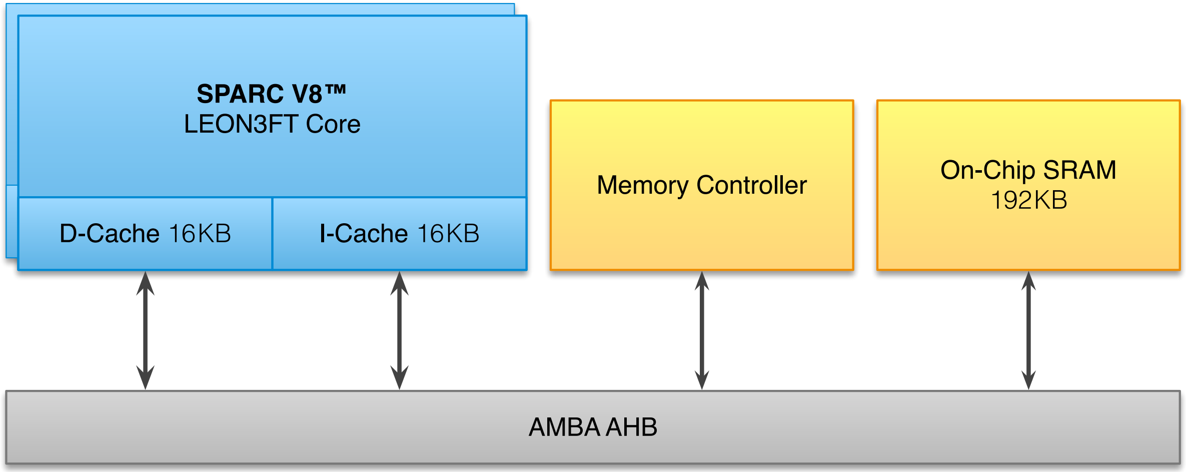
Target Architectures – ARM Cortex-M0

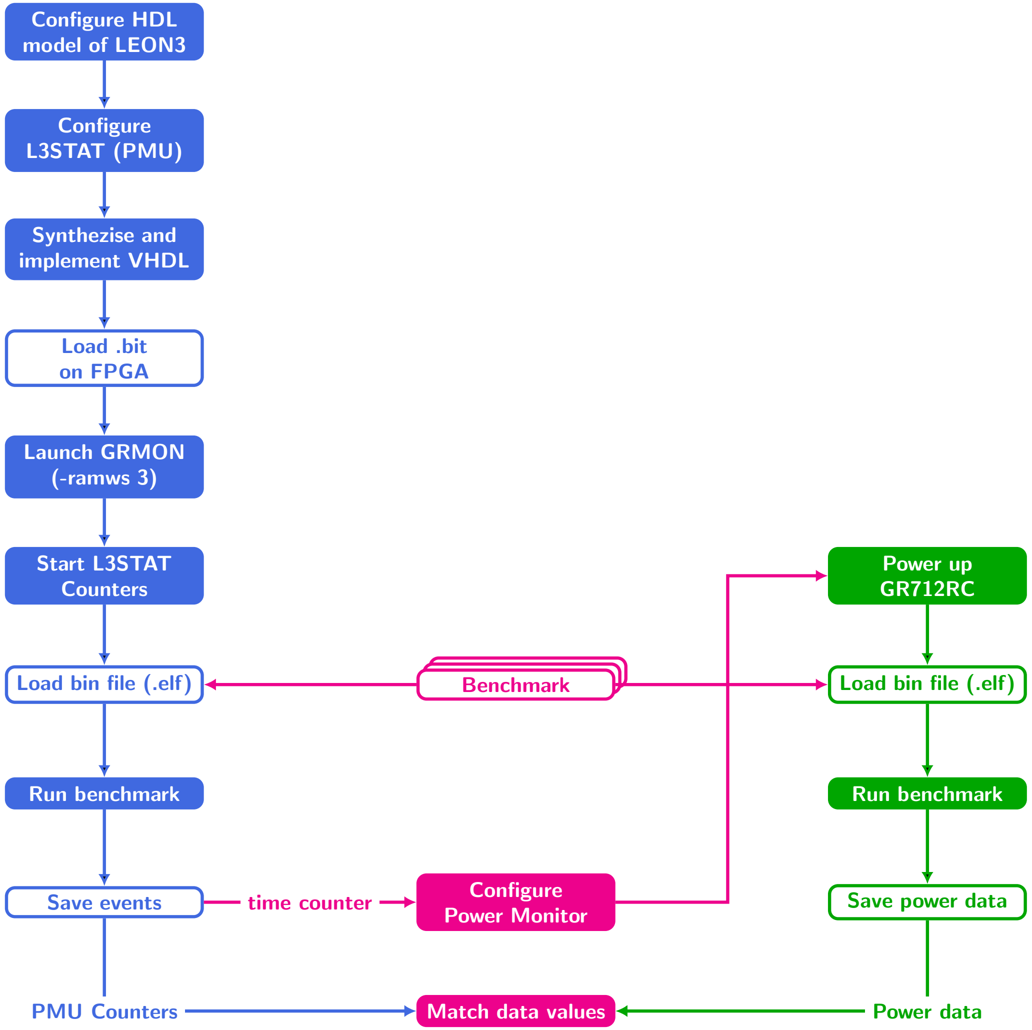
- Our specific target was the STM32F051R8T6
- This processor does not have a PMU
- We used `thumbulator`, an instruction set simulator, to count PMU events:
  - Executed instructions
  - Executed multiplication instructions
  - RAM reads
  - RAM writes
  - Flash reads
  - Taken branches



Target Architectures – LEON3

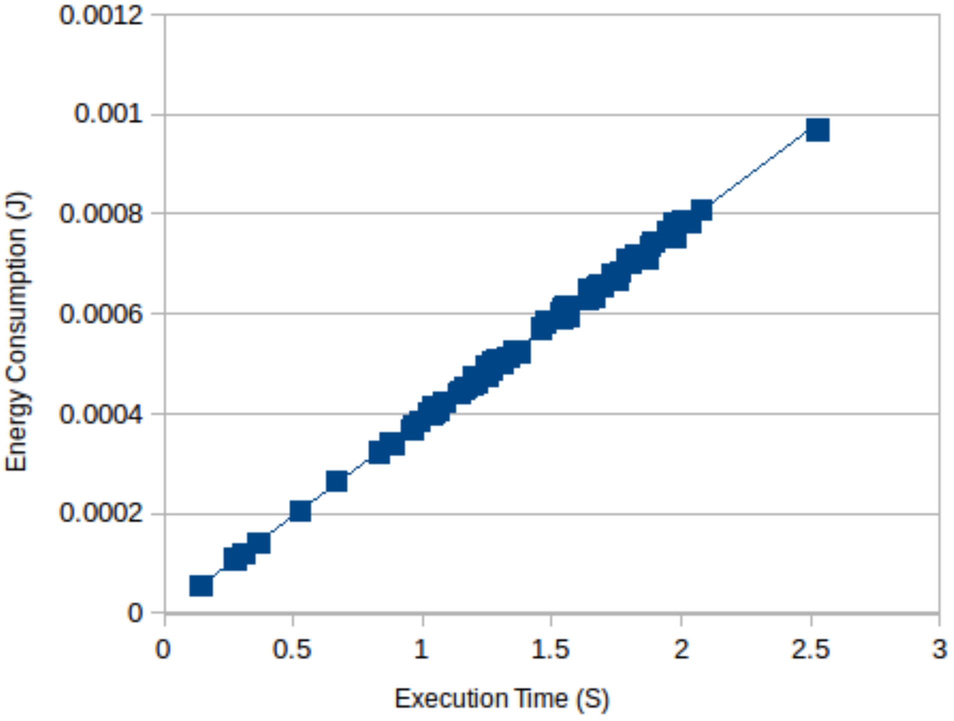
- Cobham Gaisler GR712RC
- A PMU (L3STAT) is available, but not included in this target
- We used a FPGA implementation of the LEON3 including L3STAT in sync with the original target





Energy Models – ARM Cortex-M0

- First energy model: time as proxy for energy [1, 2]
  - Derived using BEEBS benchmarks [3]
- E = 0.0004 W × t



AbsInt

Energy Models – ARM Cortex-M0

- Second energy model: time as proxy for energy [4]
  - Derived using BEEBS benchmarks and Irida Labs‘ CNN benchmarks
  - The CNN benchmarks exercise much **more memory accesses**
- $E = 0.04387\ \mathrm{J} + 0.007242\ \mathrm{W} \times t$

Energy Models – ARM Cortex-M0

- Third energy model: PMU events as proxy for energy [5]
  - Derived using BEEBS benchmarks and Irida Labs‘ CNN benchmarks

$$
\begin{aligned}
E_{\text{Cortex-M0}} = {} & 0.972565030 \times C_{\text{executed instructions without multiplications}} \\
& + 0.652871770 \times C_{\text{RAM data reads}} \\
& + 1.031341343 \times C_{\text{RAM writes}} \\
& + 1.037625441 \times C_{\text{Flash data reads}} \\
& + 1.354953706 \times C_{\text{taken branches}} \\
& + 2.274650563 \times C_{\text{multiplication instructions}}
\end{aligned}
$$

Evaluation – ARM Cortex-M0

Benchmark	Analysis Result	Model Result	Δ	Note
aha-compress	78.885 mJ	78.828 mJ	< 1 %	
aha-mont64	99.396 mJ	99.396 mJ	< 1 %	
bubblesort	366.763 mJ	366.762 mJ	< 1 %	
cnt	42.813 mJ	42.804 mJ	< 1 %	
compress	27.895 mJ	27.895 mJ	< 1 %	
crc	9.623 mJ	9.623 mJ	< 1 %	
cubic	7.801 J	4.138 J	89 %	flow constraints
duff	4.349 mJ	4.349 mJ	< 1 %	
edn	302.762 mJ	302.762 mJ	< 1 %	
expint	43.315 mJ	43.315 mJ	< 1 %	
fac	2.934 mJ	2.904 mJ	1 %	
fasta	29.383 J	21.100 J	39 %	flow constraints
fdct	12.292 mJ	12.292 mJ	< 1 %	
fibcall	1.493 mJ	1.493 mJ	< 1 %	
fir	1.994 J	1.994 J	< 1 %	
frac	1.183 J	1.183 J	< 1 %	
insertsort	3.089 mJ	3.089 mJ	< 1 %	
janne_complex	1.402 mJ	1.402 mJ	< 1 %	
jfdctint	31.481 mJ	31.476 mJ	< 1 %	
lcdnum	886.941 uJ	805.000 mJ	10 %	
levenshtein	400.926 mJ	400.926 mJ	< 1 %	
ludcmp	174.559 mJ	174.559 mJ	< 1 %	
matmult-float	1.537 J	1.537 J	< 1 %	
matmult-int	842.724 mJ	842.649 mJ	< 1 %	
minver	131.316 mJ	84.348 mJ	56 %	flow constraints
nbody	25.844 J	25.844 J	< 1 %	
ndes	293.387 mJ	293.297 mJ	< 1 %	

Benchmark	Analysis Result	Model Result	Δ	Note
nettle-arcfour	105.880 mJ	105.880 mJ	< 1 %	
nettle-cast128	23.214 mJ	23.211 mJ	< 1 %	
nettle-des	22.595 mJ	22.595 mJ	< 1 %	
nettle-md5	5.467 mJ	5.467 mJ	< 1 %	
nettle-sha256	50.507 mJ	50.507 mJ	< 1 %	
newlib-exp	70.439 mJ	70.439 mJ	< 1 %	
newlib-log	52.954 mJ	52.954 mJ	< 1 %	
newlib-sqrt	10.289 mJ	10.289 mJ	< 1 %	
nsichneu	61.017 mJ	29.185 mJ	109 %	
picojpeg	4.885 J	4.885 J	< 1 %	
prime	209.663 mJ	209.663 mJ	< 1 %	
qsort	27.294 mJ	20.408 mJ	34 %	flow constraints
qurt	139.891 mJ	139.890 mJ	< 1 %	
rijndael	7.176 J	7.042 J	2 %	
sglib-arraybinsearch	76.596 mJ	76.596 mJ	< 1 %	
sglib-arrayheapsort	86.857 mJ	86.857 mJ	< 1 %	
sglib-arrayquicksort	65.600 mJ	65.600 mJ	< 1 %	
sglib-queue	126.250 mJ	126.250 mJ	< 1 %	
slre	206.734 mJ	206.734 mJ	< 1 %	
sqrt	11.529 J	11.529 J	< 1 %	
st	4.142 J	2.945 J	41 %	flow constraints
statemate	13.331 mJ	9.308 mJ	43 %	
stb_perlin	5.145 J	5.145 J	< 1 %	
stringsearch1	46.362 mJ	46.362 mJ	< 1 %	
strstr	5.480 mJ	5.480 mJ	< 1 %	
trio-snprintf	105.378 mJ	65.427 mJ	61 %	flow constraints
trio-sscanf	139.345 mJ	71.618 mJ	95 %	flow constraints
ud	21.863 mJ	21.862 mJ	< 1 %	
whetstone	22.533 J	16.687 J	35 %	flow constraints

Table 1: Evaluation of the energy model integration into EnergyAnalyzer for ARM Cortex-M0

Energy Models – ARM Cortex-M0

- Even more energy models: PMU events as proxy for energy
  - The ARM Cortex-M0 allows a diversity of system configurations:
    - Frequency can bei either 20, 24, or 48 MHz
    - The instruction prefetch buffer can be enabled or not
    - Flash memory can be accessed with 0 or 1 waitstate
  - In total, the processor manual permits 10 combinations

Hardware Config.	Energy Consumption Model [nJ]	Meas. Energy[J]	MAPE [%]
[20, OFF, 0]	$E = 0.964258 \times C_1 + 1.652455 \times C_2 + 2.091986 \times C_3 + 1.109833 \times C_4 + 0.650563 \times C_5 + 0.633621 \times C_6$	221.4	2.80
[20, OFF, 1]	$E = 1.282474 \times C_1 + 2.110668 \times C_2 + 2.191545 \times C_3 + 1.185609 \times C_4 + 0.416602 \times C_5 + 1.178991 \times C_6$	274.9	2.97
[20, ON, 0]	$E = 1.003378 \times C_1 + 1.885309 \times C_2 + 1.802974 \times C_3 + 1.122833 \times C_4 + 0.849223 \times C_5 + 0.475831 \times C_6$	226.38	2.86
[20, ON, 1]	$E = 0.895879 \times C_1 + 2.185851 \times C_2 + 2.001178 \times C_3 + 1.493364 \times C_4 + 1.076354 \times C_5 + 1.573758 \times C_6$	227.9	3.68
[24, OFF, 0]	$E = 0.959172 \times C_1 + 1.888565 \times C_2 + 1.357556 \times C_3 + 1.089427 \times C_4 + 0.993145 \times C_5 + 0.562952 \times C_6$	214.62	3.22
[24, OFF, 1]	$E = 1.178558 \times C_1 + 2.540429 \times C_2 + 2.042475 \times C_3 + 1.190892 \times C_4 + 0.979651 \times C_5 + 0.891088 \times C_6$	264.88	3.16
[24, ON, 0]	$E = 0.985415 \times C_1 + 1.933276 \times C_2 + 1.448160 \times C_3 + 1.075671 \times C_4 + 1.011891 \times C_5 + 0.617510 \times C_6$	220.03	3.36
[24, ON, 1]	$E = 0.883755 \times C_1 + 2.156046 \times C_2 + 1.633465 \times C_3 + 1.436556 \times C_4 + 1.152560 \times C_5 + 1.455166 \times C_6$	220.05	4.15
[48, OFF, 1]	$E = 1.096677 \times C_1 + 2.364495 \times C_2 + 1.627854 \times C_3 + 1.173680 \times C_4 + 0.681475 \times C_5 + 0.652665 \times C_6$	243.44	3.65
[48, ON, 1]	$E = 0.816331 \times C_1 + 2.014612 \times C_2 + 1.372157 \times C_3 + 1.402116 \times C_4 + 0.835035 \times C_5 + 1.250446 \times C_6$	202.5	4.33

TABLE II: Energy models for selected Cortex-M0 hardware configurations – Hardware Configuration Format: [Frequency (MHz), PreFetch (ON/OFF), WaitState (0/1)] and MAPE: Mean Absolute Percentage Error

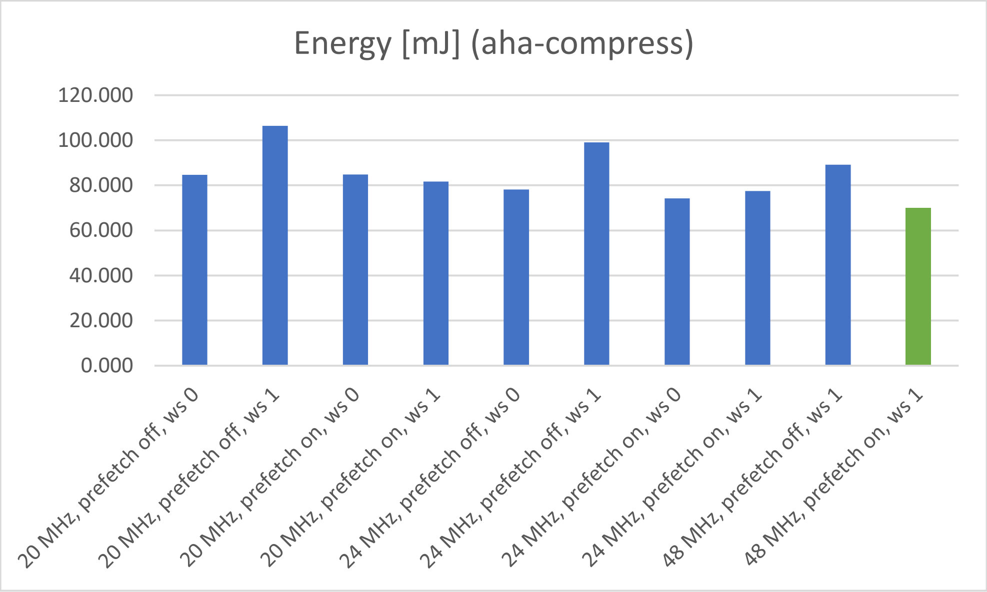
(taken from [6])

Energy Models – ARM Cortex-M0

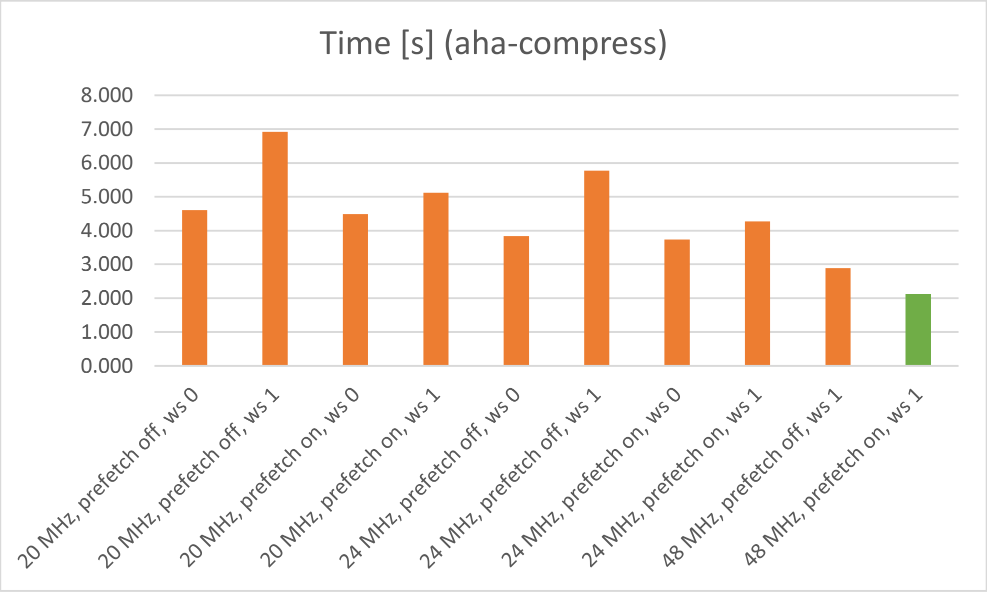
- EnergyAnalyzer allows to specify an energy model based on PMU counters

```
# EnergyAnalyzer specific attributes, all values have been rounded up to full pJ.
attribute "arm_event_energy_costs": {
    "instruction_fetch" = 973,
    "ram_data_write" = 1032,
    "ram_data_read" = 653,
    "flash_data_read" = 1038,
    "mul_instruction" = 1303,
    "taken_branch" = 1355
};
```

Energy Models – ARM Cortex-M0



(a) Comparison of different hardware configurations concerning energy usage



(b) Comparison of different hardware configurations concerning execution time

LEON3

- LEON3 PMU (L3STAT) has many available counters, not all of them being statically predictable
- Energy model has been build using the *ISA+Cache* subset of PMU counters

$$E_{\text{LEON3}} = 3.93365 \times 10^{-08} \times C_{\text{integer instructions}} + 1.87111 \times 10^{-07} \times C_{\text{store instructions}}$$

#	**Counter**	Description	#	**Counter**	Description
C_1	ICMISS	instruction cache misses	C_{13}	TYPE2	type 2 instructions
C_3	DCMISS	data cache misses	C_{14}	LDST	load and store instructions
C_7	IINST	integer instructions	C_{15}	LOAD	load instructions
C_{11}	BRANCH	branch instructions	C_{16}	STORE	store instructions
C_{12}	CALL	call instructions			

Table 2: *ISA+Cache* subset of L3STAT counters

AbsInt

LEON3

- Other subsets are viable, too
  - However, the static prediction of those are less precise

Model	Expression	MAPE[%]	
		Train	Test
Energy [J] All Supported All Events	$E = 0.155261 + 2.94155\text{e-}08 \times C_0 + 2.5661\text{e-}09 \times C_2 + 9.93453\text{e-}09 \times C_5 + 8.97535\text{e-}10 \times C_{12} + 3.21255\text{e-}09 \times C_{13} + 6.14384\text{e-}09 \times C_{15} + 4.54827\text{e-}08 \times C_{16}$	1.14	0.29
Energy [J] All Supported Bottom-Up	$E = 0 + 3.19557\text{e-}08 \times C_0 + 5.79224\text{e-}08 \times C_{16}$	1.20	1.38
Energy [J] All Supported Top-Down	$E = 0.131077 + 3.13122\text{e-}08 \times C_0 + 9.17778\text{e-}09 \times C_5 + 2.99043\text{e-}09 \times C_{15} + 3.92999\text{e-}08 \times C_{16}$	1.02	1.54
Energy [J] All Supported Full-Exhaustive	$E = 0.131087 + 3.13122\text{e-}08 \times C_0 + 9.17779\text{e-}09 \times C_5 + 2.99043\text{e-}09 \times C_{14} + 3.63095\text{e-}08 \times C_{16}$	1.02	1.54
Energy [J] IsaCache All Events	$E = 0 + 1.18567\text{e-}06 \times C_3 + 5.9072\text{e-}07 \times C_{12} + 3.88949\text{e-}08 \times C_{13} + 8.03337\text{e-}08 \times C_{14} + 6.89885\text{e-}08 \times C_{16}$	8.38	24.03
Energy [J] IsaCache Bottom-Up	$E = 0 + 3.93365\text{e-}08 \times C_7 + 1.87111\text{e-}07 \times C_{16}$	5.84	8.24
Energy [J] IsaCache Top-Down	$E = 0 + 3.93365\text{e-}08 \times C_7 + 1.87111\text{e-}07 \times C_{16}$	5.84	8.24
Energy [J] IsaCache Full-Exhaustive	$E = 0 + 3.93365\text{e-}08 \times C_7 + 1.87111\text{e-}07 \times C_{16}$	5.84	8.24

Table 3: Different energy models for LEON3 based on two different subsets of PMU counters and different search strategies

Evaluation – LEON3

Benchmark	Analysis Result	Model Result	Δ	Note
aha-compress	11.004 J	11.004 J	0 %	
aha-mont64	7.499 J	7.491 J	< 1 %	
bubblesort	3.898 J	3.889 J	< 1 %	
edn	39.186 J	39.186 J	0 %	
fir	159.469 J	159.469 J	0 %	
frac	59.391 J	59.339 J	< 1 %	
levenshtein	25.506 J	25.491 J	< 1 %	
ludcmp	10.992 J	10.814 J	2 %	
matmult-float	2.847 J	2.822 J	1 %	
minver	14.372 J	4.643 J	210 %	worst-case
minver	7.398 J	4.643 J	59 %	assumptions
nbody	4.512 J	4.496 J	< 1 %	
ndes	24.828 J	24.467 J	1 %	

Benchmark	Analysis Result	Model Result	Δ	Note
nettle-aes	19.401 J	19.389 J	< 1 %	
nettle-arcfour	9.644 J	9.639 J	< 1 %	
nettle-sha256	2.763 J	2.754 J	< 1 %	
newlib-exp	4.374 J	4.319 J	1 %	
newlib-log	3.284 J	3.252 J	1 %	
picojpeg	503.732 J	503.918 J	< -1 %	traps
prime	3.670 J	3.667 J	< 1 %	
qurt	8.001 J	7.958 J	1 %	
sglib-arraybinsearch	6.283 J	6.281 J	< 1 %	
sglib-arrayheapsort	13.066 J	13.062 J	< 1 %	
sglib-arrayquicksort	13.066 J	13.052 J	< 1 %	
sglib-queue	13.901 J	13.900 J	< 1 %	
slre	14.988 J	15.261 J	-2 %	traps

Table 4: Evaluation of the *ISA+Cache* energy model integration into EnergyAnalyzer for LEON3

Emulating Floating-point Operations in Software

- The IEEE 754 standard for floating-point arithmetic defines data formats for floating-point numbers
  - Single precision floating-point number consists of one sign bit, eight bits for the exponent, and 24 bits for the mantissa
  - Only 23 bits of the mantissa are explicitly stored
  - The first bit of the mantissa is implicitly stored and assumed to be 1 for normalised numbers, and 0 for subnormal numbers
  - Subnormal numbers are used to represent numbers near zero with tiny absolute value, which cannot be represented as normalised numbers

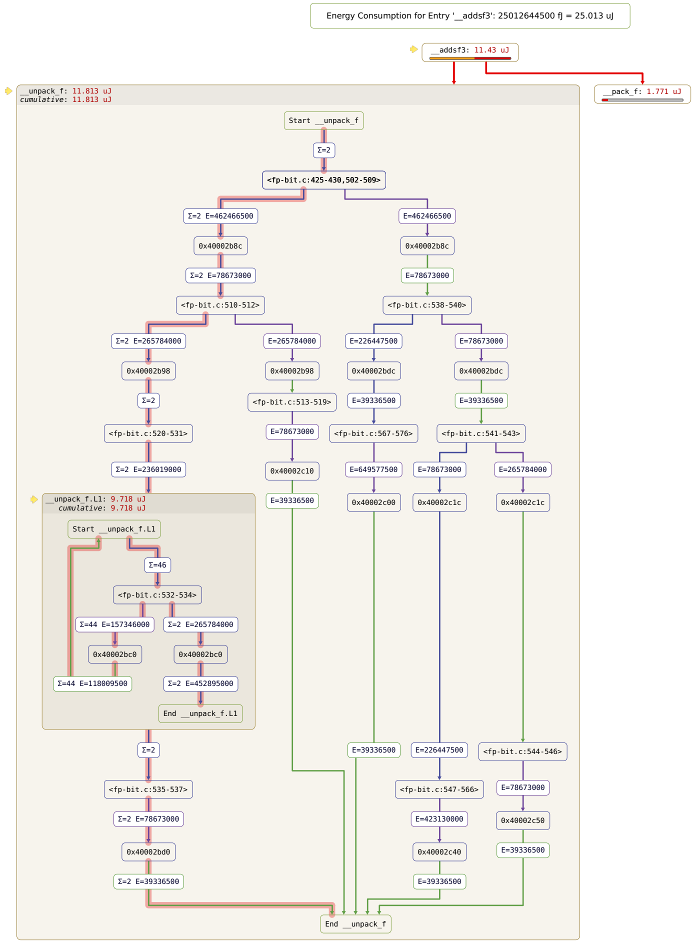
Emulating Floating-point Operations in Software

- Operations on subnormal numbers are usually more costly than operations on normalised numbers because they need to be scaled first before the actual operation is performed
  - Knowing whether a floating-point number is normalised or not has a huge impact on the analysis precision

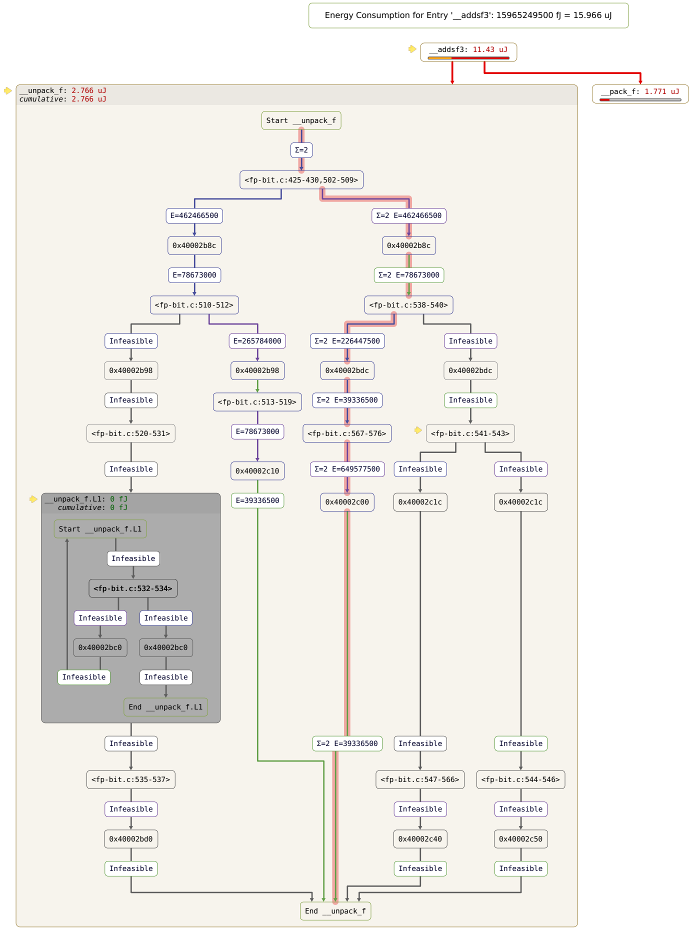
Routine	Worst-case Result	Result under Assumptions
`__mulsf3`	40.167 µJ	9.839 µJ
`__addsf3`	25.013 µJ	15.966 µJ
`__subsf3`	25.279 µJ	16.232 µJ
`__divsf3`	29.753 µJ	20.518 µJ
`__ieee754_sqrt`	180.879 µJ	113.977 µJ
`__muldf3`	85.228 µJ	14.225 µJ
`__adddf3`	76.141 µJ	42.784 µJ
`__subdf3`	76.367 µJ	43.010 µJ
`__divdf3`	103.489 µJ	69.945 µJ

Table 2.12: Analysis results using the *ISA+Cache* energy model for some routines of the software floating-point library. (taken from [7])

Emulating Floating-point Operations in Software



(a) Worst-case analysis of a software floating-point routine



(b) Assuming only normals and zero

Conclusion

- EnergyAnalyzer for ARM Cortex-M0 and LEON3 ...
  - are powerful tools for the development of embedded systems with energy constraints
  - allow to make informed decisions regarding hardware and compiler configurations
  - are versatile and allow to adapt the used energy model to the concrete platform
- Both tools have been integrated into the TeamPlay toolchain and successfully applied to various use cases [9]:
  - Multi-criterial optimization in WCC
  - Contract-based programming with CSL
  - Camera pill, CNN kernels, satellite software

Bibliography

[1] TeamPlay Consortium. *Deliverable D4.1.* 2018

[2] TeamPlay Consortium. *Deliverable D4.2.* 2018

[3] James Pallister, Simon J. Hollis, and Jeremy Bennett. *BEEBS: open benchmarks for energy measurements on embedded platforms.* 2013

[4] TeamPlay Consortium. *Deliverable D4.3.* 2019

[5] TeamPlay Consortium. *Deliverable D4.4.* 2019

[6] Kris Nikov, Kyriakos Georgiou, Zbigniew Chamski, Kerstin Eder, and José L. Núñez-Yáñez. *Accurate Energy Modelling on the Cortex-M0 Processor for Profiling and Static Analysis.* 2022

[7] TeamPlay Consortium. *Deliverable D4.5.* 2020

[8] Kris Nikov, Marcos Martínez, Simon Wegener, José L. Núñez-Yáñez, Zbigniew Chamski, Kyriakos Georgiou, and Kerstin Eder. *Robust and accurate fine-grain power models for embedded systems with no on-chip PMU*. 2022

[9] Benjamin Rouxel, et al. *The TeamPlay project: Analysing and optimising time, energy, and security for cyber-physical systems.* 2023

[10] Zbigniew Chamski et al. *Thumbulator.* https://github.com/PicoPET/thumbulator-stm32f0x.git

email: info@absint.com

http://www.absint.com