A Fully Compositional Theory of Sequential Digital Circuits
Extended abstract

George Kaye, David Sprunger and Dan R. Ghica

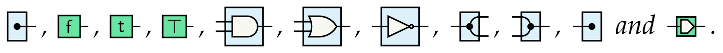
Contribution. Digital circuits are ubiquitous, so it may seem improbable that there are theoretical gaps remaining in their understanding. However, until recently we did not have a *fully compositional* model of digital circuits; by this we mean that a larger circuit can be built from smaller circuits without considering their internal structure. The sticking point was usually the presence of *non-delay-guarded feedback* [Mal94], which can lead to undesired behaviour and is often forbidden in circuit design. Nevertheless, using it carefully can lead to more efficient circuits [Rie04; RB12].

This line of work was inspired by that of Lafont on *Boolean* circuits [Laf03]; our current work is the direct successor to the more informal foundations of Ghica, Jung and Lopez, in which digital circuits are modelled as morphisms in a symmetric traced monoidal category (STMC) [GJ16; GJL17]. Our contributions are to make this work rigorous, and to this end we present three sound and complete semantics for digital circuits: a brand new *denotational semantics* for digital circuits based on stream functions with certain properties; a refinement and extension of the *operational semantics* presented in [GJL17] to operate on open circuits; and a *algebraic semantics* with which circuits can be brought to a pseudo-normal form.

Syntax. In full generality, the category of sequential digital circuits is generated over a *circuit signature* specifying the components that make up circuits and the signals that flow in the wires. Here we will restrict to a concrete signature representing *gate-level circuits*; rather than Boolean values, we use the four-valued system of *Belnap logic*.

Definition 1 (Belnap logic [Bel77])**.** *Let* $\mathbf{V} := \{\bot, \mathsf{f}, \mathsf{t}, \top\}$ *be the set of* Belnap values, *respectively represent* no signal *(a disconnected wire), a* false signal, *a* true signal, *and* both signals at once *(a short circuit). These values form an* information lattice $(\mathbf{V}, \sqcup, \sqcap)$ *and accept the usual operations of* $\wedge$, $\vee$ *and* $\neg$, *both illustrated in Fig. 1. Note that the operations are monotone.*

To aid in the presentation, we use the graphical calculus of *string diagrams* [JS91; JSV96; Sel11], in which terms equal by axioms of STMCs are depicted as isomorphism diagrams; 'only connectivity matters'.

Definition 2. *Let* $\mathbf{SCirc}_\Sigma$ *be the STMC freely generated over* .

The first four generators are the values in $\mathbf{V}$ where $\bot$ is denoted as $\boxed{\bullet}$. These are followed by AND, OR and NOT gates, and constructs for forking, joining and eliminating wires. The final generator is a *delay* of one unit of time. Light blue circuits $\boxed{f}$ are *combinational*: they model functions. Dark green circuits $\boxed{f}$ are *sequential*: they have state.

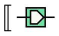
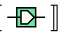
Denotational semantics. Circuits in $\mathbf{SCirc}_\Sigma$ are purely *syntax*; we now interpret circuit morphisms as *stream functions*. Streams $\sigma, \tau \in A^\omega$ can be viewed as functions $\mathbb{N} \to A$, so we will use the notation $\sigma(i)$ to obtain the i-th element of a stream.

We must identify the properties of stream functions that describe circuit behaviour. They are *causal*: the i-th element of a circuit can only be computed from the first i elements of the inputs. They are *monotone* with respect to the pointwise ordering on stream elements, as the components of circuits are monotone. Finally they are *finitely specified* as circuits have finitely many states: although streams themselves are infinite, eventually a function to compute the i-th element of an output stream based on the input stream must repeat.

Definition 3. *Let* **Stream** *be the PROP with morphisms* $m \to n$ *the monotone, causal, finitely specified functions* $(\mathbf{V}^m)^\omega \to (\mathbf{V}^n)^\omega$.

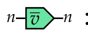
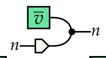
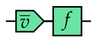
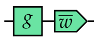
The first result we have is that this category is *traced*, so it is a suitable setting for modelling digital circuits with feedback.

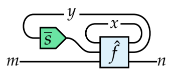
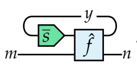
Proposition 4 ([GKS24], Prop. 25)**.** **Stream** *is traced.*

Semantics are assigned using a PROP morphism $[\![-]\!]^{\mathbf{S}} \colon \mathbf{SCirc}_\Sigma \to \mathbf{Stream}$. which acts on sequential components as $[\![\boxed{v}]\!]^{\mathbf{S}}\,()(0) := v$, $[\![\boxed{v}]\!]^{\mathbf{S}}\,()(k+1) := \bot$, $[\![$  $]\!]^{\mathbf{S}}\,(\sigma)(0) := \bot$, and $[\![$  $]\!]^{\mathbf{S}}\,(\sigma)(k+1) := \sigma(k)$.

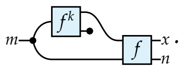
Definition 5 (Denotational equivalence)**.** *Two circuits are* denotationally equivalent $\boxed{f} \approx \boxed{g}$ *if* $[\![\boxed{f}]\!]^{\mathbf{S}} = [\![\boxed{g}]\!]^{\mathbf{S}}$.

Corollary 6 ([GKS24], Cor. 67)**.** *There is an isomorphism of PROPs* $\mathbf{SCirc}_\Sigma/\!\approx\; \cong \mathbf{Stream}$.

Operational semantics. To better relate the internal structure and behaviour of a circuit, we will now define an *operational semantics* which evaluates circuits using *reductions*. For $\overline{v} \in \mathbf{V}^n$, we write  :=  for a 'register'; we want to find the outputs of a circuit given some input registers, i.e. reductions such that  $\rightsquigarrow^\star$ .

Definition 7. *A circuit is in* pre-Mealy form *if it is in the form*  *and in* Mealy form *if it is in the form* .

Any circuit can be translated into pre-Mealy form using the (Mealy) rule in Fig. 2. However, the *non-delay-guarded feedback* blocks attempts to translate this into Mealy form; note that since the 'core' circuit $\boxed{\hat{f}}$ is combinational, the trace cannot simply be incorporated inside it. As the components of our circuits are monotone and we work in a finite lattice, we can eliminate non-delay-guarded feedback using the Kleene fixed-point theorem.

Definition 8 (Iteration). *For a combinational circuit* ${}^{x}_{m}\boxed{f}^{x}_{n}$*, let its* nth iteration ${}_{m}\boxed{f^n}^{x}_{n}$ *be defined inductively over* n *as* ${}_{m}\boxed{f^0}^{n}_{x} :=$ ${}^{\bullet}_{m}\boxed{f}^{x}_{n}$ *and* ${}_{m}\boxed{f^{k+1}}^{x}_{n} :=$ 

The (IF) rule in Fig. 2 uses the fact that the longest chain in $\mathbf{V}^x$ is $2x+1$ to replace non-delay-guarded feedback with a sequence of iterations; this means that any circuit can be reduced to one in Mealy form.

To see how a circuit in Mealy form 'processes' a value, we precompose it with a register containing the inputs. Applying the (Str) ('streaming') rule from Fig. 2 to this creates two copies of the circuit: one for what is happening 'now' and the other for what is happening 'later'. The 'now' circuit is some values applied to a combinational circuit: the four rules on the right of Fig. 2 can then be applied to reduce this to an output value and a next state.

Corollary 9 (Productivity ([GKS24], Cor. 91))**.** *For a sequential circuit* ${}_{m}\boxed{f}_{n}$ *and inputs* $\overline{v} \in \mathbf{V}^m$*, there exists* $\overline{w} \in \mathbf{V}^n$ *and sequential circuit* ${}_{m}\boxed{g}_{n}$ *such that* ${}_{m}\boxed{\overline{v}}\boxed{f}_{n} \rightsquigarrow^{\star} {}_{m}\boxed{g}\boxed{\overline{w}}_{n}$ *by applying* Mealy, IF *and* Str *once followed by the* (Fork), (Join), (Elim) *and the* ($\mathsf{Prim}_{\mathcal{I}}$) *rules exhaustively.*

The general procedure for processing the inputs to a circuit is illustrated in Fig. 3. This can be applied repeatedly in order to determine the outputs of a circuit over time, and to tell if two circuits are *observationally* equivalent.

Definition 10 (Observational equivalence)**.** *Two circuits with no more than c delay components are* observationally equivalent $\boxed{f} \sim \boxed{g}$ *if productivity produces the same outputs for all inputs of length* $|\mathbf{V}^c| + 1$.

Corollary 11 ([GKS24], Cor. 67)**.** *There is an isomorphism of PROPs* $\mathbf{SCirc}_{\Sigma}/\!\sim\; \cong \mathbf{SCirc}_{\Sigma}/\!\approx$.

Algebraic semantics. The previous section gives an *exponential* upper bound on checking observational equivalence. It is often more efficient to use *equational reasoning* to determine if two circuits have the same behaviour. An equational theory was presented [GJ16] but here we are guided by the stream semantics. The notion of Mealy form will once again come in useful, so we first reframe the (Mealy) rule in terms of equations shown in Fig. 4. The usual strategy to show that an equational theory is sound and complete is to define some sort of 'normal form' into which all terms can be translated.

Definition 12. *Let* $|-|$ *be a fixed procedure mapping a function* $f\colon \mathbf{V}^m \to \mathbf{V}^n$ *to a circuit* $\boxed{|f|}$ *such that* $[\![\boxed{|f|}]\!]^{\mathbf{S}}(\sigma)(i) = f(\sigma(i))$.

One can think of $|f|$ as the 'canonical syntactic representation' of a function; if a circuit is in this form it is easy to tell what function it represents. For Belnap circuits, this normal form is a variation of the standard disjunctive normal form for Booleans [GKS24, App. A].

Lemma 13. *Any combinational Belnap circuit* $\boxed{f}$ *is equal to a circuit in the image of* $|-|$ *by the equations in Fig. 5.*

A circuit in Mealy form has a *state* and a *combinational core*. By using productivity, one obtains a set $S \subseteq \mathbf{V}^y$ of *circuit states* that the circuit assumes over time. We need equations to translate between different state sets syntactically. Two states are equivalent with respect to a combinational core if the produced outputs are equal and the transitions are equivalent; an *encoding function* $\gamma\colon \mathbf{V}^x \to \mathbf{V}^y$ is a function with an inverse *decoder* $\gamma^{-1}\colon \mathbf{V}^y \to \mathbf{V}^x$ such that $\gamma^{-1}(\gamma(s))$ is equivalent to s.

Lemma 14. *For any encoding function* γ *with inverse* γ^{-1}*; then the equations in Fig. 6 are sound.*

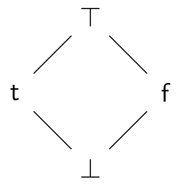
The encoding equations can be used to translate a circuit into one with a bisimilar set of circuit states. However, the cores of these circuits still may not be behaviourally equivalent; they just agree on the set of circuit states. We need one final family of equations to 'hot-swap' combinational cores that behave the same on this set of states.

Lemma 15. *The family of* restriction equations (Res) *in Fig. 7 is sound.*

We write $\mathcal{E}$ for the equations in Figs. 4 to 7, which are enough for a sound and complete equational theory. Given two denotationally equivalent circuits, we translate them into Mealy form, use the encoding equation to map between their state sets, then use the restriction equation if necessary.

Theorem 16. *There is an isomorphism of PROPs* $\mathbf{SCirc}_{\Sigma}/\mathcal{E} \cong \mathbf{SCirc}_{\Sigma}/\!\approx$.

Conclusion. The fully compositional theory of sequential digital circuits sets a foundation which will hopefully unlock new methods for working with circuits, such as partial evaluation techniques. One avenue we are particuarly interested in is that of *automating* circuit reasoning using *string diagram graph rewriting*, which has been studied in-depth recently [Bon+22; GK23].



∧	⊥	f	t	⊤
⊥	⊥	f	⊥	f
f	f	f	f	f
t	⊥	f	t	⊤
⊤	f	f	⊤	⊤

¬	
⊥	⊥
t	f
f	t
⊤	⊤

∨	⊥	f	t	⊤
⊥	⊥	⊥	t	t
f	⊥	f	t	⊤
t	t	t	t	t
⊤	t	⊤	t	⊤

Figure 1: The lattice structure on **V**, and the truth tables of Belnap logic gates [Bel77].

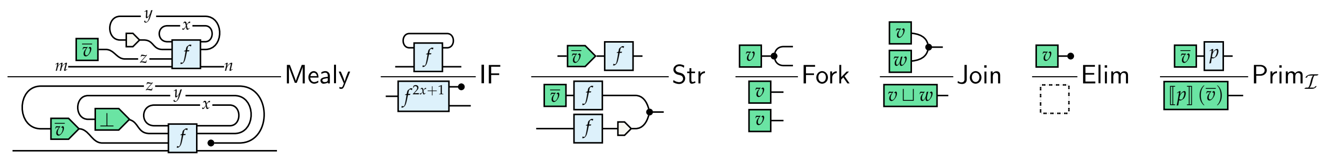


Figure 2: Productive reductions

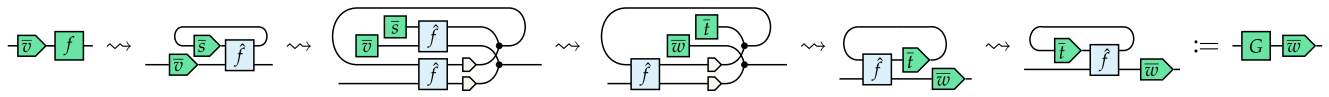


Figure 3: Processing inputs to a circuit

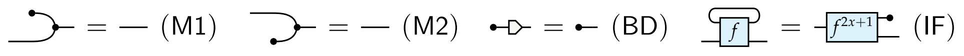


Figure 4: Set of *Mealy equations*

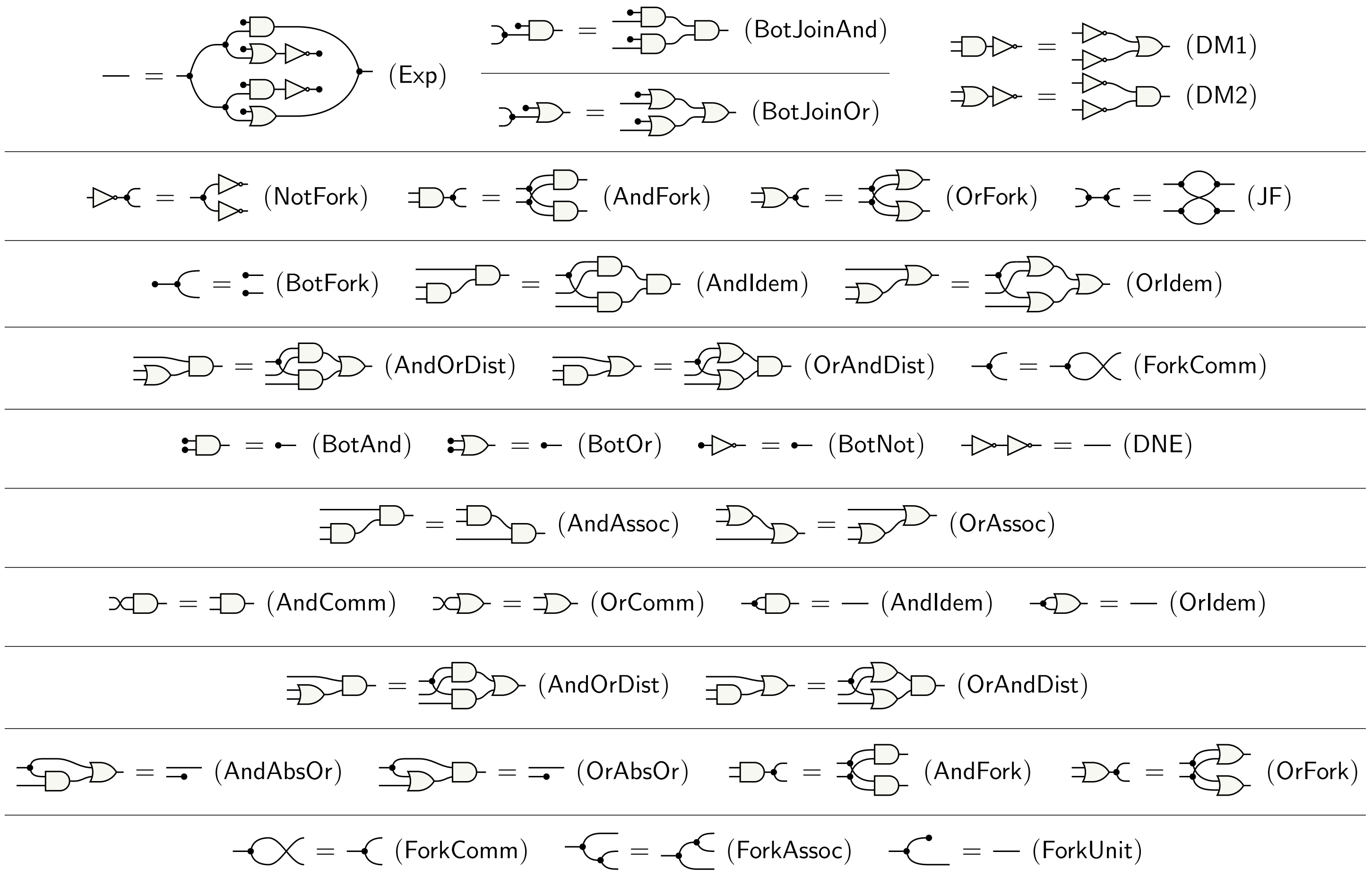


Figure 5: Set of *Belnap normalisation equations*

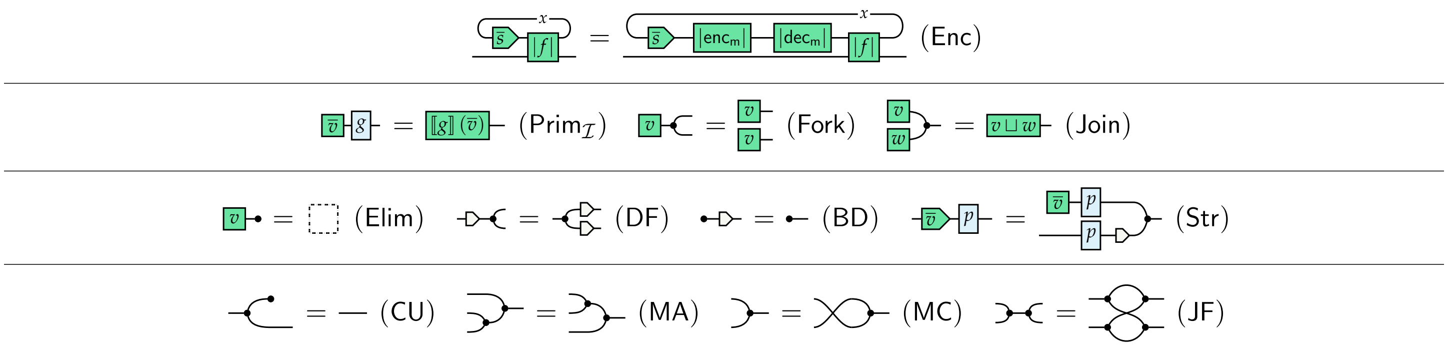
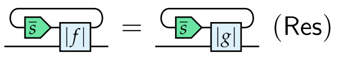


Figure 6: Equations for encoding circuit states



where f and g act the same on circuit states

Figure 7: Family of *restriction* equations

References

[Bel77] Nuel D. Belnap. "A Useful Four-Valued Logic". In: *Modern Uses of Multiple-Valued Logic*. Ed. by J. Michael Dunn and George Epstein. Episteme. Dordrecht: Springer Netherlands, 1977, pp. 5–37. ISBN: 978-94-010-1161-7. DOI: 10.1007/978-94-010-1161-7_2.

[Bon+22] Filippo Bonchi et al. "String Diagram Rewrite Theory I: Rewriting with Frobenius Structure". In: *Journal of the ACM* 69.2 (Mar. 10, 2022), 14:1–14:58. ISSN: 0004-5411. DOI: 10.1145/3502719.

[GJ16] Dan R. Ghica and Achim Jung. "Categorical Semantics of Digital Circuits". In: *2016 Formal Methods in Computer-Aided Design (FMCAD)*. 2016 Formal Methods in Computer-Aided Design (FMCAD). Oct. 2016, pp. 41–48. DOI: 10.1109/FMCAD.2016.7886659.

[GJL17] Dan R. Ghica, Achim Jung, and Aliaume Lopez. "Diagrammatic Semantics for Digital Circuits". In: *26th EACSL Annual Conference on Computer Science Logic (CSL 2017)*. Ed. by Valentin Goranko and Mads Dam. Vol. 82. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2017, 24:1–24:16. ISBN: 978-3-95977-045-3. DOI: 10.4230/LIPIcs.CSL.2017.24.

[GK23] Dan R. Ghica and George Kaye. "Rewriting Modulo Traced Comonoid Structure". In: *8th International Conference on Formal Structures for Computation and Deduction (FSCD 2023)*. Ed. by Marco Gaboardi and Femke van Raamsdonk. Vol. 260. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023, 14:1–14:21. ISBN: 978-3-95977-277-8. DOI: 10.4230/LIPIcs.FSCD.2023.14.

[GKS24] Dan R. Ghica, George Kaye, and David Sprunger. *A Fully Compositional Theory of Sequential Digital Circuits: Denotational, Operational and Algebraic Semantics*. Version 5. Jan. 30, 2024. DOI: 10.48550/arXiv.2201.10456. arXiv: 2201.10456 [cs, math]. preprint.

[JS91] André Joyal and Ross Street. "The Geometry of Tensor Calculus, I". In: *Advances in Mathematics* 88.1 (1991), pp. 55–112. ISSN: 0001-8708. DOI: 10.1016/0001-8708(91)90003-P.

[JSV96] André Joyal, Ross Street, and Dominic Verity. "Traced Monoidal Categories". In: *Mathematical Proceedings of the Cambridge Philosophical Society* 119.3 (Apr. 1996), pp. 447–468. ISSN: 1469-8064, 0305-0041. DOI: 10.1017/S0305004100074338.

[Laf03] Yves Lafont. "Towards an Algebraic Theory of Boolean Circuits". In: *Journal of Pure and Applied Algebra* 184.2 (Nov. 1, 2003), pp. 257–310. ISSN: 0022-4049. DOI: 10.1016/S0022-4049(03)00069-0.

[Mal94] S. Malik. "Analysis of Cyclic Combinational Circuits". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 13.7 (July 1994), pp. 950–956. ISSN: 1937-4151. DOI: 10.1109/43.293952.

[RB12] Marc D. Riedel and Jehoshua Bruck. "Cyclic Boolean Circuits". In: *Discrete Applied Mathematics* 160.13 (Sept. 1, 2012), pp. 1877–1900. ISSN: 0166-218X. DOI: 10.1016/j.dam.2012.03.039.

[Rie04] Marc D. Riedel. "Cyclic Combinational Circuits". PhD thesis. United States – California: California Institute of Technology, May 27, 2004. 112 pp. ISBN: 9780496071005. URL: https://www.proquest.com/docview/305199547/abstract/B04FE380B2224E4DPQ/1.

[Sel11] Peter Selinger. "A Survey of Graphical Languages for Monoidal Categories". In: *New Structures for Physics*. Ed. by Bob Coecke. Lecture Notes in Physics. Berlin, Heidelberg: Springer, 2011, pp. 289–355. ISBN: 978-3-642-12821-9. DOI: 10.1007/978-3-642-12821-9_4.