Formal Analysis of Engineering Systems Based on Signal-Flow-Graph Theory

Sidi Mohamed Beillahi(✉), Umair Siddique, and Sofiène Tahar

Department of Electrical and Computer Engineering,
Concordia University, Montreal, QC, Canada
{beillahi,muh_sidd,tahar}@ece.concordia.ca

Abstract. Signal-flow-graph theory provides an efficient framework to model various engineering and physical systems at a higher-level of abstraction. In this paper, we present the formalization of the signal-flow-graph theory with an ultimate goal to conduct the formal analysis of engineering systems within a higher-order-logic theorem prover. In particular, our formalization can tackle system models which are based on undirected graphs. We also present the formalization of the system transfer function and associated properties such as stability and resonance. In order to demonstrate the effectiveness of our work, we present the formal analysis of two engineering systems namely the PANDA Vernier resonator and the z-source impedance network, which are commonly used in photonics and power electronics, respectively.

1 Introduction

A signal-flow-graph (SFG) [17] is a diagram that represents a set of simultaneous linear algebraic equations describing the flow of different physical quantities (e.g., electric current) from one point of the system to another point. Many physical and engineering systems ranging from analog circuits to photonic signal processors can be represented using the signal-flow-graph theory. Generally, there are two types of SFGs, i.e., directed and undirected [23,24]. In a directed SFG, nodes should be ordered, whereas an undirected SFG does not require a strict ordering of nodes which makes it more convenient to model a variety of systems such as photonic filters and analog power converters.

Once an SFG representation of the underlying system has been built, the next step is to analyze the corresponding behavior to ensure that the system meets its specification. The Mason's Gain Formula (MGF) [15,16] provides an efficient way to obtain a transfer function (input-output relation) directly from the SFG representation. Moreover, the obtained transfer function is used to investigate different properties of the system such as stability (which ensures that the output is bounded whenever the input in bounded). Some of the main applications of MGF include the analysis of wireless networks [14], security protocols [7], process engineering [13], power electronics [12,20] and photonic signal processing [5].

In the past few decades, formal methods [6] have been successfully applied to improve the analysis of a variety of software, hardware and physical systems.

S. Bogomolov et al. (Eds.): NSV 2016, LNCS 10152, pp. 31–46, 2017.
DOI: 10.1007/978-3-319-54292-8_3

The main idea behind formal analysis of a system is to construct a computer based mathematical model of the given system and formally check that the model meets its specifications. The rigorous process of building a mathematical model for the given system and analyzing this model using mathematical reasoning usually increases the chances for catching subtle but critical design errors that are often ignored by traditional techniques such as paper-and-pencil based proofs, numerical methods or simulation. Given the extensive usage of signal-flow-graph to model engineering systems that are employed in safety-critical applications, we believe that there is a dire need of building a formal framework to reason about the signal-flow-graph theory.

The involvement of complex-valued parameters requires an expressive technique to formalize the notions of the signal-flow-graph theory and MGF. Higher-order-logic (HOL) theorem proving [11] is a generic formal methods technique which provides such an expressive formalism to reason about multivariate analysis. In [4], we used the HOL Light theorem prover to formalize the notion of signal-flow-graph theory and the procedure to obtain the transfer function for directed SFGs based on MGF. We applied this formalization to verify the stability of power electronic converters. However, this formalization can only be used for directed SFGs which limits its usage for many practical systems. In this paper, we build upon our previous work and formalize undirected SFG theory and MGF along with the notion of system stability. We apply this formalization to two distinct domains, i.e., power electronics and photonics. Indeed, we present in this paper the formal verification of z-source impedance network [10] and PANDA Vernier resonator [1] which are commonly used systems in power electronics and photonics, respectively. The source code of our formalization is available for download [2] and can be utilized by other researchers and engineers for further developments and the analysis of more practical systems.

The rest of the paper is organized as follows: we give a preliminary review of the SFG theory and the Mason's Gain Formula in Sect. 2. We present the HOL formalization of undirected SFGs in Sect. 3. Consequently, in Sect. 4, we provide the formalization of transfer functions along with the notion of stability and resonance. We describe the formal analysis of the z-source impedance network and PANDA Vernier resonator as illustrative practical applications in Sects. 5 and 6, respectively. Finally, Sect. 7 concludes the paper and provides hints for some future directions.

2 Signal-Flow Graphs Theory and Mason's Gain Formula

Mathematically, a signal-flow-graph represents a set of linear algebraic equations of the corresponding system [16]. An SFG is a network in which nodes are connected by directed branches. Every node in the network represents a system variable and each branch represents the signal transmission from one node to the other under the assumption that signals flow only in one direction. An example of an SFG is shown in Fig. 1 consisting of six nodes. An input or *source node* and an output or *sink node* are usually the ones which only have outgoing branches

and incoming branches, respectively (e.g., nodes 1 and 6 in Fig. 1). A branch is a directed line from node i to j and the gain of each branch is called the *transmittance*. A *path* is a traversal of connected branches from one node to the other and if no node is crossed more than once and connects the input to the output, then the path is called *forward path*, otherwise if it leads back to itself without crossing any node more than once, it is considered as a *closed path* or a *loop*. A loop containing only one node is called a *self loop* and any two loops in the SFG are said to be *touching loops* if they have any common node. The total gain of a forward path and a loop can be computed by multiplying the transmittances of each traversed branch. In the analysis of practical engineering systems (resp. process), the main task is to characterize the relation among the system's (resp. process) input and output which is called the transfer function. The total transmittance or gain between two given nodes (usually input and output) describes the transfer function of the corresponding system. In 1953, Mason [16] proposed a computational procedure (also called Mason's Gain Formula) to obtain the total gain of any arbitrary signal-flow-graph. The formula is described in Eq. 1 as follows:

$$G = \sum_{k} \frac{G_k \Delta_k}{\Delta} \tag{1}$$

$$\Delta = 1 - \sum_{m} P_{m1} + \sum_{m} P_{m2} - \sum_{m} P_{m3} + \ldots + (-1)^n \sum \ldots \tag{2}$$

where Δ represents the determinant of the graph, Δ_k represents the value of Δ for the part of the graph that is not touching the k^{th} forward path and it is called the cofactor of forward path k, P_{mr} is the gain product of m^{th} possible combination of r non-touching loops. The gain of each forward path is represented by G_k.

For example, in Fig. 1 taking the nodes 1 and 6 as the input and output nodes, respectively. There is one forward path ($1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6$) and

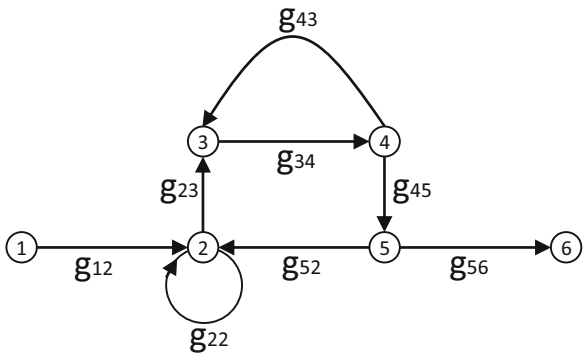

Fig. 1. A signal-flow graph

three loops ($2 \to 2$, $2 \to 3 \to 4 \to 5 \to 2$ and $3 \to 4 \to 3$). Thus, we can find the input to output transfer function using the MGF as follows:

$$G = \frac{g_{12}.g_{23}.g_{34}.g_{45}.g_{56}}{1 - g_{22} - g_{23}.g_{34}.g_{45}.g_{52} - g_{34}.g_{43} + g_{22}.g_{34}.g_{43}} \tag{3}$$

3 Formalization of Undirected Signal-Flow-Graph Theory

We model a single branch as a triplet $(\mathtt{a}, \mathtt{t_{ab}}, \mathtt{b})$, where $\mathtt{a}$, $\mathtt{t_{ab}}$ and $\mathtt{b}$ represent the start node, the transmittance and the end node, respectively. Consequently, a path can be modeled as a list of branches and furthermore an SFG can be defined as a composition of paths along with the information about the total number of nodes in the circuit, the source and the sink nodes at which we want to compute the transfer function. As mentioned before, nodes and transmittance represent the system variables and gain, respectively. These parameters are indeed complex valued, i.e., $\mathtt{a}, \mathtt{t_{ab}}, \mathtt{c} \in \mathbb{C}$ in the general context of engineering and physical systems. However, the information about the nodes is just used to find properties of signals (current) transmission and they do not appear in the gain and transfer function computation. So, we adapted the conventional approach as in [16], where nodes of an SFG are represented by natural numbers ($\mathbb{N}$). In order to simplify the reasoning process, we encode the above information by defining three type abbreviations in HOL Light[1], i.e., branch, path and signal-flow-graph as follows:

Definition 1 (Branch, Path and SFG).

```
new_type_abbrev ("branch", ‘:ℕ × ℂ × ℕ‘)
new_type_abbrev ("path", ‘:(branch)list‘)
new_type_abbrev ("sfg", ‘:path × ℕ × ℕ × ℕ‘)
```

where the second, third, and fourth elements of `sfg` represent the size, the output node and the input node of a signal-flow graphs, respectively. It is important to notice that in the definition of the `branch`, we did not add any constraint regarding the order of the two nodes of the branch which makes this formalization valid for both cases of directed and undirected graphs.

Our next main task is to find all forward paths and loops from the source node to the sink node for the given system. In [4], we implemented a search algorithm proposed in [24], which considers only directed graphs. In order to extend this work to cover both types (i.e., directed and undirected) of SFGs, we enhanced this algorithm by using some new features of the graph.

In [24], the author used two matrices (`G` and `H`) and a vector (`P`) to perform the search in an SFG. The matrix `G` contains the information about the graph, in particular, each row `i` of `G` contains the nodes where there are branches from `i` that end at these nodes. The vector `P` contains the path under consideration

[1] In this paper, we use minimal HOL Light syntax in the presentation of definitions and theorems to improve the readability.

in the search process, where the first element in the vector is the head of the path from which the search begins to construct the path. The matrix `H` is to ensure that we do not extract the same path twice. Each row `i` in the matrix `H` contains the nodes that cannot form a path with `i` (dead nodes). The procedure for extracting the loops can be subdivided into five steps as follows:

1. We first extract the graph information inside the matrix `G` and initialize the matrix `H` to `0`. Then we initialize the elements of the vector `P` to `0` except the first element (which we initialize to `1`) from which the search will start.
2. In the second step, we have a node i which is the tail of the vector P and we search for a node j in the graph that satisfies three conditions: (a) j is not in P (i.e., j is not already part of the path); (b) j is not in H(i) (i.e., j is not a dead node for i); (c) j >i (i.e., all paths are directed). The last condition constraints the application of this procedure on undirected SFGs. If `j` is found we add it in the vector `P` and repeat the process by replacing `i` by `j`. Finally, if no `j` is found, we stop the search and move to the next step.
3. In the third step, we want to confirm that the found path in the previous step is indeed a loop. If a loop is found, we report it and in all cases we go to the next step.
4. In this step, we confirm that an exhaustive search has been performed for all the loops that can start from the head of the vector `P`, otherwise, we initialize the row of `H` that contains all the dead nodes to the last node of vector `P`.
5. In this step, we check if the head of the vector `P` is the last node in the graph which indicates the termination of the search, otherwise, we replace the head of the vector `P` by the next node in the graph and we go back to the second step.

Note that the above procedure with slight modifications can be used for the forward path extraction. In order to extend this algorithm for undirected SFGs, we add two new matrices $\mathtt{G}_1$ and $\mathtt{H}_1$. The matrix $\mathtt{G}_1$ contains some extra information about the graph where each row `i` contains the nodes where there are branches that start from these nodes and end at `i`. In graph theory, any element of a loop can be the head of the loop. Therefore, for computing loops this means that if we once did exhaustive search for all the loops that start from a certain node `i` then in the search for loops that start from another node `j` we do not need to consider the node `i`. This means `i` is a dead node for `j`. Hence, the matrix $\mathtt{H}_1$ keeps track of these new dead nodes. The corresponding changes in the Steps 2 and 5 are summarized as follows:

– We change the third condition for extending the search and replace it by checking if `j` is not in $\mathtt{H}_1(\mathtt{i})$ (means `j` is not a dead node for `i`).
– We add a new procedure by inserting the head of `P` in all the rows `i` of $\mathtt{H}_1$ if there is a branch that starts from node `i` and ends at the head of `P`.

We next provide the definitions of two crucial functions in our formalization of undirected SFG that perform the Steps 2 and 5. The complete HOL formalization of the SFG theory can be found at [2].

Definition 2 (Next Node Search).

```
⊢ ∀(n m ∈ ℕ) (t1 t2 l1 l2 h1 ∈ ℕ list).
EC_next_node n m [] t2 l1 l2 h1 = 0 ∧
EC_next_node n (m + 1) (e1::t1) (e2::t2) l1 l2 h1  =
EC_next_node n m (e1::t1) t2 l1 l2 h1 ∧
EC_next_node n 0 (e1::t1) (e2::t2) l1 l2 h1  =
if ((NOT_IN_LIST e1 l1) ∧ (NOT_IN_LIST e1 l2) ∧ (NOT_IN_LIST e1 h1))
then n + 1 else EC_next_node (n + 1) 0 t1 (e2::t2) l1 l2 h1
```

where the vector `P`, the row `H(i)`, and the row $\mathtt{H}_1(\mathtt{i})$ are represented as the lists `l1`, `l2`, `h1`, respectively. The function `NOT_IN_LIST` accepts a list and an element and tests the membership of the element. The main function `EC_next_node` returns `(n + 1)`, which represents the position of the node `e1` inside the list `G[k]` (list `t1` in the definition) that satisfies the three conditions in Step 2, where `k` is the termination node for the considered path. In the process of searching for a path all the graph nodes are considered.

Next, we define the procedure to update the matrix $\mathtt{H}_1$ after completing the search for loops that start from a given node `e1`. Here, `e1` will be a dead node for all nodes for which branches exist that end at `e1`:

Definition 3 (Updating the Matrix $\mathtt{H}_1$).

```
⊢ ∀ (e1 : num) (H1 : (num list) list) (l : num list).
EC_H1_MODIFIED e1 H1 [] = H1 ∧
EC_H1_MODIFIED e1 H1 (h::l) =
EC_H1_MODIFIED e1 (EC_H1_REPLACE (EC_H_MODIFIED0 H1[h − 1] e1) (h − 1) H1) l
```

where `EC_H_MODIFIED0` accepts an element `e1` and a list `l` (i.e., $\mathtt{H}_1[\mathtt{h}-1]$) and searches for the leftmost zero in the list `l` and replaces it by `e1`. The function `EC_H1_REPLACE` takes a list `l` (i.e., `EC_H_MODIFIED0` $\mathtt{H}_1[\mathtt{h}-1]$ `e1`), an integer `(h − 1)` and the matrix then it replaces the row $\mathtt{H}_1[\mathtt{h}-1]$ by l inside the matrix `H`.

A transpose of an SFG can be obtained by inverting all the branches and swapping input and output. We formally define the inversion of branches as follows:

Definition 4 (Inverse of Graph Branches).

```
⊢ ∀ (l : path). SFG_INVERSE [] = []  ∧  SFG_INVERSE e::l  =
(lst_of_trpl e, snd_of_trpl e, fst_of_trpl e)::SFG_INVERSE l
```

Here `fst_of_trpl`, `snd_of_trpl`, and `lst_of_trpl` return the first, second, and last element of a given triple, respectively. Using this definition, we formalize the graph transposition as follows:

Definition 5 (Graph Transpose).

```
⊢ ∀ sfg. SFG_TRANSPOSE sfg  =
  (SFG_INVERSE (fst_of_four sfg), snd_of_four sfg,
  lst_of_four sfg, thd_of_four sfg)
```

In the next section, we provide the formalization of transfer function, Mason's gain formula and some associated properties.

4 Formalization of Transfer Function

As described in Sect. 2, Meson's Gain Formula requires the list of all loops to find the graph determinant and the list of forward paths to compute the graph numerator. We provide in below the main definition in the transfer function formalization, however, the complete formalization can be found at [2].

Definition 6 (Mason's gain formula).

$$\vdash \forall\ (\texttt{system} : \texttt{sfg}).\ \texttt{Mason\_Gain system} = \frac{\texttt{PRODUCT\_FORWARD\_DELTA}\ (\texttt{EC system})\ (\texttt{FC system})}{\texttt{DETERMINANT}\ (\texttt{EC system})}$$

where `Mason_Gain` accepts an SFG (i.e., `system`) and returns the Mason's gain as given in Eq. 1. Notice that the function `PRODUCT_FORWARD_DELTA` accepts the lists of loops and forward paths (computed in Sect. 3) in the graph and computes $\sum_{\texttt{k} \in \texttt{system}} \texttt{G}_\texttt{k}\Delta_\texttt{k}$, where $\texttt{G}_\texttt{k}$ and $\Delta_\texttt{k}$ represent, respectively, the product of all forward path gains and the determinant of the $\texttt{k}^{\texttt{th}}$ forward path considering the elimination of all loops touching the $\texttt{k}^{\texttt{th}}$ forward path as described in Sect. 2. The function `DETERMINANT` takes the list of loops and produces the determinant of the graph as described in Eq. (2). Finally, we utilize the above formalization along with loops and forward paths extraction procedures to formalize the transfer function of a given engineering system as follows:

Definition 7 (System Transfer Function).

$$\vdash \forall\ \texttt{system}.\ \texttt{transfer\_function system} = \texttt{Mason\_Gain}\ (\lambda \texttt{s}.\ \texttt{system s})$$

where the function `transfer_function` accepts a `system` which has type $\mathbb{C} \rightarrow$ `sfg` and returns a complex ($\mathbb{C}$) number which represents the transfer function of the engineering system (i.e., `system`). Here "`s`" is a complex parameter of the given system.

The availability of the transfer function provides the facility to analyze the stability and resonance conditions for the given system. Mathematically, the stability and resonance are concerned with the identification of all the values of `s` for which the system transfer function becomes infinite and zero, respectively. In the control theory and signal processing literature, these values are called *system poles* and *system zeros* which can be computed by the denominator and numerator of the transfer function, respectively. Furthermore, all poles and zeros must be inside the unit circle which means that their complex norm should be less or equal to `1`. We formalize the above mentioned informal description of the system properties in HOL as follows:

Definition 8 (System Poles).

$$\vdash \forall\ \texttt{system}.\ \texttt{poles system} = \{\texttt{s} \mid \texttt{denominator}\ (\texttt{system s}) = 0\}$$
$$\vdash \forall\ \texttt{system}.\ \texttt{zeros system} = \{\texttt{s} \mid \texttt{numerator}\ (\texttt{system s}) = 0\}$$

where the functions `poles` and `zeros` take the `system` as a parameter and return the set of poles and zeros, respectively. Next, we formalize the notion of stability and resonance as follows:

Definition 9 (System Stability and Resonance).

$$\vdash \forall\ \texttt{system}.\ \texttt{is\_stable system}\ [\texttt{p}_0, ..., \texttt{p}_n] \Leftrightarrow$$
$$\forall \texttt{p}_i.\ \texttt{p}_i \in (\texttt{poles system}) \wedge \parallel \texttt{p}_i \parallel \leq 1$$
$$\vdash \forall\ \texttt{system}.\ \texttt{is\_resonant system}\ [\texttt{z}_0, ..., \texttt{z}_n] \Leftrightarrow$$
$$\forall \texttt{z}_i.\ \texttt{z}_i \in (\texttt{zeros system}) \wedge \parallel \texttt{z}_i \parallel \leq 1$$

where the predicate `is_stable` accepts a signal-flow-graph model (i.e., `system`) and a list of poles $[\texttt{p}_0, ..., \texttt{p}_n]$ and verifies that each element $\texttt{p}_i$ is indeed a pole of the system and its corresponding magnitude (i.e., norm of a complex number, $\parallel \texttt{p}_i \parallel$) is smaller or equal to 1. The predicate `is_resonant` is defined in a similar way by considering the list of zeros instead of the list of poles of the system. In the above formalization, $\texttt{p}_i$ and $\texttt{z}_i$ are continuous complex functions.

In the following sections, we present the formal analysis of two engineering systems namely z-source impedance network and PANDA Vernier resonator. It is important to notice that the SFG of the PANDA Vernier resonator is undirected (e.g., the branch between the nodes 10 and 5). Furthermore, the SFG of the z-source impedance network is directed, however, its transpose is undirected.

5 Z-Source Impedance Network

The z-source is an impedance-source power converter that is considered to be more efficient than other commonly used power converters [19]. In [18], the authors claimed that the model of z-source can be applied to almost all DC-to-AC, AC-to-DC, AC-to-AC, and AC-to-DC power conversions. Thus it has been used in hybrid electric vehicles with dual mode, i.e., as a boost converter and buck converter [8]. The topology of z-source is composed of two-port network that consists of a split-inductors L1 and L2 and capacitors C1 and C2 connected in X shape [18]. The circuit configuration of the z-source impedance network is shown in Fig. 2, and its associate signal-flow-graph is depicted in Fig. 3 [10]. We first formally define the z-source impedance network model as follows:

Definition 10 (Z-source Impedance Network Model).

$$\vdash \forall\ \texttt{G1 G2 DA DD G5 R L C s} \in \mathbb{C}.$$
$$\texttt{ZSOURCE\_model G1 G2 DA DD G5 R L C s} =$$
$$[(1, \texttt{G1}, 4); (1, \texttt{G2}, 7); (2, \texttt{DA}, 4); (3, \texttt{R.DA}, 4); (3, -\texttt{DA}, 7); (4, 1, 5); (5, \tfrac{1}{\texttt{L.s}}, 6);$$
$$(6, \texttt{G5}, 4); (6, \texttt{DD}, 7); (6, 1, 9); (7, 1, 8); (8, 1, 9); (8, \tfrac{1}{\texttt{C.s}}, 11); (9, 1, 10); (11, \texttt{DD}, 4);$$
$$(11, 2, 12)]$$

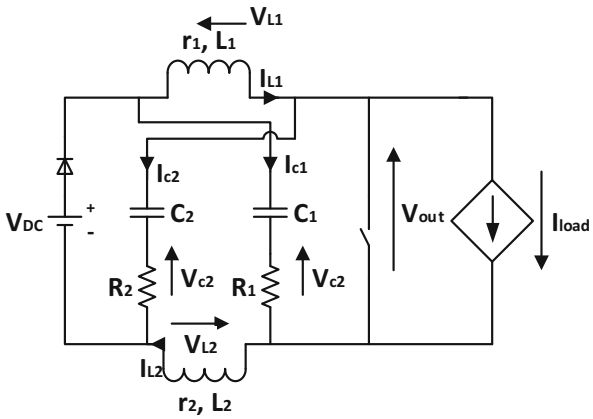

Fig. 2. Schematics of Z-source converter

where `ZSOURCE_model` represents the SFG of the z-source circuit depicted in Fig. 3 and it accepts the circuit parameters as the graph variables. We next formally verify the transfer function for the z-source impedance network between nodes `1` and `12` as follows:

Theorem 1 (Transfer Function of Z-source).

$$\vdash \ \forall\ \mathtt{G1\ G2\ DA\ DD\ G5\ R\ L\ C\ s}\ \in \mathbb{C}.\quad (\mathtt{C.L} \neq 0)\ \wedge\ (\mathtt{s} \neq 0)\ \Longrightarrow$$
$$\mathtt{transfer\_function\ (ZSOURCE\_model\ G1\ G2\ DA\ DD\ G5\ R\ L\ C\ s}, 12, 12, 1) =$$
$$\frac{2(\mathtt{G2.L.s} + \mathtt{G1.DD} - \mathtt{G2.G5})}{\mathtt{L.C.s}^2 - \mathtt{G5.C.s} - \mathtt{DD}^2}$$

where the two assumptions ensure that the gains $\frac{1}{\mathtt{Ls}}$ and $\frac{1}{\mathtt{Cs}}$ are well defined. The proof steps of the transfer function formula are mainly automated using a developed tactic that can be found at [2]. Similarly, we can derive other transfer functions which are listed in Table 1. Next, we present the formal verification of the stability conditions of the z-source impedance network under the global parameters:

Theorem 2 (Stability Conditions for Z-source).

$$\vdash \forall\ \mathtt{G1\ G2\ DA\ DD\ G5\ R\ L\ C}\ \in \mathbb{C}.$$
$$\| \tfrac{\mathtt{G5.C} - \sqrt{(\mathtt{G5.C})^2 + 4.(\mathtt{L.C}).\mathtt{DD}^2}}{2.\mathtt{L.C}} \| \leq 1\ \wedge\ \| \tfrac{\mathtt{G5.C} + \sqrt{(\mathtt{G5.C})^2 + 4.(\mathtt{L.C}).\mathtt{DD}^2}}{2.\mathtt{L.C}} \| \leq 1\ \wedge$$
$$\tfrac{\mathtt{G5.C} + \sqrt{(\mathtt{G5.C})^2 + 4.(\mathtt{L.C}).\mathtt{DD}^2}}{2.\mathtt{L.C}} \neq 0 \wedge \mathtt{C.L} \neq 0 \wedge \tfrac{\mathtt{G5.C} - \sqrt{(\mathtt{G5.C})^2 + 4.(\mathtt{L.C}).\mathtt{DD}^2}}{2.\mathtt{L.C}} \neq 0 \Longrightarrow$$
$$\mathtt{is\_stable}\ (\lambda\ \mathtt{s.}\ (\mathtt{ZSOURCE\_model\ G1\ G2\ DA\ DD\ G5\ R\ L\ C\ s}, 12, 12, 1))\ \mathtt{s}$$
$$[\tfrac{\mathtt{G5.C} - \sqrt{(\mathtt{G5.C})^2 + 4.(\mathtt{L.C}).(\mathtt{DD}^2)}}{2.\mathtt{L.C}};\ \tfrac{\mathtt{G5.C} + \sqrt{(\mathtt{G5.C})^2 + 4.(\mathtt{L.C}).(\mathtt{DD}^2)}}{2.\mathtt{L.C}}]$$

Table 1. Z-source transfer functions

Transfer Function	Formula
$\frac{\tilde{\mathtt{V}}_{\mathbf{sum}}}{\tilde{\mathtt{V}}_{\mathbf{dc}}}$	$(\mathtt{C.L} \neq 0) \wedge (\mathtt{s} \neq 0) \Rightarrow$ **SFG_Main** $((\mathbf{ZSOURCE\_model}\ \mathtt{G1\ G2\ DA\ DD\ G5\ R\ L\ C\ s}), 12, 12, 2) = \frac{2.\mathtt{DA.DD}}{(\mathtt{L.C.s}^2 - \mathtt{G5.C.s} - \mathtt{DD}^2)}$
$\frac{\tilde{\mathtt{V}}_{\mathbf{sum}}}{\tilde{\mathtt{I}}_{\mathbf{load}}}$	$(\mathtt{C.L} \neq 0) \wedge (\mathtt{s} \neq 0) \Rightarrow$ **SFG_Main** $((\mathbf{ZSOURCE\_model}\ \mathtt{G1\ G2\ DA\ DD\ G5\ R\ L\ C\ s}), 12, 12, 3) = \frac{-2.\mathtt{DA.(L.s - R.DD - G5)}}{(\mathtt{L.C.s}^2 - \mathtt{G5.C.s} - \mathtt{DD}^2)}$
$\frac{\mathbf{I_{in}}}{\tilde{\mathbf{V}}_{\mathbf{dc}}}$	$(\mathtt{C.L} \neq 0) \wedge (\mathtt{s} \neq 0) \Rightarrow$ **SFG_Main** $((\mathbf{ZSOURCE\_model}\ \mathtt{G1\ G2\ DA\ DD\ G5\ R\ L\ C\ s}), 12, 10, 2) = \frac{\mathtt{DA.(1 + DD).C.s}}{\mathtt{L.C.s}^2 - \mathtt{G5.C.s} - \mathtt{DD}^2}$

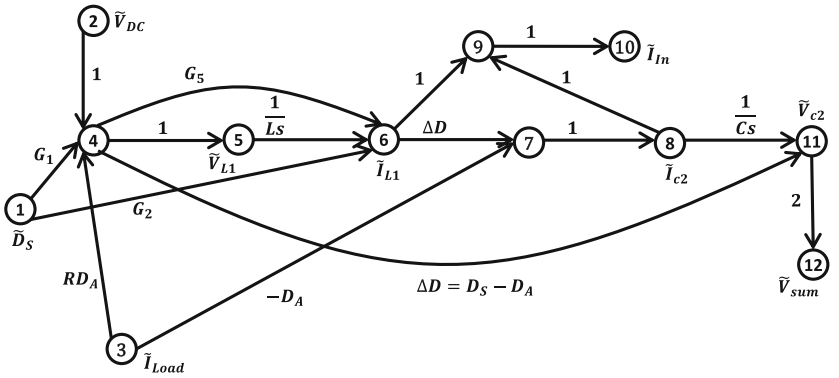

Fig. 3. Signal-flow-graph of Z-source converter

The first two assumptions ensure that both poles are inside the unit circle, whereas the last assumptions are required to prove that the poles are not equal to zero. Similarly, we verify the resonance condition for the z-source impedance network circuit as follows:

Theorem 3 (Resonance Conditions for Z-source).

$$\vdash \forall\ \mathtt{G1\ G2\ DA\ DD\ G5\ R\ L\ C}\ \in \mathbb{C}.$$
$$\| \tfrac{\mathtt{G2.G5-G1.DD}}{\mathtt{G2.L}} \| \leq 1 \wedge \mathtt{G2} \neq 0 \wedge (\mathtt{G2.G5 - G1.DD}) \neq 0 \wedge \mathtt{C.L} \neq 0 \implies$$
$$\mathtt{is\_resonant}\ (\lambda\ \mathtt{s}.\ (\mathbf{ZSOURCE\_model}\ \mathtt{G1\ G2\ DA\ DD\ G5\ R\ L\ C\ s}, 12, 12, 1))\ \mathtt{s}$$
$$[\tfrac{\mathtt{G2.G5-G1.DD}}{\mathtt{G2.L}}]$$

The assumptions in the theorem ensure that the system zero is inside the unit circle and it is not equal to zero. The proof steps for the two theorems are mainly based on first checking that the poles (resp. zeros) belong to the set of poles (resp.

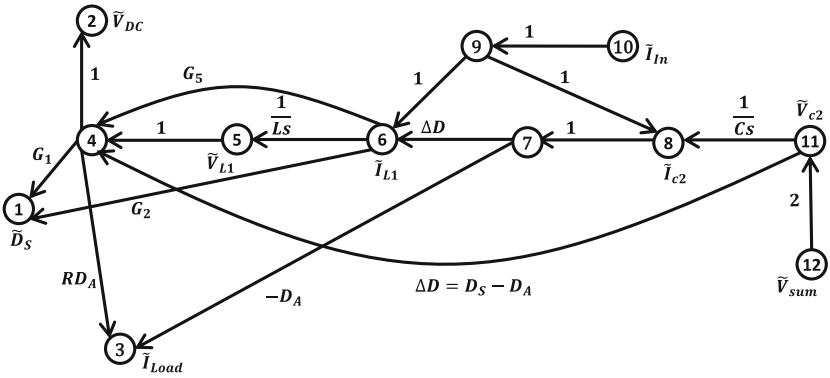

Fig. 4. Signal flow graph of Z-source converter transpose

zeros) defined in Definition 9. Then checking that the norm of these poles and zeros are less or equal to 1 using HOL rewriting rules. For electronic circuits, signal-flow-graph has a particular interesting feature, as shown in [20], namely, the transfer function of circuit transposition is the same as the original circuit. For example, Fig. 4 shows the transposed SFG of the z-source impedance network. We formally verified that the transfer functions of the z-source impedance network signal-flow-graph and its transpose (which is an *undirected* signal-flow-graph) are the same:

Theorem 4 (Transfer Function of Z-source Transpose).

$$\vdash \forall\ \texttt{G1 G2 DA DD G5 R L C s} \in \mathbb{C}.\quad (\texttt{C.L} \neq 0) \ \wedge\ (\texttt{s} \neq 0) \implies$$
$$\texttt{transfer\_function (SFG\_TRANSPOSE (ZSOURCE\_model G1 G2 DA DD G5 R L C s, 12, 12, 1))} = \frac{2.(\texttt{G2.L.s} + \texttt{G1.DD} - \texttt{G2.G5})}{\texttt{L.C.s}^2 - \texttt{G5.C.s} - \texttt{DD}^2}$$

In the next section, we show another utilization of undirected SFG formalization to verify the transfer function of PANDA Vernier Resonator.

6 PANDA Vernier Resonator

The PANDA Vernier resonator is considered as a viable optical device for communication and signal processing [1,22]. It can also be employed as a force sensing application with a resolution in the range of micro-Newton. Figure 5 shows the configuration of the PANDA Vernier resonator where we have 4 optical directional couplers and 3 rings [1]. The directional couplers are optical devices that transfer the maximum possible optical power from one or more optical devices to another one in a selected direction. Optical rings are fiber rings that confine the light in a very small volume to perform different operations such as light amplification and wavelength filtering. Each coupler i is associated with

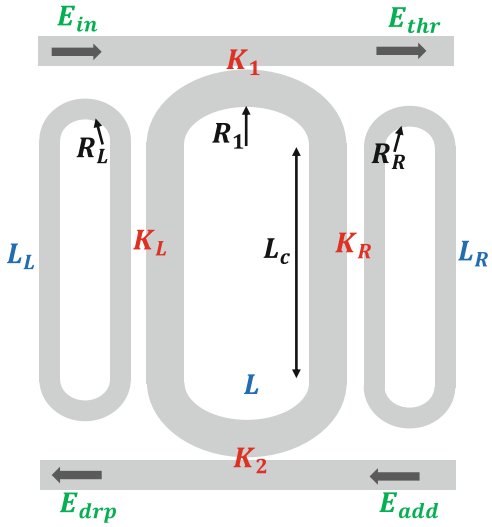

Fig. 5. The PANDA resonator

the factor k_i $(i = 1, 2, r, l)$ and the insertion loss γ. Consequently, the fraction of light passed through the through port of the coupler i is $C_i = \sqrt{(1-\gamma)(1-k_i)}$ and the fraction passed through the cross path is $S_i = \sqrt{(1-\gamma)k_i}$. Note that $C_i^2 + S_i^2 = 1 - \gamma \simeq 1$. Moreover, for each ring (i.e., main, right, and left) the parameter $e_i^p \equiv X_i Z^{-p}$ $(i = l, r)$ is the multiplication of the one round-trip loss coefficient $X_i = exp(\frac{-\alpha L_i}{2})$ and the transmission of light $Z^{-p} = exp(-j\varphi p)$, where $\varphi = kn_{eff}L$ is the phase shift and p is the integer resonant mode numbers. The associated *undirected* SFG of PANDA resonator [1] is shown in Fig. 6. We formally define the PANDA Vernier resonator model as follows:

Definition 11 (PANDA Vernier Resonator Model).

$$
\begin{aligned}
&\vdash \forall\ \texttt{er e el nr n nl sr s1 s2 sl cr c1 c2 cl} \in \mathbb{C}.\\
&\quad \texttt{PANDA\_model er e el nr n nl sr s1 s2 sl cr c1 c2 cl} =\\
&[(1, \texttt{c1}, 3); (1, -\texttt{j.s1}, 4); (2, \texttt{c1}, 4); (2, -\texttt{j.s1}, 3); (4, \sqrt[4]{e^n}, 9); (9, \texttt{cr}, 10);\\
&(9, -\texttt{j.sr}, 12); (12, \sqrt{\texttt{er}^{\texttt{nr}}}, 13); (13, \sqrt{\texttt{er}^{\texttt{nr}}}, 11); (11, -\texttt{j.sr}, 10); (11, \texttt{cr}, 12);\\
&(10, \sqrt[4]{e^n}, 5); (5, \texttt{c2}, 7); (5, -\texttt{j.s2}, 8); (6, \texttt{c2}, 8); (6, -\texttt{j.s2}, 7); (7, \sqrt[4]{e^n}, 14);\\
&(14, \texttt{cl}, 15); (14, -\texttt{j.sl}, 17); (17, \sqrt{\texttt{el}^{\texttt{nl}}}, 18); (18, \sqrt{\texttt{el}^{\texttt{nl}}}, 16); (16, \texttt{cl}, 17);\\
&(16, -\texttt{j.sl}, 15); (15, \sqrt[4]{e^n}, 2)]
\end{aligned}
$$

Based on this definition, we formally prove the optical transfer function for the through port of the PANDA Vernier resonator between nodes 1 and 3, as follows:

Theorem 5 (Transfer Function of the through port).
$\vdash \forall\ \texttt{er e el nr n nl sr s1 s2 sl cr c1 c2 cl} \in \mathbb{C}.$

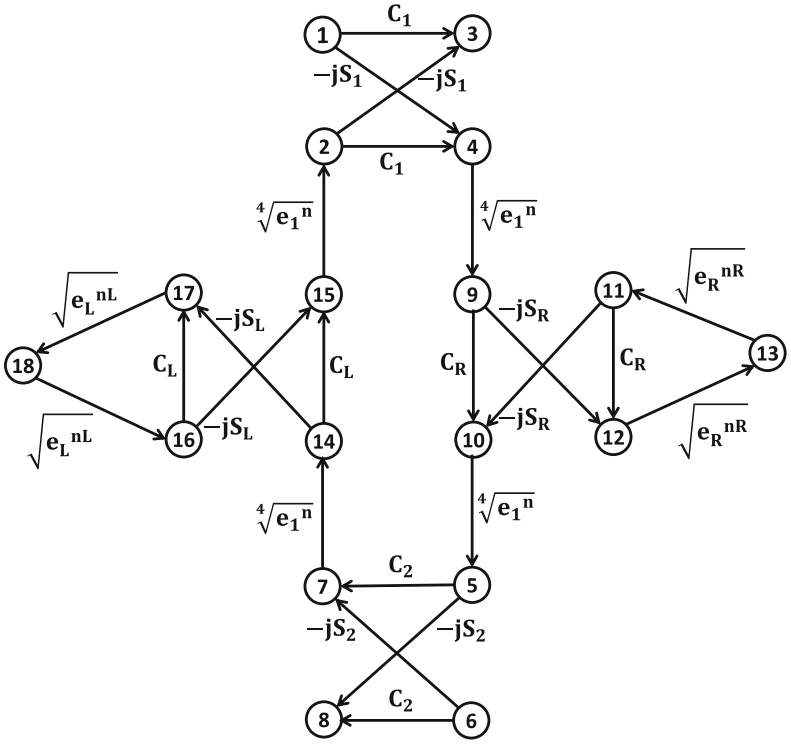

Fig. 6. Signal flow graph of the PANDA resonator

$$(\mathtt{c1}^2 + \mathtt{s1}^2) = 1 \wedge (\mathtt{c2}^2 + \mathtt{s2}^2) = 1 \wedge (\mathtt{cr}^2 + \mathtt{sr}^2) = 1 \wedge (\mathtt{cl}^2 + \mathtt{sl}^2) = 1 \implies$$
$$\mathtt{transfer\_function}\ (\mathtt{PANDA\_model\ er\ e\ el\ nr\ n\ nl\ sr\ s1\ s2\ sl\ cr}$$
$$\mathtt{c1\ c2\ cl}, 18, 3, 1) = \{\mathtt{c1}.(1 + \mathtt{cl.cr.er}^{\mathtt{nr}}.\mathtt{el}^{\mathtt{nl}} - \mathtt{cl.el}^{\mathtt{nl}} - \mathtt{cr.er}^{\mathtt{nr}}) +$$
$$\mathtt{c2.e}^{\mathtt{n}}.(\mathtt{cl.er}^{\mathtt{nr}} + \mathtt{cr.el}^{\mathtt{nl}} - \mathtt{cr.cl} - \mathtt{er}^{\mathtt{nr}}.\mathtt{e}^{\mathtt{nl}})\}$$
$$\overline{\{1 - \mathtt{cl.el}^{\mathtt{nl}} - \mathtt{cr.er}^{\mathtt{nr}} - \mathtt{cr.cl.c1.c2.e}^{\mathtt{n}} + \mathtt{cl.cr.er}^{\mathtt{nr}}.\mathtt{el}^{\mathtt{nl}} +}$$
$$\mathtt{cl.c1.c2.er}^{\mathtt{nr}}.\mathtt{e}^{\mathtt{n}} + \ \mathtt{cr.c1.c2.el}^{\mathtt{nl}}.\mathtt{e}^{\mathtt{n}} - \mathtt{c1.c2.er}^{\mathtt{nr}}.\mathtt{el}^{\mathtt{nl}}.\mathtt{e}^{\mathtt{n}}\}$$

We also formally verify the drop-port optical transfer function for the PANDA Vernier resonators between nodes 1 and 8, as follows:

Theorem 6 (Transfer Function of the drop-port).
$$\vdash \forall\ \mathtt{er\ e\ el\ nr\ n\ nl\ sr\ s1\ s2\ sl\ cr\ c1\ c2\ cl}\ \in \mathbb{C}.$$
$$(\mathtt{c1}^2 + \mathtt{s1}^2) = 1 \wedge (\mathtt{c2}^2 + \mathtt{s2}^2) = 1 \wedge (\mathtt{cr}^2 + \mathtt{sr}^2) = 1 \wedge (\mathtt{cl}^2 + \mathtt{sl}^2) = 1 \implies$$
$$\mathtt{transfer\_function}\ (\mathtt{PANDA\_model\ er\ e\ el\ nr\ n\ nl\ sr\ s1\ s2\ sl\ cr}$$
$$\mathtt{c1\ c2\ cl}, 18, 8, 1) = \{\mathtt{s1.s2.er}^{\mathtt{nr}}.\sqrt{\mathtt{e}^{\mathtt{n}}} - \mathtt{s1.s2.cl.el}^{\mathtt{nl}}.\mathtt{er}^{\mathtt{nr}}.\sqrt{\mathtt{e}^{\mathtt{n}}} -$$
$$\mathtt{cr.s1.s2}.\sqrt{\mathtt{e}^{\mathtt{n}}} + \mathtt{cr.cl.s1.s2.el}^{\mathtt{nl}}.\sqrt{\mathtt{e}^{\mathtt{n}}}\}$$
$$\overline{\{1 - \mathtt{cl.el}^{\mathtt{nl}} - \mathtt{cr.er}^{\mathtt{nr}} - \mathtt{cr.cl.c1.c2.e}^{\mathtt{n}} + \mathtt{cl.cr.er}^{\mathtt{nr}}.\mathtt{el}^{\mathtt{nl}} +}$$
$$\mathtt{cl.c1.c2.er}^{\mathtt{nr}}.\mathtt{e}^{\mathtt{n}} + \ \mathtt{cr.c1.c2.el}^{\mathtt{nl}}.\mathtt{e}^{\mathtt{n}} - \mathtt{c1.c2.er}^{\mathtt{nr}}.\mathtt{el}^{\mathtt{nl}}.\mathtt{e}^{\mathtt{n}}\}$$

To this point, we have completed our formal verification of the PANDA Vernier transfer functions based on our formalization of undirected SFG theory. Note that we can perform the stability and resonance analysis for the PANDA Vernier resonator as outlined for the z-source impedance network.

In engineering practices, verification and analysis tools must be largely automated to be effectively adopted which limits the usage of interactive theorem provers in industry. Therefore, in order to reduce the user interaction, we developed an automation procedure; `SFG_TAC` that carries `100`% of the proof steps for extracting loops and forward paths automatically and `90`% of the proof steps for the transfer function. Hence, our reported work can be considered as a one step towards an ultimate goal of building automatic tools which make use of interactive theorem provers as a certification tool in the design and analysis cycles of safety-critical real-world systems from different engineering and physical science disciplines (e.g., signal processing, control systems, power electronics, biology, optical and mechanical engineering). It is important to note that during our formal analysis of the two engineering applications, we were able to catch missing parts in the three transfer functions, given in Table 1 in reference [10]. We have also found a sign mismatch in [1]. We believe this to be a significant feature of our formalization as compared to the traditional analysis methods.

7 Conclusion

In this paper, we reported a generic formalization of undirected signal-flow-graph theory targeting any kind of engineering system that can be modeled in the form of a signal-flow-graph. We provided an overview of our formalization including the notion of loop extraction and the transposition of a signal-flow-graph. Consequently, we derive the transfer functions of two real-world engineering applications: (1) the PANDA Vernier resonator; and (2) the z-source impedance network. We described the formal analysis of the stability and resonance conditions for the z-source impedance network. Moreover, we formally verified that the z-source impedance network and its transpose have the same transfer function.

Our immediate future work is to develop upon the existing automation procedures to fully automate the proof process and to build a tool that makes use of these procedures for the analysis of engineering systems (e.g., photonics processors [21] and process engineering [3]). Another potential utilization of our formalization and developed automation tactics is to build a framework to certify the results produced by informal tools such as MATLAB based SFG analysis program (available at [9]).

References

1. Bahadoran, M., Ali, J., Yupapin, P.P.: Ultrafast all-optical switching using signal flow graph for PANDA resonator. Appl. Opt. **52**(12), 2866–2873 (2013)
2. Beillahi, S.M., Siddique, U.: Formal analysis of engineering systems based on signal-flow-graph theory (2016). http://hvg.ece.concordia.ca/projects/control/sfg.html
3. Beillahi, S.M., Siddique, U., Tahar, S.: Towards the application of formal methods in process engineering. In: Fun With Formal Methods, pp. 1–11 (2014)
4. Beillahi, S.M., Siddique, U., Tahar, S.: Formal analysis of power electronic systems. In: Butler, M., Conchon, S., Zaïdi, F. (eds.) ICFEM 2015. LNCS, vol. 9407, pp. 270–286. Springer, Cham (2015). doi:10.1007/978-3-319-25423-4_17
5. Binh, L.N.: Photonic Signal Processing: Techniques and Applications. Optical Science and Engineering. Taylor & Francis, London (2010)
6. Boca, P.P., Bowen, J.P., Siddiqi, J.I.: Formal Methods: State of the Art and New. Springer, London (2009)
7. Çapar, Ç., Goeckel, D., Paterson, K.G., Quaglia, E.A., Towsley, D., Zafer, M.: Signal-flow-based analysis of wireless security protocols. Inf. Comput. **226**, 37–56 (2013)
8. Dehghan, S.M., Mohamadian, M., Yazdian, A.: Hybrid electric vehicle based on bidirectional Z-source nine-switch inverter. IEEE Trans. Veh. Technol. **59**(6), 2641–2653 (2010)
9. Signal Flow Graph Similification Program for MATLAB (2014). http://www.mathworks.com/matlabcentral/fileexchange/22-mason-m
10. Gajanayake, C.J., Vilathgamuwa, D.M., Loh, P.C.: Small-signal and signal-flow-graph modeling of switched Z-source impedance network. IEEE Power Electron. Lett. **3**(3), 111–116 (2005)
11. Harrison, J.: Handbook of Practical Logic and Automated Reasoning. Cambridge University Press, New York (2009)
12. Hote, Y.V., Choudhury, D.R., Gupta, J.R.P.: Robust stability analysis of the PWM push-pull DC-DC converter. IEEE Trans. Power Electron. **24**(10), 2353–2356 (2009)
13. Isaksson, O., Keski-Seppala, S., Eppinger, S.D.: Evaluation of design process alternatives using signal flow graphs. In: ASME Design Engineering Technical Conferences, vol. 11(3), pp. 211–224 (2000)
14. Jennen, R., Max, S., Walke, B.: Frame delay distribution analysis of IEEE 802.11 networks using signal flow graphs. In: IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, pp. 2035–2040 (2010)
15. Mason, S.J.: Feedback theory, further properties of signal flow graphs. In: Proceeding of Institute of Radio Engineers, vol. 44, pp. 920–926 (1956)
16. Mason, S.J.: Feedback theory, some properties of signal flow graphs. In: Proceeding of Institute of Radio Engineers, vol. 41, pp. 1144–1156 (1953)
17. Mason, S.J., Zimmermann, H.J.: Electronic Circuits, Signals, and Systems. Wiley, New York (1960)
18. Peng, F.Z.: Z-Source inverter. In: IAS Annual Meeting. Conference Record of the Industry Applications Conference, vol. 2, pp. 775–781 (2002)
19. Rajakaruna, S., Jayawickrama, Y.R.L.: Designing impedance network of Z-source inverters. In: International Power Engineering Conference, vol. 2, pp. 962–967 (2005)
20. Schmid, H.: Circuit transposition using signal-flow graphs. In: IEEE International Symposium on Circuits and Systems, vol. 2, pp. II-25–II-28 (2002)

21. Siddique, U., Beillahi, S.M., Tahar, S.: On the formal analysis of photonic signal processing systems. In: Núñez, M., Güdemann, M. (eds.) FMICS 2015. LNCS, vol. 9128, pp. 162–177. Springer, Cham (2015). doi:10.1007/978-3-319-19458-5_11
22. Sirawattananon, C., Bahadoran, M., Ali, J., Mitatha, S., Yupapin, P.P.: Analytical vernier effects of a PANDA ring resonator for microforce sensing application. IEEE Trans. Nanotechnol. **11**(4), 707–712 (2012)
23. Tarjan, R.: Depth first search and linear graph algorithms. SIAM J. Comput. **1**(2), 146–160 (1972)
24. Tiernan, J.C.: An efficient search algorithm to find the elementary circuits of a graph. Commun. ACM **13**(12), 722–726 (1970)