



# Self-Healing Approaches for FPGAs and Wiring Manifolds

Sarah Thompson

USRA/RIACS, NASA Ames Research Center

[thompson@email.arc.nasa.gov](mailto:thompson@email.arc.nasa.gov)

Alan Mycroft

University of Cambridge

Guillaume Brat

USRA/RIACS, NASA Ames

Arnaud Venet

Kestrel Technologies



# Repairing FPGAs with SAT Solvers



# FPGAs

FPGA = Field Programmable Gate Array

Offer many of the advantages of full-custom ASICs

- High density, >10 million gates per chip is possible
- Mass savings roughly equivalent to similar gate count ASIC (depends mostly on package)

Disadvantages:

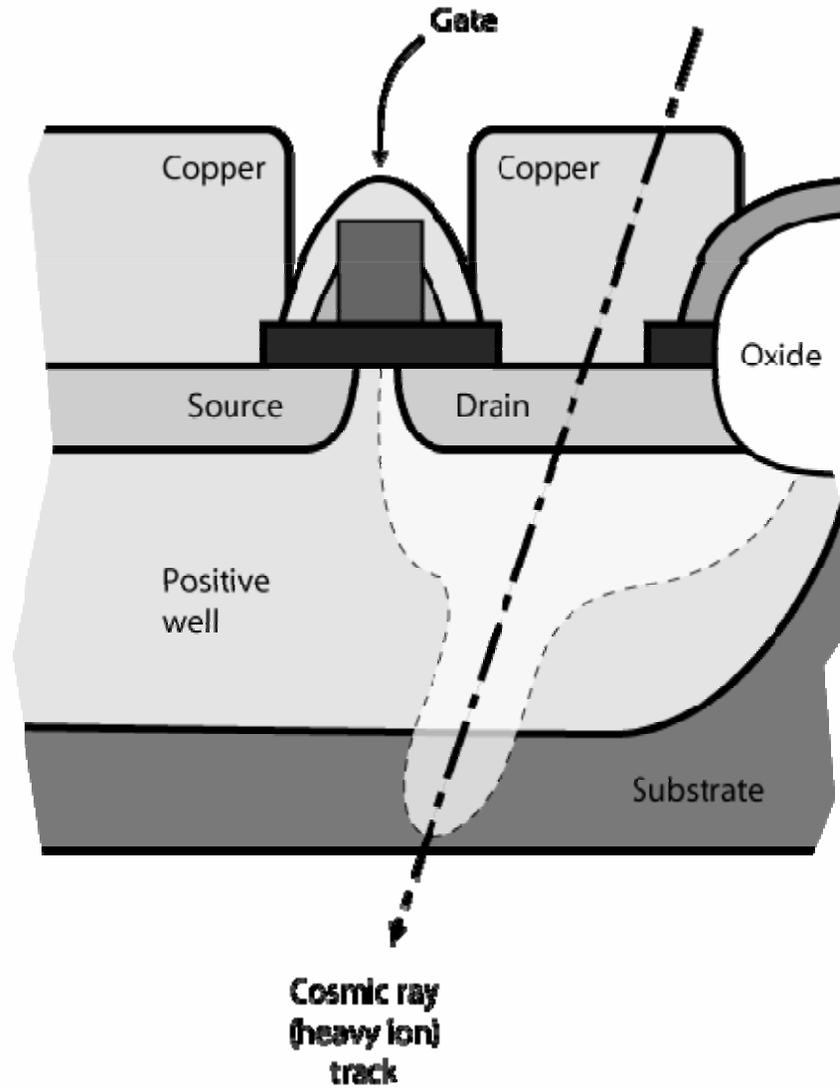
- Order of magnitude worse power consumption
- Order of magnitude slower

But...

- Extremely cheap (no NRE charges whatsoever)
- Can be reconfigured in-flight



# Cosmic ray damage





# Single-Event Effects

(Relatively) low energy impacts can cause Single Event Effects (SEUs and SETs)

One or more transistors turn on for a brief time, causing a voltage spike to be propagated around the circuit

## Features:

- Many SEEs are harmless and do not affect functionality
- Some may alter memory contents, or corrupt state machines
- SEEs affecting critical control signals must be carefully handled



# Permanent Damage

Higher energy impacts can cause *permanent damage*

## Various mechanisms

- Gate rupture due to latch-up
- Failure due to long term total dose effects

## May manifest as:

- Stuck at 1
- Stuck at 0
- Power to ground short



# Radiation Hardening Techniques

## Non-Standard Fab

- Large geometry transistors are less susceptible to damage
- Wafer processing can be tweaked to reduce susceptibility

## Internally-Redundant Standard Cell

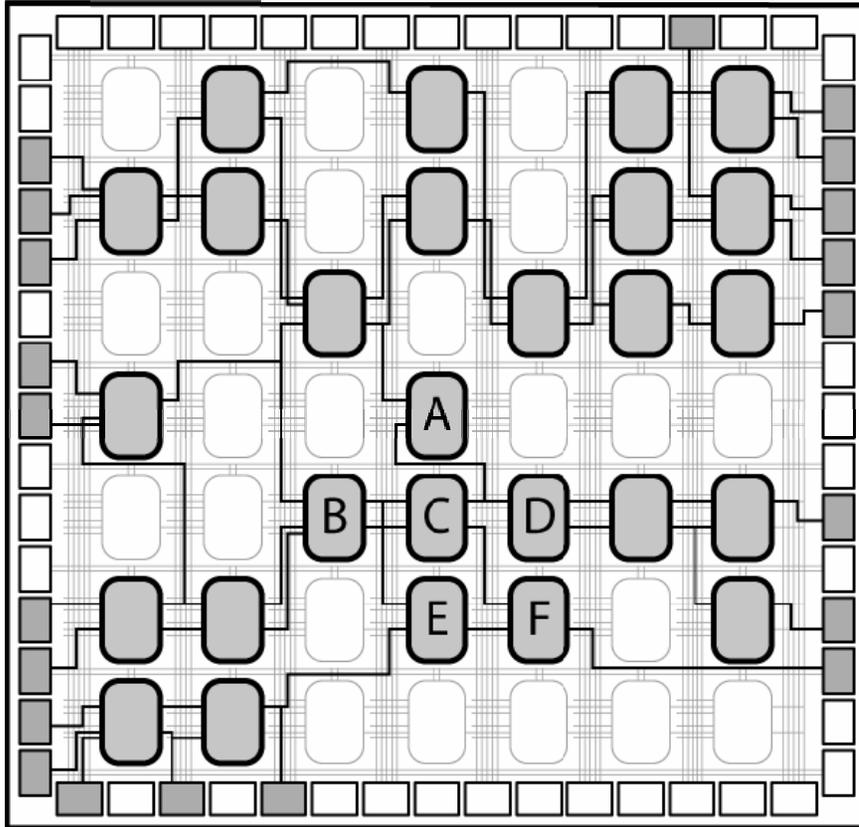
- Rather than e.g. 2 FETs per inverter, use 4, 6 or more
- Reduces performance, increases power consumption
- Works well (e.g. RAD6000, RAD750)

## FPGA Dynamic Reconfiguration

- New approach
- Uses internal redundancy in FPGAs to work around permanent damage



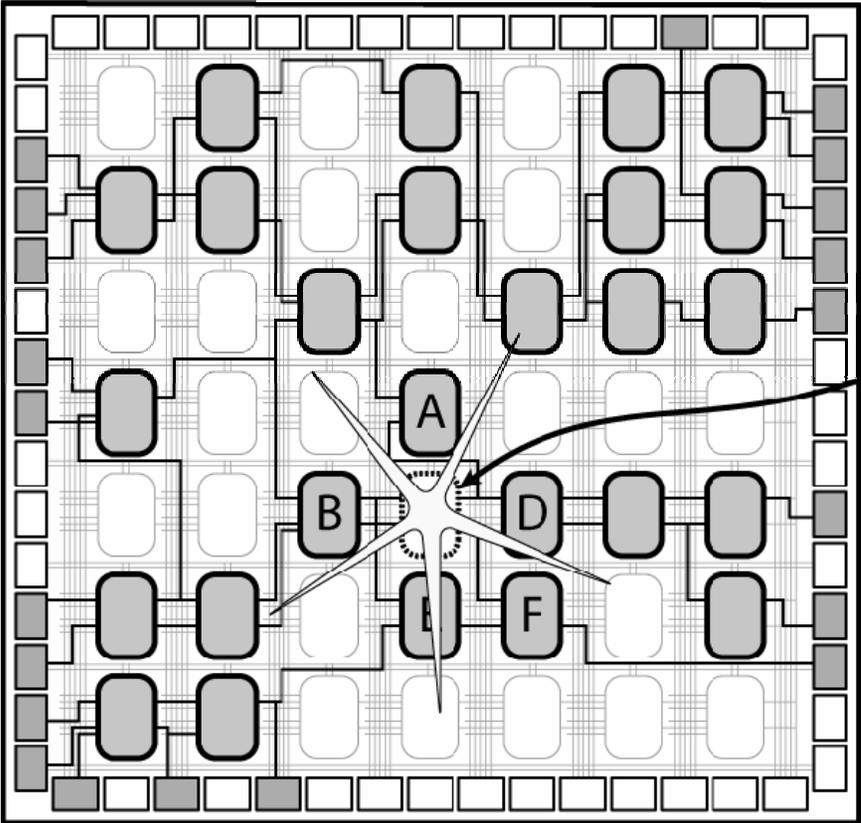
# Undamaged FPGA



Undamaged FPGA



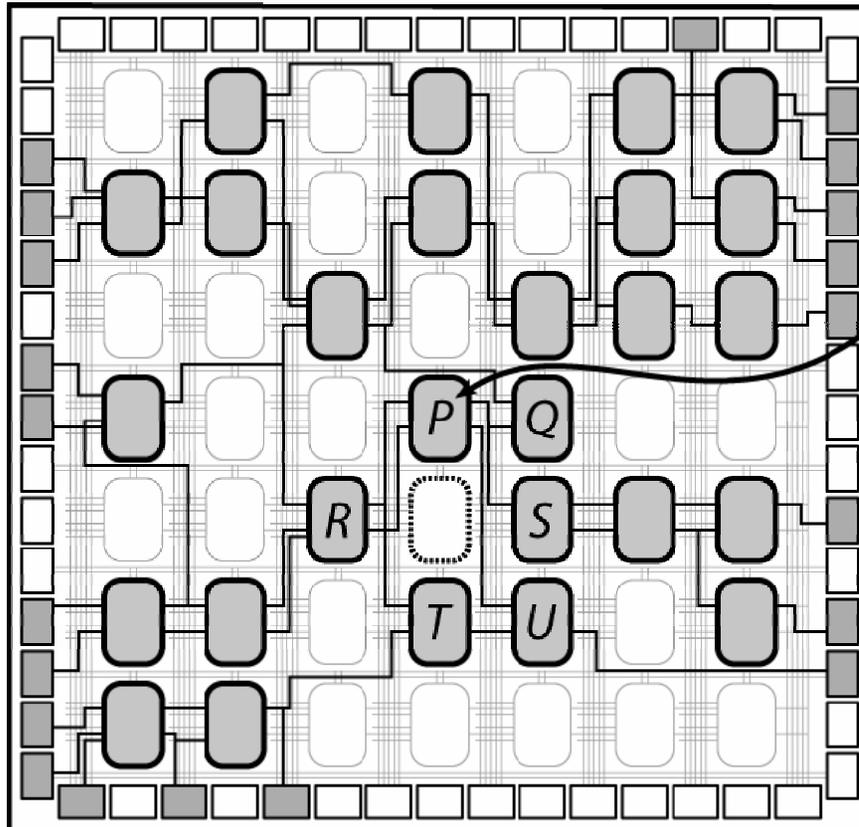
# Cosmic Ray Impact



Cosmic ray (heavy ion)  
impact permanently  
damages logic resource



# Repaired FPGA



Functionality of damaged resource implemented in redundant, previously unused logic by local resynthesis

Note that the resulting look up tables need not resemble the original versions – local resynthesis does not just naïvely move blocks and reroute wires



# The Genetic Approach

Several groups (e.g. Lohn at Ames, Stoica at JPL) have used genetic algorithms to 'evolve around' damage.

Results are encouraging

There are drawbacks, however:

- Very CPU intensive
- Soundness problem



# The SAT Approach

David Greaves at Cambridge has suggested the use of *SAT solvers* for generating FPGA layouts

**The big idea:** since FPGA layout is thought to be *NP*-complete, why not attack the problem with a SAT solver?

We propose modifying this approach in order to generate work-around configurations for damaged FPGAs.



# Boolean SAT in a Nutshell

It is known that all  $NP$ -complete problems can be converted to Boolean SAT problems in  $P$  time.

Convert your problem into a Boolean expression such that:

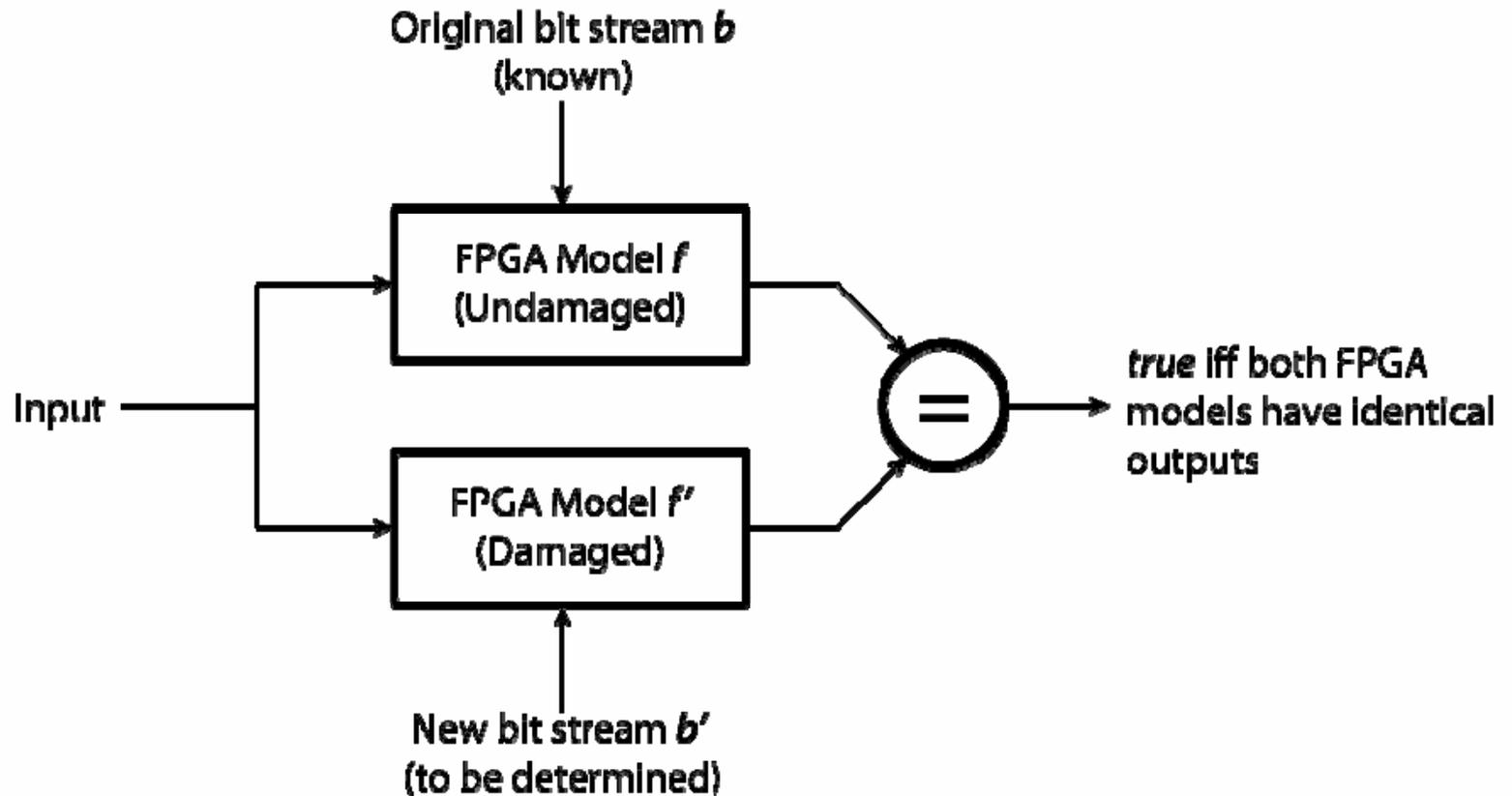
- If a solution exists, some set of variable bindings will cause the expression to evaluate to *true*
- If no solution exists, the expression evaluates to *false* for all possible bindings

Usually:

- If a solution is possible, the variable bindings themselves represent the solution in some way



# FPGA Repair as a SAT Problem





# FPGA Repair as a SAT Problem

$$\forall i . f(b, i) \Leftrightarrow f'(b', i)$$

Where

- $i$  is the inputs of the slice under repair
- $f$  is a model of a correctly functioning FPGA
- $f'$  is a model of the damaged FPGA
- $b$  is the original bit stream
- $b'$  is the derived work-around bit stream



# Proof-of-Concept Demonstrator

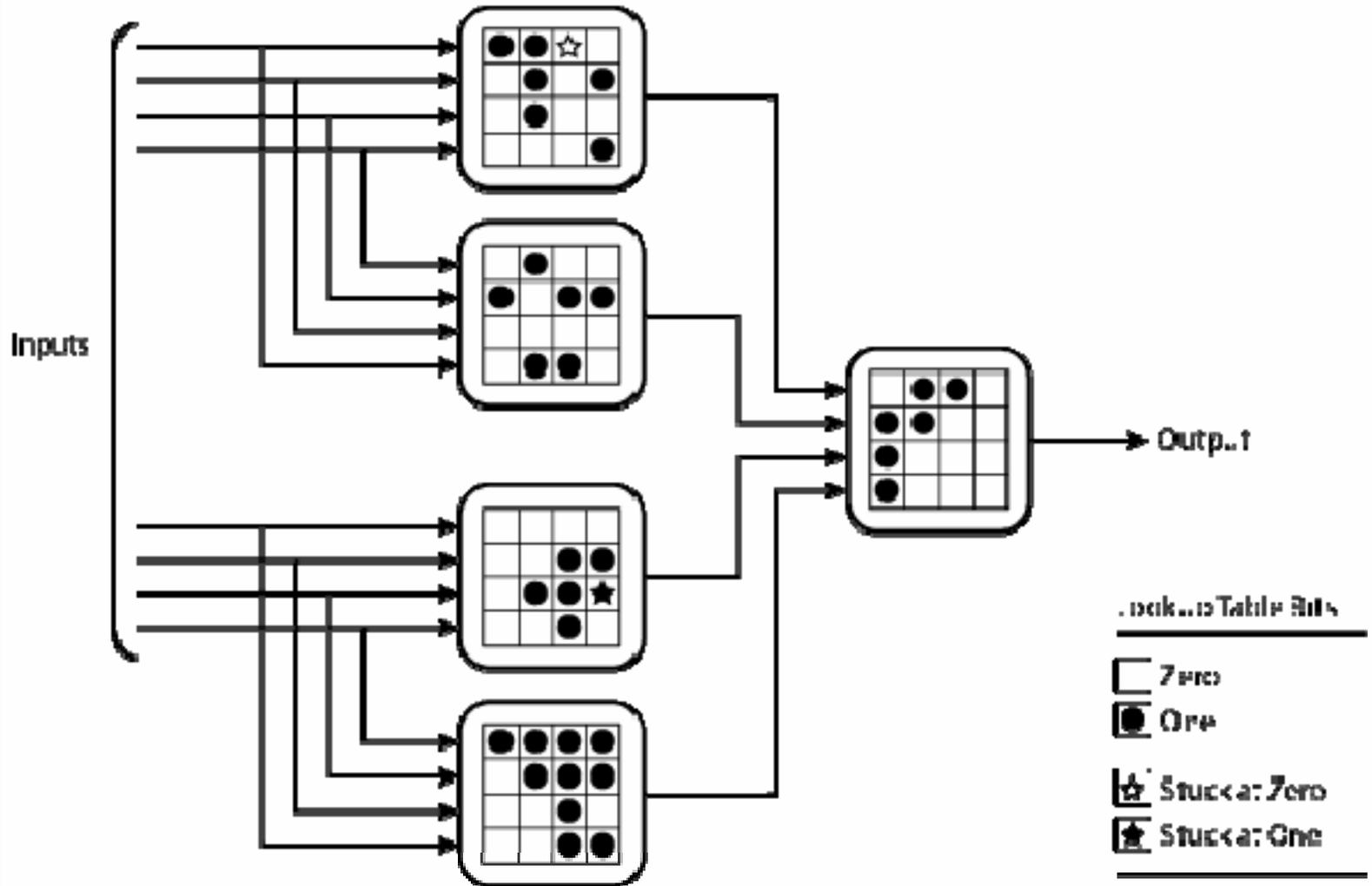
Zero budget simulation

Our implementation:

- Implemented in C++ under Linux
- FPGA-like circuit modeled with HarPE
- Circuit was flattened using partial evaluation
- Initially we converted to CNF then used *zChaff*
- Later we implemented a non-CNF (non-clausal) SAT solver from scratch (NNF-WALKSAT)



# Test Circuit





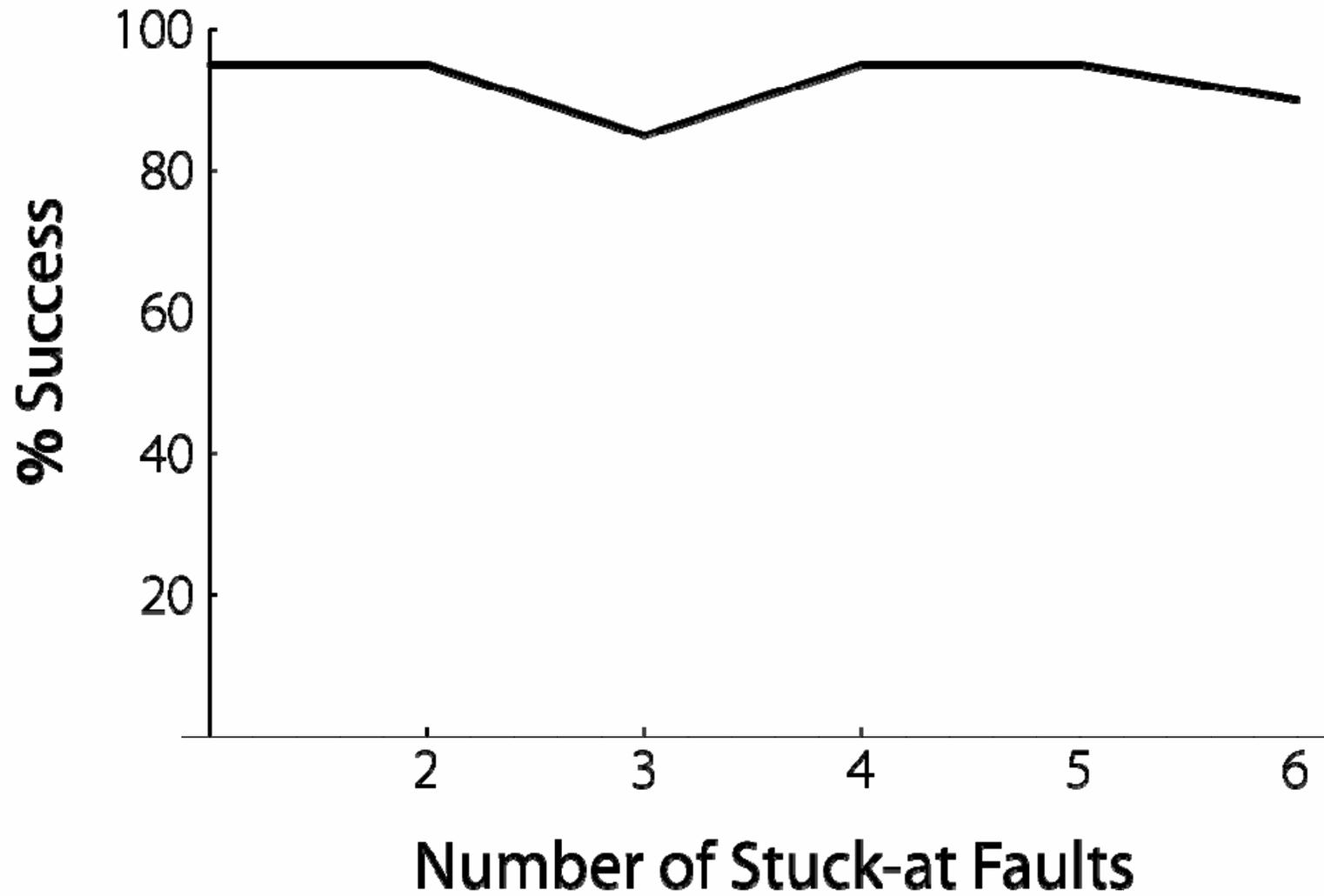
# Experimental Results

Initial results were promising:

- A surprisingly large amount of damage can be accommodated in many cases
- To be repairable, resource utilisation needs to be less than 100%
  - *(which is normally the case)*
- Time-to-repair is usually rapid (seconds) on typical CPUs

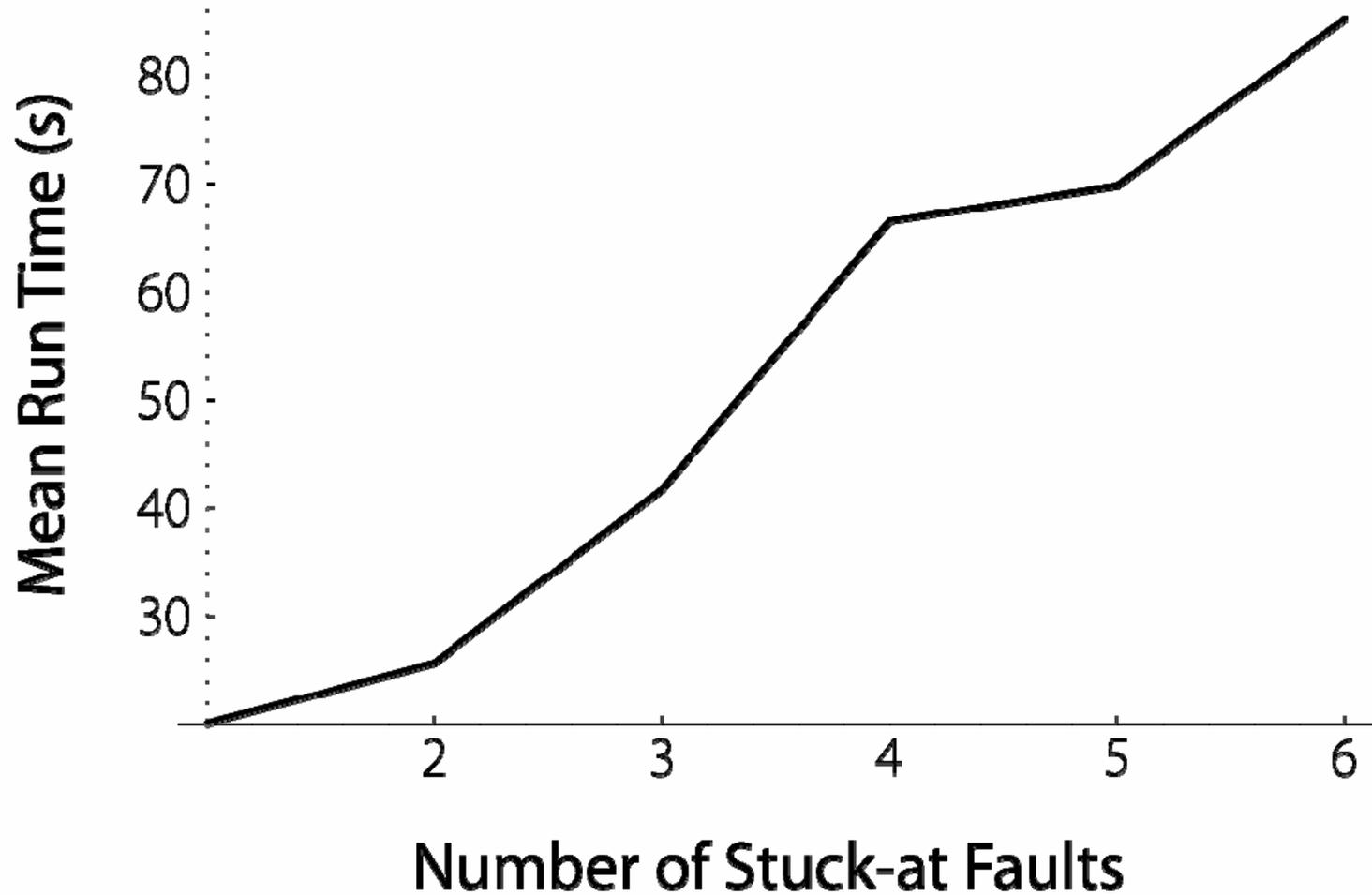


# Repair Success Rate





# Mean Repair Time





# Conclusions

The proof-of-concept exercise was successful.

As-yet untried in real FPGAs

Many open questions remain:

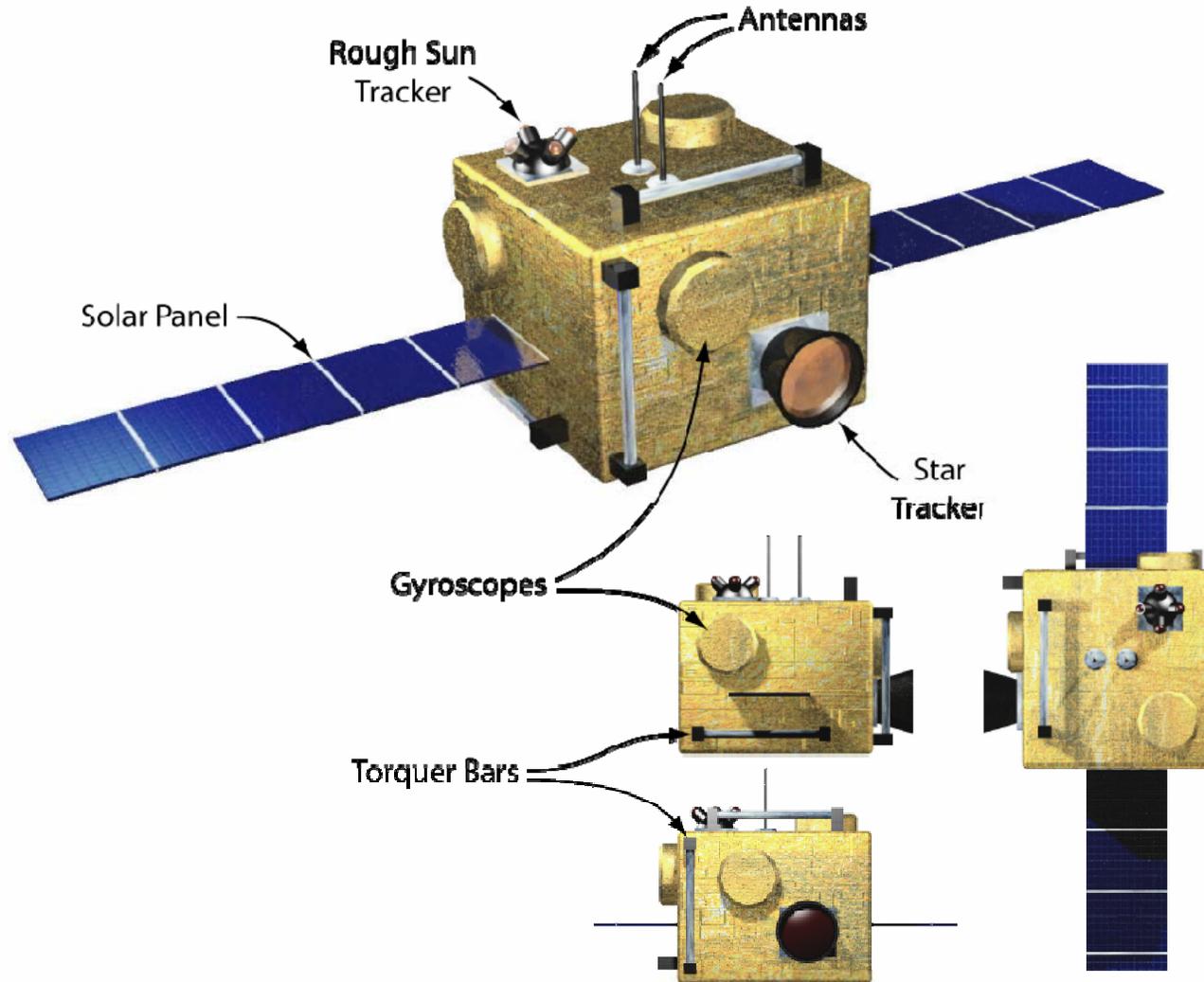
- Can COTS FPGAs be tested (e.g. via JTAG) accurately enough to enable this approach?
- Can faults be fixed within a realistic time scale given typical available CPU performance?
- Under radiation testing, what proportion of faults can be worked around, and what proportion render the FPGA too badly damaged to allow recovery?



# **Self-Healing Reconfigurable Wiring Manifolds**

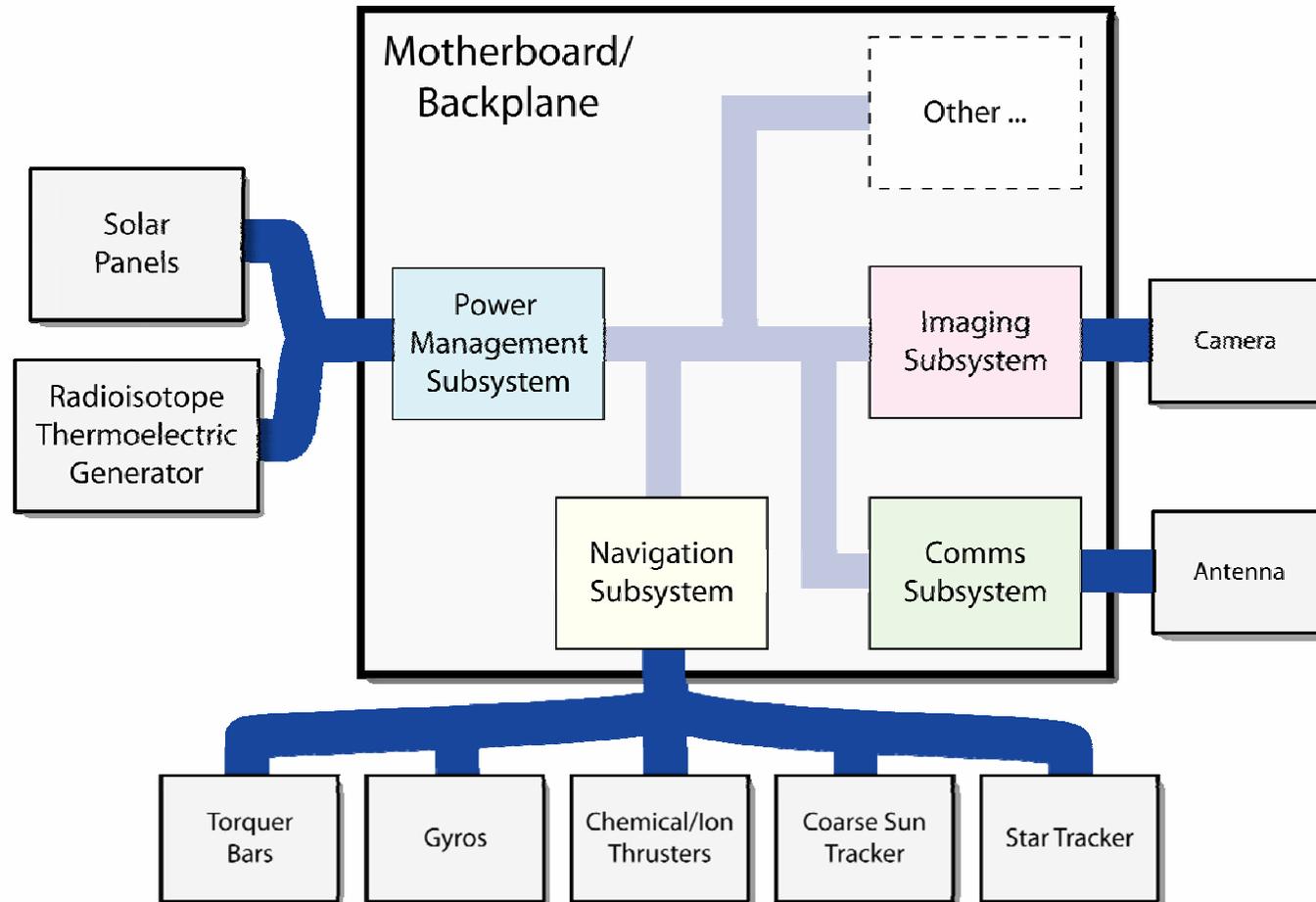


# Typical Small Satellite Configuration



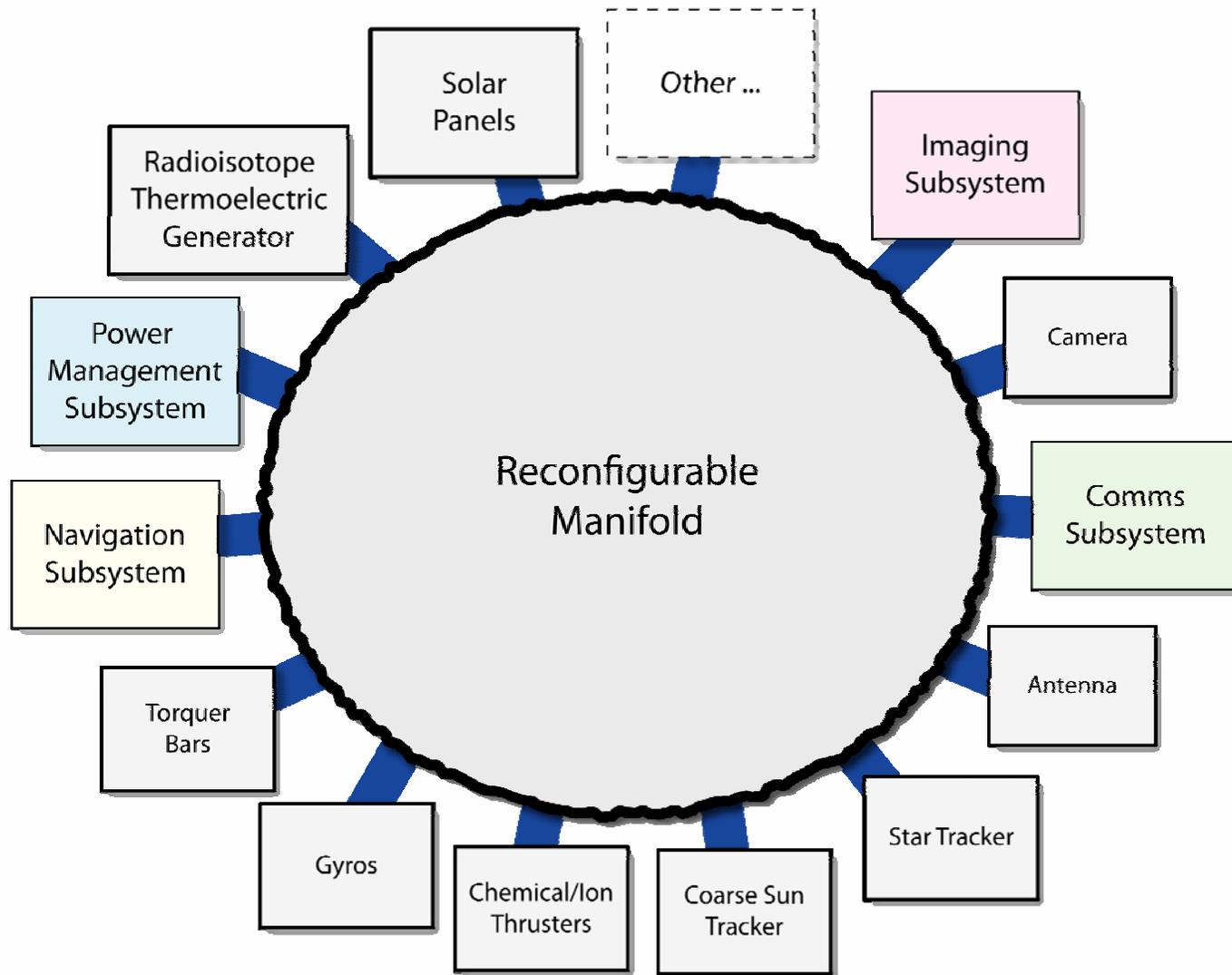


# Typical Fixed-Architecture Satellite Block Diagram





# Reconfigurable Manifold





# Just like a big FPGA?

Not quite...

- Need to support analogue, power, high current and microwave
- FPGA architectures don't scale up well for this application – satellites aren't a 'sea of gates' interspersed with wiring
- Need to be able to compute wiring plans rapidly (*NP*-complete is too slow)
- FPGA routing is not *complete*.
  
- Pre-digital era circuit-switched telephone exchanges are actually a better model.



# Signal Types

## Power

- +28V Supply Rails

## High current analogue

- Motor/solenoid drives, torquer bar drives

## Low Current, low-speed analogue

- Temperature sensors

## Low Current, High-Speed Analogue

- CCD image sensor feeds

## Low Speed Digital

- Simple sensors, limit switches

## High Speed Digital

- Networking, USB

## Low Power Microwave

- Antenna feeds

## High Power Microwave

## Optical



# Switch Technologies

FPGA

FPTA

Digital crossbar switch ASIC

Analogue crossbar switch ASIC

Digital/Analogue permutation network ASIC?

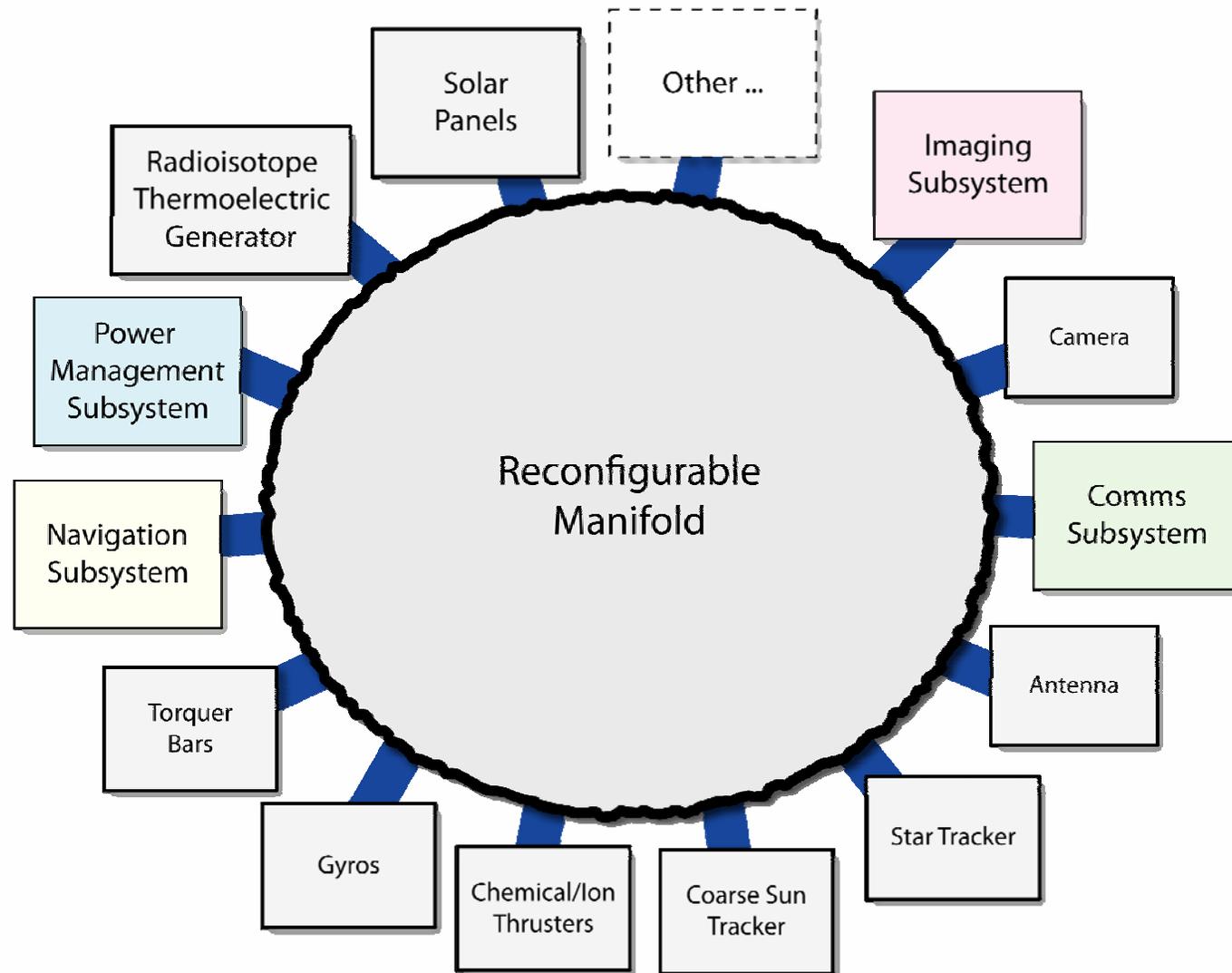
MEMS relays

Full-size relays

Discrete MOSFET/IGBT

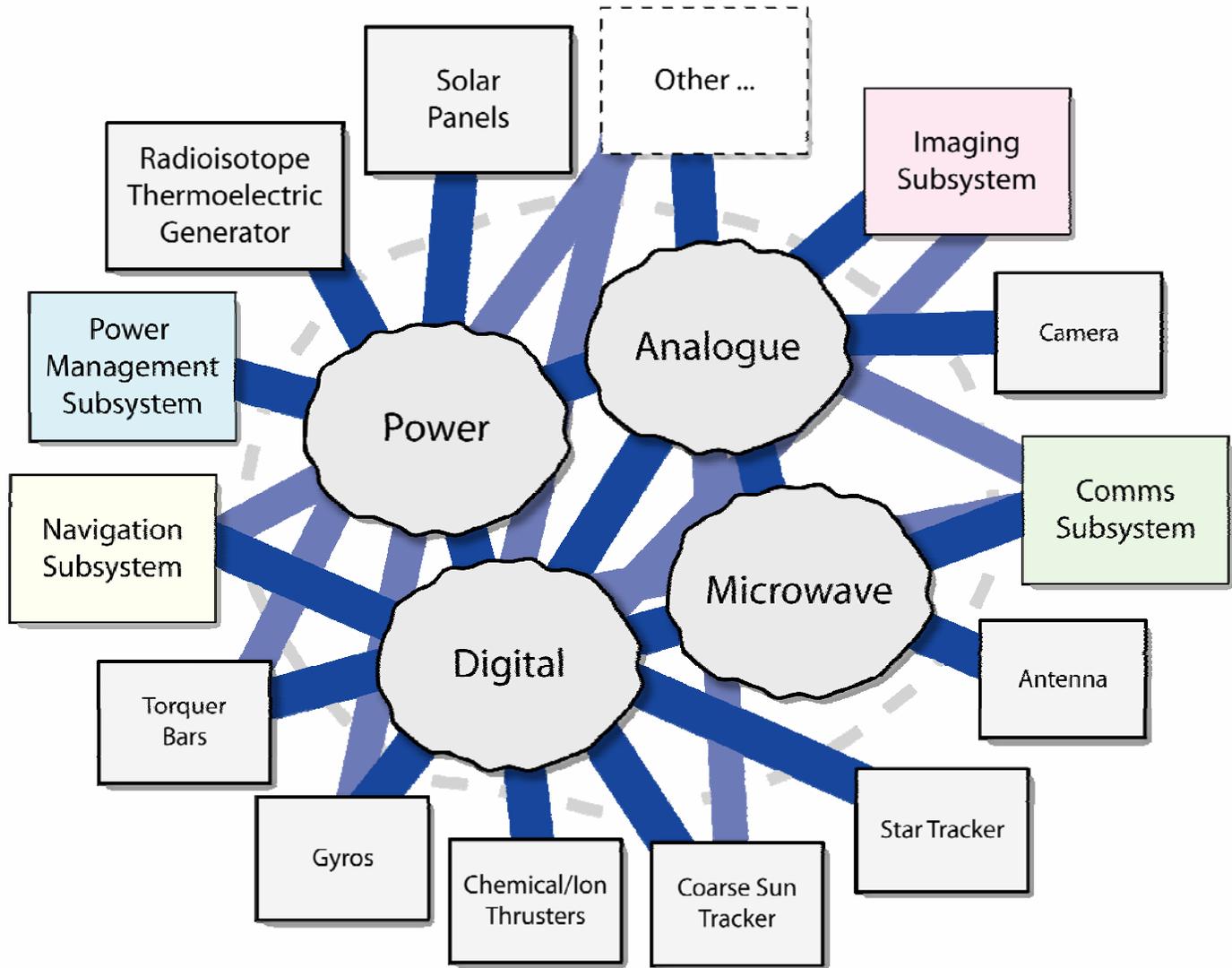


# Handling Multiple Signal Types





# Handling Multiple Signal Types





# Physical Architectures

## Single Manifold

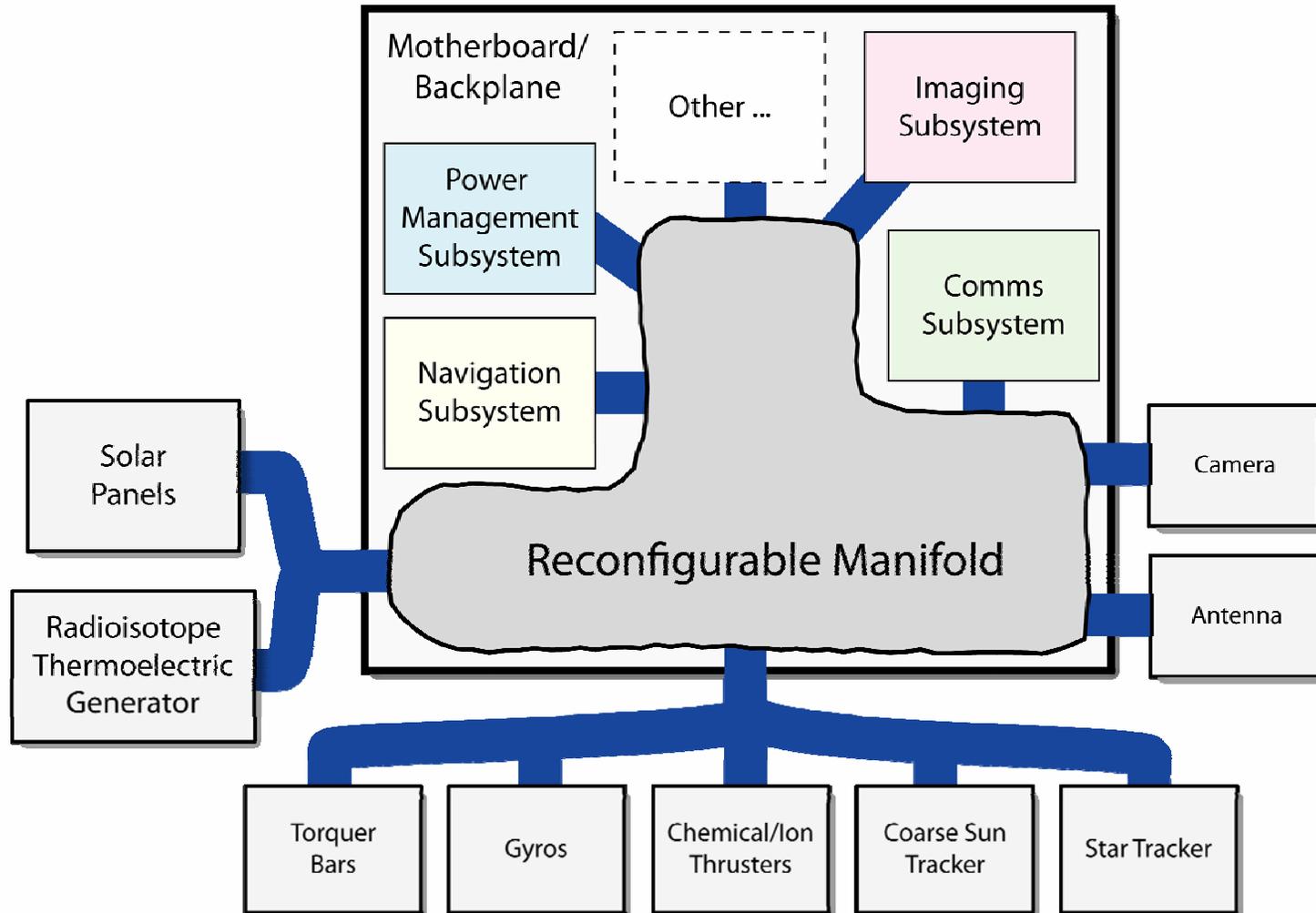
- Appropriate for small satellites

## Manifold-of-Manifolds

- Better bet for larger systems
- Advantageous for 'parts-bin' construction, because the manifold-of-manifolds scales with the number of parts added.

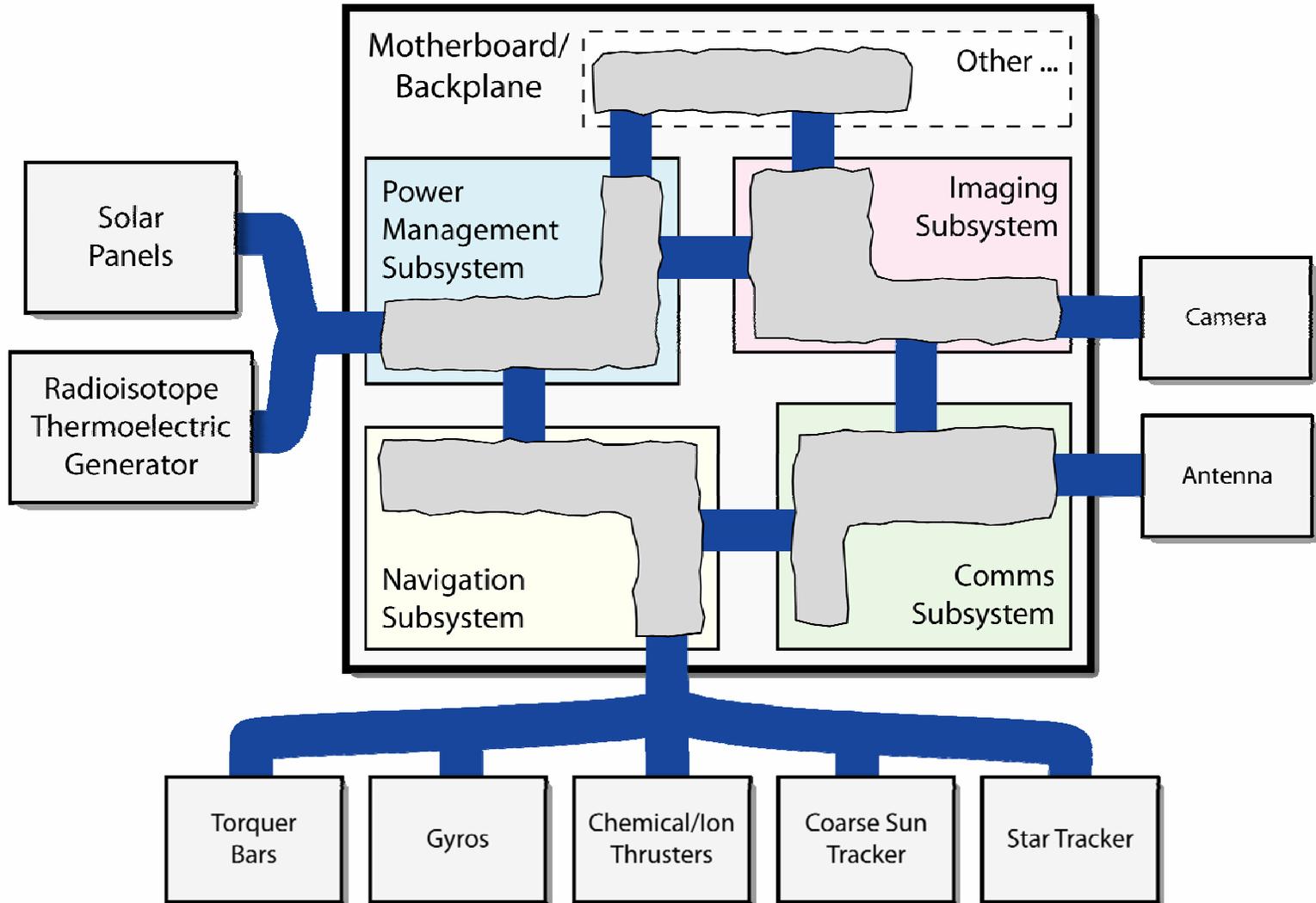


# Single Manifold





# Manifold-of-Manifolds





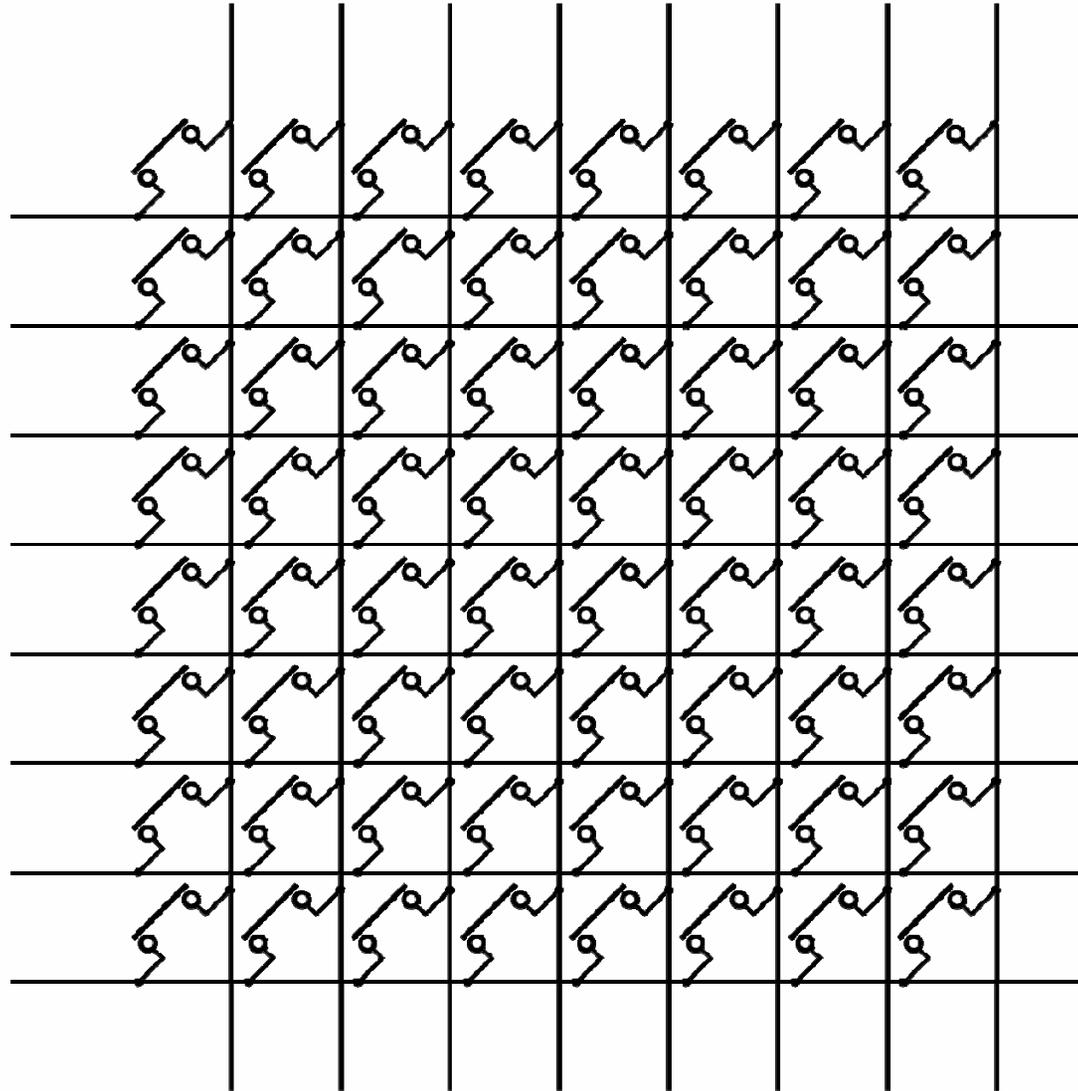
# Switching Architectures

FPGA 'sea of gates' approach doesn't scale

- Satellites simply aren't that (logical) shape
- Computing configuration bit streams is too difficult (NP-complete)
- They can not support all possible permutations of inputs and outputs (*incompleteness*)
- Self-healing is possible, but computationally difficult (NP-complete, though feasible with a good SAT solver)



# Crossbar Switch





# Switching Architectures

## Crossbar Switch

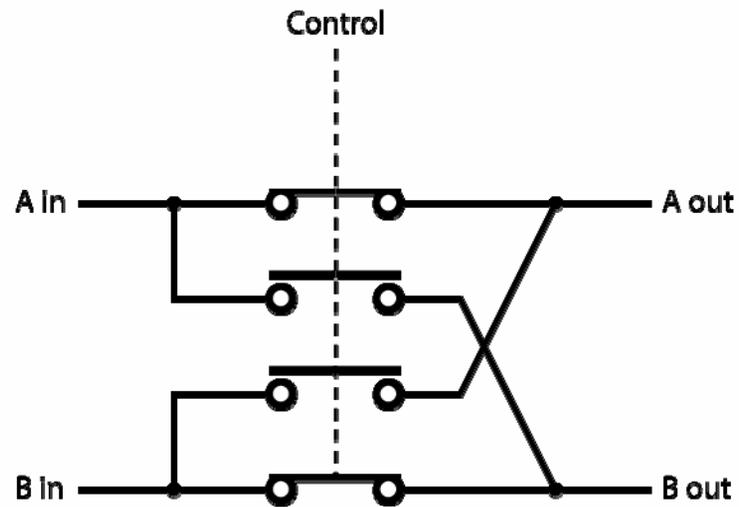
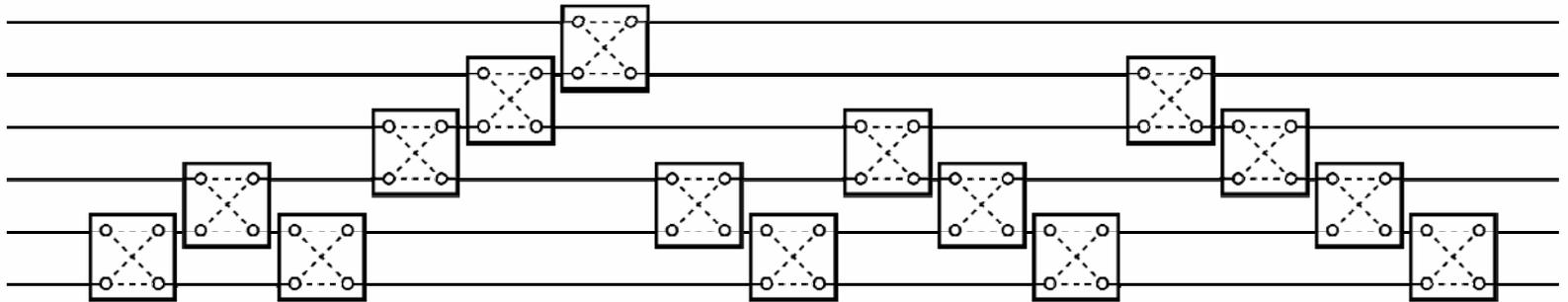
- $O(N^2)$  complexity is less than ideal, but OK for small switches
- Computing switch plan is relatively trivial
- Supports Make-Before-Break and many-to-many connections
- Complete

## Permutation Network

- $O(N \log N)$  complexity, better for larger switches
- Computing switch plan is also  $O(N \log N)$
- Doesn't support Make-Before-Break directly
- Doesn't support many-to-many connections directly
- Complete



# Permutation Network





# Other Possibilities

## Clos Networks

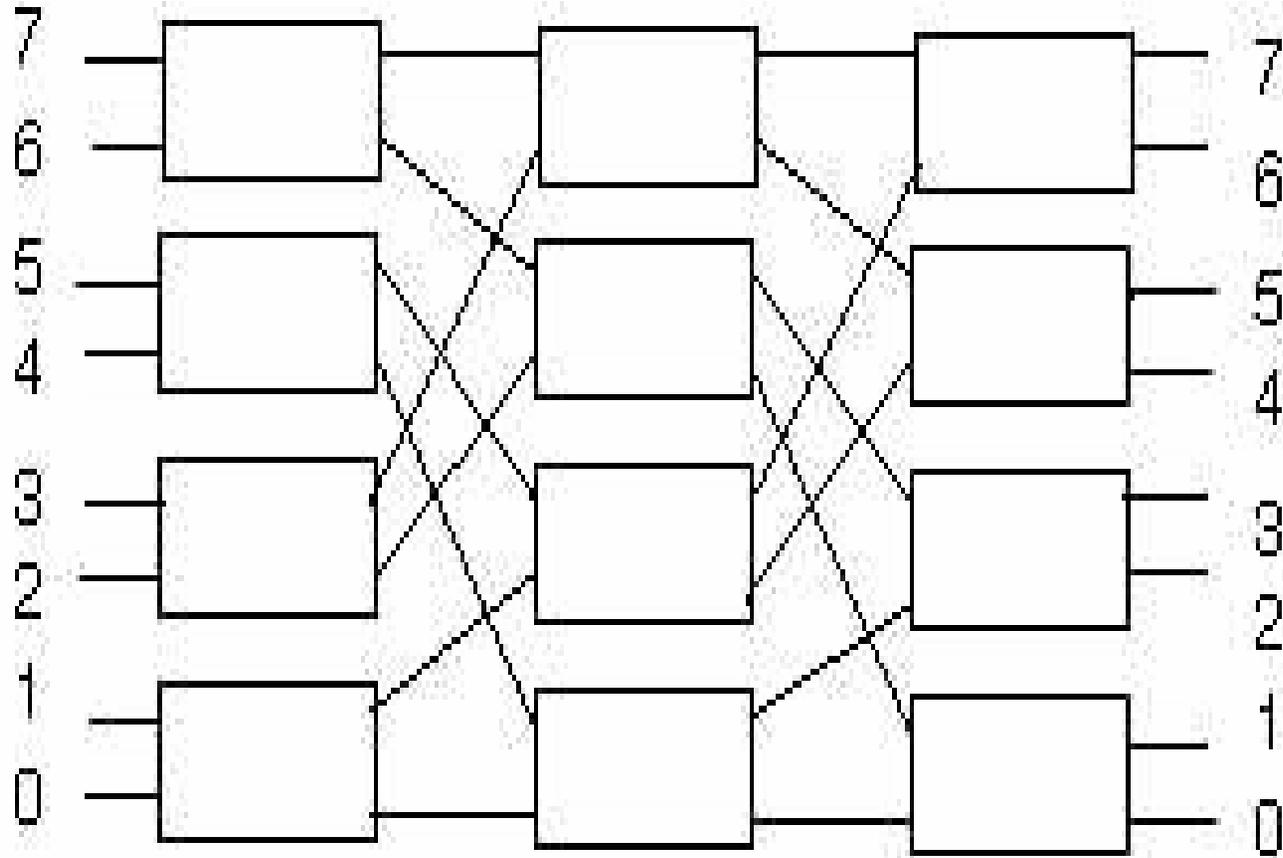
- Multiple layers of interconnected, small crossbar switches
- Well known in telecommunications
- Switch plan can be computed efficiently

## Shuffle Networks

- E.g. Omega Network
- Essentially an incomplete permutation network
- *Recent result proved that composing two Omega networks in sequence gives a true (complete) permutation network*



# Omega Network





# Other Possibilities

## Embedding into Arbitrary Graph

- Most general solution – all other architectures are special cases
- Can be complete or incomplete
- Generalised routing is *NP*-complete, which is *very bad* for dynamic reconfiguration and self-healing

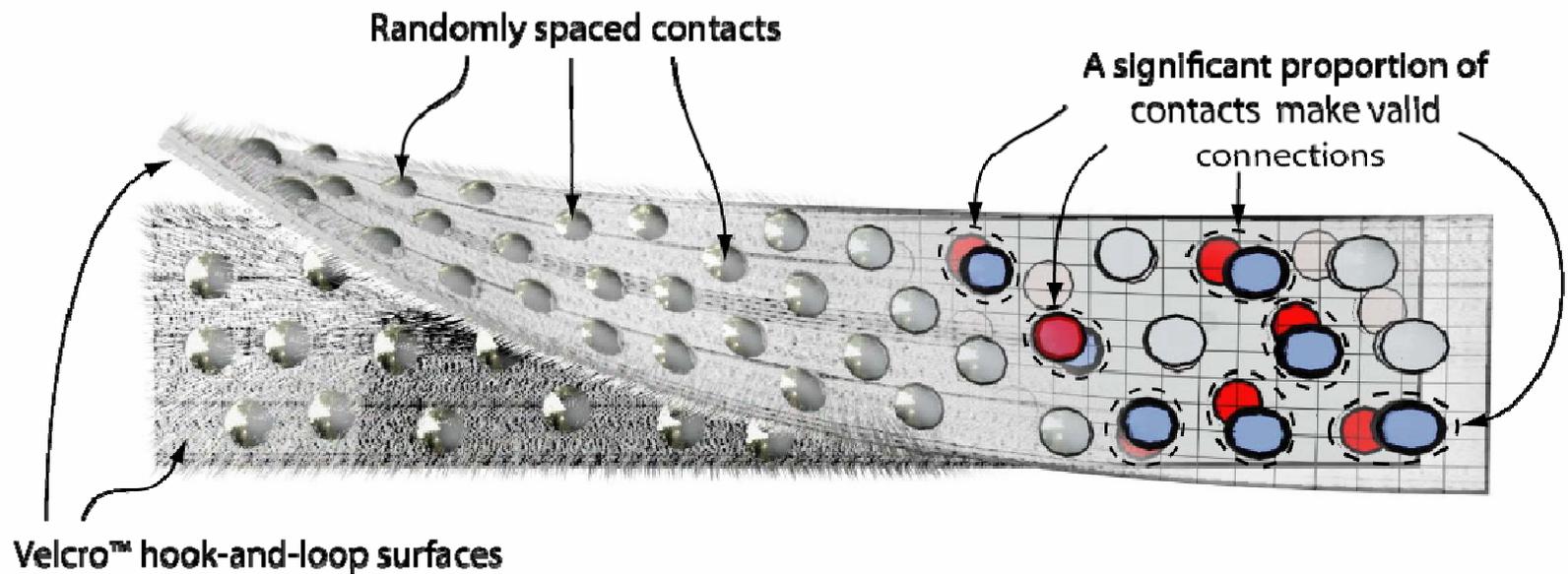


# Dynamic Discovery

To enable the **responsive space** paradigm, it is essential that a reconfigurable manifold should be able to discover its connections automatically, and configure itself dynamically without outside intervention.



# Space Velcro





# Space Velcro

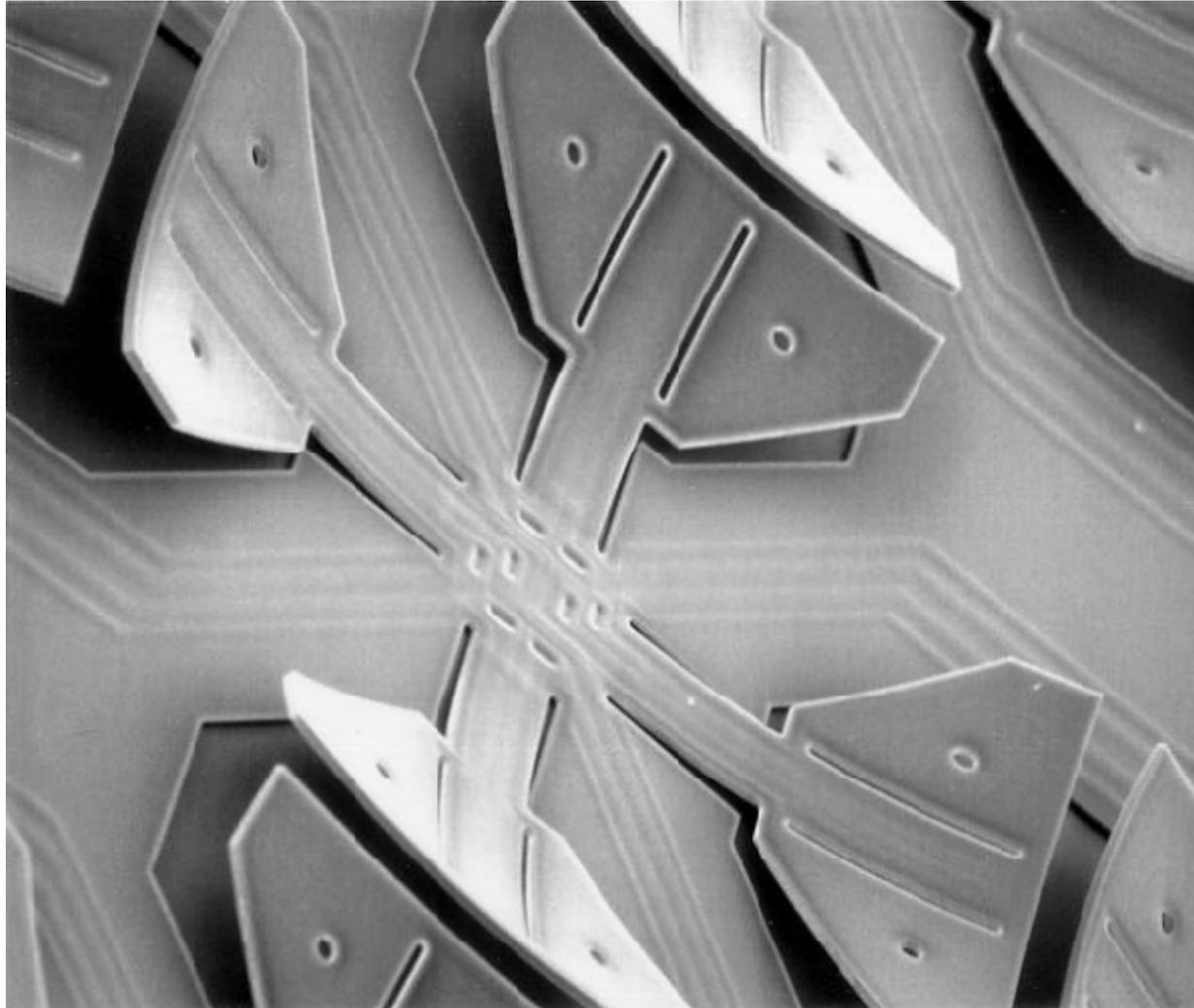
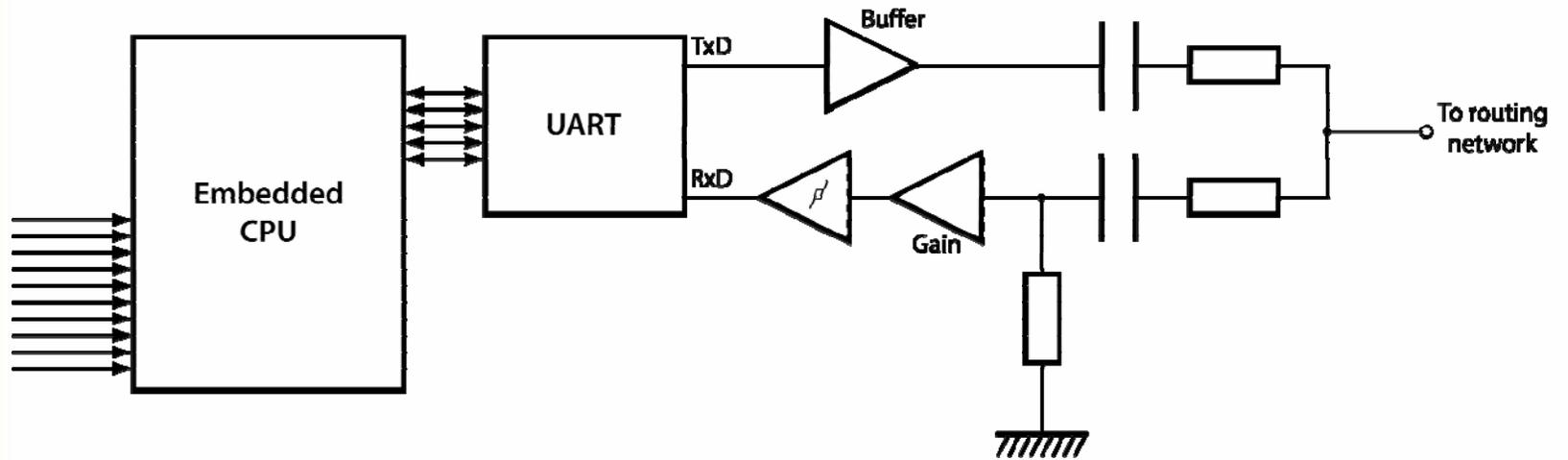


Photo: John Suh, University of Washington



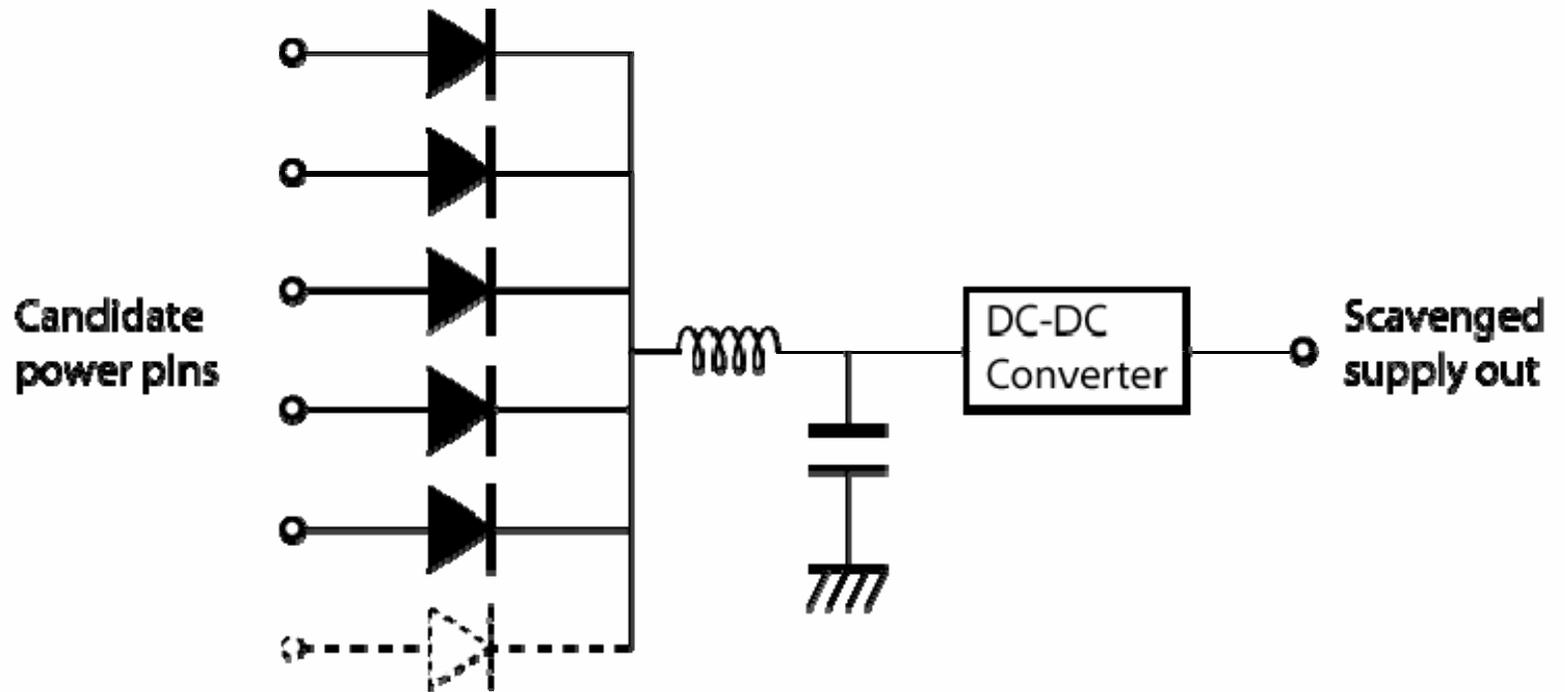
# Probe Circuit



Sync Waveform	Packet Header	Payload	Checksum
------------------	------------------	---------	----------



# Power Scavenging





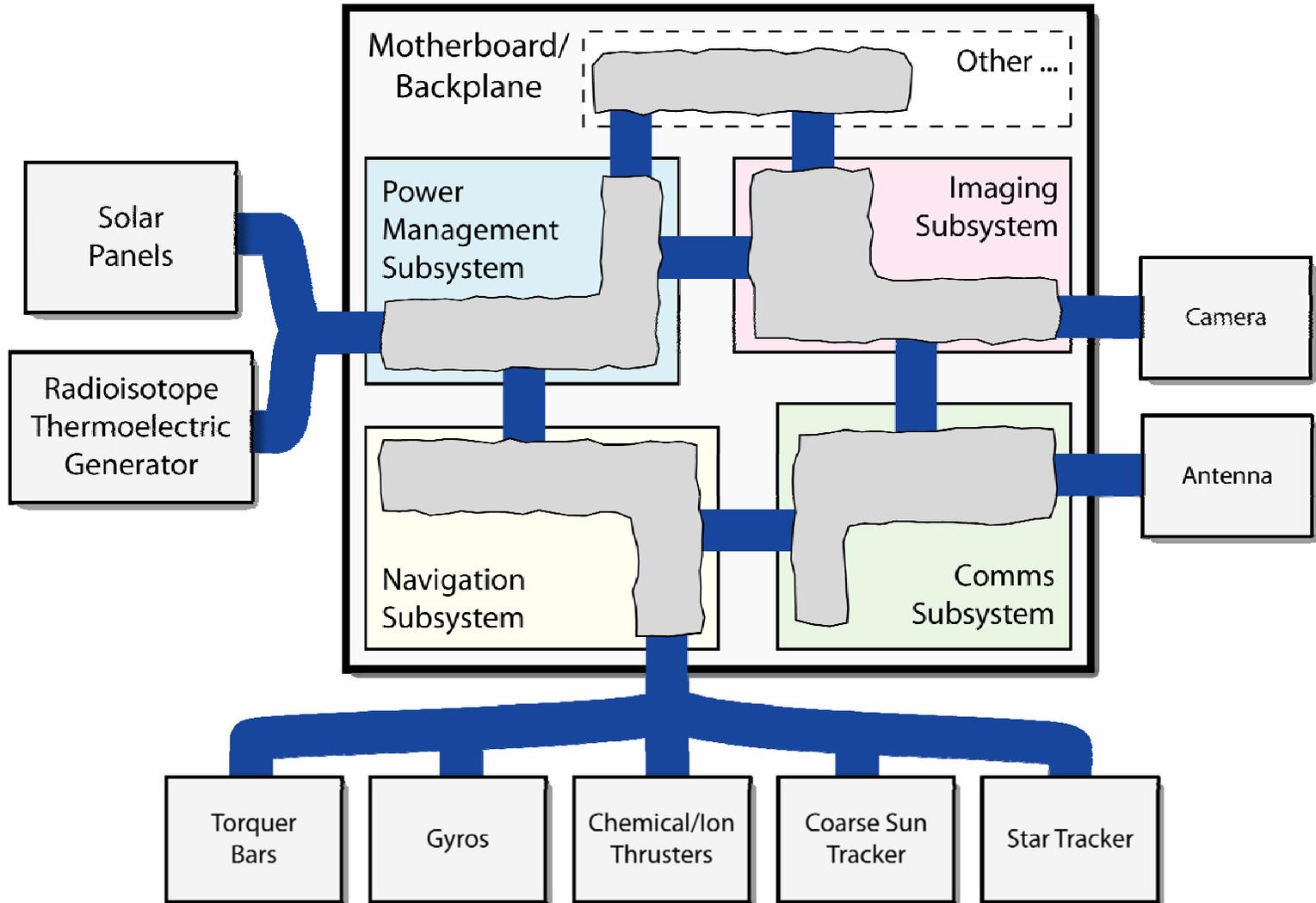
# Discovery Algorithm

1. Power scavenging provides power for manifold's CPU
2. All connections are announced/detected via protocol
3. Manifold reconfigured to route new connections
4. All connections refreshed periodically via make-before-break
5. Old connections time out and are torn down by default

... which gives self-healing for-free.

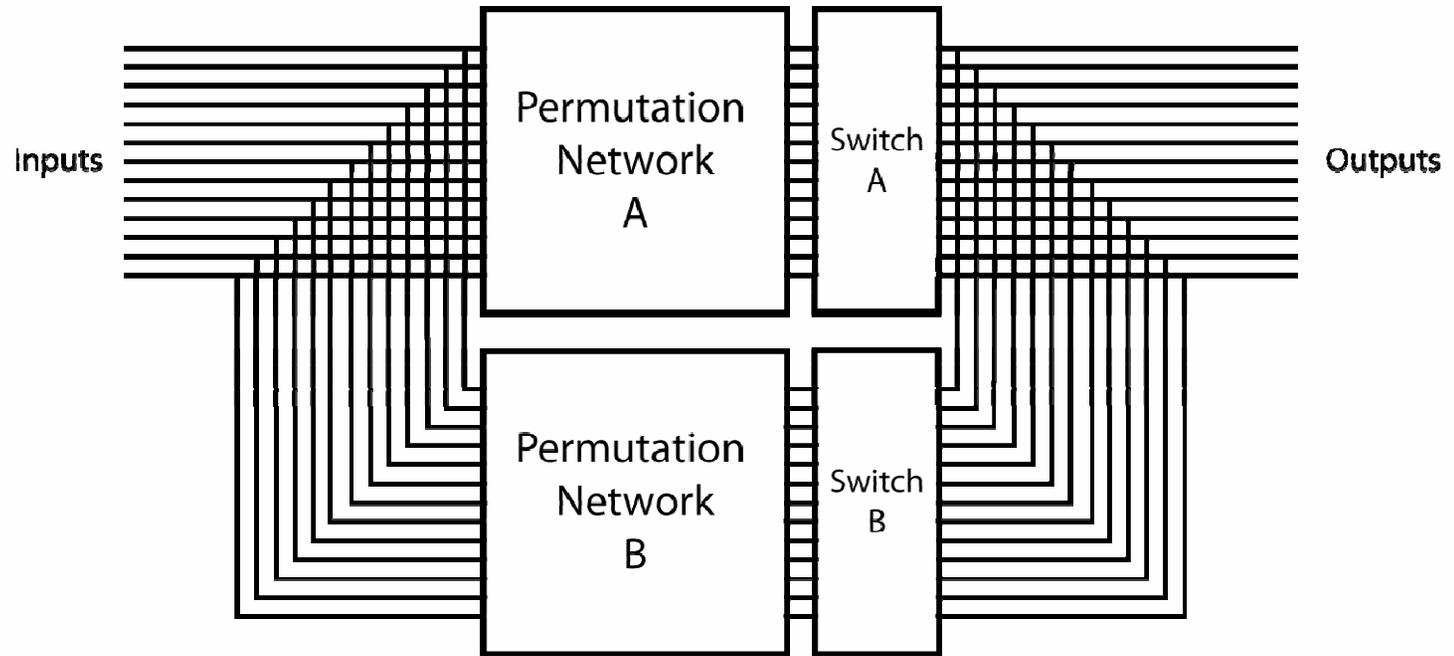


# Make-Before-Break





# Make-Before-Break Workaround



(Can add a 3<sup>rd</sup>/4<sup>th</sup> permutation network for modular redundancy)



# Conclusions

This work is still at an early stage

- Not yet flown
- Has been ground-tested in a limited sense by AFOSR (small MEMS relay based prototype)
- Plenty of spin-off applications both within and outside aerospace

Formal methods challenges

- Reliability estimation
- Completeness/correctness proofs for hardware architecture and routing algorithms
- Correctness proofs for glitch-free, make before break switching



# Acknowledgements

The author wishes to acknowledge the support and encouragement of Jim Lyke and others at AFOSR, Kirtland AFB, NM.

Much of the work presented here was carried out in collaboration with Alan Mycroft, Guillaume Brat and Arnaud Venet, for which thanks are due.

This work has been financially supported by:

- US Air Force Office of Scientific Research, Space Vehicles Directorate via the European Office of Aerospace Research & Development
- NASA Ames
- St Edmund's College, Cambridge
- Intel
- EPSRC



# Questions



# Contact Information

Sarah Thompson

- [sarah.thompson@cl.cam.ac.uk](mailto:sarah.thompson@cl.cam.ac.uk)

Alan Mycroft

- [alan.mycroft@cl.cam.ac.uk](mailto:alan.mycroft@cl.cam.ac.uk)

Guillaume Brat

- [brat@email.arc.nasa.gov](mailto:brat@email.arc.nasa.gov)

Arnaud Venet

- [arnaud@kestreltechnology.com](mailto:arnaud@kestreltechnology.com)