Poster presentation speaker notes

Introduction

- Hi my name is Matt Young, this is my thesis TaMaRa
- For safety-critical sectors, particularly spaceflight computing, FPGAs and ASICs need to be fault tolerant
- In space, specifically we need to protect against SEUs **(gesture to SEU section)** caused by ionising radiation
- This is problematic because it invalidates results which can cause loss of life in safety critical scenarios
- Can be mitigated using TMR **(gesture to description)**; replicates an element 3 times and inserts a majority voter
- Typically done manually, but this is error prone and time consuming
- Instead, I propose that using the open-source Yosys EDA synthesis tool, we can do it automatically
- This is what TaMaRa is

Literature

- The literature is generally divided into what I call design level and netlist level approaches
- Design level approaches treat the design as HDL modules and introduce TMR by replicating these modules
- Netlist level approaches treat the design as a circuit/netlist and use graph theory algorithms to insert TMR
- Netlist level algorithms generally better: although less intelligible, they can account for synthesis optimisations that would remove the TMR and work with any HDL on any FPGA or ASIC

Methodology

- TaMaRa is a netlist level algorithm
- The general process works as follows:
  - Analyse Yosys' RTL Intermediate Language netlist
  - Perform a backwards BFS to map out combinatorial logic cones
  - Lift RTLIL logic primitives to TaMaRa's own internal representation
  - Bundle this into a logic cone
  - Insert voters on a per-bit basis
  - Splice voter into the circuit, perform the necessary wiring surgery and rewiring
  - Check for successor logic cones, and repeat until we reach the edge of the circuit

Results: Circuits

- Here is a circuit before and after TMR, a 2-bit multiplexer
- You can see that afterwards, the voter has been inserted, the mux has been replicated, and the 2-bit bus has been extracted

Results: Reliability

- Here are some results from a large-scale formally verified fault injection campaign I performed
- Using 100 samples of SAT-backed equivalence checking
- TaMaRa performs well when faults are specifically excluded from being injected into the voter (blue line)
- Performs acceptably but worse when faults can be injected into the voter as well (orange line)
- All of these in turn perform significantly better than a circuit without any TMR (green line)
- All of the graphs have this short of inverse-logarithmic-ish decay vs. the number of faults

DEMONSTRATION