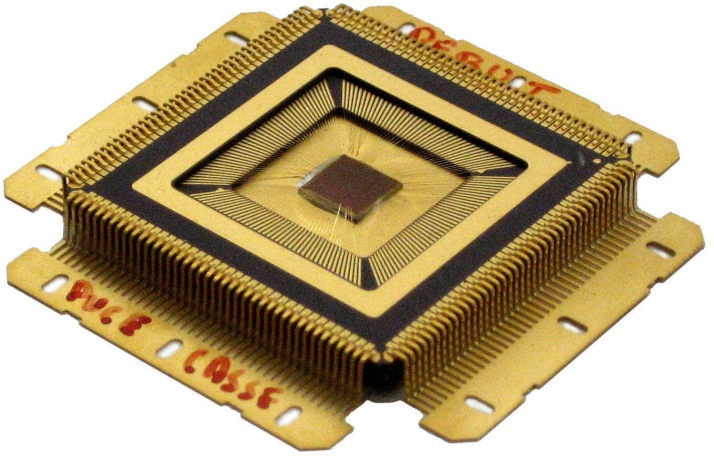
Evolving Digital Hardware
EHW Module 2008



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

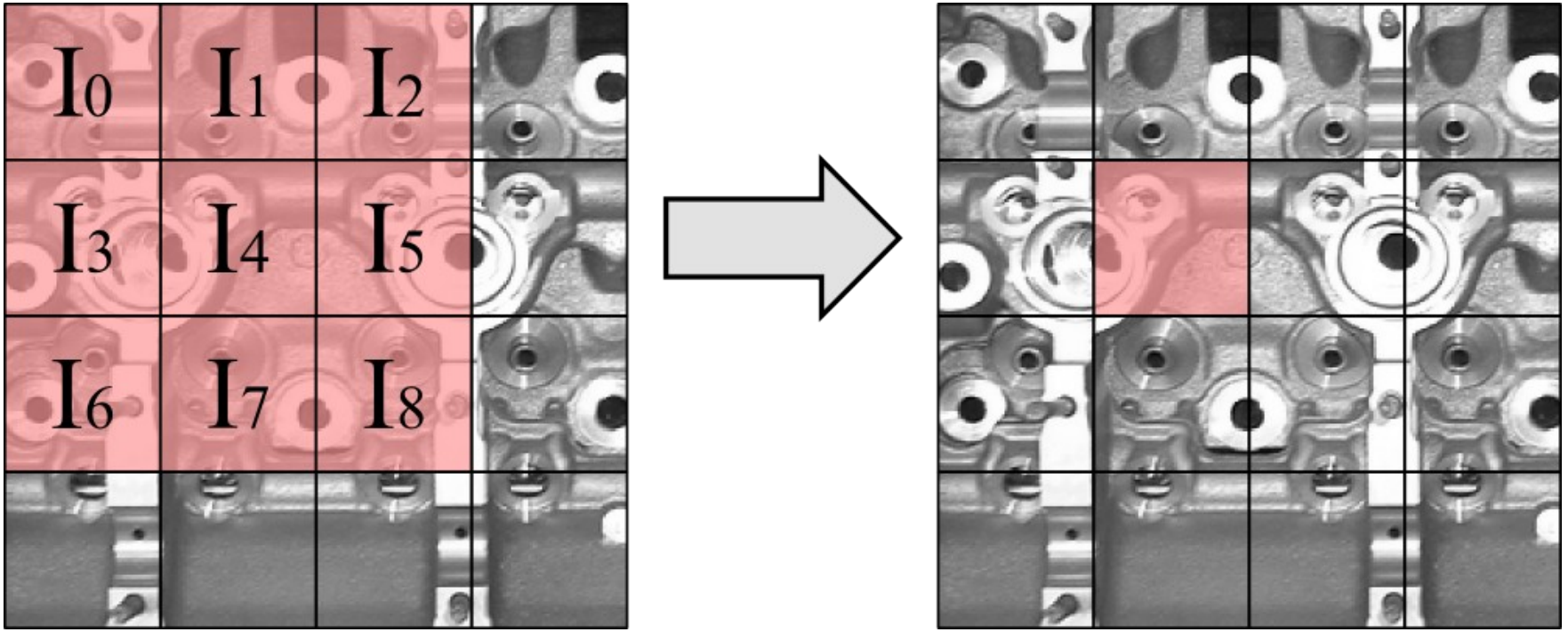
www.bioinspired.com/users/ajg112/teaching/evoHW.shtml

Lecture 4

Case Study 5
CGP Image Filter

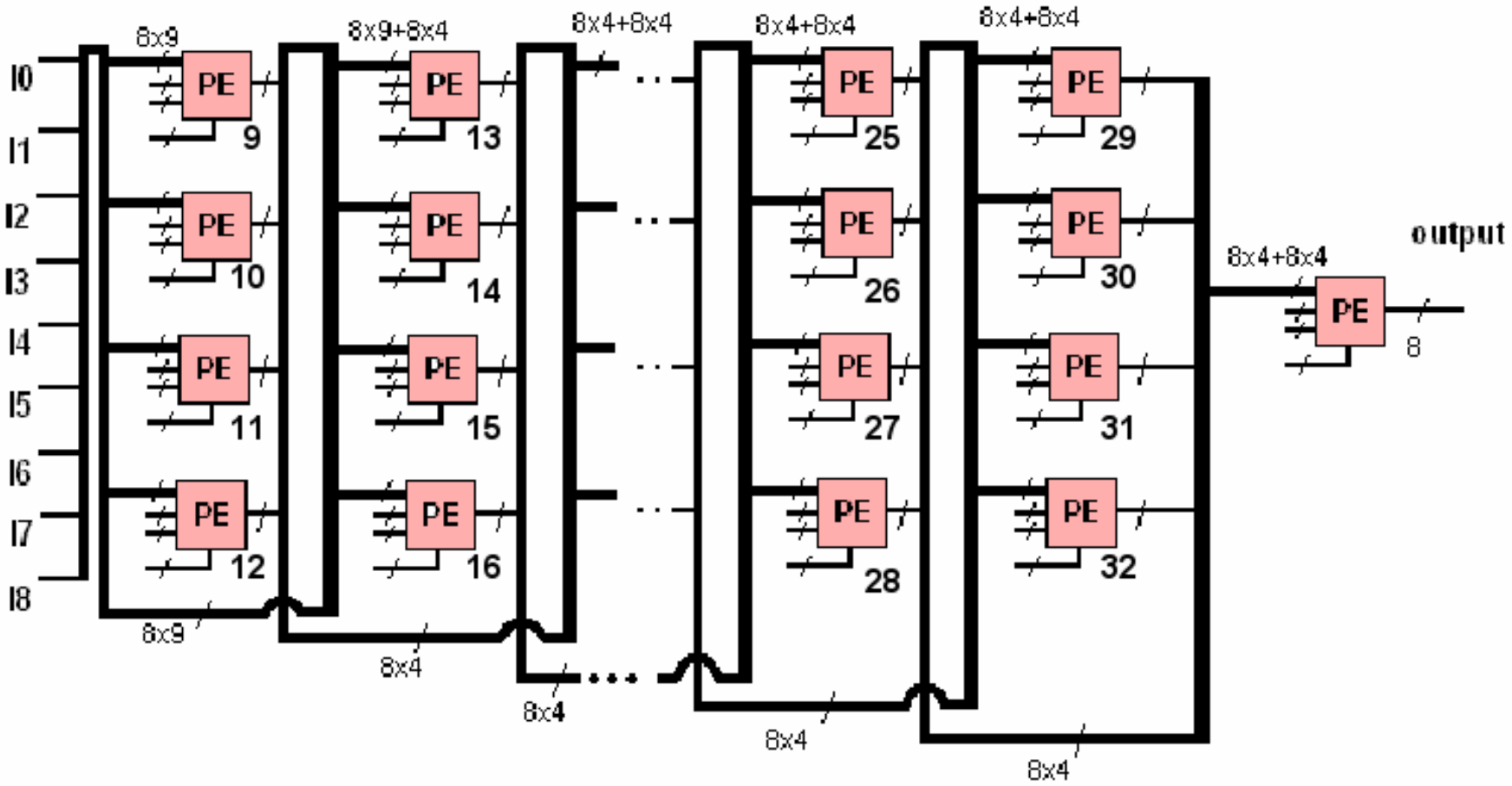
Intrinsic Evolution of a Digital Filter

- This case study uses Cartesian Genetic Programming to evolve a digital filter.
- The processing array's input is 9 pixel values (each a 8-bit grayscale). The output is a filtered pixel value.



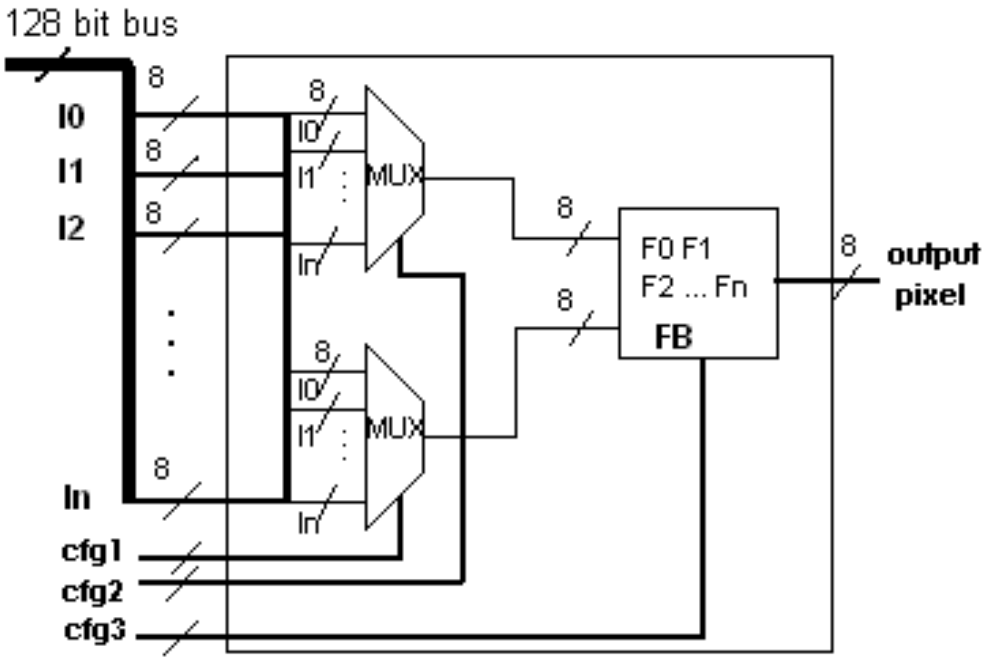
Digital Circuit Design using Intrinsic Evovable Hardware, Yang Zhang, Stephen Smith, Andy Tyrrell, 2004 NASA/DoD Conference on Evolvable Hardware.

Configurable Circuit Structure



• Each cell input must be able to connect to any previous cell's output.

The Processing Element



Code	Function	Code	Function
F0: 0000	X >> 1	F8: 1000 (3)	(X+Y+1) >> 1
F1: 0001	X >> 2	F9: 1001	X & 0x0F
F2: 0010	~ X	F10: 1010	X & 0xF0
F3: 0011	X & Y	F11: 1011	X | 0x0F
F4: 0100	X | Y	F12: 1100	X | 0x F0
F5: 0101 (1)	X ^ Y	F13: 1101 (4)	(X&0x0F) | (Y&0xF0)
F6: 0110	X + Y	F14: 1110	(X&0x0F) ^ (Y&0xF0)
F7: 0111 (2)	(X+Y) >> 1	F15: 1111	(X&0x0F) & (Y&0xF0)

Chromosome Encoding

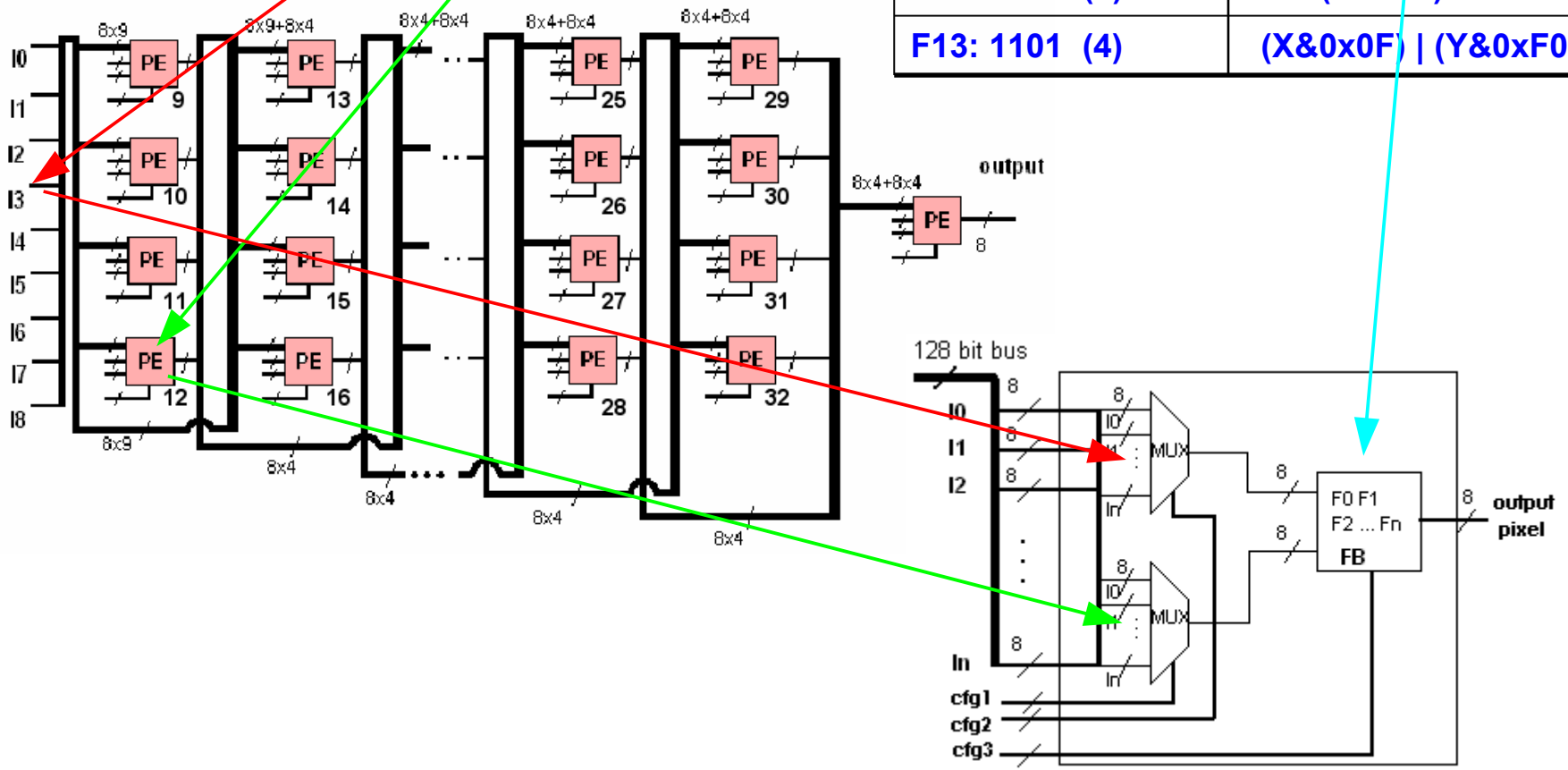
2 7 3 8 6 3 3 4 1 1 7 2 0 5 3 **3 12 1** 13 4 2 10 11 1 9 2 3 3 13 1 13 18 1

16 15 3 11 17 2 23 17 4 21 32 2 24 28 1 25 27 2 22 26 1 29 32 4

Input Input Function

3 12 1

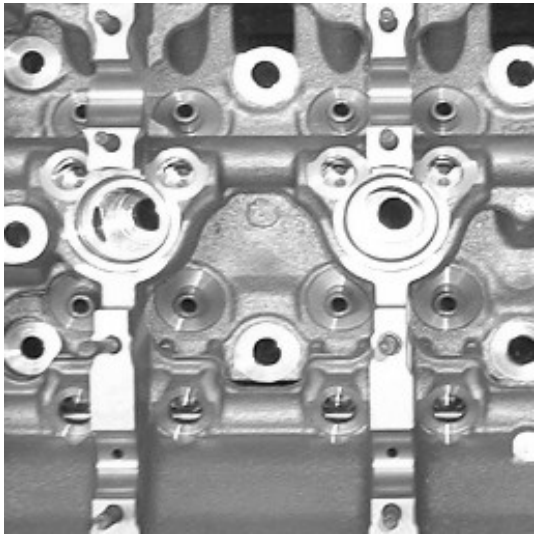
F5: 0101 (1)	X ^ Y
F7: 0111 (2)	(X+Y) >> 1
F8: 1000 (3)	(X+Y+1) >> 1
F13: 1101 (4)	(X&0x0F) | (Y&0xF0)



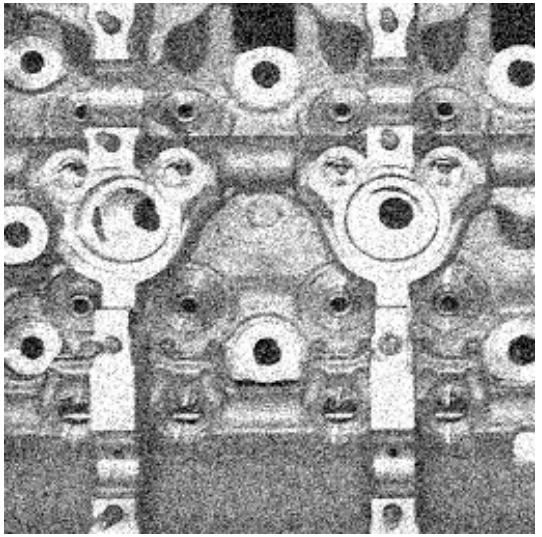
The Evolutionary Algorithm

- Population of 16 individuals.
- Initial population generated randomly.
- The individual with the highest fitness is selected as a template for next generation.
- The next generation is created from mutations of the template.
- The level of mutation is inversely proportional to the template's fitness.
- Fitness is measured using:
  - The picture's signal-to-noise ratio.
  - The mean difference per pixel.

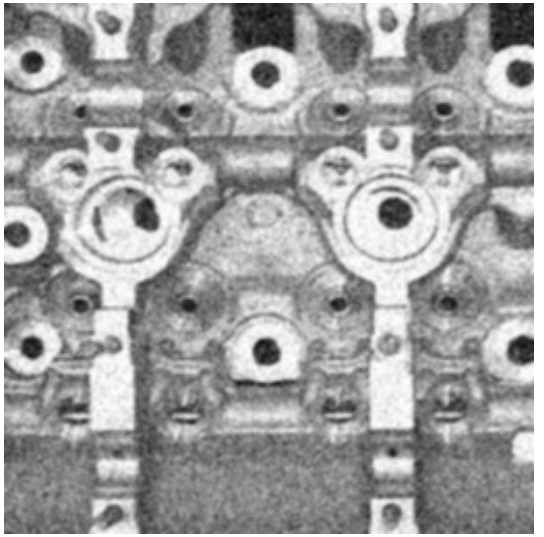
Results



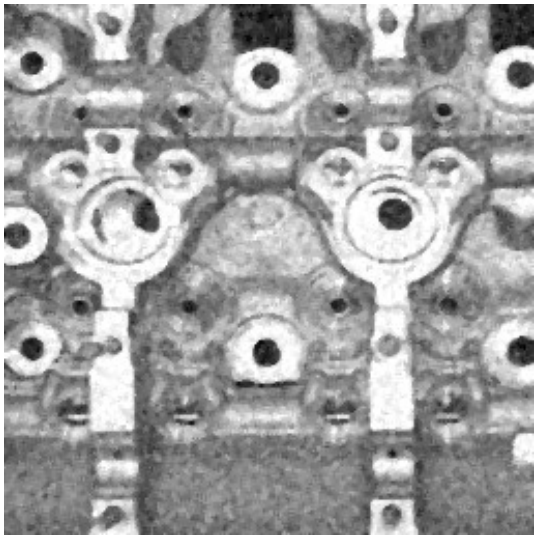
Original



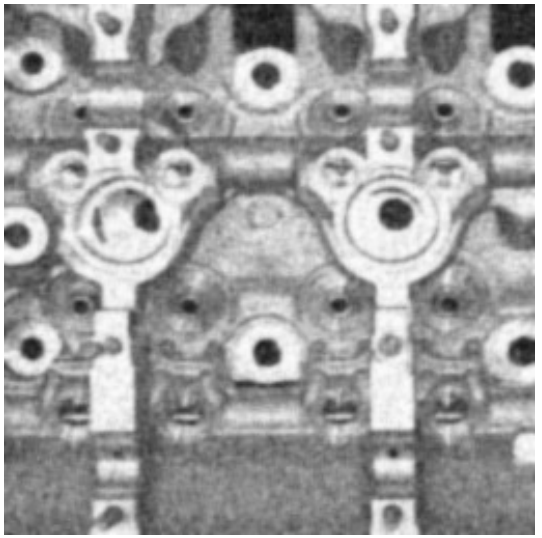
Gaussian Noise Applied
(SD=32)



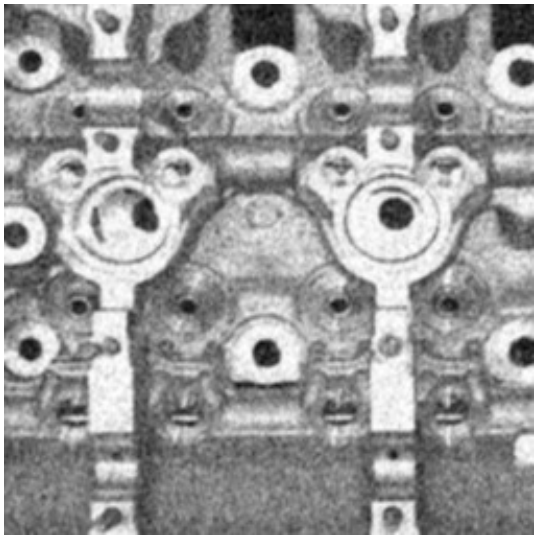
Mean Filter



Median Filter



Gaussian Filter



EHW Filter

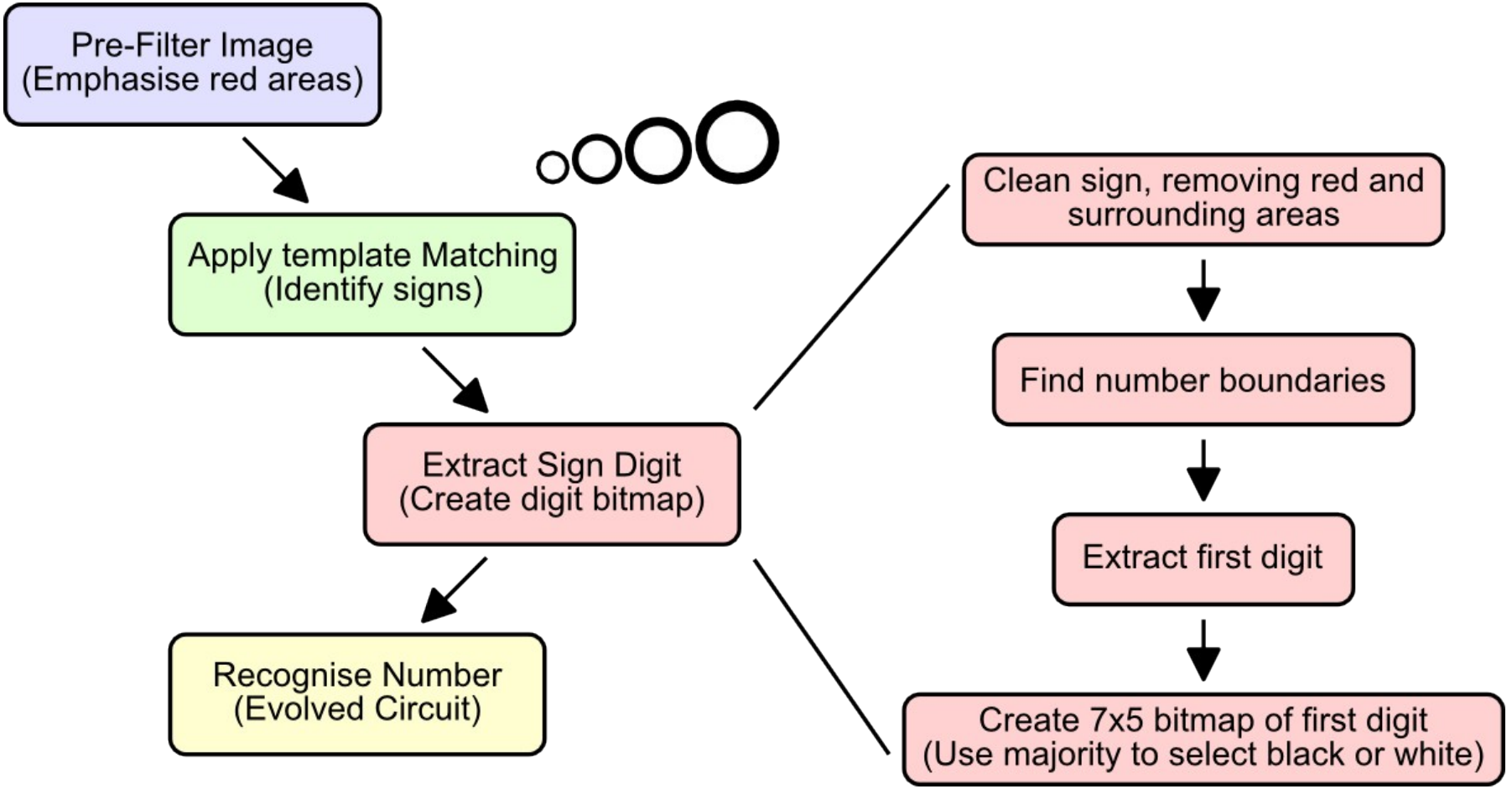
Case Study 6
Speed Limit Sign Recognition

Speed Limit Sign Recognition

- An evolved circuit is used to recognise speed limit signs.

Recognizing Speed Limit Sign Numbers by Evolvable Hardware, Jim Torresen, Jorgen W. Bakke and Lukas Sekanina, 8th International Conference on Parallel Problem Solving from Nature (PPSN VIII) , September 2004

Pre-Processing Stages



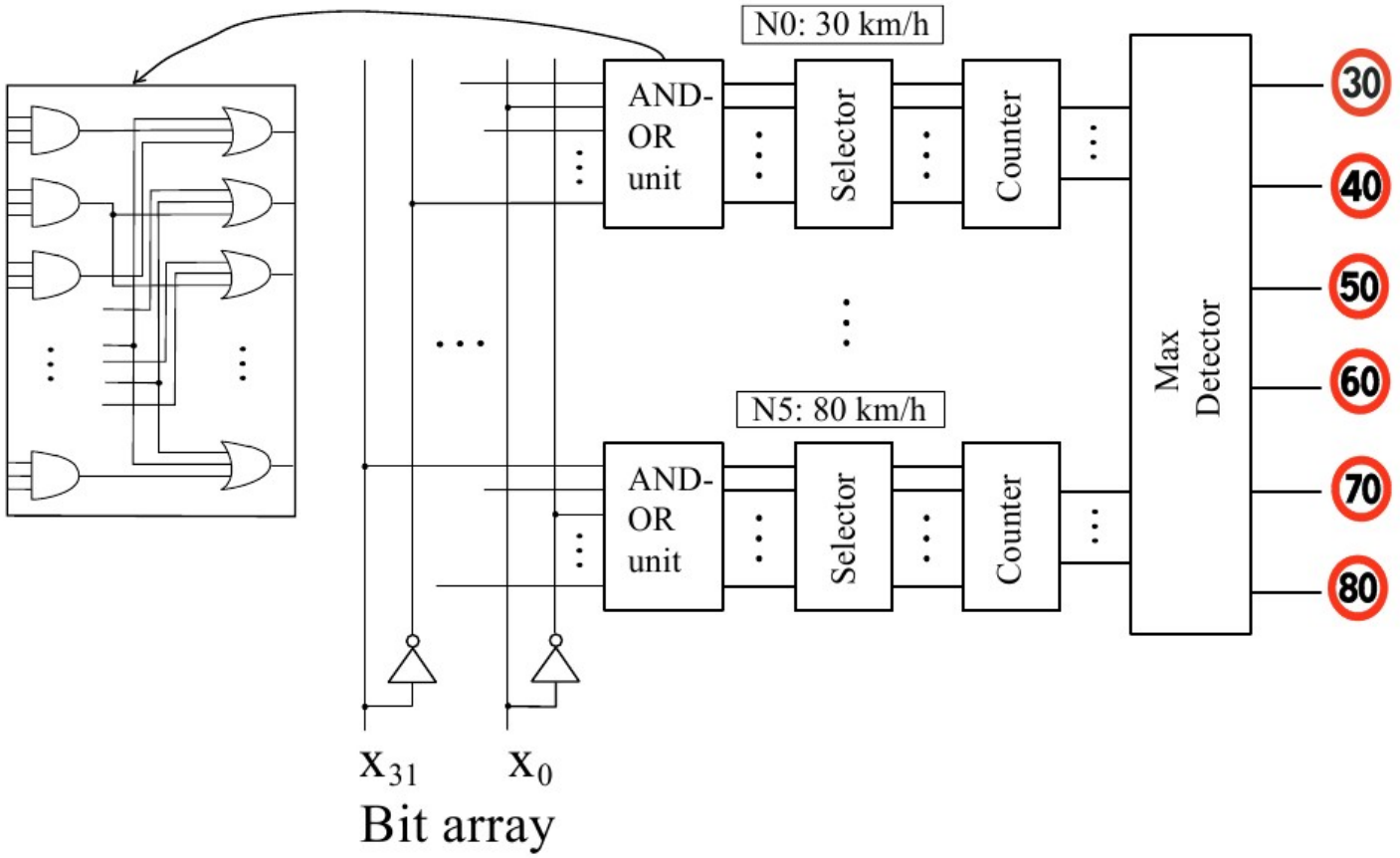
- A number of stages are required before the evolved number recognition circuit is used.

Recognition System

- After the pre-filtering stage, a 7x5 monochrome bitmap of the speed limit is produced.

The Classifier

- One subsystem for each digit to be classified (3, 4, 5, 6, 7 & 8)
- 5x7 = 35, 32 bit inputs are used (3 are discarded)
- AND/OR Unit and Selector Evolved
- Each counter counts the number of '1's from its associated Selector
- Maximise counter output when presented with associated digit.



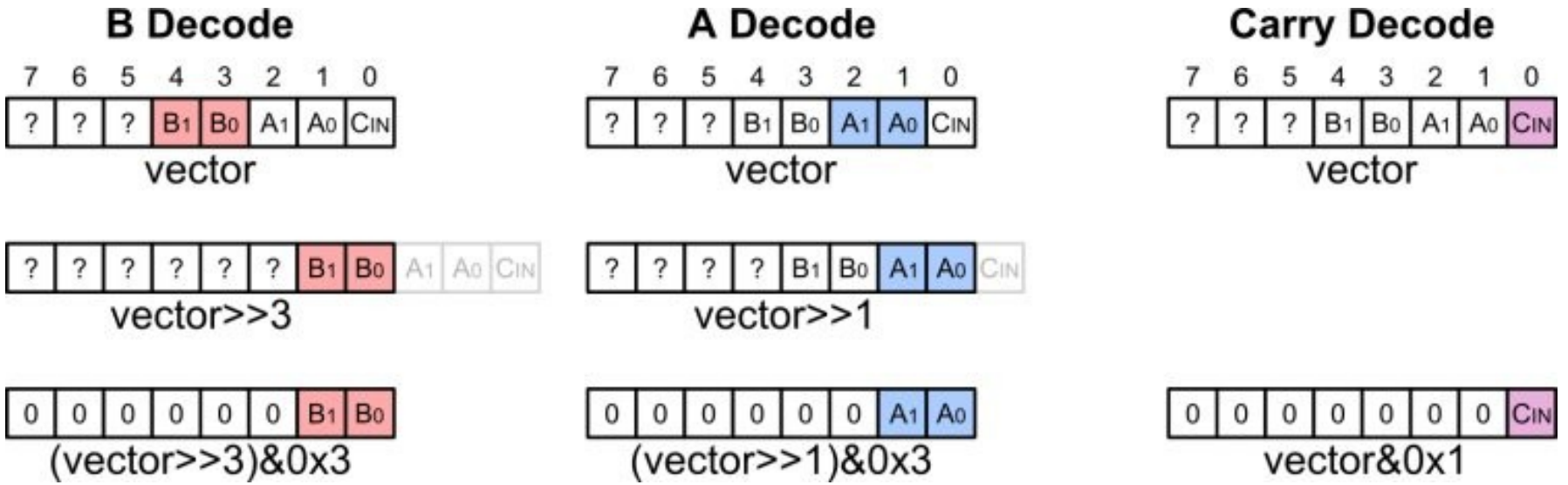
Results

Speed Limit	30	40	50	60	70	80
Performance (in %)	100	100	84.4	91.4	100	100

Lab Solutions

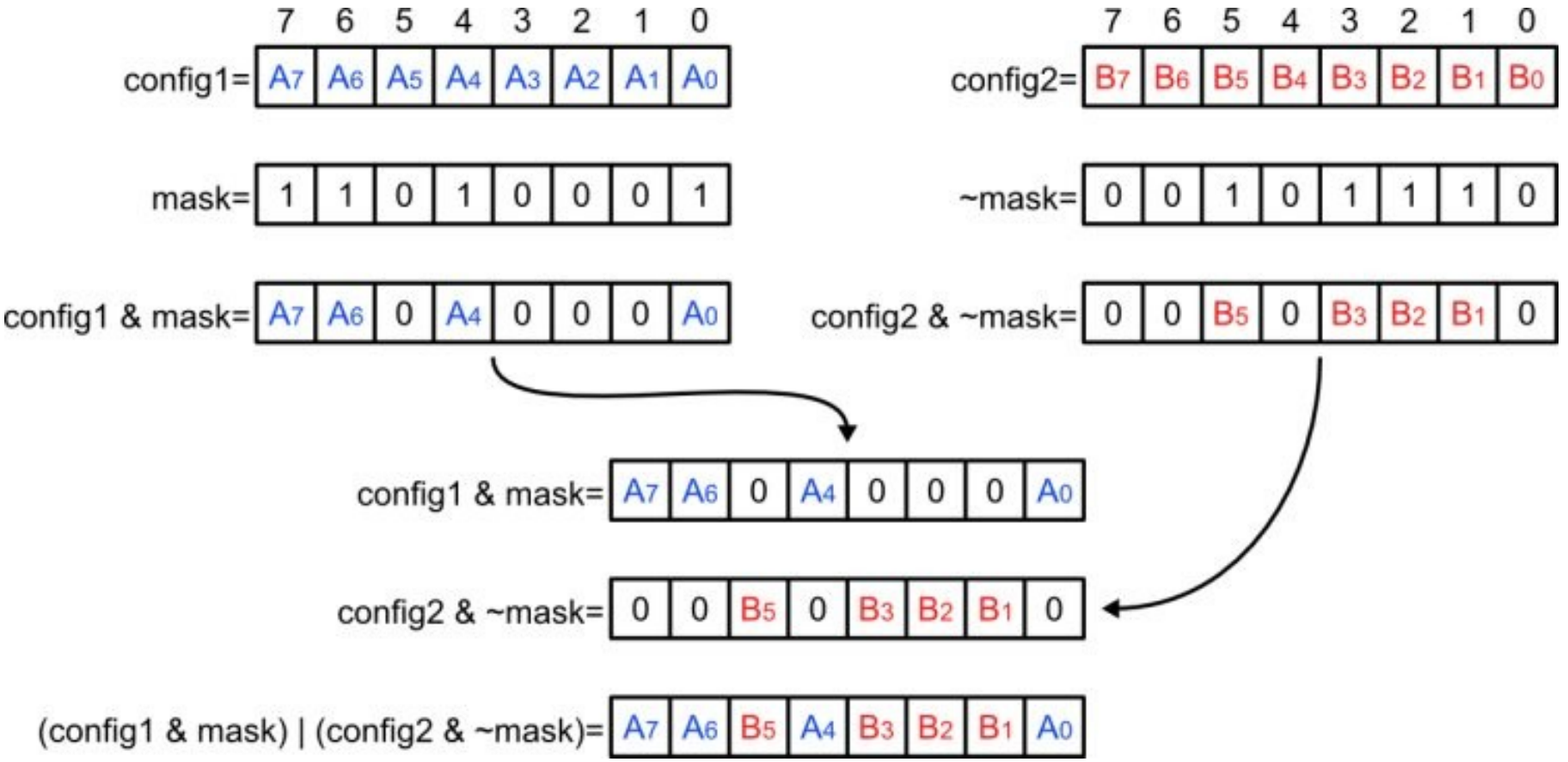
2-bit Adder Output Calculation

```
for(vector=0 ; vector<256 ; vector++)
{
   data = ((vector>>3)&0x3) + ((vector>>1)&0x3) + (vector&0x1);
   EVOBLOCK_writeTargetRAM(data);
}
```



Crossover

- A random configuration mask is created, that sets which parent the configuration will be copied from



Crossover Function

```
maskBit = 0;

for (cellNum=0 ; cellNum<NUM_CELLS ; cellNum++)
{
   mask = 0;

   for (bit=0 ; bit<NUM_CONFIG_BITS ; bit++)
   {
      // If crossing over, flip mask bit
      if ((rand() & 0xFF) < crossoverRate) maskBit ^= 0x1;

      mask |= (maskBit << bit);
   }

   // Get source config
   config1 = source1->cellConfigs[cellNum];
   config2 = source2->cellConfigs[cellNum];

   // Calculate and write config to destination
   dest->cellConfigs[cellNum] = (config1 & mask) | (config2 & ~mask);
}
```