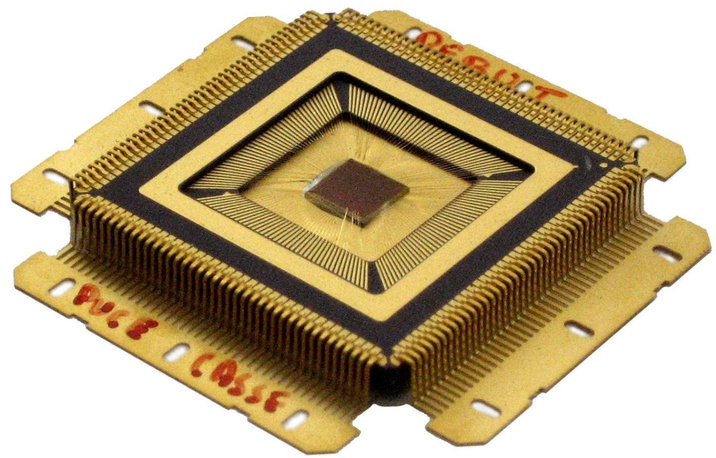
Evolving Digital Hardware
EHW Module 2008



Andy Greensted
Department of Electronics
ajg112@ohm.york.ac.uk

www.bioinspired.com/users/ajg112/teaching/evoHW.shtml

Lecture 3

CGP
(Cartesian Genetic Programming)

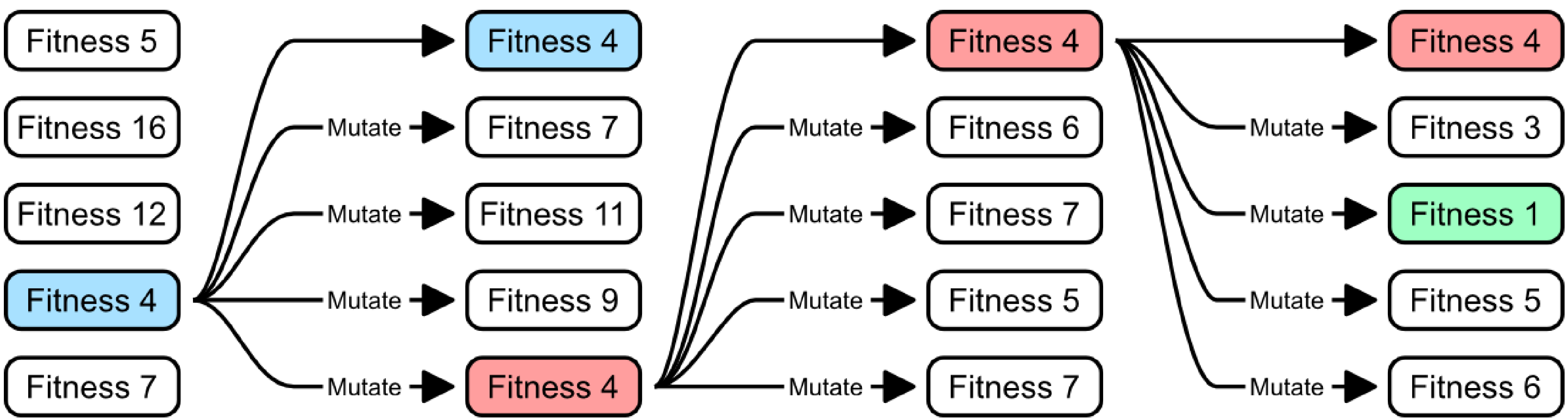
CGP

- Form of GP based on Acyclic directed graphs
  - Re-use of nodes in graph.
- Fixed genotype length
  - List of integers encoding nodes and connection in graph.
- Bounded variable length phenotype
  - Not all nodes are connected.

Cartesian Genetic Programming. J. Miller and P. Thomson,Proceedings of the 3rd European Conference on Genetic Programming, Edinburgh, (2000) 121-132.

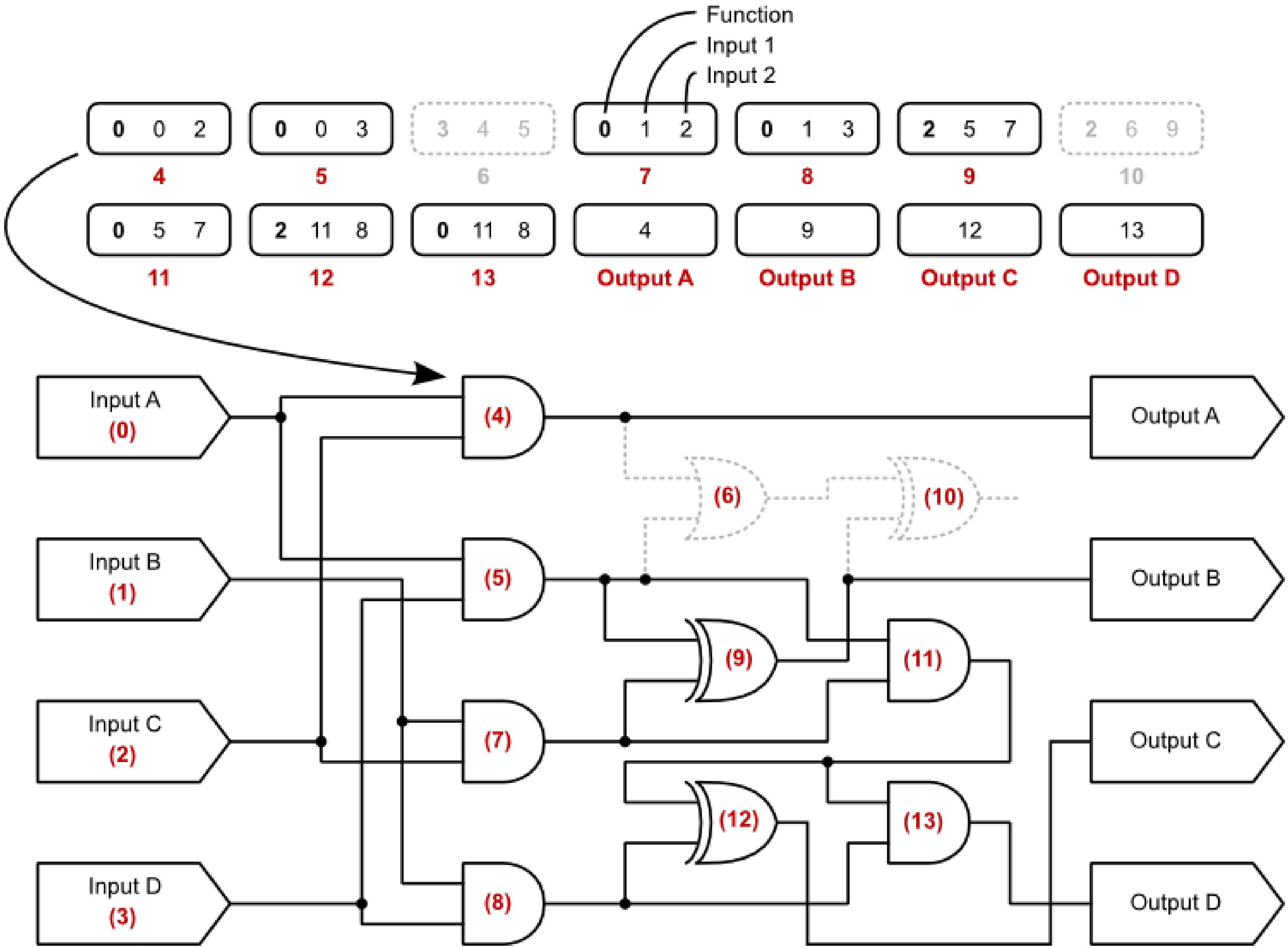
CGP Evolution Strategy

- CGP uses a (1+4) evolution strategy
- Generations are created from:
  - The current fittest, unless a new equally fit or fitter solution has been found.
  - Mutations of the fittest from the previous generation.



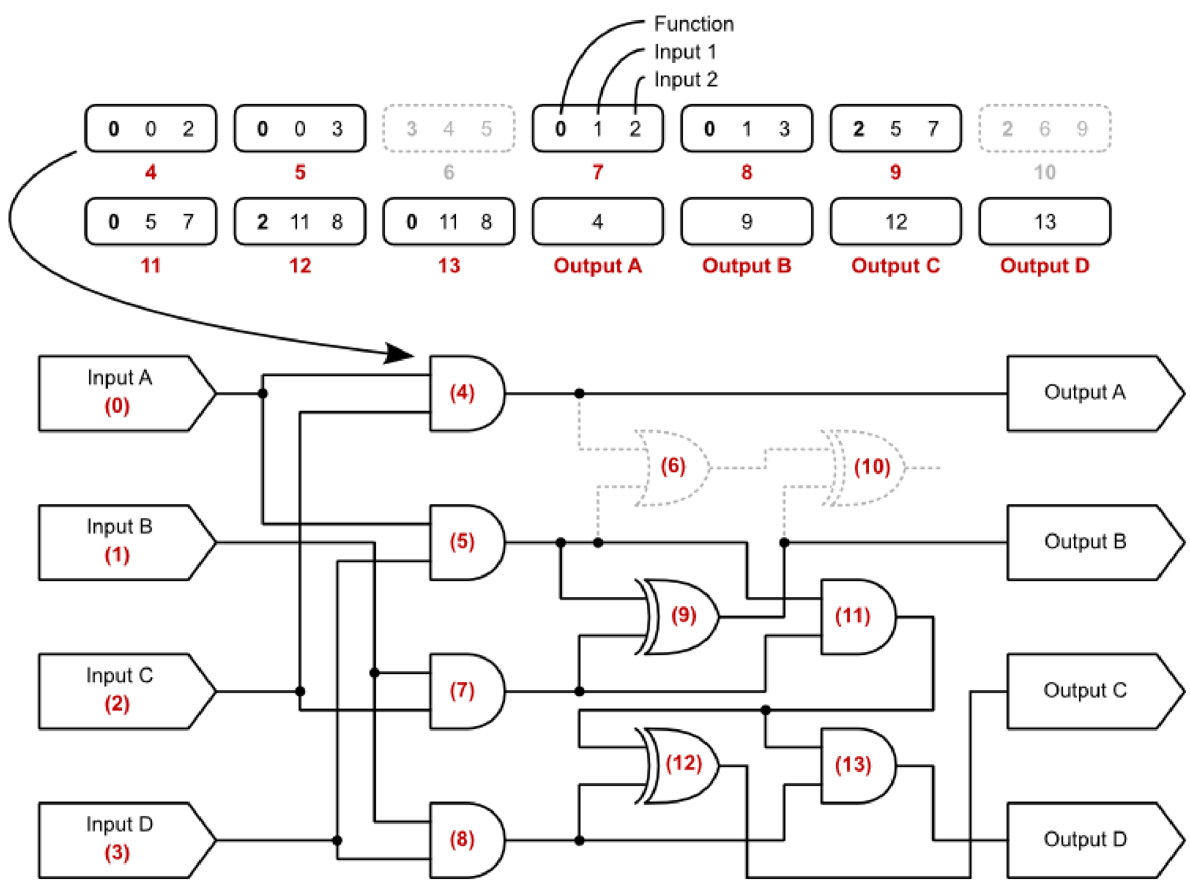
Fitter individuals have lower fitness score

Example Circuit in CGP Representation



CGP Encoding Features

- The genotype is a fixed length
- Function inputs can only connect to previous function outputs
  - Stops combinatorial loops



Case Study 1
Evolving Fault Tolerance

Evolved Fault Tolerance

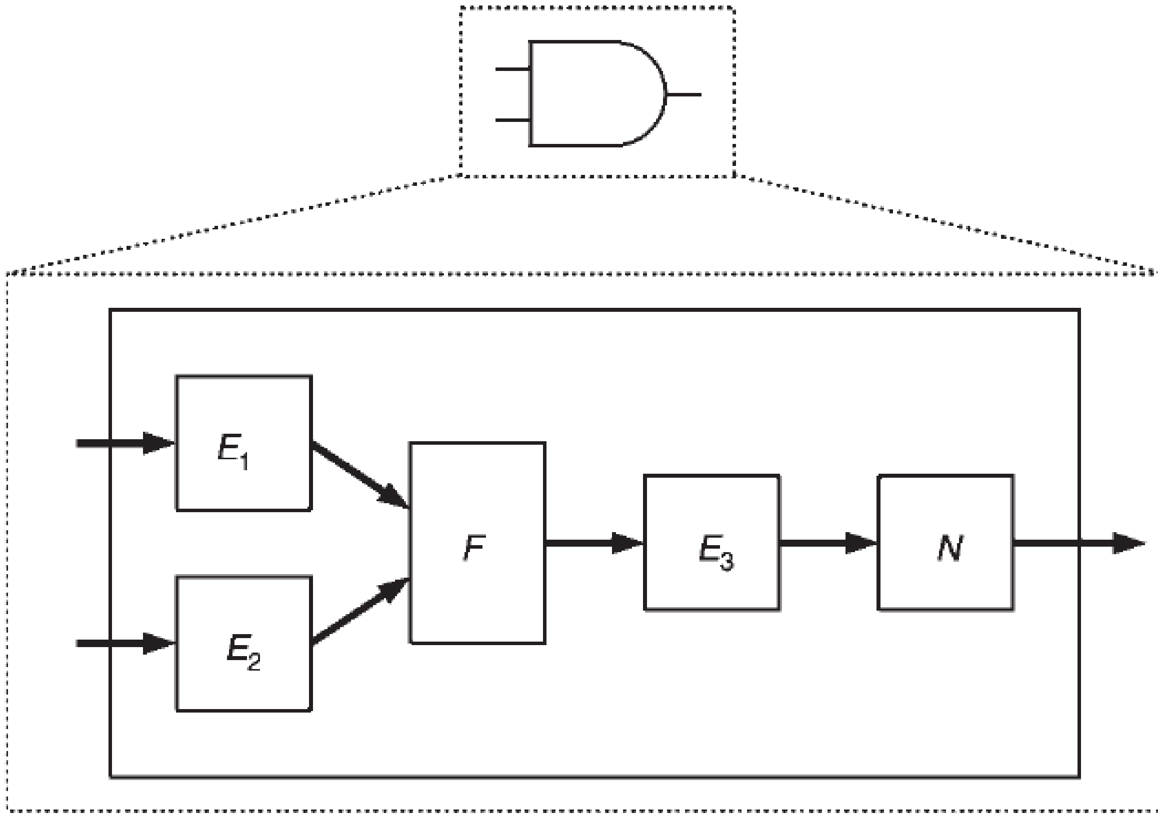
- Use evolution to create fault-tolerant circuits.
- A circuit's fitness is based upon:
  - Its ability to operate correctly under different fault conditions.

$$F = -\left(\frac{\sum_{n=1}^{TPI} diff(C_m, T)}{TPI}\right)$$

F	Average fitness for all environments
Cm	A circuit under environment *m*
T	The target circuit
diff()	The number of different outputs bits
TPI	Number of Environments

Evolution of Fault-Tolerant and Noise-Robust Digital Designs M.Hartmann, P.C. Haddow, IEE Proc.-Comput. Digit. Tech. Vol 151, No.4 , July 2004

The Gate Model



E_1: Input Error

E_2: Input Error

F: Function Generator

E_3: Output Error

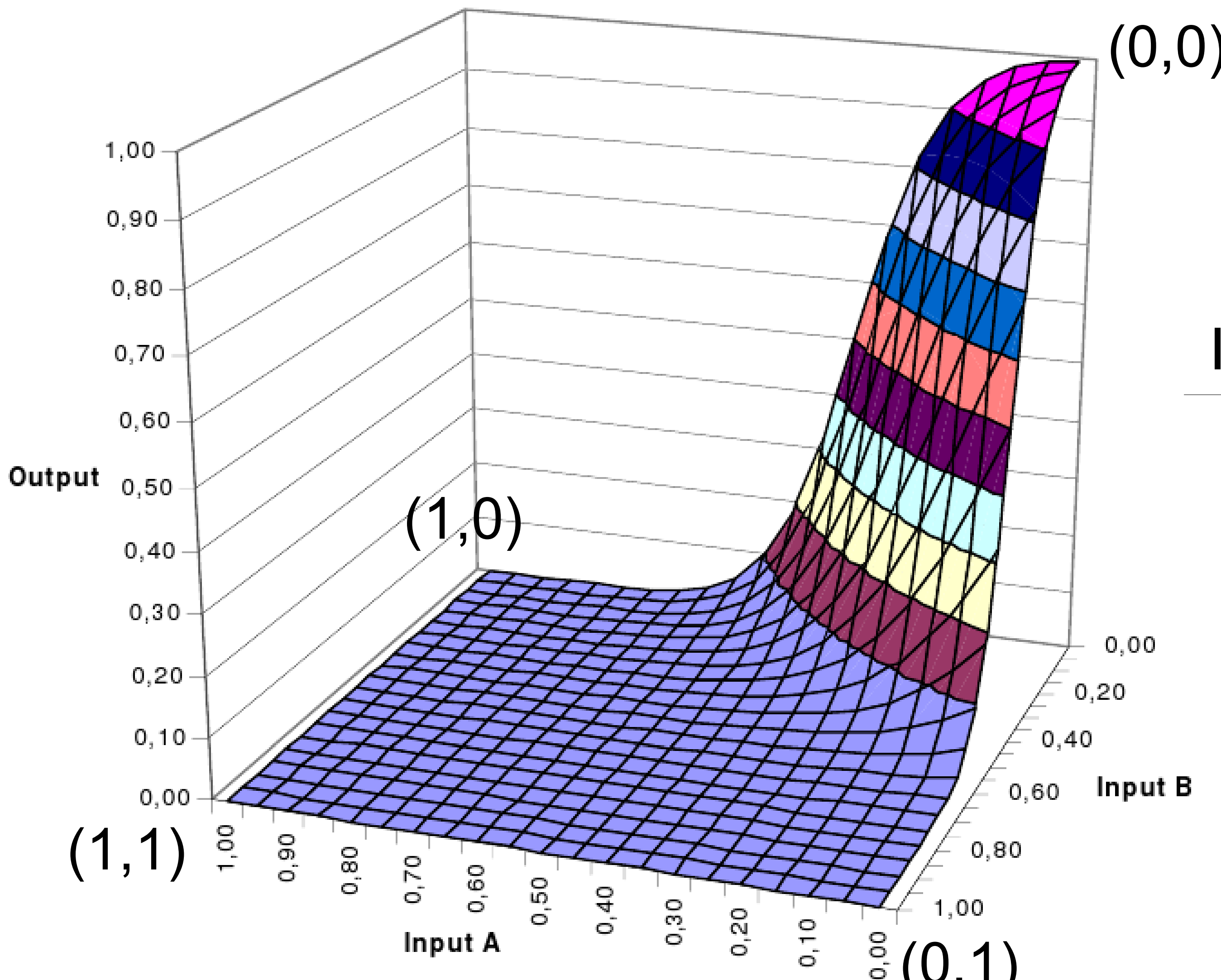
N: Noise

Errors: Stuck-at errors
Floating output
Partially random output

Functions: NOR, NAND, OR, AND, NOT, VCC & GND

The Gate Model

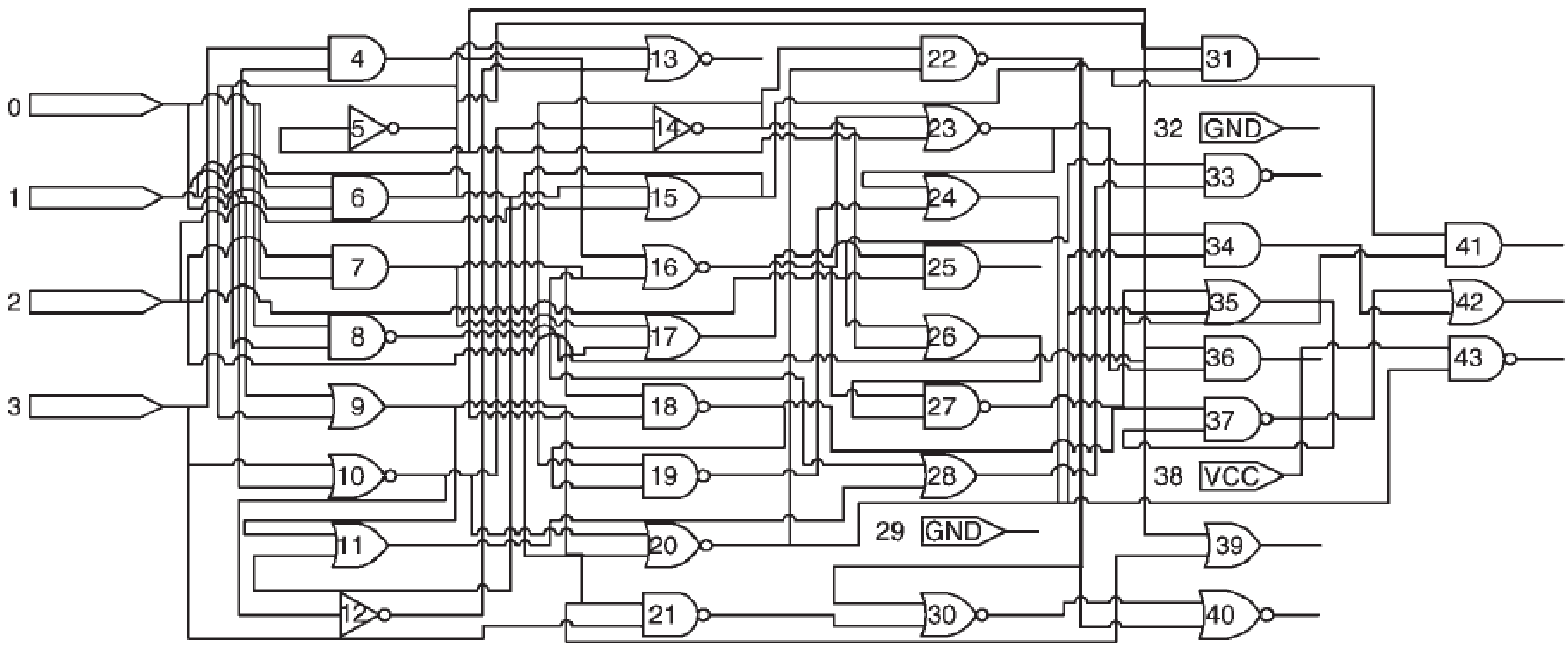
- The model for a NOR gate



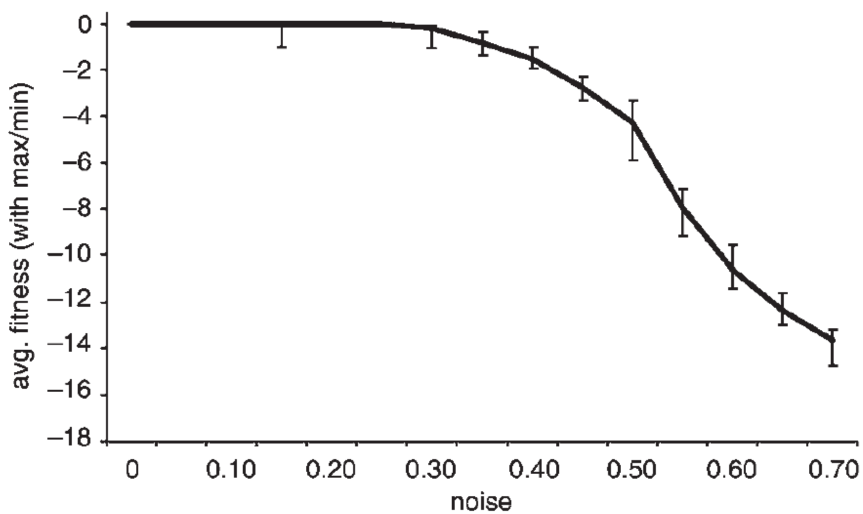
InA	InB	NOR
0	0	1
0	1	0
1	0	0
1	1	0

Target Circuits

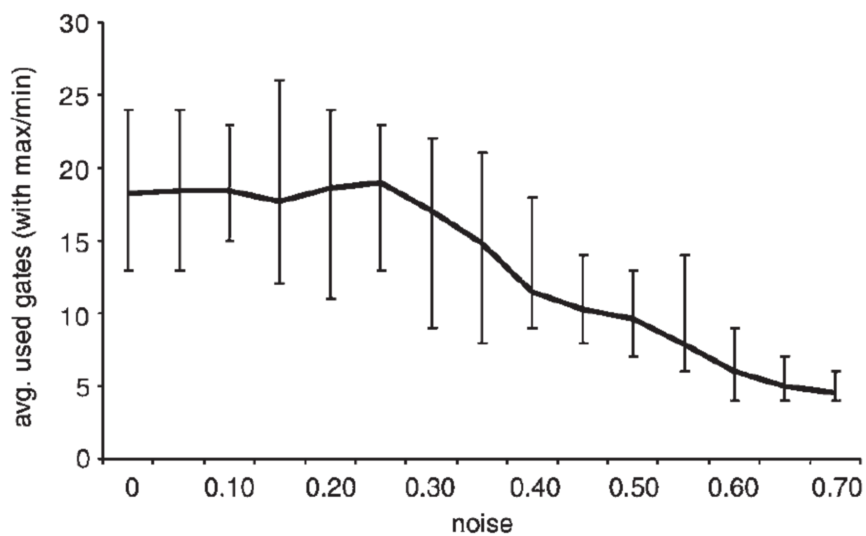
- Both 2bit Adders and Multipliers were tested



Results



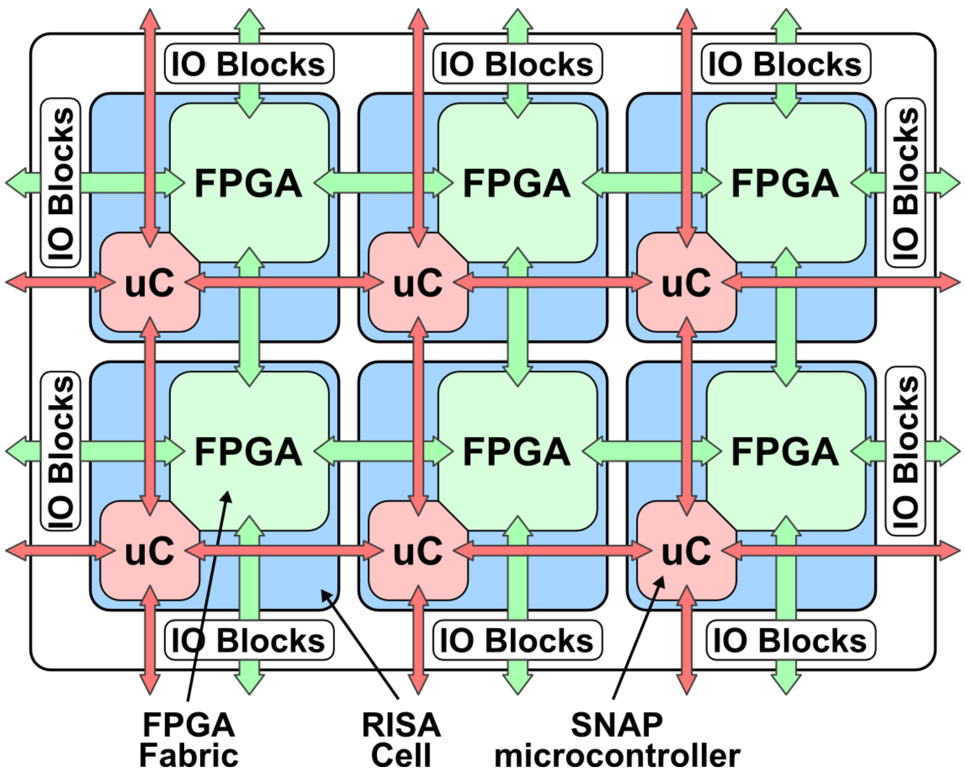
- Average fitness of circuits against applied noise level.



- Average number of used gates against noise applied during evolution.
- (Evolution prefers circuits with less gates as noise level increases, this reduces the likelihood of error)

Case Study 2
RISA

The Reconfigurable Integrated System Array (RISA) Architecture

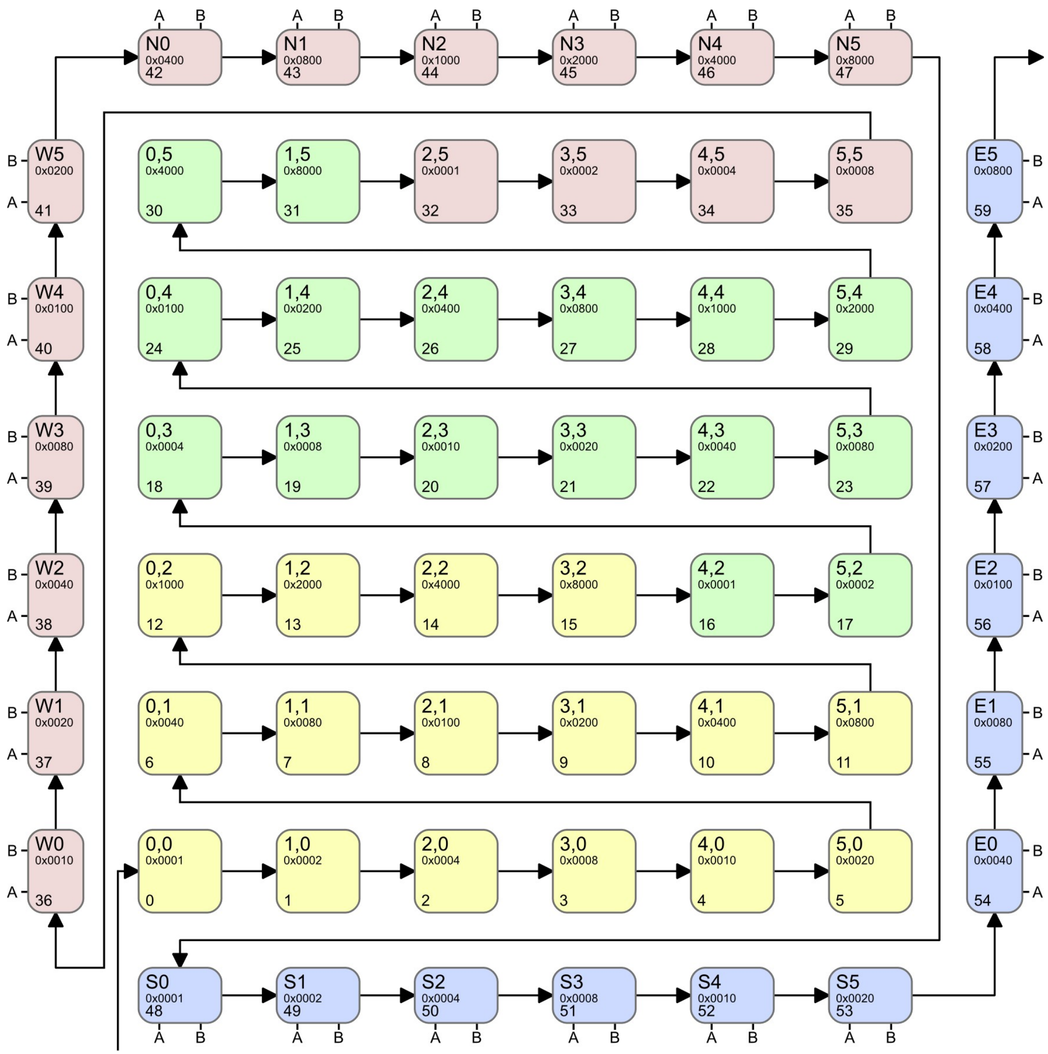


- A new embryonic tissue
  - Processor/FPGA Array

Supports Different Configurations:

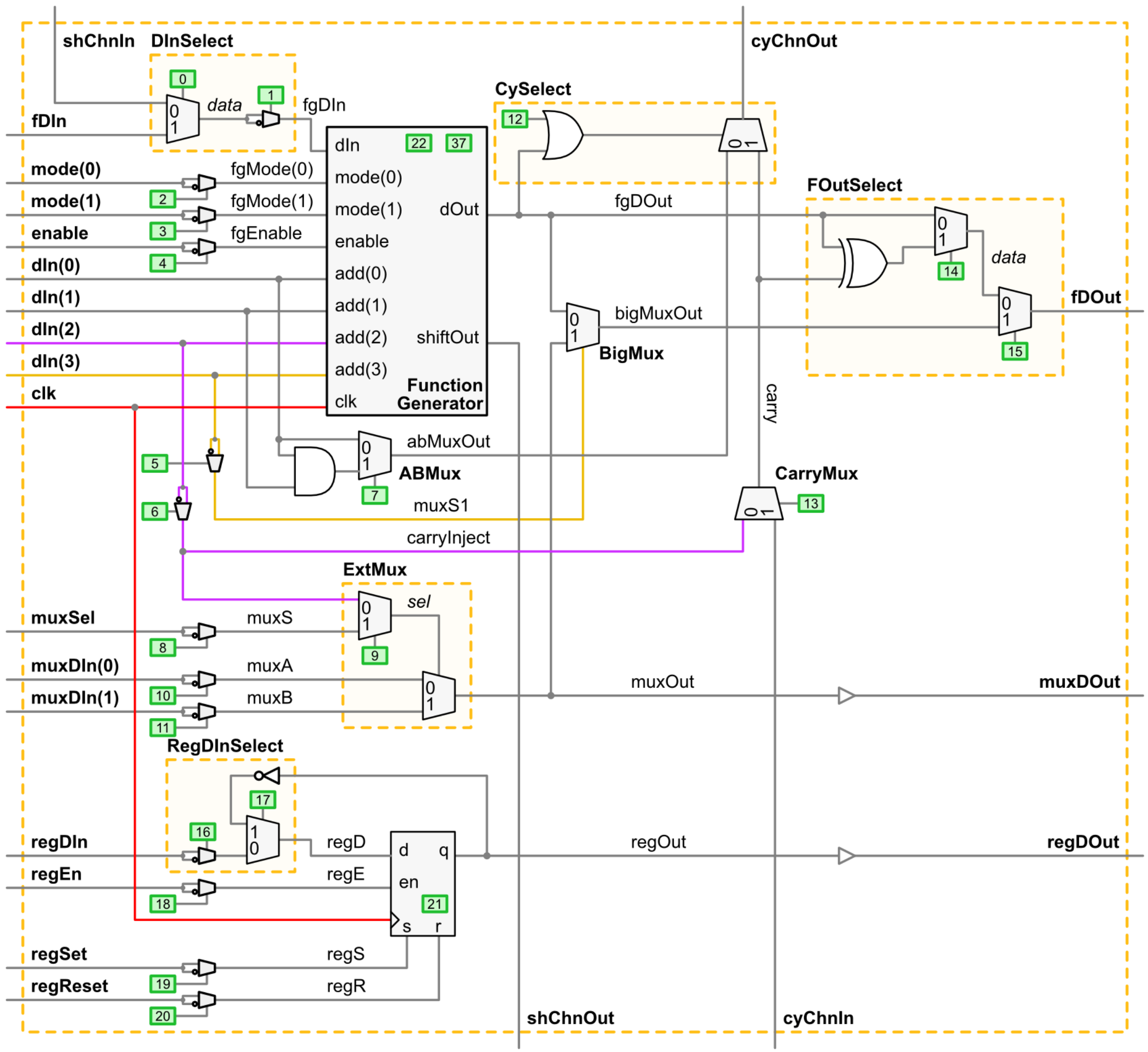
  - Processor Array
    - Systolic Arrays
  - FPGA
    - Normal FPGA Applications
  - Processor/FPGA Array
    - Intrinsic reconfiguration
- Seamless creation of larger array by connecting devices

RISA FPGA Fabric



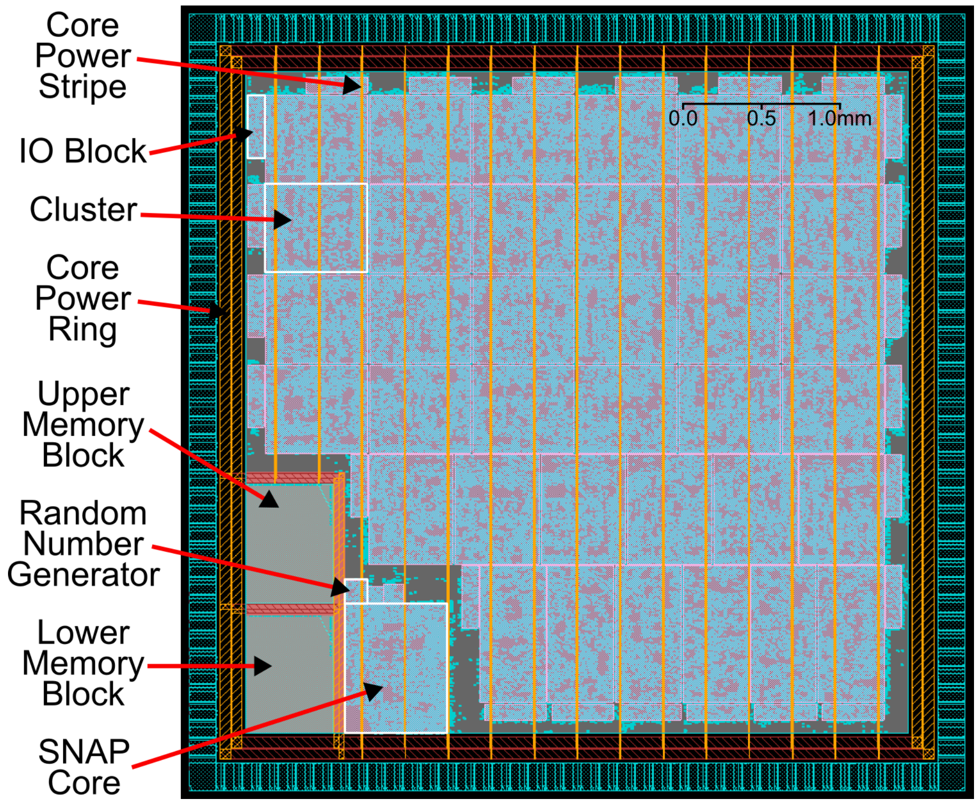
- The RISA FPGA Fabric provides a platform for combinatorial circuit evolution

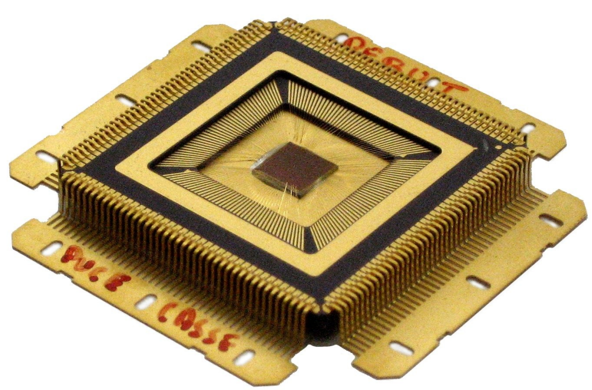
RISA FPGA Fabric - Function Unit



- Function Generator
  - 4 input LUT
  - 4x1 RAM Block
  - Variable length Shift Register
- Full Adder Gates
- Dedicated Multiplexor
- D Flip Flop
- Carry Chain
- Shift Chain

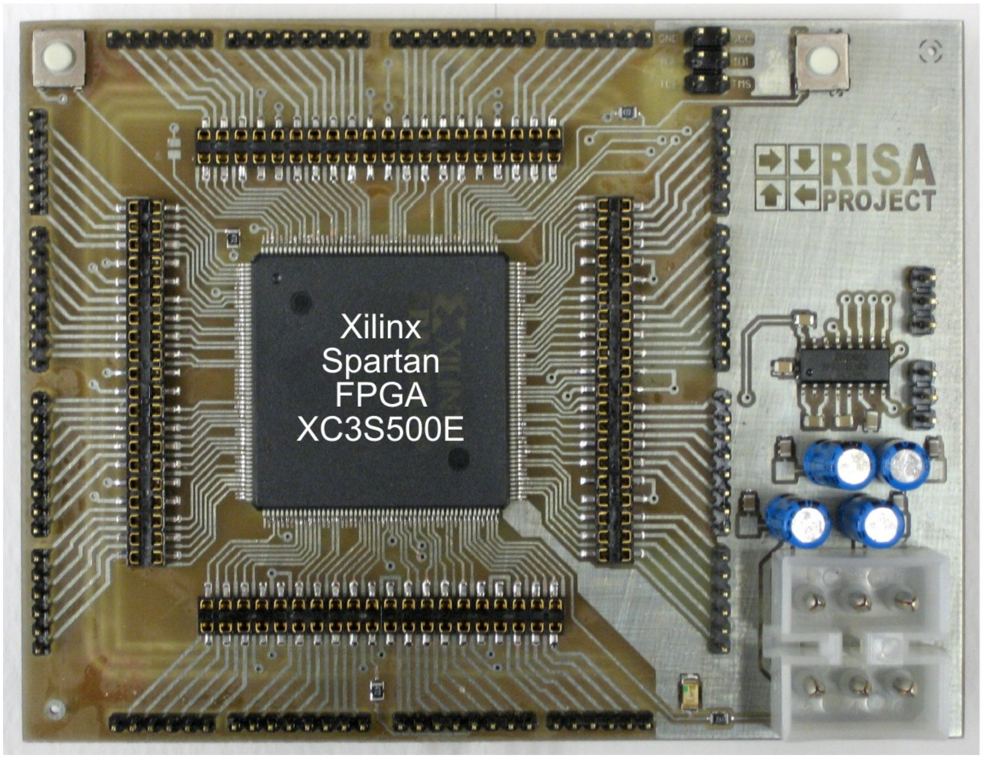
Fabrication

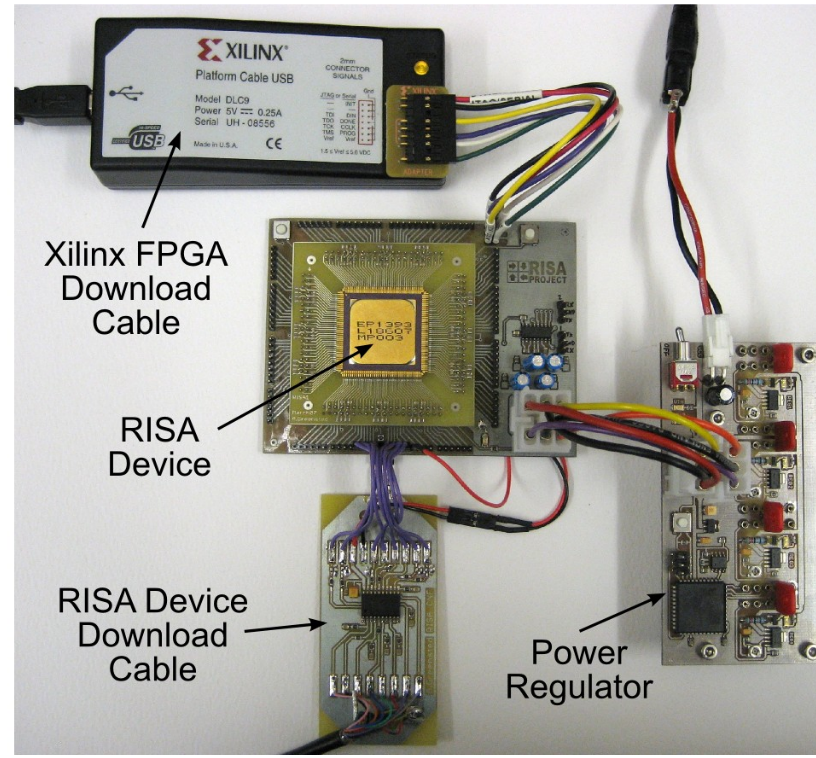




- Each device contains 1 RISA cell
  - 6x6 Cluster FPGA Fabric
  - 6 IO Blocks per side
  - 1 SNAP core
  - 16kB Memory (8192 16b words)
- 180nm process
- 5x5mm die area
- Cell based ASIC

First Experiments using the RISA Platform

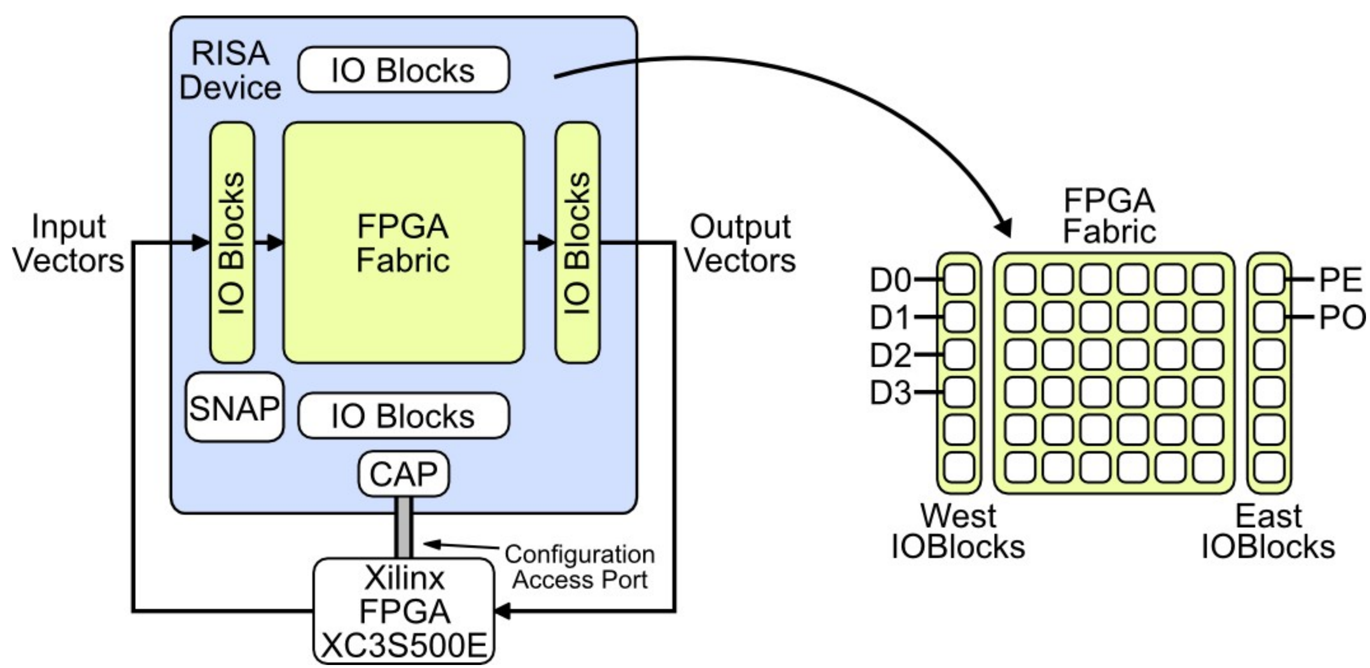




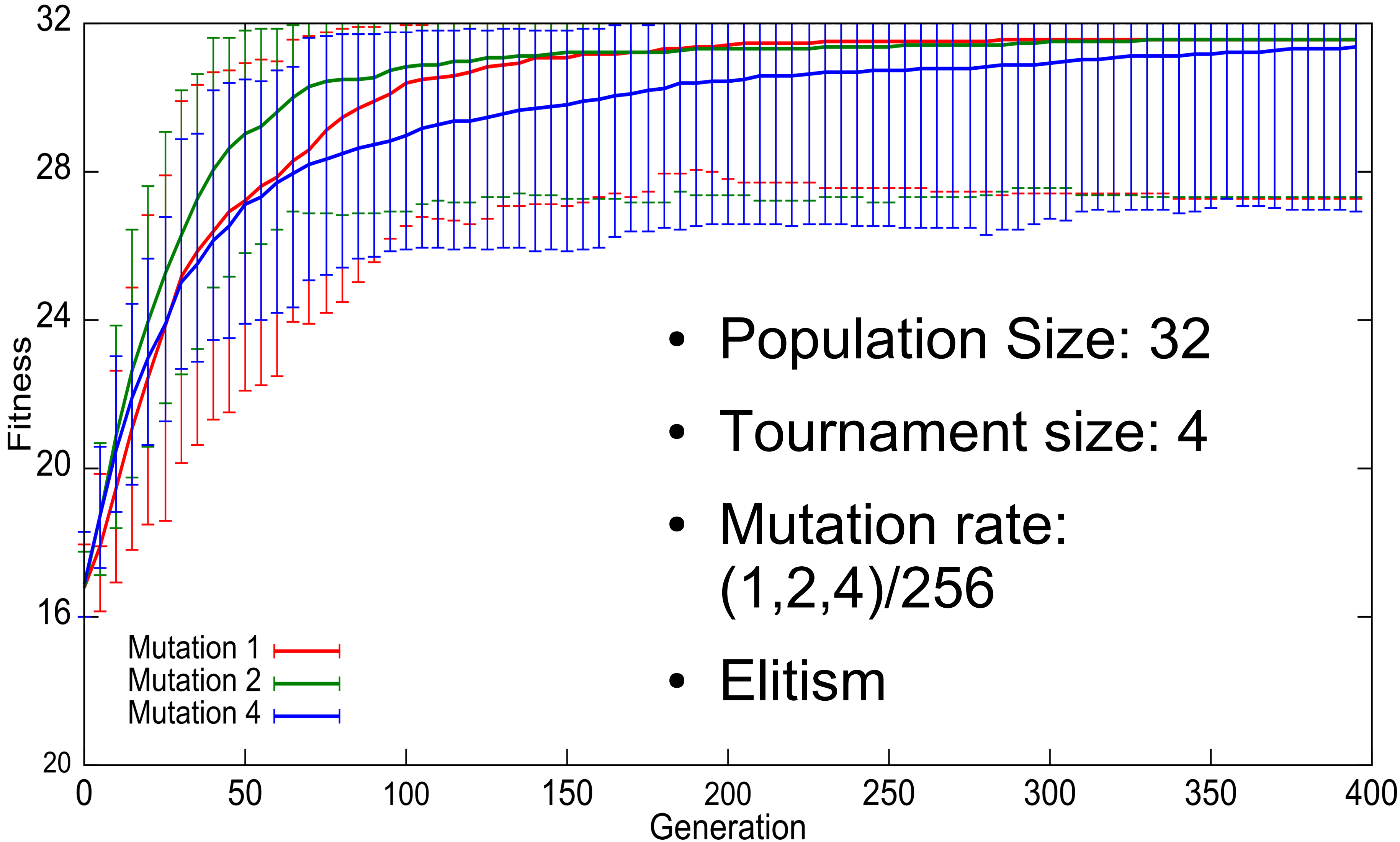
- Xilinx Spartan-3E device connected under RISA device
  - Able to apply test vectors to RISA device
  - The spartan controls RISA configuration

Evolving a simple digital circuit

- A simple evolvable hardware experiment for testing the device
  - Evolution of a 4 bit parity generator (Both even and odd parity)
- Evolutionary algorithm runs on a Microblaze core within the Spartan FPGA
- Candidate solutions loaded into RISA FPGA fabric and test vectors applied through RISA IO Blocks



Fitness Curves



- Population Size: 32
- Tournament size: 4
- Mutation rate: (1,2,4)/256
- Elitism

Extrinsic Evolvable Hardware on the RISA Architecture, Andrew Greensted and Andy Tyrrell, ICES 2007, LNCS Evolvable Systems: From Biology to Hardware, 4684:244-255, Wuhan, China, September 2007

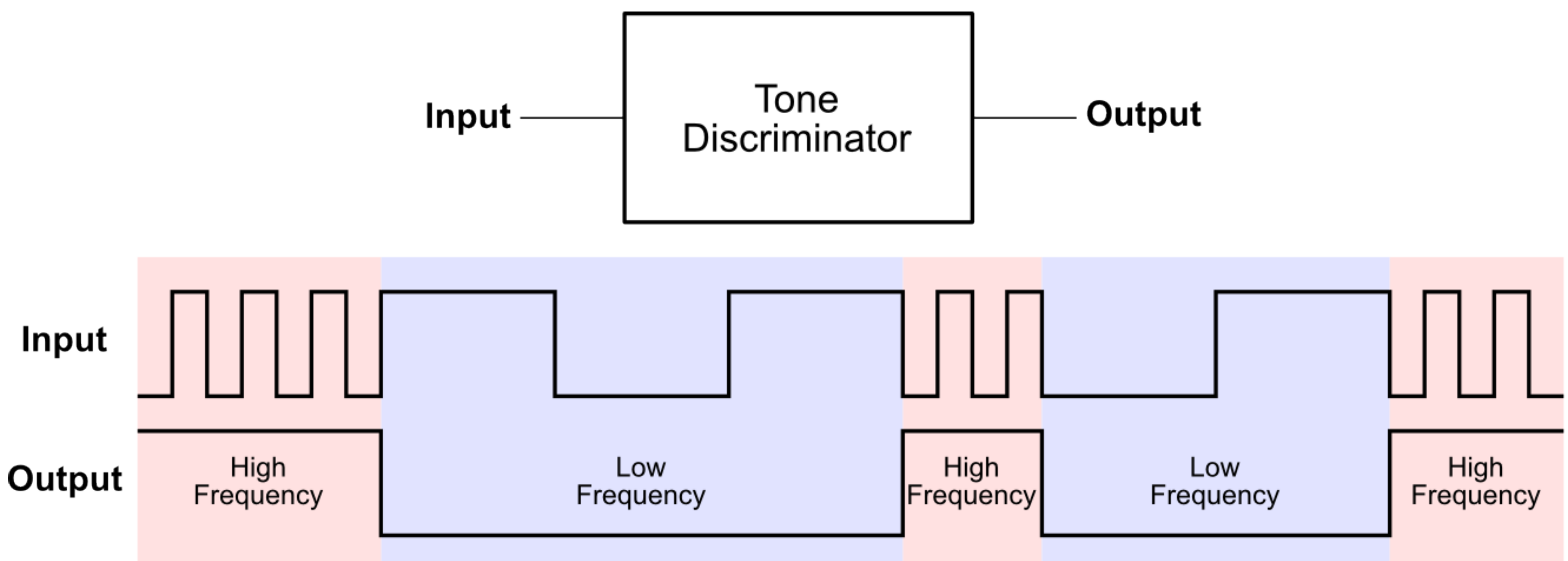
Case Study 3
Tone Discriminator

Thompson's Tone Discriminator

• Adrian Thompson (Sussex University)

• The aim was to evolve a digital circuit that could discriminate between two frequencies of a signal input to the system.

• The experiment was conducted in 1996

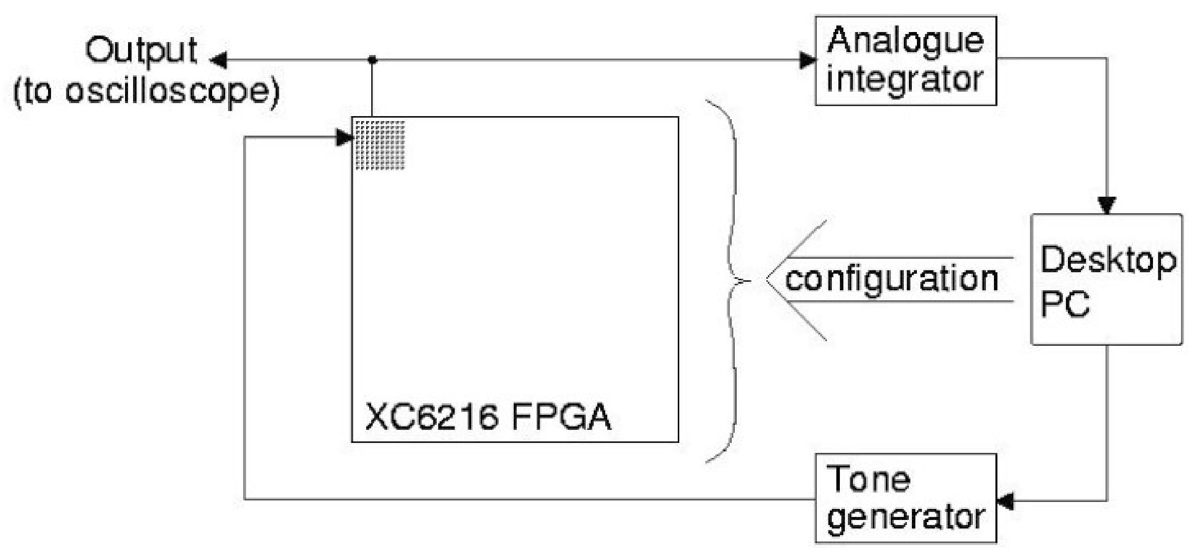


An Evolved Circuit, Intrinsic in Silicon, Entwined with Physics. Adrian Thompson, 1st International Conference on Evolvable Systems 1996, (c) Springer Verlag 1997

Thompson's Tone Discriminator

A Xilinx XC6216 was used to evolve the tone discriminator.

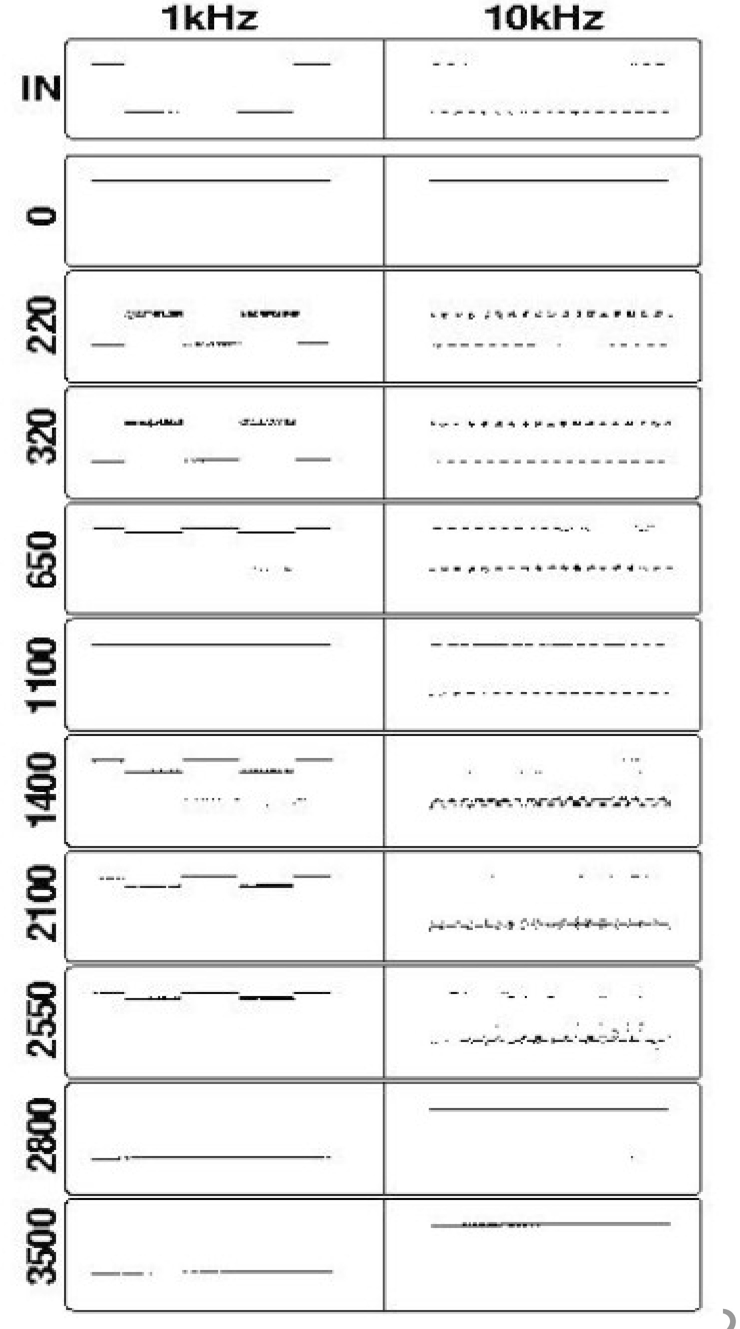
- No clock input signal was used.
  - Therefore there was no timing reference
- The evolved circuit had to discriminate between a 1KHz and 10KHz input signal.
- The area used by evolution was limited to a 10x10 array of cells in one corner. 100 out of the device's 4096 cells.



Tone Discriminator Results

- Signal propagation time in the XC6216 technology is in the order of nanoseconds.

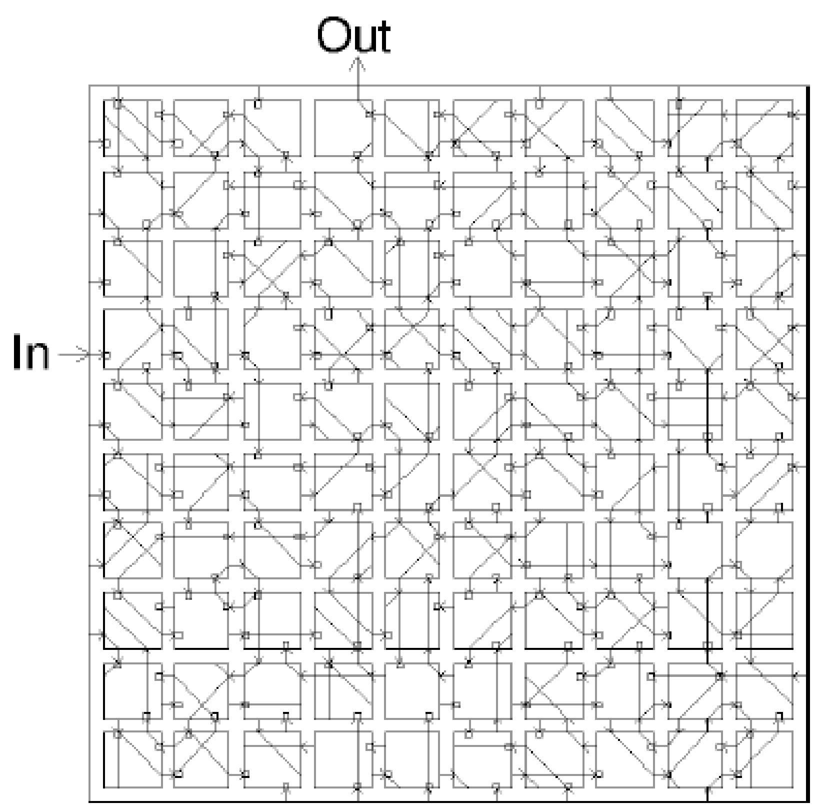
- The circuit was still able to discriminate between signals with periods in the order of milliseconds

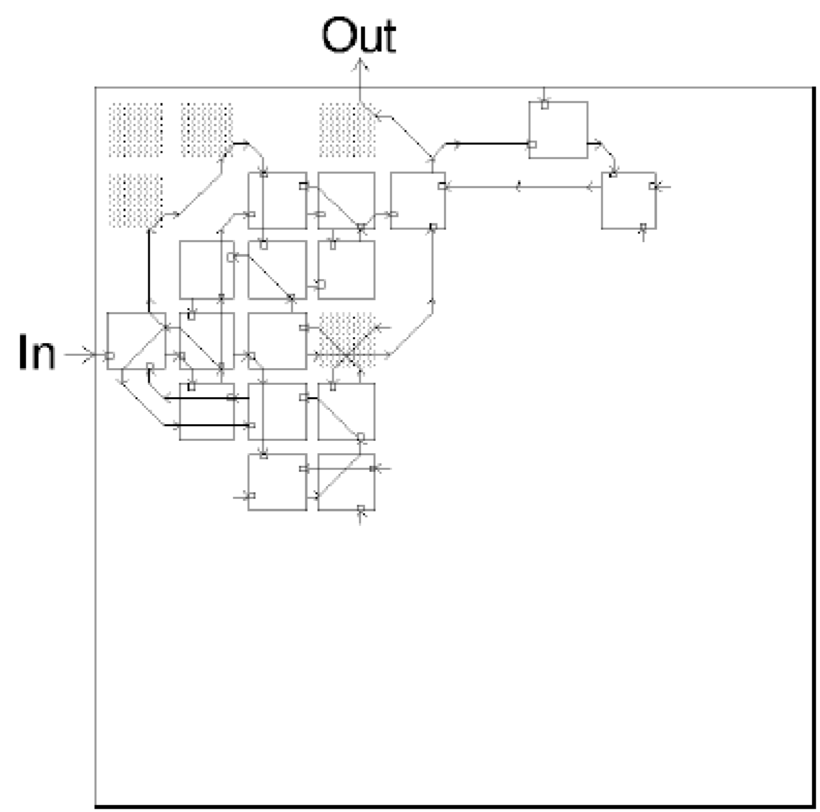


Tone Discriminator Results

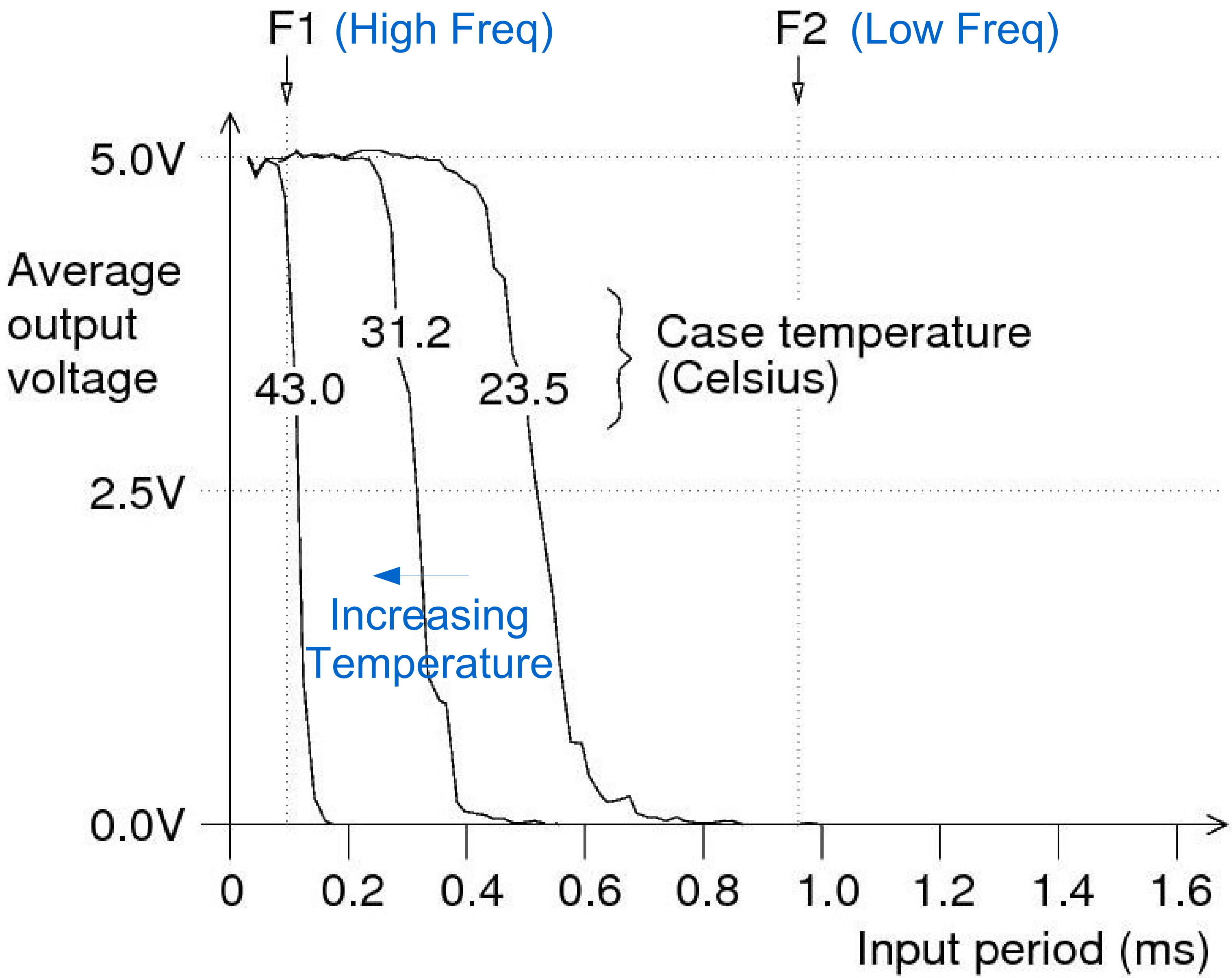
A successful individual (after 5000 generations).

- The left hand image shows the cell connectivity for the full 10x10 array.
- The right hand image shows the cells required for correct operation.
- The cells shaded grey contain no connected logic, but their configuration is required for correct operation.
- The solution was very device and environment dependent.





Tone Discriminator Results



Case Study 4
Prime Number Generator

The Competition

- GECCO 2006 (Genetic and Evolutionary Computation Conference) Consecutive Primes Competition

Evolve a polynomial with integer coefficients such that given an integer value i as input produces the i^{th} prime number, p(i), for the largest possible value of i.

So, if f(i) is the evolved function, we expect:

f(1)=2,
f(2)=3,
f(3)=5,
f(4)=7,
f(5)=11, etc...

Prime Generating Functions

Polynomial	Len	All Positive	All Integer coefficients?	Discoverer
$i^2 + i + 41$	40	Y	Y	Euler
$36i^2 - 810i + 2753$	45	N	Y	Ruby, Fung
$i^5 - 61i^4 + 1339i^3 - 12523i^2 + 42398i + 11699$	41	Y	Y	Wroblewski, Meyrignac
$(i^5 - 133i^4 + 6729i^3 - 158379i^2 + 1720294i - 6823316)/4$	57	N	N	Sunder Gupta
$44546738095860i + 56211383760397$	23	Y	Y	Frind, Jobling, Underwood

The best so far...

Adapting to CGP

Convert polynomial to a boolean form:

$p(i) = b_m 2^m + b_{m-1} 2^{m-1} + \ldots + b_4 2^4 + b_3 2^3 + b_2 2^2 + b_1 2^1 + b_0 2^0$

Each coefficient is a function of the binary values of the input prime number:

$b_m = f_m(a_0, a_1, \ldots, a_n)$

...

$b_1 = f_1(a_0, a_1, \ldots, a_n)$

$b_0 = f_0(a_0, a_1, \ldots, a_n)$

Where:

$a \in \{0,1\}$

$b \in \{0,1\}$

- Multi-Chromosome CGP was used:
  - Essentially one instance of CGP per coefficient.
- Available operations:
  - AND NAND, OR, NOR, XOR, XNOR, AND ($!in_0$), OR ($!in_0$)

16 Consecutive Primes

$p(i) = b_5 2^5 + b_4 2^4 + b_3 2^3 + b_2 2^2 + b_1 2^1 + b_0 2^0$ where

$$b_5 = a_0 + a_1a_2 - 2a_0a_1a_2$$

$$b_4 = -((a_1(2a_2 - 1) - a_2)(1 + a_2(a_3a_4 - 1))) + a_0(1 - 2a_2 + a_2^2(2 - 2a_3a_4) + 2a_1(2a_2 - 1)(1 + a_2(a_3a_4 - 1)))$$

$$b_3 = a_2(a_3 + a_4 - 2a_3a_4) + a_1a_3(1 + a_2(2a_3a_4 - a_3 - a_4))$$

$$b_2 = 1 - a_3 - a_4 + 2a_3^2a_4 - 2a_2^4(2a_3 - 1)^3(a_4 - 1)a_4 + 2a_3a_4^2 - 2a_3^2a_4^2 - a_2^2(2a_3 - 1)(a_3(2 - 6a_4) + (3 - 2a_4)a_4 + 6a_3^2(a_4 - 1)a_4) + a_2^3(1 - 2a_3)^2(1 - (1 + 6a_3)a_4 + (6a_3 - 2)a_4^2) + a_2(2a_3^3(a_4 - 1)a_4 + 2a_4^2 + a_3(2 + 5a_4 - 8a_4^2) + a_3^2(1 - 9a_4 + 6a_4^2) - 1) + a_1(1 - a_2a_3 + a_2^2(2a_3 - 1))(2a_3 + 2a_4 - 1 - a_3a_4 - 2a_3^2a_4 + 2a_2^4(2a_3 - 1)^3(a_4 - 1)a_4 - 2a_3a_4^2 + 2a_3^2a_4^2 + 2a_2^2(2a_3 - 1)(a_3(2 - 4a_4) - (a_4 - 2)a_4 + 3a_3^2(a_4 - 1)a_4) - 2a_2^3(1 - 2a_3)^2(1 - (1 + 3a_3)a_4 + (3a_3 - 1)a_4^2) - a_2(a_4 - 2 + 2a_3^3(a_4 - 1)a_4 + 2a_4^2 + a_3(4 + 4a_4 - 8a_4^2) + 2a_3^2(1 - 5a_4 + 3a_4^2)))$$

$$b_1 = 1 - a_3 + a_3^2 - a_2a_3^2 - a_1(a_2(1 + a_3^2 + a_3(a_4 - 3)) + a_3^2(1 - 2a_4) + a_2^2a_3(2a_3 - 1)(a_4 - 1)) - a_3^2a_4 + a_2a_3^2a_4 + a_1^2(a_2 - 1)a_2a_3(2a_3 - 1)(2a_4 - 1) - a_0(2a_1a_2 - 1)(a_3 - 1)(1 + (a_2 - 1)a_3(1 - a_4 + a_1(2a_4 - 1)))$$

$$b_0 = a_2 - a_0(a_1 - 1)(a_2 - 1)(a_3 - 1) + a_1(a_2 - 1)(a_3 - 1) + a_3 - a_2a_3$$

These equations have been simplified from their boolean operation form.

16 Consecutive Primes