Harnessing an FPGA for Build-in Self-Repair in a 3D Die Stack

X. Shen, K. Nepal (University of St. Thomas), J. Dworak, T. Manikas, I. Bahar (Brown University),

Department of Computer Science and Engineering

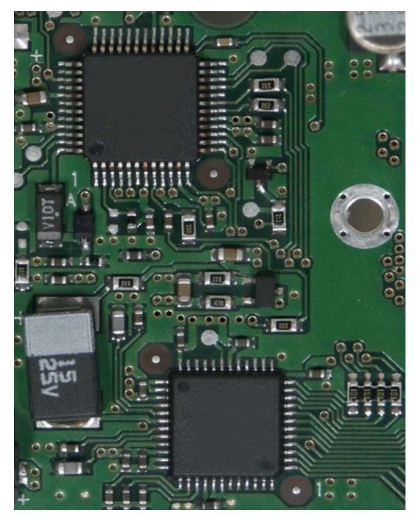
Southern Methodist University, Dallas, Texas, USA

Why might 3D chips may be in your future?

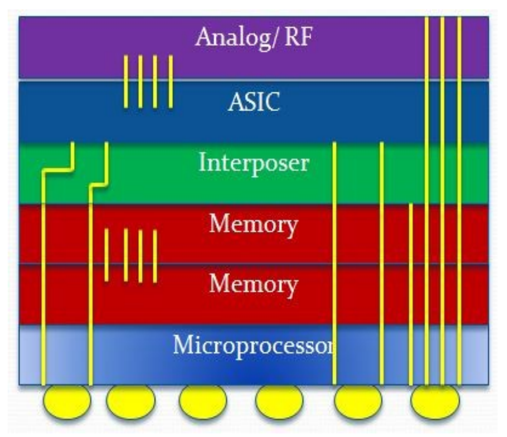
Time to walk several blocks? Probably 5 minutes

But it only takes 30 seconds to go up an elevator!

3D can make chips faster



Going from chip to another one on a board may be 2 inches away



Through-Silicon Vias (TSVs) are like elevators: distance = 30 µm/level

What is the problem?

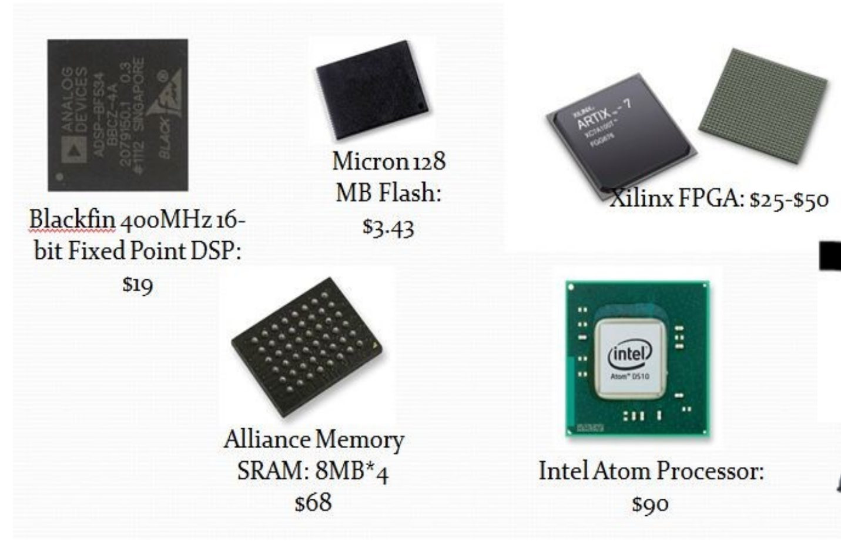
What happens if one of the chips in the stack goes bad???

Throwing out the whole stack is <u>expensive!!</u>

Error Sources in 3D:

- **Manufacturing Defects**
- **Assembly Problems**
- **Warping and Thermal Issues**
- **Wearout**

Once I put a chip in the stack, I can't take it out!


Blackfin 400MHz 16-bit Fixed Point DSP: $19
Micron 128 MB Flash: $3-43
Xilinx FPGA: $25-$50
Alliance Memory SRAM: 8MB*4 $68
Intel Atom Processor: $90

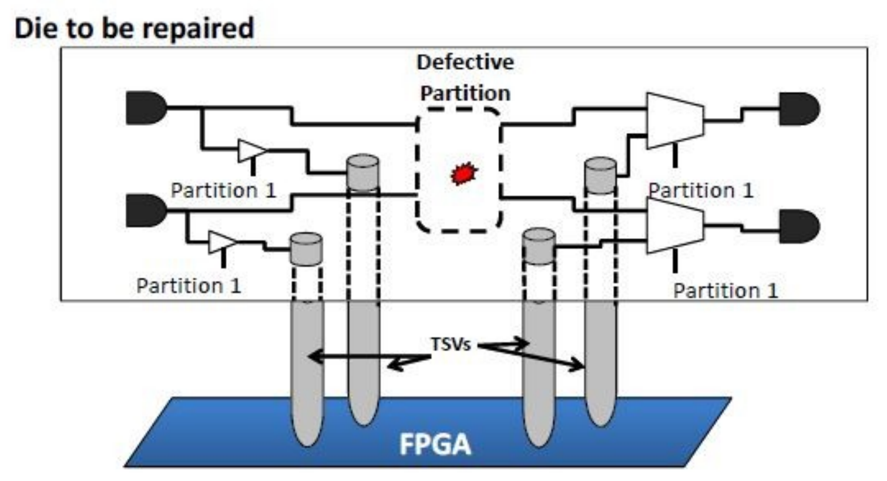
Total amount wasted: $270

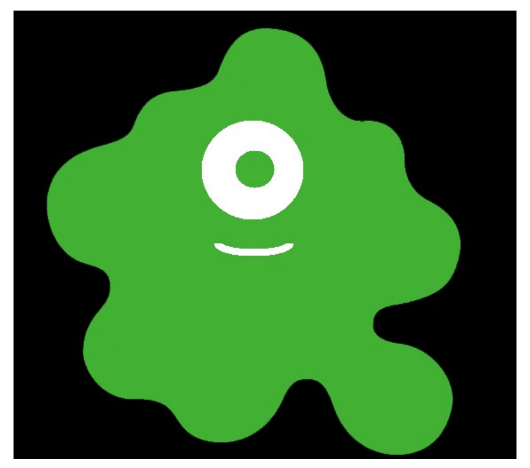
FPGA's to the Rescue:

Use an FPGA to *repair* a 3D stackl

What is FPGA? Why we can use it to repair 3D stack?

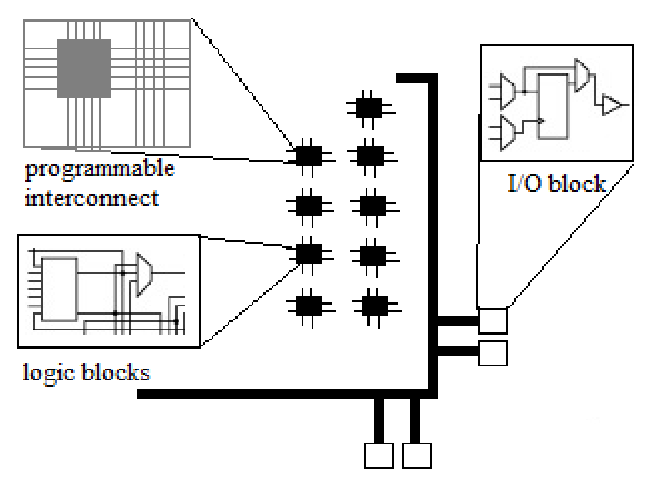
Like an amoeba that can transform into other shapes, an FPGA is a reconfigurable chip that can realize almost any type of logic function .





When there is an FPGA in a 3D stack, we can bypass the defective portion and realize the same functionality on the FPGA. Data flows to and from the FPGA through TSVs. We can then reuse this 3D stack again !

Is it a perfect solution?



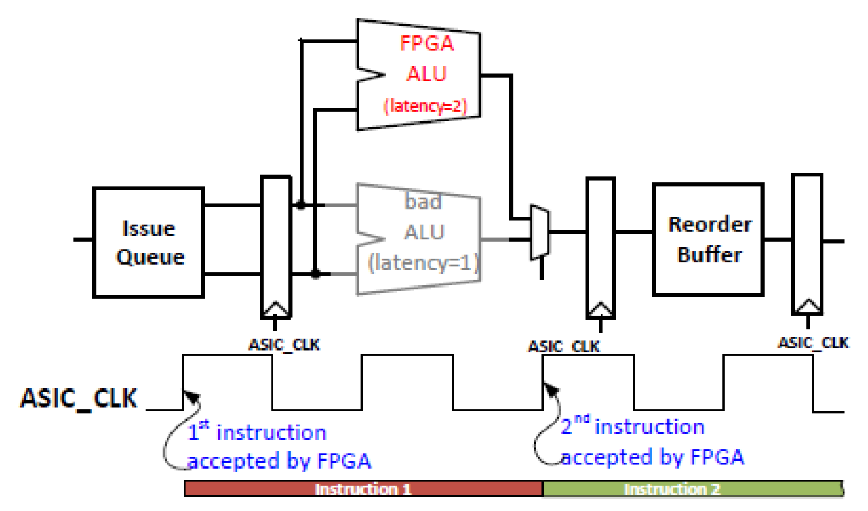
Fpga structure

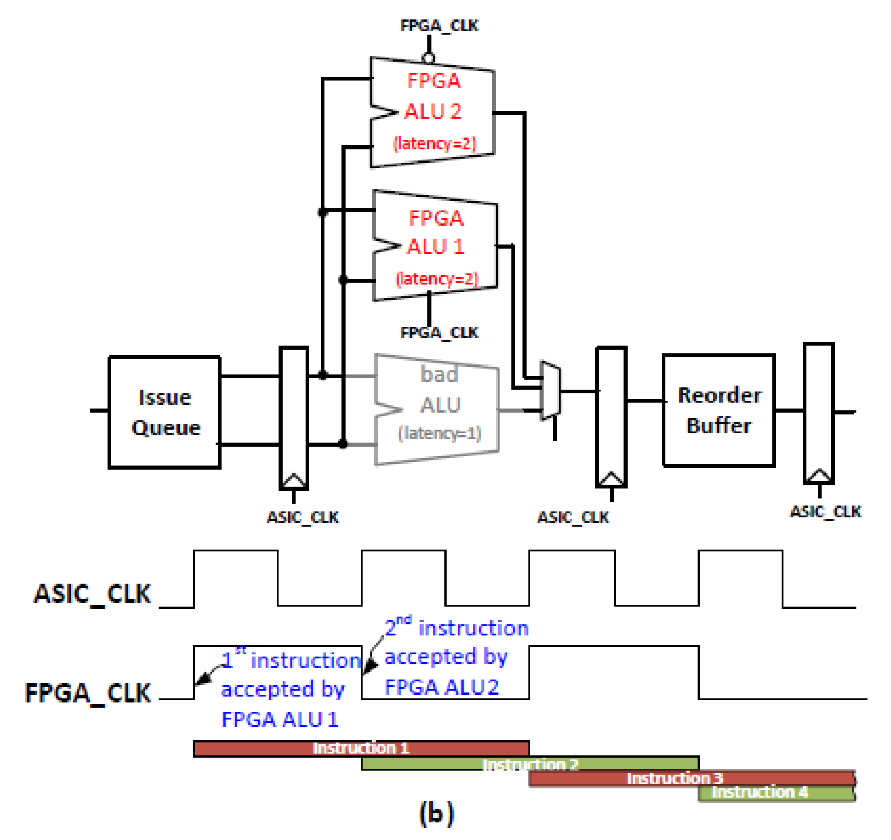
FPGA's are often considered slower than ASICs (although not always). If the FPGA is slower than the ASIC containing the defect, it can significantly reduce performance. We need to hide the difference in speed.

Does that mean it is impossible? NO!

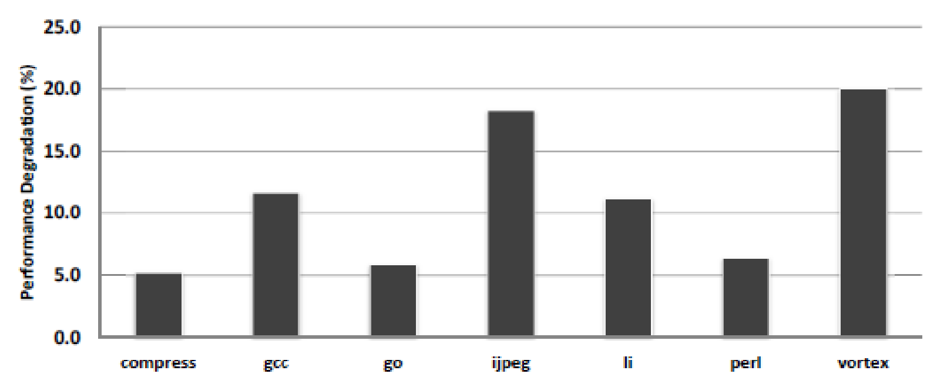
- If we have a pipelined circuit, we can put multiple copies of the defective logic in the FPGAs. On each clock cycle, a different copy can start to execute .
- As long as the instructions aren't interdependent, we can hide the increase in latency by maintaining throughput .

To evaluate the performance impact, we performed experiments on a simulator called **SimpleScalar** for several benchmark programs. Simple Scalar is an out-of-order processor model that reproduces the super-scalar processor's internal operations and provides a detailed micro-architectural timing model. We assumed one of three integer ALU's was defective and replaced it with up to 4 copies in the FPGA .





Replacement of bad ALU with (a) single FPGA ALU (b) two FPGA ALUs triggered by rising and falling clock edges respectively.



Performance Degradation without repair

Parameters	Baseline Configuration
Decode/Issue/Commit width	4 (inst/cycle)
Fetch Queue (IFQ) size	16
Register Update Unit (RUU) size	64
Load/Store Queue Size	16
Functional Units (Integer)	3 ALU
	1 Multiplier/Divider
Functional Units (Floating Pt)	2 ALU
	1 Multiplier/Divider

Baseline Configuration

	# of FPGA ALU Latency 2				# of FPGA ALU Latency 3				# of FPGA ALU Latency 4			
Benchmark	1	2	3	4	1	2	3	4	1	2	3	4
compress	3.3	3.3	3.3	3.3	1.8	1.7	1.7	1.7	-0.5	-1.1	-1.1	-1.1
gcc	10.9	12.3	12.3	12.3	7.6	9.3	9.3	9.3	3.3	3.7	3.7	3.7
go	5.1	5.8	5.8	5.8	3.5	3.9	3.9	3.9	1.8	1.7	1.7	1.7
ijpeg	20.9	28.7	28.7	28.7	18.8	24.3	24.3	24.3	16	19.6	19.6	19.6
li	10.4	12.7	12.7	12.7	6.8	8.6	8.6	8.6	3.1	4.2	4.2	4.2
perl	5.3	5.2	5.2	5.2	2.6	0.9	0.9	0.9	-2.9	-2.4	-2.4	-2.4
vortex	6.7	7.7	7.7	7.7	12.2	10.4	10.4	10.4	2.5	3	3	3
AVERAGE	8	10.8	10.8	10.8	7.6	8.4	8.4	8.4	3.3	4.1	4.1	4.1

Performance Improvement over the "No Repair Case"

As the latency of the FPGA ALU increases, more copies are needed to reduce the performance hit. However, with enough copies, we approach the performance of the non-defective version.

These FPGA's may be harnessed for not only repair, but also other usage like monitoring or testing in the future.