

Polynomial Fault Detection Without Bounded Width: Exact Reachable Sets in Priority Controllers

Idris Ali Shaik

Independent researcher (unaffiliated)

shaikidris@apache.org

Preprint, version 1.1, 22 October 2026.

Abstract

Circuit-width restrictions explain several polynomial cases of automatic test pattern generation. We give a different explanation for an arbitrary-size extension of the c432 priority controller, including a subfamily with linear treewidth and net cut-width in its native layout. For a specified first-stage bridge fault, we characterize exactly the reachable qualifier sets and construct a primary input realizing each one. On drive-one inputs where the first selector flag changes, the healthy set is the target singleton; the faulty sets are precisely the target-containing sets with one request bus and all nonempty sets with at least two buses. Drive zero is undetectable. This image reduces detection to comparing decoder values on sets. For supplied guarded decoders, built from zero, positive channel leaves, disjunction and conjunction of one computed child with negative channel literals, supporting paths and dual-Horn reasoning find a detecting set in $O((n + q)G)$ time for n channels, q outputs and counted decoder size G , or $O(G)$ for formula forests. Unsatisfiable instances of the complete guarded native CNF encoding have resolution refutations with $O(N^3)$ inferences, where N is the encoding size. Detection is P-complete, while minimum-set optimization and counting detecting sets are NP-hard and #P-complete, respectively. A specified final-qualifier stuck-at fault makes detection NP-complete on shared guarded decoders. The exact first-stage bridge-image statements are formalized in Lean and registered as PALOMAR-2026-10-19-000010, version 1.

1. Introduction

Automatic test pattern generation asks for a primary input on which a fault-free circuit and its faulty copy disagree at an observed output. The decision problem is NP-complete for general combinational circuits [IS75, FT82]. Several polynomial cases are explained by structural restrictions: fanout-free logic, Fujiwara's k -bounded circuits, and bounded-input blocks with bounded block-graph treewidth [Fuj90, CAB90]. Under bounded fan-in and fan-out assumptions, [PCK99] obtains polynomial time for logarithmic net cut-width. The question here is whether detection can remain polynomial in a natural controller family whose native circuits escape these width restrictions.

We study an arbitrary-size extension of the c432 selector. The published description calls c432 a 27-channel interrupt controller: three 9-bit request buses sharing nine enable bits [HYH99]. These give $n = 9$, $b = 3$ in our notation. Hansen and Hayes used that high-level description for test generation on the fixed netlist [HH95]. We retain its selector recurrence for every channel count n and bus count b . The qualifier set contains every enabled channel with a lexicographically maximal request vector. Our outputs are supplied decoders of these qualifiers, before the later channel encoder in the published model. The fault is the first-stage bridge interface extracted from the Nemesis encodings [FL91, Lar92]. It has a target channel t , a disjoint nonempty set of driver channels, and a drive $d \in \{0, 1\}$. Section 3 specifies its equations and the otherwise unrestricted primary inputs. We call the fault-free copy *healthy* and the specified complete gate encoding the *native encoding*.

The width obstruction already occurs in a one-bus, single-output subfamily with no shared internal

decoder gates. Proposition 6.11 gives native layouts of size N with treewidth and net cut-width $\Omega(N)$. For every partition into blocks with at most K input signals and block-graph treewidth w , it also gives $K(w + 1) = \Omega(N)$. These instances therefore lie outside fixed (K, w) and logarithmic cut-width restrictions, while their supplied-structure detection problem is solvable in linear time. Proposition 6.10 gives a separate $K_{n,b}$ minor in the native CNF’s primal and incidence graphs. These comparisons concern the specified layouts and encodings; they do not assert width lower bounds for every equivalent representation.

The central contribution is an exact, size-uniform description of the fault’s image, together with a primary input realizing every admissible qualifier set. It removes the need to search the physical controller when deciding which decoder values can change. For the guarded decoder syntax, this comparison is solved by supporting paths and the classical linear-time Horn algorithm [DG84]. The contribution is the image characterization and its constructive composition with those methods. The two-copy SAT encoding [Lar92], Horn satisfiability, and the source hardness results used below are established tools. Section 10 gives the precise attribution and the limits of the priority assessment.

The exact first-stage bridge-image statements have a registered Lean formalization [Sha26a]. Registered selections also cover supplied-structure detection and extensions [Sha26b]; Section 13 lists their precise scope and the other formal packages, separately from whole-statement correspondence.

1.1. From the image to detection and its boundaries

Theorem 4.1 supplies the common interface. An eligible set is the set of channels whose qualifiers are one. The first selector flag records whether the first bus has an enabled requester in the healthy copy. Drive zero leaves the observed qualifiers unchanged. If the first flags agree, all subsequent stages and qualifiers agree as well. For drive one, whenever those flags differ, the healthy set is the target singleton $\{t\}$. The faulty set can be any set containing t with one bus, or any nonempty set with at least two buses. The realization part of the theorem produces a primary input for each set. Corollary 4.2 therefore reduces detection to finding a reachable set on which a decoder differs from its value on $\{t\}$.

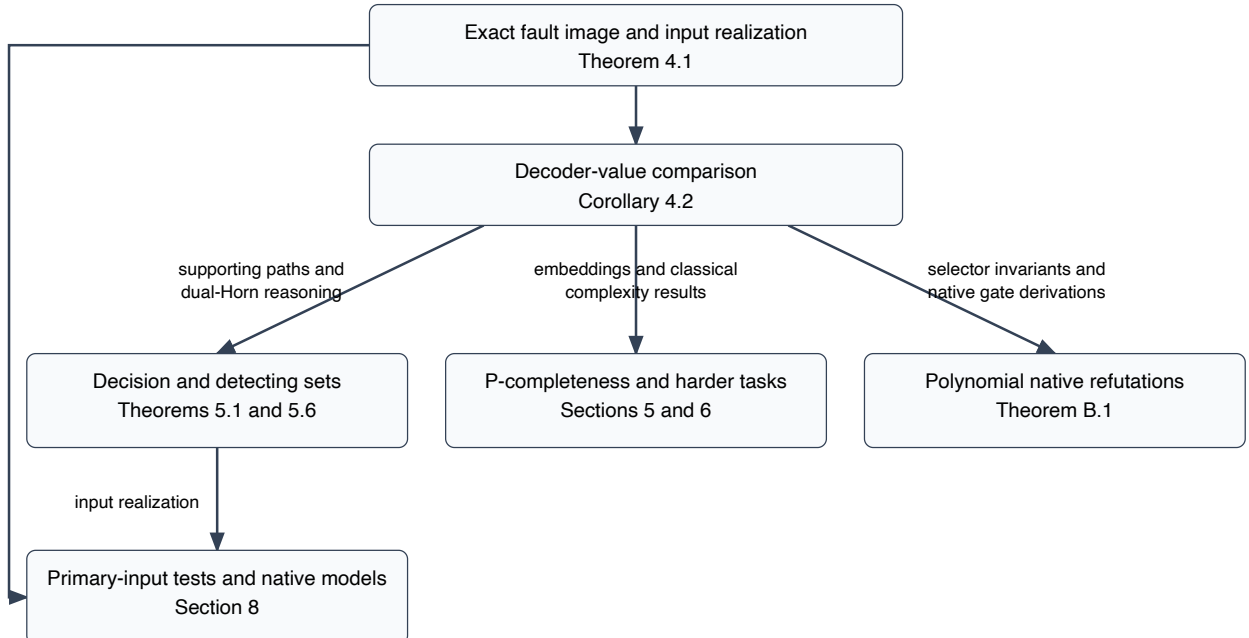


Figure 1. Main dependencies. The image is the common interface; the branches require the additional ingredients named on the arrows. The native refutation branch also uses the guarded-decoder argument. The width constructions in Propositions 6.10 and 6.11 provide the structural comparison independently of the decision procedure.

A guarded decoder is an acyclic circuit built from zero, positive channel leaves, disjunction, and conjunction of one computed child with negative channel guards. For a healthy-zero output, every true evaluation has a supporting path ending at one positive channel. This reduces the search to singletons, with the forced target adjoined in the one-bus case. For a healthy-one output, making it false is a dual-Horn condition, solved by a greatest-model computation. Theorem 5.1 constructs a detecting eligible set in $O((n + q)G)$ time; the realization step writes a primary-input test in an additional $O(N)$. Theorem 5.6 reduces the decoder work to $O(G)$ when internal gates are not shared. Repeated channel leaves are allowed. Thus even the linear-width subfamily above admits linear-time detection and test construction once its circuit roles are supplied.

The image also makes the boundaries precise. Proposition 5.3 combines the dual-Horn embedding with monotone circuit-value P-completeness [Gol77; GHR95, Theorem 6.2.2]. It already applies to one healthy-one formula with at most two negative guards per term. Together with the polynomial upper bound, this gives P-completeness under logspace reductions. Uniform polylogarithmic parallel time with polynomially many processors would imply $P = NC$, and deterministic logarithmic space would imply $P = L$; no superlinear sequential lower bound follows. Separately, the graph-cover embedding transfers classical vertex-cover optimization and counting hardness and the expander-CNF representation bound [Kar72, PB83, BCMS15]. These use additional reductions on the reachable sets. The native resolution bound uses explicit gate derivations of the selector invariants and guarded comparison, rather than only the semantic equivalence.

The remaining results delimit the model. Section 7 classifies the basis consisting of disjunction, conjunction of two computed children, and negative channel guarding; it also bounds the effect of limiting the number of those conjunctions. Proposition 4.5 extends the image to a specified stage-local bridge. Theorem 4.6 describes the exact pairs for a final-qualifier stuck-at fault, generalizing the fixed observation in Theorem 4.4. On this pair image guarded-DAG detection is NP-complete, whereas unshared formulas remain polynomial (Theorems 7.8 and 7.9). Appendix D gives a compatible-path algorithm parameterized by the multi-positive clauses of the opposite zero encoding. Appendix A specifies how to verify a supplied representation against a CNF; acceptance permits the guarded decision procedure. Failure to recover or verify a representation returns **OPEN** and establishes neither nonmembership nor unsatisfiability. The worst-case cost of finding the roles remains open.

The table lists one theorem for each question. The bounds that decide, extend, or enumerate take the selector roles and the decoder as input.

Question	Result	Statement
Which faulty qualifier sets occur?	Exact sets and a primary input for each	Theorem 4.1
Does a detecting test exist for the first-stage bridge?	$O((n + q)G)$; unshared formulas $O(G)$	Theorems 5.1, 5.6
Which images extend to other specified fault interfaces?	Stage-local bridge image; exact final-qualifier stuck-at pairs	Proposition 4.5, Theorem 4.6
Does final-qualifier stuck-at detection remain polynomial?	NP-complete on guarded DAGs; $O(G^2)$ on formula forests	Theorems 7.8, 7.9

Question	Result	Statement
Do bounded-width conditions cover the family?	Native treewidth at least $\min(n, b)$; a linear-width subfamily with $K(w + 1) = \Omega(N)$ for every gate partition	Propositions 6.10, 6.11
How hard is this polynomial decision?	P-complete under logspace reductions, for one healthy-one formula	Proposition 5.3
Can a partial qualifier assignment be extended, and can the sets be enumerated?	Polynomial extension and polynomial-delay enumeration	Theorem 5.4, Corollary 5.5
Can detecting eligible sets be minimized or counted, full native models counted, or all detecting inputs compactly represented?	NP-hard eligible-set minimization, #P-complete eligible-set and native-model counting, exponential DNNF lower bound	Corollary 6.5, Theorems 6.8, 6.9
Which added gates preserve a polynomial algorithm?	Each proper subset of the three operations is polynomial; all three are NP-complete; k binary conjunctions of computed children give XP, with an FPT/W[1] split by the healthy value	Theorems 7.4–7.6
Do UNSAT instances of the guarded family’s specified native encoding have short proofs?	An ordinary resolution refutation of $O(N^3)$ inferences	Theorem B.1

Here G counts all n channel inputs, decoder nodes, and child, guard and output incidences; N counts variables and literal occurrences in the native CNF. The formula-forest and bounded-conjunction algorithms read an indexed decoder DAG and are separate from native CNF recognition and proof export. Their indices occupy logarithmic-size machine words. The bounds count word operations and working words. Bit-level reading, the bit cost of weights, and the size of the written output are charged where the statements say so. The upper bounds and the P-completeness, NP-completeness, #P-completeness, and representation bounds are unconditional. Reading W[1]-hardness as the absence of an FPT algorithm requires $\text{FPT} \neq \text{W}[1]$. Corollary 7.7 uses the Exponential Time Hypothesis for its exponent, through the hypothesis of [CHKX06, Theorem 5.5] recorded in Section 2. No result separates P from NP.

1.2. Scope

Two companions use the qualifier sets as an application and are not required here. The reachability companion [Sha26B] studies paths that avoid channel-specific vertices and gives conditional fine-grained bounds under separately named hypotheses. The factor companion [Sha26C] studies conjunctions of guarded factors and transfers binary-CSP and graph-class bounds to that representation. The SATLIB audit [Sha26U] and the structural note [Sha26S] record benchmark and width comparisons. Neither those comparisons nor the timing panel proves the uniform theorems. On every generated decision instance in Section 9, MiniSat and Glucose were faster than the family implementation. The panel checks the implementation and records its cost.

Section 2 fixes the notation used in the reductions and the proof bound. Section 3 defines the controller. Section 4 proves the reachable sets. Section 5 gives the decision procedure and its P-completeness. Sections 6 and 7 give the boundaries for optimization, counting, compilation, width, and gate basis, including the stuck-at boundary. Appendix D studies compatible paths and zero constraints. Section

8 and Appendices A and B give the CNF contract and the resolution construction. Section 9 reports the finite checks. Section 10 matches each cited theorem with the use made of it. Section 11 lists the open obligations.

2. Preliminaries

Circuits and CNF. The symbol $\oplus$ denotes XOR and multiplication denotes AND. Gates are AND, OR, and XOR with bounded fan-in, encoded by the standard complete clause blocks (Tseitin encoding). A *detection formula* conjoins a healthy copy, a faulty copy, a fault-activation constraint, and the requirement that some observed output differs [Lar92]. The bridge interface is an abstract Boolean model extracted from the SATLIB/Nemesis encodings [FL91, Lar92]. Admitting both drive values is a mathematical generalization; it does not assert that a particular physical CMOS bridge has both behaviors.

For signed input literals l_j and a fresh output variable z , the complete gate blocks are as follows. Constants use a unit clause.

Gate	Clauses
$z = \bigwedge_j l_j$	$\neg z \vee l_j$ for each j ; $z \vee \bigvee_j \neg l_j$
$z = \bigvee_j l_j$	$z \vee \neg l_j$ for each j ; $\neg z \vee \bigvee_j l_j$
$z = x \oplus y$	$x \vee y \vee \neg z$; $\neg x \vee \neg y \vee \neg z$; $x \vee \neg y \vee z$; $\neg x \vee y \vee z$
$z = l$	$\neg z \vee l$; $z \vee \neg l$

Larger conjunctions and disjunctions use acyclic bounded-fan-in trees. Complete blocks specify both directions of a gate identity; an isolated implication is insufficient for role recovery. Signed aliases can be contracted after their two-clause equality or disequality is checked.

Functional extension. In a finite acyclic circuit with one complete defining gate per internal variable, every valuation of all free roots extends uniquely to the defined gate variables. Topological evaluation constructs the extension; induction in the same order proves uniqueness. Appending fixed-drive and observation assertions either retains that extension or excludes it. Optional activity variables and other existentially projected private auxiliaries may have multiple extensions. The native-model counting theorem uses the complete functional miter.

Compilation. A DNNF is a negation-normal-form circuit whose AND children have disjoint variable sets. OR children need not be disjoint or deterministic. Its size here is its wire count. An OBDD tests variables in one fixed order along every path. These representations encode all satisfying assignments, rather than only a decision or one witness.

Horn and dual-Horn. A clause is *Horn* if it has at most one positive literal and *dual-Horn* if it has at most one negative literal. A formula is *renamable Horn* if consistently complementing some variables makes every clause Horn. A *2-CNF* has at most two literals per clause, including unit and empty clauses. A satisfiable dual-Horn formula has a unique greatest model, which is computable in linear time by downward propagation from the all-true assignment [DG84].

Width parameters.

- *Treewidth* and *pathwidth* are taken on the primal graph of a formula or the undirected producer-consumer graph of a circuit. The latter has vertices for gates, primary inputs and primary outputs, with an edge for each signal connection; signs do not affect adjacency.

- *Cut-width* is defined on a circuit hypergraph whose vertices are gates, primary inputs and outputs, and whose hyperedges are signal nets. Under a linear vertex order it is the maximum number of nets crossing a cut [PCK99].
- The *two-section graph* of a hypergraph has the same vertices and joins exactly those pairs of distinct vertices that share a hyperedge.
- For a linear vertex order, its *vertex separation* is the maximum number of vertices to the left of a cut that have a neighbor to the right. The vertex separation of a graph is the minimum over all such orders; it equals pathwidth [Kin92].
- We distinguish blocks with at most K inputs and a block graph of treewidth at most w [CAB90] from Fujiwara’s K -bounded circuits, which also restrict reconvergent paths in the block DAG [Fuj90, PCK99]. The quantitative block obstruction below applies to each fixed (K, w) . For a partition of the gates, the block graph joins distinct blocks whenever a gate in one supplies a signal to a gate in the other. A block input is a distinct signal entering from outside the block, including a primary-input signal; repeated uses of that signal count once. If a block-graph convention also joins consumers sharing an external signal, the resulting graph contains this producer-consumer block graph. Adding such edges cannot decrease treewidth, so Proposition 6.11 applies to that convention as well.

Resolution. Refutations are DAG-like with no extension variables. A clause passes the *reverse unit propagation* (RUP) test when adding units that falsify all its literals to the current clauses yields a unit-propagation contradiction. For a non-tautological requested clause, reversing this propagation conflict gives an ordinary-resolution derivation of a subclause of that clause. Stronger subclauses are retained. Tautological goals are omitted; weakening is not used to force an exact requested clause.

Parameterized bounds. FPT means a running time $f(k)G^c$ for a computable function f and a constant c independent of k ; XP allows the exponent to depend on k . Parameterized Clique is W[1]-complete. The Exponential Time Hypothesis (ETH) asserts that 3-SAT on v variables has no $2^{o(v)}$ -time algorithm, apart from a polynomial input-length factor. Chen, Huang, Kanj and Xia [CHKX06, Theorem 5.5] exclude $f(r)m^{o(r)}$ time for Clique, for any computable f , unless every SNP problem is solvable in subexponential time; m is the input length. ETH implies the required non-collapse assumption. Since a graph on p vertices has polynomial encoding length, this supplies the $f(r)p^{o(r)}$ exclusion under ETH.

3. The controller family

The model follows the high-level c432 description [HYH99]. Channel i , with $1 \leq i \leq n$, has an enable e_i and requests r_{ij} on buses $1 \leq j \leq b$, where $n, b \geq 1$. All of these are independent primary inputs, with no additional restrictions. Write $x_i = e_i r_{i1}$ and $P_1 = \bigvee_i x_i$.

Selector stages. The flags and masks of one copy are

$$\begin{aligned} q_{i1} &= P_1 \oplus \neg x_i, \\ t_{ij} &= e_i r_{ij} \wedge \bigwedge_{k < j} q_{ik}, \quad P_j = \bigvee_i t_{ij}, \\ q_{ij} &= P_j \oplus \neg t_{ij} \quad (j \geq 2). \end{aligned}$$

and the final *qualifiers* are $h_i = e_i \wedge \bigwedge_j (\neg P_j \vee r_{ij})$. For healthy flags this recurrence implements lexicographic priority selection. Write $t_{i1} = x_i$. Since $a \oplus \neg c$ is one exactly when $a = c$, the mask q_{ij}

is one exactly when $P_j = t_{ij}$, for every j . For an enabled channel whose earlier masks are all one, $t_{ij} = r_{ij}$; for every other channel, $t_{ij} = 0$. If no channel is enabled, every flag and qualifier is zero. Otherwise the enabled channels whose masks through bus j are all one form the population C_j of the prefix-survivor invariant below, started from the enabled channels, and P_j is its flag p_j . The *eligible set* of an input is $\{i : h_i = 1\}$. Empty conjunctions and disjunctions have values one and zero, respectively. In the statements below, the fault-free copy is the healthy copy.

In the c432 behavioral model [HYH99], modules M1–M4 provide this selector and qualifier interface. With $n = 9$, $b = 3$ and active-high requests $r_{i1} = \neg A_i$, $r_{i2} = \neg B_i$, $r_{i3} = \neg C_i$, the flags are PA, PB, PC, the first two mask vectors are X1, X2, and the qualifier vector is the complement of I . The published description counts the three request buses as 27 request channels; our n counts their nine shared enabled positions. The qualifier vector retains all ties between maximal request vectors. Module M5 subsequently maps that vector to the four encoded channel outputs. Our image theorem concerns the qualifier vector before that encoding; detection observes the specified functions of that vector, and direct bus observations have the separate contract stated below.

Prefix-survivor invariant. Let C_0 be a finite nonempty candidate population, and suppose each processed flag is computed from the current population. Set

$$p_j = \bigvee_{i \in C_{j-1}} r_{ij}, \quad C_j = \{i \in C_{j-1} : r_{ij} = p_j\}.$$

Then C_j is nonempty and consists exactly of the channels in C_0 with lexicographically greatest processed request prefix. If $p_j = 0$, every candidate requests zero and survives. If $p_j = 1$, at least one requester survives, and exactly the candidates with the larger available next bit remain. Induction proves both assertions, retaining all ties.

Bridge fault model. Here $n \geq 2$. Choose a target t , a nonempty driver set S not containing t , and let T be the remaining channels. Write

$$\begin{aligned} U &= \neg x_t, & D &= \bigwedge_{i \in S} \neg x_i, & R &= \bigwedge_{i \in T} \neg x_i, \\ P_1 &= \neg(UDR), & P'_1 &= \neg(WDR). \end{aligned}$$

Here W is the value of the target term U as seen by the faulty first flag. The recovered fault interface forces $(U, D, W) = (1 - d, d, d)$ for a drive d in $\{0, 1\}$. The faulty copy runs the same stages from P'_1 and has qualifiers h'_i . The overwritten first flag is not a computed flag on the initial enabled population. In the separating drive-one cofactor, where $P_1 = 1$, $P'_1 = 0$ and every non-target first local request is zero, apply the prefix-survivor invariant to the suffix beginning at bus two, seeded by the enabled non-target channels; treat an empty seed separately. The final faulty qualifier set may additionally contain the target, so it need not equal the suffix candidate population.

Observation. Decoder functions $F_1, \dots, F_q$ read the qualifiers of each copy. The fault is *detected* by an input if $F_k(h) \neq F_k(h')$ for some k . The detection theorem observes functions of the qualifier vector. Direct observations of selector flags constitute an additional interface, handled separately in the compact implementation described in Appendix A.

Stuck-at model (SSA). The observed output is $y = \neg h_s \wedge \bigvee_{i \neq s} h_i$, with an empty OR equal to 0. The active-low request branch at channel t , bus j , is stuck at z only inside the final qualifier: that

qualifier uses $r'_{tj} = 1 - z$. Every other occurrence of the request, including its selector input, retains the healthy value.

4. Exact reachable sets

Section 3 gives the fault-free recurrence and the bridge drive. This section determines the qualifier sets those equations can produce, and gives a primary input for each set that occurs. Section 5 uses Corollary 4.2 for the detection reduction and Theorem 4.1 to realize a detecting set as a primary input. Theorem 4.4 separately classifies the specified stuck-at observation and is not an input to the bridge algorithm.

Theorem 4.1 (bridge faults). With drive 0, no input detects the fault. With drive 1, every input whose first flags differ has healthy eligible set $\{t\}$. As the input ranges over these drive-one inputs with differing first flags, the faulty eligible sets that occur are exactly

$$R_{n,b,t} = \begin{cases} \{K \subseteq [n] : t \in K\}, & b = 1, \\ \{K \subseteq [n] : K \neq \emptyset\}, & b \geq 2. \end{cases}$$

Proof.

Drive 0. $D = 0$ forces $P_1 = P'_1 = 1$. The identical tails then agree stage by stage.

Drive 1. $U = 0$ gives $x_t = 1$, and $D = 1$ gives $x_i = 0$ for every driver. If $R = 0$, both flags are 1 and nothing differs. So assume $R = 1$; then $P_1 = 1$, $P'_1 = 0$, and $x_i = 0$ for all $i \neq t$.

Healthy copy. The first mask keeps only t . The later flags equal t 's request suffix, so the eligible set is $\{t\}$.

Faulty copy. The first mask excludes t and keeps every other enabled channel.

- If there is no such channel, the later flags are 0 and the set is $\{t\}$.
- Otherwise let M be the lexicographically greatest request suffix among those channels. By the prefix-survivor invariant applied from bus two and seeded by these channels, the later flags equal M , and such a channel qualifies exactly when its suffix equals M : any other suffix covering M 's one-bits would be larger. The target qualifies exactly when its suffix covers M coordinatewise: $M_j = 1$ implies $r_{tj} = 1$ for every $j \geq 2$.

In both cases the set is nonempty, and it contains t when $b = 1$.

Realization. Let K be a set in $R_{n,b,t}$.

- If $t \in K$: enable exactly K , set $r_{t1} = 1$, and set all other requests to 0.
- If $t \notin K$ (so $b \geq 2$): enable exactly $K \cup \{t\}$, set $r_{t1} = 1$ as the only first request, set $r_{i2} = 1$ exactly for $i \in K$, and set every remaining request to zero.

Both inputs satisfy $U = 0$, $D = R = W = 1$ and produce the required pair of sets. $\square$

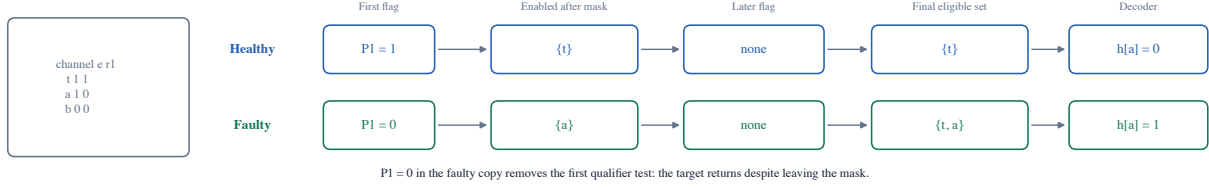
For example, with target t , two buses and a desired non-target singleton $\{i\}$, enable t and i , request bus one only at t , and request bus two only at i . Every other request is zero. The healthy qualifiers are $\{t\}$ and the faulty qualifiers are $\{i\}$. Additional buses can have all requests zero.

Figure 2 follows the two realization regimes through the intermediate mask and the final qualifier test. The target can qualify after leaving the faulty first mask; a second bus can instead exclude it from the final set.

How the bridge changes the reachable eligible set

Target t ; driver a ; drive 1. Within each panel, the healthy and faulty rows share the input.

(a) One bus: the target can qualify again



(b) Two buses: the target can be absent

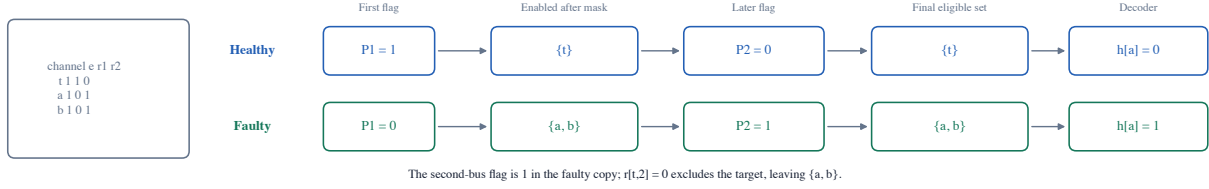


Image under differing first flags: one bus $\hat{a}^+$ target-containing sets; at least two buses $\hat{a}^+$ nonempty sets.

Theorem 4.1. These are circuit-evaluation stages; the final qualifier set is distinct from the intermediate mask.

Figure 2. Two drive-one separating inputs with target t , driver a , and three channels. Within each panel both copies share the displayed primary input. In (a), the faulty first mask removes the target, but the final qualifier test admits it again, producing $\{t, a\}$. In (b), the second bus excludes the target and produces $\{a, b\}$. The decoder h_a distinguishes the copies in both cases. Arrows follow circuit evaluation; the intermediate mask and the final eligible set are distinct objects.

Corollary 4.2. Detection holds if and only if $d = 1$ and there exist $K \in R_{n,b,t}$ and k with $F_k(\mathbf{1}_K) \neq F_k(\mathbf{1}_{\{t\}})$.

For fixed $n \geq 2$, target, drive and qualifier-only decoder functions, and independent unrestricted primary inputs, detection is independent of the choice of nonempty driver set disjoint from the target. All bus counts at least two give the same existence answer. Every detecting eligible set has a realization using requests on at most the first two buses. These claims follow directly from the image and canonical realizations in Theorem 4.1. The bus threshold is strict: with drive one, $F(h) = h_t$ cannot detect with one bus but detects a non-target singleton with two buses. These existence statements do not equate input distributions, native-model counts or detection under additional request constraints.

Lemma 4.3 (the nonemptiness constraint is global). Let $R_n = \{h : \bigvee_i h_i = 1\}$. Every projection of R_n onto a proper subset of coordinates is the full cube. Hence any conjunction of constraints on proper coordinate subsets that is valid on R_n admits the unreachable vector $h = 0$.

For $b \geq 2$ and the decoder $F = \bigvee_i h_i$, such a relaxation wrongly reports a detecting state, while the exact one-clause constraint shows the problem is UNSAT.

Proof. A partial row containing a 1 extends with zeros. An all-zero partial row extends by setting one outside coordinate to 1. $\square$

Theorem 4.4 (stuck-at qualifier faults). For $n = 1$ the fault is undetectable. For $n \geq 2$:

Stuck value of the active-low branch	Fault channel	Detectable?
0 (asserts a request)	$t \neq s$	No
0	$t = s$	Yes
1 (suppresses a request)	any t	Yes

Proof.

The “No” row. Asserting a request can only raise h_t from 0 to 1, and only when $P_j = 1$ while the healthy $r_{tj} = 0$. A channel $k \neq t$ with request vector equal to the flags is then already eligible.

- If $h_s = 1$, the output is 0 both before and after the fault.
- Otherwise $k \neq s$, so the OR over ordinary channels is already 1.

The “Yes” rows. In each case, let $u \neq t$, enable only the named channels, and set all unlisted requests to 0. Write ε_j for the request vector with only bus j set.

- Stuck-0 at $t = s$: give t the zero vector and u the vector ε_j .
- Stuck-1 at $t \neq s$: enable only t , with vector ε_j .
- Stuck-1 at $t = s$: give both t and u the vector ε_j .

Each input activates the fault and changes the output. $\square$

4.1. Bridges at a specified selector stage

The image argument also applies to a stage-local version of the bridge interface. Fix j with $1 \leq j \leq b$. Both copies compute the same prefix through bus $j - 1$. At bus j , replace x_i in the bridge equations of Section 3 by the actual local term t_{ij} (with $t_{i1} = x_i$), impose the same drive triple $(U, D, W) = (1 - d, d, d)$, and continue the faulty recurrence from the overwritten flag. This defines a Boolean fault interface on those local terms; primary inputs remain unrestricted.

Proposition 4.5 (stage-local bridge image). For this interface, drive zero is undetectable. In the separating drive-one case the healthy eligible set is $\{t\}$. The faulty image is all sets containing t when $j = b$, and all nonempty sets when $j < b$. Every admitted set has a primary-input realization with requests on at most buses j and $j + 1$.

Proof. Drive zero makes the two flags at bus j equal; their prefixes and all later computations therefore agree. With drive one, unequal flags require a true local request only at t . Let C be the healthy candidate population immediately before bus j . It contains t ; its other members have request zero at this bus. The healthy stage retains only t . In the faulty copy the zero flag retains the mask population $C \setminus \{t\}$. Channels outside C fail an earlier qualifier factor in both copies and cannot reappear.

If $j = b$, there is no further qualifier test and the final faulty set is C , which contains t . Otherwise run the prefix-survivor invariant of Section 3 on the suffix with initial population $C \setminus \{t\}$. If this population is empty, all suffix flags are zero and t qualifies. If it is nonempty, its lexicographically maximal suffix channels qualify, and t qualifies exactly when it covers all suffix flag one-bits. The faulty set is thus nonempty in every case.

For converse realizability, set every request before bus j to zero. For a set containing t , enable that set and make t its only requester at bus j . For a nonempty set excluding t , enable it together with t , make t the only requester at bus j , and make exactly the requested set request bus $j + 1$. Set every other request to zero. These are the canonical realizations of Theorem 4.1 shifted to the specified stage. $\square$

The supplied-decoder decision rules consequently apply with the terminal versus nonterminal stage distinction in place of the one-bus versus multiple-bus distinction. The image statement ranges over all primary inputs; fixing a prefix can restrict the candidate population. Native recovery and proof export for this additional interface are separate from the first-stage encoding contract of Appendix A.

4.2. The pair image of a final-qualifier stuck-at fault

The fixed observation in Theorem 4.4 is one instance of a more general pair description. Keep its fault location and active-low convention, but allow supplied functions of the qualifier vector as observations.

Theorem 4.6 (exact stuck-at pair image). Fix channel t and bus j . The differing healthy/faulty eligible-set pairs are exactly

- $(K \cup \{t\}, K)$ for stuck value $z = 1$, for every K excluding t ;
- $(K, K \cup \{t\})$ for stuck value $z = 0$, for every nonempty K excluding t .

Each pair has a realization using requests only on bus j .

Proof. The fault changes only the occurrence of r_{tj} inside the final qualifier. All selector flags and every other qualifier remain healthy. Suppressing that occurrence can only remove t ; asserting it can only add t . In the latter case activation requires $e_t = 1$, so the healthy enabled population is nonempty and has a maximal request vector. Its eligible set is therefore nonempty. This proves necessity.

For $z = 1$, enable exactly $K \cup \{t\}$ and give every enabled channel a request on bus j . The healthy set is $K \cup \{t\}$, and suppressing the target request in its final qualifier removes t . This includes empty K . For $z = 0$, enable exactly $K \cup \{t\}$, give every member of K a request on bus j , and give t no request. The healthy set is K ; asserting the target occurrence adds t . In both constructions every other request is zero. $\square$

Write $F_0(K) = F(\mathbf{1}_K)$ and $F_1(K) = F(\mathbf{1}_{K \cup \{t\}})$. Detection asks whether these values differ for a common K , with K nonempty for $z = 0$. This simultaneous comparison replaces the fixed healthy reference of Corollary 4.2. Section 7.1 identifies its complexity for guarded decoders.

5. Detection, P-completeness and constructive witnesses

Corollary 4.2 reduces bridge detection to a comparison: the supplied decoder must differ on some reachable faulty qualifier set from its value on the target singleton. This section solves that comparison for one decoder syntax, then shows that the resulting polynomial problem is P-complete. The witness it returns is consumed by the model lifting in Section 8.

A *guarded decoder* is an acyclic graph generated by

$$f ::= 0 \mid h_i \mid \bigvee_j f_j \mid \left(f \wedge \bigwedge_{j \in B} \neg h_j \right).$$

Subgraphs may be shared within an output cone or between output cones, and guards may overlap. Different output bits may also use disjoint graphs.

Algorithm 1 (DECIDE).

Input: guarded decoders $F_1, \dots, F_q$ over n channels, bus count b , target t , and drive d . **Output:** SAT with a detecting eligible set K , or UNSAT.

Line	Operation
1	If $d = 0$, return UNSAT.
2	Set $c_k \leftarrow F_k(\mathbf{1}_{\{t\}})$ for every k .
3	For each channel $i \neq t$, unless every $c_k = 1$:
4	$K \leftarrow \{i\}$ if $b \geq 2$, and $K \leftarrow \{i, t\}$ if $b = 1$.
5	If some k has $c_k = 0$ and $F_k(\mathbf{1}_K) = 1$:
6	Return SAT(K).
7	For each k with $c_k = 1$:
8	Form the dual-Horn system Z_k with the following clauses.
9	$\neg a_u \vee a_v$, for each child u of an OR node v .
10	$\neg a_u \vee \bigvee_{j \in B} h_j \vee a_v$, for a guard node v with child u and guards B .
11	$\neg a_{\text{root}}, \bigvee_i h_i$, and h_t when $b = 1$.
12	At leaves use $a_v = h_i$; at a constant-zero node use $\neg a_v$.
13	$M \leftarrow \text{GREATEST_MODEL}(Z_k)$, by linear-time downward propagation.
14	If M exists, return SAT($\{i : M(h_i) = 1\}$).
15	Return UNSAT.

The returned eligible set K is realized as a primary-input test by Theorem 4.1. This subsequent step writes $nb + n$ primary values; lifting them to a full native-CNF model takes a topological pass. Theorem 5.1 charges the additional output cost separately from the decoder kernel.

Theorem 5.1. Algorithm 1 correctly decides detection and constructs a detecting eligible set in $O((n + q)G)$ time and $O(G)$ working space, where G counts decoder nodes, child, guard and output incidences, and all n channel inputs. Writing the primary test input or a full native-CNF model takes total time $O(N + (n + q)G)$ and space $O(N + G)$, where N is the native encoded size.

Proof. By Corollary 4.2 it suffices to decide, for each bit k , whether some reachable K has $F_k(K) \neq c_k$.

Bits with $c_k = 0$. If $F_k(K) = 1$, follow a true child at every OR and the live child through every guard. The path ends at a leaf h_i whose guards all lie outside K . The terminal cannot be the target: otherwise the same path would already make the healthy target singleton true. Then the set $\{i\}$, or $\{i, t\}$ when $b = 1$, is reachable and keeps the path true. Lines 3–6 test exactly these at most n sets, and each evaluation serves all bits at once.

Bits with $c_k = 1$. Projected to h , the models of Z_k are exactly the reachable h with $F_k(h) = 0$:

- Given such an h , setting every a_v to the node’s value satisfies Z_k .
- Conversely, if evaluation makes a node true, its activity variable is forced true by induction. A true root contradicts $\neg a_{\text{root}}$.

Each clause of Z_k has at most one negative literal, so Z_k is dual-Horn. The greatest model therefore exists exactly when Z_k is satisfiable, and it is computed in time linear in G [DG84].

Cost. At most n evaluations of the graph, plus at most q fixed points. Reuse their working arrays. Realization writes $nb + n$ primary values; a topological pass lifts these to all native gate variables in $O(N)$ time. $\square$

Proposition 5.2 (dual-Horn transfer). For any dual-Horn CNF H on $x_1, \dots, x_m$, including unit and empty clauses, a one-bus controller with channels $t, s, 1, \dots, m$, driver $\{s\}$, and guarded decoder of size $O(m + |H|)$ has detecting eligible sets exactly $\{t, s\} \cup \{i : x_i = 1\}$ for x satisfying H . Here $|H|$ includes clause and literal counts.

Proof. For $C = \neg x_i \vee \bigvee_{j \in A} x_j$, use the violation term $V_C = h_i \wedge \bigwedge_{j \in A} \neg h_j$. For a positive clause use $V_C = h_t \wedge \bigwedge_{j \in A} \neg h_j$. Set

$$F_H = (h_t \wedge \neg h_s) \vee \bigvee_{C \in H} V_C.$$

The healthy value is one. Every reachable faulty set has $h_t = 1$, on which $F_H = 0$ is exactly $h_s = 1 \wedge H$. An empty positive clause contributes h_t , making detection impossible. Theorem 4.1 realizes every permitted set, giving the claimed bijection. $\square$

For each healthy-one output, the greatest model in Algorithm 1 projects to the inclusion-wise greatest detecting set for that output. Such a greatest set exists because these detecting sets are closed under union: if $F(A) = F(B) = 0$ but $F(A \cup B) = 1$, a true path on $A \cup B$ has its head in A or B and no guard in $A \cup B$, so it is already true on the set containing its head. The greatest set maximizes every nonnegative linear weight on those sets. Minimum cardinality is a different task, as the graph construction below shows. The sparse implementation in Theorem 5.6 may return a smaller witness when an unused channel suffices; its contract is to find a witness, not the greatest one.

Proposition 5.3 (P-completeness). Detection on supplied guarded representations is P-complete under deterministic logspace many-one reductions, even for one bus, one driver and one healthy-one output that is a guarded-term formula with at most two negative guards per term. Here internal gates have fanout at most one; channel variables may occur more than once.

Proof. Membership follows from Theorem 5.1. Reduce the monotone circuit value problem [Gol77; GHR95, Theorem 6.2.2]. Give each circuit input and gate a variable and fix the input variables by units. For $z = x \wedge y$, impose $\neg z \vee x$ and $\neg z \vee y$; for $z = x \vee y$, impose $\neg z \vee x \vee y$. Finally require the output to be one. These clauses are dual-Horn. Every model is componentwise at most the actual circuit evaluation, by topological induction; the actual evaluation itself satisfies the gate constraints. Thus this system is satisfiable exactly when the given circuit outputs one.

Apply Proposition 5.2. Each violation term has at most two negative guards. Use private internal gates for each term and an OR tree; repeated channel leaves are permitted in a formula forest. Both transformations have linear output size and can be emitted using logarithmic counters and input scans. This proves logspace hardness. $\square$

This transfers classical P-completeness through the guarded representation. A uniform parallel algorithm with polylogarithmic time and polynomially many processors would imply $P = NC$; a deterministic logarithmic-space algorithm would imply $P = L$. The result excludes neither all parallel speedups nor arbitrary sublinear working space. It gives no superlinear sequential-time lower bound: a single healthy-one output takes one linear-time dual-Horn pass.

Theorem 5.4 (extension of eligibility assignments). For a drive-one controller with supplied guarded outputs, deciding whether an arbitrary partial assignment α to eligibility coordinates extends to a detecting eligible set, and constructing one, takes $O((n + q)G)$ time and $O(G)$ working words. Healthy output values are still evaluated at $\{t\}$.

Proof. For one bus impose $h_t = 1$, rejecting α if it fixes $h_t = 0$. Let T be the fixed-true channels. A true path of any healthy-zero output under an extension ends at a positive head p . Keeping T and that head, and clearing the other free channels, preserves the path. Thus evaluate T and $T \cup \{p\}$ for every unfixed head; each pass serves all healthy-zero outputs. A true guarded output guarantees nonemptiness, since the grammar evaluates to zero when all channels are false.

For a healthy-one output, append both positive and negative fixed units to its dual-Horn zero system, retaining the reachable-image constraints. Units of either sign are dual-Horn. The greatest model exists exactly when a reachable zero extends α . At most $n + 1$ graph evaluations and q fixed points give the bound, reusing working arrays. $\square$

This theorem constrains eligibility coordinates, not arbitrary physical enable or request assignments. Native input lifting adds $O(N)$.

Corollary 5.5 (polynomial-delay enumeration). All distinct detecting eligible sets can be enumerated without repetition with delay $O(n(n + q)G)$ and $O(G)$ working words, including initial search and termination. Emitting a canonical primary input or native model adds $O(N)$ time and output storage per result.

Proof. Traverse the binary prefix tree of eligibility assignments in lexicographic order and prune a prefix exactly when Theorem 5.4 rejects it. Every nonempty surviving subtree contains a result. Between successive results, at most $O(n)$ prefixes and rejected siblings are inspected. Store one mutable prefix and a linear stack. Results are streamed; the bound excludes storage of a retained output list. This enumerates eligible sets and their canonical realizations, not all native-CNF models. $\square$

Theorem 5.6 (formula forests). If the guarded outputs form a formula forest with no shared internal nodes, detection and an eligible-set witness take $O(G)$ time and working space. Channel leaves may repeat. Native model construction adds $O(N)$ time and space.

Proof. Evaluate the target singleton once. At a healthy-zero root, a depth-first traversal keeps a counter for every guard on the current root-to-leaf path. A leaf h_i supplies a witness exactly when i is not blocked, and, for one bus, t is not blocked. Increment and decrement the counters on entering and leaving each guard. Each incidence is visited a constant number of times; the witness is $\{i\}$ or $\{i, t\}$.

At a healthy-one root, build the dual-Horn system on its own cone. For one bus, the target unit already ensures nonemptiness. For at least two buses, mark the channel indices occurring as leaves or guards. Every expression in the guarded grammar is zero when all its channel inputs are zero, by structural induction. If a channel is unmarked, its singleton therefore makes this entire formula zero. The first unmarked index is found after scanning at most one more index than the number marked. Otherwise all n indices occur in the cone, so the global nonemptiness clause costs at most the cone size. Use sparse maps or a single timestamp array to avoid n initialization steps per output. Since internal cones are disjoint and repeated leaves are counted as incidences, the total fixed-point work is $O(G)$. $\square$

6. Witnesses, representations and width

Section 5 decides whether any detecting qualifier set exists. This section keeps that decision procedure and changes the question. Minimum cardinality, counting, and a compact representation of every detecting input can be hard for the same decoders. The width bounds exhibit tractable instances of unbounded treewidth and cut-width, so the fixed-width conditions cited in Section 1 do not cover the whole family. Section 7 uses the same sets for a different extension, the gate basis.

6.1. Why explicit syntax matters

Example 6.1 (why a simultaneous set is needed). Take the cycle decoder $F = \bigvee_i (h_i \wedge \neg h_{i+1 \bmod n})$. Every proper nonempty set gives output 1. The only nonempty zero set is all n channels. The fixed point finds it in one linear pass. Tests restricted to sets of cardinality below n cannot detect

this fault. Since every proper nonempty set gives the same output as the healthy set, adding channels one at a time changes the output only at the last step.

Proposition 6.2 (exponentially many critical sets). For every $K \in R_{n,b,t}$ with $K \neq \{t\}$ there is an $O(n)$ -node guarded decoder F_K whose detecting inputs produce faulty eligible set exactly K . Take a cyclic successor map s on K and set

$$F_K(h) = \bigvee_{i \in K} (h_i \wedge \neg h_{s(i)}) \vee \bigvee_{j \notin K} h_j.$$

For a singleton K , omit its identically false cycle term.

Consequently, a list of test inputs that must work for every such decoder without reading it needs at least $2^{n-1} - 1$ entries ($b = 1$) or $2^n - 2$ entries ($b \geq 2$). Algorithm 1 reads the decoder and recovers K in $O(n)$.

Proof. The unconditional terms exclude every channel outside K . Inside K , a zero cycle term requires each selected channel's successor to be selected, and the only nonempty successor-closed subset is K . Each test input produces a single eligible set, so it detects at most one F_K . $\square$

Corollary 6.3 (adaptive query bound). Suppose a decoder is accessible only through queries of its value on eligible sets. On the promise class consisting of the decoders F_K of Proposition 6.2 and the undetectable decoder $F_\perp = \bigvee_i h_i$, deterministic detection needs at least M queries in the worst case, where $M = 2^{n-1} - 1$ for one bus and $M = 2^n - 2$ otherwise. Randomized detection with error at most $1/3$ on every input needs $\Omega(M)$ queries in the worst case.

Proof. Over nonempty sets, F_K differs from $F_\perp$ only at K . All these decoders are zero on the empty set. On the answer transcript for $F_\perp$, an unqueried K remains indistinguishable, proving the deterministic bound even for adaptive queries. Put probability $1/2$ on $F_\perp$ and distribute the rest uniformly over the M critical decoders. Every deterministic decision tree making at most r queries has success probability at most $1/2 + r/(2M)$: until it queries the hidden K , it sees the $F_\perp$ transcript. Averaging over random coins gives $r \geq M/3$ for error at most $1/3$. This access model excludes inspection of the syntax. $\square$

6.2. Optimization, counting and compilation

Proposition 6.4 (graph-cover representation). Let $J = (V, E)$ be a finite simple graph with at least one edge. Take one bus, channels $V \cup \{t\}$, any vertex as the single driver, and the guarded decoder

$$F_J = \bigvee_{uv \in E} (h_t \wedge \neg h_u \wedge \neg h_v).$$

The detecting eligible sets are exactly $\{t\} \cup C$ for vertex covers C of J . The construction has size $O(|V| + |E|)$.

Proof. The healthy value is one. On reachable sets $h_t = 1$, and $F_J = 0$ is precisely $\bigwedge_{uv \in E} (h_u \vee h_v)$. Theorem 4.1 realizes every such set. Each term has one live child and two negative guards, so it belongs to the original grammar. Give each edge term its own two-guard chain and combine the terms with a private OR tree. No internal decoder node is shared; repeated channel leaves are permitted. This gives a formula-forest decoder of the same asymptotic size. $\square$

Figure 3 illustrates Proposition 6.4 on a triangle. Each uncovered edge contributes a true guarded term; covering every edge makes the output zero.

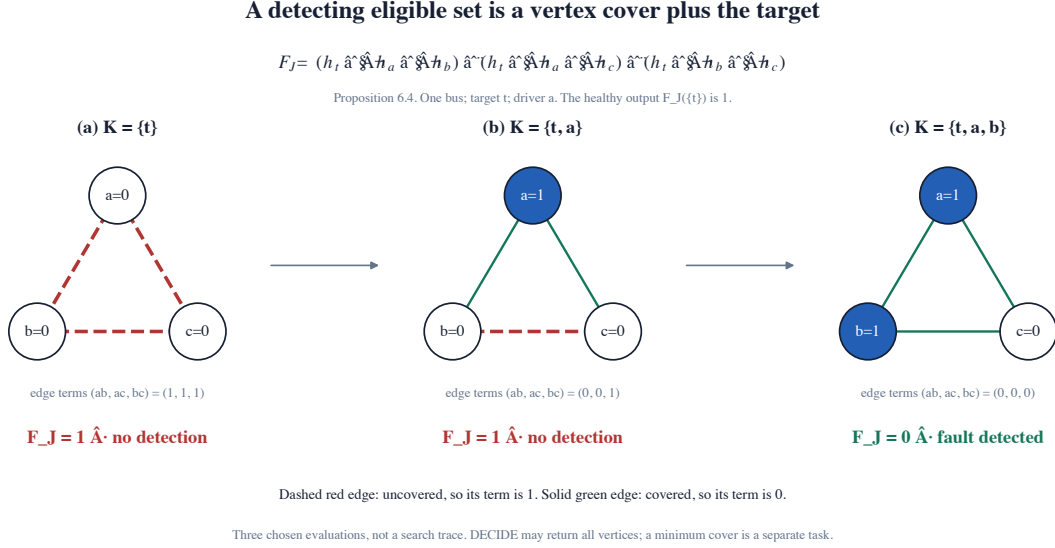


Figure 3. One bus, target t , driver a , and triangle graph J . The healthy output is one. For $K = \{t\}$, $\{t, a\}$, and $\{t, a, b\}$, the faulty outputs are respectively one, one, and zero. Dashed edges are uncovered and contribute true terms; solid edges are covered. Detection occurs precisely when $K = \{t\} \cup C$ for a vertex cover C . The arrows compare chosen evaluations, not an algorithmic search order. Choosing every graph vertex gives a detecting set; finding a smallest one is a separate task.

Corollary 6.5 (minimum witnesses and counting). Within the guarded class, already with one bus, one driver and one output, deciding whether there is a detecting set of size at most a supplied bound is NP-complete. Counting distinct detecting eligible sets is #P-complete under polynomial-time Turing reductions.

Proof. Proposition 6.4 gives the exact minimum $1 + \tau(J)$, so a vertex-cover bound k becomes $k + 1$. The same construction is a bijection for counting. Cardinality-bounded vertex cover is NP-complete [Kar72], and counting vertex covers is #P-complete [PB83], even for bipartite graphs in the latter case. Membership follows by guessing a set and checking reachability and decoder evaluation. The imported counting hardness uses counting reductions; the transfer from a particular graph to its controller preserves the count. $\square$

This counts eligible sets, equivalently their canonical inputs in Theorem 4.1. It does not count all native-CNF models. In the activated one-bus cofactor, a cover C has $2^{|V|-|C|}$ primary realizations because a disabled non-target channel's request is free. The complete native gate values are uniquely determined by the primary input and fault interface.

Proposition 6.6 (maximum witnesses and healthy-zero counting). For one bus, one driver and one healthy-zero guarded output, the threshold problem of finding a detecting eligible set of size at least a supplied bound is NP-complete. Counting such eligible sets is #P-complete under polynomial-time Turing reductions, even with one distinct positive head.

Proof. For a graph $J = (V, E)$, use channels $V \cup \{t, s\}$, target t and driver $\{s\}$. Start at $f_0 = h_s$. For each edge $e_j = \{u, v\}$, append

$$f_j = (f_{j-1} \wedge \neg h_u) \vee (f_{j-1} \wedge \neg h_v).$$

Store f_{j-1} once. Each new AND has only one live child, and the total size is $O(|V| + |E|)$. The resulting output is $h_s \wedge \bigwedge_{uv \in E} (\neg h_u \vee \neg h_v)$, with healthy value zero. Detecting eligible sets are exactly $\{t, s\} \cup I$ for independent sets I of J . Thus the maximum is $2 + \alpha(J)$. Complementation transfers the vertex-cover decision and counting hardness used in Corollary 6.5; membership follows by checking a proposed eligible set. $\square$

For a single output the cardinality classification is therefore:

Healthy value	Minimum detecting-set cardinality	Maximum detecting-set cardinality
0	Exactly 2 for one bus, 1 for at least two buses, when detectable	NP-complete threshold problem
1	NP-complete threshold problem	Polynomial: greatest zero model

For a detectable healthy-zero output, a true path remains true on its positive head, together with the target for one bus. This proves the upper bounds in the table. Nonemptiness gives the multi-bus lower bound; the only reachable one-bus singleton is the healthy target singleton, which does not detect. The minimum is undefined for an undetectable output. Cardinality here counts eligible channels, not primary-input Hamming weight.

Counting detecting eligible sets is #P-complete for either healthy value. The polynomial directions also optimize nonnegative channel weights: shrinking to a positive support cannot increase weight, and the greatest zero set maximizes it.

Corollary 6.7 (maximum weight for formula forests). For supplied healthy-zero formula-forest outputs, a maximum nonnegative-weight detecting set can be found with $O(G)$ arithmetic operations and $O(G)$ working words, apart from the encoded-weight and output costs.

Proof. Traverse each leaf-occurrence path with a counter per blocked channel, maintaining the weight of their union. A path ending at h_i is feasible if it blocks neither i nor the forced target, when present. Its best set contains every unblocked channel and has weight $W_{\text{total}} - W_{\text{blocked}}$. Each guard incidence is entered and left once; weight changes occur only on counter transitions to or from zero. Record the best leaf occurrence, then reconstruct its path once, avoiding a copy of the whole blocked set at every improvement. $\square$

Theorem 6.8 (full native-model counting). Counting satisfying assignments of the complete native guarded bridge CNF is #P-complete under polynomial-time Turing reductions, already on the one-bus, one-output graph-cover subfamily, without extra primary restrictions.

Proof. Let $m = |V|$ and assume J has an edge. Use the drive-one construction of Proposition 6.4. For a vertex cover C and an arbitrary bit vector ρ indexed by $V \setminus C$, set

$$e_t = r_{t1} = 1, \quad (e_v, r_{v1}) = \begin{cases} (1, 0), & v \in C, \\ (0, \rho_v), & v \notin C. \end{cases}$$

These inputs satisfy the drive interface and make $P_1 = 1, P'_1 = 0$. The healthy eligible set is $\{t\}$ and the faulty set is $\{t\} \cup C$, so the graph-cover output differs. Conversely, every detecting input has different first flags, since equal first flags propagate equal outputs through the identical tails. The

drive interface then forces the target enable and request to one and every non-target product $e_v r_{v1}$ to zero. Its enabled non-target set must be a cover, and its request values have exactly the displayed form. Thus (C, ρ) is in bijection with detecting primary inputs. Complete native gate definitions give each such input a unique extension to all internal variables (Section 2). Therefore the full count is

$$\# \text{Mod}(\Phi_J) = \sum_{C \text{ cover}} 2^{m-|C|} = \sum_{I \text{ independent}} 2^{|I|} = P_J(2), \quad P_J(z) = \sum_{I \text{ independent}} z^{|I|}.$$

Replace each vertex by r independent clones and each edge by all edges between its two clone groups, obtaining $J^{[r]}$. An independent set of $J^{[r]}$ is an independent set I of J together with a nonempty set of clones for each vertex of I . A nonempty occupied group has total weight $\sum_{\emptyset \neq A \subseteq [r]} 2^{|A|} = 3^r - 1$, so $P_{J^{[r]}}(2) = P_J(3^r - 1)$. Native counting queries for $r = 1, \dots, m+1$ give distinct evaluations of the degree-at-most- m polynomial. Exact interpolation recovers $P_J(1)$, the number of independent sets and covers. Edgeless source graphs are counted directly. Query graphs have $O(m^2)$ vertices and $O(m^2|E|)$ edges, and values and rational interpolation intermediates have polynomial bit length. Membership in $\#P$ follows by guessing a complete CNF assignment and checking its clauses. $\square$

Eligible-set counts and native-model counts differ; this reduction proves hardness despite their different multiplicities. It is a Turing reduction, not a claim of a parsimonious native-model correspondence.

Theorem 6.9 (DNNF and OBDD size). There is an infinite one-bus, one-output subfamily of the guarded native encoding with $N = \Theta(n)$ and formula-forest decoders, each term having one positive head and at most two negative guards, for which the primary detection function and the full native CNF each require DNNF size $2^{\Omega(N)}$. Repeated channel leaves are allowed. Every variable order also requires exponentially many OBDD nodes.

Proof. Use the bounded-degree vertex-expander graphs of [BCMS15, Theorems 4–5]. Their graph CNFs $C_J(x) = \bigwedge_{uv \in E} (x_u \vee x_v)$ require $2^{\Omega(|V|)}$ DNNF size. In the controller of Proposition 6.4 fix

$$e_t = r_{t1} = 1, \quad r_{v1} = 0 \quad (v \in V).$$

and leave e_v free. These settings satisfy the drive-one interface and give $P_1 = 1$, $P'_1 = 0$, healthy set $\{t\}$, and faulty set $\{t\} \cup \{v : e_v = 1\}$. The detection cofactor is exactly $C_J(e)$. Conditioning cannot enlarge a DNNF. For the full native CNF, condition in the same way and existentially forget gate and observation variables. Forgetting preserves decomposability and does not increase size: it distributes over OR, and over AND because the children's variable sets are disjoint. The result again represents C_J . This proves both lower bounds. An OBDD has a linear-size translation to DNNF, so the OBDD bound follows for every order. Use the private term chains and OR tree of Proposition 6.4. Bounded degree preserves linear size, and the same conditioning and projection apply. The one-bus circuit and complete bounded-fan-in encoding have size $\Theta(|V|) = \Theta(n)$. The forest restriction applies to the decoder, not to the complete physical circuit or its CNF interaction graph. $\square$

DNNF size counts wires; for bounded fan-in this agrees with gate count up to constant factors. The exponent uses the combinatorial encoded size N , not file bytes. Dense binary identifiers give bit length $L = \Theta(N \log N)$, and hence the corresponding bound is $2^{\Omega(L/\log L)}$. The expander lower bound is imported; the physical restriction and exact projection above transfer it to this family.

6.3. Circuit width and syntactic backdoors

Proposition 6.10 (treewidth). The primal and incidence graphs of the native encoding contain a $K_{n,b}$ minor, so their treewidth is at least $\min(n, b)$.

Proof. Work inside one copy of the uncontracted native layout. Write $g_{ij} = \neg P_j \vee r_{ij}$. For channel i , take the outputs of its private qualifier-AND tree together with all its factor variables g_{ij} as a branch set C_i ; the tree edges make this set connected. Take $D_j = \{P_j\}$ for each bus. These sets are pairwise disjoint. The complete OR block for g_{ij} contains $g_{ij} \vee P_j$, so the primal graph has an edge from C_i to D_j for every pair (i, j) .

In the incidence graph, connect each qualifier-tree output to its non-primary child through the private binary clause of that AND gate. Assign these clause vertices to C_i . Also assign the distinct clause vertex for $g_{ij} \vee P_j$ to C_i ; it joins g_{ij} to the bus branch D_j . Thus the incidence branch sets are disjoint and connected and have every required cross-edge. Explicit inverter wires, if present instead of signed inputs, and their connecting definition clauses can be absorbed into the corresponding bus branch. Contracting the branch sets gives $K_{n,b}$ in both graphs. The treewidth lower bound follows. This argument uses the complete native layout before simplification or elimination of native variables. $\square$

Proposition 6.11 (linear width obstruction). The subfamily in Theorem 6.9 has an explicit native gate layout with treewidth and net cut-width $\Omega(N)$. For every partition of its gates into blocks having at most K input signals, if the undirected block graph has treewidth w , then $K(w+1) = \Omega(N)$.

Proof. Orient the edges arbitrarily. For every oriented edge $v \rightarrow u$, create private gates $a_{vu} = h_t \wedge \neg h_v$ and $d_{vu} = a_{vu} \wedge \neg h_u$. A private binary OR tree joins the edge terms. No internal decoder gate is shared; channel leaves may repeat. For each v , let B_v contain its faulty qualifier h_v and all its outgoing gates a_{vu}, d_{vu} . Signed negative inputs require no additional inverter gates. These sets are disjoint and connected through h_v ; each connection from h_u to d_{vu} gives the required edge between B_u and B_v . Hence J is a minor. Each gate of B_v is reachable from the primary enable e_v along a path of signals belonging only to v , first through the channel's private qualifier tree and then through its private edge term. Each edge contributes two guards, so bounded degree preserves linear total size.

A bounded-degree vertex expander on m vertices has treewidth $\Omega(m)$. To see this, remove a balanced-separator bag of a tree decomposition. If the bag is small, components of size at most $m/2$ can be grouped into a set of size between $m/4$ and $m/2$; all its outside neighbors lie in the bag, so vertex expansion forces a linear-size bag. If the bag is already large, the conclusion holds directly. The minor therefore gives the circuit treewidth bound. Complete gate clauses also give the corresponding native primal-graph minor; binary output-to-input clauses supply the subdivided edges for an incidence-graph minor.

For net cut-width, retain the vertices in $\bigcup_v B_v$, and delete entire shared target and selector nets before restricting the remaining nets. Deleting just a shared net's producer would leave a large hyperedge. The retained two-section graph still has the J minor. If D bounds the degree of J , each retained net has at most $D+3$ vertices. Under every ordering, each vertex counted by vertex separation belongs to a crossing net; therefore pathwidth $\leq (D+3)$ cut-width [Kin92]. Net/vertex deletion does not increase cut-width, proving the linear bound for the full circuit.

Finally, a block with at most K inputs meets at most K of the B_v : follow the v -owned path from primary e_v to a gate in that block and take its first entering signal. Primary signals count as block inputs. Different v give distinct input signals. Replace every bag of a block-graph decomposition by the vertices whose B_v meets a block in that bag. The blocks meeting a connected branch set B_v form a connected subgraph of the block graph. Their bag subtrees have a connected union, because adjacent

blocks occur together in a bag. Thus the replacement bags satisfy the running-intersection property for v . Each edge of J has endpoints in the same block or in adjacent blocks, so a replacement bag covers it. The resulting decomposition of J has bags of size at most $K(w + 1)$. Thus $\text{tw}(J) + 1 \leq K(w + 1)$. Since $N = \Theta(m)$, all bounds follow. $\square$

These are bounds for the specified gate layout and encoding, not for all equivalent representations or for the running times of [CAB90, Fuj90, PCK99]. They exclude every fixed (K, w) and logarithmic cut-width bound while detection remains linear after role recovery on this single-output, healthy-one subfamily. The explicit grid comparison used by the timing panel is given in the structural companion [Sha26S].

The native CNF also contains a simple obstruction to small syntactic backdoors. For a CNF class C , a *strong backdoor* is a variable set B such that every assignment to B , followed only by deleting satisfied clauses and false assigned literals, leaves a formula in C . A *deletion backdoor* is a set whose literals can be removed from every clause to leave a formula in C . Here the base classes are purely syntactic: the presence of an empty clause does not excuse other clauses from their class conditions [DGS14, OSS24].

Proposition 6.12 (syntactic backdoors). Every strong or deletion backdoor of the native complete-gate encoding into Horn, renamable Horn, or 2-CNF has size at least b .

Proof. The complete encoding of $x \oplus y \oplus z = c$ consists of four ternary clauses. Any complementation of its three variables merely switches between two parities of clause-sign patterns. One contains the all-positive clause; the other contains three clauses with two positive literals. Thus the block is never renamable Horn, and it is not 2-CNF.

Fix one healthy channel and take its mask XOR at each bus. The three variables in each block are the bus flag, its local complemented contribution, and a fresh mask output. In the native encoding these b triples are pairwise disjoint. If a backdoor B misses one triple, all four clauses of that block survive unchanged under every assignment to B or deletion of the B -literals. No renaming of the whole residual formula can make this subformula Horn. A backdoor must therefore hit every triple, requiring at least b variables. The Horn and 2-CNF claims follow from the same obstruction. $\square$

This comparison concerns the complete blocks retained by the native encoding. Taking $n = b$ makes both the treewidth and these backdoor sizes unbounded while the family remains polynomially decidable. Propagation, algebraic rewriting, and richer backdoor measures can exploit other constraints; their running times are not bounded below by this argument.

7. Gate-basis and live-conjunction boundaries

The algorithm in Section 5 allows disjunction and conjunction of one computed child with negative channel guards. This section classifies detection when a conjunction may also have two computed children. The original guarded syntax remains the polynomial case already proved; the lower bounds below apply only after those conjunctions are added.

Theorem 7.1 (NP-completeness). If the grammar allows an unbounded number of binary ANDs with two live children, detection is NP-complete, even with $b = 1$, a one-channel driver set, and one output bit.

Proof. Reduce from 3-SAT. Use channels t , a marker s , and $z_1, \dots, z_k$, with s as the single driver. The marker makes the healthy value zero, because s is not eligible in the healthy set $\{t\}$.

- Translate a positive literal z_j as h_{z_j} .

- Translate $\neg z_j$ as the guarded term $h_t \wedge \neg h_{z_j}$.
- Join the literals of each clause with OR, and the clauses with the new ANDs, giving Φ . Observe $F = h_s \wedge \Phi$.

The healthy value is $F(\{t\}) = 0$. Every reachable faulty set contains t , so each translated literal takes its literal's value. A detecting set therefore exists exactly when ϕ is satisfiable, as witnessed by $\{t, s\} \cup \{\text{true variables}\}$. Membership in NP is immediate. $\square$

Figure 4 shows the reduction on three clauses. Changing z from zero to one satisfies the second clause while preserving the other two, and the added live-AND gates propagate this change to the output.

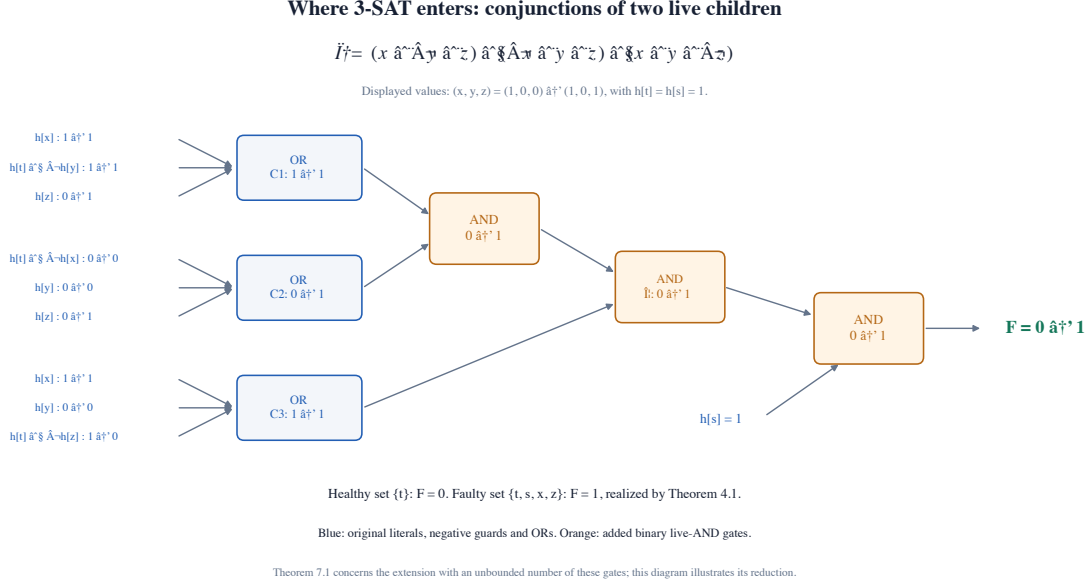


Figure 4. Translation of $(x \vee \neg y \vee z) \wedge (\neg x \vee y \vee z) \wedge (x \vee y \vee \neg z)$ from Theorem 7.1, with one bus and drive one. Blue operations belong to the original guarded grammar; orange gates conjoin two live children. Each annotation compares $K_0 = \{t, s, x\}$ with $K_1 = \{t, s, x, z\}$. The healthy set $\{t\}$ has output zero; K_1 has output one and detects the fault. The NP-completeness theorem concerns the extension allowing an unbounded number of the orange gates.

Lemma 7.2 (bounded positive support). Extend the guarded grammar by k distinct binary live-AND nodes in the whole DAG. If an output is true on an eligible set K , it is true on some subset of K of size at most $k + 1$. If K contains the target, adjoining the target to that subset preserves truth and gives a reachable set of size at most $k + 2$.

Proof. In the true subgraph rooted at the output, choose one true child at each OR, the live child at each guard, and both children at each AND. Choose the same child whenever a shared OR is encountered. Every internal node has outdegree one, except at most k nodes of outdegree two. If there are I internal nodes and l distinct leaves, connectedness gives $I + l - 1 \leq \text{number of edges} \leq I + k$, hence $l \leq k + 1$. A true leaf is a channel. Retain just these channels from K : all chosen positive leaves stay true and every chosen negative guard remains satisfied. Adjoining t is safe when t was already in K . $\square$

Lemma 7.3 (guarded expansion). At each binary live AND of a decoder F , select one of its two children and replace the node by that child. Writing F_σ for the resulting guarded DAG, with one

consistent choice per shared node, one has $F = \bigwedge_{\sigma} F_{\sigma}$ pointwise. At most 2^k choices are needed, where k counts the live ANDs in its cone.

Proof. All surrounding operations are monotone in their live children, so removing either child of a conjunction gives $F \leq F_{\sigma}$. For a fixed channel assignment, choose a false child at every false AND and either child at every true AND, using values in the original DAG. Topological induction shows that this choice preserves every node value. Thus some F_{σ} equals F on that assignment, proving the identity. $\square$

Theorem 7.4 (bounded live conjunction). Put $B_k(n) = \sum_{j=0}^{\min(k+1, n)} \binom{n}{j}$. For supplied decoder DAGs with at most k additional binary live ANDs, detection and an eligible-set witness can be found in

$$O((B_k(n) + q2^k)G) \text{ time,} \quad O(G) \text{ working space.}$$

Writing and lifting a native model adds $O(N)$ time and space. For each fixed k this is polynomial; for $k = 1$ the time is $O((n^2 + q)G + N)$ including lifting. The parameterized bound is XP in k . The branch for healthy-one outputs alone has time $O(q2^k G)$ and is FPT in k .

Proof. For healthy-zero outputs, enumerate the nonempty supports of size at most $k + 1$. For one bus, adjoin t , including the empty support as a candidate. Each evaluation checks all relevant outputs simultaneously. Lemma 7.2 proves completeness. For a healthy-one output, Lemma 7.3 says that some reachable zero of F exists exactly when some guarded expansion has a reachable zero. Test these expansions with Algorithm 1's dual-Horn branch. Generate one support or expansion at a time and reuse the working arrays. The model realization of Theorem 4.1 is unchanged. $\square$

Thus one extra live AND does not yield the hardness of Theorem 7.1. The extension is an algorithm for a supplied decoder representation; Appendix A's native recognition guarantee concerns the original guarded grammar.

Theorem 7.5 (typed gate-basis dichotomy). Let B be a fixed subset of the following three operations: OR of live children, binary AND of two live children, and a negative channel guard $f \wedge \neg h_i$. Let $D(B)$ contain all finite acyclic decoder DAGs made from zero and positive channel leaves using operations in B , with unrestricted sharing and repeated input occurrences. Detection on supplied $D(B)$ representations is polynomial for every proper subset B , and NP-complete when all three operations are available. The hard case already has one bus, one driver and one output.

Proof. Multi-channel guards are chains of single-channel guards, so their admission makes no difference to the classification.

- **No live AND.** This is the guarded class of Theorem 5.1.
- **No negative guard.** Every output is monotone. A healthy-zero output has a reachable one exactly when its value on the full channel set is one. A healthy-one output has a reachable zero exactly when it is zero on a minimal reachable set: $\{t\}$ for one bus, or one of the singletons for at least two buses. Evaluate these at most $n + 1$ sets for all outputs in $O(nG)$ time.
- **No OR.** Every output is zero or one conjunction of positive and negative channel literals. Compute these literal sets by topological union, marking a node zero if a child is zero or both signs of a channel occur. This takes $O(nG)$ time and space. To find a reachable one, retain all positive literals and the forced target and reject any negative conflict. If no channel has been selected, add one channel that does not occur negatively; if every channel does, no reachable

one exists. To find a reachable zero, falsify one conjunct while obeying the target/nonemptiness condition. These tests are polynomial for all outputs.

Every proper subset omits an operation and is covered by one of these cases. Theorem 7.1 supplies NP-hardness for the full basis; evaluating a guessed reachable set proves membership in NP. $\square$

The negative input of a guard is a channel literal, not an arbitrary subdecoder. The theorem classifies eight syntax languages. It does not classify every further graph restriction of the full language, or assign hardness to individual inputs containing all three gate types.

Theorem 7.6 (live-AND parameter hardness). Detection for supplied decoders with k binary live ANDs is $W[1]$ -hard parameterized by k , even with one healthy-zero output, one bus and one driver. This holds when every internal gate has fanout at most one, every guard has one negative channel literal, and the live ANDs form a binary tree joining guarded subformulas.

Proof. Let a graph $J = (V, E)$ have a partition into $r \geq 2$ nonempty color classes $V_1, \dots, V_r$. Multicolored Clique asks for a clique containing exactly one vertex of each class. Use channels $V \cup \{t\}$ and one bus. For $v \in V_i$, set

$$\begin{aligned} B_v &= (V_i \setminus \{v\}) \cup \{u \in V \setminus V_i : uv \notin E\}, \\ T_v &= h_v \wedge \bigwedge_{u \in B_v} \neg h_u, \\ A_i &= \bigvee_{v \in V_i} T_v, \\ F_J &= \bigwedge_{i=1}^r A_i. \end{aligned}$$

Each T_v is guarded; binary OR trees build the A_i . A binary tree joining their roots uses exactly $k = r - 1$ live ANDs. Use separate internal gates for each term, so internal fanout is at most one. Negative guards can be implemented one literal at a time. The decoder has size $O(|V|^2)$.

The healthy value $F_J(\{t\})$ is zero. Suppose $F_J(K) = 1$. For each class choose a true term T_v . Its same-class guards exclude every other vertex of that class from K ; its other guards exclude all its nonneighbors. Consequently $K \setminus \{t\}$ contains exactly one vertex per class and these vertices form a clique. Conversely, every such clique C makes all its chosen terms true on $\{t\} \cup C$. By Theorem 4.1 these sets are reachable. Thus the detecting eligible sets are in bijection with multicolored cliques. Any vertex may serve as the single driver, whose enable can be one while its first request is zero.

For completeness, ordinary r -Clique reduces to this colored problem by making r copies of the input vertex set. Vertices in distinct copies are adjacent exactly when their original vertices are adjacent. The graph is simple, so a colored clique cannot reuse an original vertex. The construction preserves r , and $k = r - 1$ is a parameter-preserving reduction from the $W[1]$ -complete Clique problem. $\square$

In this construction every detecting set has exactly $k + 2$ channels including the target. Every true set needs $k + 1$ vertex channels. Thus the positive-support and target-adjointing bounds of Lemma 7.2 are attained. In particular, internal gate sharing and guards with multiple negative inputs are unnecessary for the parameterized hardness. Repeated channel occurrences across different terms remain essential to this construction.

Corollary 7.7 (conditional exponent lower bound). Unless $FPT = W[1]$, the full problem of Theorem 7.4 has no FPT algorithm in k . Under ETH it has no algorithm of time $f(k)G^{o(k)}$ for any

computable f . The latter bound also holds with the size N of a complete native controller encoding in place of G , even when its circuit roles are supplied.

Proof. The first assertion follows from Theorem 7.6. For an r -Clique instance on p vertices, its colored graph has rp vertices and the decoder has size $G = O(r^2p^2)$. With one bus the complete native encoding also has $N = O(r^2p^2)$. A hypothetical algorithm of either stated form, including the polynomial construction cost, would solve Clique in $f'(r)p^{o(r)}$ time. Under ETH, not every SNP problem admits a subexponential algorithm, so [CHKX06, Theorem 5.5], with the polynomial encoding-length conversion stated in Section 2, excludes this Clique algorithm. $\square$

The corollary rules out a sublinear dependence on k in the polynomial exponent under ETH. It does not establish the optimal constant in that exponent. Its lower bound concerns the live-AND extension; it supplies no quadratic or cubic lower bound for decision in the original guarded class. Fine-grained bounds for the original guarded class are treated in the guarded-reachability companion [Sha26B] under their own named hypotheses.

7.1. A fault-model boundary inside the guarded grammar

The preceding results vary the decoder gates while keeping the first-stage bridge interface fixed. The exact pair image in Theorem 4.6 permits a second comparison: retain the guarded grammar and change the fault to the specified final-qualifier occurrence. Here a witness must make one cofactor true and the other false on the same non-target channel set.

Theorem 7.8 (stuck-at detection on guarded DAGs). For either fixed stuck value, detection of a final-qualifier stuck-at fault by a supplied guarded DAG is NP-complete, even with one bus and one observed output. No conjunction of two computed children is required.

Proof. Membership in NP follows by guessing a primary input and evaluating the controller and decoder. For hardness, let ϕ be a 3-SAT formula on variables $x_1, \dots, x_m$. Introduce channels t, s and pairs p_i, q_i . Replace each positive source literal by p_i and each negative literal by q_i , and append all positive pair clauses $p_i \vee q_i$. Call the resulting clause list C . Construct

$$\begin{aligned} A &= \bigvee_{C \text{ in the list}} \left(h_s \wedge \bigwedge_{u \in C} \neg h_u \right), \\ B_0 &= h_s, \\ B_i &= (B_{i-1} \wedge \neg h_{p_i}) \vee (B_{i-1} \wedge \neg h_{q_i}), \\ F &= A \vee (B_m \wedge \neg h_t). \end{aligned}$$

The two uses of B_{i-1} share a node. Every gate is allowed by the guarded grammar, and the size is linear in the source variable and literal counts. Induction gives $B_m = h_s \wedge \bigwedge_i (\neg h_{p_i} \vee \neg h_{q_i})$. The two target cofactors are $F_0 = A \vee B_m$ and $F_1 = A$. They differ precisely when $B_m = 1$ and $A = 0$.

The first condition requires s and forbids selecting both channels of any pair. Given s , the second requires at least one of each pair and at least one translated literal of every source clause. Thus the detecting sets choose exactly one channel per source variable and encode precisely the satisfying assignments of ϕ , with s included. Conversely each satisfying assignment gives such a set. It is nonempty, so Theorem 4.6 realizes the discrepancy with one bus for either stuck value. $\square$

The reduction uses the target only in one negative guard. Its source formula is encoded in the simultaneous truth and falsity conditions; either condition alone admits the guarded reasoning used in Section 5.

Theorem 7.9 (path-parameter algorithm and unshared formulas). For a single supplied guarded output with P structural root-to-leaf paths, final-qualifier stuck-at detection has an exact algorithm taking $O((P + 1)G)$ time and $O(G)$ working words. For an output formula forest with no shared internal gates and total counted size G , detection takes $O(G^2)$ time and $O(G)$ working words. Writing a physical test adds $O(nb)$ time; constructing a complete native model adds $O(N)$.

Proof. Search the two directions $F_v(K) = 1$, $F_{1-v}(K) = 0$, for $v = 0, 1$. Build the one-way dual-Horn zero encoding of Algorithm 1 (lines 8–12) for the opposite copy, require its root false, and fix its target to $1 - v$. For stuck value zero add the nonemptiness clause $\bigvee_{i \neq t} h_i$.

Every true evaluation of the first copy has a supporting path: choose one true OR child and retain the live child at each guard. Enumerate such structural paths with the target fixed to v . A target guard kills a path when $v = 1$ and disappears when $v = 0$. A target leaf is false when $v = 0$; when $v = 1$ it is a constant-true terminal and supplies no head unit to the opposite copy. A non-target leaf supplies a positive head unit. Every non-target path guard supplies a negative unit. Reject paths whose head is also guarded.

Append these path units to the opposite zero system and compute its greatest model. All clauses remain dual-Horn. A consistent model supplies a common set K making the path true and the opposite cofactor false. Conversely any detecting K has a true support path in one direction, and its actual opposite-copy gate values extend it to a model of that path’s zero system. Testing all paths in both directions is therefore sound and complete. Theorem 4.6 lifts any returned K to a physical test.

Each path test costs $O(G)$. A depth-first traversal with reversible guard counts streams paths without retaining all guard sets, using $O(G)$ working words and $O((P + 1)G)$ total work. In an unshared formula each path ends at a distinct leaf occurrence. For a forest, the total number of paths and outputs is $O(G)$. Process the outputs sequentially, retaining at most one zero system and path state; each test costs at most $O(G)$. This gives the asserted total time and space. $\square$

For a single direction only the true cofactor’s path count enters: the opposite cofactor may remain an arbitrary shared DAG. Both directions must still be resolved to conclude undetectability. Appendix D gives counterexamples to immediate shortcuts and a second algorithm whose parameter is the number and width of multi-positive zero-system clauses.

Fault and supplied decoder	Detection complexity
First-stage bridge; guarded DAG	$O((n + q)G)$ (Theorem 5.1)
First-stage bridge; formula forest	$O(G)$ (Theorem 5.6)
Final-qualifier stuck-at; guarded DAG	NP-complete (Theorem 7.8)
Final-qualifier stuck-at; formula forest	$O(G^2)$ (Theorem 7.9)

The table concerns the specified Boolean interfaces and qualifier observations. It does not classify arbitrary physical bridge or stuck-at faults elsewhere in a circuit.

8. Native encodings and independent answer certificates

The guarded algorithm in Section 5 receives the selector roles and the decoder. To apply it to an unannotated CNF, the implementation must establish that the clauses describe this guarded family. The live-AND extension of Section 7 remains a separate supplied-DAG algorithm. Checking a supplied description of those roles and discovering the roles are separate tasks; a worst-case discovery bound remains open. Appendix A gives a recognition contract for exposed complete gate blocks,

including the physical recurrences, shared primary inputs, drive interface, paired outputs and exact auxiliary projection. Every input clause must be accounted for. Failure to recover or verify a permitted representation returns **OUTSIDE** (or implementation status **OPEN**). These statuses mean that no representation was verified; they establish neither nonmembership nor unsatisfiability.

For a recovered SAT member, the eligible-set witness lifts by Theorem 4.1 to a primary test. Topological evaluation supplies the functional gate values. The bridge and observation projection records supply any private auxiliary extensions. Inverse alias transport gives an assignment to the original variables, and every original clause is then checked. Standard SAT solvers also return models with this interpretation on a complete native encoding. The family algorithm supplies the explicit semantic route to the witness.

For an UNSAT member of the guarded family’s specified native encoding, Theorem B.1 gives an ordinary DAG-resolution refutation with $O(N^3)$ inferences and no extension variables. Its proof first derives the selector image inside the native clauses, then compiles the guarded decision argument. The implementation emits the derivation and an independent checker replays it. Appendix B states the exact encoding, local derivation rules, inference bound and executable contract. The $O(N^3)$ bound counts inferences, not the byte size or wall time of the compiler.

An optional-activity observation uses auxiliary activity variables whose assertion requires a corresponding output discrepancy; a requirement circuit selects the activities that must hold. These variables need not be functional gate outputs. The optional-activity observation theorem (Theorem B.2) extends the proof bound over a native physical core, with a separate wrapper term. Its full general implementation remains open. The accompanying formalization contains an independent supplied-record checker: an accepted complete record establishes the recovered family projection of the exact original indexed CNF and provides model lifting. Discovering such a record and proving total discovery costs are separate obligations. Section 13 states the formalization coverage and remaining integration boundaries.

Proof-producing CDCL already supports independently checkable UNSAT certificates [WHH14], and resolution simulation results relate it to the native upper bound [PD11]. The specific result here is a guaranteed polynomial proof-generation schedule for the stated family. It supplies no lower bound on the runtime of CDCL or other general SAT solvers.

9. Experiments

The measurements compare a Python family implementation, including role recovery, with general SAT solvers on retained generated inputs. They validate executable behavior and document cost; the uniform complexity claims follow from the proofs. Appendix C records the exhaustive small checks, reduction checks and certificate controls.

Timed decoder families. The scaling panel uses drive one, target $t = 1$ and the single driver $s = 2$. With channel labels $1, \dots, n$, its decoders are

Label	Decoder
cycle	$\bigvee_{i=1}^n (h_i \wedge \neg h_{i+1})$, with $h_{n+1} = h_1$
chain	$h_n \vee \bigvee_{i=1}^{n-1} (h_i \wedge \neg h_{i+1})$
all-OR	$\bigvee_{i=1}^n h_i$
grid	$(h_t \wedge \neg h_s) \vee \bigvee_{uv \in E, u \in L} (h_u \wedge \neg h_v)$

For the grid, $n = k^2 + 2$; the grid channels are $v(a, b) = 3 + ak + b$ for $0 \leq a, b < k$. The edge set E

joins cells at Manhattan distance one, without wraparound, and L consists of cells with even $a + b$. Each edge contributes one term, directed from its even endpoint to its odd endpoint. The cycle, chain and all-OR cases use $n = 8, 16, 32, 64, 128$ with one and three buses; the grid cases use $k = 3, 4, 6, 8, 11$ with one bus.

All four decoders have healthy value one; the panel does not measure the healthy-zero support-search branch of Algorithm 1. The cycle is zero on a nonempty eligible set exactly when every channel is eligible. The chain and all-OR decoders are zero only on the unreachable empty set, while the grid decoder is zero on the reachable set $\{t, s\}$. Thus the cycle and grid cases are SAT, and the chain and all-OR cases are UNSAT.

Scaling against CDCL. We generated native controllers at growing sizes. The cycle decoders are SAT; the chain and all-OR decoders are UNSAT; the grid subfamily is SAT. Each input is given, as a bare CNF file, to:

- the native recognition route of Algorithm 2 (`recognize_hidden_priority.solve`);
- MiniSat 2.2;
- Glucose 4;
- iterated failed-literal unit propagation, returning UNSAT on contradiction and UNDECIDED at a fixed point. It has no SAT-output rule. This baseline is not an implementation of FIRE, which also propagates unobservability and records fault-specific requirements [IA96].

All three decision solvers agree with the theorem on all 35 inputs: 15 SAT and 20 UNSAT. Every returned SAT model is checked against every input clause. Probing refutes 10 inputs and leaves 25 undecided. The per-mode time limit is 60 s; no run times out.

Each mode runs once per input in a fresh child process, under Python 3.11.9 and PySAT 1.9.dev15 on macOS 15.4.1, ARM64. The measured child timer includes imports, input parsing, solver setup, model checks and the family route’s temporary report I/O. It excludes interpreter startup and input generation. These are single-run elapsed times, not repeated measurements of solver kernels. Recognition uses the native route; this experiment does not time the compact fallback. The generated CNFs, hashes and environment metadata are retained with the results. A representative subset is shown:

Decoder (answer)	Buses	Channels	Variables / clauses	Native family route	MiniSat	Glucose	Probing
cycle (SAT)	1	8	158 / 428	0.03 s	0.01 s	0.01 s	undecided
cycle (SAT)	1	32	614 / 1,628	0.12 s	0.01 s	0.01 s	undecided
cycle (SAT)	1	128	2,438 / 6,428	1.33 s	0.01 s	0.01 s	undecided
cycle (SAT)	3	8	418 / 1,158	0.05 s	0.01 s	0.01 s	undecided
cycle (SAT)	3	32	1,642 / 4,518	0.30 s	0.02 s	0.01 s	undecided
cycle (SAT)	3	128	6,538 / 17,958	3.75 s	0.19 s	0.14 s	undecided
chain (UN- SAT)	1	8	156 / 422	0.03 s	0.01 s	0.01 s	refutes

Decoder (answer)	Buses	Channels	Variables / clauses	Native family route	MiniSat	Glucose	Probing
chain (UN- SAT)	1	32	612 / 1,622	0.12 s	0.01 s	0.01 s	refutes
chain (UN- SAT)	1	128	2,436 / 6,422	1.35 s	0.01 s	0.01 s	refutes
chain (UN- SAT)	3	8	416 / 1,152	0.05 s	0.01 s	0.01 s	undecided
chain (UN- SAT)	3	32	1,640 / 4,512	0.30 s	0.02 s	0.02 s	undecided
chain (UN- SAT)	3	128	6,536 / 17,952	3.73 s	0.18 s	0.16 s	undecided
all-OR (UN- SAT)	1	8	142 / 380	0.03 s	0.01 s	0.01 s	refutes
all-OR (UN- SAT)	1	32	550 / 1,436	0.10 s	0.01 s	0.01 s	refutes
all-OR (UN- SAT)	1	128	2,182 / 5,660	1.13 s	0.01 s	0.01 s	refutes
all-OR (UN- SAT)	3	8	402 / 1,110	0.05 s	0.01 s	0.01 s	undecided
all-OR (UN- SAT)	3	32	1,578 / 4,326	0.29 s	0.02 s	0.02 s	undecided
all-OR (UN- SAT)	3	128	6,282 / 17,190	3.54 s	0.13 s	0.14 s	undecided
grid (SAT)	1	11	223 / 602	0.04 s	0.01 s	0.01 s	undecided
grid (SAT)	1	38	820 / 2,204	0.18 s	0.01 s	0.01 s	undecided
grid (SAT)	1	123	2,735 / 7,354	1.52 s	0.02 s	0.01 s	undecided

Reading the table.

- **Native family route.** It decides every input and constructs a model whenever one exists. Its Python time includes recognition and reaches 3.75 s. This finite panel establishes no average-case or asymptotic exponent.
- **CDCL.** MiniSat and Glucose are faster on every input, at most 0.2 s. Theorem B.1 gives short proofs for UNSAT instances of the specified guarded native encoding; it does not predict a solver's search time or explain performance on SAT instances. The scaling harness records answers and statistics, not resolution traces.
- **Probing.** It refutes only the one-bus UNSAT inputs. It cannot establish SAT.

The experiment establishes correctness on the retained finite panel and an end-to-end runtime comparison. The polynomial guarantees come from the structural proofs. This panel does not measure the separate extension kernels, and it does not separate parsing, discovery, decision and lifting costs or report repeated timing distributions.

Graph-cover expanders. A second panel uses Proposition 6.4 with the Margulis–Gabber–Galil construction [HLW06, Construction 8.1, Theorem 8.2]. For modulus a , its vertices are $\mathbb{Z}_a^2$; each (x, y) is joined to $(x + 2y, y)$, $(x + 2y + 1, y)$, $(x, y + 2x)$, and $(x, y + 2x + 1)$, with undirected edges and arithmetic modulo a . We delete loops and merge parallel edges before making the graph-cover controller. The resulting simple graphs retain every external vertex neighbourhood and have maximum degree eight. The multigraph spectral bound gives at least $(8 - 5\sqrt{2})|S|/2$ crossing edges when $|S|$ is at most half the vertices. Each outside vertex receives at most eight such edges, so the simple graph has vertex expansion at least $(8 - 5\sqrt{2})/16$. These inputs therefore belong to a uniform bounded-degree expander family.

We use all six moduli 3, 4, 5, 6, 8, 10, giving 9 through 100 graph vertices and one extra target channel. All controllers have one bus, drive one, a single driver, and no auxiliary Horn chain. Every input is SAT. Three fresh, sequential processes per method and size run under the environment above, with a 10-second wall limit per worker and a rotating method order. No run times out. Inputs and all Python sources are retained with hashes.

The decision methods receive the native CNF. RC2 [IMS19] minimizes the number of true faulty qualifiers on that same CNF by giving each negative qualifier a unit-weight soft clause. A second RC2 run receives the exact projected vertex-cover formula, with hard clauses $h_u \vee h_v$ and soft clauses $\neg h_v$; one is added to its optimum for the target. Both use Glucose3 as their SAT oracle. This compares two formulations of the same minimum-eligible-set objective. Each returned native model is checked against every clause. All six optimum values also pass a separate MiniSat check that no smaller cover exists, and each minimum cover lifts to a clause-checked native model. These optimum cross-checks use a second solver; they do not export lower-bound proof certificates.

The table gives medians in seconds, including imports, parsing, solver setup, decision or optimization, and witness checking. Interpreter startup, input generation, and the separate optimum cross-check are excluded. Raw repetitions and their minima and maxima are in the retained summary.

Graph vertices	Native variables / clauses	Native family decision	MiniSat	Glucose	RC2	RC2	Minimum eligible size
					minimum, native CNF	minimum, graph formula	
9	225 / 644	0.0376	0.0099	0.0099	0.0129	0.0099	7
16	418 / 1,195	0.0699	0.0108	0.0113	0.0141	0.0097	11
25	673 / 1,924	0.1332	0.0123	0.0132	0.0173	0.0101	18
36	988 / 2,825	0.2392	0.0133	0.0136	0.0227	0.0106	25
64	1,804 / 5,161	0.6615	0.0176	0.0182	0.1807	0.0126	44
100	2,870 / 8,215	1.5580	0.0235	0.0243	1.8598	0.1459	67

CDCL remains faster than the native family route on this panel. Minimum witness solving becomes more expensive at the larger sizes, especially on the native encoding, while all instances still finish within the limit. Neither this finite growth nor the NP-hardness theorem implies exponential runtime on this particular sequence. Corollary 6.5 establishes hardness over arbitrary embedded graphs; it does not establish optimization hardness for the fixed Margulis–Gabber–Galil sequence. The representation lower bound applies uniformly to its growing expander controllers by Theorem 6.9.

10. Related work and attribution

The comparisons are separated by the circuit, the decision procedure, and the tasks that remain hard after the reachable sets are known. Each citation is used for the theorem named with it.

- **Origin of the instances and encoding.** The SATLIB SSA and BF files were generated by Nemesis [Lar92], using the bridge-fault model of [FL91]. We analyze their compact CNFs and use complete Tseitin gate blocks for the native generalized family.
- **The c432 model.** The architecture comes from the reverse engineering in [HYH99], which [HH95] used for high-level test generation of the fixed circuit. We extend the qualifier recurrence to every n and b . The reachable-set theorem applies to that qualifier vector, before the channel encoder.
- **Tractable test-generation classes.** We compare against [IS75, FT82, Fuj90, CAB90, PCK99] in Sections 6–7. The original guarded controller result uses its semantic image; the circuit-width obstructions concern the specified native layout.
- **Untestability identification.** FIRE [IA96] and related implication methods are sound but incomplete. Algorithm 1 is complete for the grammar.
- **Regular structures.** C-testability of iterative arrays [Fri73] and the CAM priority-encoder test [Li05] bound test-set size for different devices and questions.
- **Horn tools.** The fixed point is [DG84]. More precisely, for a guarded decoder F , the transformation $H(x) = \neg F(\neg x_1, \dots, \neg x_n)$ is Horn-NNF [Ima21]. OR nodes become conjunctions, while a guarded child becomes its transformed child disjoined with negative channel literals. Each disjunction has at most one nonnegative disjunct. This syntax connection does not assert a polynomial-size clausal expansion on the original variables.

The direct union-closure argument behind the greatest-zero-set construction follows Proposition 5.2.

- **Backdoors.** Proposition 6.12 uses static syntactic classes. Dynamic sub-solvers [DGS14] and backdoor DNFs [OSS24] distinguish other measures of distance from tractability; the proposition supplies no runtime comparison with them.
- **Optimization and compilation.** Vertex-cover decision and counting hardness are classical [Kar72, PB83]. Theorem 6.9 imports the expander-CNF DNNF lower bound [BCMS15]. Proposition 6.4 embeds the graph relation into a physically realizable controller image; the restriction and projection justify applying those results to both primary detection and native CNF.
- **Proof complexity.** The resolution simulation in [PD11] applies to Theorem B.1’s native bridge encoding. Proof-producing CDCL solvers already support independent UNSAT checking, for example through DRAT [WHH14]. Their SAT models also give primary-input tests for the

complete native encoding. The family-specific result here is the semantic construction and its guaranteed polynomial native proof-generation schedule. The DNNF-hard controllers are satisfiable and concern representation of all models, so their bound is compatible with short refutations of UNSAT native members.

- **Parameterized lower bounds.** Clique hardness and its ETH-based exponent lower bound are classical [CHKX06]. Theorem 7.6 transfers them through an exact eligible-set bijection using $r - 1$ live ANDs, single negative guards and no shared internal gates. Theorem 7.5 is a classification of the three typed operations used here, rather than a classification of arbitrary Boolean constraint languages.

Proposition 5.3 is a transfer of monotone circuit-value P-completeness [Gol77; GHR95, Theorem 6.2.2]. Proposition 6.6 transfers independent-set optimization and counting, and Theorem 6.8 uses clone interpolation to handle native-model multiplicities. The extension oracle and prefix enumeration specialize standard self-reduction. These are applications to the controller representation; the general source hardness and algorithmic methods are classical.

The stuck-at reduction of Theorem 7.8 specializes classical SAT hardness to the exact pair image, with shared guarded gates and one target guard. Theorem 7.9 combines support paths with the classical dual-Horn algorithm; Theorem D.2 uses elementary clause strengthening and implication-graph reachability. These supplied-structure results have separate scope from the first-stage native proof compiler.

The central contribution is the size-uniform description of the reachable qualifier sets, together with its composition with decision, test lifting, and native refutation. The graph-cover restriction gives the task and representation boundaries for those same sets. The literature search is bounded: priority for the exact reachable-set theorem, the controller embeddings and these structural characterizations remains unestablished.

11. Limitations and open problems

1. **Fault model.** The native recognition and proof contract concerns the first-stage bridge of Section 3. Proposition 4.5 covers a specified stage-local bridge with unrestricted primary inputs. Theorem 4.6 covers only the final-qualifier request occurrence, and Theorems 7.8–7.9 use supplied guarded decoders. Their general CNF recovery and proof export are not established. Other physical bridges, enable-wire faults and decoder faults require separate image characterizations.
2. **Recognition.** The supplied-record checker verifies the exposed representation and exact projection contract of Appendix A. Discovery completeness on an explicit syntactic input class and total worst-case discovery time and space bounds remain open. Equivalent CNF rewrites can defeat recovery. The cubic raw-interface candidate count does not settle these obligations; every original clause and any restriction on primary inputs must still be accounted for.
3. **Proof translation.** Theorem B.2 covers optional-activity observations over a native physical core. Translation for every accepted compact physical encoding, and full implementation of the general wrapper theorem, remain open. Formula-forest and bounded-AND kernels operate on supplied DAGs and are separate from CNF recognition and native proof export for the original grammar.
4. **Engineering comparisons.** The reported family implementation is slower than MiniSat and Glucose on the generated decision panels. The measurements do not isolate recognition cost or establish an asymptotic solver comparison. The scaling and expander panels exercise healthy-one outputs; healthy-zero SAT and UNSAT cases and phase-separated costs need their own timing

panel. Direct comparisons with FIRE and recursive learning have not been reported.

5. **Tighter family bounds.** The native $O(N^3)$ resolution bound may admit improvements. A total discovery bound must first be established for the recognition procedure. The live-AND ETH bound controls the order of dependence on k in the exponent, not its optimal constant. Neither result supplies an unconditional superpolynomial time lower bound.

12. Conclusion

The controller recurrence admits an exact description of its reachable qualifier sets, and every set in that description has an explicit primary input. Composing these sets with guarded decoders gives constructive polynomial detection. The native resolution construction explains why unsatisfiable members of the specified functional encoding have short proofs, while optimization, counting and representation tasks can remain hard on the same architecture. The exact stuck-at pair image marks a further boundary: a supplied shared guarded decoder gives an NP-complete problem, while an unshared formula forest remains polynomial. The observation interface and encoding contract determine which conclusions apply; formal and experimental coverage is specified separately below.

13. Reproducibility

The source archive on [Zenodo](#) contains this manuscript, four vector figures, a finite-check supplement for the fault-model extensions, and an amendment verification supplement. Their READMEs give replay commands and coverage. The finite checks establish reported finite behavior; uniform statements are proved in the manuscript.

Registered Lean formalizations. Eight Palomar entries record selected Lean statements at immutable source commits. Their theorem names, hypotheses, definition implementations and verification evidence are specified by the versioned records below. The scope column summarizes those selections; it does not identify every manuscript statement with a selected declaration.

Record	Selected formal scope
[Sha26a]	Exact first-stage bridge images, observation equivalence, nonemptiness obstruction and a specified qualifier-only stuck-at classification
[Sha26b]	Supplied guarded-DAG detection correctness and completeness, constrained extensions and native cold-detector execution
[Sha26c]	Canonical enumeration, iterator and streaming execution, emitted witnesses and selected delay and storage bounds
[Sha26d]	Formula-forest execution and attained nonnegative-weight maxima for healthy-zero observations
[Sha26e]	Dual-Horn and vertex-cover embeddings, counted decoder size and detecting eligible-set counts
[Sha26f]	Critical sets, test-list and query bounds, source recovery, native width and backdoor bounds; cover-layout width under the selected hypotheses
[Sha26g]	Bounded-support and selector expansion, bounded-conjunction and proper-basis execution, with selected XP and healthy-one FPT inequalities

Record	Selected formal scope
[Sha26h]	Ordinary-resolution construction for native encodings and supported observation wrappers, with inference and derived-literal storage bounds

Each entry verifies its selected formal statements at its recorded commit. The workspace correspondence table below separately records which complete statements of this manuscript have an accepted paper-to-Lean match. Registry verification does not extend that correspondence automatically, and companion source declarations are not additional selected registry targets.

Formalization coverage. The workspace coverage map records statement-level acceptance separately from acceptance of component tasks. The following table accounts for all 41 numbered items using this paper’s numbering.

Recorded coverage	Statements
Statement-level acceptance (6)	4.1, 4.2, 4.3, 4.4, 6.3, 6.4
All directly listed component tasks accepted; whole-statement correspondence not certified (19)	5.1, 5.2, 5.4, 5.5, 5.6, 6.1, 6.2, 6.7, 6.10, 6.12, 7.2, 7.3, 7.4, A.1, A.2, A.3, A.4, B.1, B.2
Required component tasks incomplete; source-level full statements (10)	5.3, 6.5, 6.6, 6.8, 6.9, 6.11, 7.1, 7.5, 7.6, 7.7
Source proofs without recorded formal coverage (6)	4.5, 4.6, 7.8, 7.9, D.1, D.2

The amendment supplement contains a snapshot of the workspace correspondence records. Acceptance of a component checks its stated contract; a full paper statement additionally requires matching its hypotheses, conclusion and resource bounds. No P-, NP-, or #P-completeness, W[1]-hardness, ETH consequence or DNNF lower bound is claimed as formally verified.

The public Lean sources and selected verification evidence are available at the immutable commits linked by the Palomar records above. This archive does not include the separate Lean repository, the Python native compiler and answer checker, or a complete reproduction package for the timing panels. The formal Lean selections and the reported Python experiments are distinct artifacts.

The amendment supplement reconstructs the canonical manuscript from its pinned baseline and 40 declared replacements. It checks the declared changed blocks, recorded coverage and preservation of all 41 numbered statement/proof blocks in this deposit source. It checks document integrity, not mathematical proofs or Lean execution.

Appendix A. Recognition from exposed CNF encodings

Algorithm 2 (RECOGNIZE-AND-DECIDE) receives only a CNF file.

1. Enumerate at most $O(N^3)$ raw bridge-interface candidates, as counted below. Check that their bounded signed blocks project exactly to $U = 1 - d, D = d, W = d$.
2. Recover the functional circuit from complete local gate blocks, orienting each producer along the DAG. The unique-producer route fixes AND/OR outputs first, then orients each XOR by its sole variable not among those outputs. Infer roots as variables with no producer and check acyclicity. For a route admitting alternative producers, verify the descendant condition of Lemma A.4 on the recovered DAG itself.
3. Recognize the selector chain:

- the first flag is an OR of disjoint two-primary products;
- each later stage is the unique continuation by one fresh request per channel.

Then check both copies against the physical XOR recurrence and the qualifiers.

4. Normalize the paired observed expressions to a common guarded decoder.
5. Project the remaining auxiliary clauses (Theorem A.1). They must reduce to the single requirement that the observed outputs differ.
6. For a candidate passing all representation checks, run Algorithm 1. Return its decision, lifting any detecting set to a full model checked against every original clause.
7. If every candidate fails the representation checks, return **OUTSIDE** (or implementation status **OPEN**): no representation was verified. Neither status certifies nonmembership or unsatisfiability.

Accepted representation. The native route requires complete exposed gate blocks. Primary inputs are not candidate outputs; each XOR block has exactly one variable not produced by an AND/OR block, its output. The selector continuation and final qualifier expressions must be recoverable from their factor footprints, and paired outputs must normalize to the same guarded DAG. Private residual auxiliaries have, after a consistent polarity choice, Horn pure clauses and only negative auxiliary occurrences in signal attachments. Their least-model projection must consist of units requiring the detected output discrepancy. The optional-activity grammar of Theorem A.2 instead supplies checked constant extensions. All clauses must belong to recovered definitions or the verified projection; no unexplained constraints may remain. These are sufficient representation conditions, not a recognizer for arbitrary equivalent CNFs.

The complete local templates are the blocks of Section 2, their signed variants and bounded truth-table compositions. A local table is accepted only after every row agrees with its defining clause block. A selector role map labels distinct free roots as the enables and requests and matches both copies to every displayed recurrence in Section 3. It also records the target, the disjoint driver and remaining-channel sets, and the bridge cofactor. Unused roots may serve only as private projection auxiliaries; an additional restriction on a primary input is not admitted.

Permitted normalization. Expression nodes are interned after constant evaluation, associative/commutative flattening of AND/OR, duplicate removal, complement-pair simplification, and cancellation of repeated XOR operands modulo two. OR is represented by De Morgan complementation when convenient. Disjoint factoring of a bounded-support truth table is allowed only when all its rows verify the proposed factors. The decoder parser accepts the grammar in Section 5 and requires identical canonical node descriptions on the two sides after qualifiers have been replaced by corresponding channel atoms. Equality under these operations is a sufficient acceptance test; arbitrary Boolean equivalence is not invoked.

Representation-certificate contract. A checkable recovery record contains the following data, all indexed to the original clause list:

1. The interface clause IDs, signed alias substitutions and their defining clauses, and the exact bounded projection table relative to the retained driver definitions, including both directions of the extension property.
2. The gate output/input map, defining clause IDs and local truth tables, a topological order, and the inferred roots. If a producer has an alternative, list its inputs and check that at least one is a proper descendant of the output in this recovered order's DAG. This makes Lemma A.4's premise checkable without an external intended circuit.
3. The selector/qualifier role map and the paired decoder normalization described above, with the bounded factor identities used in matching.
4. The auxiliary polarity choice and projection evidence. For negative Horn attachments, supply

the least auxiliary model and its propagation reasons. For optional activities, supply the tree and the constant extensions of Theorem A.2. The general semantic condition in Theorem A.1 is not itself a polynomial certificate for arbitrary definitions.

5. A clause-coverage table. Every original clause must be a verified gate, interface or auxiliary clause, or become a tautology under a justified alias substitution. Retain every duplicate clause ID. Reject a record with any unexplained clause or unproved transformation.

To justify the negative-Horn projection in item 4, orient the private auxiliaries so that their pure clauses are Horn and every signal-bearing attachment has only negative auxiliary literals. Let L be the least model of the pure clauses. In an attachment, if all its auxiliary premises belong to L , its signal clause is necessary in every extension. Otherwise L satisfies the attachment independently of the signals. Hence these necessary signal clauses are exactly the existential projection. Verify that the resulting clauses require the accepted output discrepancy.

Replaying these checks proves equisatisfiability with the recovered controller and its exact observation condition. Algorithm 1 therefore justifies an UNSAT answer as well as a SAT answer. Checking a single returned SAT model is a separate, weaker validation. The implementation records recovered roles, projection data and clause coverage and checks models. The certified entry point additionally exports a model or an ordinary-resolution refutation and checks it against the original CNF (Appendix B). This certifies the SAT/UNSAT answer without trusting role recovery. The formalized supplied-record checker separately validates complete representation payloads, including original clause identities, family projection and model lifting. This soundness statement consumes a supplied record; it is not a completeness theorem for the discovery procedure. The accepted representation conditions are those stated above.

Theorem A.1 (observation projection). Let $D(Z, X)$ be acyclic gate definitions. Let $H(A, X)$ be the remaining clauses, with private auxiliaries A , activity literals s_i over A , and guarded observations

$$s_i \rightarrow (u_i \oplus v_i).$$

Suppose two conditions hold for a set J of these observations.

1. The auxiliary-only clauses together with $\{\neg s_i : i \in J\}$ have a unit-propagation contradiction.
2. For each $i \in J$ and each unequal orientation of (u_i, v_i) , either the orientation contradicts D , or there is a constant assignment $a_{i,\text{orientation}}$ to all of A such that **every** assignment to Z, X satisfying D and that orientation satisfies $H(a_{i,\text{orientation}}, X)$.

Then $D \models [(\exists A H) \leftrightarrow \bigvee_{i \in J} (u_i \oplus v_i)]$.

Proof. By (1), any model makes some accepted activity true, and its guard clauses force the discrepancy. Conversely, a realized discrepancy cannot contradict D , so (2) supplies an extension. $\square$

The next grammar makes both conditions structural. Its connectivity graph can be arbitrarily complicated because its internal activities can all remain inactive; the requirement to observe a discrepancy is carried by a separate tree.

Theorem A.2 (optional activities). In the setting of Theorem A.1, partition the auxiliaries into internal activities a_v , observed activities $s_1, \dots, s_r$ with $r \geq 1$, and requirement-tree variables. After one consistent choice of auxiliary polarity, let H consist exactly of the following clauses:

1. Discrepancy guards $a_v \rightarrow (u_v \oplus v_v)$ and $s_i \rightarrow (u_i \oplus v_i)$, each encoded by its two clauses.

2. Clauses $\neg a_v \vee (\text{disjunction of positive auxiliary literals}) \vee C(X)$, where $C(X)$ is any clause on signal literals and may be empty. In particular, connectivity clauses $\neg a_v \vee \bigvee_{w \in S} a_w$ are allowed without restrictions on cycles, shared successors, or overlapping successor sets.
3. A rooted binary requirement tree whose leaves are exactly the distinct observed activities, whose internal nodes are fresh auxiliary variables, and whose clauses are $\neg p \vee \text{child}_1 \vee \text{child}_2$, together with the positive root unit. For one observation the root may be that leaf.

Then, at every size,

$$D \models \left[(\exists A H) \leftrightarrow \bigvee_{i=1}^r (u_i \oplus v_i) \right].$$

A realizing assignment to the auxiliaries can be constructed in time linear in their representation size once a discrepant observed pair is known.

Proof. If all observed activities are false, unit propagation forces each requirement-tree node false from its children, contradicting the root unit. This establishes condition 1 of Theorem A.1.

For condition 2, choose one discrepant observed pair. Set every internal activity false, set its observed leaf and the path to the root true, and set all other observed leaves and off-path tree nodes false. Every clause of type 2 is satisfied by its negative internal premise. Each active tree node has an active child, and the only active discrepancy guard is true by the chosen unequal pair. The assignment is independent of every other signal value, so it is a constant extension under either unequal orientation. The construction visits each auxiliary and tree edge at most a constant number of times. $\square$

Corollary A.3 (selected observations). For any nonempty $J \subseteq \{1, \dots, r\}$, append the clause $\bigvee_{i \in J} s_i$ to the grammar of Theorem A.2 and retain all its clauses. Its exact projection is $\bigvee_{i \in J} (u_i \oplus v_i)$.

Proof. The added clause and guards force a discrepancy in J . Conversely, the extension in Theorem A.2 for a discrepant pair in J satisfies the added clause as well. It imposes no equality on unobserved signal pairs. $\square$

Theorem A.2 allows arbitrary numbers of activities and edges. Requiring an internal fault-site activity to become true or to traverse a full path would change its hypotheses. For composition with Algorithm 2, the added clauses must also preserve the exposed selector and qualifier recovery conditions.

Lemma A.4 (producer recovery). Suppose the candidate producers contain the intended acyclic producer for every non-root signal and none for a root. Suppose every alternative producer for a signal depends on a proper descendant of that signal in the intended DAG. Starting with the roots, repeatedly select any candidate whose inputs are known and whose output is unknown. This recovers all intended producers, independently of the candidate iteration order.

Proof. Maintain that known signals contain all their intended ancestors and were computed by their intended producers. An alternative for an unknown signal cannot be ready: its known descendant would already require that signal to be known. Selecting an intended producer preserves the invariant. If any signal remains unknown, the earliest unknown signal in a topological order has a ready intended producer. Thus the process terminates with the entire DAG recovered. $\square$

Raw-interface candidate count. Fix an absolute constant maximum clause width w , let M count input clause occurrences, and let g be the number of extracted raw gates. Each distinct clause and head produces at most one gate, so $g \leq wM$. The pairing routine emits each unordered gate pair at most once, with at most $w - 1$ common-input choices per pair. A yielded record has three distinct

absolute variables for U, W, D : signed alias relaxation cannot separate literals of the same variable. Thus at most one vertex-biconnected alias block can contain all three. Indeed, two distinct maximal vertex-biconnected blocks share at most one vertex; if they shared two, their union would remain connected after any single-vertex deletion and would form one block. Finally there are at most g driver choices. Clones are collected deterministically, and final deduplication can only reduce the count. The number of yielded records is therefore at most

$$(w - 1) \binom{g}{2} g = O(M^3) = O(N^3).$$

This argument assumes the alias graph’s vertex-biconnected decomposition and consistent signed aliases; inconsistent aliases are rejected. It is a count of yielded candidates, not a bound on the time spent scanning failed candidates, rebuilding aliases or constructing each record.

Discovery scope. The implementation restricts local gate supports to five inputs and normalized bridge interfaces to ten variables, rejecting candidates exceeding these limits. These are independent representation restrictions. Discovery uses exposed gate pairs, alias blocks, bounded local tables and selector-factor matches. The native and compact routes use different recovery promises. The compact fallback checks at most three buses; increasing this limit exposes a 2^{2b} local-table factor, while the growing-bus route uses native factor footprints.

No complete worst-case discovery-time or space bound is established here. Such a bound requires a schedule accounting for all searches, intermediate representations and failed candidates, together with a precise syntactic class on which discovery is complete. The supplied-record checker proves correctness of accepted records, including exact projection and model lifting; it does not supply that discovery theorem. Condition 2 of Theorem A.1 uses explicit projection evidence, not a general test for arbitrary Boolean definitions. Variable renaming, checked signed aliases and permutations are supported; arbitrary equivalent rewrites may prevent recovery.

Transfer to the SATLIB anchor. On the compact bf0432-007 encoding, the implementation recovers channel roles and selector recurrences without source names. It abstracts 20 paired guarded signal functions: eight standalone qualifier pairs, eight intermediate guarded expressions and four decoder sinks. The observation projection selects seven discrepancies: four guarded decoder pairs and three bus-flag discrepancies.

The signed observation table below is in normalized alias coordinates. For decoder pairs, swapping the listed pair and complementing both entries gives the healthy and faulty instances of the same guarded function. Bus rows identify equality of pair-XOR functions; they do not identify the individual literals with historical PA/PB/PC wires. Bus numbers in this table are one-based.

Activity literal	Signed pair	Observed function
56	(−57, −55)	Guarded decoder, replay DAG root 11
53	(−54, −52)	Guarded decoder, replay DAG root 15
50	(−51, −49)	Guarded decoder, replay DAG root 17
47	(−48, −46)	Guarded decoder, replay DAG root 18
44	(−45, −43)	Bus-3 healthy/faulty flag discrepancy
41	(−42, −40)	Bus-2 healthy/faulty flag discrepancy
38	(−39, −37)	Bus-1 healthy/faulty flag discrepancy

The implementation handles bus discrepancies directly and applies Algorithm 1 when the observation

interface contains only the guarded decoder pairs. The full anchor therefore uses the combined bus-or-decoder route. The main image-based detection theorem retains its qualifier-only observation hypothesis.

The signed map and its decoder DAG are retained with the exact CNF hash `ff22b8c5c381229a7ec7ff2f8d93afc7219a73034c4180b056b8a5594af1a75a`. The replay accounts for all 3,668 clauses and returns UNSAT for the fixed drive-zero anchor. Independent SAT checks of the recovered functional definitions and residual clauses verify 65 projection implications, exclude three impossible orientations and refute the all-observations-equal necessity miter. These projection checks omit the bridge-interface constraints. They establish the finite anchor projection, not a uniform discovery-cost theorem.

A separate cofactor-congruence simplifier also refutes it directly. For `ssa0432-003`, an analogous recognizer applies Theorem 4.4. Its current exposed representation guarantee requires exactly three buses and $n \geq 2$; the semantic classification of Theorem 4.4 holds at every stated n, b .

Appendix B. Native resolution bounds and proof compilation

Theorem B.1. Every UNSAT member of the native bridge encoding with guarded decoders has an ordinary DAG-resolution refutation with $O(N^3)$ inferences and no extension variables.

Here the native encoding consists of complete bounded-fan-in gate blocks for Section 3, explicit drive units $U = 1 - d, D = W = d$, two identical guarded decoder DAGs, and the asserted OR of their output XORs. There are no extra primary restrictions. Any auxiliary wrapper requires its own proof translation.

Proof. We separate the selector derivations from decoder reasoning. All local premises are the complete blocks in Section 2; signed wires stand for their native literals. The following elementary expansion rules will be used throughout.

Local derivation rules. For an AND tree with root a and leaves $l_1, \dots, l_s$, resolving down the tree gives every $\neg a \vee l_j$ and $a \vee \neg l_1 \vee \dots \vee \neg l_s$. The dual clauses hold for an OR tree. For a guard chain these give $\neg a \vee u$, every $\neg a \vee \neg h_j$, and $\neg u \vee \bigvee_{j \in B} h_j \vee a$, where u is its live child. They use existing gate variables and at most linear work per path or combined clause. For a non-tautological requested clause whose falsification gives a unit-propagation contradiction, resolve the final conflict with its reasons in reverse propagation order. This derives a subclause of the request in at most $O(N)$ inferences. Retain the stronger subclause and omit tautological goals. When a fixed number of correlated gate inputs requires case splitting, branch on those inputs and resolve the resulting constant-size tree. No tautological clause is introduced as an axiom.

Selector nonemptiness. First take drive one and restrict to $P'_1 = 0$. The interface forces $e_t = r_{t1} = 1$ and every non-target first local request to zero. Write $p_j = P'_j$, $z_{ij} = t'_{ij}$, $g_{ik} = \neg p_k \vee r_{ik}$, and $H = \bigvee_i h'_i$. The symbols g and h' are existing gate outputs; H below denotes a clause, not a new variable. For $j \geq 2$, derive

$$\begin{aligned} &\neg z_{ij} \vee g_{ik} && (k \leq j), \\ &\neg z_{ij} \vee p_{j+1} \vee \dots \vee p_b \vee h'_i, \\ &\neg p_j \vee p_{j+1} \vee \dots \vee p_b \vee H. \end{aligned}$$

For the first clause, assuming $z_{ij} = 1, g_{ik} = 0$ forces $p_k = 1$ and $r_{ik} = 0$. The earlier mask then forces the earlier local request true, a contradiction; $k = j$ uses the direct request factor and $k = 1$ uses

$p_1 = 0$. These are RUP derivations. The first clauses and zero later flags force every factor of h'_i , giving the second clause. Resolving through the OR tree for p_j gives the third. Descending induction on j eliminates the later flags and yields $\neg p_j \vee H$. The known target enable also gives $p_2 \vee \dots \vee p_b \vee H$, since zero suffix flags force h'_t . Resolving these clauses derives H ; for one bus it derives h'_t . There are $O(nb^2) = O(N^2)$ such macro clauses. Each RUP macro expands into at most $O(N)$ ordinary resolutions along its reason trail.

Healthy singleton. The healthy first flag is one. Its non-target first masks are zero, killing every later non-target local contribution. The target mask is one. Inductively, the target's local contribution at stage j equals r_{tj} , the flag equals that contribution, and the target's next mask is $r_{tj} \oplus \neg r_{tj} = 1$. Each correlation uses only the constant local XOR/OR truth table after preceding identities have been derived. Every target qualifier factor is therefore one. For $i \neq t$, assuming $h_i = 1$ forces $e_i = r_{i1} = 1$ through its first qualifier factor, contradicting $x_i = 0$ from the interface and first-flag cofactor. Thus derive $h_t = 1$ and $h_i = 0$ for $i \neq t$, and evaluate the healthy decoder to obtain each c_k . At most $O(N)$ such gate identities suffice; each identity is represented by its two clause directions, and evaluation then follows the gate order.

Healthy-one decoder outputs. If $c_k = 1$, the forward decoder clauses, H , and $\neg a_{\text{root}}$ form the dual-Horn system of Algorithm 1; add h'_t for one bus. An UNSAT family member has no detecting set, so this system is inconsistent. Complementing every variable makes it a Horn system. Mark variables true by the propagation of [DG84]: whenever every negative literal of a clause is marked, mark its positive literal. When this stops, the marked set satisfies every clause that has a positive literal, so an unsatisfiable system contains a clause without one whose variables are all marked. Resolving each fired clause against the units of its negative literals derives its positive literal, and the final clause then yields the empty clause. Each clause is used at most once, so this refutation has at most one inference per literal occurrence. Without the premise $\neg a_{\text{root}}$, the same inferences derive a_{root} or a subclause of it, as in the cofactor lifting below, in $O(N)$ inferences. Decoder variables here are existing gate outputs. Clauses for a multi-guard node are obtained by resolving along its native AND tree.

Healthy-zero decoder outputs. If $c_k = 0$, associate a support A_v with each faulty decoder node. A channel leaf has support $\{i\}$, zero has empty support, OR takes unions, and a guard removes its blocked indices. In the one-bus case a guard on t gives empty support. Derive

$$\neg a_v \vee \bigvee_{i \in A_v} h'_i.$$

For a channel leaf the displayed summary is the tautology $\neg h'_i \vee h'_i$; omit it. At its parent, the definition already contains the leaf literal needed by the summary induction. A constant zero uses its defining negative unit. At OR nodes, resolve the output-to-children clause with nonleaf child summaries, retaining leaf literals directly. At guards, transfer the child summary and remove blocked leaves using $a_v \rightarrow \neg h'_j$; if the child is a leaf, start from $\neg a_v \vee h'_i$ in the guard definition. A one-bus guard on t immediately yields $\neg a_v$ from h'_t . A surviving root leaf supplies a true path for the reachable singleton or target-plus-singleton. Since none detects, A_{root} is empty and the summary gives $\neg a_{\text{root}}$. Shared nodes reuse their summaries. Thus all paired outputs agree, contradicting detection.

Cofactor assembly and cost. For drive zero, or drive one with $P'_1 = 1$, the first flags agree. Derive equality of corresponding gate outputs in topological order from the common inputs. Bounded fan-in makes each step a constant-size derivation. The two cofactors in drive one lift to complementary clauses and resolve to the empty clause. Assumption discharge and restriction lifting can be made weakening-free: keep the derived subclause whenever a pivot disappears. Equivalently, lifting a

refutation under $P'_1 = \varepsilon$ yields a subclause of the complementary literal. Either it is empty already, or the two cofactor literals resolve to the empty clause.

The selector, decoder summaries and equality steps fit $O(N^2)$ macros, each costing at most $O(N)$ resolutions. There are only two global cofactors. This proves the inference bound; total derived literal occurrences are at most $O(N^4)$. Encoded proof size and compiler running time require separate accounting; neither is identified with the cubic inference count. $\square$

Theorem B.2 (native core with optional observations). Retain the native physical core, drive interface and paired guarded decoders of Theorem B.1, replacing its final observation assertion by the wrapper of Theorem A.2. Observed pairs are decoder outputs. Every UNSAT such formula has an ordinary-resolution refutation using $O(N_0^3 + L_H)$ inferences and no extension variables. Here N_0 counts the native core and observed output incidences, and L_H counts the wrapper. The selected-observation variant of Corollary A.3 has the same bound.

Proof. Observation projection makes an UNSAT wrapper equivalent to the absence of any accepted output discrepancy. Fix a native cofactor. When the first flags agree, gate congruence derives output agreement from the common inputs. In the differing cofactor, the physical clauses derive the healthy singleton and faulty nonemptiness, with h'_t in the one-bus case. For a healthy-one accepted output, its zero system is inconsistent: otherwise Theorem 4.1 would realize a detecting input. Unit resolution and discharge of the zero-root assumption therefore derive the faulty root true. For a healthy-zero accepted output, the support-summary argument of Theorem B.1 gives empty root support and derives the faulty root false. Healthy evaluation supplies the matching unit in each case.

These derivations use only native physical and decoder clauses within the cofactor, before any observation clause is used. They establish equality of each accepted output pair, or stronger subclauses, within the stated cubic budget. In Corollary A.3, this argument is applied only to selected outputs; unselected outputs may still differ.

For an output pair (u, v) with equality clauses $\neg u \vee v$ and $u \vee \neg v$, and activity s , resolve its discrepancy clauses $\neg s \vee u \vee v$ and $\neg s \vee \neg u \vee \neg v$ against these equalities. Two resolutions give $\neg s \vee v$ and $\neg s \vee \neg v$; a third gives $\neg s$. Propagate these units through the requirement tree with two resolutions per internal node, then against its positive root unit. For q leaves this suffix has at most $3q + 2(q - 1) + 1 = 5q - 1$ inferences after the equality clauses; an identical-wire pair needs only its guard clauses to derive the negative activity. In the selected variant, resolve the selected negative activities against the added disjunction instead.

For a weakening-free construction, append the same suffix inside each native cofactor, use stronger subclauses whenever a resolution pivot disappears, and lift the two resulting refutations as in Theorem B.1. The constant number of cofactors changes only the constant in $O(L_H)$. Internal optional-activity clauses need not be used, so their arbitrary connectivity, including cycles, causes no additional proof cost. $\square$

This theorem requires the native core variables and definitions. It does not establish a simulation for compact physical rewrites eliminating those wires. The finite suffix check validates its new local resolutions; it does not constitute an implementation of a general wrapper compiler.

By [PD11], CDCL with restarts polynomially simulates resolution, so the family has no proof-length separation from CDCL. That does not mean a fixed solver heuristic finds the proof. The SATLIB anchors have checked ordinary refutations of 12,100 inferences (bf0432-007) and 887 inferences (ssa0432-003).

Executable answer certificates. The certified entry point runs the structural solver and requires in-

dependent validation before returning SAT or UNSAT. SAT certificates contain a complete assignment. The native proof compiler implements Theorem B.1 using gate congruence, last-active-bus induction, guarded-decoder summaries, and the two first-flag cofactors. It uses no SAT solver. Correlated gate identities split on at most five local inputs; all assumptions are discharged into proved clauses. Shared physical DAGs are traversed with memoization rather than unfolded into paths. A supplied native circuit description is checked against every original clause, or the exposed-wrapper recognizer recovers the roles. Aliases and projected observation units are themselves derived. The compiler emits ordinary resolution over the original clauses and variables.

The checker uses only Python’s standard library: it verifies the input and proof hashes, every signed pivot and exact binary resolvent, and the final empty clause. It imports neither the recognizer nor a SAT solver and uses explicit validation that remains active under `python -O`. No extension variables are accepted.

The native compiler’s schedule has $O(N^2)$ macros, each with a constant local case tree and $O(N)$ resolution expansion. It therefore realizes the $O(N^3)$ inference and $O(N^4)$ literal-storage bounds. These are proof-size bounds. For propagation, the active database contains the original clauses and completed lemma conclusions, including the constant-size local case trees: $O(N^2)$ clauses of width at most $O(N)$. Intermediate resolution clauses are stored in the proof DAG and are not automatically activated. Each RUP check initially scans the active database, then revisits a clause only when one of its literals becomes false. Each literal can trigger this revisit once, so a clause is inspected at most $O(N)$ times, at $O(N)$ literal cost per inspection. One check therefore costs at most $O(N^4)$, and $O(N^2)$ checks give a conservative $O(N^6)$ kernel allowance with constant-cost hash operations. This is a word-operation allowance, not an unconditional worst-case bound for Python hashing; recognition is charged separately. The Python native compiler and its input-format documentation are not included in this source archive; the selected Lean resolution results are available through the records in Section 13.

Structural-only certification uses this compiler and returns OPEN on unsupported representations. Automatic mode can instead use a bounded Glucose3 run, expanded from RUP into ordinary resolution, for compact or other recognized representations outside the native compiler. That fallback has no polynomial proof-discovery guarantee. If proof production exceeds its budget or checking fails, the entry point returns OPEN. Answer certification and independent verification of recovered family membership remain separate contracts.

Appendix C. Finite validation and certificate controls

Correctness checks. These were run on the reference implementation; the evidence is listed in Section 13.

- 17 exhaustive reachability checks on small circuits.
- 200 random guarded decoders, with 4,162 eligible-set rows and 1,200 fixed points.
- All 512 qualifier assignments of the c432 decoder.
- 86 complete generated CNFs (32 SAT, 54 UNSAT) up to 16 channels, 32 buses, and 22,820 variables. Every SAT model was checked against every clause.
- 494 unique-set decoders.
- 138 compact-composition cases.
- 336 translated formulas for the NP-completeness reduction.

The separate extension module checks the constructions against exhaustive Boolean evaluation. The deterministic seed is 20261009:

- All 1,094 nonempty simple graphs on two through five vertices: 33,804 set evaluations confirm the exact graph-cover bijection, counts and minima.
- 250 dual-Horn inputs, including units, tautologies and empty clauses: 1,890 assignment rows confirm Proposition 5.2.
- 400 bitset, 400 formula-forest and 400 bounded-AND decision checks, compared with exhaustive evaluation over 22,698 reachable-set rows; 1,200 additional checks of sparse zero-output compilation.
- 13,932 rows check the guarded-expansion identity on random DAGs with at most three live ANDs; a fixed two-leaf conjunction requires the predicted nonsingleton witness. Four malformed/out-of-scope decoder controls reject.
- Seven native CNFs from the graph and dual-Horn generators (five SAT, two UNSAT) agree with the existing recognizer, MiniSat and Glucose. All 15 SAT models satisfy every clause. Both UNSAT inputs also have independently replayed ordinary resolution proofs with no extension variables. The graph restriction used in Theorem 6.9 is checked on all 28 enable assignments for three small graphs. An unexplained primary restriction is rejected by the recognizer.

These finite checks exercise the semantic algorithms and the native embedding. They do not test an exponential DNNF lower bound empirically. The extension kernels take supplied DAG roles; they are separate from the native recognizer used for the timing panel in Section 9.

Certificate checks. The certified entry point checks four generated SAT models and ten generated UNSAT refutations, covering one, three and five buses, both drives, auxiliary chains, and a signed variable renaming. The UNSAT proofs have 16 to 33,321 resolution inferences. Both SATLIB anchors also pass through the certified dispatch. Out-of-scope input and a forced proof timeout return OPEN. The standalone checker passes 21 controls, including rejection of 18 corrupted proofs, models or bindings; every control is repeated with Python assertions disabled. These checks establish the reported executable behavior, while the correctness theorem remains the argument in Sections 4–7 and Appendices A–B.

Native structural proof compilation. A separate suite checks 80 native CNFs: 48 UNSAT refutations and 32 SAT models, all agreeing with MiniSat. Fifty-six drive-one answers are also compared with exhaustive reachable-set evaluation; 19 use the drive-zero rule and five larger chain instances use the family theorem. The suite includes 48 seeded guarded DAGs, shared nodes, overlapping guards, multiple outputs and driver channels, both drives, and signed renaming. Channel and bus counts grow independently, up to 64 channels and 16 buses. The emitted refutations contain 9 to 24,767 inferences. Every producer runs with SAT library imports blocked, and every certificate is replayed with Python assertions disabled. The 14 exposed-wrapper inputs also pass through the compiler. Additional controls check five-input gates, depth 2,200, and Fibonacci DAG sharing; malformed records, two live children and extra primary restrictions return OPEN. These finite checks validate the tested proofs; the uniform inference bound follows from the construction.

Gate-basis and parameterized-boundary checks.

`check_decoder_complexity_boundary.py` checks Theorems 7.5–7.6 through independent graph and truth-table oracles. It exhausts all labeled graphs with edges only between color classes of sizes (1), (2), (1,1), (1,2), (2,2), (1,1,1), (1,2,2), and (2,2,2): 4,384 graphs in total. For every graph it tests every target-containing eligible set, giving 270,702 graph/set pairs. All detecting sets match the clique bijection. The existing bounded-AND algorithm agrees in 8,768 decisions across one and two buses.

A further 146 decisions check the ordinary-to- colored Clique reduction. The suite also checks all seven proper gate bases on 1,680 generated decisions against 29,328 reachable truth-table rows, plus explicit boundary and rejection controls.

On 24 native controller CNFs, MiniSat and Glucose agree with the graph oracle in every case (16 SAT, 8 UNSAT). Each solver’s SAT model satisfies every original clause, and every supplied-DAG SAT witness also has a clause-checked native realization with its faulty qualifiers fixed. The native proof compiler’s guarded scope is unchanged. These finite checks validate the constructions and algorithms; the hardness and asymptotic bounds follow from the reductions above.

Appendix D. Compatible paths and zero constraints

This appendix studies the simultaneous cofactor test in Theorem 7.9. All algorithms receive their guarded DAGs and constraints explicitly.

D.1. Limits of two immediate shortcuts

For $F = (s \wedge \neg p) \vee (s \wedge \neg q \wedge \neg t)$, the set $K = \{s, p\}$ distinguishes the target cofactors, but the empty set and every non-target singleton do not. The greatest zero model of $F_1 = s \wedge \neg p$ sets all of s, p, q true, where F_0 is also zero. Thus neither singleton testing nor evaluating only the greatest zero model decides the simultaneous condition.

Proposition D.1 (an exact dual-Horn projection obstruction). A stuck-at discrepancy relation of an unshared guarded formula need not have a dual-Horn CNF representation on the same channel variables, even with existential auxiliary variables.

Proof. Take

$$\begin{aligned} Q = & (s \wedge \neg p \wedge \neg q) \\ & \vee (s \wedge \neg p \wedge \neg t) \\ & \vee (s \wedge \neg q \wedge \neg t). \end{aligned}$$

Its discrepancy relation is $s \wedge (p \oplus q)$. The sets $\{s, p\}$ and $\{s, q\}$ belong to it but their union does not. Models of a dual-Horn formula are closed under coordinatewise union: for a clause with a negative literal, a union making its variable true contains a constituent model making that variable true, which supplies a positive literal. Positive-only clauses are preserved as well. Existential projection preserves this property by taking the union of two satisfying extensions. The discrepancy relation therefore has no such representation. $\square$

The obstruction concerns an exact representation of all witnesses on these coordinates. It does not exclude decision reductions, several Horn queries, or other algorithms.

D.2. Branching on multi-positive zero clauses

Let C be a guarded DAG, with target value v on the true side, and let Z be an explicitly supplied dual-Horn system on the channels and private auxiliary variables, with its target fixed to $1 - v$. Compatibility asks for a non-target channel set K such that $C_v(K) = 1$ and K extends to a model of Z ; nonemptiness of K may additionally be required. The two copies share only the non-target channels. For detection use the opposite cofactor’s exact zero system as Z .

Let G include the channel count, all nodes and incidences of C , and all variables and literal incidences of Z . Normalize duplicate literals and remove tautological clauses. Let k count its clauses with at least two positive literals, and let their positive-literal counts be $d_1, \dots, d_k$. Set $D = \prod_i d_i$, with empty product one. Auxiliary activity literals count toward these parameters.

Theorem D.2 (few multi-positive clauses). Compatibility is decidable, with a witness, in $O(D(n+1)G)$ time and $O(G)$ working words. If every $d_i \leq d$, this is $O(d^k(n+1)G)$. For full single-output stuck-at detection, the costs of the two directions add.

Proof. An empty clause immediately rejects. For each multi-positive clause use the identity

$$(\neg a \vee b_1 \vee \dots \vee b_d) = \bigvee_j (\neg a \vee b_j).$$

For a positive-only clause use the corresponding choice of a positive unit. Distributing the choices across clauses expresses the system as an existential choice of D strengthened systems. A branch implies Z . Every model of Z satisfies a branch: choose a true positive when a is true, and any positive when a is false. Keeping the negative literal is essential in the latter case. Stream these choices rather than materializing their disjunction.

Every strengthened system consists of units and implications $\neg a \vee b$. Draw an edge $b \rightarrow a$ for propagation of false values. Let S contain negative-unit variables and R positive-unit variables, including the opposite target fixing. Write $\text{Reach}(X)$ for the vertices reachable from X , including X . Added negative units B and positive units H are consistent exactly when

$$\text{Reach}(S \cup B) \cap (R \cup H) = \emptyset.$$

Necessity follows by propagation; for sufficiency set exactly the complement of the reachability set true. This satisfies every implication and unit and is the greatest model. Reject any branch with $\text{Reach}(S) \cap R$ nonempty.

For a candidate non-target head s , take $H = \{s\}$. Reject it if the base closure reaches a required positive. A reverse traversal from $R \cup H$ computes all variables U whose falsification would force a required positive false. Delete every guard node containing a non-target guard in U . Treat target guards according to v , independently of the opposite target coordinate. A topological traversal then determines whether the root has a path to a leaf labelled s .

This deletion is exact because graph reachability is additive: $\text{Reach}(S \cup B) = \text{Reach}(S) \cup \bigcup_{j \in B} \text{Reach}(\{j\})$. Any surviving path's guard units are consistent with the required head and the binary system. Recover the path, propagate its negative units, and take the greatest model to construct K .

A true target leaf is a constant terminal, permitted only when $v = 1$. If nonemptiness is required, for each non-target channel s require it true and allow either leaf s or that constant terminal. Every nonempty constant-terminal witness supplies such an s . If empty K is permitted, add one pass with H empty and only the constant terminal allowed. Thus at most $n+1$ traversals suffice per branch. An existing compatible model selects a strengthened branch and a support path surviving the appropriate pass, proving completeness. Soundness follows from the recovered path and its model of the strengthened system.

Each traversal and witness reconstruction costs $O(G)$. Branches and heads are processed sequentially, with $O(G)$ working storage. $\square$

This is fixed-parameter tractable in (k, d) , and in k for fixed d . With unbounded $d \leq G$ it supplies an XP bound in k . Equivalently, $r = \sum_i \lceil \log_2 d_i \rceil$ gives the bound $O(2^r(n+1)G)$. Nonemptiness is handled by candidate enumeration and need not be charged as an extra wide clause. For two cofactors with products D_0, D_1 , full detection costs $O((D_0 + D_1)(n+1)G)$; a cheap direction alone is insufficient when it finds no witness.

The case $k = 0$ is a binary implication system with units, a stricter condition than dual-Horn 2-CNF, which can contain positive clauses of size two. It is also stricter than unary channel guards: the zero encoding of a unary guard is $\neg a_{\text{child}} \vee h_{\text{guard}} \vee a_{\text{parent}}$, with two positive literals. For example, requiring $(s \wedge \neg p) \wedge \neg q$ to be zero forces s false when both p and q are false, but not when only one is false. Unary guards therefore do not imply additive closure.

AI assistance

OpenAI Codex assisted with mathematical exploration, Lean formalization, and manuscript preparation. The author reviewed the AI-assisted material and takes full responsibility for the manuscript.

References

- [Sha26a] I. A. Shaik. Exact fault images in priority controllers. Lean 4 formalization, Palomar Registry, PALOMAR-2026-10-19-000010, version 1, 2026. [Immutable registry record](#).
- [Sha26b] I. A. Shaik. Guarded detection and constrained eligible-set extensions. Lean 4 formalization, Palomar Registry, PALOMAR-2026-10-19-000014, version 1, 2026. [Immutable registry record](#).
- [Sha26c] I. A. Shaik. Canonical streaming enumeration of eligible detections. Lean 4 formalization, Palomar Registry, PALOMAR-2026-10-19-000016, version 2, 2026. [Immutable registry record](#).
- [Sha26d] I. A. Shaik. Forest detection and attained healthy-zero maximum weight. Lean 4 formalization, Palomar Registry, PALOMAR-2026-10-20-000013, version 1, 2026. [Immutable registry record](#).
- [Sha26e] I. A. Shaik. Dual-Horn models and vertex covers in priority controllers. Lean 4 formalization, Palomar Registry, PALOMAR-2026-10-20-000015, version 1, 2026. [Immutable registry record](#).
- [Sha26f] I. A. Shaik. Critical recovery, query lower bounds and binary-native width. Lean 4 formalization, Palomar Registry, PALOMAR-2026-10-22-000011, version 1, 2026. [Immutable registry record](#).
- [Sha26g] I. A. Shaik. Bounded conjunction and proper-basis algorithms. Lean 4 formalization, Palomar Registry, PALOMAR-2026-10-21-000014, version 1, 2026. [Immutable registry record](#).
- [Sha26h] I. A. Shaik. Ordinary resolution for native encodings and observation wrappers. Lean 4 formalization, Palomar Registry, PALOMAR-2026-10-21-000016, version 1, 2026. [Immutable registry record](#).
- [BCMS15] S. Bova, F. Capelli, S. Mengel, F. Slivovsky. A strongly exponential separation of DNNFs from CNF formulas. arXiv:1411.1995v3, revised 19 February 2015. [Author paper](#), [Theorems 4–5](#).
- [CAB90] S. Chakradhar, V. Agrawal, M. Bushnell. Polynomial time solvable fault detection problems. FTCS 1990.

- [CHKX06] J. Chen, X. Huang, I. A. Kanj, G. Xia. Strong computational lower bounds via parameterized complexity. *JCSS* 72(8), 1346–1367, 2006. [Author manuscript](#), [Theorems 4.5 and 5.5](#).
- [DG84] W. Dowling, J. Gallier. Linear-time algorithms for testing the satisfiability of propositional Horn formulae. *J. Logic Programming* 1(3), 1984.
- [DGS14] B. Dilkina, C. P. Gomes, A. Sabharwal. Tradeoffs in the complexity of backdoors to satisfiability: dynamic sub-solvers and learning during search. *Annals of Mathematics and Artificial Intelligence* 70, 399–431, 2014. [Paper](#).
- [FL91] F. J. Ferguson, T. Larrabee. Test pattern generation for realistic bridge faults in CMOS ICs. *ITC* 1991.
- [FT82] H. Fujiwara, S. Toida. The complexity of fault detection problems for combinational logic circuits. *IEEE Trans. Computers* C-31(6), 1982.
- [Fri73] A. D. Friedman. Easily testable iterative systems. *IEEE Trans. Computers* C-22(12), 1973.
- [Fuj90] H. Fujiwara. Computational complexity of controllability/observability problems for combinational circuits. *IEEE Trans. Computers* 39(6), 1990.
- [GHR95] R. Greenlaw, H. J. Hoover, W. L. Ruzzo. *Limits to Parallel Computation: P-Completeness Theory*. Oxford University Press, 1995. [Author copy](#), [Theorem 6.2.2 and proof](#).
- [Gol77] L. M. Goldschlager. The monotone and planar circuit value problems are log space complete for P. *ACM SIGACT News* 9(2), 25–29, 1977. [Paper](#).
- [HH95] M. C. Hansen, J. P. Hayes. High-level test generation using symbolic scheduling. *ITC* 1995.
- [HLW06] S. Hoory, N. Linial, A. Wigderson. Expander graphs and their applications. *Bulletin of the AMS* 43(4), 439–561, 2006. [Author paper](#), [Construction 8.1 and Theorem 8.2](#).
- [HYH99] M. C. Hansen, H. Yalcin, J. P. Hayes. Unveiling the ISCAS-85 benchmarks: a case study in reverse engineering. *IEEE Design & Test* 16(3), 1999. [Author behavioral model](#).
- [IA96] M. A. Iyer, M. Abramovici. FIRE: a fault-independent combinational redundancy identification algorithm. *IEEE Trans. VLSI* 4(2), 1996.
- [IMS19] A. Ignatiev, A. Morgado, J. Marques-Silva. RC2: An efficient MaxSAT solver. *Journal on Satisfiability, Boolean Modeling and Computation* 11(1), 53–64, 2019. [Implementation and documentation](#).
- [IS75] O. Ibarra, S. Sahni. Polynomially complete fault detection problems. *IEEE Trans. Computers* C-24(3), 1975.
- [Ima21] G. E. Imaz. The Horn non-clausal class and its polynomiality. *arXiv:2108.13744*, 2021.
- [Kar72] R. M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, 85–103, 1972. [Paper](#).
- [Kin92] N. G. Kinnersley. The vertex separation number of a graph equals its path-width. *IPL* 42(6), 1992.
- [Lar92] T. Larrabee. Test pattern generation using Boolean satisfiability. *IEEE Trans. CAD* 11(1), 1992.
- [Li05] J.-F. Li. Testing priority address encoder faults of content addressable memories. *ITC* 2005.
- [OSS24] S. Ordyniak, A. Schidler, S. Szeider. Backdoor DNFs. *JCSS* 144, Article 103547, 2024. [Published paper](#).
- [PB83] J. S. Provan, M. O. Ball. The complexity of counting cuts and of computing the probability that a graph is connected. *SIAM Journal on Computing* 12(4), 777–788, 1983. [Paper](#).
- [PCK99] M. R. Prasad, P. Chong, K. Keutzer. Why is ATPG easy? *DAC* 1999.

- [PD11] K. Pipatsrisawat, A. Darwiche. On the power of clause-learning SAT solvers as resolution engines. *AIJ* 175, 2011.
- [Sha26B] I. A. Shaik. Guarded Reachability: Algorithms, Conditional Bounds, and Selector Costs. Manuscript, 2026.
- [Sha26C] I. A. Shaik. Explicit and Compact Guarded Factors: Interaction Graphs and Shared Interfaces. Manuscript, 2026.
- [Sha26S] I. A. Shaik. Structural SAT Algorithms and Resolution: Companion Results. Technical note, 2026.
- [Sha26U] I. A. Shaik. What Decides SATLIB? A Certified Structural Audit and an Exact Tractable Family of Priority-Controller Fault Formulas. Research manuscript, 2026.
- [WHH14] N. Wetzler, M. J. H. Heule, W. A. Hunt, Jr. DRAT-trim: Efficient checking and trimming using expressive clausal proofs. *SAT 2014*, LNCS 8561, 422–429. [Author paper](#), [Sections 1–3](#).