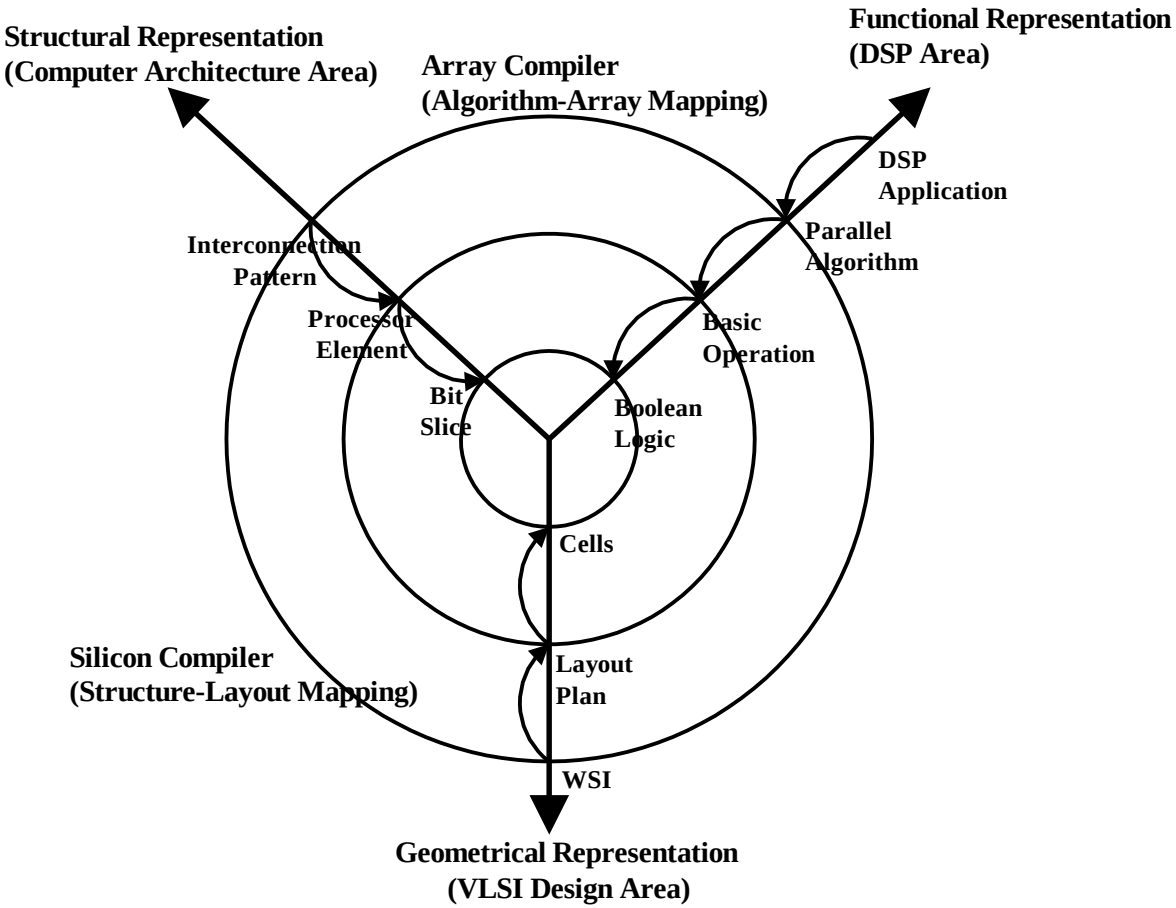
ΕΝΣΩΜΑΤΩΜΈΝΑ ΣΥΣΤΉΜΑΤΑ

ΑΛΓΟΡΙΘΜΙΚΟΊ ΜΕΤΑΣΧΗΜΑΤΙΣΜΟΊ

Μετασχηματισμοί πηγαίου κώδικα

Y-CHART


Structural Representation
(Computer Architecture Area)
Array Compiler
(Algorithm-Array Mapping)
Functional Representation
(DSP Area)
DSP
Application
Parallel
Algorithm
Interconnection
Pattern
Processor
Element
Basic
Operation
Bit
Slice
Boolean
Logic
Cells
Silicon Compiler
(Structure-Layout Mapping)
Layout
Plan
WSI
Geometrical Representation
(VLSI Design Area)

Τύποι Εξαρτήσεων Δεδομένων
Types of Data Dependences

- **Εξάρτηση Ροής (Flow dependence)**
  - **S1:** a = c*10
  - **S2:** d = 2*a + c
- **Αντιεξάρτηση (Anti-dependence)**
  - **S1:** e = f*4 + g
  - **S2:** g = 2*h

Ανάλυση Εξαρτήσεων Βρόχου

```
do i1 = l1, u1
   do i2 = l2, u2
      ...
         do id = ld, ud
1           a[f1(i1, ..., id), ..., fm(i1, ..., id)] = ...
2           ...= a[g1(i1, ..., id), ..., gm(i1, ..., id)]
         end do
      ...
   end do
end do
```

```
do i = 2, n
1    a[i] = a[i] + c
2    b[i] = a[i-1] * b[i]
end do
```

Για τον υπολογισμό της πληροφορίας εξάρτησης σε φωλιασμένους για βρόχους, το βασικό πρόβλημα είναι η κατανόηση της χρήσης των πινάκων. Τα βαθμωτά δεδομένα είναι σχετικά εύκολο να διαχειριστούν. Για να παρακολουθήσετε τη συμπεριφορά του πίνακα, ο μεταγλωττιστής πρέπει να αναλύσει το δείκτη εκφράσεις σε κάθε αναφορά σε πίνακα.

Αλγοριθμικοί Μετασχηματισμοί

- **Μετασχηματισμοί με βάση τη ροή δεδομένων στο βρόχο**
- **Αναδιάταξη Βρόχων**
- **Αναδιάρθρωση Βρόχων**
- **Μετασχηματισμοί Αντικατάστασης Βρόχου**
- **Μετασχηματισμοί πρόσβασης στη μνήμη**
- **Μερική αξιολόγηση**
- **Εξάλειψη του πλεονασμού**
- **Μετασχηματισμοί για κλήσης συναρτήσεων/υποπρογραμμάτων**

Μετασχηματισμοί Αναδιάταξης Βρόχων
Loop Reordering Transformations

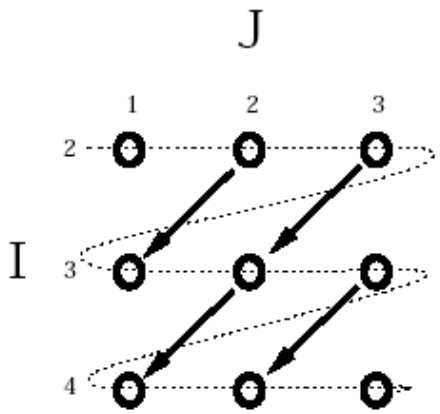
- Αλλαγή της σχετικής ακολουθίας εκτέλεσης των επαναλήψεων του/των φωλιασμένου/νων βρόχου/ων
  - Ανάδειξη παραλληλίας και βελτίωση της τοπικότητας στην μνήμη.

Μετασχηματισμοί Αναδιάταξης Βρόχων (1)

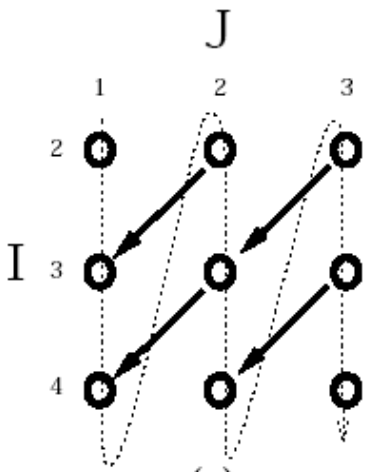
- **Εναλλαγή βρόχου (Loop Interchange)**

```
do i = 2, n
  do j = 1, n-1
    a[i,j] = a[i-1,j+1]
  end do
end do
```

(a)



(b)



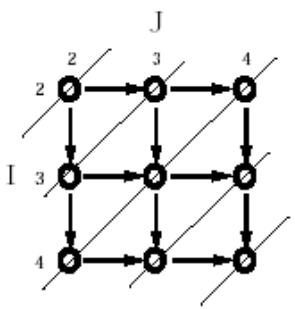
(c)

Μετασχηματισμοί Αναδιάταξης Βρόχων (2)

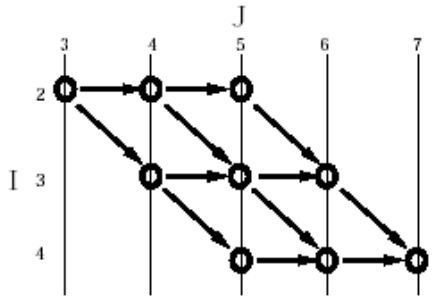
- Στρέβλωση Βρόχου (Loop Skewing) ➔ Στρέβλωση της εκτέλεσης των επαναλήψεων (skew iterations execution)
- Useful for Loop Interchange

```
do i = 2, n-1
  do j = 2, m-1
    a[i,j] = (a[i-1,j] + a[i,j-1] +
              a[i+1,j] + a[i,j+1])/4
  end do
end do
```

(a) original code: dependences $\{(1,0),(0,1)\}$



(b) original space



(c) skewed space

Parallel iterations

Skewing factor "i"

```
do i = 2, n-1
  do j = i+2, i+m-1
    a[i,j-i] = (a[i-1,j-i] + a[i,j-1-i] +
                a[i+1,j-i] + a[i,j+1-i])/4
  end do
end do
```

(d) skewed code: dependences $\{(1,1),(0,1)\}$

```
do j = 4, m+n-2
  do i = max(2,j-m+1), min(n-1,j-2)
    a[i,j-i] = (a[i-1,j-i] + a[i,j-1-i] +
                a[i+1,j-i] + a[i,j+1-i])/4
  end do
end do
```

(e) skewed and interchanged code: dependences $\{(1,0),(1,1)\}$

Partial Evaluation

- Partial evaluation refers to the general technique of performing part of a computation at compile time.
- Constant propagation is one of the most important optimizations that a compiler can perform and a good optimizing compiler will apply it aggressively.
- Programs typically contain many constants; by propagating them through the program, the compiler can do a significant amount of pre-computation.
- The propagation reveals many opportunities for other optimizations.

```
n = 64
c = 3
do i = 1, n
  a[i] = a[i] + c
end do
```

(a) original code

```
do i = 1, 64
  a[i] = a[i] + 3
end do
```

(b) after constant propagation

- Constant folding is a companion to constant propagation: when an expression contains an operation with constant values as operands, the compiler can replace the expression with the result.

Partial Evaluation (2)

- Strength Reduction: replace an expensive operator with an equivalent less expensive operator.

Expression	Reduced Expr.	Datatypes
$x \times 2$	$x + x$	integer, real
x^2	$x \times x$	integer, real
$x^{c.5}$	$x^c \times \sqrt{x}$	real
$i \times 2^c$	$i \ll c$	integer
$(a, 0) + (b, 0)$	$(a + b, 0)$	complex
$\text{len}(s_1 \ \& \ s_2)$	$\text{len}(s_1) + \text{len}(s_2)$	string

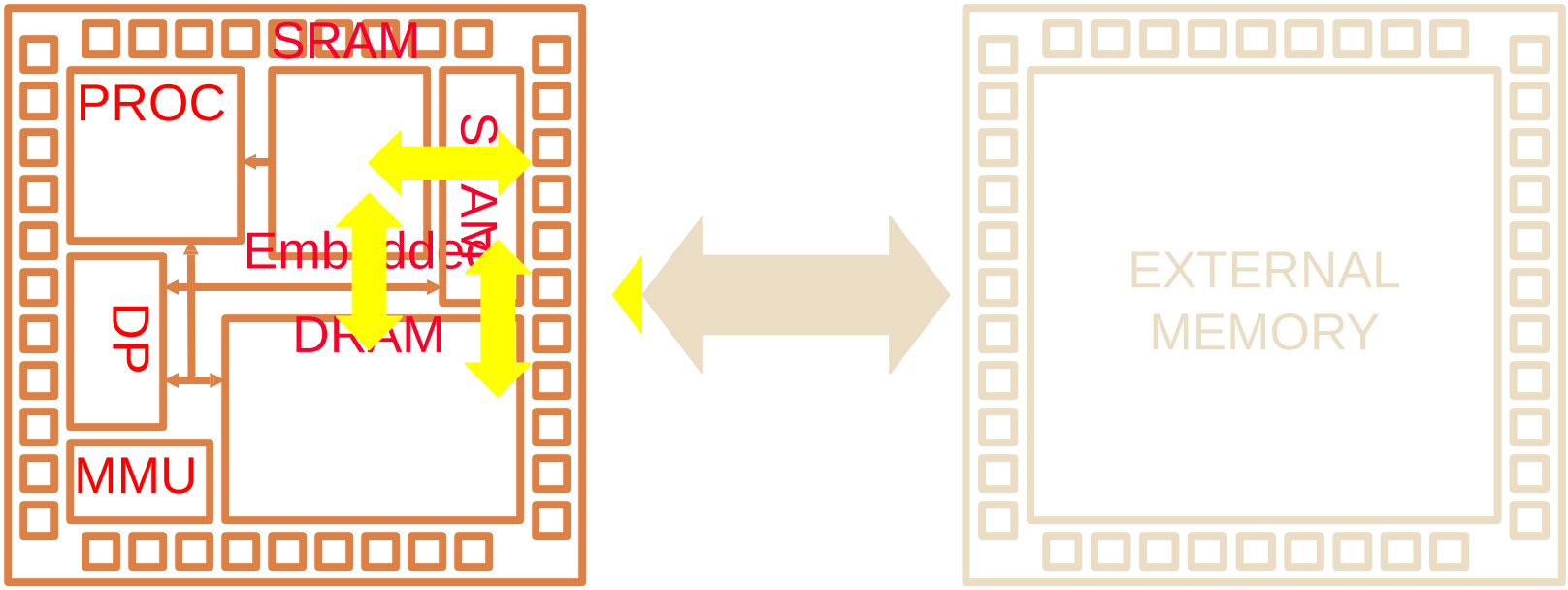
ΕΝΣΩΜΑΤΩΜΈΝΑ ΣΥΣΤΉΜΑΤΑ

Στατική Διαχείριση Μνήμης:
Αρχές και Εφαρμογές

Ορισμός Προβλήματος

- Οι εφαρμογές πολυμέσων (multimedia) (π.χ., mpeg-4, jpeg, h.264) χαρακτηρίζονται από μεγάλο αριθμό προσπελάσεων στην μνήμη
- Οι προσπελάσεις στην μνήμη είναι το μεγαλύτερο ποσοστό της συνολικής κατανάλωσης ισχύος (με άμεση επίδραση στην ταχύτητα εκτέλεσης)
- Για την λύση του προβλήματος αυτού απαιτείται η ανάπτυξη συστηματικής μεθοδολογίας

Embedded Systems



P(Ext. Access) = typ. 30 x P(Arithmetic Operations)

P(Int. Memory) = typ. 40 % - 60 % P(Chip)

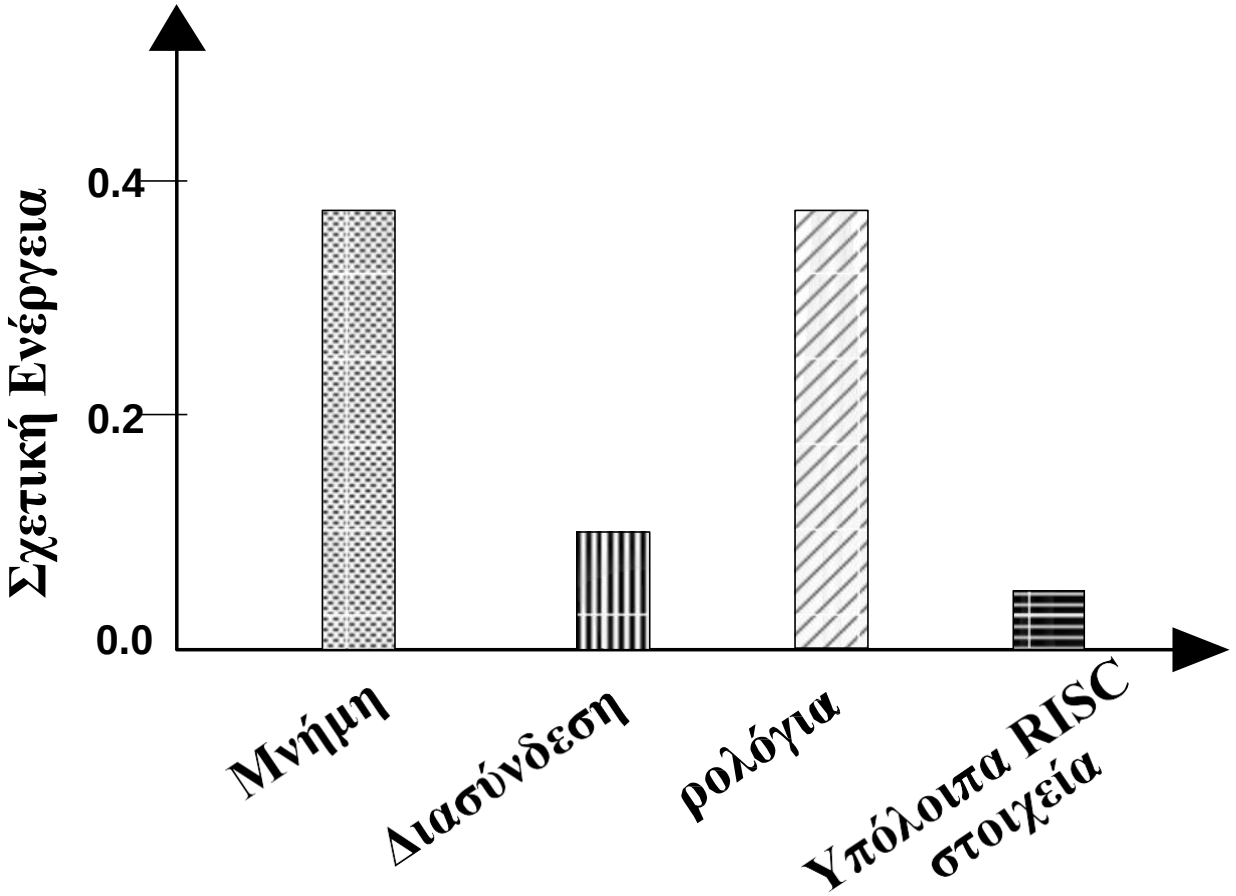
Κατανάλωση ενέργειας στην μνήμη για ενσωματωμένα συστήματα

- Ενα τυπικό ενσωματωμένο σύστημα πολυμέσων περιλαμβάνει δύο τύπους μνημών:
  - Μνήμη Δεδομένων (Data Memory)
  - Μνήμη Εντολών (Instruction Memory)

Στόχος: Ελαχιστοποίηση της συνολικής κατανάλωσης ισχύος, P_{total}, και των δύο συνιστωσών, P_{data} και $P_{instruction}$:

- $P_{total} = P_{data} + P_{instruction}$

Κατανομή Ενέργειας



Μνήμη= Το εμπόδιο στην απόδοση
Memory = Performance Bottleneck