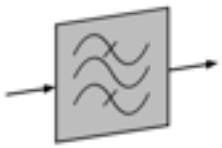
Towards reliable implementation of Digital Filters

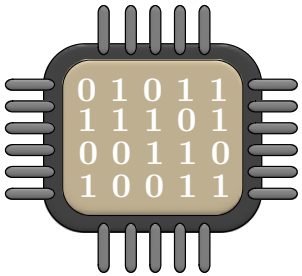
Anastasia Volkova, Thibault Hilaire, Christoph Lauter

Sorbonne Universités, University Pierre and Marie Curie, LIP6,
Paris, France

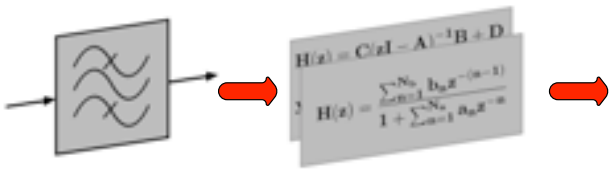
EJCIM
April 5, 2016

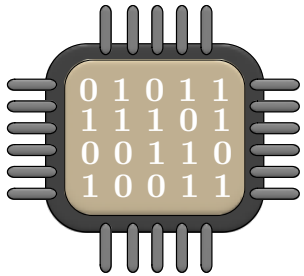
Motivation





Motivation

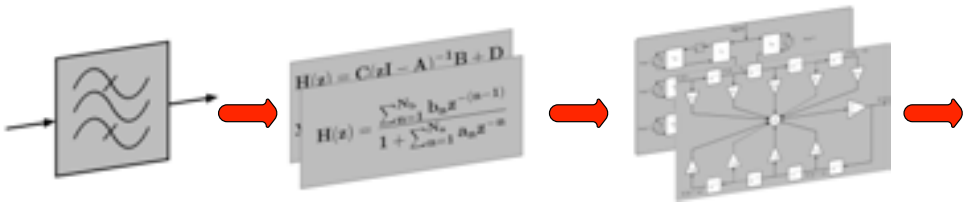


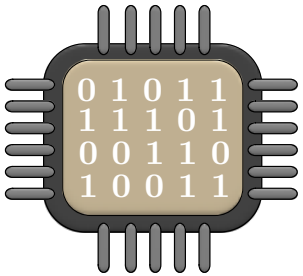


Filter implementation flow:

- Transfer function generation

Motivation

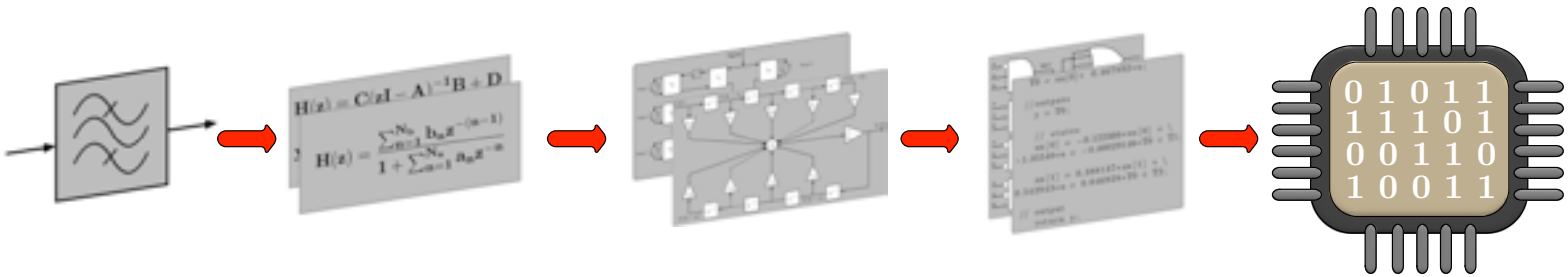




Filter implementation flow:

- Transfer function generation
- State-space, DFI, DFII, ...

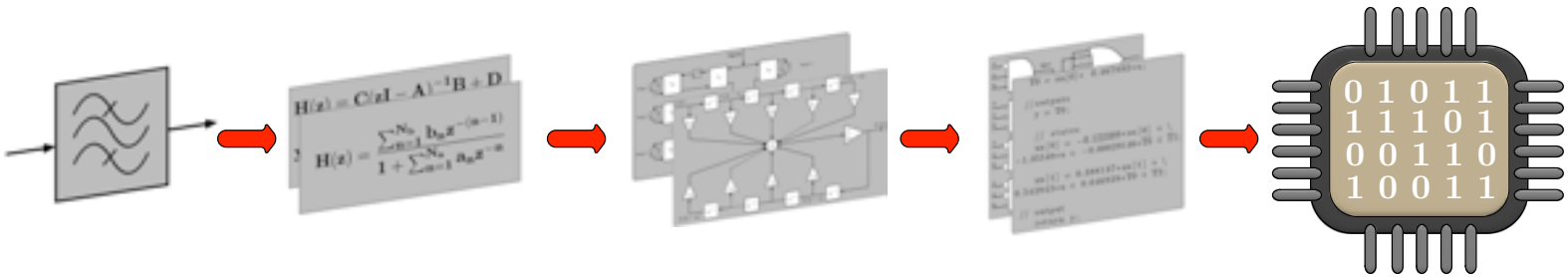
Motivation



Filter implementation flow:

- Transfer function generation
- State-space, DFI, DFII, ...
- Software or Hardware implementation

Motivation

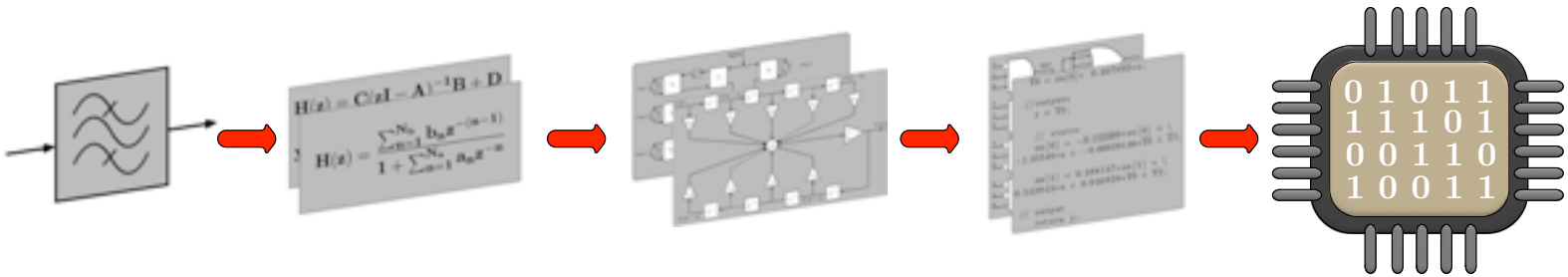


Filter implementation flow:

- Transfer function generation
  - ! Coefficient quantization
- State-space, DFI, DFII, ...

- Software or Hardware implementation

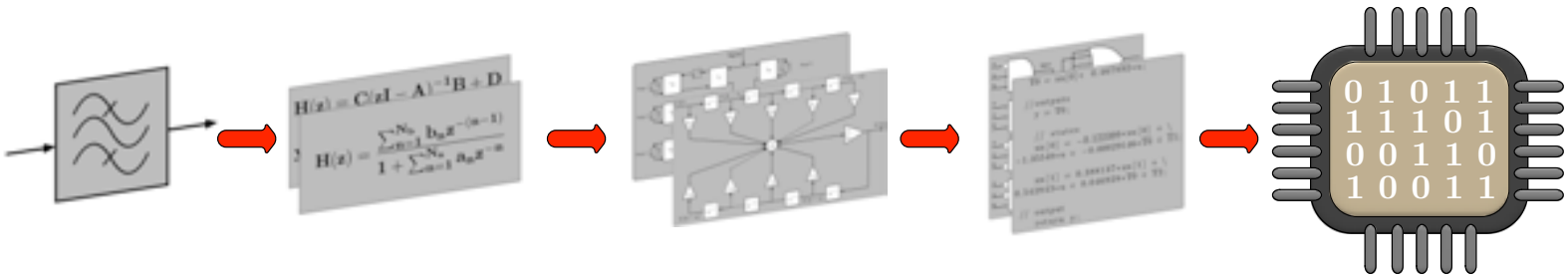
Motivation



Filter implementation flow:

- Transfer function generation
  - ! Coefficient quantization
- State-space, DFI, DFII, ...
  - ! Large variety of structures with no common quality criteria
- Software or Hardware implementation

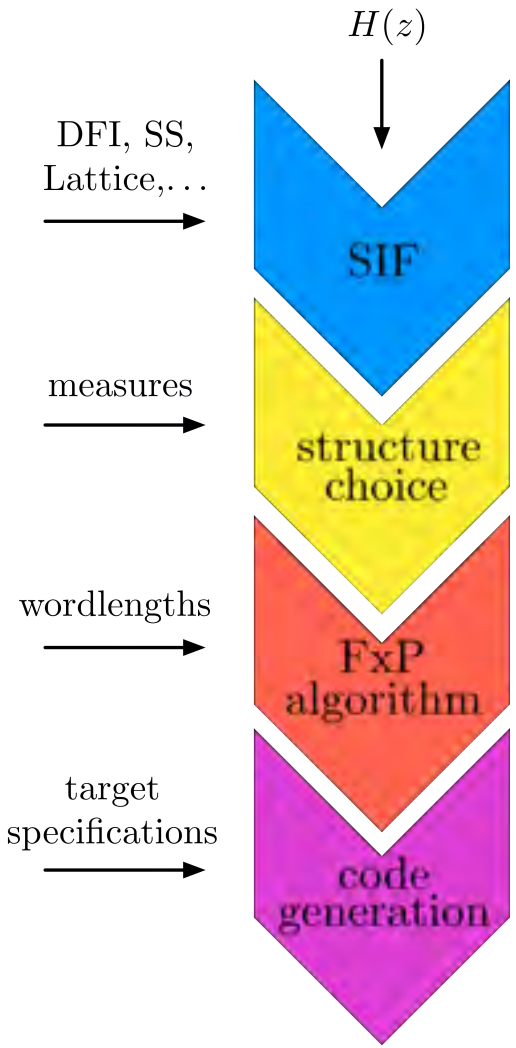
Motivation



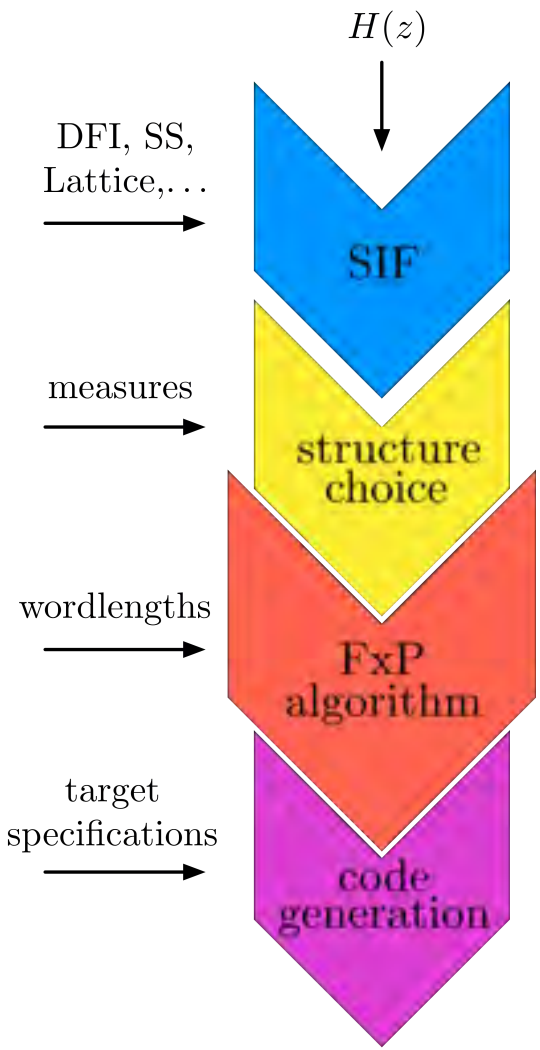
Filter implementation flow:

- Transfer function generation
  - ! Coefficient quantization
- State-space, DFI, DFII, ...
  - ! Large variety of structures with no common quality criteria
- Software or Hardware implementation
  - ! Constraints: power consumption, area, error, speed, etc.

Motivation: Automatized filter implementation flow



Motivation: Automatized filter implementation flow



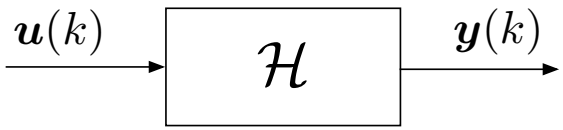
Focus on Fixed-Point realization.
Optimization of wordlengths:

- Take more - pay more
- Take less - overflow risk

What we want in the end:

- Rigorous algorithm for Fixed-Point Formats (FxPF) determination
- No simulations but mathematical proofs
- Integration into automatized code generator for filters

LTI filters



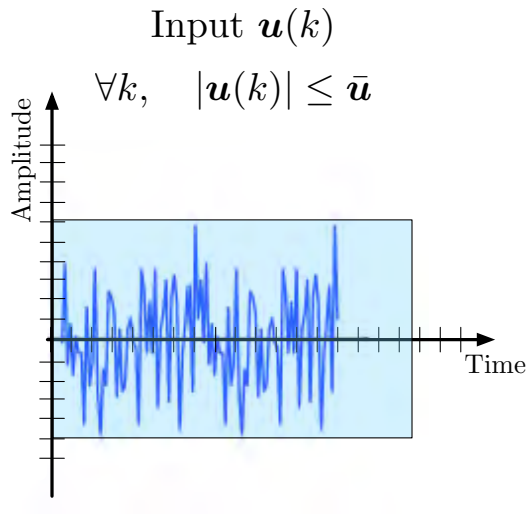
$\mathcal{H} := (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D})$ is a Bounded Input Bounded Output LTI filter in state-space representation:

$$\mathcal{H} \left\{ \begin{array}{rcl} \boldsymbol{x}(k+1) & = & \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{u}(k) \\ \boldsymbol{y}(k) & = & \boldsymbol{C}\boldsymbol{x}(k) + \boldsymbol{D}\boldsymbol{u}(k) \end{array} \right.$$

with q inputs, n states and p outputs and state matrices

Basic brick: interval through filter

The Worst-Case Peak Gain theorem

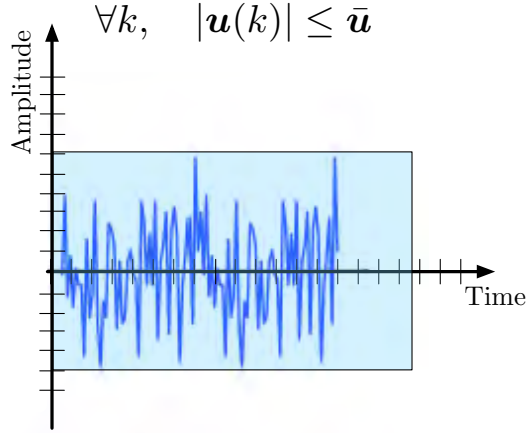


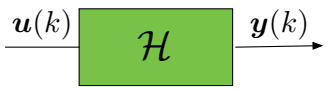
Basic brick: interval through filter

The Worst-Case Peak Gain theorem

Input $\boldsymbol{u}(k)$

$\forall k, \quad |\boldsymbol{u}(k)| \leq \bar{\boldsymbol{u}}$





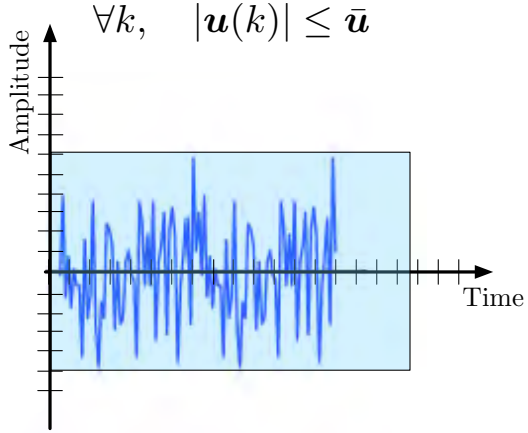
amplification/attenuation

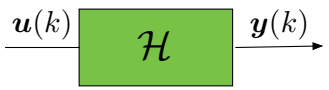
Basic brick: interval through filter

The Worst-Case Peak Gain theorem

Input $\boldsymbol{u}(k)$

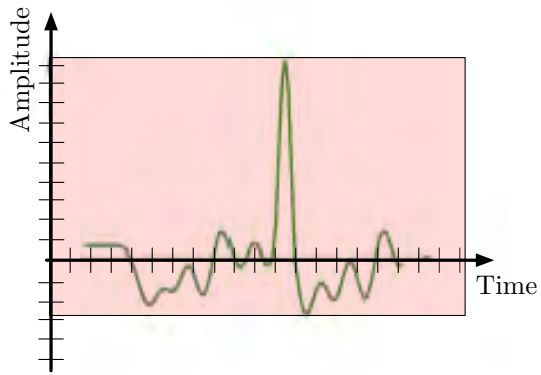
$\forall k, \quad |\boldsymbol{u}(k)| \leq \bar{\boldsymbol{u}}$





amplification/attenuation

Output $\boldsymbol{y}(k)$

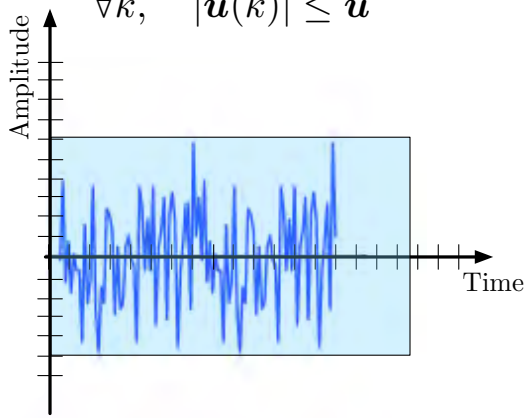


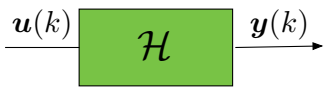
Basic brick: interval through filter

The Worst-Case Peak Gain theorem

Input $\boldsymbol{u}(k)$

$\forall k, \quad |\boldsymbol{u}(k)| \leq \bar{\boldsymbol{u}}$

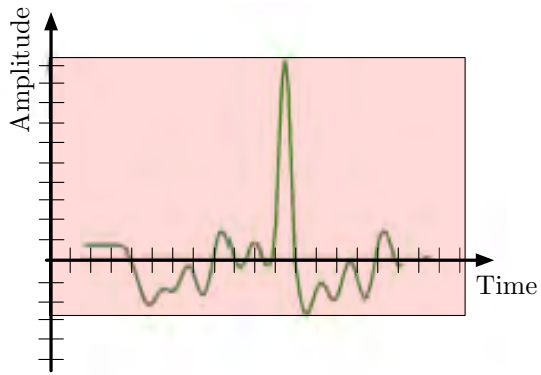




amplification/attenuation

Output $\boldsymbol{y}(k)$

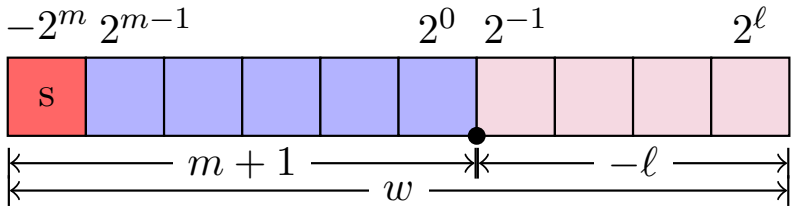
$\forall k, \quad |\boldsymbol{y}(k)| \leq \langle\langle \mathcal{H} \rangle\rangle \, \bar{\boldsymbol{u}}$



Worst-Case Peak Gain

$$\langle\langle \mathcal{H} \rangle\rangle = |\mathbf{D}| + \sum_{k=0}^{\infty} |\mathbf{C}\mathbf{A}^k\mathbf{B}|$$

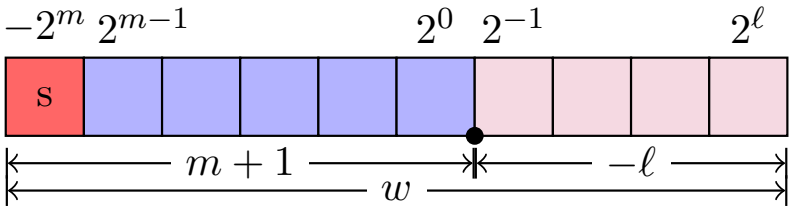
Two's complement Fixed-Point arithmetic



$$t = -2^m t_m + \sum_{i=\ell}^{m-1} 2^i t_i$$

- Wordlength: w
- **M**ost **S**ignificant **B**it position: m
- **L**east **S**ignificant **B**it position: $\ell := m - w + 1$

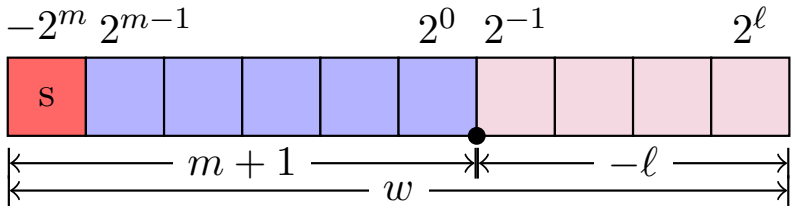
Two's complement Fixed-Point arithmetic



$$t = -2^m t_m + \sum_{i=\ell}^{m-1} 2^i t_i$$

- quantization step: 2^ℓ
- t represented by integer $T = t \cdot 2^{-\ell}$
- $T \in [-2^m; 2^m - 2^\ell] \cap \mathbb{Z}$

Two's complement Fixed-Point arithmetic



$$t = -2^m t_m + \sum_{i=\ell}^{m-1} 2^i t_i$$

- $y(k) \in \mathbb{R}$
- wordlength w bits
- minimal Fixed-Point Format (FPF) is the least m:

$$\forall k, \quad y(k) \in [-2^m; 2^m - 2^{m-w+1}]$$

Taking the quantization errors into account

The exact filter $\mathcal{H}$ is:

$$\mathcal{H}^{\Diamond} \left\{ \begin{array}{rcl} \boldsymbol{x}\ (k+1) & = & \boldsymbol{A}\boldsymbol{x}\ (k) + \boldsymbol{B}\boldsymbol{u}(k) \\ \boldsymbol{y}\ (k) & = & \boldsymbol{C}\boldsymbol{x}\ (k) + \boldsymbol{D}\boldsymbol{u}(k) \end{array} \right.$$

Taking the quantization errors into account

The actually implemented filter $\mathcal{H}^{\Diamond}$ is:

$$\mathcal{H}^{\Diamond}\left\{\begin{array}{rcl} \boldsymbol{x}^{\Diamond}(k+1) & = & \boldsymbol{A}\boldsymbol{x}^{\Diamond}(k)+\boldsymbol{B}\boldsymbol{u}(k)+\boldsymbol{\varepsilon}_x(k) \\ \boldsymbol{y}^{\Diamond}(k) & = & \boldsymbol{C}\boldsymbol{x}^{\Diamond}(k)+\boldsymbol{D}\boldsymbol{u}(k)+\boldsymbol{\varepsilon}_y(k) \end{array}\right.$$

with

$$|\boldsymbol{\varepsilon}_x(k)| \leqslant 2^{\boldsymbol{m}_x-\boldsymbol{w}_x+1} \quad \text{and} \quad |\boldsymbol{\varepsilon}_y(k)| \leqslant 2^{\boldsymbol{m}_y-\boldsymbol{w}_y+1}.$$

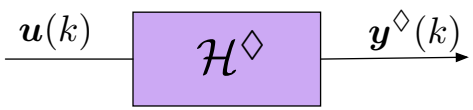
Taking the quantization errors into account

The actually implemented filter $\mathcal{H}^{\Diamond}$ is:

$$\mathcal{H}^{\Diamond} \left\{ \begin{array}{rcl} \boldsymbol{x}^{\Diamond}(k+1) & = & \boldsymbol{A}\boldsymbol{x}^{\Diamond}(k) + \boldsymbol{B}\boldsymbol{u}(k) + \boldsymbol{\varepsilon}_x(k) \\ \boldsymbol{y}^{\Diamond}(k) & = & \boldsymbol{C}\boldsymbol{x}^{\Diamond}(k) + \boldsymbol{D}\boldsymbol{u}(k) + \boldsymbol{\varepsilon}_y(k) \end{array} \right.$$

with

$$|\boldsymbol{\varepsilon}_x(k)| \leqslant 2^{\boldsymbol{m}_x - \boldsymbol{w}_x + 1} \quad \text{and} \quad |\boldsymbol{\varepsilon}_y(k)| \leqslant 2^{\boldsymbol{m}_y - \boldsymbol{w}_y + 1}.$$



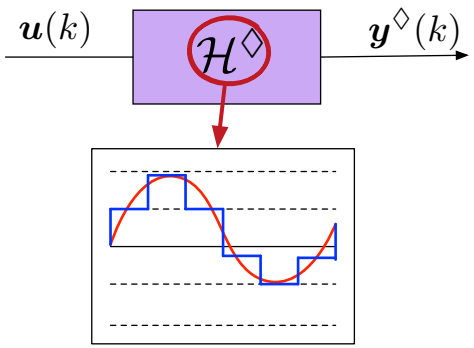
Taking the quantization errors into account

The actually implemented filter $\mathcal{H}^\Diamond$ is:

$$\mathcal{H}^\Diamond \left\{ \begin{array}{rcl} \boldsymbol{x}^\Diamond(k+1) & = & \boldsymbol{A}\boldsymbol{x}^\Diamond(k) + \boldsymbol{B}\boldsymbol{u}(k) + \boldsymbol{\varepsilon}_x(k) \\ \boldsymbol{y}^\Diamond(k) & = & \boldsymbol{C}\boldsymbol{x}^\Diamond(k) + \boldsymbol{D}\boldsymbol{u}(k) + \boldsymbol{\varepsilon}_y(k) \end{array} \right.$$

with

$$|\boldsymbol{\varepsilon}_x(k)| \leqslant 2^{\boldsymbol{m}_x - \boldsymbol{w}_x + 1} \quad \text{and} \quad |\boldsymbol{\varepsilon}_y(k)| \leqslant 2^{\boldsymbol{m}_y - \boldsymbol{w}_y + 1}.$$



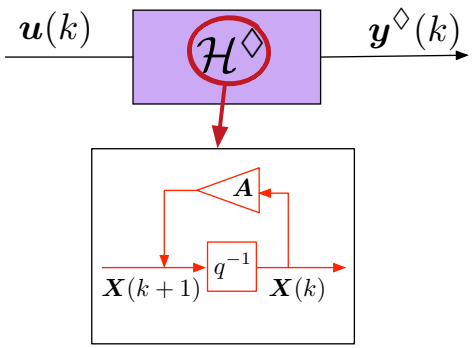
Taking the quantization errors into account

The actually implemented filter $\mathcal{H}^{\Diamond}$ is:

$$\mathcal{H}^{\Diamond}\left\{\begin{array}{rcl} \boldsymbol{x}^{\Diamond}(k+1) & = & \boldsymbol{A}\boldsymbol{x}^{\Diamond}(k)+\boldsymbol{B}\boldsymbol{u}(k)+\boldsymbol{\varepsilon}_x(k) \\ \boldsymbol{y}^{\Diamond}(k) & = & \boldsymbol{C}\boldsymbol{x}^{\Diamond}(k)+\boldsymbol{D}\boldsymbol{u}(k)+\boldsymbol{\varepsilon}_y(k) \end{array}\right.$$

with

$$|\boldsymbol{\varepsilon}_x(k)| \leqslant 2^{\boldsymbol{m}_x-\boldsymbol{w}_x+1} \quad \text{and} \quad |\boldsymbol{\varepsilon}_y(k)| \leqslant 2^{\boldsymbol{m}_y-\boldsymbol{w}_y+1}.$$



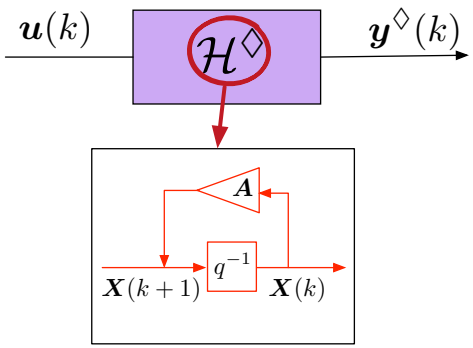
Taking the quantization errors into account

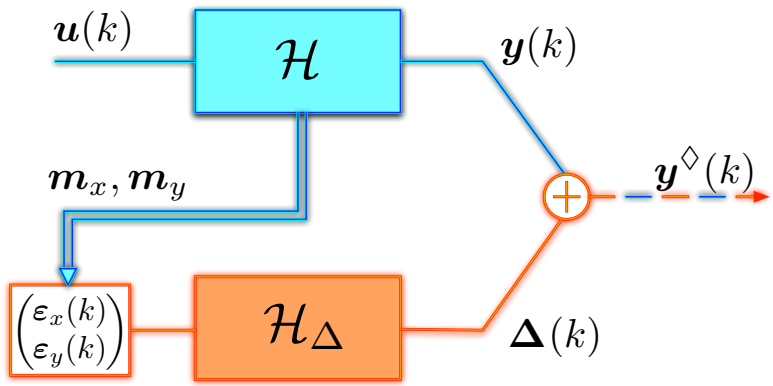
The actually implemented filter $\mathcal{H}^\Diamond$ is:

$$\mathcal{H}^\Diamond \left\{ \begin{array}{rcl} \boldsymbol{x}^\Diamond(k+1) & = & \boldsymbol{A}\boldsymbol{x}^\Diamond(k) + \boldsymbol{B}\boldsymbol{u}(k) + \boldsymbol{\varepsilon}_x(k) \\ \boldsymbol{y}^\Diamond(k) & = & \boldsymbol{C}\boldsymbol{x}^\Diamond(k) + \boldsymbol{D}\boldsymbol{u}(k) + \boldsymbol{\varepsilon}_y(k) \end{array} \right.$$

with

$$|\boldsymbol{\varepsilon}_x(k)| \leqslant 2^{\boldsymbol{m}_x - \boldsymbol{w}_x + 1} \quad \text{and} \quad |\boldsymbol{\varepsilon}_y(k)| \leqslant 2^{\boldsymbol{m}_y - \boldsymbol{w}_y + 1}.$$





Computing the MSB using the WCPG theorem

Applying the WCPG theorem on filter $\mathcal{H}$ gives

$$\forall k, \quad |\boldsymbol{y}_i(k)| \leqslant \left(\langle\!\langle \mathcal{H} \rangle\!\rangle \, \bar{\boldsymbol{u}}\right)_i .$$

We can compute the MSB positions with

$$\boldsymbol{m}_{y_i} = \left\lceil \log_2 \left(\langle\!\langle \mathcal{H} \rangle\!\rangle \bar{\boldsymbol{u}} \right)_i - \log_2 \left(1 - 2^{1-\boldsymbol{w}_{y_i}}\right) \right\rceil .$$

Computing the MSB using the WCPG theorem

Applying the WCPG theorem on filter $\mathcal{H}$ gives

$$\forall k, \quad |\boldsymbol{y}_i(k)| \leqslant \left(\langle\!\langle \mathcal{H} \rangle\!\rangle \, \bar{\boldsymbol{u}}\right)_i .$$

We can compute the MSB positions with

$$\boldsymbol{m}_{y_i} = \left\lceil \log_2 \left(\langle\!\langle \mathcal{H} \rangle\!\rangle \bar{\boldsymbol{u}} \right)_i - \log_2 \left(1 - 2^{1-\boldsymbol{w}_{y_i}}\right) \right\rceil .$$

Computing the MSB using the WCPG theorem

Applying the WCPG theorem on filter $\mathcal{H}$ gives

$$\forall k, \quad |\boldsymbol{y}_i(k)| \leqslant \left(\langle\!\langle \mathcal{H} \rangle\!\rangle \, \bar{\boldsymbol{u}}\right)_i .$$

We can compute the MSB positions with

$$\begin{aligned}\widehat{\boldsymbol{m}}_{y_i} &= \left\lceil \log_2 \left(\langle\!\langle \mathcal{H} \rangle\!\rangle \, \bar{\boldsymbol{u}}\right)_i - \log_2 \left(1 - 2^{1-\boldsymbol{w}_{y_i}}\right) + \log_2 \left(1 + \frac{\left(\varepsilon_{WCPG} \boldsymbol{u}\right)_i}{\left(\langle\!\langle \mathcal{H} \rangle\!\rangle \, \bar{\boldsymbol{u}}\right)_i}\right) \right\rceil \\ &= \boldsymbol{m}_{y_i} + \lceil \ldots \rceil\end{aligned}$$

Computing the MSB using the WCPG theorem

Applying the WCPG theorem on filter $\mathcal{H}$ gives

$$\forall k, \quad |\boldsymbol{y}_i(k)| \leqslant \left(\langle\!\langle \mathcal{H} \rangle\!\rangle \, \bar{\boldsymbol{u}}\right)_i .$$

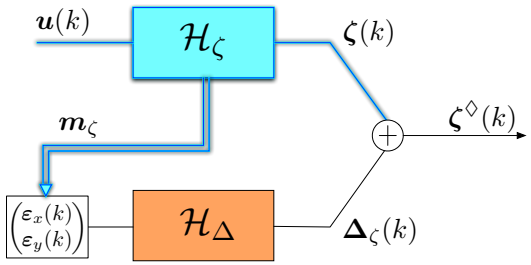
We can compute the MSB positions with

$$\begin{aligned} \widehat{\boldsymbol{m}}_{y_i} &= \left\lceil \log_2 \left(\langle\!\langle \mathcal{H} \rangle\!\rangle \, \bar{\boldsymbol{u}}\right)_i - \log_2 \left(1 - 2^{1-\boldsymbol{w}_{y_i}}\right) + \log_2 \left(1 + \frac{\left(\varepsilon_{WCPG} \boldsymbol{u}\right)_i}{\left(\langle\!\langle \mathcal{H} \rangle\!\rangle \, \bar{\boldsymbol{u}}\right)_i}\right) \right\rceil \\ &= \boldsymbol{m}_{y_i} + \lceil \ldots \rceil \end{aligned}$$

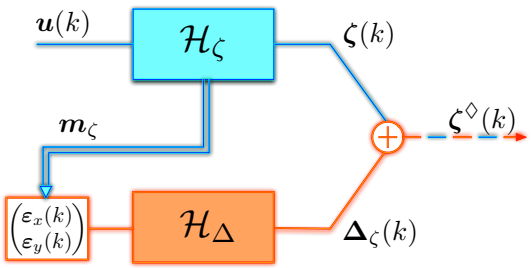
Adjust error term ε_{WCPG} in order to be at most **off by one**.

Algorithm

Step 1 Determine the MSBs $\boldsymbol{m}_y$ for the exact filter $\mathcal{H}$, applying the WCPG theorem;



Step 2 Compute the error-filter $\mathcal{H}_\Delta$, induced by the format $\boldsymbol{m}_y$ and deduce the MSB $\boldsymbol{m}_y^\Diamond$ of the implemented filter;



Step 3 If $\boldsymbol{m}_{y_i}^\Diamond == \boldsymbol{m}_{y_i}$ then return $\boldsymbol{m}_{y_i}^\Diamond$ otherwise $\boldsymbol{m}_{y_i} \leftarrow \boldsymbol{m}_{y_i} + 1$ and go to Step 2.

Numerical examples

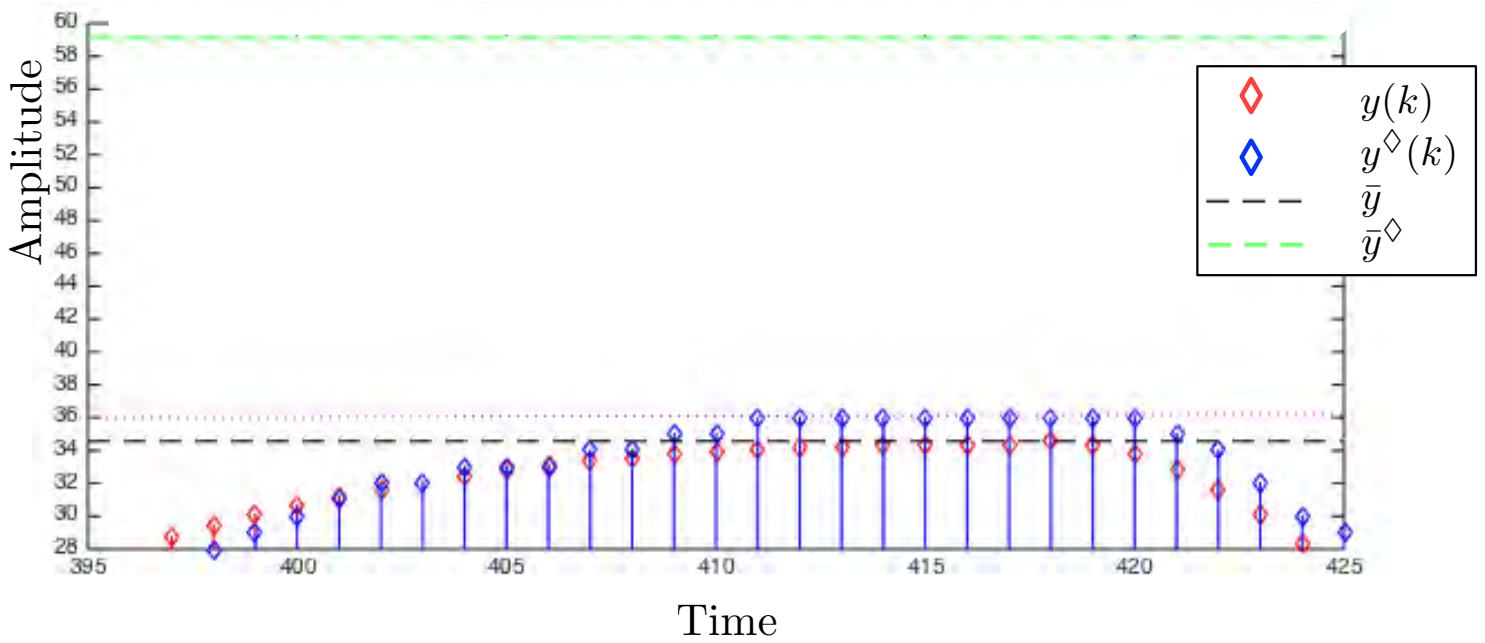
Example:

- Random filter with 3 states, 1 input, 1 output
- $\bar{u} = 5.125$, wordlengths set to 7 bits

	states			output
	$\boldsymbol{x}_1(k)$	$\boldsymbol{x}_2(k)$	$\boldsymbol{x}_3(k)$	$\boldsymbol{y}(k)$
Step 1	6	7	5	6
Step 2	6	7	6	6
Step 3	6	7	6	6

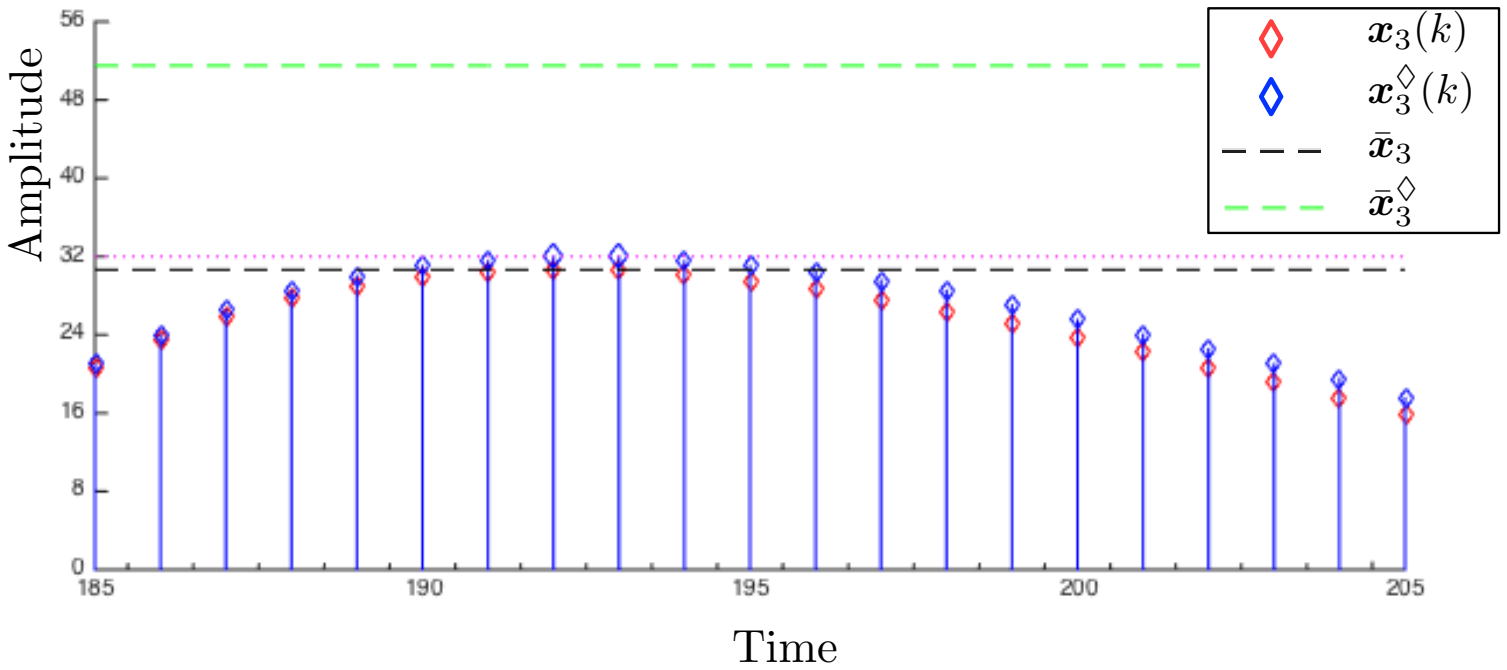
Table Evolution of the MSB positions

Numerical examples



The exact and quantized outputs of the example. Quantized output does not pass over to the next binade.

Numerical examples



The exact and quantized third state of the example. Quantized output passes over to the next binade.

Conclusion and Perspectives

Conclusion

- Computing WCPG in arbitrary precision
- Rigorous procedure for Fixed-Point Formats determination
- Filter computation errors are taken into account, ensuring that no overflow occurs
- Do not use any (non-exhaustive) simulations but proofs.

Perspectives

- Integrate into optimization procedures in automatic workflow
- Solve off-by-one problem for the MSB computation (e.g. using linear integer programming, lattices)
- Compute WCPG for interval matrices (requires interval solver for Lyapunov equations)