```
// Copilot.go
// 29 July 2025 @ 1830 hrs
// Separated Search/Simulation Phase Implementation
// Based on rudd_Large_070925.go with phase separation and enhanced UI

package main

import (
	"fmt"
	"strings"

	"github.com/dalzilio/rudd"
)

type (
	nd = rudd.Node
	g  struct {
		si, out_4, out_2, out_1, ns nd
	}
	S []g
	// Structure to store accumulated S/I sequence
	SISequence struct {
		entries []string
	}
	// Structure for simulation results
	SimResult struct {
		si        string
		outputs   string
		nextState string
	}
	// Add a structure to store S/I mappings for user selection
	SIMapping struct {
		fp int
		ns nd
	}
)

var (
	// Global variables for phase separation
	accumulatedSI                SISequence
	originalFaultA               string
	first, ns16h, ns8h, ns4h, ns2h, ns1h = true, false, false, false, false, false
)

func main() {

	// RUDD SETUP
============================================================
	//
================================================================

	bdd, _ := rudd.New(8, rudd.Nodesize(10000), rudd.Cachesize(3000))

	nd128 := bdd.Ithvar(7)
```

```
        nd64 := bdd.Ithvar(6)
        nd32 := bdd.Ithvar(5)
        nd16 := bdd.Ithvar(4)
        nd8 := bdd.Ithvar(3)
        nd4 := bdd.Ithvar(2)
        nd2 := bdd.Ithvar(1)
        nd1 := bdd.Ithvar(0)

        // Define the logical operations on BDD nodes
        not := bdd.Not
        and := bdd.And
        or := bdd.Or
        eq := bdd.Equal

        null := bdd.False()

        // COMMON SETUP
================================================
        //
================================================================

        ps16 := nd128
        ps8 := nd64
        ps4 := nd32
        ps2 := nd16
        ps1 := nd8
        in4 := nd4
        in2 := nd2
        in1 := nd1

        nps16 := not(ps16)
        nps8 := not(ps8)
        nps4 := not(ps4)
        nps2 := not(ps2)
        nps1 := not(ps1)
        nin4 := not(in4)
        nin2 := not(in2)
        nin1 := not(in1)

        s0 := and(nps16, and(nps8, and(nps4, and(nps2, nps1))))
        s1 := and(nps16, and(nps8, and(nps4, and(nps2, ps1))))
        s2 := and(nps16, and(nps8, and(nps4, and(ps2, nps1))))
        s3 := and(nps16, and(nps8, and(nps4, and(ps2, ps1))))
        s4 := and(nps16, and(nps8, and(ps4, and(nps2, nps1))))
        s5 := and(nps16, and(nps8, and(ps4, and(nps2, ps1))))
        s6 := and(nps16, and(nps8, and(ps4, and(ps2, nps1))))
        s7 := and(nps16, and(nps8, and(ps4, and(ps2, ps1))))
        s8 := and(nps16, and(ps8, and(nps4, and(nps2, nps1))))
        s9 := and(nps16, and(ps8, and(nps4, and(nps2, ps1))))
        s10 := and(nps16, and(ps8, and(nps4, and(ps2, nps1))))
        s11 := and(nps16, and(ps8, and(nps4, and(ps2, ps1))))
        s12 := and(nps16, and(ps8, and(ps4, and(nps2, nps1))))
        s13 := and(nps16, and(ps8, and(ps4, and(nps2, ps1))))
        s14 := and(nps16, and(ps8, and(ps4, and(ps2, nps1))))
```

```
s15 := and(nps16, and(ps8, and(ps4, and(ps2, ps1))))
s16 := and(ps16, and(nps8, and(nps4, and(nps2, nps1))))
s17 := and(ps16, and(nps8, and(nps4, and(nps2, ps1))))
s18 := and(ps16, and(nps8, and(nps4, and(ps2, nps1))))
s19 := and(ps16, and(nps8, and(nps4, and(ps2, ps1))))
s20 := and(ps16, and(nps8, and(ps4, and(nps2, nps1))))
s21 := and(ps16, and(nps8, and(ps4, and(nps2, ps1))))
s22 := and(ps16, and(nps8, and(ps4, and(ps2, nps1))))
s23 := and(ps16, and(nps8, and(ps4, and(ps2, ps1))))
s24 := and(ps16, and(ps8, and(nps4, and(nps2, nps1))))
s25 := and(ps16, and(ps8, and(nps4, and(nps2, ps1))))
s26 := and(ps16, and(ps8, and(nps4, and(ps2, nps1))))
s27 := and(ps16, and(ps8, and(nps4, and(ps2, ps1))))
s28 := and(ps16, and(ps8, and(ps4, and(nps2, nps1))))
s29 := and(ps16, and(ps8, and(ps4, and(nps2, ps1))))
s30 := and(ps16, and(ps8, and(ps4, and(ps2, nps1))))
s31 := and(ps16, and(ps8, and(ps4, and(ps2, ps1))))

i0 := and(nin4, and(nin2, nin1))
i1 := and(nin4, and(nin2, in1))
i2 := and(nin4, and(in2, nin1))
i3 := and(nin4, and(in2, in1))
i4 := and(in4, and(nin2, nin1))
i5 := and(in4, and(nin2, in1))
i6 := and(in4, and(in2, nin1))
i7 := and(in4, and(in2, in1))

s0i0 := and(s0, i0)
s0i1 := and(s0, i1)
s0i2 := and(s0, i2)
s0i3 := and(s0, i3)
s0i4 := and(s0, i4)
s0i5 := and(s0, i5)
s0i6 := and(s0, i6)
s0i7 := and(s0, i7)
s1i0 := and(s1, i0)
s1i1 := and(s1, i1)
s1i2 := and(s1, i2)
s1i3 := and(s1, i3)
s1i4 := and(s1, i4)
s1i5 := and(s1, i5)
s1i6 := and(s1, i6)
s1i7 := and(s1, i7)
s2i0 := and(s2, i0)
s2i1 := and(s2, i1)
s2i2 := and(s2, i2)
s2i3 := and(s2, i3)
s2i4 := and(s2, i4)
s2i5 := and(s2, i5)
s2i6 := and(s2, i6)
s2i7 := and(s2, i7)
s3i0 := and(s3, i0)
s3i1 := and(s3, i1)
s3i2 := and(s3, i2)
```

```
s3i3 := and(s3, i3)
s3i4 := and(s3, i4)
s3i5 := and(s3, i5)
s3i6 := and(s3, i6)
s3i7 := and(s3, i7)
s4i0 := and(s4, i0)
s4i1 := and(s4, i1)
s4i2 := and(s4, i2)
s4i3 := and(s4, i3)
s4i4 := and(s4, i4)
s4i5 := and(s4, i5)
s4i6 := and(s4, i6)
s4i7 := and(s4, i7)
s5i0 := and(s5, i0)
s5i1 := and(s5, i1)
s5i2 := and(s5, i2)
s5i3 := and(s5, i3)
s5i4 := and(s5, i4)
s5i5 := and(s5, i5)
s5i6 := and(s5, i6)
s5i7 := and(s5, i7)
s6i0 := and(s6, i0)
s6i1 := and(s6, i1)
s6i2 := and(s6, i2)
s6i3 := and(s6, i3)
s6i4 := and(s6, i4)
s6i5 := and(s6, i5)
s6i6 := and(s6, i6)
s6i7 := and(s6, i7)
s7i0 := and(s7, i0)
s7i1 := and(s7, i1)
s7i2 := and(s7, i2)
s7i3 := and(s7, i3)
s7i4 := and(s7, i4)
s7i5 := and(s7, i5)
s7i6 := and(s7, i6)
s7i7 := and(s7, i7)
s8i0 := and(s8, i0)
s8i1 := and(s8, i1)
s8i2 := and(s8, i2)
s8i3 := and(s8, i3)
s8i4 := and(s8, i4)
s8i5 := and(s8, i5)
s8i6 := and(s8, i6)
s8i7 := and(s8, i7)
s9i0 := and(s9, i0)
s9i1 := and(s9, i1)
s9i2 := and(s9, i2)
s9i3 := and(s9, i3)
s9i4 := and(s9, i4)
s9i5 := and(s9, i5)
s9i6 := and(s9, i6)
s9i7 := and(s9, i7)
s10i0 := and(s10, i0)
```

s10i1 := and(s10, i1)
s10i2 := and(s10, i2)
s10i3 := and(s10, i3)
s10i4 := and(s10, i4)
s10i5 := and(s10, i5)
s10i6 := and(s10, i6)
s10i7 := and(s10, i7)
s11i0 := and(s11, i0)
s11i1 := and(s11, i1)
s11i2 := and(s11, i2)
s11i3 := and(s11, i3)
s11i4 := and(s11, i4)
s11i5 := and(s11, i5)
s11i6 := and(s11, i6)
s11i7 := and(s11, i7)
s12i0 := and(s12, i0)
s12i1 := and(s12, i1)
s12i2 := and(s12, i2)
s12i3 := and(s12, i3)
s12i4 := and(s12, i4)
s12i5 := and(s12, i5)
s12i6 := and(s12, i6)
s12i7 := and(s12, i7)
s13i0 := and(s13, i0)
s13i1 := and(s13, i1)
s13i2 := and(s13, i2)
s13i3 := and(s13, i3)
s13i4 := and(s13, i4)
s13i5 := and(s13, i5)
s13i6 := and(s13, i6)
s13i7 := and(s13, i7)
s14i0 := and(s14, i0)
s14i1 := and(s14, i1)
s14i2 := and(s14, i2)
s14i3 := and(s14, i3)
s14i4 := and(s14, i4)
s14i5 := and(s14, i5)
s14i6 := and(s14, i6)
s14i7 := and(s14, i7)
s15i0 := and(s15, i0)
s15i1 := and(s15, i1)
s15i2 := and(s15, i2)
s15i3 := and(s15, i3)
s15i4 := and(s15, i4)
s15i5 := and(s15, i5)
s15i6 := and(s15, i6)
s15i7 := and(s15, i7)
s16i0 := and(s16, i0)
s16i1 := and(s16, i1)
s16i2 := and(s16, i2)
s16i3 := and(s16, i3)
s16i4 := and(s16, i4)
s16i5 := and(s16, i5)
s16i6 := and(s16, i6)

s16i7 := and(s16, i7)
s17i0 := and(s17, i0)
s17i1 := and(s17, i1)
s17i2 := and(s17, i2)
s17i3 := and(s17, i3)
s17i4 := and(s17, i4)
s17i5 := and(s17, i5)
s17i6 := and(s17, i6)
s17i7 := and(s17, i7)
s18i0 := and(s18, i0)
s18i1 := and(s18, i1)
s18i2 := and(s18, i2)
s18i3 := and(s18, i3)
s18i4 := and(s18, i4)
s18i5 := and(s18, i5)
s18i6 := and(s18, i6)
s18i7 := and(s18, i7)
s19i0 := and(s19, i0)
s19i1 := and(s19, i1)
s19i2 := and(s19, i2)
s19i3 := and(s19, i3)
s19i4 := and(s19, i4)
s19i5 := and(s19, i5)
s19i6 := and(s19, i6)
s19i7 := and(s19, i7)
s20i0 := and(s20, i0)
s20i1 := and(s20, i1)
s20i2 := and(s20, i2)
s20i3 := and(s20, i3)
s20i4 := and(s20, i4)
s20i5 := and(s20, i5)
s20i6 := and(s20, i6)
s20i7 := and(s20, i7)
s21i0 := and(s21, i0)
s21i1 := and(s21, i1)
s21i2 := and(s21, i2)
s21i3 := and(s21, i3)
s21i4 := and(s21, i4)
s21i5 := and(s21, i5)
s21i6 := and(s21, i6)
s21i7 := and(s21, i7)
s22i0 := and(s22, i0)
s22i1 := and(s22, i1)
s22i2 := and(s22, i2)
s22i3 := and(s22, i3)
s22i4 := and(s22, i4)
s22i5 := and(s22, i5)
s22i6 := and(s22, i6)
s22i7 := and(s22, i7)
s23i0 := and(s23, i0)
s23i1 := and(s23, i1)
s23i2 := and(s23, i2)
s23i3 := and(s23, i3)
s23i4 := and(s23, i4)

```
s23i5 := and(s23, i5)
s23i6 := and(s23, i6)
s23i7 := and(s23, i7)
s24i0 := and(s24, i0)
s24i1 := and(s24, i1)
s24i2 := and(s24, i2)
s24i3 := and(s24, i3)
s24i4 := and(s24, i4)
s24i5 := and(s24, i5)
s24i6 := and(s24, i6)
s24i7 := and(s24, i7)
s25i0 := and(s25, i0)
s25i1 := and(s25, i1)
s25i2 := and(s25, i2)
s25i3 := and(s25, i3)
s25i4 := and(s25, i4)
s25i5 := and(s25, i5)
s25i6 := and(s25, i6)
s25i7 := and(s25, i7)
s26i0 := and(s26, i0)
s26i1 := and(s26, i1)
s26i2 := and(s26, i2)
s26i3 := and(s26, i3)
s26i4 := and(s26, i4)
s26i5 := and(s26, i5)
s26i6 := and(s26, i6)
s26i7 := and(s26, i7)
s27i0 := and(s27, i0)
s27i1 := and(s27, i1)
s27i2 := and(s27, i2)
s27i3 := and(s27, i3)
s27i4 := and(s27, i4)
s27i5 := and(s27, i5)
s27i6 := and(s27, i6)
s27i7 := and(s27, i7)
s28i0 := and(s28, i0)
s28i1 := and(s28, i1)
s28i2 := and(s28, i2)
s28i3 := and(s28, i3)
s28i4 := and(s28, i4)
s28i5 := and(s28, i5)
s28i6 := and(s28, i6)
s28i7 := and(s28, i7)
s29i0 := and(s29, i0)
s29i1 := and(s29, i1)
s29i2 := and(s29, i2)
s29i3 := and(s29, i3)
s29i4 := and(s29, i4)
s29i5 := and(s29, i5)
s29i6 := and(s29, i6)
s29i7 := and(s29, i7)
s30i0 := and(s30, i0)
s30i1 := and(s30, i1)
s30i2 := and(s30, i2)
```

```
s30i3 := and(s30, i3)
s30i4 := and(s30, i4)
s30i5 := and(s30, i5)
s30i6 := and(s30, i6)
s30i7 := and(s30, i7)
s31i0 := and(s31, i0)
s31i1 := and(s31, i1)
s31i2 := and(s31, i2)
s31i3 := and(s31, i3)
s31i4 := and(s31, i4)
s31i5 := and(s31, i5)
s31i6 := and(s31, i6)
s31i7 := and(s31, i7)

// ---- type changing function S/I string -> S/I nd ----
str2nd := func(f string) nd {
        mapping := map[string]nd{
                "null": null,
                // s0 with all inputs
                "s0i0": s0i0, "s0i1": s0i1, "s0i2": s0i2, "s0i3": s0i3, "s0i4": s0i4,
                "s0i5": s0i5, "s0i6": s0i6, "s0i7": s0i7,
                // s1 with all inputs
                "s1i0": s1i0, "s1i1": s1i1, "s1i2": s1i2, "s1i3": s1i3, "s1i4": s1i4,
                "s1i5": s1i5, "s1i6": s1i6, "s1i7": s1i7,
                // s2 with all inputs
                "s2i0": s2i0, "s2i1": s2i1, "s2i2": s2i2, "s2i3": s2i3, "s2i4": s2i4,
                "s2i5": s2i5, "s2i6": s2i6, "s2i7": s2i7,
                // s3 with all inputs
                "s3i0": s3i0, "s3i1": s3i1, "s3i2": s3i2, "s3i3": s3i3, "s3i4": s3i4,
                "s3i5": s3i5, "s3i6": s3i6, "s3i7": s3i7,
                // s4 with all inputs
                "s4i0": s4i0, "s4i1": s4i1, "s4i2": s4i2, "s4i3": s4i3, "s4i4": s4i4,
                "s4i5": s4i5, "s4i6": s4i6, "s4i7": s4i7,
                // s5 with all inputs
                "s5i0": s5i0, "s5i1": s5i1, "s5i2": s5i2, "s5i3": s5i3, "s5i4": s5i4,
                "s5i5": s5i5, "s5i6": s5i6, "s5i7": s5i7,
                // s6 with all inputs
                "s6i0": s6i0, "s6i1": s6i1, "s6i2": s6i2, "s6i3": s6i3, "s6i4": s6i4,
                "s6i5": s6i5, "s6i6": s6i6, "s6i7": s6i7,
                // s7 with all inputs
                "s7i0": s7i0, "s7i1": s7i1, "s7i2": s7i2, "s7i3": s7i3, "s7i4": s7i4,
                "s7i5": s7i5, "s7i6": s7i6, "s7i7": s7i7,
                // s8 with all inputs
                "s8i0": s8i0, "s8i1": s8i1, "s8i2": s8i2, "s8i3": s8i3, "s8i4": s8i4,
                "s8i5": s8i5, "s8i6": s8i6, "s8i7": s8i7,
                // s9 with all inputs
                "s9i0": s9i0, "s9i1": s9i1, "s9i2": s9i2, "s9i3": s9i3, "s9i4": s9i4,
                "s9i5": s9i5, "s9i6": s9i6, "s9i7": s9i7,
                // s10 with all inputs
                "s10i0": s10i0, "s10i1": s10i1, "s10i2": s10i2, "s10i3": s10i3,
                "s10i4": s10i4, "s10i5": s10i5, "s10i6": s10i6, "s10i7": s10i7,
                // s11 with all inputs
                "s11i0": s11i0, "s11i1": s11i1, "s11i2": s11i2, "s11i3": s11i3,
                "s11i4": s11i4, "s11i5": s11i5, "s11i6": s11i6, "s11i7": s11i7,
```

```
// s12 with all inputs
"s12i0": s12i0, "s12i1": s12i1, "s12i2": s12i2, "s12i3": s12i3,
"s12i4": s12i4, "s12i5": s12i5, "s12i6": s12i6, "s12i7": s12i7,
// s13 with all inputs
"s13i0": s13i0, "s13i1": s13i1, "s13i2": s13i2, "s13i3": s13i3,
"s13i4": s13i4, "s13i5": s13i5, "s13i6": s13i6, "s13i7": s13i7,
// s14 with all inputs
"s14i0": s14i0, "s14i1": s14i1, "s14i2": s14i2, "s14i3": s14i3,
"s14i4": s14i4, "s14i5": s14i5, "s14i6": s14i6, "s14i7": s14i7,
// s15 with all inputs
"s15i0": s15i0, "s15i1": s15i1, "s15i2": s15i2, "s15i3": s15i3,
"s15i4": s15i4, "s15i5": s15i5, "s15i6": s15i6, "s15i7": s15i7,
// s16 with all inputs
"s16i0": s16i0, "s16i1": s16i1, "s16i2": s16i2, "s16i3": s16i3,
"s16i4": s16i4, "s16i5": s16i5, "s16i6": s16i6, "s16i7": s16i7,
// s17 with all inputs
"s17i0": s17i0, "s17i1": s17i1, "s17i2": s17i2, "s17i3": s17i3,
"s17i4": s17i4, "s17i5": s17i5, "s17i6": s17i6, "s17i7": s17i7,
// s18 with all inputs
"s18i0": s18i0, "s18i1": s18i1, "s18i2": s18i2, "s18i3": s18i3,
"s18i4": s18i4, "s18i5": s18i5, "s18i6": s18i6, "s18i7": s18i7,
// s19 with all inputs
"s19i0": s19i0, "s19i1": s19i1, "s19i2": s19i2, "s19i3": s19i3,
"s19i4": s19i4, "s19i5": s19i5, "s19i6": s19i6, "s19i7": s19i7,
// s20 with all inputs
"s20i0": s20i0, "s20i1": s20i1, "s20i2": s20i2, "s20i3": s20i3,
"s20i4": s20i4, "s20i5": s20i5, "s20i6": s20i6, "s20i7": s20i7,
// s21 with all inputs
"s21i0": s21i0, "s21i1": s21i1, "s21i2": s21i2, "s21i3": s21i3,
"s21i4": s21i4, "s21i5": s21i5, "s21i6": s21i6, "s21i7": s21i7,
// s22 with all inputs
"s22i0": s22i0, "s22i1": s22i1, "s22i2": s22i2, "s22i3": s22i3,
"s22i4": s22i4, "s22i5": s22i5, "s22i6": s22i6, "s22i7": s22i7,
// s23 with all inputs
"s23i0": s23i0, "s23i1": s23i1, "s23i2": s23i2, "s23i3": s23i3,
"s23i4": s23i4, "s23i5": s23i5, "s23i6": s23i6, "s23i7": s23i7,
// s24 with all inputs
"s24i0": s24i0, "s24i1": s24i1, "s24i2": s24i2, "s24i3": s24i3,
"s24i4": s24i4, "s24i5": s24i5, "s24i6": s24i6, "s24i7": s24i7,
// s25 with all inputs
"s25i0": s25i0, "s25i1": s25i1, "s25i2": s25i2, "s25i3": s25i3,
"s25i4": s25i4, "s25i5": s25i5, "s25i6": s25i6, "s25i7": s25i7,
// s26 with all inputs
"s26i0": s26i0, "s26i1": s26i1, "s26i2": s26i2, "s26i3": s26i3,
"s26i4": s26i4, "s26i5": s26i5, "s26i6": s26i6, "s26i7": s26i7,
// s27 with all inputs
"s27i0": s27i0, "s27i1": s27i1, "s27i2": s27i2, "s27i3": s27i3,
"s27i4": s27i4, "s27i5": s27i5, "s27i6": s27i6, "s27i7": s27i7,
// s28 with all inputs
"s28i0": s28i0, "s28i1": s28i1, "s28i2": s28i2, "s28i3": s28i3,
"s28i4": s28i4, "s28i5": s28i5, "s28i6": s28i6, "s28i7": s28i7,
// s29 with all inputs
"s29i0": s29i0, "s29i1": s29i1, "s29i2": s29i2, "s29i3": s29i3,
"s29i4": s29i4, "s29i5": s29i5, "s29i6": s29i6, "s29i7": s29i7,
```

```
                // s30 with all inputs
                "s30i0": s30i0, "s30i1": s30i1, "s30i2": s30i2, "s30i3": s30i3,
                "s30i4": s30i4, "s30i5": s30i5, "s30i6": s30i6, "s30i7": s30i7,
                // s31 with all inputs
                "s31i0": s31i0, "s31i1": s31i1, "s31i2": s31i2, "s31i3": s31i3,
                "s31i4": s31i4, "s31i5": s31i5, "s31i6": s31i6, "s31i7": s31i7,
        }
        return mapping[f]
}

// ---- type changing function s nd -> s string ----
nd2str := func(sy nd) string {
        mapping := map[nd]string{
                null: "null", s0: "s0", s1: "s1", s2: "s2", s3: "s3", s4: "s4",
                s5: "s5", s6: "s6", s7: "s7", s8: "s8", s9: "s9", s10: "s10",
                s11: "s11", s12: "s12", s13: "s13", s14: "s14", s15: "s15",
                s16: "s16", s17: "s17", s18: "s18", s19: "s19", s20: "s20",
                s21: "s21", s22: "s22", s23: "s23", s24: "s24", s25: "s25",
                s26: "s26", s27: "s27", s28: "s28", s29: "s29", s30: "s30",
                s31: "s31",
        }
        return mapping[sy]
}

// allSat receives a BCD object and parses it into constituent S/Is,
// returning a list of zero or more S/Is, each of type string.
allSAT := func(f nd, str2nd func(string) nd) []string {
        // Define the states and inputs
        states := 32 // Number of states (s0 to s31)
        inputs := 8  // Number of inputs (i0 to i7)

        // Initialize the result slice
        g := []string{}

        // Iterate through all states and inputs
        for s := 0; s < states; s++ {
                for i := 0; i < inputs; i++ {
                        // Construct the state-input string (e.g., "s0i0", "s1i1", etc.)
                        si := fmt.Sprintf("s%di%d", s, i)

                        // Convert the string to the nd type using str2nd
                        ndValue := str2nd(si)

                        // Debug: Check for nil values before calling and()
                        if f == nil {
                                fmt.Printf("ERROR: f is nil at si=%s\n", si)
                                continue
                        }
                        if ndValue == nil {
                                fmt.Printf("ERROR: ndValue is nil at si=%s\n", si)
                                continue
                        }
```

```
                                // Check if the conjunction of `f` and the current state-input is not
null
                                if and(f, ndValue) != null {
                                        g = append(g, si)
                                }
                        }
                }

                return g // Return the list of S/I's in the nd function
        }

        // REAL LOGIC GATE RULE FUNCTIONS ================================
        // 2- and 3-input AND and OR gates for BDD circuit simulation
        //
================================================================

        piRule := func(s1, i1 nd) (nd, nd) {
                // computes the propagation function
                o_s := s1
                // computes the 1-set
                o_1 := i1
                return o_s, o_1
        }

        notRule := func(s1, i1 nd) (nd, nd) {
                // computes the propagation function
                o_s := s1
                // computes the 1-set
                o_1 := not(i1)
                return o_s, o_1
        }

        and2Rule := func(s1, s2, i1, i2 nd) (nd, nd) {
                // computes the propagation function
                o_s := or(and(not(i1), s1, not(i2), s2),
                        and(i2, s1, not(s2)),
                        and(i1, s2, not(s1)),
                        and(i1, i2, or(s1, s2)))
                // computes the 1-set
                o_1 := and(i1, i2)
                return o_s, o_1
        }

        or2Rule := func(s1, s2, i1, i2 nd) (nd, nd) {
                // computes the propagation function
                o_s := or(and(i1, s1, i2, s2),
                        and(not(i2), s1, not(s2)),
                        and(not(i1), s2, not(s1)),
                        and(not(i1), not(i2), or(s1, s2)))
                // computes the 1-set
                o_1 := or(i1, i2)
                return o_s, o_1
        }
```

```
	and3Rule := func(s1, s2, s3, i1, i2, i3 nd) (nd, nd) {
		s, i := and2Rule(s1, s2, i1, i2)
		o_s, o_1 := and2Rule(s, s3, i, i3)
		return o_s, o_1
	}

	or3Rule := func(s1, s2, s3, i1, i2, i3 nd) (nd, nd) {
		s, i := or2Rule(s1, s2, i1, i2)
		o_s, o_1 := or2Rule(s, s3, i, i3)
		return o_s, o_1
	}

	//
================================================================
	// Beginning of CORE functions
	//
================================================================

	// ps2ns function (from original file)
	ps2ns := func(ps16_i, ps8_i, ps4_i, ps2_i, ps1_i nd,
		fault_A string) (nd, nd, nd, nd, nd, nd, nd, nd) {

		// Receives S/Is via present-state lines ps16_s, ps8_s. ect.
		// Uses fault to activate local_fault; propagates local_fault
		// to output and next_state lines, out4_s, out2_s, out1_s,
		// ns16_s, ns8_s, etc., as S/Is.

		// Helper function to apply faults to signals
		applyFault := func(signal nd, faultSignal nd,
			fault string, faultName string) nd {
			if fault == faultName+":0" {
				return faultSignal
			}
			if fault == faultName+":1" {
				return not(faultSignal)
			}
			return signal
		}

		// A circuit-level-sorted sequence of gates
		// Such that each signal has already been defined when used

		// ---- Fault Propagation NETLIST for LARGE Circuit ----

		// level 1

		ps16_s, ps16_1 := piRule(ps16_i, ps16)
		ps16_s = applyFault(ps16_s, ps16_1, fault_A, "ps16")

		ps8_s, ps8_1 := piRule(ps8_i, ps8)
		ps8_s = applyFault(ps8_s, ps8_1, fault_A, "ps8")

		ps4_s, ps4_1 := piRule(ps4_i, ps4)
		ps4_s = applyFault(ps4_s, ps4_1, fault_A, "ps4")
```

```
ps2_s, ps2_1 := piRule(ps2_i, ps2)
ps2_s = applyFault(ps2_s, ps2_1, fault_A, "ps2")

ps1_s, ps1_1 := piRule(ps1_i, ps1)
ps1_s = applyFault(ps1_s, ps1_1, fault_A, "ps1")

in4_s, in4_1 := piRule(null, in4)
in4_s = applyFault(in4_s, in4_1, fault_A, "in4")

in2_s, in2_1 := piRule(null, in2)
in2_s = applyFault(in2_s, in2_1, fault_A, "in2")

in1_s, in1_1 := piRule(null, in1)
in1_s = applyFault(in1_s, in1_1, fault_A, "in1")

nps1_s, nps1_1 := notRule(ps1_s, ps1_1)
nps1_s = applyFault(nps1_s, nps1_1, fault_A, "nps1")

nps2_s, nps2_1 := notRule(ps2_s, ps2_1)
nps2_s = applyFault(nps2_s, nps2_1, fault_A, "nps2")

nps4_s, nps4_1 := notRule(ps4_s, ps4_1)
nps4_s = applyFault(nps4_s, nps4_1, fault_A, "nps4")

nps8_s, nps8_1 := notRule(ps8_s, ps8_1)
nps8_s = applyFault(nps8_s, nps8_1, fault_A, "nps8")

nps16_s, nps16_1 := notRule(ps16_s, ps16_1)
nps16_s = applyFault(nps16_s, nps16_1, fault_A, "nps16")

nin1_s, nin1_1 := notRule(in1_s, in1_1)
nin1_s = applyFault(nin1_s, nin1_1, fault_A, "nin1")

nin2_s, nin2_1 := notRule(in2_s, in2_1)
nin2_s = applyFault(nin2_s, nin2_1, fault_A, "nin2")

nin4_s, nin4_1 := notRule(in4_s, in4_1)
nin4_s = applyFault(nin4_s, nin4_1, fault_A, "nin4")

i7_s, i7_1 := and3Rule(in4_s, in2_s, in1_s, in4_1, in2_1, in1_1)
i7_s = applyFault(i7_s, i7_1, fault_A, "i7")

// level 2

i0_s, i0_1 := and3Rule(nin4_s, nin2_s, nin1_s, nin4_1, nin2_1, nin1_1)
i0_s = applyFault(i0_s, i0_1, fault_A, "i0")

i1_s, i1_1 := and3Rule(nin4_s, nin2_s, in1_s, nin4_1, nin2_1, in1_1)
i1_s = applyFault(i1_s, i1_1, fault_A, "i1")

i2_s, i2_1 := and3Rule(nin4_s, in2_s, nin1_s, nin4_1, in2_1, nin1_1)
i2_s = applyFault(i2_s, i2_1, fault_A, "i2")
```

```
i3_s, i3_1 := and3Rule(nin4_s, in2_s, in1_s, nin4_1, in2_1, in1_1)
i3_s = applyFault(i3_s, i3_1, fault_A, "i3")

i4_s, i4_1 := and3Rule(in4_s, nin2_s, nin1_s, in4_1, nin2_1, nin1_1)
i4_s = applyFault(i4_s, i4_1, fault_A, "i4")

i5_s, i5_1 := and3Rule(in4_s, nin2_s, in1_s, in4_1, nin2_1, in1_1)
i5_s = applyFault(i5_s, i5_1, fault_A, "i5")

i6_s, i6_1 := and3Rule(in4_s, in2_s, nin1_s, in4_1, in2_1, nin1_1)
i6_s = applyFault(i6_s, i6_1, fault_A, "i6")

ls0_s, ls0_1 := and3Rule(nps4_s, nps2_s, nps1_s, nps4_1, nps2_1, nps1_1)
ls0_s = applyFault(ls0_s, ls0_1, fault_A, "ls0")

ls1_s, ls1_1 := and3Rule(nps4_s, nps2_s, ps1_s, nps4_1, nps2_1, ps1_1)
ls1_s = applyFault(ls1_s, ls1_1, fault_A, "ls1")

ls2_s, ls2_1 := and3Rule(nps4_s, ps2_s, nps1_s, nps4_1, ps2_1, nps1_1)
ls2_s = applyFault(ls2_s, ls2_1, fault_A, "ls2")

ls3_s, ls3_1 := and3Rule(nps4_s, ps2_s, ps1_s, nps4_1, ps2_1, ps1_1)
ls3_s = applyFault(ls3_s, ls3_1, fault_A, "ls3")

ls4_s, ls4_1 := and3Rule(ps4_s, nps2_s, nps1_s, ps4_1, nps2_1, nps1_1)
ls4_s = applyFault(ls4_s, ls4_1, fault_A, "ls4")

ls5_s, ls5_1 := and3Rule(ps4_s, nps2_s, ps1_s, ps4_1, nps2_1, ps1_1)
ls5_s = applyFault(ls5_s, ls5_1, fault_A, "ls5")

ls6_s, ls6_1 := and3Rule(ps4_s, ps2_s, nps1_s, ps4_1, ps2_1, nps1_1)
ls6_s = applyFault(ls6_s, ls6_1, fault_A, "ls6")

ls7_s, ls7_1 := and3Rule(ps4_s, ps2_s, ps1_s, ps4_1, ps2_1, ps1_1)
ls7_s = applyFault(ls7_s, ls7_1, fault_A, "ls7")

ni7_s, ni7_1 := notRule(i7_s, i7_1)
ni7_s = applyFault(ni7_s, ni7_1, fault_A, "ni7")

s31_s, s31_1 := and3Rule(ps16_s, ps8_s, ls7_s, ps16_1, ps8_1, ls7_1)
s31_s = applyFault(s31_s, s31_1, fault_A, "0")

// level 3

ni0_s, ni0_1 := notRule(i0_s, i0_1)
ni0_s = applyFault(ni0_s, ni0_1, fault_A, "ni0")

ni1_s, ni1_1 := notRule(i1_s, i1_1)
ni1_s = applyFault(ni1_s, ni1_1, fault_A, "ni1")

ni2_s, ni2_1 := notRule(i2_s, i2_1)
ni2_s = applyFault(ni2_s, ni2_1, fault_A, "ni2")

ni3_s, ni3_1 := notRule(i3_s, i3_1)
```

```
ni3_s = applyFault(ni3_s, ni3_1, fault_A, "ni3")

ni5_s, ni5_1 := notRule(i5_s, i5_1)
ni5_s = applyFault(ni5_s, ni5_1, fault_A, "ni5")

ni6_s, ni6_1 := notRule(i6_s, i6_1)
ni6_s = applyFault(ni6_s, ni6_1, fault_A, "ni6")

s0_s, s0_1 := and3Rule(nps16_s, nps8_s, ls0_s, nps16_1, nps8_1, ls0_1)
s0_s = applyFault(s0_s, s0_1, fault_A, "s0")

s1_s, s1_1 := and3Rule(nps16_s, nps8_s, ls1_s, nps16_1, nps8_1, ls1_1)
s1_s = applyFault(s1_s, s1_1, fault_A, "s1")

s2_s, s2_1 := and3Rule(nps16_s, nps8_s, ls2_s, nps16_1, nps8_1, ls2_1)
s2_s = applyFault(s2_s, s2_1, fault_A, "s2")

s3_s, s3_1 := and3Rule(nps16_s, nps8_s, ls3_s, nps16_1, nps8_1, ls3_1)
s3_s = applyFault(s3_s, s3_1, fault_A, "s3")

s4_s, s4_1 := and3Rule(nps16_s, nps8_s, ls4_s, nps16_1, nps8_1, ls4_1)
s4_s = applyFault(s4_s, s4_1, fault_A, "s4")

s5_s, s5_1 := and3Rule(nps16_s, nps8_s, ls5_s, nps16_1, nps8_1, ls5_1)
s5_s = applyFault(s5_s, s5_1, fault_A, "s5")

s6_s, s6_1 := and3Rule(nps16_s, nps8_s, ls6_s, nps16_1, nps8_1, ls6_1)
s6_s = applyFault(s6_s, s6_1, fault_A, "s6")

s7_s, s7_1 := and3Rule(nps16_s, nps8_s, ls7_s, nps16_1, nps8_1, ls7_1)
s7_s = applyFault(s7_s, s7_1, fault_A, "s7")

s8_s, s8_1 := and3Rule(nps16_s, ps8_s, ls0_s, nps16_1, ps8_1, ls0_1)
s8_s = applyFault(s8_s, s8_1, fault_A, "s8")

s9_s, s9_1 := and3Rule(nps16_s, ps8_s, ls1_s, nps16_1, ps8_1, ls1_1)
s9_s = applyFault(s9_s, s9_1, fault_A, "s9")

s10_s, s10_1 := and3Rule(nps16_s, ps8_s, ls2_s, nps16_1, ps8_1, ls2_1)
s10_s = applyFault(s10_s, s10_1, fault_A, "s10")

s11_s, s11_1 := and3Rule(nps16_s, ps8_s, ls3_s, nps16_1, ps8_1, ls3_1)
s11_s = applyFault(s11_s, s11_1, fault_A, "s11")

s12_s, s12_1 := and3Rule(nps16_s, ps8_s, ls4_s, nps16_1, ps8_1, ls4_1)
s12_s = applyFault(s12_s, s12_1, fault_A, "s12")

s13_s, s13_1 := and3Rule(nps16_s, ps8_s, ls5_s, nps16_1, ps8_1, ls5_1)
s13_s = applyFault(s13_s, s13_1, fault_A, "s13")

s14_s, s14_1 := and3Rule(nps16_s, ps8_s, ls6_s, nps16_1, ps8_1, ls6_1)
s14_s = applyFault(s14_s, s14_1, fault_A, "s14")

s15_s, s15_1 := and3Rule(nps16_s, ps8_s, ls7_s, nps16_1, ps8_1, ls7_1)
```

```
s15_s = applyFault(s15_s, s15_1, fault_A, "s15")

s16_s, s16_1 := and3Rule(ps16_s, nps8_s, ls0_s, ps16_1, nps8_1, ls0_1)
s16_s = applyFault(s16_s, s16_1, fault_A, "s16")

s17_s, s17_1 := and3Rule(ps16_s, nps8_s, ls1_s, ps16_1, nps8_1, ls1_1)
s17_s = applyFault(s17_s, s17_1, fault_A, "s17")

s18_s, s18_1 := and3Rule(ps16_s, nps8_s, ls2_s, ps16_1, nps8_1, ls2_1)
s18_s = applyFault(s18_s, s18_1, fault_A, "s18")

s19_s, s19_1 := and3Rule(ps16_s, nps8_s, ls3_s, ps16_1, nps8_1, ls3_1)
s19_s = applyFault(s19_s, s19_1, fault_A, "s19")

s20_s, s20_1 := and3Rule(ps16_s, nps8_s, ls4_s, ps16_1, nps8_1, ls4_1)
s20_s = applyFault(s20_s, s20_1, fault_A, "s20")

s21_s, s21_1 := and3Rule(ps16_s, nps8_s, ls5_s, ps16_1, nps8_1, ls5_1)
s21_s = applyFault(s21_s, s21_1, fault_A, "s21")

s22_s, s22_1 := and3Rule(ps16_s, nps8_s, ls6_s, ps16_1, nps8_1, ls6_1)
s22_s = applyFault(s22_s, s22_1, fault_A, "s22")

s23_s, s23_1 := and3Rule(ps16_s, nps8_s, ls7_s, ps16_1, nps8_1, ls7_1)
s23_s = applyFault(s23_s, s23_1, fault_A, "s23")

s24_s, s24_1 := and3Rule(ps16_s, ps8_s, ls0_s, ps16_1, ps8_1, ls0_1)
s24_s = applyFault(s24_s, s24_1, fault_A, "s24")

s25_s, s25_1 := and3Rule(ps16_s, ps8_s, ls1_s, ps16_1, ps8_1, ls1_1)
s25_s = applyFault(s25_s, s25_1, fault_A, "s25")

s26_s, s26_1 := and3Rule(ps16_s, ps8_s, ls2_s, ps16_1, ps8_1, ls2_1)
s26_s = applyFault(s26_s, s26_1, fault_A, "s26")

s27_s, s27_1 := and3Rule(ps16_s, ps8_s, ls3_s, ps16_1, ps8_1, ls3_1)
s27_s = applyFault(s27_s, s27_1, fault_A, "s27")

s28_s, s28_1 := and3Rule(ps16_s, ps8_s, ls4_s, ps16_1, ps8_1, ls4_1)
s28_s = applyFault(s28_s, s28_1, fault_A, "s28")

s29_s, s29_1 := and3Rule(ps16_s, ps8_s, ls5_s, ps16_1, ps8_1, ls5_1)
s29_s = applyFault(s29_s, s29_1, fault_A, "s29")

s30_s, s30_1 := and3Rule(ps16_s, ps8_s, ls6_s, ps16_1, ps8_1, ls6_1)
s30_s = applyFault(s30_s, s30_1, fault_A, "s30")

b2_s, b2_1 := or3Rule(i5_s, i3_s, i2_s, i5_1, i3_1, i2_1)
b2_s = applyFault(b2_s, b2_1, fault_A, "b2")

b7_s, b7_1 := or2Rule(i5_s, i1_s, i5_1, i1_1)
b7_s = applyFault(b7_s, b7_1, fault_A, "b7")

c5_s, c5_1 := or2Rule(i7_s, i5_s, i7_1, i5_1)
```

```
c5_s = applyFault(c5_s, c5_1, fault_A, "c5")

c13_s, c13_1 := or2Rule(i3_s, i2_s, i3_1, i2_1)
c13_s = applyFault(c13_s, c13_1, fault_A, "c13")

e5_s, e5_1 := or2Rule(i5_s, i4_s, i5_1, i4_1)
e5_s = applyFault(e5_s, e5_1, fault_A, "e5")

e10_s, e10_1 := or3Rule(i6_s, i2_s, i0_s, i6_1, i2_1, i0_1)
e10_s = applyFault(e10_s, e10_1, fault_A, "e10")

e12_s, e12_1 := or2Rule(i6_s, i3_s, i6_1, i3_1)
e12_s = applyFault(e12_s, e12_1, fault_A, "e12")

e18_s, e18_1 := or2Rule(i6_s, i2_s, i6_1, i2_1)
e18_s = applyFault(e18_s, e18_1, fault_A, "e18")

e24_s, e24_1 := or2Rule(i7_s, i1_s, i7_1, i1_1)
e24_s = applyFault(e24_s, e24_1, fault_A, "e24")

// level 4

a1_s, a1_1 := and2Rule(s10_s, i0_s, s10_1, i0_1)
a1_s = applyFault(a1_s, a1_1, fault_A, "a1")

a2_s, a2_1 := and2Rule(s15_s, i5_s, s15_1, i5_1)
a2_s = applyFault(a2_s, a2_1, fault_A, "a2")

a3_s, a3_1 := and2Rule(s18_s, ni6_s, s18_1, ni6_1)
a3_s = applyFault(a3_s, a3_1, fault_A, "a3")

a4_s, a4_1 := or3Rule(s20_s, s21_s, s22_s, s20_1, s21_1, s22_1)
a4_s = applyFault(a4_s, a4_1, fault_A, "a4")

a5_s, a5_1 := and2Rule(s24_s, ni7_s, s24_1, ni7_1)
a5_s = applyFault(a5_s, a5_1, fault_A, "a5")

a6_s, a6_1 := or2Rule(s25_s, s26_s, s25_1, s26_1)
a6_s = applyFault(a6_s, a6_1, fault_A, "a6")

a7_s, a7_1 := or3Rule(s27_s, s28_s, s29_s, s27_1, s28_1, s29_1)
a7_s = applyFault(a7_s, a7_1, fault_A, "a7")

a9_s, a9_1 := and3Rule(s31_s, ni5_s, ni2_s, s31_1, ni5_1, ni2_1)
a9_s = applyFault(a9_s, a9_1, fault_A, "a9")

b1_s, b1_1 := and2Rule(s3_s, i2_s, s3_1, i2_1)
b1_s = applyFault(b1_s, b1_1, fault_A, "b1")

b3_s, b3_1 := and2Rule(b2_s, s7_s, b2_1, s7_1)
b3_s = applyFault(b3_s, b3_1, fault_A, "b3")

b4_s, b4_1 := and2Rule(s10_s, ni6_s, s10_1, ni6_1)
b4_s = applyFault(b4_s, b4_1, fault_A, "b4")
```

```
b5_s, b5_1 := or3Rule(s12_s, s13_s, s14_s, s12_1, s13_1, s14_1)
b5_s = applyFault(b5_s, b5_1, fault_A, "b5")

b6_s, b6_1 := and2Rule(s15_s, ni5_s, s15_1, ni5_1)
b6_s = applyFault(b6_s, b6_1, fault_A, "b6")

b8_s, b8_1 := and2Rule(s23_s, b7_s, s23_1, b7_1)
b8_s = applyFault(b8_s, b8_1, fault_A, "b8")

b9_s, b9_1 := or2Rule(s25_s, s26_s, s25_1, s26_1)
b9_s = applyFault(b9_s, b9_1, fault_A, "b9")

b10_s, b10_1 := or3Rule(s27_s, s28_s, s29_s, s27_1, s28_1, s29_1)
b10_s = applyFault(b10_s, b10_1, fault_A, "b10")

c2_s, c2_1 := and3Rule(s7_s, ni5_s, ni3_s, s7_1, ni5_1, ni3_1)
c2_s = applyFault(c2_s, c2_1, fault_A, "c2")

c3_s, c3_1 := and2Rule(s11_s, i7_s, s11_1, i7_1)
c3_s = applyFault(c3_s, c3_1, fault_A, "c3")

c4_s, c4_1 := and2Rule(s15_s, ni5_s, s15_1, ni5_1)
c4_s = applyFault(c4_s, c4_1, fault_A, "c4")

c6_s, c6_1 := and2Rule(s23_s, ni5_s, s23_1, ni5_1)
c6_s = applyFault(c6_s, c6_1, fault_A, "c6")

c8_s, c8_1 := and2Rule(s19_s, c5_s, s19_1, c5_1)
c8_s = applyFault(c8_s, c8_1, fault_A, "c8")

c14_s, c14_1 := and2Rule(c13_s, s3_s, c13_1, s3_1)
c14_s = applyFault(c14_s, c14_1, fault_A, "c14")

c15_s, c15_1 := and2Rule(s27_s, i7_s, s27_1, i7_1)
c15_s = applyFault(c15_s, c15_1, fault_A, "c15")

d1_s, d1_1 := and2Rule(s1_s, i2_s, s1_1, i2_1)
d1_s = applyFault(d1_s, d1_1, fault_A, "d1")

d2_s, d2_1 := and3Rule(s3_s, ni3_s, ni2_s, s3_1, ni3_1, ni2_1)
d2_s = applyFault(d2_s, d2_1, fault_A, "d2")

d3_s, d3_1 := and2Rule(s5_s, i0_s, s5_1, i0_1)
d3_s = applyFault(d3_s, d3_1, fault_A, "d3")

d5_s, d5_1 := and2Rule(s9_s, i2_s, s9_1, i2_1)
d5_s = applyFault(d5_s, d5_1, fault_A, "d5")

d6_s, d6_1 := and2Rule(s11_s, ni7_s, s11_1, ni7_1)
d6_s = applyFault(d6_s, d6_1, fault_A, "d6")

d7_s, d7_1 := and2Rule(s13_s, i0_s, s13_1, i0_1)
d7_s = applyFault(d7_s, d7_1, fault_A, "d7")
```

```
d9_s, d9_1 := and2Rule(s15_s, ni5_s, s15_1, ni5_1)
d9_s = applyFault(d9_s, d9_1, fault_A, "d9")

d10_s, d10_1 := and2Rule(s17_s, i2_s, s17_1, i2_1)
d10_s = applyFault(d10_s, d10_1, fault_A, "d10")

d11_s, d11_1 := and2Rule(s19_s, ni7_s, s19_1, ni7_1)
d11_s = applyFault(d11_s, d11_1, fault_A, "d11")

d12_s, d12_1 := and2Rule(s21_s, i0_s, s21_1, i0_1)
d12_s = applyFault(d12_s, d12_1, fault_A, "d12")

d13_s, d13_1 := and3Rule(s23_s, ni5_s, ni1_s, s23_1, ni5_1, ni1_1)
d13_s = applyFault(d13_s, d13_1, fault_A, "d13")

d14_s, d14_1 := and2Rule(s25_s, i2_s, s25_1, i2_1)
d14_s = applyFault(d14_s, d14_1, fault_A, "d14")

d15_s, d15_1 := and2Rule(s29_s, i0_s, s29_1, i0_1)
d15_s = applyFault(d15_s, d15_1, fault_A, "d15")

d27_s, d27_1 := and2Rule(s27_s, ni7_s, s27_1, ni7_1)
d27_s = applyFault(d27_s, d27_1, fault_A, "d27")

e1_s, e1_1 := and2Rule(s0_s, i1_s, s0_1, i1_1)
e1_s = applyFault(e1_s, e1_1, fault_A, "e1")

e2_s, e2_1 := and2Rule(s1_s, ni2_s, s1_1, ni2_1)
e2_s = applyFault(e2_s, e2_1, fault_A, "e2")

e3_s, e3_1 := and2Rule(s2_s, i2_s, s2_1, i2_1)
e3_s = applyFault(e3_s, e3_1, fault_A, "e3")

e4_s, e4_1 := and3Rule(s3_s, ni3_s, ni2_s, s3_1, ni3_1, ni2_1)
e4_s = applyFault(e4_s, e4_1, fault_A, "e4")

e6_s, e6_1 := and2Rule(s5_s, ni0_s, s5_1, ni0_1)
e6_s = applyFault(e6_s, e6_1, fault_A, "e6")

e7_s, e7_1 := and2Rule(s6_s, i7_s, s6_1, i7_1)
e7_s = applyFault(e7_s, e7_1, fault_A, "e7")

e8_s, e8_1 := and2Rule(s8_s, i1_s, s8_1, i1_1)
e8_s = applyFault(e8_s, e8_1, fault_A, "e8")

e9_s, e9_1 := and2Rule(s9_s, ni2_s, s9_1, ni2_1)
e9_s = applyFault(e9_s, e9_1, fault_A, "e9")

e11_s, e11_1 := and2Rule(s11_s, ni7_s, s11_1, ni7_1)
e11_s = applyFault(e11_s, e11_1, fault_A, "e11")

e13_s, e13_1 := and2Rule(s13_s, ni0_s, s13_1, ni0_1)
e13_s = applyFault(e13_s, e13_1, fault_A, "e13")
```

```
e14_s, e14_1 := and2Rule(s14_s, i7_s, s14_1, i7_1)
e14_s = applyFault(e14_s, e14_1, fault_A, "e14")

e15_s, e15_1 := and2Rule(s15_s, ni5_s, s15_1, ni5_1)
e15_s = applyFault(e15_s, e15_1, fault_A, "e15")

e16_s, e16_1 := and2Rule(s16_s, i1_s, s16_1, i1_1)
e16_s = applyFault(e16_s, e16_1, fault_A, "e16")

e17_s, e17_1 := and2Rule(s17_s, ni2_s, s17_1, ni2_1)
e17_s = applyFault(e17_s, e17_1, fault_A, "e17")

e19_s, e19_1 := and2Rule(s19_s, ni7_s, s19_1, ni7_1)
e19_s = applyFault(e19_s, e19_1, fault_A, "e19")

e20_s, e20_1 := and2Rule(s20_s, e12_s, s20_1, e12_1)
e20_s = applyFault(e20_s, e20_1, fault_A, "e20")

e21_s, e21_1 := and2Rule(s21_s, ni0_s, s21_1, ni0_1)
e21_s = applyFault(e21_s, e21_1, fault_A, "e21")

e22_s, e22_1 := and2Rule(s22_s, i7_s, s22_1, i7_1)
e22_s = applyFault(e22_s, e22_1, fault_A, "e22")

e23_s, e23_1 := and2Rule(s23_s, ni5_s, s23_1, ni5_1)
e23_s = applyFault(e23_s, e23_1, fault_A, "e23")

e25_s, e25_1 := and2Rule(s25_s, ni2_s, s25_1, ni2_1)
e25_s = applyFault(e25_s, e25_1, fault_A, "e25")

e26_s, e26_1 := and2Rule(s26_s, i2_s, s26_1, i2_1)
e26_s = applyFault(e26_s, e26_1, fault_A, "e26")

e27_s, e27_1 := and2Rule(s27_s, ni7_s, s27_1, ni7_1)
e27_s = applyFault(e27_s, e27_1, fault_A, "e27")

e28_s, e28_1 := and2Rule(s28_s, e12_s, s28_1, e12_1)
e28_s = applyFault(e28_s, e28_1, fault_A, "e28")

e29_s, e29_1 := and2Rule(s29_s, ni0_s, s29_1, ni0_1)
e29_s = applyFault(e29_s, e29_1, fault_A, "e29")

e30_s, e30_1 := and2Rule(s30_s, i7_s, s30_1, i7_1)
e30_s = applyFault(e30_s, e30_1, fault_A, "e30")

e31_s, e31_1 := and2Rule(s4_s, e5_s, s4_1, e5_1)
e31_s = applyFault(e31_s, e31_1, fault_A, "e31")

e32_s, e32_1 := and2Rule(s10_s, e10_s, s10_1, e10_1)
e32_s = applyFault(e32_s, e32_1, fault_A, "e32")

e33_s, e33_1 := and2Rule(s12_s, e12_s, s12_1, e12_1)
e33_s = applyFault(e33_s, e33_1, fault_A, "e33")
```

```
e34_s, e34_1 := and2Rule(s18_s, e18_s, s18_1, e18_1)
e34_s = applyFault(e34_s, e34_1, fault_A, "e34")

e35_s, e35_1 := and2Rule(s24_s, e24_s, s24_1, e24_1)
e35_s = applyFault(e35_s, e35_1, fault_A, "e35")

f1_s, f1_1 := and2Rule(s12_s, i5_s, s12_1, i5_1)
f1_s = applyFault(f1_s, f1_1, fault_A, "f1")

f2_s, f2_1 := and2Rule(s27_s, i4_s, s27_1, i4_1)
f2_s = applyFault(f2_s, f2_1, fault_A, "f2")

f3_s, f3_1 := and2Rule(s15_s, i0_s, s15_1, i0_1)
f3_s = applyFault(f3_s, f3_1, fault_A, "f3")

f4_s, f4_1 := and2Rule(s27_s, i2_s, s27_1, i2_1)
f4_s = applyFault(f4_s, f4_1, fault_A, "f4")

f5_s, f5_1 := and2Rule(s0_s, i7_s, s0_1, i7_1)
f5_s = applyFault(f5_s, f5_1, fault_A, "f5")

f6_s, f6_1 := and2Rule(s27_s, i1_s, s27_1, i1_1)
f6_s = applyFault(f6_s, f6_1, fault_A, "f6")

// level 5

a8_s, a8_1 := or2Rule(a7_s, s30_s, a7_1, s30_1)
a8_s = applyFault(a8_s, a8_1, fault_A, "a8")

a10_s, a10_1 := or3Rule(a1_s, a2_s, s16_s, a1_1, a2_1, s16_1)
a10_s = applyFault(a10_s, a10_1, fault_A, "a10")

b11_s, b11_1 := or2Rule(b10_s, s30_s, b10_1, s30_1)
b11_s = applyFault(b11_s, b11_1, fault_A, "b11")

b12_s, b12_1 := or3Rule(b1_s, b3_s, s8_s, b1_1, b3_1, s8_1)
b12_s = applyFault(b12_s, b12_1, fault_A, "b12")

b13_s, b13_1 := or3Rule(s9_s, b4_s, s11_s, s9_1, b4_1, s11_1)
b13_s = applyFault(b13_s, b13_1, fault_A, "b13")

c1_s, c1_1 := or3Rule(c14_s, s4_s, s5_s, c14_1, s4_1, s5_1)
c1_s = applyFault(c1_s, c1_1, fault_A, "c1")

c7_s, c7_1 := and2Rule(c2_s, ni2_s, c2_1, ni2_1)
c7_s = applyFault(c7_s, c7_1, fault_A, "c7")

c16_s, c16_1 := or3Rule(c15_s, s28_s, s29_s, c15_1, s28_1, s29_1)
c16_s = applyFault(c16_s, c16_1, fault_A, "c16")

d17_s, d17_1 := or3Rule(d1_s, d2_s, d3_s, d1_1, d2_1, d3_1)
d17_s = applyFault(d17_s, d17_1, fault_A, "d17")
```

```
d19_s, d19_1 := or3Rule(s10_s, d6_s, d7_s, s10_1, d6_1, d7_1)
d19_s = applyFault(d19_s, d19_1, fault_A, "d19")

d20_s, d20_1 := or3Rule(s14_s, d9_s, d10_s, s14_1, d9_1, d10_1)
d20_s = applyFault(d20_s, d20_1, fault_A, "d20")

d21_s, d21_1 := or3Rule(s18_s, d11_s, d12_s, s18_1, d11_1, d12_1)
d21_s = applyFault(d21_s, d21_1, fault_A, "d21")

d22_s, d22_1 := or3Rule(s22_s, d13_s, d14_s, s22_1, d13_1, d14_1)
d22_s = applyFault(d22_s, d22_1, fault_A, "d22")

d23_s, d23_1 := or3Rule(s26_s, d15_s, s30_s, s26_1, d15_1, s30_1)
d23_s = applyFault(d23_s, d23_1, fault_A, "d23")

e36_s, e36_1 := or3Rule(e1_s, e2_s, e3_s, e1_1, e2_1, e3_1)
e36_s = applyFault(e36_s, e36_1, fault_A, "e36")

e37_s, e37_1 := or3Rule(e4_s, e31_s, e6_s, e4_1, e31_1, e6_1)
e37_s = applyFault(e37_s, e37_1, fault_A, "e37")

e39_s, e39_1 := or2Rule(e9_s, e32_s, e9_1, e32_1)
e39_s = applyFault(e39_s, e39_1, fault_A, "e39")

e40_s, e40_1 := or3Rule(e11_s, e33_s, e13_s, e11_1, e33_1, e13_1)
e40_s = applyFault(e40_s, e40_1, fault_A, "e40")

e41_s, e41_1 := or3Rule(e14_s, e15_s, e16_s, e14_1, e15_1, e16_1)
e41_s = applyFault(e41_s, e41_1, fault_A, "e41")

e42_s, e42_1 := or3Rule(e17_s, e34_s, e19_s, e17_1, e34_1, e19_1)
e42_s = applyFault(e42_s, e42_1, fault_A, "e42")

e43_s, e43_1 := or3Rule(e20_s, e30_s, a9_s, e20_1, e30_1, a9_1)
e43_s = applyFault(e43_s, e43_1, fault_A, "e43")

e44_s, e44_1 := or3Rule(e21_s, e22_s, e23_s, e21_1, e22_1, e23_1)
e44_s = applyFault(e44_s, e44_1, fault_A, "e44")

e45_s, e45_1 := or3Rule(e35_s, e25_s, e26_s, e35_1, e25_1, e26_1)
e45_s = applyFault(e45_s, e45_1, fault_A, "e45")

e46_s, e46_1 := or3Rule(e27_s, e28_s, e29_s, e27_1, e28_1, e29_1)
e46_s = applyFault(e46_s, e46_1, fault_A, "e46")

out4_s, out2_s, out1_s := or(f1_s, f2_s), or(f3_s, f4_s), or(f5_s, f6_s)

// level 6

a11_s, a11_1 := or3Rule(a10_s, s17_s, a3_s, a10_1, s17_1, a3_1)
a11_s = applyFault(a11_s, a11_1, fault_A, "a11")

b14_s, b14_1 := or3Rule(b12_s, b13_s, b5_s, b12_1, b13_1, b5_1)
b14_s = applyFault(b14_s, b14_1, fault_A, "b14")
```

```
c9_s, c9_1 := or3Rule(c1_s, s6_s, c7_s, c1_1, s6_1, c7_1)
c9_s = applyFault(c9_s, c9_1, fault_A, "c9")

c17_s, c17_1 := or2Rule(c16_s, s30_s, c16_1, s30_1)
c17_s = applyFault(c17_s, c17_1, fault_A, "c17")

d18_s, d18_1 := or3Rule(s6_s, c7_s, d5_s, s6_1, c7_1, d5_1)
d18_s = applyFault(d18_s, d18_1, fault_A, "d18")

d28_s, d28_1 := or2Rule(d23_s, d27_s, d23_1, d27_1)
d28_s = applyFault(d28_s, d28_1, fault_A, "d28")

e38_s, e38_1 := or3Rule(e7_s, c7_s, e8_s, e7_1, c7_1, e8_1)
e38_s = applyFault(e38_s, e38_1, fault_A, "e38")

e47_s, e47_1 := or3Rule(e36_s, e37_s, e44_s, e36_1, e37_1, e44_1)
e47_s = applyFault(e47_s, e47_1, fault_A, "e47")

e49_s, e49_1 := or3Rule(e39_s, e40_s, e41_s, e39_1, e40_1, e41_1)
e49_s = applyFault(e49_s, e49_1, fault_A, "e49")

// level 7

a12_s, a12_1 := or3Rule(a11_s, s19_s, a4_s, a11_1, s19_1, a4_1)
a12_s = applyFault(a12_s, a12_1, fault_A, "a12")

b15_s, b15_1 := or3Rule(b14_s, b6_s, b8_s, b14_1, b6_1, b8_1)
b15_s = applyFault(b15_s, b15_1, fault_A, "b15")

c10_s, c10_1 := or3Rule(c9_s, c3_s, b5_s, c9_1, c3_1, b5_1)
c10_s = applyFault(c10_s, c10_1, fault_A, "c10")

d24_s, d24_1 := or3Rule(d17_s, d18_s, d19_s, d17_1, d18_1, d19_1)
d24_s = applyFault(d24_s, d24_1, fault_A, "d24")

e48_s, e48_1 := or3Rule(e45_s, e46_s, e38_s, e45_1, e46_1, e38_1)
e48_s = applyFault(e48_s, e48_1, fault_A, "e48")

// level 8

a13_s, a13_1 := or3Rule(a12_s, s23_s, a5_s, a12_1, s23_1, a5_1)
a13_s = applyFault(a13_s, a13_1, fault_A, "a13")

b16_s, b16_1 := or3Rule(b15_s, s24_s, b9_s, b15_1, s24_1, b9_1)
b16_s = applyFault(b16_s, b16_1, fault_A, "b16")

c11_s, c11_1 := or3Rule(c10_s, c4_s, c8_s, c10_1, c4_1, c8_1)
c11_s = applyFault(c11_s, c11_1, fault_A, "c11")

d25_s, d25_1 := or3Rule(d24_s, d20_s, d21_s, d24_1, d20_1, d21_1)
d25_s = applyFault(d25_s, d25_1, fault_A, "d25")

e50_s, e50_1 := or3Rule(e47_s, e48_s, e49_s, e47_1, e48_1, e49_1)
```

```
        e50_s = applyFault(e50_s, e50_1, fault_A, "e50")

        // level 9

        ns1_s, ns1_1 := or3Rule(e50_s, e42_s, e43_s, e50_1, e42_1, e43_1)
        ns1_s = applyFault(ns1_s, ns1_1, fault_A, "ns1")

        ns8_s, ns8_1 := or3Rule(b16_s, b11_s, a9_s, b16_1, b11_1, a9_1)
        ns8_s = applyFault(ns8_s, ns8_1, fault_A, "ns8")

        a14_s, a14_1 := or3Rule(a13_s, a6_s, a8_s, a13_1, a6_1, a8_1)
        a14_s = applyFault(a14_s, a14_1, fault_A, "a14")

        c12_s, c12_1 := or3Rule(c11_s, a4_s, c6_s, c11_1, a4_1, c6_1)
        c12_s = applyFault(c12_s, c12_1, fault_A, "c12")

        d26_s, d26_1 := or3Rule(d25_s, d22_s, d28_s, d25_1, d22_1, d28_1)
        d26_s = applyFault(d26_s, d26_1, fault_A, "d26")

        // level 10

        ns2_s, ns2_1 := or3Rule(d26_s, s2_s, a9_s, d26_1, s2_1, a9_1)
        ns2_s = applyFault(ns2_s, ns2_1, fault_A, "ns2")

        ns4_s, ns4_1 := or3Rule(c12_s, c17_s, a9_s, c12_1, c17_1, a9_1)
        ns4_s = applyFault(ns4_s, ns4_1, fault_A, "ns4")

        ns16_s, ns16_1 := or2Rule(a14_s, a9_s, a14_1, a9_1)
        ns16_s = applyFault(ns16_s, ns16_1, fault_A, "ns16")

        return out4_s, out2_s, out1_s, ns16_s, ns8_s, ns4_s, ns2_s, ns1_s

}

// End of ps2ns() ================================================

// ns2fp() determines a fault pattern for each S/I that reaches an ns-line
ns2fp := func(out4_s, out2_s, out1_s, ns16_s, ns8_s, ns4_s, ns2_s,
        ns1_s nd) (S, S, S, S, S, S, S, S, S, S, S, S, S, S, S, S, S, S,
        S, S, S, S, S, S, S, S, S, S, S, S, S) {

        // create fault pattern collections
        fp1 := and(not(ns16_s), not(ns8_s), not(ns4_s), not(ns2_s), ns1_s)
        fp2 := and(not(ns16_s), not(ns8_s), not(ns4_s), ns2_s, not(ns1_s))
        fp3 := and(not(ns16_s), not(ns8_s), not(ns4_s), ns2_s, ns1_s)
        fp4 := and(not(ns16_s), not(ns8_s), ns4_s, not(ns2_s), not(ns1_s))
        fp5 := and(not(ns16_s), not(ns8_s), ns4_s, not(ns2_s), ns1_s)
        fp6 := and(not(ns16_s), not(ns8_s), ns4_s, ns2_s, not(ns1_s))
        fp7 := and(not(ns16_s), not(ns8_s), ns4_s, ns2_s, ns1_s)
        fp8 := and(not(ns16_s), ns8_s, not(ns4_s), not(ns2_s), not(ns1_s))
        fp9 := and(not(ns16_s), ns8_s, not(ns4_s), not(ns2_s), ns1_s)
        fp10 := and(not(ns16_s), ns8_s, not(ns4_s), ns2_s, not(ns1_s))
        fp11 := and(not(ns16_s), ns8_s, not(ns4_s), ns2_s, ns1_s)
        fp12 := and(not(ns16_s), ns8_s, ns4_s, not(ns2_s), not(ns1_s))
```

```
fp13 := and(not(ns16_s), ns8_s, ns4_s, not(ns2_s), ns1_s)
fp14 := and(not(ns16_s), ns8_s, ns4_s, ns2_s, not(ns1_s))
fp15 := and(not(ns16_s), ns8_s, ns4_s, ns2_s, ns1_s)
fp16 := and(ns16_s, not(ns8_s), not(ns4_s), not(ns2_s), not(ns1_s))
fp17 := and(ns16_s, not(ns8_s), not(ns4_s), not(ns2_s), ns1_s)
fp18 := and(ns16_s, not(ns8_s), not(ns4_s), ns2_s, not(ns1_s))
fp19 := and(ns16_s, not(ns8_s), not(ns4_s), ns2_s, ns1_s)
fp20 := and(ns16_s, not(ns8_s), ns4_s, not(ns2_s), not(ns1_s))
fp21 := and(ns16_s, not(ns8_s), ns4_s, not(ns2_s), ns1_s)
fp22 := and(ns16_s, not(ns8_s), ns4_s, ns2_s, not(ns1_s))
fp23 := and(ns16_s, not(ns8_s), ns4_s, ns2_s, ns1_s)
fp24 := and(ns16_s, ns8_s, not(ns4_s), not(ns2_s), not(ns1_s))
fp25 := and(ns16_s, ns8_s, not(ns4_s), not(ns2_s), ns1_s)
fp26 := and(ns16_s, ns8_s, not(ns4_s), ns2_s, not(ns1_s))
fp27 := and(ns16_s, ns8_s, not(ns4_s), ns2_s, ns1_s)
fp28 := and(ns16_s, ns8_s, ns4_s, not(ns2_s), not(ns1_s))
fp29 := and(ns16_s, ns8_s, ns4_s, not(ns2_s), ns1_s)
fp30 := and(ns16_s, ns8_s, ns4_s, ns2_s, not(ns1_s))
fp31 := and(ns16_s, ns8_s, ns4_s, ns2_s, ns1_s)

var collectedSIs S

decompMap := func(fpx, out4_s, out2_s, out1_s nd, cSI S) S {

        //-------------------------------------------------------
        if and(fpx, s0i0) != null {
                cSI = append(cSI, g{s0i0, and(s0i0,
                        out4_s), and(s0i0, out2_s), and(s0i0, out1_s), s0})
        }
        if and(fpx, s0i1) != null {
                cSI = append(cSI, g{s0i1, and(s0i1,
                        out4_s), and(s0i1, out2_s), and(s0i1, out1_s), s1})
        }
        if and(fpx, s0i2) != null {
                cSI = append(cSI, g{s0i2, and(s0i2,
                        out4_s), and(s0i2, out2_s), and(s0i2, out1_s), s0})
        }
        if and(fpx, s0i3) != null {
                cSI = append(cSI, g{s0i3, and(s0i3,
                        out4_s), and(s0i3, out2_s), and(s0i3, out1_s), s0})
        }
        if and(fpx, s0i4) != null {
                cSI = append(cSI, g{s0i4, and(s0i4,
                        out4_s), and(s0i4, out2_s), and(s0i4, out1_s), s0})
        }
        if and(fpx, s0i5) != null {
                cSI = append(cSI, g{s0i5, and(s0i5,
                        out4_s), and(s0i5, out2_s), and(s0i5, out1_s), s0})
        }
        if and(fpx, s0i6) != null {
                cSI = append(cSI, g{s0i6, and(s0i6,
                        out4_s), and(s0i6, out2_s), and(s0i6, out1_s), s0})
        }
        if and(fpx, s0i7) != null {
```

```
        cSl = append(cSl, g{s0i7, and(s0i7,
                out4_s), and(s0i7, out2_s), and(s0i7, out1_s), s0})
}
//-------------------------------------------------------
if and(fpx, s1i0) != null {
        cSl = append(cSl, g{s1i0, and(s1i0,
                out4_s), and(s1i0, out2_s), and(s1i0, out1_s), s1})
}
if and(fpx, s1i1) != null {
        cSl = append(cSl, g{s1i1, and(s1i1,
                out4_s), and(s1i1, out2_s), and(s1i1, out1_s), s1})
}
if and(fpx, s1i2) != null {
        cSl = append(cSl, g{s1i2, and(s1i2,
                out4_s), and(s1i2, out2_s), and(s1i2, out1_s), s2})
}
if and(fpx, s1i3) != null {
        cSl = append(cSl, g{s1i3, and(s1i3,
                out4_s), and(s1i3, out2_s), and(s1i3, out1_s), s1})
}
if and(fpx, s1i4) != null {
        cSl = append(cSl, g{s1i4, and(s1i4,
                out4_s), and(s1i4, out2_s), and(s1i4, out1_s), s1})
}
if and(fpx, s1i5) != null {
        cSl = append(cSl, g{s1i5, and(s1i5,
                out4_s), and(s1i5, out2_s), and(s1i5, out1_s), s1})
}
if and(fpx, s1i6) != null {
        cSl = append(cSl, g{s1i6, and(s1i6,
                out4_s), and(s1i6, out2_s), and(s1i6, out1_s), s1})
}
if and(fpx, s1i7) != null {
        cSl = append(cSl, g{s1i7, and(s1i7,
                out4_s), and(s1i7, out2_s), and(s1i7, out1_s), s1})
}
//-------------------------------------------------------
if and(fpx, s2i0) != null {
        cSl = append(cSl, g{s2i0, and(s2i0,
                out4_s), and(s2i0, out2_s), and(s2i0, out1_s), s2})
}
if and(fpx, s2i1) != null {
        cSl = append(cSl, g{s2i1, and(s2i1,
                out4_s), and(s2i1, out2_s), and(s2i1, out1_s), s2})
}
if and(fpx, s2i2) != null {
        cSl = append(cSl, g{s2i2, and(s2i2,
                out4_s), and(s2i2, out2_s), and(s2i2, out1_s), s3})
}
if and(fpx, s2i3) != null {
        cSl = append(cSl, g{s2i3, and(s2i3,
                out4_s), and(s2i3, out2_s), and(s2i3, out1_s), s2})
}
if and(fpx, s2i4) != null {
```

```
        cSl = append(cSl, g{s2i4, and(s2i4,
                out4_s), and(s2i4, out2_s), and(s2i4, out1_s), s2})
}
if and(fpx, s2i5) != null {
        cSl = append(cSl, g{s2i5, and(s2i5,
                out4_s), and(s2i5, out2_s), and(s2i5, out1_s), s2})
}
if and(fpx, s2i6) != null {
        cSl = append(cSl, g{s2i6, and(s2i6,
                out4_s), and(s2i6, out2_s), and(s2i6, out1_s), s2})
}
if and(fpx, s2i7) != null {
        cSl = append(cSl, g{s2i7, and(s2i7,
                out4_s), and(s2i7, out2_s), and(s2i7, out1_s), s2})
}
//-------------------------------------------------------
if and(fpx, s3i0) != null {
        cSl = append(cSl, g{s3i0, and(s3i0,
                out4_s), and(s3i0, out2_s), and(s3i0, out1_s), s3})
}
if and(fpx, s3i1) != null {
        cSl = append(cSl, g{s3i1, and(s3i1,
                out4_s), and(s3i1, out2_s), and(s3i1, out1_s), s3})
}
if and(fpx, s3i2) != null {
        cSl = append(cSl, g{s3i2, and(s3i2,
                out4_s), and(s3i2, out2_s), and(s3i2, out1_s), s12})
}
if and(fpx, s3i3) != null {
        cSl = append(cSl, g{s3i3, and(s3i3,
                out4_s), and(s3i3, out2_s), and(s3i3, out1_s), s4})
}
if and(fpx, s3i4) != null {
        cSl = append(cSl, g{s3i4, and(s3i4,
                out4_s), and(s3i4, out2_s), and(s3i4, out1_s), s3})
}
if and(fpx, s3i5) != null {
        cSl = append(cSl, g{s3i5, and(s3i5,
                out4_s), and(s3i5, out2_s), and(s3i5, out1_s), s3})
}
if and(fpx, s3i6) != null {
        cSl = append(cSl, g{s3i6, and(s3i6,
                out4_s), and(s3i6, out2_s), and(s3i6, out1_s), s3})
}
if and(fpx, s3i7) != null {
        cSl = append(cSl, g{s3i7, and(s3i7,
                out4_s), and(s3i7, out2_s), and(s3i7, out1_s), s3})
}
//-------------------------------------------------------
if and(fpx, s4i0) != null {
        cSl = append(cSl, g{s4i0, and(s4i0,
                out4_s), and(s4i0, out2_s), and(s4i0, out1_s), s4})
}
if and(fpx, s4i1) != null {
```

```
        cSl = append(cSl, g{s4i1, and(s4i1,
                out4_s), and(s4i1, out2_s), and(s4i1, out1_s), s4})
}
if and(fpx, s4i2) != null {
        cSl = append(cSl, g{s4i2, and(s4i2,
                out4_s), and(s4i2, out2_s), and(s4i2, out1_s), s4})
}
if and(fpx, s4i3) != null {
        cSl = append(cSl, g{s4i3, and(s4i3,
                out4_s), and(s4i3, out2_s), and(s4i3, out1_s), s4})
}
if and(fpx, s4i4) != null {
        cSl = append(cSl, g{s4i4, and(s4i4,
                out4_s), and(s4i4, out2_s), and(s4i4, out1_s), s5})
}
if and(fpx, s4i5) != null {
        cSl = append(cSl, g{s4i5, and(s4i5,
                out4_s), and(s4i5, out2_s), and(s4i5, out1_s), s5})
}
if and(fpx, s4i6) != null {
        cSl = append(cSl, g{s4i6, and(s4i6,
                out4_s), and(s4i6, out2_s), and(s4i6, out1_s), s4})
}
if and(fpx, s4i7) != null {
        cSl = append(cSl, g{s4i7, and(s4i7,
                out4_s), and(s4i7, out2_s), and(s4i7, out1_s), s4})
}
//-------------------------------------------------------
if and(fpx, s5i0) != null {
        cSl = append(cSl, g{s5i0, and(s5i0,
                out4_s), and(s5i0, out2_s), and(s5i0, out1_s), s6})
}
if and(fpx, s5i1) != null {
        cSl = append(cSl, g{s5i1, and(s5i1,
                out4_s), and(s5i1, out2_s), and(s5i1, out1_s), s5})
}
if and(fpx, s5i2) != null {
        cSl = append(cSl, g{s5i2, and(s5i2,
                out4_s), and(s5i2, out2_s), and(s5i2, out1_s), s5})
}
if and(fpx, s5i3) != null {
        cSl = append(cSl, g{s5i3, and(s5i3,
                out4_s), and(s5i3, out2_s), and(s5i3, out1_s), s5})
}
if and(fpx, s5i4) != null {
        cSl = append(cSl, g{s5i4, and(s5i4,
                out4_s), and(s5i4, out2_s), and(s5i4, out1_s), s5})
}
if and(fpx, s5i5) != null {
        cSl = append(cSl, g{s5i5, and(s5i5,
                out4_s), and(s5i5, out2_s), and(s5i5, out1_s), s5})
}
if and(fpx, s5i6) != null {
        cSl = append(cSl, g{s5i6, and(s5i6,
```

```
                out4_s), and(s5i6, out2_s), and(s5i6, out1_s), s5})
}
if and(fpx, s5i7) != null {
        cSl = append(cSl, g{s5i7, and(s5i7,
                out4_s), and(s5i7, out2_s), and(s5i7, out1_s), s5})
}
//-------------------------------------------------------
if and(fpx, s6i0) != null {
        cSl = append(cSl, g{s6i0, and(s6i0,
                out4_s), and(s6i0, out2_s), and(s6i0, out1_s), s6})
}
if and(fpx, s6i1) != null {
        cSl = append(cSl, g{s6i1, and(s6i1,
                out4_s), and(s6i1, out2_s), and(s6i1, out1_s), s6})
}
if and(fpx, s6i2) != null {
        cSl = append(cSl, g{s6i2, and(s6i2,
                out4_s), and(s6i2, out2_s), and(s6i2, out1_s), s6})
}
if and(fpx, s6i3) != null {
        cSl = append(cSl, g{s6i3, and(s6i3,
                out4_s), and(s6i3, out2_s), and(s6i3, out1_s), s6})
}
if and(fpx, s6i4) != null {
        cSl = append(cSl, g{s6i4, and(s6i4,
                out4_s), and(s6i4, out2_s), and(s6i4, out1_s), s6})
}
if and(fpx, s6i5) != null {
        cSl = append(cSl, g{s6i5, and(s6i5,
                out4_s), and(s6i5, out2_s), and(s6i5, out1_s), s6})
}
if and(fpx, s6i6) != null {
        cSl = append(cSl, g{s6i6, and(s6i6,
                out4_s), and(s6i6, out2_s), and(s6i6, out1_s), s6})
}
if and(fpx, s6i7) != null {
        cSl = append(cSl, g{s6i7, and(s6i7,
                out4_s), and(s6i7, out2_s), and(s6i7, out1_s), s7})
}
//-------------------------------------------------------
if and(fpx, s7i0) != null {
        cSl = append(cSl, g{s7i0, and(s7i0,
                out4_s), and(s7i0, out2_s), and(s7i0, out1_s), s7})
}
if and(fpx, s7i1) != null {
        cSl = append(cSl, g{s7i1, and(s7i1,
                out4_s), and(s7i1, out2_s), and(s7i1, out1_s), s7})
}
if and(fpx, s7i2) != null {
        cSl = append(cSl, g{s7i2, and(s7i2,
                out4_s), and(s7i2, out2_s), and(s7i2, out1_s), s8})
}
if and(fpx, s7i3) != null {
        cSl = append(cSl, g{s7i3, and(s7i3,
```

```
                out4_s), and(s7i3, out2_s), and(s7i3, out1_s), s8})
}
if and(fpx, s7i4) != null {
        cSl = append(cSl, g{s7i4, and(s7i4,
                out4_s), and(s7i4, out2_s), and(s7i4, out1_s), s7})
}
if and(fpx, s7i5) != null {
        cSl = append(cSl, g{s7i5, and(s7i5,
                out4_s), and(s7i5, out2_s), and(s7i5, out1_s), s8})
}
if and(fpx, s7i6) != null {
        cSl = append(cSl, g{s7i6, and(s7i6,
                out4_s), and(s7i6, out2_s), and(s7i6, out1_s), s7})
}
if and(fpx, s7i7) != null {
        cSl = append(cSl, g{s7i7, and(s7i7,
                out4_s), and(s7i7, out2_s), and(s7i7, out1_s), s7})
}
//-------------------------------------------------------
if and(fpx, s8i0) != null {
        cSl = append(cSl, g{s8i0, and(s8i0,
                out4_s), and(s8i0, out2_s), and(s8i0, out1_s), s8})
}
if and(fpx, s8i1) != null {
        cSl = append(cSl, g{s8i1, and(s8i1,
                out4_s), and(s8i1, out2_s), and(s8i1, out1_s), s9})
}
if and(fpx, s8i2) != null {
        cSl = append(cSl, g{s8i2, and(s8i2,
                out4_s), and(s8i2, out2_s), and(s8i2, out1_s), s8})
}
if and(fpx, s8i3) != null {
        cSl = append(cSl, g{s8i3, and(s8i3,
                out4_s), and(s8i3, out2_s), and(s8i3, out1_s), s8})
}
if and(fpx, s8i4) != null {
        cSl = append(cSl, g{s8i4, and(s8i4,
                out4_s), and(s8i4, out2_s), and(s8i4, out1_s), s8})
}
if and(fpx, s8i5) != null {
        cSl = append(cSl, g{s8i5, and(s8i5,
                out4_s), and(s8i5, out2_s), and(s8i5, out1_s), s8})
}
if and(fpx, s8i6) != null {
        cSl = append(cSl, g{s8i6, and(s8i6,
                out4_s), and(s8i6, out2_s), and(s8i6, out1_s), s8})
}
if and(fpx, s8i7) != null {
        cSl = append(cSl, g{s8i7, and(s8i7,
                out4_s), and(s8i7, out2_s), and(s8i7, out1_s), s8})
}
//-------------------------------------------------------
if and(fpx, s9i0) != null {
        cSl = append(cSl, g{s9i0, and(s9i0,
```

```
                out4_s), and(s9i0, out2_s), and(s9i0, out1_s), s9})
}
if and(fpx, s9i1) != null {
        cSI = append(cSI, g{s9i1, and(s9i1,
                out4_s), and(s9i1, out2_s), and(s9i1, out1_s), s9})
}
if and(fpx, s9i2) != null {
        cSI = append(cSI, g{s9i2, and(s9i2,
                out4_s), and(s9i2, out2_s), and(s9i2, out1_s), s10})
}
if and(fpx, s9i3) != null {
        cSI = append(cSI, g{s9i3, and(s9i3,
                out4_s), and(s9i3, out2_s), and(s9i3, out1_s), s9})
}
if and(fpx, s9i4) != null {
        cSI = append(cSI, g{s9i4, and(s9i4,
                out4_s), and(s9i4, out2_s), and(s9i4, out1_s), s9})
}
if and(fpx, s9i5) != null {
        cSI = append(cSI, g{s9i5, and(s9i5,
                out4_s), and(s9i5, out2_s), and(s9i5, out1_s), s9})
}
if and(fpx, s9i6) != null {
        cSI = append(cSI, g{s9i6, and(s9i6,
                out4_s), and(s9i6, out2_s), and(s9i6, out1_s), s9})
}
if and(fpx, s9i7) != null {
        cSI = append(cSI, g{s9i7, and(s9i7,
                out4_s), and(s9i7, out2_s), and(s9i7, out1_s), s9})
}
//-------------------------------------------------------
if and(fpx, s10i0) != null {
        cSI = append(cSI, g{s10i0, and(s10i0,
                out4_s), and(s10i0, out2_s), and(s10i0, out1_s), s27})
}
if and(fpx, s10i1) != null {
        cSI = append(cSI, g{s10i1, and(s10i1,
                out4_s), and(s10i1, out2_s), and(s10i1, out1_s), s10})
}
if and(fpx, s10i2) != null {
        cSI = append(cSI, g{s10i2, and(s10i2,
                out4_s), and(s10i2, out2_s), and(s10i2, out1_s), s11})
}
if and(fpx, s10i3) != null {
        cSI = append(cSI, g{s10i3, and(s10i3,
                out4_s), and(s10i3, out2_s), and(s10i3, out1_s), s10})
}
if and(fpx, s10i4) != null {
        cSI = append(cSI, g{s10i4, and(s10i4,
                out4_s), and(s10i4, out2_s), and(s10i4, out1_s), s10})
}
if and(fpx, s10i5) != null {
        cSI = append(cSI, g{s10i5, and(s10i5,
                out4_s), and(s10i5, out2_s), and(s10i5, out1_s), s10})
```

```
}
if and(fpx, s10i6) != null {
        cSl = append(cSl, g{s10i6, and(s10i6,
                out4_s), and(s10i6, out2_s), and(s10i6, out1_s), s3})
}
if and(fpx, s10i7) != null {
        cSl = append(cSl, g{s10i7, and(s10i7,
                out4_s), and(s10i7, out2_s), and(s10i7, out1_s), s10})
}
//-------------------------------------------------------
if and(fpx, s11i0) != null {
        cSl = append(cSl, g{s11i0, and(s11i0,
                out4_s), and(s11i0, out2_s), and(s11i0, out1_s), s11})
}
if and(fpx, s11i1) != null {
        cSl = append(cSl, g{s11i1, and(s11i1,
                out4_s), and(s11i1, out2_s), and(s11i1, out1_s), s11})
}
if and(fpx, s11i2) != null {
        cSl = append(cSl, g{s11i2, and(s11i2,
                out4_s), and(s11i2, out2_s), and(s11i2, out1_s), s11})
}
if and(fpx, s11i3) != null {
        cSl = append(cSl, g{s11i3, and(s11i3,
                out4_s), and(s11i3, out2_s), and(s11i3, out1_s), s11})
}
if and(fpx, s11i4) != null {
        cSl = append(cSl, g{s11i4, and(s11i4,
                out4_s), and(s11i4, out2_s), and(s11i4, out1_s), s11})
}
if and(fpx, s11i5) != null {
        cSl = append(cSl, g{s11i5, and(s11i5,
                out4_s), and(s11i5, out2_s), and(s11i5, out1_s), s11})
}
if and(fpx, s11i6) != null {
        cSl = append(cSl, g{s11i6, and(s11i6,
                out4_s), and(s11i6, out2_s), and(s11i6, out1_s), s11})
}
if and(fpx, s11i7) != null {
        cSl = append(cSl, g{s11i7, and(s11i7,
                out4_s), and(s11i7, out2_s), and(s11i7, out1_s), s12})
}
//-------------------------------------------------------
if and(fpx, s12i0) != null {
        cSl = append(cSl, g{s12i0, and(s12i0,
                out4_s), and(s12i0, out2_s), and(s12i0, out1_s), s12})
}
if and(fpx, s12i1) != null {
        cSl = append(cSl, g{s12i1, and(s12i1,
                out4_s), and(s12i1, out2_s), and(s12i1, out1_s), s12})
}
if and(fpx, s12i2) != null {
        cSl = append(cSl, g{s12i2, and(s12i2,
                out4_s), and(s12i2, out2_s), and(s12i2, out1_s), s12})
```

```
}
if and(fpx, s12i3) != null {
        cSl = append(cSl, g{s12i3, and(s12i3,
                out4_s), and(s12i3, out2_s), and(s12i3, out1_s), s13})
}
if and(fpx, s12i4) != null {
        cSl = append(cSl, g{s12i4, and(s12i4,
                out4_s), and(s12i4, out2_s), and(s12i4, out1_s), s12})
}
if and(fpx, s12i5) != null {
        cSl = append(cSl, g{s12i5, and(s12i5,
                out4_s), and(s12i5, out2_s), and(s12i5, out1_s), s12})
}
if and(fpx, s12i6) != null {
        cSl = append(cSl, g{s12i6, and(s12i6,
                out4_s), and(s12i6, out2_s), and(s12i6, out1_s), s13})
}
if and(fpx, s12i7) != null {
        cSl = append(cSl, g{s12i7, and(s12i7,
                out4_s), and(s12i7, out2_s), and(s12i7, out1_s), s12})
}
//-------------------------------------------------------
if and(fpx, s13i0) != null {
        cSl = append(cSl, g{s13i0, and(s13i0,
                out4_s), and(s13i0, out2_s), and(s13i0, out1_s), s14})
}
if and(fpx, s13i1) != null {
        cSl = append(cSl, g{s13i1, and(s13i1,
                out4_s), and(s13i1, out2_s), and(s13i1, out1_s), s13})
}
if and(fpx, s13i2) != null {
        cSl = append(cSl, g{s13i2, and(s13i2,
                out4_s), and(s13i2, out2_s), and(s13i2, out1_s), s13})
}
if and(fpx, s13i3) != null {
        cSl = append(cSl, g{s13i3, and(s13i3,
                out4_s), and(s13i3, out2_s), and(s13i3, out1_s), s13})
}
if and(fpx, s13i4) != null {
        cSl = append(cSl, g{s13i4, and(s13i4,
                out4_s), and(s13i4, out2_s), and(s13i4, out1_s), s13})
}
if and(fpx, s13i5) != null {
        cSl = append(cSl, g{s13i5, and(s13i5,
                out4_s), and(s13i5, out2_s), and(s13i5, out1_s), s13})
}
if and(fpx, s13i6) != null {
        cSl = append(cSl, g{s13i6, and(s13i6,
                out4_s), and(s13i6, out2_s), and(s13i6, out1_s), s13})
}
if and(fpx, s13i7) != null {
        cSl = append(cSl, g{s13i7, and(s13i7,
                out4_s), and(s13i7, out2_s), and(s13i7, out1_s), s13})
}
```

```
//-------------------------------------------------------
if and(fpx, s14i0) != null {
        cSl = append(cSl, g{s14i0, and(s14i0,
                out4_s), and(s14i0, out2_s), and(s14i0, out1_s), s14})
}
if and(fpx, s14i1) != null {
        cSl = append(cSl, g{s14i1, and(s14i1,
                out4_s), and(s14i1, out2_s), and(s14i1, out1_s), s14})
}
if and(fpx, s14i2) != null {
        cSl = append(cSl, g{s14i2, and(s14i2,
                out4_s), and(s14i2, out2_s), and(s14i2, out1_s), s14})
}
if and(fpx, s14i3) != null {
        cSl = append(cSl, g{s14i3, and(s14i3,
                out4_s), and(s14i3, out2_s), and(s14i3, out1_s), s14})
}
if and(fpx, s14i4) != null {
        cSl = append(cSl, g{s14i4, and(s14i4,
                out4_s), and(s14i4, out2_s), and(s14i4, out1_s), s14})
}
if and(fpx, s14i5) != null {
        cSl = append(cSl, g{s14i5, and(s14i5,
                out4_s), and(s14i5, out2_s), and(s14i5, out1_s), s14})
}
if and(fpx, s14i6) != null {
        cSl = append(cSl, g{s14i6, and(s14i6,
                out4_s), and(s14i6, out2_s), and(s14i6, out1_s), s14})
}
if and(fpx, s14i7) != null {
        cSl = append(cSl, g{s14i7, and(s14i7,
                out4_s), and(s14i7, out2_s), and(s14i7, out1_s), s15})
}
//-------------------------------------------------------
if and(fpx, s15i0) != null {
        cSl = append(cSl, g{s15i0, and(s15i0,
                out4_s), and(s15i0, out2_s), and(s15i0, out1_s), s15})
}
if and(fpx, s15i1) != null {
        cSl = append(cSl, g{s15i1, and(s15i1,
                out4_s), and(s15i1, out2_s), and(s15i1, out1_s), s15})
}
if and(fpx, s15i2) != null {
        cSl = append(cSl, g{s15i2, and(s15i2,
                out4_s), and(s15i2, out2_s), and(s15i2, out1_s), s15})
}
if and(fpx, s15i3) != null {
        cSl = append(cSl, g{s15i3, and(s15i3,
                out4_s), and(s15i3, out2_s), and(s15i3, out1_s), s15})
}
if and(fpx, s15i4) != null {
        cSl = append(cSl, g{s15i4, and(s15i4,
                out4_s), and(s15i4, out2_s), and(s15i4, out1_s), s15})
}
```

```
if and(fpx, s15i5) != null {
        cSl = append(cSl, g{s15i5, and(s15i5,
                out4_s), and(s15i5, out2_s), and(s15i5, out1_s), s16})
}
if and(fpx, s15i6) != null {
        cSl = append(cSl, g{s15i6, and(s15i6,
                out4_s), and(s15i6, out2_s), and(s15i6, out1_s), s15})
}
if and(fpx, s15i7) != null {
        cSl = append(cSl, g{s15i7, and(s15i7,
                out4_s), and(s15i7, out2_s), and(s15i7, out1_s), s15})
}
//-------------------------------------------------------
if and(fpx, s16i0) != null {
        cSl = append(cSl, g{s16i0, and(s16i0,
                out4_s), and(s16i0, out2_s), and(s16i0, out1_s), s16})
}
if and(fpx, s16i1) != null {
        cSl = append(cSl, g{s16i1, and(s16i1,
                out4_s), and(s16i1, out2_s), and(s16i1, out1_s), s17})
}
if and(fpx, s16i2) != null {
        cSl = append(cSl, g{s16i2, and(s16i2,
                out4_s), and(s16i2, out2_s), and(s16i2, out1_s), s16})
}
if and(fpx, s16i3) != null {
        cSl = append(cSl, g{s16i3, and(s16i3,
                out4_s), and(s16i3, out2_s), and(s16i3, out1_s), s16})
}
if and(fpx, s16i4) != null {
        cSl = append(cSl, g{s16i4, and(s16i4,
                out4_s), and(s16i4, out2_s), and(s16i4, out1_s), s16})
}
if and(fpx, s16i5) != null {
        cSl = append(cSl, g{s16i5, and(s16i5,
                out4_s), and(s16i5, out2_s), and(s16i5, out1_s), s16})
}
if and(fpx, s16i6) != null {
        cSl = append(cSl, g{s16i6, and(s16i6,
                out4_s), and(s16i6, out2_s), and(s16i6, out1_s), s16})
}
if and(fpx, s16i7) != null {
        cSl = append(cSl, g{s16i7, and(s16i7,
                out4_s), and(s16i7, out2_s), and(s16i7, out1_s), s16})
}
//-------------------------------------------------------
if and(fpx, s17i0) != null {
        cSl = append(cSl, g{s17i0, and(s17i0,
                out4_s), and(s17i0, out2_s), and(s17i0, out1_s), s17})
}
if and(fpx, s17i1) != null {
        cSl = append(cSl, g{s17i1, and(s17i1,
                out4_s), and(s17i1, out2_s), and(s17i1, out1_s), s17})
}
```

```
if and(fpx, s17i2) != null {
        cSl = append(cSl, g{s17i2, and(s17i2,
                out4_s), and(s17i2, out2_s), and(s17i2, out1_s), s18})
}
if and(fpx, s17i3) != null {
        cSl = append(cSl, g{s17i3, and(s17i3,
                out4_s), and(s17i3, out2_s), and(s17i3, out1_s), s17})
}
if and(fpx, s17i4) != null {
        cSl = append(cSl, g{s17i4, and(s17i4,
                out4_s), and(s17i4, out2_s), and(s17i4, out1_s), s17})
}
if and(fpx, s17i5) != null {
        cSl = append(cSl, g{s17i5, and(s17i5,
                out4_s), and(s17i5, out2_s), and(s17i5, out1_s), s17})
}
if and(fpx, s17i6) != null {
        cSl = append(cSl, g{s17i6, and(s17i6,
                out4_s), and(s17i6, out2_s), and(s17i6, out1_s), s17})
}
if and(fpx, s17i7) != null {
        cSl = append(cSl, g{s17i7, and(s17i7,
                out4_s), and(s17i7, out2_s), and(s17i7, out1_s), s17})
}
//-------------------------------------------------------
if and(fpx, s18i0) != null {
        cSl = append(cSl, g{s18i0, and(s18i0,
                out4_s), and(s18i0, out2_s), and(s18i0, out1_s), s18})
}
if and(fpx, s18i1) != null {
        cSl = append(cSl, g{s18i1, and(s18i1,
                out4_s), and(s18i1, out2_s), and(s18i1, out1_s), s18})
}
if and(fpx, s18i2) != null {
        cSl = append(cSl, g{s18i2, and(s18i2,
                out4_s), and(s18i2, out2_s), and(s18i2, out1_s), s19})
}
if and(fpx, s18i3) != null {
        cSl = append(cSl, g{s18i3, and(s18i3,
                out4_s), and(s18i3, out2_s), and(s18i3, out1_s), s18})
}
if and(fpx, s18i4) != null {
        cSl = append(cSl, g{s18i4, and(s18i4,
                out4_s), and(s18i4, out2_s), and(s18i4, out1_s), s18})
}
if and(fpx, s18i5) != null {
        cSl = append(cSl, g{s18i5, and(s18i5,
                out4_s), and(s18i5, out2_s), and(s18i5, out1_s), s18})
}
if and(fpx, s18i6) != null {
        cSl = append(cSl, g{s18i6, and(s18i6,
                out4_s), and(s18i6, out2_s), and(s18i6, out1_s), s7})
}
if and(fpx, s18i7) != null {
```

```
        cSl = append(cSl, g{s18i7, and(s18i7,
                out4_s), and(s18i7, out2_s), and(s18i7, out1_s), s18})
}
//-------------------------------------------------------
if and(fpx, s19i0) != null {
        cSl = append(cSl, g{s19i0, and(s19i0,
                out4_s), and(s19i0, out2_s), and(s19i0, out1_s), s19})
}
if and(fpx, s19i1) != null {
        cSl = append(cSl, g{s19i1, and(s19i1,
                out4_s), and(s19i1, out2_s), and(s19i1, out1_s), s19})
}
if and(fpx, s19i2) != null {
        cSl = append(cSl, g{s19i2, and(s19i2,
                out4_s), and(s19i2, out2_s), and(s19i2, out1_s), s19})
}
if and(fpx, s19i3) != null {
        cSl = append(cSl, g{s19i3, and(s19i3,
                out4_s), and(s19i3, out2_s), and(s19i3, out1_s), s19})
}
if and(fpx, s19i4) != null {
        cSl = append(cSl, g{s19i4, and(s19i4,
                out4_s), and(s19i4, out2_s), and(s19i4, out1_s), s19})
}
if and(fpx, s19i5) != null {
        cSl = append(cSl, g{s19i5, and(s19i5,
                out4_s), and(s19i5, out2_s), and(s19i5, out1_s), s23})
}
if and(fpx, s19i6) != null {
        cSl = append(cSl, g{s19i6, and(s19i6,
                out4_s), and(s19i6, out2_s), and(s19i6, out1_s), s19})
}
if and(fpx, s19i7) != null {
        cSl = append(cSl, g{s19i7, and(s19i7,
                out4_s), and(s19i7, out2_s), and(s19i7, out1_s), s20})
}
//-------------------------------------------------------
if and(fpx, s20i0) != null {
        cSl = append(cSl, g{s20i0, and(s20i0,
                out4_s), and(s20i0, out2_s), and(s20i0, out1_s), s20})
}
if and(fpx, s20i1) != null {
        cSl = append(cSl, g{s20i1, and(s20i1,
                out4_s), and(s20i1, out2_s), and(s20i1, out1_s), s20})
}
if and(fpx, s20i2) != null {
        cSl = append(cSl, g{s20i2, and(s20i2,
                out4_s), and(s20i2, out2_s), and(s20i2, out1_s), s20})
}
if and(fpx, s20i3) != null {
        cSl = append(cSl, g{s20i3, and(s20i3,
                out4_s), and(s20i3, out2_s), and(s20i3, out1_s), s21})
}
if and(fpx, s20i4) != null {
```

```
        cSl = append(cSl, g{s20i4, and(s20i4,
                out4_s), and(s20i4, out2_s), and(s20i4, out1_s), s20})
}
if and(fpx, s20i5) != null {
        cSl = append(cSl, g{s20i5, and(s20i5,
                out4_s), and(s20i5, out2_s), and(s20i5, out1_s), s20})
}
if and(fpx, s20i6) != null {
        cSl = append(cSl, g{s20i6, and(s20i6,
                out4_s), and(s20i6, out2_s), and(s20i6, out1_s), s21})
}
if and(fpx, s20i7) != null {
        cSl = append(cSl, g{s20i7, and(s20i7,
                out4_s), and(s20i7, out2_s), and(s20i7, out1_s), s20})
}
//-------------------------------------------------------
if and(fpx, s21i0) != null {
        cSl = append(cSl, g{s21i0, and(s21i0,
                out4_s), and(s21i0, out2_s), and(s21i0, out1_s), s22})
}
if and(fpx, s21i1) != null {
        cSl = append(cSl, g{s21i1, and(s21i1,
                out4_s), and(s21i1, out2_s), and(s21i1, out1_s), s21})
}
if and(fpx, s21i2) != null {
        cSl = append(cSl, g{s21i2, and(s21i2,
                out4_s), and(s21i2, out2_s), and(s21i2, out1_s), s21})
}
if and(fpx, s21i3) != null {
        cSl = append(cSl, g{s21i3, and(s21i3,
                out4_s), and(s21i3, out2_s), and(s21i3, out1_s), s21})
}
if and(fpx, s21i4) != null {
        cSl = append(cSl, g{s21i4, and(s21i4,
                out4_s), and(s21i4, out2_s), and(s21i4, out1_s), s21})
}
if and(fpx, s21i5) != null {
        cSl = append(cSl, g{s21i5, and(s21i5,
                out4_s), and(s21i5, out2_s), and(s21i5, out1_s), s21})
}
if and(fpx, s21i6) != null {
        cSl = append(cSl, g{s21i6, and(s21i6,
                out4_s), and(s21i6, out2_s), and(s21i6, out1_s), s21})
}
if and(fpx, s21i7) != null {
        cSl = append(cSl, g{s21i7, and(s21i7,
                out4_s), and(s21i7, out2_s), and(s21i7, out1_s), s21})
}
//-------------------------------------------------------
if and(fpx, s22i0) != null {
        cSl = append(cSl, g{s22i0, and(s22i0,
                out4_s), and(s22i0, out2_s), and(s22i0, out1_s), s22})
}
if and(fpx, s22i1) != null {
```

```
        cSl = append(cSl, g{s22i1, and(s22i1,
                out4_s), and(s22i1, out2_s), and(s22i1, out1_s), s22})
}
if and(fpx, s22i2) != null {
        cSl = append(cSl, g{s22i2, and(s22i2,
                out4_s), and(s22i2, out2_s), and(s22i2, out1_s), s22})
}
if and(fpx, s22i3) != null {
        cSl = append(cSl, g{s22i3, and(s22i3,
                out4_s), and(s22i3, out2_s), and(s22i3, out1_s), s22})
}
if and(fpx, s22i4) != null {
        cSl = append(cSl, g{s22i4, and(s22i4,
                out4_s), and(s22i4, out2_s), and(s22i4, out1_s), s22})
}
if and(fpx, s22i5) != null {
        cSl = append(cSl, g{s22i5, and(s22i5,
                out4_s), and(s22i5, out2_s), and(s22i5, out1_s), s22})
}
if and(fpx, s22i6) != null {
        cSl = append(cSl, g{s22i6, and(s22i6,
                out4_s), and(s22i6, out2_s), and(s22i6, out1_s), s22})
}
if and(fpx, s22i7) != null {
        cSl = append(cSl, g{s22i7, and(s22i7,
                out4_s), and(s22i7, out2_s), and(s22i7, out1_s), s23})
}
//-------------------------------------------------------
if and(fpx, s23i0) != null {
        cSl = append(cSl, g{s23i0, and(s23i0,
                out4_s), and(s23i0, out2_s), and(s23i0, out1_s), s23})
}
if and(fpx, s23i1) != null {
        cSl = append(cSl, g{s23i1, and(s23i1,
                out4_s), and(s23i1, out2_s), and(s23i1, out1_s), s29})
}
if and(fpx, s23i2) != null {
        cSl = append(cSl, g{s23i2, and(s23i2,
                out4_s), and(s23i2, out2_s), and(s23i2, out1_s), s23})
}
if and(fpx, s23i3) != null {
        cSl = append(cSl, g{s23i3, and(s23i3,
                out4_s), and(s23i3, out2_s), and(s23i3, out1_s), s23})
}
if and(fpx, s23i4) != null {
        cSl = append(cSl, g{s23i4, and(s23i4,
                out4_s), and(s23i4, out2_s), and(s23i4, out1_s), s23})
}
if and(fpx, s23i5) != null {
        cSl = append(cSl, g{s23i5, and(s23i5,
                out4_s), and(s23i5, out2_s), and(s23i5, out1_s), s24})
}
if and(fpx, s23i6) != null {
        cSl = append(cSl, g{s23i6, and(s23i6,
```

```
                out4_s), and(s23i6, out2_s), and(s23i6, out1_s), s23})
}
if and(fpx, s23i7) != null {
        cSl = append(cSl, g{s23i7, and(s23i7,
                out4_s), and(s23i7, out2_s), and(s23i7, out1_s), s23})
}
//-------------------------------------------------------
if and(fpx, s24i0) != null {
        cSl = append(cSl, g{s24i0, and(s24i0,
                out4_s), and(s24i0, out2_s), and(s24i0, out1_s), s24})
}
if and(fpx, s24i1) != null {
        cSl = append(cSl, g{s24i1, and(s24i1,
                out4_s), and(s24i1, out2_s), and(s24i1, out1_s), s25})
}
if and(fpx, s24i2) != null {
        cSl = append(cSl, g{s24i2, and(s24i2,
                out4_s), and(s24i2, out2_s), and(s24i2, out1_s), s24})
}
if and(fpx, s24i3) != null {
        cSl = append(cSl, g{s24i3, and(s24i3,
                out4_s), and(s24i3, out2_s), and(s24i3, out1_s), s24})
}
if and(fpx, s24i4) != null {
        cSl = append(cSl, g{s24i4, and(s24i4,
                out4_s), and(s24i4, out2_s), and(s24i4, out1_s), s24})
}
if and(fpx, s24i5) != null {
        cSl = append(cSl, g{s24i5, and(s24i5,
                out4_s), and(s24i5, out2_s), and(s24i5, out1_s), s24})
}
if and(fpx, s24i6) != null {
        cSl = append(cSl, g{s24i6, and(s24i6,
                out4_s), and(s24i6, out2_s), and(s24i6, out1_s), s24})
}
if and(fpx, s24i7) != null {
        cSl = append(cSl, g{s24i7, and(s24i7,
                out4_s), and(s24i7, out2_s), and(s24i7, out1_s), s9})
}
//-------------------------------------------------------
if and(fpx, s25i0) != null {
        cSl = append(cSl, g{s25i0, and(s25i0,
                out4_s), and(s25i0, out2_s), and(s25i0, out1_s), s25})
}
if and(fpx, s25i1) != null {
        cSl = append(cSl, g{s25i1, and(s25i1,
                out4_s), and(s25i1, out2_s), and(s25i1, out1_s), s25})
}
if and(fpx, s25i2) != null {
        cSl = append(cSl, g{s25i2, and(s25i2,
                out4_s), and(s25i2, out2_s), and(s25i2, out1_s), s26})
}
if and(fpx, s25i3) != null {
        cSl = append(cSl, g{s25i3, and(s25i3,
```

```
                out4_s), and(s25i3, out2_s), and(s25i3, out1_s), s25})
}
if and(fpx, s25i4) != null {
        cSl = append(cSl, g{s25i4, and(s25i4,
                out4_s), and(s25i4, out2_s), and(s25i4, out1_s), s25})
}
if and(fpx, s25i5) != null {
        cSl = append(cSl, g{s25i5, and(s25i5,
                out4_s), and(s25i5, out2_s), and(s25i5, out1_s), s25})
}
if and(fpx, s25i6) != null {
        cSl = append(cSl, g{s25i6, and(s25i6,
                out4_s), and(s25i6, out2_s), and(s25i6, out1_s), s25})
}
if and(fpx, s25i7) != null {
        cSl = append(cSl, g{s25i7, and(s25i7,
                out4_s), and(s25i7, out2_s), and(s25i7, out1_s), s25})
}
//-------------------------------------------------------
if and(fpx, s26i0) != null {
        cSl = append(cSl, g{s26i0, and(s26i0,
                out4_s), and(s26i0, out2_s), and(s26i0, out1_s), s26})
}
if and(fpx, s26i1) != null {
        cSl = append(cSl, g{s26i1, and(s26i1,
                out4_s), and(s26i1, out2_s), and(s26i1, out1_s), s26})
}
if and(fpx, s26i2) != null {
        cSl = append(cSl, g{s26i2, and(s26i2,
                out4_s), and(s26i2, out2_s), and(s26i2, out1_s), s27})
}
if and(fpx, s26i3) != null {
        cSl = append(cSl, g{s26i3, and(s26i3,
                out4_s), and(s26i3, out2_s), and(s26i3, out1_s), s26})
}
if and(fpx, s26i4) != null {
        cSl = append(cSl, g{s26i4, and(s26i4,
                out4_s), and(s26i4, out2_s), and(s26i4, out1_s), s26})
}
if and(fpx, s26i5) != null {
        cSl = append(cSl, g{s26i5, and(s26i5,
                out4_s), and(s26i5, out2_s), and(s26i5, out1_s), s26})
}
if and(fpx, s26i6) != null {
        cSl = append(cSl, g{s26i6, and(s26i6,
                out4_s), and(s26i6, out2_s), and(s26i6, out1_s), s26})
}
if and(fpx, s26i7) != null {
        cSl = append(cSl, g{s26i7, and(s26i7,
                out4_s), and(s26i7, out2_s), and(s26i7, out1_s), s26})
}
//-------------------------------------------------------
if and(fpx, s27i0) != null {
        cSl = append(cSl, g{s27i0, and(s27i0,
```

```
                out4_s), and(s27i0, out2_s), and(s27i0, out1_s), s27})
}
if and(fpx, s27i1) != null {
        cSl = append(cSl, g{s27i1, and(s27i1,
                out4_s), and(s27i1, out2_s), and(s27i1, out1_s), s27})
}
if and(fpx, s27i2) != null {
        cSl = append(cSl, g{s27i2, and(s27i2,
                out4_s), and(s27i2, out2_s), and(s27i2, out1_s), s27})
}
if and(fpx, s27i3) != null {
        cSl = append(cSl, g{s27i3, and(s27i3,
                out4_s), and(s27i3, out2_s), and(s27i3, out1_s), s27})
}
if and(fpx, s27i4) != null {
        cSl = append(cSl, g{s27i4, and(s27i4,
                out4_s), and(s27i4, out2_s), and(s27i4, out1_s), s27})
}
if and(fpx, s27i5) != null {
        cSl = append(cSl, g{s27i5, and(s27i5,
                out4_s), and(s27i5, out2_s), and(s27i5, out1_s), s27})
}
if and(fpx, s27i6) != null {
        cSl = append(cSl, g{s27i6, and(s27i6,
                out4_s), and(s27i6, out2_s), and(s27i6, out1_s), s27})
}
if and(fpx, s27i7) != null {
        cSl = append(cSl, g{s27i7, and(s27i7,
                out4_s), and(s27i7, out2_s), and(s27i7, out1_s), s28})
}
//-------------------------------------------------------
if and(fpx, s28i0) != null {
        cSl = append(cSl, g{s28i0, and(s28i0,
                out4_s), and(s28i0, out2_s), and(s28i0, out1_s), s28})
}
if and(fpx, s28i1) != null {
        cSl = append(cSl, g{s28i1, and(s28i1,
                out4_s), and(s28i1, out2_s), and(s28i1, out1_s), s28})
}
if and(fpx, s28i2) != null {
        cSl = append(cSl, g{s28i2, and(s28i2,
                out4_s), and(s28i2, out2_s), and(s28i2, out1_s), s28})
}
if and(fpx, s28i3) != null {
        cSl = append(cSl, g{s28i3, and(s28i3,
                out4_s), and(s28i3, out2_s), and(s28i3, out1_s), s29})
}
if and(fpx, s28i4) != null {
        cSl = append(cSl, g{s28i4, and(s28i4,
                out4_s), and(s28i4, out2_s), and(s28i4, out1_s), s28})
}
if and(fpx, s28i5) != null {
        cSl = append(cSl, g{s28i5, and(s28i5,
                out4_s), and(s28i5, out2_s), and(s28i5, out1_s), s28})
```

```
}
if and(fpx, s28i6) != null {
	cSl = append(cSl, g{s28i6, and(s28i6,
		out4_s), and(s28i6, out2_s), and(s28i6, out1_s), s29})
}
if and(fpx, s28i7) != null {
	cSl = append(cSl, g{s28i7, and(s28i7,
		out4_s), and(s28i7, out2_s), and(s28i7, out1_s), s28})
}
//-------------------------------------------------------
if and(fpx, s29i0) != null {
	cSl = append(cSl, g{s29i0, and(s29i0,
		out4_s), and(s29i0, out2_s), and(s29i0, out1_s), s30})
}
if and(fpx, s29i1) != null {
	cSl = append(cSl, g{s29i1, and(s29i1,
		out4_s), and(s29i1, out2_s), and(s29i1, out1_s), s29})
}
if and(fpx, s29i2) != null {
	cSl = append(cSl, g{s29i2, and(s29i2,
		out4_s), and(s29i2, out2_s), and(s29i2, out1_s), s29})
}
if and(fpx, s29i3) != null {
	cSl = append(cSl, g{s29i3, and(s29i3,
		out4_s), and(s29i3, out2_s), and(s29i3, out1_s), s29})
}
if and(fpx, s29i4) != null {
	cSl = append(cSl, g{s29i4, and(s29i4,
		out4_s), and(s29i4, out2_s), and(s29i4, out1_s), s29})
}
if and(fpx, s29i5) != null {
	cSl = append(cSl, g{s29i5, and(s29i5,
		out4_s), and(s29i5, out2_s), and(s29i5, out1_s), s29})
}
if and(fpx, s29i6) != null {
	cSl = append(cSl, g{s29i6, and(s29i6,
		out4_s), and(s29i6, out2_s), and(s29i6, out1_s), s29})
}
if and(fpx, s29i7) != null {
	cSl = append(cSl, g{s29i7, and(s29i7,
		out4_s), and(s29i7, out2_s), and(s29i7, out1_s), s29})
}
//-------------------------------------------------------
if and(fpx, s30i0) != null {
	cSl = append(cSl, g{s30i0, and(s30i0,
		out4_s), and(s30i0, out2_s), and(s30i0, out1_s), s30})
}
if and(fpx, s30i1) != null {
	cSl = append(cSl, g{s30i1, and(s30i1,
		out4_s), and(s30i1, out2_s), and(s30i1, out1_s), s30})
}
if and(fpx, s30i2) != null {
	cSl = append(cSl, g{s30i2, and(s30i2,
		out4_s), and(s30i2, out2_s), and(s30i2, out1_s), s30})
```

```
}
if and(fpx, s30i3) != null {
        cSl = append(cSl, g{s30i3, and(s30i3,
                out4_s), and(s30i3, out2_s), and(s30i3, out1_s), s30})
}
if and(fpx, s30i4) != null {
        cSl = append(cSl, g{s30i4, and(s30i4,
                out4_s), and(s30i4, out2_s), and(s30i4, out1_s), s30})
}
if and(fpx, s30i5) != null {
        cSl = append(cSl, g{s30i5, and(s30i5,
                out4_s), and(s30i5, out2_s), and(s30i5, out1_s), s30})
}
if and(fpx, s30i6) != null {
        cSl = append(cSl, g{s30i6, and(s30i6,
                out4_s), and(s30i6, out2_s), and(s30i6, out1_s), s30})
}
if and(fpx, s30i7) != null {
        cSl = append(cSl, g{s30i7, and(s30i7,
                out4_s), and(s30i7, out2_s), and(s30i7, out1_s), s31})
}
//-------------------------------------------------------
if and(fpx, s31i0) != null {
        cSl = append(cSl, g{s31i0, and(s31i0,
                out4_s), and(s31i0, out2_s), and(s31i0, out1_s), s31})
}
if and(fpx, s31i1) != null {
        cSl = append(cSl, g{s31i1, and(s31i1,
                out4_s), and(s31i1, out2_s), and(s31i1, out1_s), s31})
}
if and(fpx, s31i2) != null {
        cSl = append(cSl, g{s31i2, and(s31i2,
                out4_s), and(s31i2, out2_s), and(s31i2, out1_s), s0})
}
if and(fpx, s31i3) != null {
        cSl = append(cSl, g{s31i3, and(s31i3,
                out4_s), and(s31i3, out2_s), and(s31i3, out1_s), s31})
}
if and(fpx, s31i4) != null {
        cSl = append(cSl, g{s31i4, and(s31i4,
                out4_s), and(s31i4, out2_s), and(s31i4, out1_s), s31})
}
if and(fpx, s31i5) != null {
        cSl = append(cSl, g{s31i5, and(s31i5,
                out4_s), and(s31i5, out2_s), and(s31i5, out1_s), s0})
}
if and(fpx, s31i6) != null {
        cSl = append(cSl, g{s31i6, and(s31i6,
                out4_s), and(s31i6, out2_s), and(s31i6, out1_s), s31})
}
if and(fpx, s31i7) != null {
        cSl = append(cSl, g{s31i7, and(s31i7,
                out4_s), and(s31i7, out2_s), and(s31i7, out1_s), s31})
}
```

```
                //-------------------------------------------------------

                return cSI

        } // end of decompMap()

        dMfp1 := decompMap(fp1, out4_s, out2_s, out1_s, collectedSIs)
        dMfp2 := decompMap(fp2, out4_s, out2_s, out1_s, collectedSIs)
        dMfp3 := decompMap(fp3, out4_s, out2_s, out1_s, collectedSIs)
        dMfp4 := decompMap(fp4, out4_s, out2_s, out1_s, collectedSIs)
        dMfp5 := decompMap(fp5, out4_s, out2_s, out1_s, collectedSIs)
        dMfp6 := decompMap(fp6, out4_s, out2_s, out1_s, collectedSIs)
        dMfp7 := decompMap(fp7, out4_s, out2_s, out1_s, collectedSIs)
        dMfp8 := decompMap(fp8, out4_s, out2_s, out1_s, collectedSIs)
        dMfp9 := decompMap(fp9, out4_s, out2_s, out1_s, collectedSIs)
        dMfp10 := decompMap(fp10, out4_s, out2_s, out1_s, collectedSIs)
        dMfp11 := decompMap(fp11, out4_s, out2_s, out1_s, collectedSIs)
        dMfp12 := decompMap(fp12, out4_s, out2_s, out1_s, collectedSIs)
        dMfp13 := decompMap(fp13, out4_s, out2_s, out1_s, collectedSIs)
        dMfp14 := decompMap(fp14, out4_s, out2_s, out1_s, collectedSIs)
        dMfp15 := decompMap(fp15, out4_s, out2_s, out1_s, collectedSIs)
        dMfp16 := decompMap(fp16, out4_s, out2_s, out1_s, collectedSIs)
        dMfp17 := decompMap(fp17, out4_s, out2_s, out1_s, collectedSIs)
        dMfp18 := decompMap(fp18, out4_s, out2_s, out1_s, collectedSIs)
        dMfp19 := decompMap(fp19, out4_s, out2_s, out1_s, collectedSIs)
        dMfp20 := decompMap(fp20, out4_s, out2_s, out1_s, collectedSIs)
        dMfp21 := decompMap(fp21, out4_s, out2_s, out1_s, collectedSIs)
        dMfp22 := decompMap(fp22, out4_s, out2_s, out1_s, collectedSIs)
        dMfp23 := decompMap(fp23, out4_s, out2_s, out1_s, collectedSIs)
        dMfp24 := decompMap(fp24, out4_s, out2_s, out1_s, collectedSIs)
        dMfp25 := decompMap(fp25, out4_s, out2_s, out1_s, collectedSIs)
        dMfp26 := decompMap(fp26, out4_s, out2_s, out1_s, collectedSIs)
        dMfp27 := decompMap(fp27, out4_s, out2_s, out1_s, collectedSIs)
        dMfp28 := decompMap(fp28, out4_s, out2_s, out1_s, collectedSIs)
        dMfp29 := decompMap(fp29, out4_s, out2_s, out1_s, collectedSIs)
        dMfp30 := decompMap(fp30, out4_s, out2_s, out1_s, collectedSIs)
        dMfp31 := decompMap(fp31, out4_s, out2_s, out1_s, collectedSIs)

        // each dMfp is a slice of g{si, out4, out2, out1, ns} structs
        return dMfp1, dMfp2, dMfp3, dMfp4, dMfp5, dMfp6, dMfp7, dMfp8, dMfp9,
                dMfp10, dMfp11, dMfp12, dMfp13, dMfp14, dMfp15, dMfp16, dMfp17,
                dMfp18, dMfp19, dMfp20, dMfp21, dMfp22, dMfp23, dMfp24, dMfp25,
                dMfp26, dMfp27, dMfp28, dMfp29, dMfp30, dMfp31

} // end of ns2fp() ================================================

// EMD of CORE
=====================================================

// --- SIMULATION Functions ---
// setUP function (from original file)
setUP := func(usi string, first, ns16b, ns8b, ns4b, ns2b, ns1b bool) (bool, bool, bool,
bool, bool, bool, bool, bool) {
        // usi is like "s12i5"
```

```
		var ps16i, ps8i, ps4i, ps2i, ps1i, in4i, in2i, in1i bool

		// Parse state and input numbers
		var s, i int
		n, err := fmt.Sscanf(usi, "s%di%d", &s, &i)
		if n != 2 || err != nil {
			// fallback: all false if parsing fails
			return false, false, false, false, false, false, false, false
		}

		// State bits (5 bits: s0..s31)
		ps16i = (s & 0x10) != 0
		ps8i = (s & 0x08) != 0
		ps4i = (s & 0x04) != 0
		ps2i = (s & 0x02) != 0
		ps1i = (s & 0x01) != 0

		// Input bits (3 bits: i0..i7)
		in4i = (i & 0x04) != 0
		in2i = (i & 0x02) != 0
		in1i = (i & 0x01) != 0

		// Now handle the "first" logic as before
		var ps16a, ps8a, ps4a, ps2a, ps1a bool
		ps16a = (!first && ns16b) || (first && ps16i)
		ps8a = (!first && ns8b) || (first && ps8i)
		ps4a = (!first && ns4b) || (first && ps4i)
		ps2a = (!first && ns2b) || (first && ps2i)
		ps1a = (!first && ns1b) || (first && ps1i)

		return ps16a, ps8a, ps4a, ps2a, ps1a, in4i, in2i, in1i
	}

	// one_set_BOOL function (from original file)
	// uses Boolean values true, false to provide
	// a Boolean simulation of a time-frame

	one_set_BOOL := func(ps16a, ps8a, ps4a, ps2a, ps1a, in4i, in2i, in1i bool, fault_C
string) (bool, bool, bool, bool, bool, bool, bool, bool, string) {

		// Helper function to apply faults
		applyFault := func(value bool, faultKey string) bool {
			if fault_C == faultKey+":0" {
				return false
			}
			if fault_C == faultKey+":1" {
				return true
			}
			return value
		}

		// Receives S/Is via present-state lines ps16_i, ps8_i. ect.
		// Uses fault_C to activate local_fault; propagates local_fault
		// to output and next_state lines, out4, out2, out1,
```

```
// ns16, ns8, etc., as S/ls.

// ----------- Simulation NETLIST ------------
// -- level 0 --
ps16b := applyFault(ps16a, "ps16")
ps8b := applyFault(ps8a, "ps8")
ps4b := applyFault(ps4a, "ps4")
ps2b := applyFault(ps2a, "ps2")
ps1b := applyFault(ps1a, "ps1")
in4b := applyFault(in4i, "in4")
in2b := applyFault(in2i, "in2")
in1b := applyFault(in1i, "in1")
// -- level 1 --
nps1b := applyFault(!ps1b, "nps1")
nps2b := applyFault(!ps2b, "nps2")
nps4b := applyFault(!ps4b, "nps4")
nps8b := applyFault(!ps8b, "nps8")
nps16b := applyFault(!ps16b, "nps16")
nin1b := applyFault(!in1b, "nin1")
nin2b := applyFault(!in2b, "nin2")
nin4b := applyFault(!in4b, "nin4")
i7b := applyFault(in4b && in2b && in1b, "i7")
// -- level 2 --
i0b := applyFault(nin4b && nin2b && nin1b, "i0")
i1b := applyFault(nin4b && nin2b && in1b, "i1")
i2b := applyFault(nin4b && in2b && nin1b, "i2")
i3b := applyFault(nin4b && in2b && in1b, "i3")
i4b := applyFault(in4b && nin2b && nin1b, "i4")
i5b := applyFault(in4b && nin2b && in1b, "i5")
i6b := applyFault(in4b && in2b && nin1b, "i6")
ls0b := applyFault(nps4b && nps2b && nps1b, "ls0")
ls1b := applyFault(nps4b && nps2b && ps1b, "ls1")
ls2b := applyFault(nps4b && ps2b && nps1b, "ls2")
ls3b := applyFault(nps4b && ps2b && ps1b, "ls3")
ls4b := applyFault(ps4b && nps2b && nps1b, "ls4")
ls5b := applyFault(ps4b && nps2b && ps1b, "ls5")
ls6b := applyFault(ps4b && ps2b && nps1b, "ls6")
ls7b := applyFault(ps4b && ps2b && ps1b, "ls7")
ni7b := applyFault(!i7b, "ni7")
s31b := applyFault(ps16b && ps8b && ls7b, "s31")
// level 3
ni0b := applyFault(!i0b, "ni0")
ni1b := applyFault(!i1b, "ni1")
ni2b := applyFault(!i2b, "ni2")
ni3b := applyFault(!i3b, "ni3")
ni5b := applyFault(!i5b, "ni5")
ni6b := applyFault(!i6b, "ni6")
s0b := applyFault(nps16b && nps8b && ls0b, "s0")
s1b := applyFault(nps16b && nps8b && ls1b, "s1")
s2b := applyFault(nps16b && nps8b && ls2b, "s2")
s3b := applyFault(nps16b && nps8b && ls3b, "s3")
s4b := applyFault(nps16b && nps8b && ls4b, "s4")
s5b := applyFault(nps16b && nps8b && ls5b, "s5")
s6b := applyFault(nps16b && nps8b && ls6b, "s6")
```

```
s7b := applyFault(nps16b && nps8b && ls7b, "s7")
s8b := applyFault(nps16b && ps8b && ls0b, "s8")
s9b := applyFault(nps16b && ps8b && ls1b, "s9")
s10b := applyFault(nps16b && ps8b && ls2b, "s10")
s11b := applyFault(nps16b && ps8b && ls3b, "s11")
s12b := applyFault(nps16b && ps8b && ls4b, "s12")
s13b := applyFault(nps16b && ps8b && ls5b, "s13")
s14b := applyFault(nps16b && ps8b && ls6b, "s14")
s15b := applyFault(nps16b && ps8b && ls7b, "s15")
s16b := applyFault(ps16b && nps8b && ls0b, "s16")
s17b := applyFault(ps16b && nps8b && ls1b, "s17")
s18b := applyFault(ps16b && nps8b && ls2b, "s18")
s19b := applyFault(ps16b && nps8b && ls3b, "s19")
s20b := applyFault(ps16b && nps8b && ls4b, "s20")
s21b := applyFault(ps16b && nps8b && ls5b, "s21")
s22b := applyFault(ps16b && nps8b && ls6b, "s22")
s23b := applyFault(ps16b && nps8b && ls7b, "s23")
s24b := applyFault(ps16b && ps8b && ls0b, "s24")
s25b := applyFault(ps16b && ps8b && ls1b, "s25")
s26b := applyFault(ps16b && ps8b && ls2b, "s26")
s27b := applyFault(ps16b && ps8b && ls3b, "s27")
s28b := applyFault(ps16b && ps8b && ls4b, "s28")
s29b := applyFault(ps16b && ps8b && ls5b, "s29")
s30b := applyFault(ps16b && ps8b && ls6b, "s30")
b2b := applyFault(i5b || i3b || i2b, "b2")
b7b := applyFault(i5b || i1b, "b7")
c5b := applyFault(i7b || i5b, "c5")
c13b := applyFault(i3b || i2b, "c13")
e5b := applyFault(i5b || i4b, "e5")
e10b := applyFault(i6b || i2b || i0b, "e10")
e12b := applyFault(i6b || i3b, "e12")
e18b := applyFault(i6b || i2b, "e18")
e24b := applyFault(i7b || i1b, "e24")
// -- level 4 --
a1b := applyFault(s10b && i0b, "a1")
a2b := applyFault(s15b && i5b, "a2")
a3b := applyFault(s18b && ni6b, "a3")
a4b := applyFault(s20b || s21b || s22b, "a4")
a5b := applyFault(s24b && ni7b, "a5")
a6b := applyFault(s25b || s26b, "a6")
a7b := applyFault(s27b || s28b || s29b, "a7")
a9b := applyFault(s31b && ni5b && ni2b, "a9")
b1b := applyFault(s3b && i2b, "b1")
b3b := applyFault(b2b && s7b, "b3")
b4b := applyFault(s10b && ni6b, "b4")
b5b := applyFault(s12b || s13b || s14b, "b5")
b6b := applyFault(s15b && ni5b, "b6")
b8b := applyFault(s23b && b7b, "b8")
b9b := applyFault(s25b || s26b, "b9")
b10b := applyFault(s27b || s28b || s29b, "b10")
c2b := applyFault(s7b && ni5b && ni3b, "c2")
c3b := applyFault(s11b && i7b, "c3")
c4b := applyFault(s15b && ni5b, "c4")
c6b := applyFault(s23b && ni5b, "c6")
```

```
c8b := applyFault(s19b && c5b, "c8")
c14b := applyFault(c13b && s3b, "c14")
c15b := applyFault(s27b && i7b, "c15")
d1b := applyFault(s1b && i2b, "d1")
d2b := applyFault(s3b && ni3b && ni2b, "d2")
d3b := applyFault(s5b && i0b, "d3")
d5b := applyFault(s9b && i2b, "d5")
d6b := applyFault(s11b && ni7b, "d6")
d7b := applyFault(s13b && i0b, "d7")
d9b := applyFault(s15b && ni5b, "d9")
d10b := applyFault(s17b && i2b, "d10")
d11b := applyFault(s19b && ni7b, "d11")
d12b := applyFault(s21b && i0b, "d12")
d13b := applyFault(s23b && ni5b && ni1b, "d13")
d14b := applyFault(s25b && i2b, "d14")
d15b := applyFault(s29b && i0b, "d15")
d27b := applyFault(s27b && ni7b, "d27")
e1b := applyFault(s0b && i1b, "e1")
e2b := applyFault(s1b && ni2b, "e2")
e3b := applyFault(s2b && i2b, "e3")
e4b := applyFault(s3b && ni3b && ni2b, "e4")
e6b := applyFault(s5b && ni0b, "e6")
e7b := applyFault(s6b && i7b, "e7")
e8b := applyFault(s8b && i1b, "e8")
e9b := applyFault(s9b && ni2b, "e9")
e11b := applyFault(s11b && ni7b, "e11")
e13b := applyFault(s13b && ni0b, "e13")
e14b := applyFault(s14b && i7b, "e14")
e15b := applyFault(s15b && ni5b, "e15")
e16b := applyFault(s16b && i1b, "e16")
e17b := applyFault(s17b && ni2b, "e17")
e19b := applyFault(s19b && ni7b, "e19")
e20b := applyFault(s20b && e12b, "e20")
e21b := applyFault(s21b && ni0b, "e21")
e22b := applyFault(s22b && i7b, "e22")
e23b := applyFault(s23b && ni5b, "e23")
e25b := applyFault(s25b && ni2b, "e25")
e26b := applyFault(s26b && i2b, "e26")
e27b := applyFault(s27b && ni7b, "e27")
e28b := applyFault(s28b && e12b, "e28")
e29b := applyFault(s29b && ni0b, "e29")
e30b := applyFault(s30b && i7b, "e30")
e31b := applyFault(s4b && e5b, "e31")
e32b := applyFault(s10b && e10b, "e32")
e33b := applyFault(s12b && e12b, "e33")
e34b := applyFault(s18b && e18b, "e34")
e35b := applyFault(s24b && e24b, "e35")
f1b := applyFault(s12b && i5b, "f1")
f2b := applyFault(s27b && i4b, "f2")
f3b := applyFault(s15b && i0b, "f3")
f4b := applyFault(s27b && i2b, "f4")
f5b := applyFault(s0b && i7b, "f5")
f6b := applyFault(s27b && i1b, "f6")
```

```
// level 5

a8b := applyFault(a7b || s30b, "a8")
a10b := applyFault(a1b || a2b || s16b, "a10")
b11b := applyFault(b10b || s30b, "b11")
b12b := applyFault(b1b || b3b || s8b, "b12")
b13b := applyFault(s9b || b4b || s11b, "b13")
c1b := applyFault(c14b || s4b || s5b, "c1")
c7b := applyFault(c2b && ni2b, "c7")
c16b := applyFault(c15b || s28b || s29b, "c16")
d17b := applyFault(d1b || d2b || d3b, "d17")
d19b := applyFault(s10b || d6b || d7b, "d19")
d20b := applyFault(s14b || d9b || d10b, "d20")
d21b := applyFault(s18b || d11b || d12b, "d2`")
d22b := applyFault(s22b || d13b || d14b, "d22")
d23b := applyFault(s26b || d15b || s30b, "d23")
e36b := applyFault(e1b || e2b || e3b, "e36")
e37b := applyFault(e4b || e31b || e6b, "e37")
e39b := applyFault(e9b || e32b, "r39")
e40b := applyFault(e11b || e33b || e13b, "e40")
e41b := applyFault(e14b || e15b || e16b, "e41")
e42b := applyFault(e17b || e34b || e19b, "e42")
e43b := applyFault(e20b || e30b || a9b, "e43")
e44b := applyFault(e21b || e22b || e23b, "e44")
e45b := applyFault(e35b || e25b || e26b, "e45")
e46b := applyFault(e27b || e28b || e29b, "e46")
out4b := applyFault(f1b || f2b, "out4")
out2b := applyFault(f3b || f4b, "out2")
out1b := applyFault(f5b || f6b, "out1")
// -- level 6 --
a11b := applyFault(a10b || s17b || a3b, "a11")
b14b := applyFault(b12b || b13b || b5b, "b14")
c9b := applyFault(c1b || s6b || c7b, "c9")
c17b := applyFault(c16b || s30b, "c17")
d18b := applyFault(s6b || c7b || d5b, "d18")
d28b := applyFault(d23b || d27b, "d28")
e38b := applyFault(e7b || c7b || e8b, "e38")
e47b := applyFault(e36b || e37b || e44b, "e47")
e49b := applyFault(e39b || e40b || e41b, "e49")
// -- level 7 --
a12b := applyFault(a11b || s19b || a4b, "a12")
b15b := applyFault(b14b || b6b || b8b, "b15")
c10b := applyFault(c9b || c3b || b5b, "c10")
d24b := applyFault(d17b || d18b || d19b, "d24")
e48b := applyFault(e45b || e46b || e38b, "e48")
// -- level 8 --
a13b := applyFault(a12b || s23b || a5b, "a13")
b16b := applyFault(b15b || s24b || b9b, "b16")
c11b := applyFault(c10b || c4b || c8b, "c11")
d25b := applyFault(d24b || d20b || d21b, "d25")
e50b := applyFault(e47b || e48b || e49b, "e50")
// -- level 9 --
ns1b := applyFault(e50b || e42b || e43b, "ns1")
ns8b := applyFault(b16b || b11b || a9b, "ns8")
```

```
a14b := applyFault(a13b || a6b || a8b, "a14")
c12b := applyFault(c11b || a4b || c6b, "c12")
d26b := applyFault(d25b || d22b || d28b, "d26")
// -- level 10 --
ns2b := applyFault(d26b || s2b || a9b, "ns2")
ns4b := applyFault(c12b || c17b || a9b, "ns4")
ns16b := applyFault(a14b || a9b, "ns16")
// -- END OF CIRCUIT NET LIST --

// Determine state
var state string
switch {
case !ns16b && !ns8b && !ns4b && !ns2b && !ns1b:
        state = "s0"
case !ns16b && !ns8b && !ns4b && !ns2b && ns1b:
        state = "s1"
case !ns16b && !ns8b && !ns4b && ns2b && !ns1b:
        state = "s2"
case !ns16b && !ns8b && !ns4b && ns2b && ns1b:
        state = "s3"
case !ns16b && !ns8b && ns4b && !ns2b && !ns1b:
        state = "s4"
case !ns16b && !ns8b && ns4b && !ns2b && ns1b:
        state = "s5"
case !ns16b && !ns8b && ns4b && ns2b && !ns1b:
        state = "s6"
case !ns16b && !ns8b && ns4b && ns2b && ns1b:
        state = "s7"
case !ns16b && ns8b && !ns4b && !ns2b && !ns1b:
        state = "s8"
case !ns16b && ns8b && !ns4b && !ns2b && ns1b:
        state = "s9"
case !ns16b && ns8b && !ns4b && ns2b && !ns1b:
        state = "s10"
case !ns16b && ns8b && !ns4b && ns2b && ns1b:
        state = "s11"
case !ns16b && ns8b && ns4b && !ns2b && !ns1b:
        state = "s12"
case !ns16b && ns8b && ns4b && !ns2b && ns1b:
        state = "s13"
case !ns16b && ns8b && ns4b && ns2b && !ns1b:
        state = "s14"
case !ns16b && ns8b && ns4b && ns2b && ns1b:
        state = "s15"
case ns16b && !ns8b && !ns4b && !ns2b && !ns1b:
        state = "s16"
case ns16b && !ns8b && !ns4b && !ns2b && ns1b:
        state = "s17"
case ns16b && !ns8b && !ns4b && ns2b && !ns1b:
        state = "s18"
case ns16b && !ns8b && !ns4b && ns2b && ns1b:
        state = "s19"
case ns16b && !ns8b && ns4b && !ns2b && !ns1b:
        state = "s20"
```

```
		case ns16b && !ns8b && ns4b && !ns2b && ns1b:
			state = "s21"
		case ns16b && !ns8b && ns4b && ns2b && !ns1b:
			state = "s22"
		case ns16b && !ns8b && ns4b && ns2b && ns1b:
			state = "s23"
		case ns16b && ns8b && !ns4b && !ns2b && !ns1b:
			state = "s24"
		case ns16b && ns8b && !ns4b && !ns2b && ns1b:
			state = "s25"
		case ns16b && ns8b && !ns4b && ns2b && !ns1b:
			state = "s26"
		case ns16b && ns8b && !ns4b && ns2b && ns1b:
			state = "s27"
		case ns16b && ns8b && ns4b && !ns2b && !ns1b:
			state = "s28"
		case ns16b && ns8b && ns4b && !ns2b && ns1b:
			state = "s29"
		case ns16b && ns8b && ns4b && ns2b && !ns1b:
			state = "s30"
		case ns16b && ns8b && ns4b && ns2b && ns1b:
			state = "s31"
		}

		return out4b, out2b, out1b, ns16b, ns8b, ns4b, ns2b, ns1b, state

	}

	// End of one_set_BOOL() ========================================

	// END of SIMULATION Functions ==================================

	// SEARCH Functions =============================================

	// peek
============================================================
	//
============================================================

	peek := func(fp int, s nd, fault_A string, fault_C string,
		allSATFunc func(nd, func(string) nd) []string,
		nd2strFunc func(nd) string, siMap map[string]SIMapping) map[string]SIMapping
{

		// Helper function to convert `fp` to keys
		convertKeys := func(fp int, sx nd) (nd, nd, nd, nd, nd) {
			var keys [5]nd
			for i := 0; i < 5; i++ {
				if fp&(1<<i) != 0 {
					keys[4-i] = sx
				} else {
					keys[4-i] = null
				}
			}
```

```
		return keys[0], keys[1], keys[2], keys[3], keys[4]
	}

	// Convert `fp, s` to keys
	pt16, pt8, pt4, pt2, pt1 := convertKeys(fp, s)

	// ps2ns
==========================================================
	out4_s, out2_s, out1_s, ns16_s, ns8_s, ns4_s, ns2_s, ns1_s :=
		ps2ns(pt16, pt8, pt4, pt2, pt1, fault_A)
	//
============================================================

	// ns2fp
==========================================================
	dMfp1, dMfp2, dMfp3, dMfp4, dMfp5, dMfp6, dMfp7, dMfp8, dMfp9,
		dMfp10, dMfp11, dMfp12, dMfp13, dMfp14, dMfp15, dMfp16, dMfp17,
		dMfp18, dMfp19, dMfp20, dMfp21, dMfp22, dMfp23, dMfp24, dMfp25,
		dMfp26, dMfp27, dMfp28, dMfp29, dMfp30, dMfp31 :=
		ns2fp(out4_s, out2_s, out1_s, ns16_s, ns8_s, ns4_s, ns2_s, ns1_s)
	//
============================================================

	lns := or(pt1, or(pt2, or(pt4, or(pt8, pt16))))
	// siMap := make(map[string]SIMapping)

	//fmt.Println("-----------------------------------------------")
	//fmt.Println("circuit: Large,  fault_A: ", fault_A, ",  fault_C: ", fault_C)
	//fmt.Println("fp: ", fp, ", s: ", nd2str(s))
	//fmt.Println("-----------------------------------------------")
	//
fmt.Printf("============================================================\n")

	// Modify dispdM to accept nd2str as a parameter and populate the S/I mapping
	dispdM := func(dMfpx S, lns nd, fpn int, allSATFunc func(f nd,
		str2nd func(string) nd) []string,
		nd2strFunc func(nd) string,
		siMap map[string]SIMapping) map[string]SIMapping {

		a := len(dMfpx)
		if a != 0 {
			for i := 0; i < a; i++ {

				if !eq(and(lns, dMfpx[i].si), null) || eq(lns, null) {
					siSlice := allSAT(dMfpx[i].si, str2nd)
					siStr := ""
					if len(siSlice) > 0 {
						siStr = siSlice[0] // Extract the actual
						//  S/I string without brackets
					}
					nsStr := nd2str(dMfpx[i].ns)

					// Store the mapping of S/I to fp and ns
					// (using nd type directly)
```

```
                                        if siStr != "" {
                                                siMap[siStr] = SIMapping{
                                                        fp: fpn,
                                                        ns: dMfpx[i].ns, // Store nd directly,
                                                        //  no conversion needed
                                                }
                                        }

                                        fmt.Println(allSAT(dMfpx[i].si, str2nd), "  ",
                                                allSAT(dMfpx[i].out_4, str2nd),
                                                allSAT(dMfpx[i].out_2, str2nd),
                                                allSAT(dMfpx[i].out_1, str2nd),
                                                "  fp =", fpn, "  ns =", nsStr)
                                }
                        }
                }
                return siMap // Return the updated siMap
}

// Now call dispdM and pass nd2str as an argument
siMap = dispdM(dMfp1, lns, 1, allSAT, nd2str, siMap)
siMap = dispdM(dMfp2, lns, 2, allSAT, nd2str, siMap)
siMap = dispdM(dMfp3, lns, 3, allSAT, nd2str, siMap)
siMap = dispdM(dMfp4, lns, 4, allSAT, nd2str, siMap)
siMap = dispdM(dMfp5, lns, 5, allSAT, nd2str, siMap)
siMap = dispdM(dMfp6, lns, 6, allSAT, nd2str, siMap)
siMap = dispdM(dMfp7, lns, 7, allSAT, nd2str, siMap)
siMap = dispdM(dMfp8, lns, 8, allSAT, nd2str, siMap)
siMap = dispdM(dMfp9, lns, 9, allSAT, nd2str, siMap)
siMap = dispdM(dMfp10, lns, 10, allSAT, nd2str, siMap)
siMap = dispdM(dMfp11, lns, 11, allSAT, nd2str, siMap)
siMap = dispdM(dMfp12, lns, 12, allSAT, nd2str, siMap)
siMap = dispdM(dMfp13, lns, 13, allSAT, nd2str, siMap)
siMap = dispdM(dMfp14, lns, 14, allSAT, nd2str, siMap)
siMap = dispdM(dMfp15, lns, 15, allSAT, nd2str, siMap)
siMap = dispdM(dMfp16, lns, 16, allSAT, nd2str, siMap)
siMap = dispdM(dMfp17, lns, 17, allSAT, nd2str, siMap)
siMap = dispdM(dMfp18, lns, 18, allSAT, nd2str, siMap)
siMap = dispdM(dMfp19, lns, 19, allSAT, nd2str, siMap)
siMap = dispdM(dMfp20, lns, 20, allSAT, nd2str, siMap)
siMap = dispdM(dMfp21, lns, 21, allSAT, nd2str, siMap)
siMap = dispdM(dMfp22, lns, 22, allSAT, nd2str, siMap)
siMap = dispdM(dMfp23, lns, 23, allSAT, nd2str, siMap)
siMap = dispdM(dMfp24, lns, 24, allSAT, nd2str, siMap)
siMap = dispdM(dMfp25, lns, 25, allSAT, nd2str, siMap)
siMap = dispdM(dMfp26, lns, 26, allSAT, nd2str, siMap)
siMap = dispdM(dMfp27, lns, 27, allSAT, nd2str, siMap)
siMap = dispdM(dMfp28, lns, 28, allSAT, nd2str, siMap)
siMap = dispdM(dMfp29, lns, 29, allSAT, nd2str, siMap)
siMap = dispdM(dMfp30, lns, 30, allSAT, nd2str, siMap)
siMap = dispdM(dMfp31, lns, 31, allSAT, nd2str, siMap)

return siMap // Return the S/I mapping
```

```
	} // end of peek() ================================================

	//
=================================================================
	// --- New Material: Copilot.go ---
	//
=================================================================

	// I believe that from the top down to here Copilot.go is identical to Buckley.go.
	// Buckley.go uses foo() to organize the above functions in a combined search and
	// simulation operation. Copilot.go, on the other hand, separates 'search' and
	// 'simulation,' using the above functions in a different manner.

	// Function to reset accumulated S/I sequence
	resetAccumulatedSI := func() {
		accumulatedSI.entries = []string{}
		fmt.Println("S/I sequence reset.")
	}

	// Function to add S/I to accumulated sequence
	addToAccumulatedSI := func(si string) {
		accumulatedSI.entries = append(accumulatedSI.entries, si)
		fmt.Printf("Added %s to sequence. Total entries: %d\n",
			si, len(accumulatedSI.entries))
	}

	// --- a SIMULATION of a single Time-Frame ---
	// Simulation function for a single S/I with specific fault
	// inputs: S/I, fault_C
	// outuputs: SimResult
	simulateSI := func(si string, fault_C string) SimResult {

		fmt.Println("Simulating S/I:", si, "with fault:", fault_C)
		// var first, ns16h, ns8h, ns4h, ns2h, ns1h = true, false, false, false, false, false

		// Parse S/I string to extract state and input numbers
		var stateNum, inputNum int
		fmt.Sscanf(si, "s%di%d", &stateNum, &inputNum)

		// Perform simulation using setUP and one_set_BOOL
		// setUP := func(usi string, first, ns16b, ns8b, ns4b, ns2b, ns1b bool)
		ps16a, ps8a, ps4a, ps2a, ps1a, in4i, in2i, in1i := setUP(si, first, ns16h, ns8h,
ns4h, ns2h, ns1h)
		// one_set_BOOL := func(ps16a, ps8a, ps4a, ps2a, ps1a, in4i, in2i, in1i
bool,fault_C string)
		out4b, out2b, out1b, ns16b, ns8b, ns4b, ns2b, ns1b, state :=
one_set_BOOL(ps16a, ps8a, ps4a, ps2a, ps1a, in4i, in2i, in1i, fault_C)

		// Establish next state based on current state
		ns16h = ns16b
		ns8h = ns8b
		ns4h = ns4b
		ns2h = ns2b
		ns1h = ns1b
```

```
		first = false

		// Format outputs and next state
		outputs := fmt.Sprintf("out4:%t out2:%t out1:%t", out4b, out2b, out1b)
		nextState := state // Use the state return value instead of individual bits

		return SimResult{
			si:        si,
			outputs:   outputs,
			nextState: nextState,
		}
	}

	// END of SIMULATION operation

	// Function to get all possible faults in the circuit
	getAllPossibleFaults := func() []string {
		// Return list of all signal names with :0 and :1 faults
		// --- NEEDS ALL THE FAULTS IN ps2ns() ---   <<<
======================
		signals := []string{
			//--- level 0 ---
			"ps16", "ps8", "ps4", "ps2", "ps1", "in4", "in2", "in1",
			//--- level 1 ---
			"nps16", "nps8", "nps4", "nps2", "nps1", "nin4", "nin2", "nin1", "i7",
			//--- level 2 ---
			"i0", "i1", "i2", "i3", "i4", "i5", "i6", "ls0", "ls1", "ls2", "ls3",
			"ls4", "ls5", "ls6", "ls7", "ni7", "s31",
			//--- level 3 ---
			"ni0", "ni1", "ni2", "ni3", "ni5", "ni6", "s0", "s1", "s2", "s3",
			"s4", "s5", "s6", "s7", "s8", "s9", "s10", "s11", "s12", "s13",
			"s14", "s15", "s16", "s17", "s18", "s19", "s20", "s21", "s22", "s23",
			"s24", "s25", "s26", "s27", "s28", "s29", "s30", "b2", "b7", "c5",
			"c13", "e5", "e10", "e12", "e18", "e24",
			//--- level 4 ---
			"a1", "a2", "a3", "a4", "a5", "a6", "a7", "a9", "b1", "b3", "b4",
			"b5", "b6", "b8", "b9", "b10", "c2", "c3", "c4", "c6", "c8", "c14",
			"c15", "d1", "d2", "d3", "d5", "d6", "d7", "d9", "d10", "d11", "d12",
			"d13", "d14", "d15", "d27", "e1", "e2", "e3", "e4", "e6", "e7", "e8",
			"e9", "e11", "e13", "e14", "e15", "e16", "e17", "e19", "e20", "e21",
			"e22", "e23", "e25", "e26", "e27", "e28", "e29", "e30", "e31", "e32",
			"e33", "e34", "e35", "f1", "f2", "f3", "f4", "f5", "f6",
			//--- level 5 ---
			"a8", "a10", "b11", "b12", "b13", "c1", "c7", "c16", "d17", "d19",
			"d20", "d21", "d22", "d23", "e36", "e37", "e39", "e40", "e41", "e42",
			"e43", "e44", "e45", "e46", "out4", "out2", "out1",
			//--- level 6 ---
			"a11", "b14", "c9", "c17", "d18", "d28", "e38", "e47", "e49",
			//--- level 7 ---
			"a12", "b15", "c10", "d24", "e48",
			//--- level 8 ---
			"a13", "b16", "c11", "d25", "e50",
			//--- level 9 ---
			"ns1", "ns8", "a14", "c12", "d26",
```

```
                //--- level 10 ---
                "ns2", "ns4", "ns16",
        }

        var faults []string
        for _, signal := range signals {
                faults = append(faults, signal+":0")
                faults = append(faults, signal+":1")
        }
        return faults
}

// Function to check if fault exists in circuit
isValidFault := func(fault string) bool {
        allFaults := getAllPossibleFaults()
        for _, f := range allFaults {
                if f == fault {
                        return true
                }
        }
        return false
}

// Simulation phase choice xx1: fault-free simulation
simulationPhaseXX1 := func() {
        fmt.Println("\n=== SIMULATION PHASE XX1: Fault-Free Circuit ===")
        fmt.Println("Simulating accumulated S/I sequence with no faults (fault_C = \"\")")
        fmt.Println("first, ns16h, ns8h, ns4h, ns2h, ns1h =", first, ns16h, ns8h, ns4h,
ns2h, ns1h)

        if len(accumulatedSI.entries) == 0 {
                fmt.Println("No S/I sequence accumulated.")
                return
        }

        fmt.Println("\nPress ENTER to step through each S/I simulation:")
        for i, si := range accumulatedSI.entries {
                fmt.Printf("\nStep %d: %s\n", i+1, si)
                fmt.Scanln() // Wait for ENTER

                result := simulateSI(si, "")
                fmt.Println("========================================")
                fmt.Printf("  Outputs: %s\n", result.outputs)
                fmt.Printf("  Next State: %s\n", result.nextState)
                fmt.Println("========================================")
        }
        fmt.Println("\nFault-free simulation complete.")
}

// Simulation phase choice xx2: fault_A simulation
simulationPhaseXX2 := func() {
        fmt.Println("\n=== SIMULATION PHASE XX2: Faulty Circuit (fault_A) ===")
        fmt.Printf("Simulating accumulated S/I sequence with fault_C = %s\n",
originalFaultA)
```

```
	if len(accumulatedSI.entries) == 0 {
		fmt.Println("No S/I sequence accumulated.")
		return
	}

	fmt.Println("\nPress ENTER to step through each S/I simulation:")
	for i, si := range accumulatedSI.entries {
		fmt.Printf("\nStep %d: %s\n", i+1, si)
		fmt.Scanln() // Wait for ENTER

		fmt.Println("S/I:", si, "fault_C:", originalFaultA)
		result := simulateSI(si, originalFaultA)
		fmt.Println("=========================================")
		fmt.Printf("  Outputs: %s\n", result.outputs)
		fmt.Printf("  Next State: %s\n", result.nextState)
		fmt.Println("=========================================")
	}
	fmt.Println("\nFault_A simulation complete.")
}

// Simulation phase choice xx3: user-provided fault list
simulationPhaseXX3 := func() {
	fmt.Println("\n=== SIMULATION PHASE XX3: User-Provided Fault List ===")

	if len(accumulatedSI.entries) == 0 {
		fmt.Println("No S/I sequence accumulated.")
		return
	}

	// Get fault list from user
	fmt.Println("Enter fault list (format: signal:0 or signal:1, separated by spaces):")
	fmt.Print("Faults: ")
	var faultListStr string
	fmt.Scanln(&faultListStr)

	faultList := strings.Fields(faultListStr)

	// Validate faults
	var validFaults []string
	for _, fault := range faultList {
		if isValidFault(fault) {
			validFaults = append(validFaults, fault)
		} else {
			fmt.Printf("Warning: Invalid fault %s ignored.\n", fault)
		}
	}

	if len(validFaults) == 0 {
		fmt.Println("No valid faults provided.")
		return
	}

	// Simulate fault-free and fault_A responses for comparison
```

```
	var faultFreeResults []SimResult
	var faultAResults []SimResult

	for _, si := range accumulatedSI.entries {
		faultFreeResults = append(faultFreeResults, simulateSI(si, ""))
		faultAResults = append(faultAResults, simulateSI(si, originalFaultA))
	}

	// Categorize each fault
	sameFaultFree := []string{}
	sameFaultA := []string{}
	different := []string{}

	for _, fault := range validFaults {
		var faultResults []SimResult
		for _, si := range accumulatedSI.entries {
			faultResults = append(faultResults, simulateSI(si, fault))
		}

		// Compare with fault-free and fault_A
		matchesFaultFree := true
		matchesFaultA := true

		for i := range faultResults {
			if faultResults[i].outputs != faultFreeResults[i].outputs ||
				faultResults[i].nextState != faultFreeResults[i].nextState {
				matchesFaultFree = false
			}
			if faultResults[i].outputs != faultAResults[i].outputs ||
				faultResults[i].nextState != faultAResults[i].nextState {
				matchesFaultA = false
			}
		}

		if matchesFaultFree {
			sameFaultFree = append(sameFaultFree, fault)
		} else if matchesFaultA {
			sameFaultA = append(sameFaultA, fault)
		} else {
			different = append(different, fault)
		}
	}

	// Display categorization results
	fmt.Printf("\nCategorization Results:\n")
	fmt.Printf("Faults with same response as fault-free: %v\n", sameFaultFree)
	fmt.Printf("Faults with same response as fault_A: %v\n", sameFaultA)
	fmt.Printf("Faults with different responses: %v\n", different)
}

// Simulation phase choice xx4: all possible faults
simulationPhaseXX4 := func() {
	fmt.Println("\n=== SIMULATION PHASE XX4: All Possible Faults ===")
	fmt.Println("Testing every possible fault in the circuit...")
```

```
if len(accumulatedSI.entries) == 0 {
	fmt.Println("No S/I sequence accumulated.")
	return
}

allFaults := getAllPossibleFaults()
fmt.Printf("Testing %d possible faults...\n", len(allFaults))

// Simulate fault-free and fault_A responses for comparison
var faultFreeResults []SimResult
var faultAResults []SimResult

for _, si := range accumulatedSI.entries {
	faultFreeResults = append(faultFreeResults, simulateSI(si, ""))
	faultAResults = append(faultAResults, simulateSI(si, originalFaultA))
}

// Categorize all faults
sameFaultFree := []string{}
sameFaultA := []string{}
different := []string{}

for _, fault := range allFaults {
	var faultResults []SimResult
	for _, si := range accumulatedSI.entries {
		faultResults = append(faultResults, simulateSI(si, fault))
	}

	// Compare with fault-free and fault_A
	matchesFaultFree := true
	matchesFaultA := true

	for i := range faultResults {
		if faultResults[i].outputs != faultFreeResults[i].outputs ||
			faultResults[i].nextState != faultFreeResults[i].nextState {
			matchesFaultFree = false
		}
		if faultResults[i].outputs != faultAResults[i].outputs ||
			faultResults[i].nextState != faultAResults[i].nextState {
			matchesFaultA = false
		}
	}

	if matchesFaultFree {
		sameFaultFree = append(sameFaultFree, fault)
	} else if matchesFaultA {
		sameFaultA = append(sameFaultA, fault)
	} else {
		different = append(different, fault)
	}
}

// Display categorization results
```

```
		fmt.Printf("\nAll Faults Categorization Results:\n")
		fmt.Printf("Faults with same response as fault-free (%d): %v\n",
			len(sameFaultFree), sameFaultFree)
		fmt.Printf("Faults with same response as fault_A (%d): %v\n",
			len(sameFaultA), sameFaultA)
		fmt.Printf("Faults with different responses (%d): %v\n",
			len(different), different)
	}

	//
============================================================

	// Modified main loop with separated phases
	var fault_A, fault_C string

	// Initialize --- THIS IS AN EQIVALENT OF FOO() IN THE ORIGINAL CODE
	resetAccumulatedSI()

	// clear the siMap

outerLoop:
	for {
		// Get fault_A and fault_C from user
		fmt.Print("\nEnter fault_A (leave empty to exit): ")
		fmt.Scanln(&fault_A)
		if fault_A == "" {
			break
		}
		originalFaultA = fault_A

		fmt.Print("Enter fault_C (leave empty for search phase): ")
		fmt.Scanln(&fault_C)

		if fault_C == "" {
			// SEARCH PHASE
			fmt.Println("\n=== SEARCH PHASE ===")
			fmt.Printf("Searching for test sequence for fault_A: %s\n", fault_A)

			var fp int = 0                  // Fault position
			var ns nd = s0                    // next-state
			siMap := make(map[string]SIMapping) // Reset S/I mapping

			for {
				fmt.Println("fp =", fp, ", ns =", nd2str(ns))
				// --- peek() --- Get S/I mapping for current fault_A and fault_C
				siMap = peek(fp, ns, fault_A, fault_C, allSAT, nd2str, siMap)
				// ------------------------------------------------------------

				// Display current accumulated sequence
				if len(accumulatedSI.entries) > 0 {
					fmt.Printf("\nCurrent S/I sequence (%d entries):\n",
						len(accumulatedSI.entries))
					for i, si := range accumulatedSI.entries {
						fmt.Printf("  %d: %s\n", i+1, si)
```

```
                }
            }

            // Get S/I selection from user
            fmt.Println("Select an S/I from the displayed sequence")
            fmt.Print("\nEnter selected S/I (example: s12i5) or control (999/
xx1/xx2/xx3/xx4): ")
            var input string
            fmt.Scanln(&input)

            switch input {
            case "999":
                // Reset and return to search phase
                resetAccumulatedSI()
                fp = 0
                ns = s0
                fault_A = "" // Reset fault_A
                fault_C = ""
                siMap = make(map[string]SIMapping) // Reset S/I
mapping
                fmt.Println("Resetting to search phase.")
                break outerLoop

            case "xx1":
                // Simulation phase: fault-free
                fmt.Println("fp:", fp, ", ns =", nd2str(ns))
                fmt.Println("first, ns16h, ns8h, ns4h, ns2h, ns1h =", first,
ns16h, ns8h, ns4h, ns2h, ns1h)
                simulationPhaseXX1()

            case "xx2":
                // Simulation phase: fault_A
                fmt.Println("fp:", fp, ", ns =", nd2str(ns))
                fmt.Println("first, ns16h, ns8h, ns4h, ns2h, ns1h =", first,
ns16h, ns8h, ns4h, ns2h, ns1h)
                simulationPhaseXX2()

            case "xx3":
                // Simulation phase: user fault list
                fmt.Println("fp:", fp, ", ns =", nd2str(ns))
                fmt.Println("first, ns16h, ns8h, ns4h, ns2h, ns1h =", first,
ns16h, ns8h, ns4h, ns2h, ns1h)
                simulationPhaseXX3()

            case "xx4":
                // Simulation phase: all faults
                fmt.Println("fp:", fp, ", ns =", nd2str(ns))
                fmt.Println("first, ns16h, ns8h, ns4h, ns2h, ns1h =", first,
ns16h, ns8h, ns4h, ns2h, ns1h)
                simulationPhaseXX4()

            default:
                // Reset and return to search phase || assumes input is S/I
                // addToAccumulatedSI(input)
```

```
                                        // fmt.Println("Resetting to search phase.")

                                        // Parse S/I input
                                        if strings.HasPrefix(input, "s") && strings.Contains(input,
"i") {
                                                // Validate S/I format
                                                var testS, testI int
                                                if n, _ := fmt.Sscanf(input, "s%di%d", &testS,
&testI); n == 2 {
                                                        if testS >= 0 && testS <= 31 && testI >= 0
&& testI <= 7 {
                                                                // Look up the correct fp and ns
from siMap FIRST
                                                                if mapping, exists := siMap[input];
exists {
                                                                        // Only add to accumulated
S/I if it's displayed
                                                                        addToAccumulatedSI(input)
                                                                        fp = mapping.fp
                                                                        ns = mapping.ns
                                                                        fmt.Printf("Retrieved from
siMap: %s -> fp=%d\n", input, fp)
                                                                        // Now continue to the top of
the for loop
                                                                        continue
                                                                } else {
                                                                        fmt.Printf("Invalid S/I: %s is
not one of the displayed options.\n", input)
                                                                        fmt.Println("Please select
one of the displayed S/I values.")
                                                                }
                                                        } else {
                                                                fmt.Println("Invalid S/I range. State:
0-31, Input: 0-7")
                                                        }
                                                } else {
                                                        fmt.Println("Invalid S/I format. Example:
s12i5")
                                                }
                                        } else {
                                                fmt.Println("Invalid input. Example S/I format
(s12i5) or control codes (999/xx1/xx2/xx3/xx4)")
                                        }

                                }
                        }
                } else {
                        // Direct simulation mode (original behavior)
                        fmt.Printf("Direct simulation mode: fault_A=%s, fault_C=%s\n", fault_A,
fault_C)
                        // Implement direct simulation if needed
                }
        }
```

```
} // end of main()

// END END END ============================================================
```