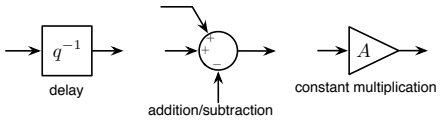
Fixed-Point implementation of Simulink block diagrams

Advisor: Thibault HILAIRE (`thibault.hilaire@lip6.fr`)

Context:

Simulink (or Simulink-like softwares) allows to design and simulate systems (dynamic systems, signal processing or control systems, etc.) described with by block diagrams. Block diagrams are visual representations of dynamic systems, and they consist of interconnected blocks performing some operations on the input(s) in order to produce output(s).

We focus here on linear algorithms, *ie* algorithms that only used additions/subtractions, multiplications by constants and delay operators. They can be described in block diagrams with the following blocks:



On the other hands, it has been shown that such algorithms can be exactly described with a dedicated state-space equation (denoted SIF, for Specialized Implicit Framework), that allow to represent the operations and the way they are chained:

$$\begin{pmatrix} \boldsymbol{J} & \boldsymbol{0} & \boldsymbol{0} \\ -\boldsymbol{K} & \boldsymbol{I}_n & \boldsymbol{0} \\ -\boldsymbol{L} & \boldsymbol{0} & \boldsymbol{I}_p \end{pmatrix} \begin{pmatrix} \boldsymbol{t}(k+1) \\ \boldsymbol{x}(k+1) \\ \boldsymbol{y}(k) \end{pmatrix} = \begin{pmatrix} \boldsymbol{0} & \boldsymbol{M} & \boldsymbol{N} \\ \boldsymbol{0} & \boldsymbol{P} & \boldsymbol{Q} \\ \boldsymbol{0} & \boldsymbol{R} & \boldsymbol{S} \end{pmatrix} \begin{pmatrix} \boldsymbol{t}(k) \\ \boldsymbol{x}(k) \\ \boldsymbol{u}(k) \end{pmatrix}$$

where $\boldsymbol{u}(k)$ is the inputs vector at time k, $\boldsymbol{y}(k)$ the outputs vector, $\boldsymbol{x}(k)$ collects the states (informations stored from one step to another) and $\boldsymbol{t}$ the intermediates values. The $\boldsymbol{J}$ matrice is lower triangular, with 1's on the diagonal. The structure (sparsity) of the matrices $\boldsymbol{K}, \boldsymbol{L}, \boldsymbol{M}, \boldsymbol{N}, \boldsymbol{P}, \boldsymbol{Q}, \boldsymbol{R}, \boldsymbol{S}$ reflects the structure of the operations, and their coefficients the constants used.

With these framework have been developed tools and methods for the generation of fixed-point algorithms, specially *FiPoGen* and *Stratus*, both developed in the lab:

- *FiPoGen* helps to automatically produce reliable fixed-point algorithms for embedded targets and solve the tradeoff between implementation cost and numerical properties (specially for finite precision arithmetic)
- *Stratus* is a generic hardware component generator, where the operators can be parametrized by their bit width, numeration system, etc.

Subject:

The aim of this internship is to transform Simulink block diagrams into VHDL code. First, simulink block diagrams will be transformed in equivalent[1] SIF realizations, and then transformed in hardware implementation with *FiPoGen* and /emphStratus tools. The outline will be the following:

[1] equivalent in the sense that it has the same input-output relationship, but also preserves the structure of the computations.

1. Study `.mdl` structure file (or `.sxl`), and generate a block diagram in Simulink (a `.mdl` or .sxl) from a given SIF realization;
2. Develop an algorithm that will iterate on a block diagram (on the equivalent graph) in order to aggregate the blocks and build the equivalent SIF;
3. Use *FiPoGen* to generate fixed-point algorithm from the various sum-of-products of the SIF;
4. Use *Stratus* to generate hardware implementation from fixed-point algorithm.

We will first start with basic block diagrams (where the signals are scalar), and extend it, if possible, to more complex diagrams (with vectors and some non-linear blocks).
This internship can be followed by a 6 months engineer position (CDD) at Laboratoire d'Informatique de Paris 6 (LIP6), if the student is enough motivated and the internship successful.

Prerequisites: programming skills (Python is preferable, but it can be C or C++), graph theory, linear algebra.It is not mandatory to know Simulink.