

# Software Reconfigurability - Algorithm Level Approach

Reza Hoshyar, S. Gultchev, K.Seo, R. Tafazolli

CCSR, University of Surrey, UK

- Operation of Different radio access systems of cellular, satellite, broadband short/long range wireless access, and broadcast systems
- Reconfigurable networks and multi-mode terminals to enable working with different radio systems
- Software Defined Radio systems should provide Multi-mode capability

## Multi-mode reconfigurability →

new flexibility requirements to be considered in the design of the architecture, as well as the system algorithms in PHY and upper layers

## HW/SW design and implementation →

to comply with different multi-mode requirements, + to achieve fine trade offs based on the state of the art technologies.

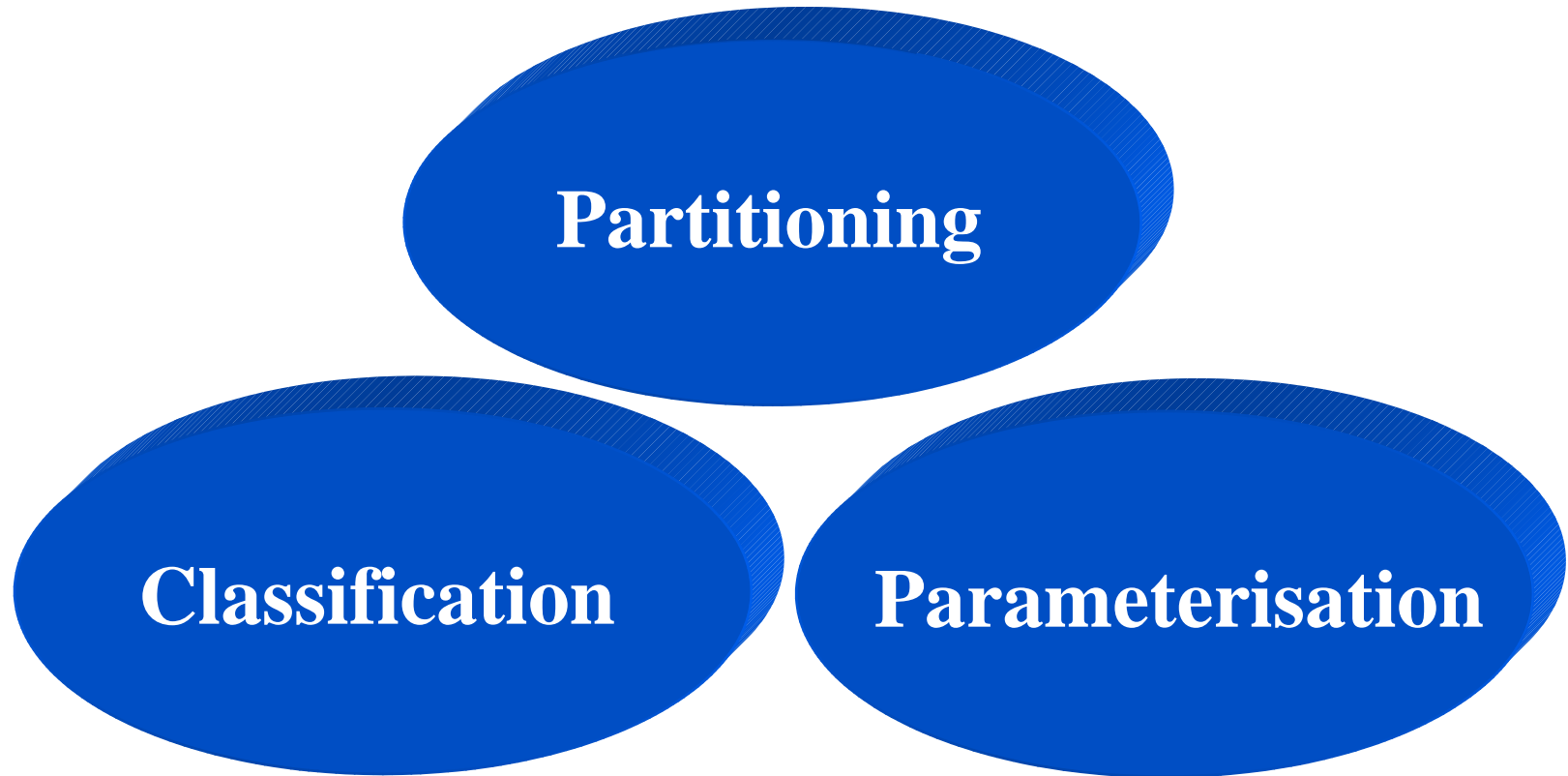
- We need algorithms reconfigurable to different modes with efficient complexity/performance trade off
- Commonalities and similarities among algorithms should be explored, and based on them appropriate HW/SW implementation solutions be provided

- Algorithm level reconfigurability analysis is necessary to:
  - provide structured implementation-independent information that can be easily used in HW/SW implementation and final design decision
- A classic method will be presented for:
  - provision and optimisation of reconfigurable algorithms

# UniS Three-Phase Approach



Mobile Communications Research Group



Centre for Communication Systems Research

25 Feb, 2004

Software Reconfigurability - Algorithm Level Approach

R. Hoshyar

6

- **Partitioning:**  
to find common parts among algorithms, and reduce some of the complexity factors by sharing the common parts
- **The optimum level of partitioning:**  
trade off between the saved complexity due to the common parts, and the overhead on data and program flow control, and reconfigurability control
- **Partitioning → Algorithm Level Primitives (ALP)**

- Classification:

A Technique to exploit similarity

Put **similar** ALPs to the same class

All ALPs belonging to the same class can be represented and implemented by a **general parameterisable ALP**

- Parameterisation:

By parameterisation a general ALP component can be shared amongst the algorithms

“classification”  $\Leftrightarrow$  “parameterisation” :

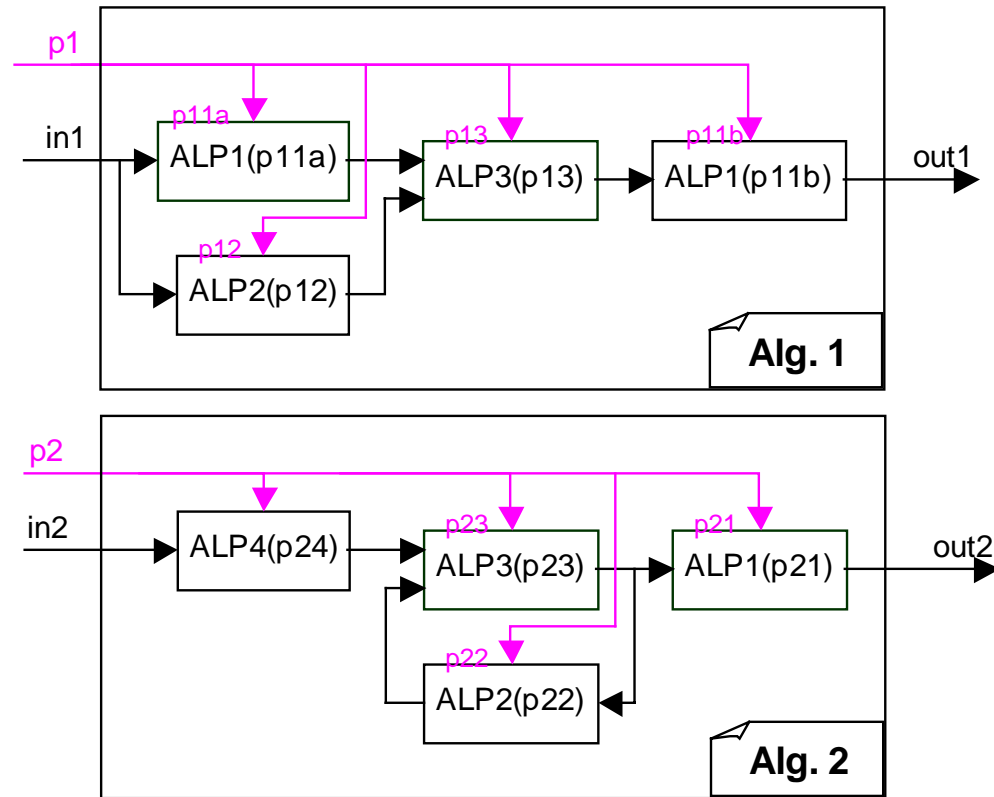
classes of ALPs should be set in a way that an appropriate parameterisation could be developed for each class

- Parameterisation may affect:
  - Functionality
  - Performance
  - Complexity Requirement

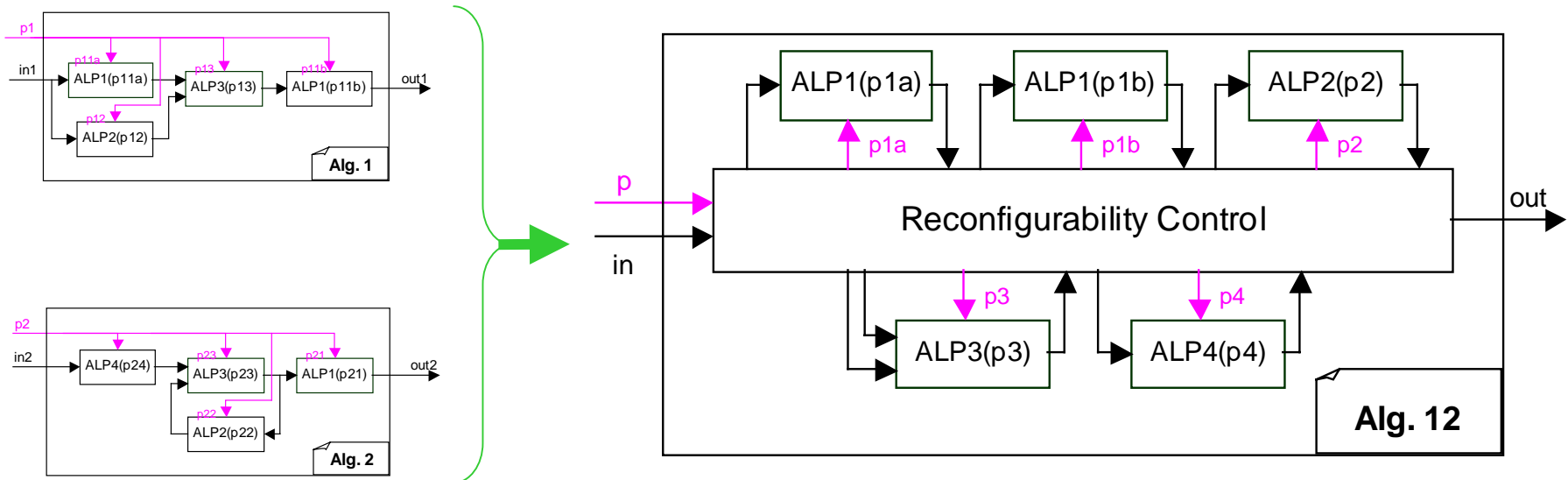
# UniS Three Phase Approach: Example

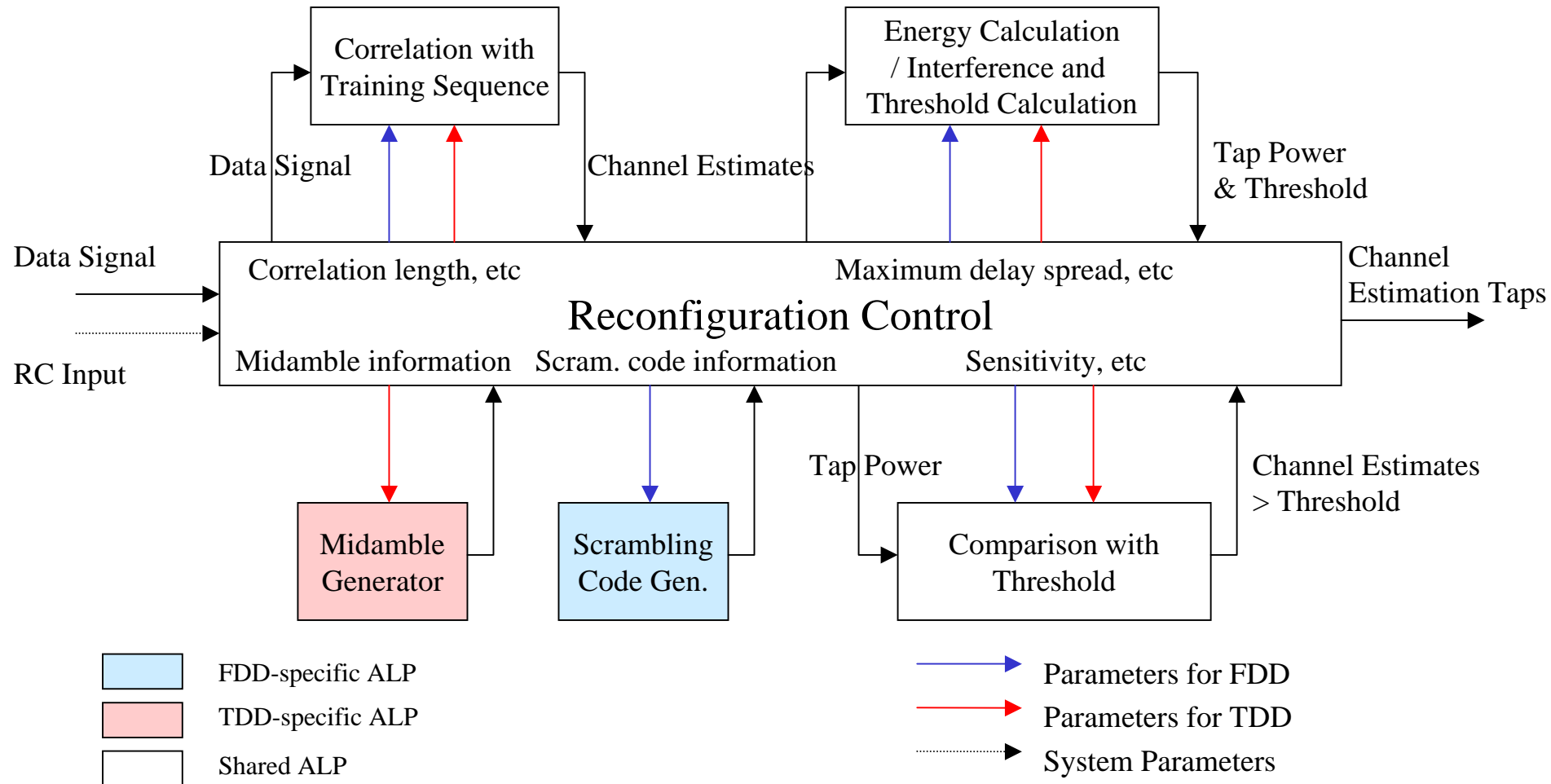


Application of the three phases to Alg.1 and Alg.2 algorithms:

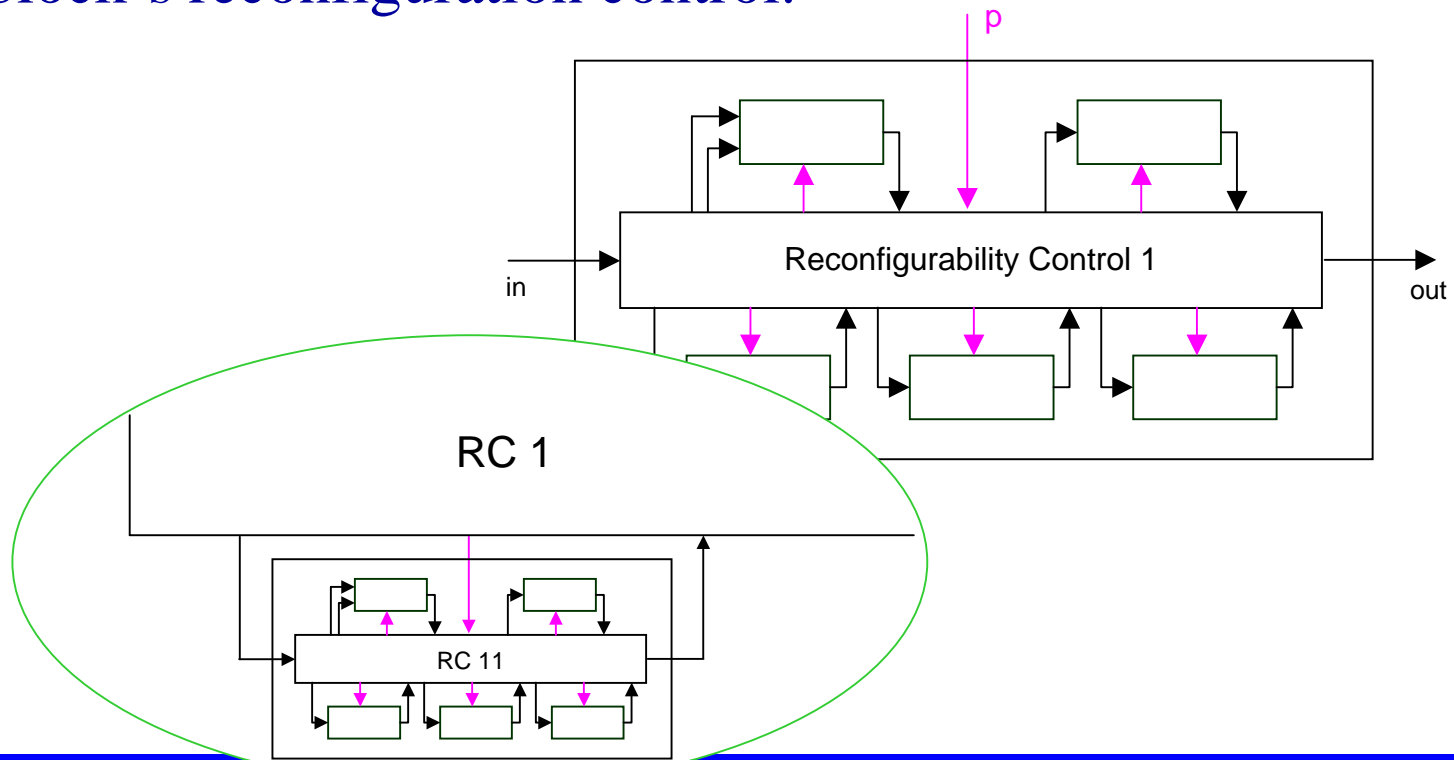


By Introduction of a Reconfigurability Control the two Alg.1 and Alg.2 algorithms can be realized by one multi-mode Reconfigurable algorithm:

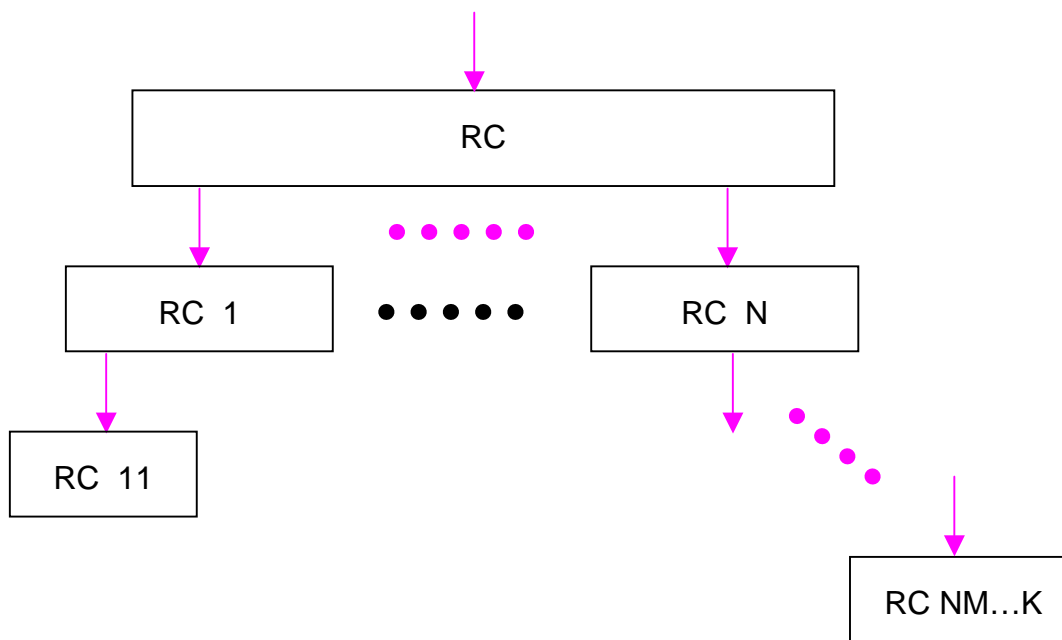




Multi-mode blocks can be controlled by a reconfiguration control, and ALPs of each multi-mode block are controlled by the block's reconfiguration control.



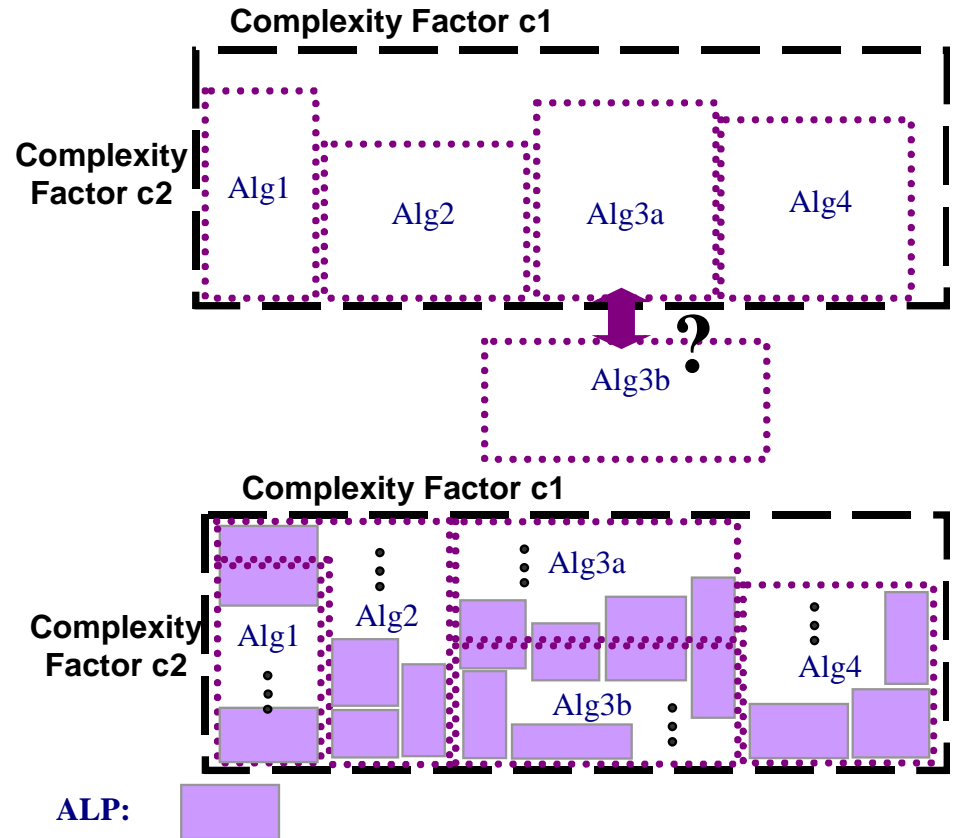
Based on global and local reconfiguration control requirements appropriate levels of hierarchy should be set up to minimize reconfiguration control overhead.



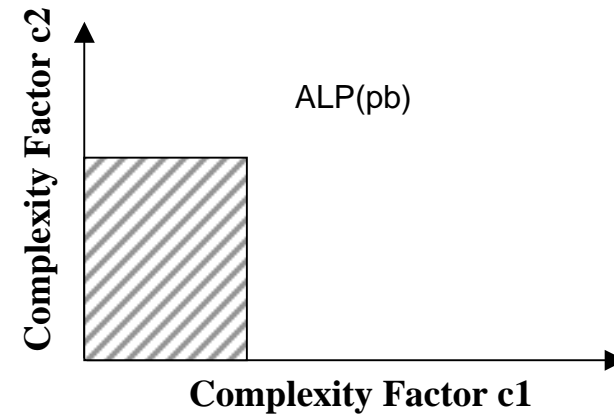
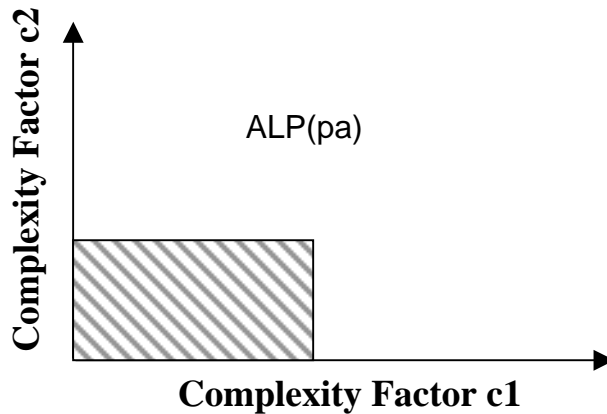
- Multi-Dimensional Complexity Constraint Space

Granularity

provided by ALPs →



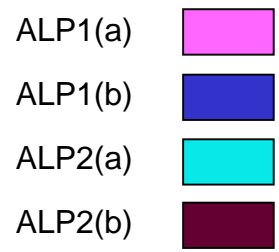
- Parameterisation of an ALP may change its complexity requirement:



**Granularity** provided by ALPs and their **parameterisation** will provide **flexibility** in fitting algorithms into complexity constraint space.

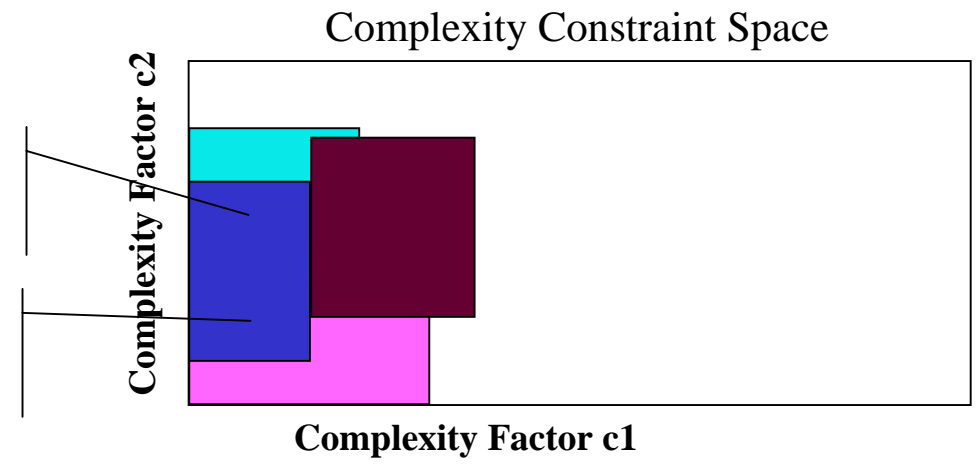
⇒ Efficient **packing** and **desirable trade-offs** between different complexity factors

- Depending on reconfiguration implementation, HW/SW resources (complexity factors) may be shared:
  1. Different Modes of the same ALP
  2. Different Modes of different ALPs



Sharing of  
(ALP2(a) , ALP1(b))

Sharing of  
(ALP1(a) , ALP1(b))



- Algorithms partitioning is not always easy and straightforward.
- How to adjust appropriate partitioning level? It should be based on trade-off between complexity saving and reconfiguration control overhead, but this trade-off is not easily measurable.
- How to measure reconfiguration gain?

- Reconfigurability analysis based on three phases of “Partitioning”, “Classification”, and “Parameterisation”
- provides HW/SW design phase with a structured implementation-independent view on how to achieve multi-mode flexibility requirements within implementation constraints.

- A **Hierarchical** structure can provide efficient reconfiguration control.
- The **granularity** and **parameterisability** achieved by the proposed method, creates HW/SW design flexibility in making desired **trade-off** between **different complexity factors**, and fitting the required features and functionalities into the multi-dimensional complexity constraint space.