Optimized digital design flow for embedded sensor applications using high level synthesis

Savino Bellopede, Marco Castellano, Ugo Garozzo
Digital Designer
STMicroelectronics

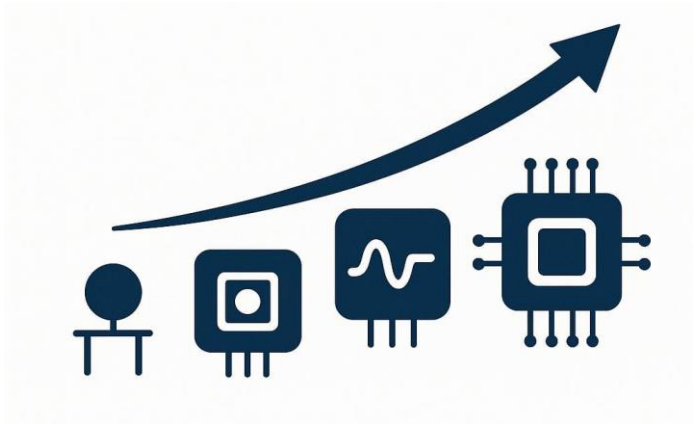
62 THE CHIPS TO SYSTEMS CONFERENCE

SPONSORED BY CEDA IEEE Council on Electronic Design Automation sigda acm

Motivation

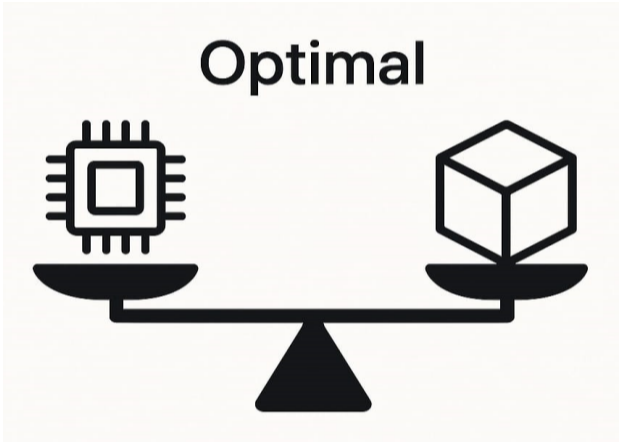
Sensors are evolving

From data to embedded applications



Algorithm quality/accuracy and area occupation

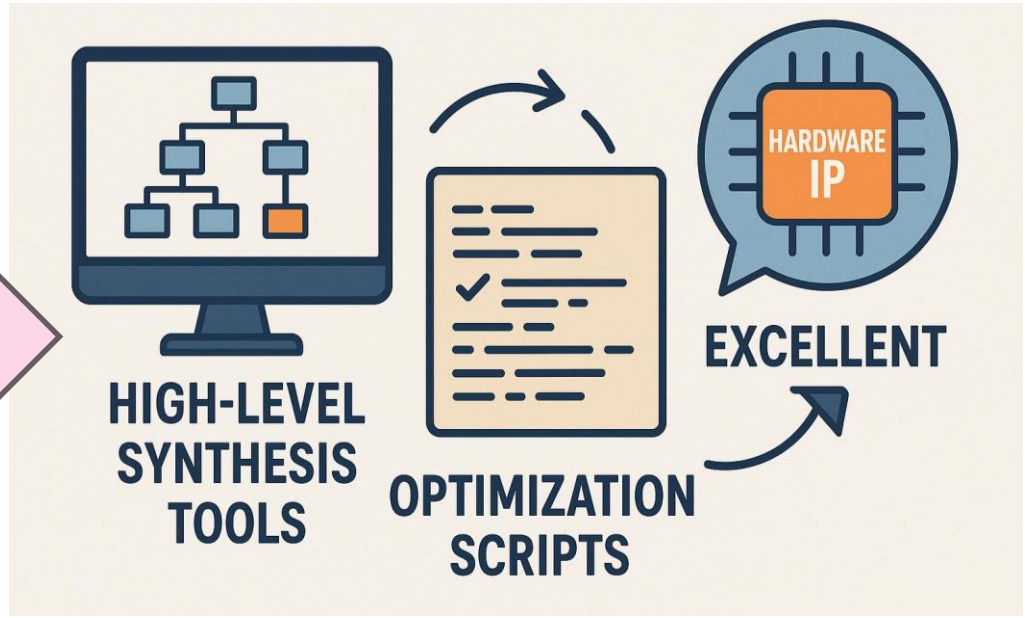
Best compromise between algorithm quality/accuracy and area occupation



Timelines & time to market

Project timelines are often very tight

Idea



Advantages of synergistic use of high-level synthesis tools and optimization scripts:

- HW implementation tool: HLS Catapult
- SW optimization tool: Python

The Synergy tool for best in-sensor embedded algorithms

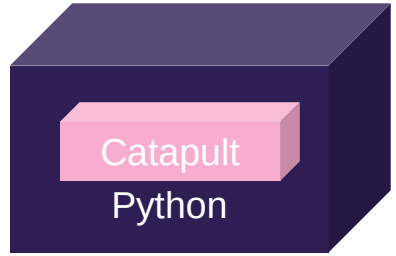
Enhanced productivity	Improved design quality	Integration and scalability	Resource management

Build a best-in-class embedded algorithm

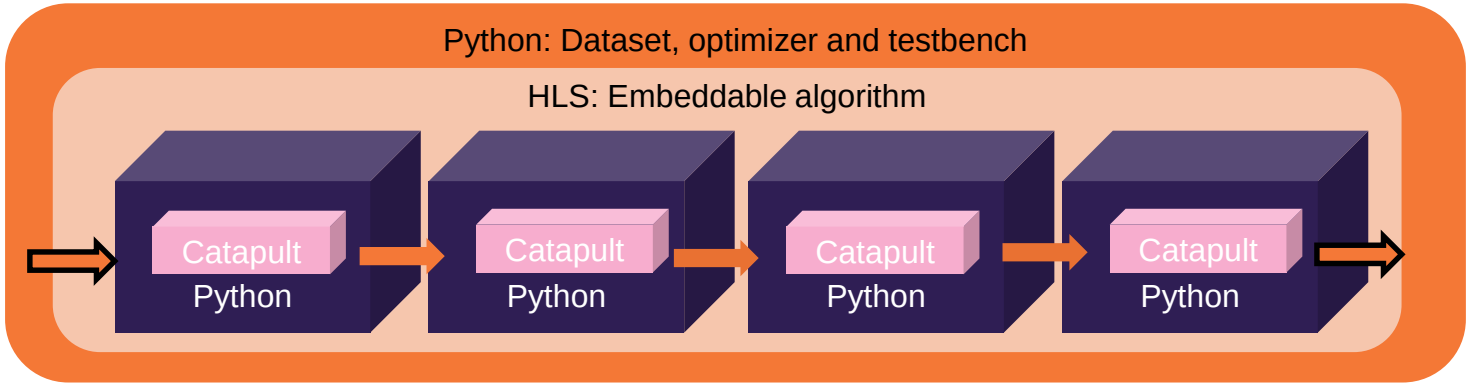
First pillar: basic bricks
Go up and down from algorithm to implementation and back

Second pillar: the chain
Embed an algorithm as a chain of basic bricks

Third pillar: the flow
Create a flow that uses the synergy of Catapult and python to find the best tradeoff between latency accuracy and area

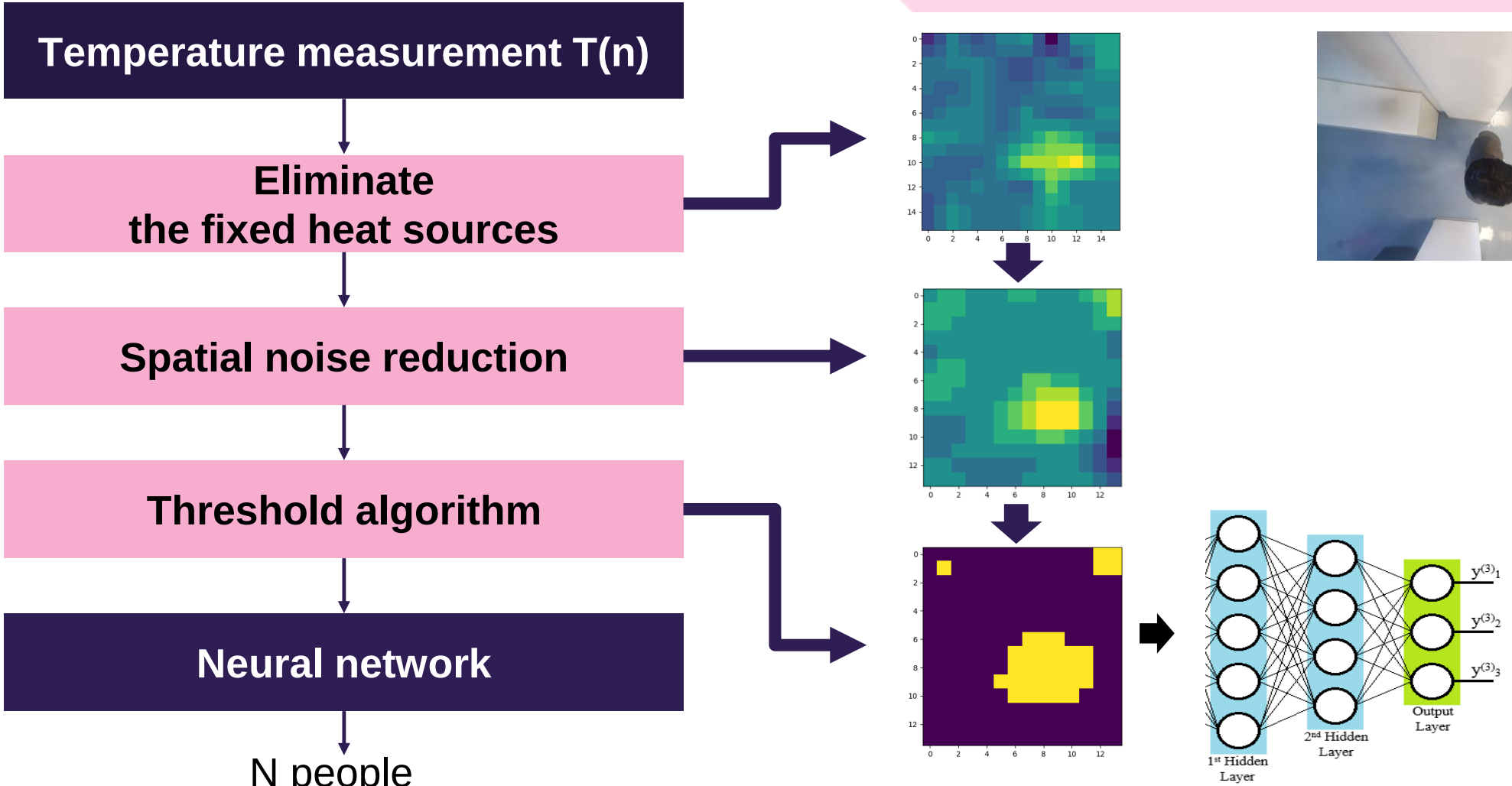


Basic bricks model



Evidence: people counting

Algorithm as chain of basic bricks



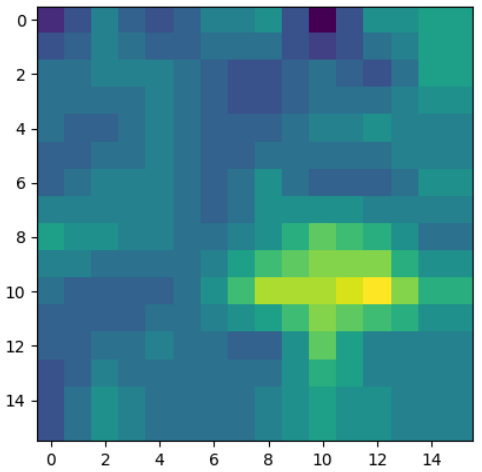
Brick1: eliminate fixed heat sources

- The infrared array data collected by a thermal sensor has background noise.
- This has a great impact on the detection and estimation accuracy.
- An Exponentially Weighted Moving Average (EWMA) filter is a simple and effective technique for smoothing data and reducing noise

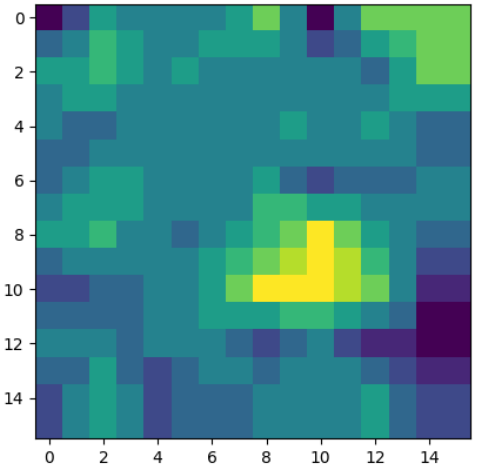
$$EWMA(t) = \sigma F(t) + (1 - \sigma) EWMA(t - 1)$$

$$\sigma = \frac{2}{N + 1}$$

- Possible implementations:
  - MUL EWMA
  - SHIFT EWMA



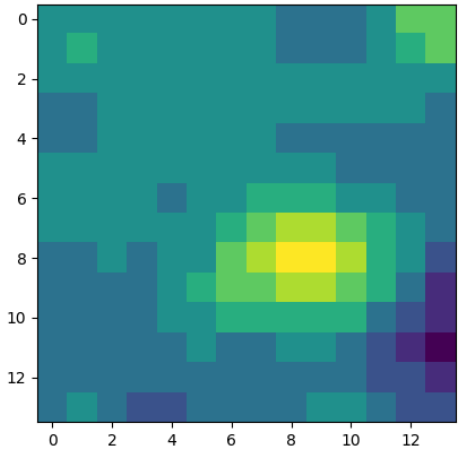
Original frame



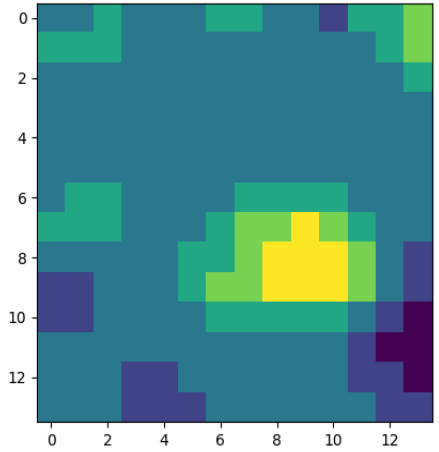
Frame data after noise reduction

Brick2: spatial filter

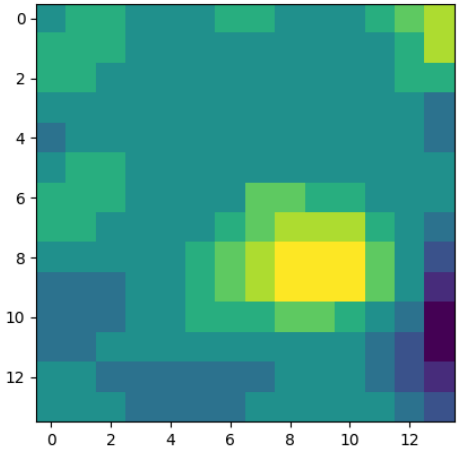
- Random noise in infrared array data can significantly affect the quality of the thermal images.
- Potential sources:
  - sensor electronics
  - background radiation
  - environmental factors
- Possible implementations:
  - Average filter
  - Median filter
  - Gaussian filter



Average filter



Median filter

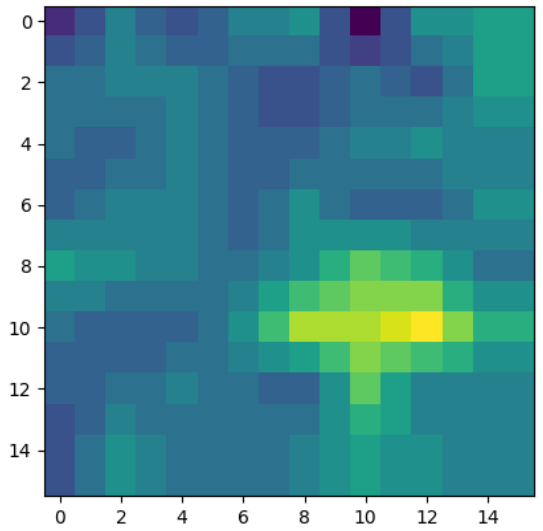


Gaussian filter

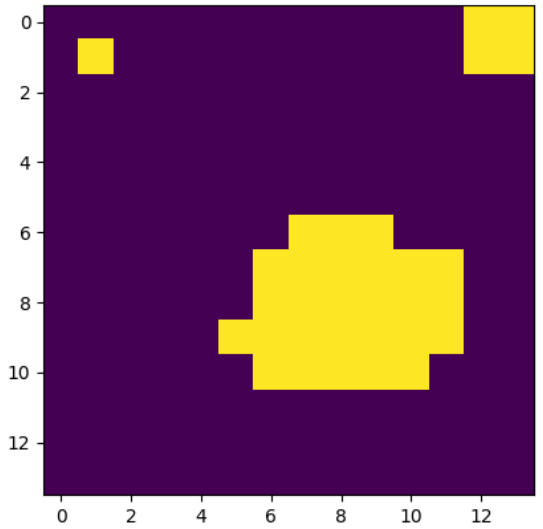
Brick3: threshold algorithm

- This algorithm separates the foreground region from the background region

- Possible implementations:
  - Fixed threshold algorithm
  - Median threshold algorithm

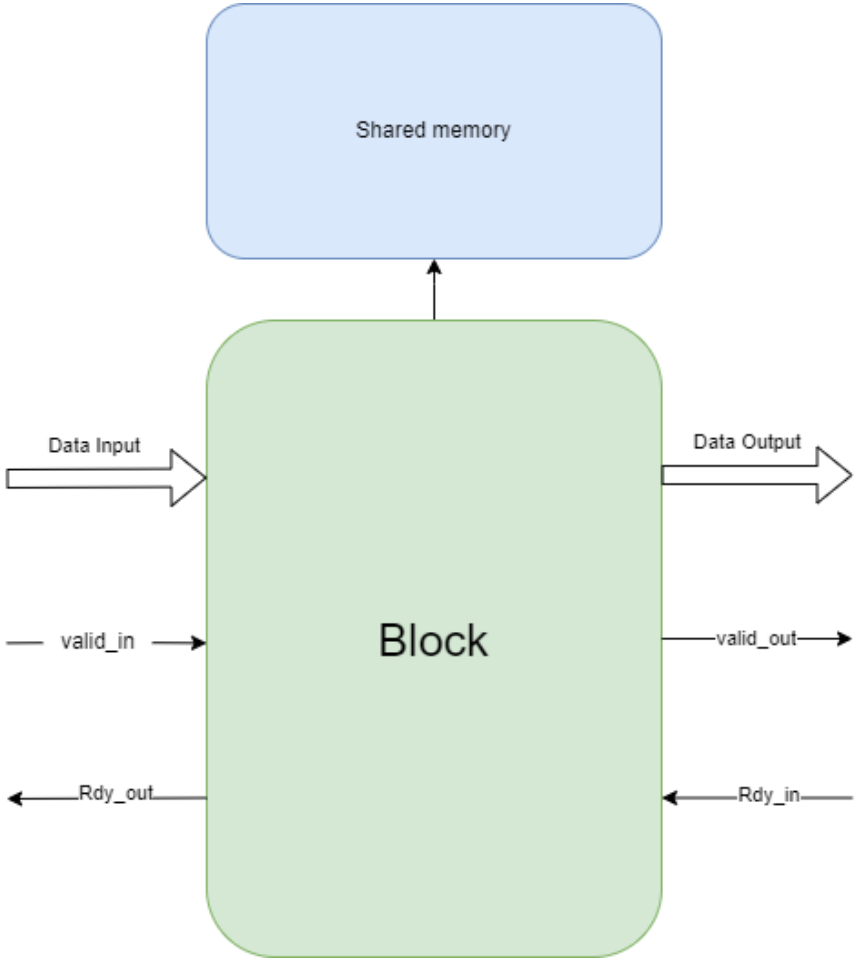


Original frame



Frame data after DSP chain

Protocol for bricks connections

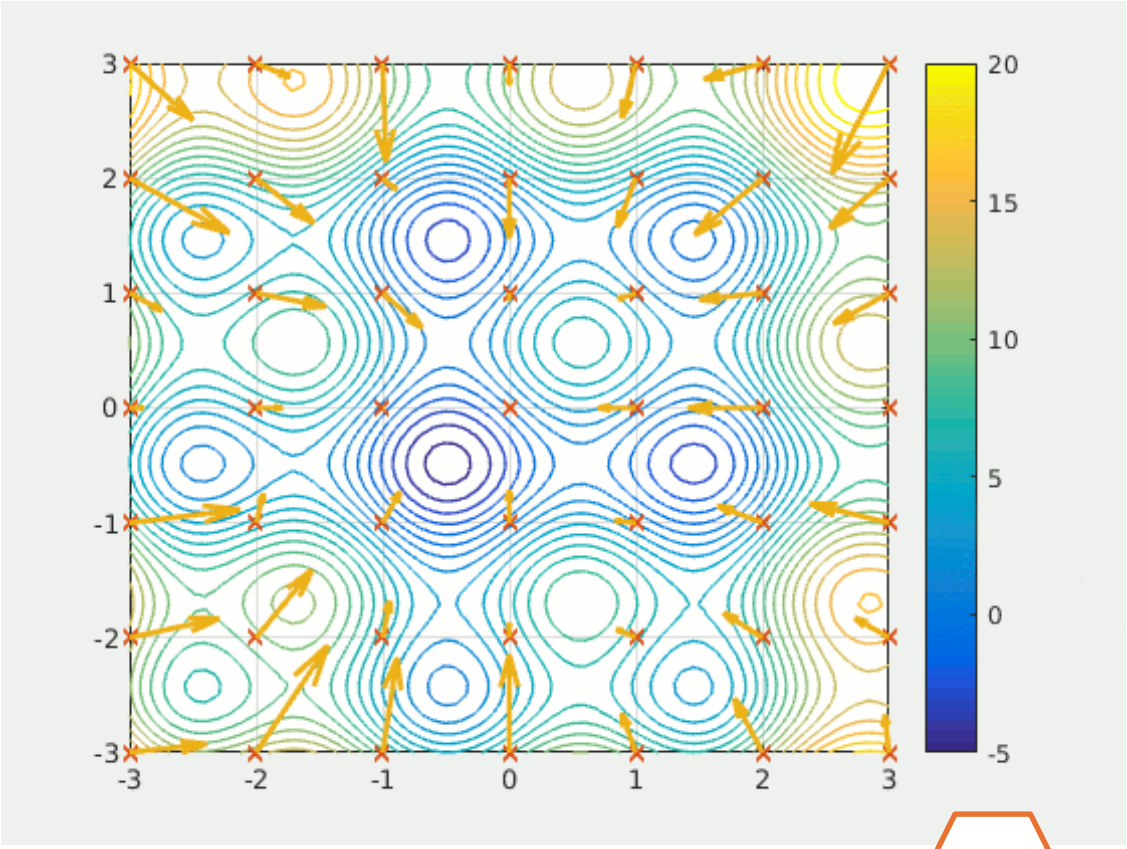


Signals for brick communication:

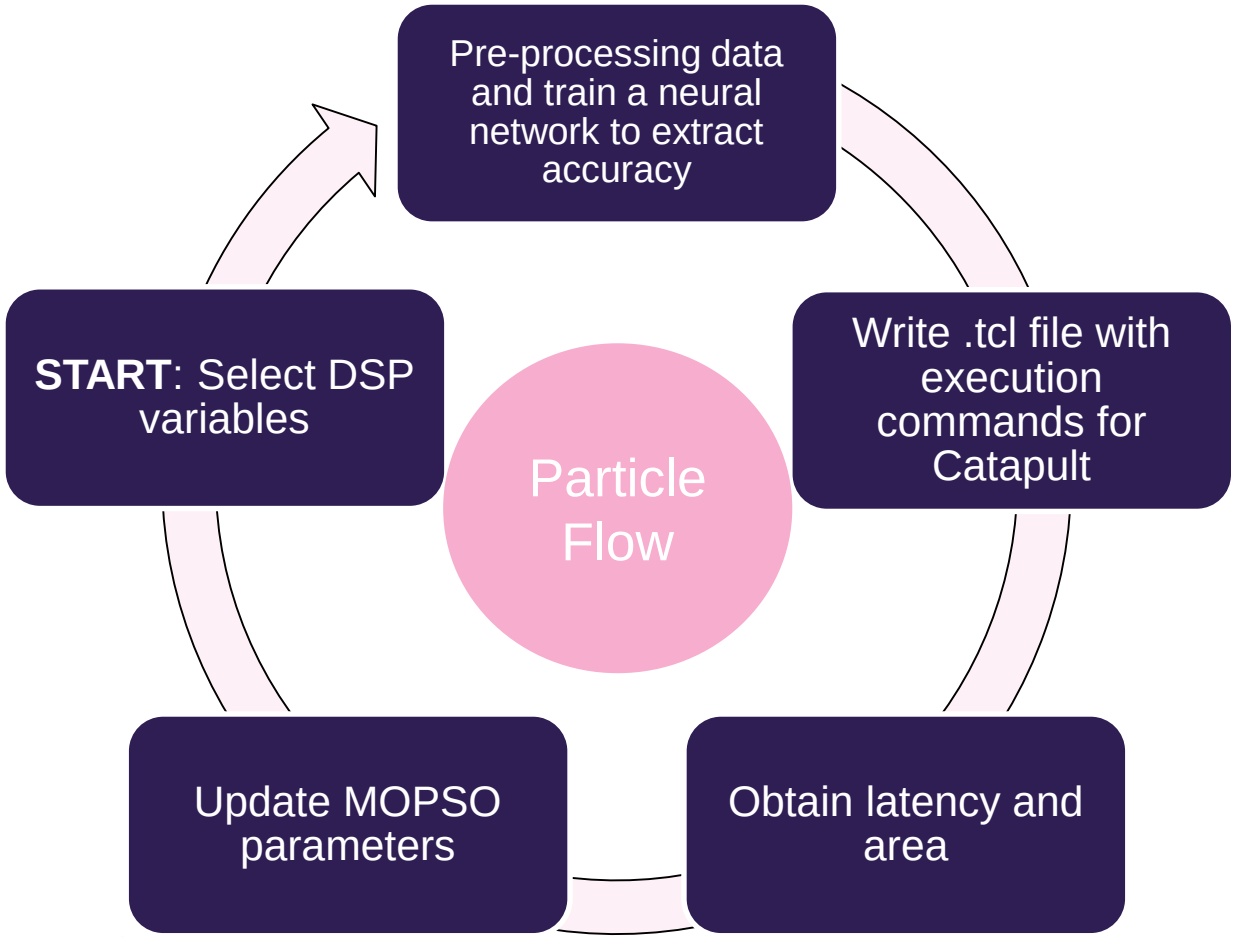
- **Valid:** Driven by the source when data output is ready
- **Ready:** Active high, driven by the target when the data is correctly received
- **Data_input/Data_output:** can be stored in shared memory or directly connected to the next block

Multi-objective Particle Swarm Optimization (MOPSO)

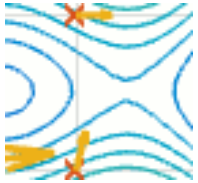
- **Particle Swarm Optimization (PSO)**: Optimizes a problem by iteratively improving a candidate solution.
  - Solves a problem by having a population of candidate solutions, so called particles
  - Each particle's movement is influenced by:
    - Its local best-known position
    - Guided toward the best-known positions in the search-space
- **Multi-objective Particle Swarm optimization:** Extension of the standard PSO to handle optimization problems involving multiple objectives



Optimization flow



- In a MOPSO algorithm we can select:
  - Particles number(number of x)
  - Iterations number(number of ->)



- These particles are moved around in the search-space

Variables	Description
Spatial filter	Type of spatial filter[average, gaussian or median filter]
Threshold algorithm	Type of threshold algorithm [Fixed or median threshold]
K	Additive constant threshold algorithm
N	Widow size in EWMA filter

- **Goal**: find the best trade-off among
  - Accuracy
  - Area
  - Latency

Implementation results

- Clock frequency: 16 MHz
- Technology: CMOS 130nm

Results			Parameters			
Accuracy(%)	Latency cycles	Area (kum2)	K(additive constant threshold)	N(windows average)	Spatial filter	Thr algorithm
90	415	56	3	4 (IIR SHIFT)	Gaussian	Simple
89	412	47	2	17 (IIR MUL)	Median	Median
86	355	36	3	16 (IIR SHIFT)	Median	Simple
87	564	35	2	18 (IIR MUL)	Average	Simple
92	621	45	2	20 (IIR MUL)	Average	Median
93	416	51	2	20 (IIR MUL)	Gaussian	Median

Latency

Area

Accuracy

Conclusions

- A new approach **merging the Python and Catapult** worlds
- **Goal**: to implement the best embedded algorithms on sensors
- **The time** required to design complex algorithms is **hugely reduced**
- A test case has shown a very good result in fast prototyping

For these reasons, this methodology is widely used in the development of our algorithms, allowing us to be fast and competitive in the market.

Thank You

SPONSORED BY