HERIOT WATT UNIVERSITY

Automatic Approximation for 1-Dimensional Feedback-Loop Computations: a PID Benchmark

Yun Wu, Yun Zhang, Anis Hamadouche,

João F. C. Mota, Andrew M. Wallace

14/09/2022

LEADERS IN IDEAS AND SOLUTIONS

[dstl] The Science Inside

UDRC

UKRI Engineering and Physical Sciences Research Council

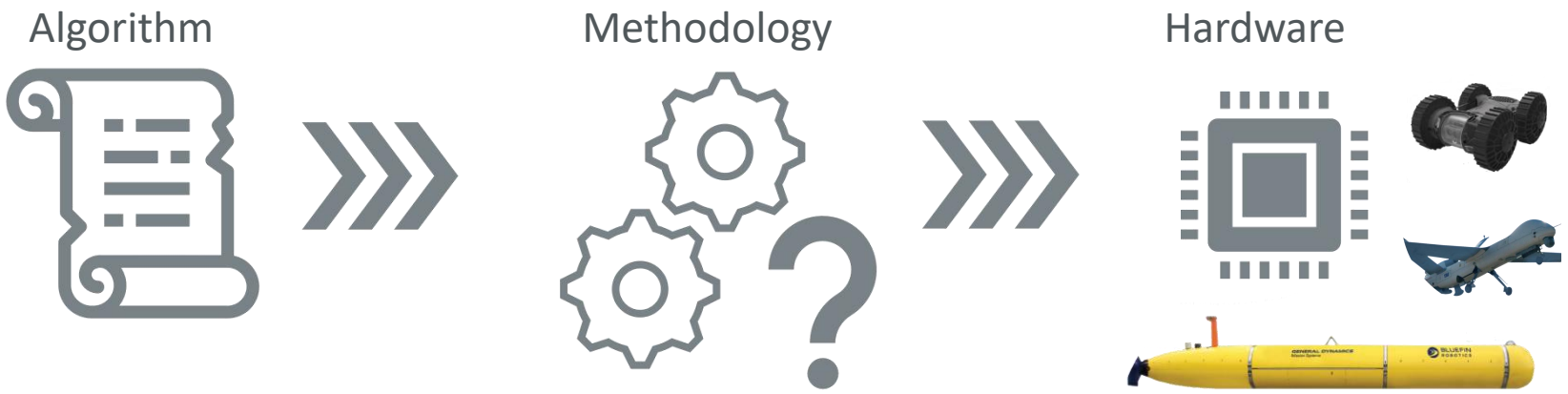
HERIOT WATT UNIVERSITY

Motivation

PID (Proportional-Integrative-Derivative) control is widely adopted in robotics and autonomous systems in both the commercial and defence sectors.

(Research Question) How can we automate the estimation of the necessary precision for PID control in power-constrained systems?

How to design and implement approximate PID control on an FPGA, for significant resource/power savings.



Affine Arithmetic (AA)

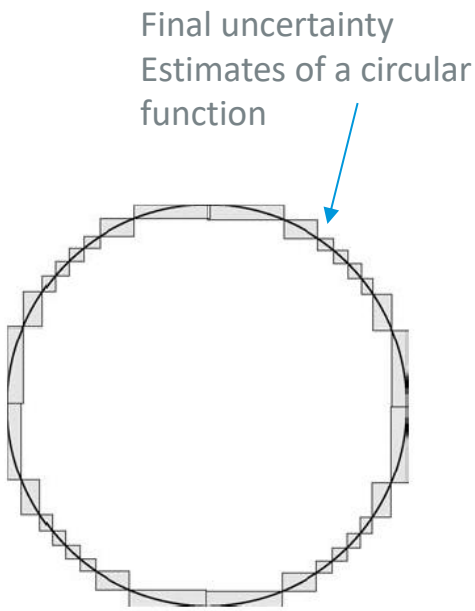
Affine arithmetic (AA) is one of many proposed models for function self-validation. In AA, the uncertainty of a variable a can be represented as

$\hat{a} = a_0 + \sum_{i=1}^{n} a_i \cdot \epsilon_i$,

where ϵ_i is an interval $[-1,1]$, a_i is a partial deviation,

$and\ a_i \cdot \epsilon_i$ represents the range of uncertainty

caused by i^{th} incidence (source of error).



Advantages:

- uncertainty cancellation
- avoids catastrophic overestimation of errors
- potentially long-iterative support

In our work, we use the YalAA library, because

- It uses the C++ programming language, easy to integrate with our framework
- non-affine forms approximate

PID Controller

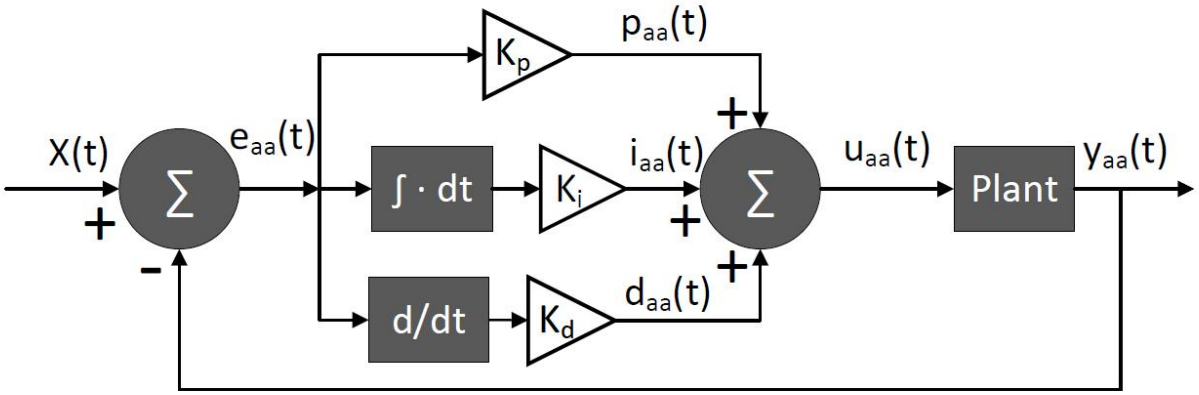
Proportional-Integrative-Derivative (PID) controller Definition:

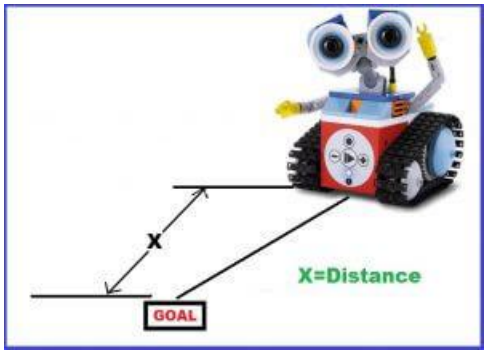
$$u_{aa}(t) = K_p e_{aa}(t) + K_i \int e_{aa}(t)dt + K_d \frac{de_{aa}(t)}{dt},$$

where K_p, K_i, and K_d are the coefficients of the proportional, integral and derivative control processes.

Given the reference input $X(t)$, the error term $e_{aa}(t)$ is derived by subtracting the measured plant output $Y_{aa}(t)$ from $X(t)$.

This is repeated until the plant output matches the reference.



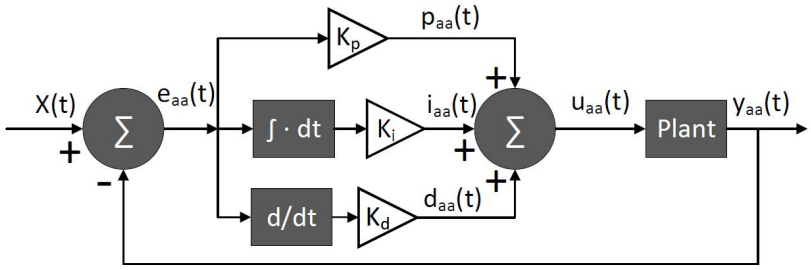


PID Benchmarks

We consider nine standard benchmarks for the Plant Transfer Function

For each benchmark, Table 1 shows the uncertainty intervals, [lb, ub]. of each parameter in the PID control process, using full precision, when using AA.

Plant Transfer Function	e_{aa}	p_{aa}	i_{aa}	d_{aa}	u_{aa}	y_{aa}
c1: $\frac{1}{s+1}$	[-0.001705, 0.001238]	[-0.001705, 0.001238]	[0.384654, 0.387893]	[-0.007070, 0.003848]	[0.997140, 1.006484]	[0.998771, 1.001712]
c2: $\frac{1}{(0.1s+1)(s+1)}$	[-0.010378, 0.020331]	[-0.010378, 0.020331]	[0.352566, 0.359014]	[-0.036406, 0.014746]	[0.994849, 1.016622]	[0.979850, 1.010305]
c3: $\frac{1}{(0.01s+1)(0.1s+1)(s+1)}$	[-0.002425, 0.001583]	[-0.002425, 0.001583]	[0.365121, 0.370106]	[-0.009494, 0.005258]	[0.996281, 1.007964]	[0.998431, 1.002432]
c4: $\frac{1-0.1s}{(s+1)^3}$	[-0.006933, 0.006462]	[-0.006933, 0.006462]	[1.183874, 5.947544]	[-0.002716, 0.004606]	[0.999921, 5.001432]	[0.993552, 1.00691]
c5: $\frac{1}{(0.1s+1)}e^{-s}$	[-0.004284, 0.008109]	[-0.004284, 0.008109]	[1.314815, 1.349334]	[-0.004763, 0.002516]	[0.982410, 1.004009]	[0.991915, 1.004271]
c6: $\frac{1}{(0.1s+1)^2}e^{-s}$	[-0.005145, 0.011130]	[-0.005145, 0.011130]	[1.400613, 1.443956]	[-0.006247, 0.002888]	[0.980136, 1.004560]	[0.988901, 1.005131]
c7: $\frac{100}{(s+10)^2}(\frac{1}{(s+1)}+\frac{0.5}{(s+0.05)})$	[0.004182, 0.005610]	[0.004182, 0.005610]	[5.336462, 15.987360]	[-0.000419, 0.000166]	[0.091070, 0.271908]	[0.994390, 0.995820]
c8: $\frac{1}{(s+1)(s^2+0.2s+1)}$	[-0.071520, 0.004774]	[-0.071520, 0.004774]	[8.021174, 8.025332]	[-0.000687, 0.008533]	[0.999168, 0.999939]	[0.995228, 1.071478]
c9: $\frac{1}{(s^2-1)}$	[-0.000135, 0.000082]	[-0.000135, 0.000082]	[-0.201893, -0.201722]	[-0.000092, 0.000159]	[-1.000015, -0.999961]	[0.999918, 1.000134]



AA output and the adjusted approximation:

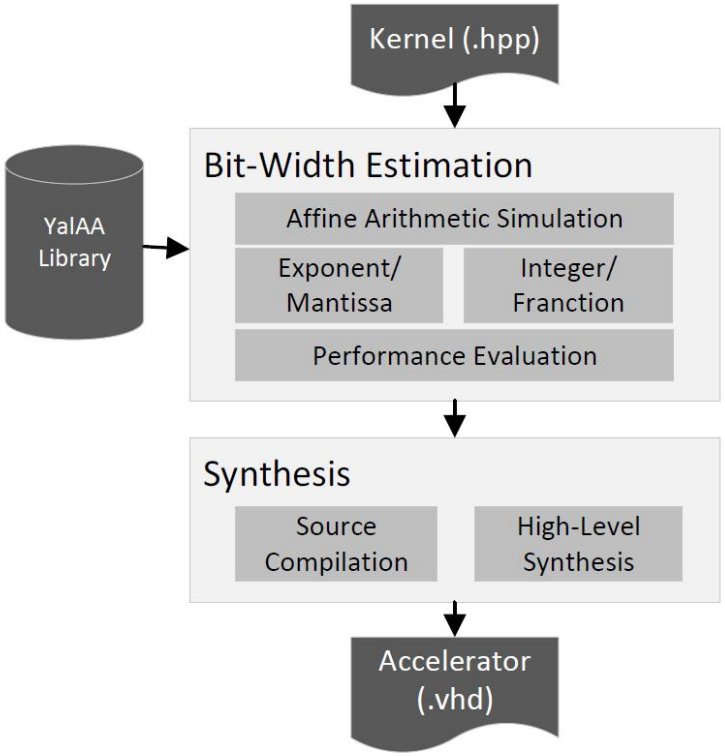
$$[\frac{v-\hat{v}}{v}] < \varepsilon$$

where $v = (ub - lb)/2$ represents the variance of approximate error propagation.

Design Process: Automated Infrastructure

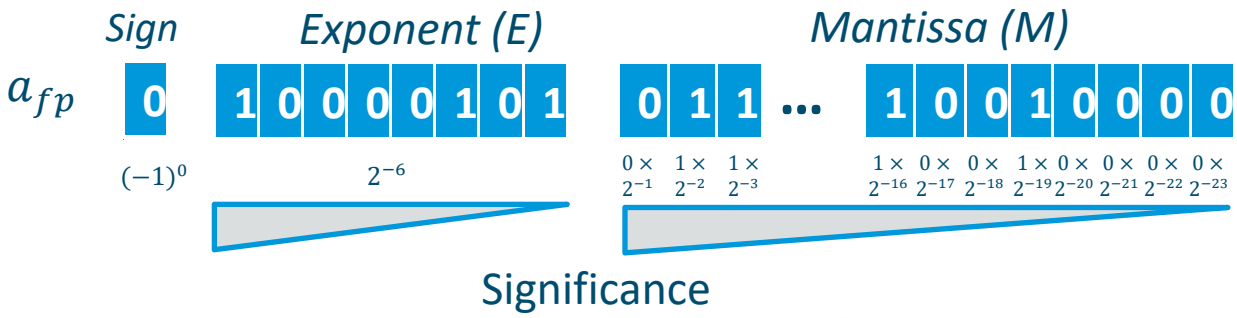
Top-down flow:

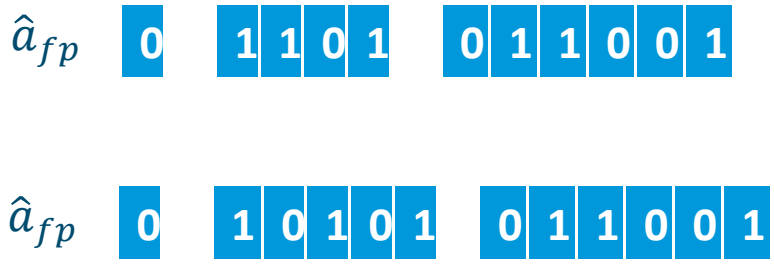
- C++ template kernel (PID)
- Apply AA data type in YalAA1 library
- Bit-width estimation through AA simulation
- Evaluate approximation performance
- Compile the kernel with estimated precision
- Synthesize the code onto a hardware accelerator



Bit-width Estimation: Floating Point

$$a_{fp} = (-1)^S \times M \times 2^{(127-E)}$$





Algorithm 1 Floating-Point Bit-Width Estimation

Input: υ

Output: $\hat{M}, \hat{E}, \hat{bw_m}, \hat{bw_e}$

1: $\hat{bw_e} \leftarrow \arg\min_{\hat{bw_e}}((2^{\hat{bw_e}-1} - 1) - (127 - E)) > 0$

2: $\hat{bw_e} \leftarrow \hat{bw_e} + 1$

3: $\hat{E} \leftarrow (2^{\hat{bw_e}-1} - 1) - (127 - E)$

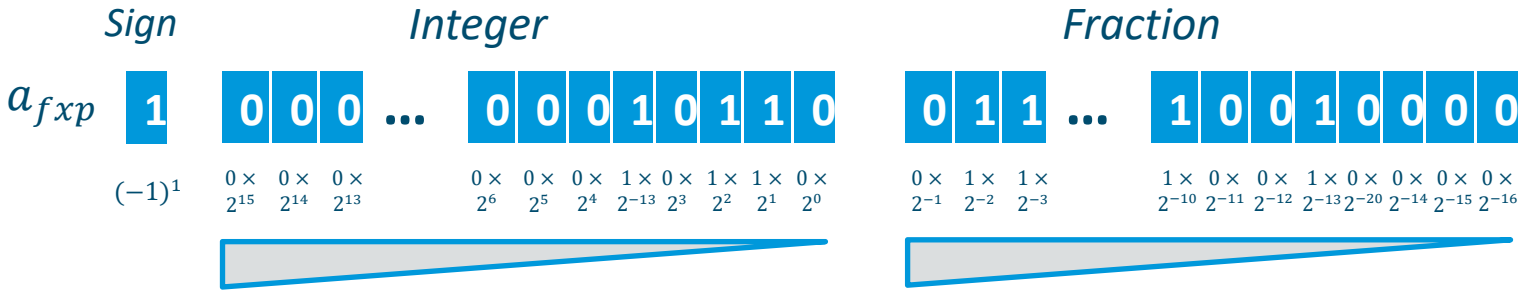
4: $\hat{bw_m} \leftarrow \arg\max_{\hat{bw_m}} \left\| \frac{\upsilon_{fp} - \hat{\upsilon}_{fp}}{\upsilon_{fp}} \right\|_1 \leq \omega,$

5: $\hat{M} = (M >> \hat{bw_m}) \& (2^{(\hat{bw_m}+1)} - 1)$

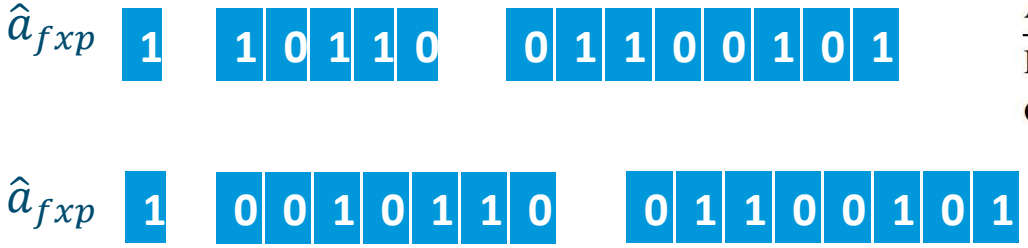
HERIOT WATT UNIVERSITY

Bit-width Estimation: Fixed Point

$$a_{fxp} = (-1)^S \times (2^I + 2^{-F})$$



Significance



Algorithm 2 Fixed-Point Bit-Width Estimation

Input: υ

Output: $\hat{I}, \hat{F}, b\hat{w}_i, b\hat{w}_f$

1: $b\hat{w}_i \leftarrow \arg\min_{b\hat{w}_i}(2^{b\hat{w}_i - 1} - I) > 0$

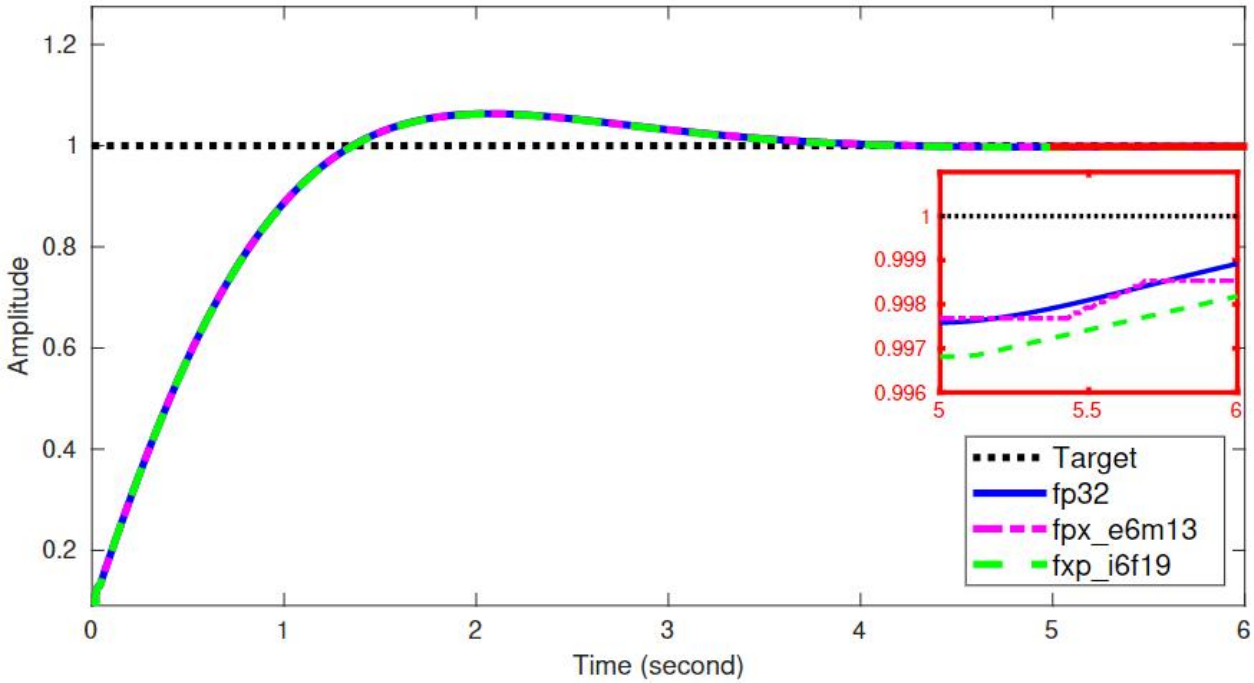
2: $b\hat{w}_i \leftarrow b\hat{w}_i + 2$

3: $b\hat{w}_f \leftarrow \arg\max_{b\hat{w}_f} \left\| \frac{\upsilon_{fxp} - \hat{\upsilon}_{fxp}}{\upsilon_{fxp}} \right\|_1 \leq \omega,$

4: $\hat{F} = (F >> b\hat{w}_f) \& (2^{(b\hat{w}_f + 1)} - 1)$

HERIOT WATT UNIVERSITY

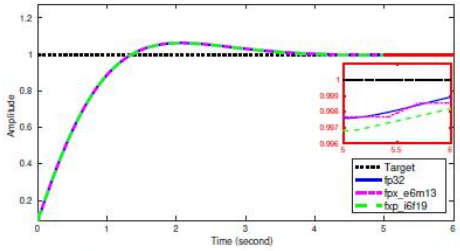
Approximate PID Performance: one benchmark



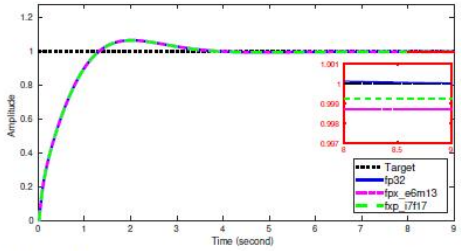
- Fp32 – single precision floating point
- Fpx_e6m13 – 6 bits exponent, 13 bits mantissa
- Fxp_i6f19 – 6 bits integer, 19 bits fraction

HERIOT WATT UNIVERSITY

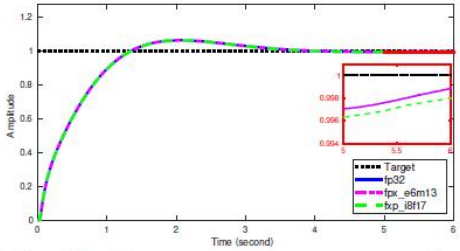
Approximate PID Performance: all benchmarks



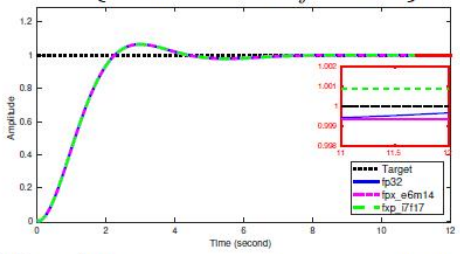
(a) c1:$\{bw_e = 6, bw_m = 13\}$, $\{bw_i = 6, bw_f = 19\}$



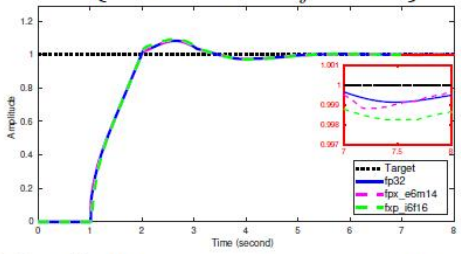
(b) c2:$\{bw_e = 6, bw_m = 13\}$, $\{bw_i = 7, bw_f = 17\}$



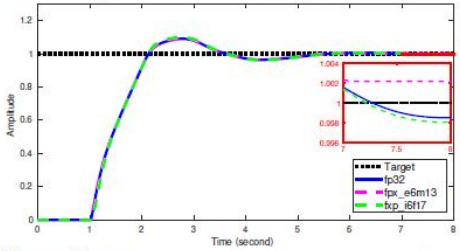
(c) c3:$\{bw_e = 6, bw_m = 13\}$, $\{bw_i = 8, bw_f = 17\}$



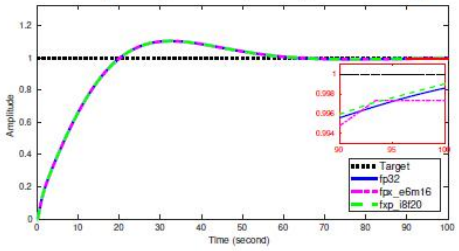
(d) c4:$\{bw_e = 6, bw_m = 14\}$, $\{bw_i = 7, bw_f = 17\}$



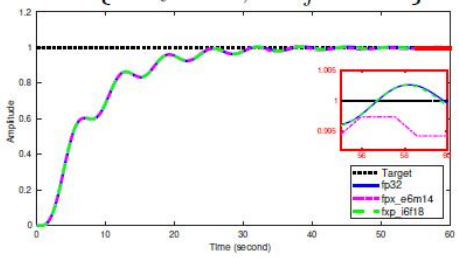
(e) c5:$\{bw_e = 6, bw_m = 14\}$, $\{bw_i = 6, bw_f = 16\}$



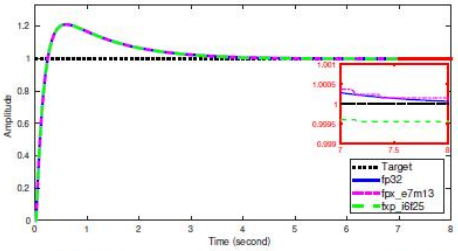
(f) c6:$\{bw_e = 6, bw_m = 13\}$, $\{bw_i = 6, bw_f = 17\}$



(g) c7:$\{bw_e = 6, bw_m = 16\}$, $\{bw_i = 8, bw_f = 20\}$



(h) c8:$\{bw_e = 6, bw_m = 14\}$, $\{bw_i = 6, bw_f = 18\}$

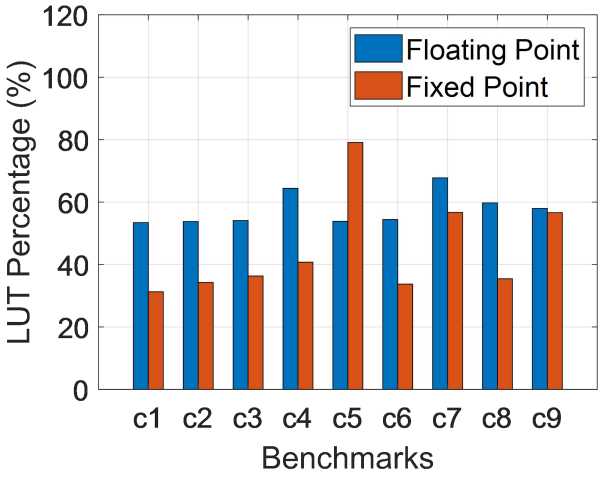


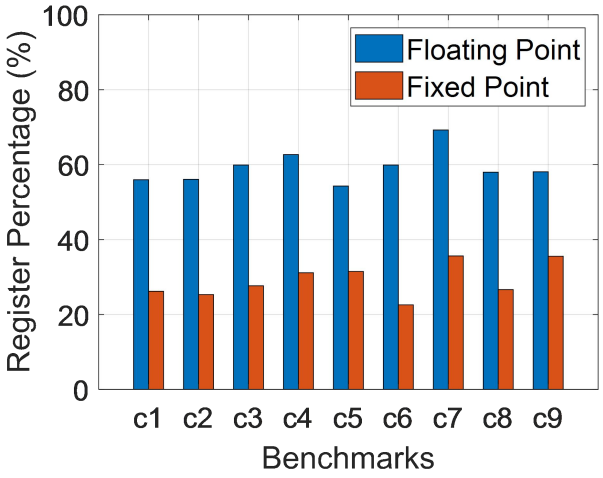
(i) c9:$\{bw_e = 7, bw_m = 13\}$, $\{bw_i = 6, bw_f = 25\}$

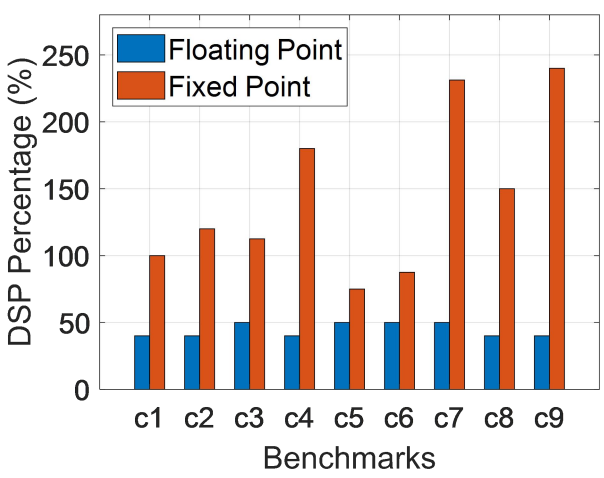
HERIOT WATT UNIVERSITY

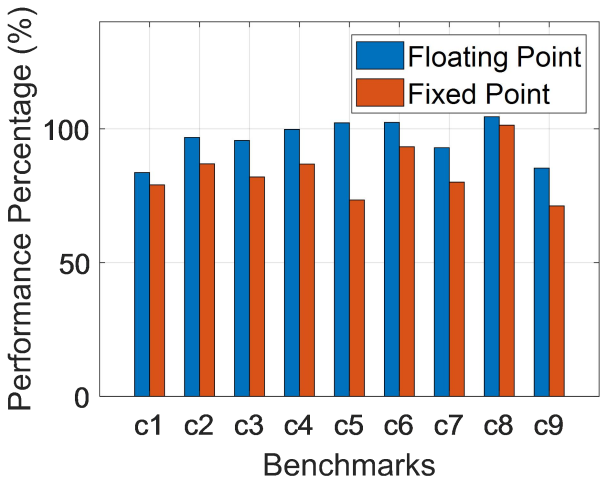
Approximation: Resources and Performance

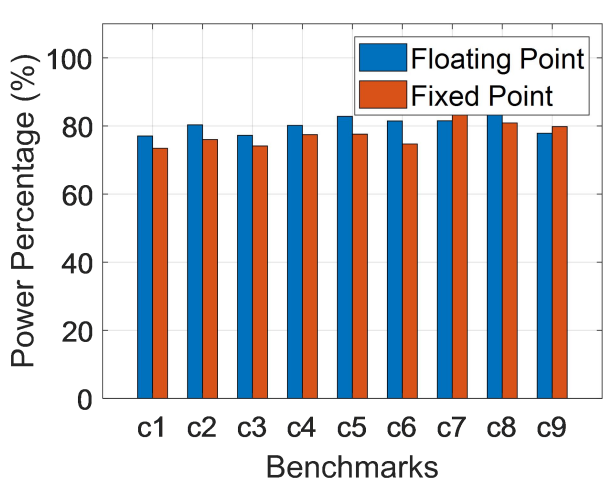
All implemented PID benchmarks are compared to single precision floating point.











HERIOT WATT UNIVERSITY

Summary

- Automated framework for optimal precision estimation for both floating-point and fixed-point binary formats
- Custom accelerator generation on a Xilinx FPGA platform, specifically an approximate PID controller, leads to reductions in both resource cost and power consumption
- Our work is potentially applicable to multi-dimensional iterative applications, such as a multi-variable PID controller and iterative solvers for convex optimization