

DART, une architecture reconfigurable dynamiquement pour applications mobiles



Raphaël David, Sébastien Pillement, Olivier Sentieys
IRISA/ENSSAT

Séminaire ETSN

25/03/04

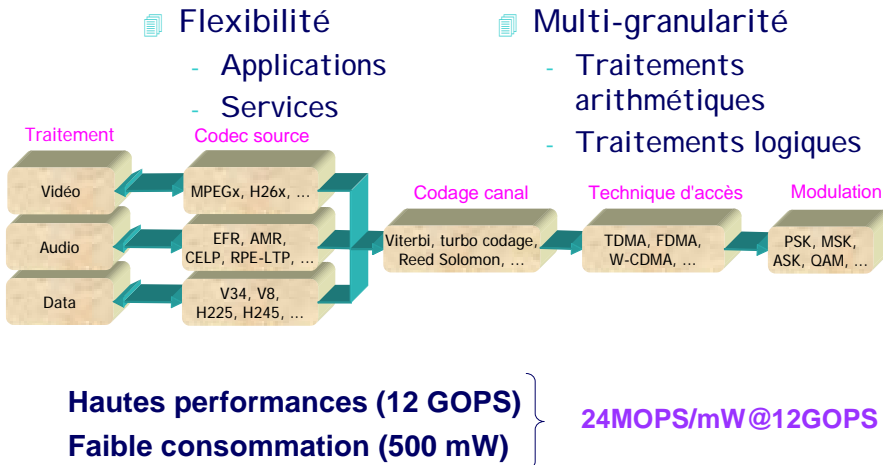
Plan



Introduction

1. Architecture et modèles de programmation de DART
2. Caractérisation et validation de l'architecture
3. Le flot de développement
4. État d'avancement du projet et perspectives

Chaîne de communication de troisième génération (côté terminal mobile)



3

Architectures reconfigurables et télécommunications 3G

Contraintes des télécommunications 3G	Influence sur les architectures	Exemples
Variété de grains de calcul	Architectures multi-grains	- Pleiades - VirtexII
Variété de motifs de calcul	- Interconnexions flexibles - Reconfiguration dynamique	- Chameleon - RaPiD
Variété de tailles de données	Sub Word Processings (SWP)	- Tiger Sharc - FPGA
Traitements concurrents	Partitionnement du circuit, découpe en clusters	- Pleiades - FPFA
Faible consommation	Distribution des ressources	Pleiades
Faible temps de mise sur le Marché	- Outils de développement - Platform based design	- DSP - RaPiD

4

Plan



1. Une architecture enfouie reconfigurable dynamiquement : DART

- ↙ Présentation générale
- ↙ Architecture des clusters
- ↙ Les DPRs
 - Architecture
 - Coût de configuration
- ↙ Modèles de programmation
 - Le SCMD
 - Reconfigurations HW/SW

5

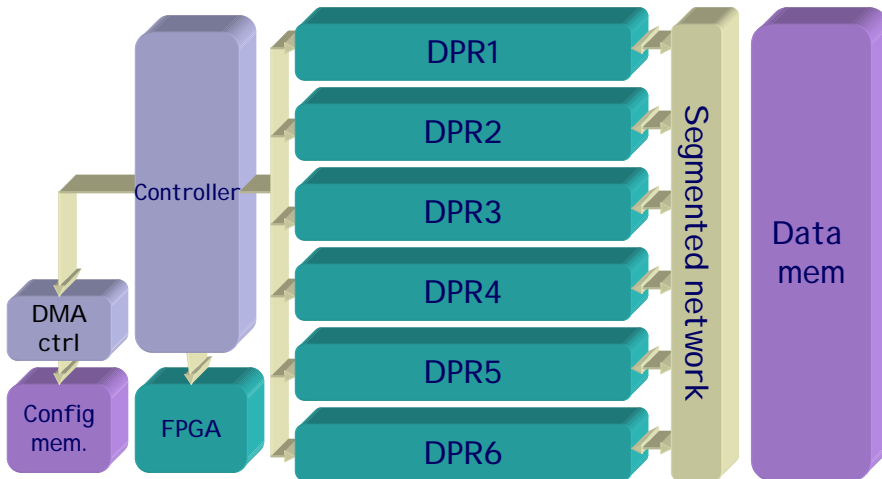
DART : Généralités



- 📄 Architecture autonome
- 📄 Reconfiguration multigrain
 - Fonctionnel (DPR), logique (FPGA)
- 📄 Reconfiguration dynamique
- 📄 Faible consommation
- 📄 Distribution hiérarchique des ressources
 - calcul, stockage, interconnexions, contrôle

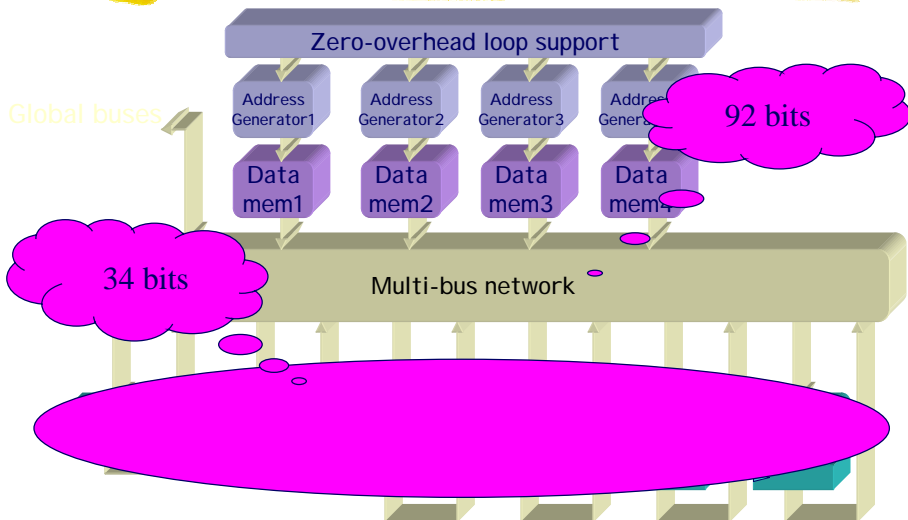
6

Architecture des clusters



7

Architecture des DPRs



8

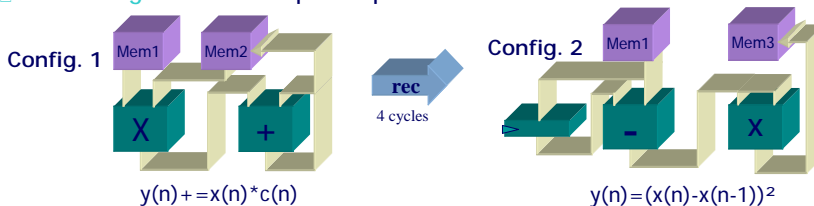
Le SCMD : Single Configuration Multiple Data

- Les traitements irréguliers ont très peu de parallélisme
 - Implémentation sur 1 seul DPR
 - Les traitements massivement parallèles sont très réguliers
 - Redondance des configurations des DPRs
- ☺ Il est possible de réduire le volume de données de configuration en transmettant simultanément les même données vers plusieurs cibles (DPR)

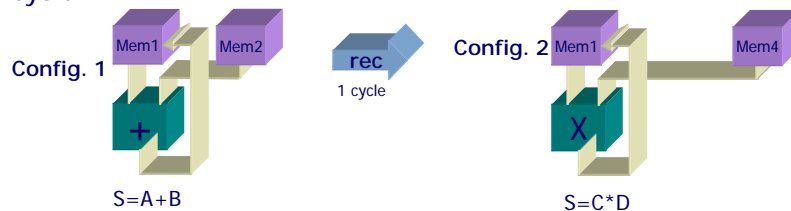
9

Performance vs flexibilité

- Reconfiguration HW pour optimiser le chemin de données :



- Reconfiguration SW pour reconfigurer le DPR à chaque cycle :



10

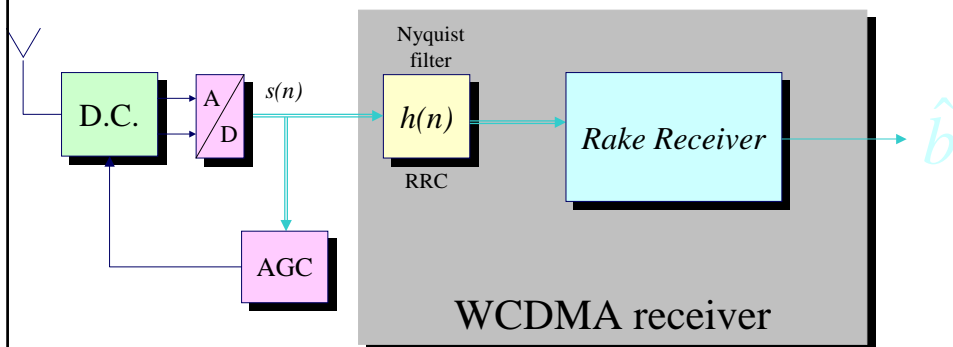
Plan

2. Estimation des performances de DART

- ↳ Résultats d'implémentations
 - Récepteur W-CDMA
 - Filtre de réception
 - Rake receiver
- ↳ Volume de configuration

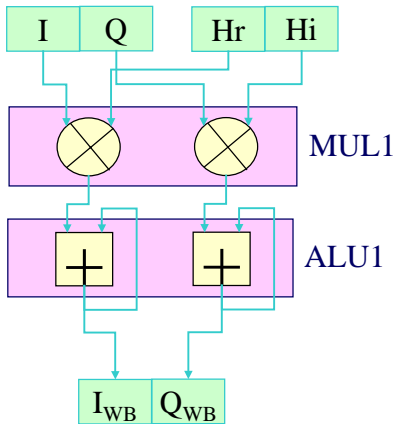
11

Synoptique d'un récepteur W-CDMA



12

Implémentation du filtre de réception



Caractéristique du filtre:

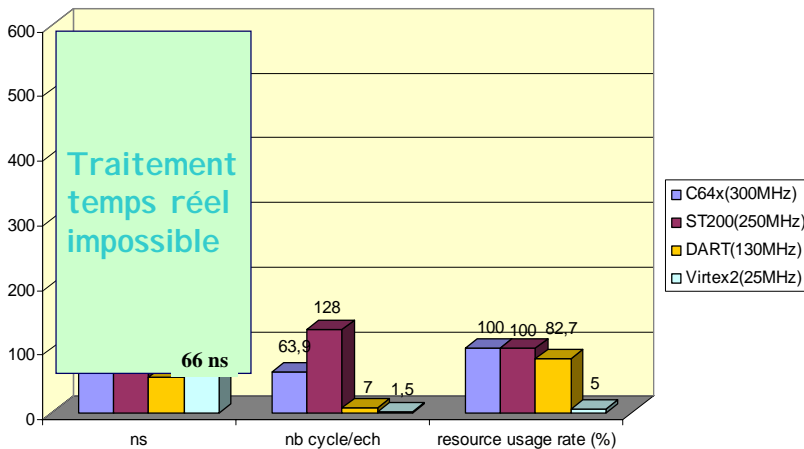
- filtre complexe
- $F_c=15.36\text{MHz}$
- 64 points
- données 8 bits

complexité

- 986 MMACs
- 1.966 GOPS

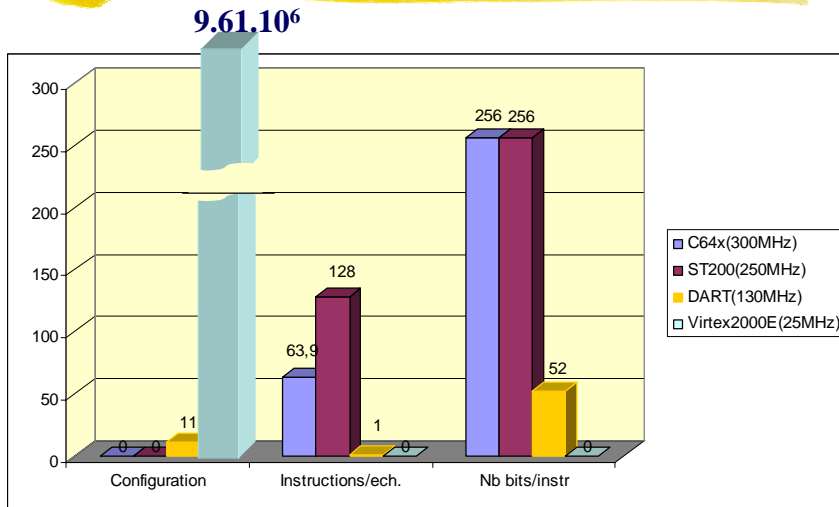
13

Résultats d'implémentations pour le FIR



DART : 32,9MOPS/mW, Virtex E : 1.96MOPS/mW

Volume des configurations pour le FIR



15

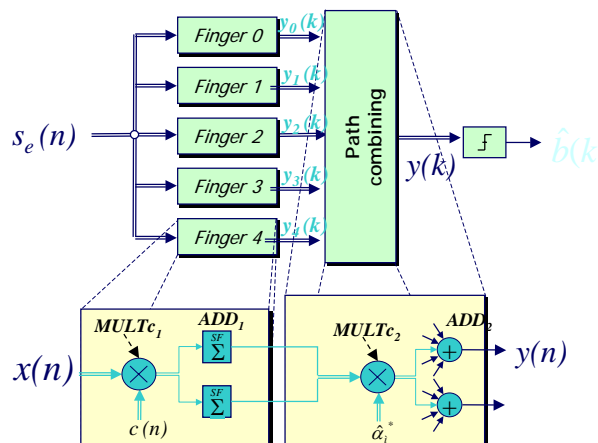
Synoptique d'un Rake receiver

Caractéristique du Rake:

- Complex despreading + path combining
- $F_e = 15.36 \text{ MHz}$
- $SF = 256$
- données 8 bits

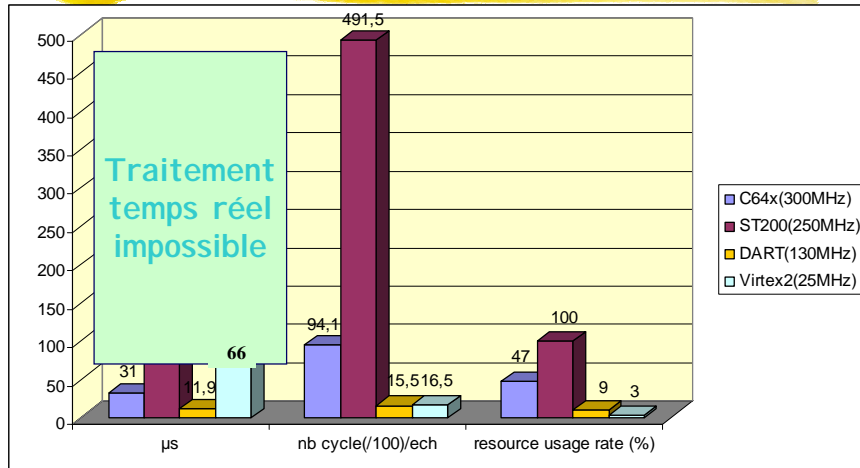
complexité

- 184.7 MOPS



16

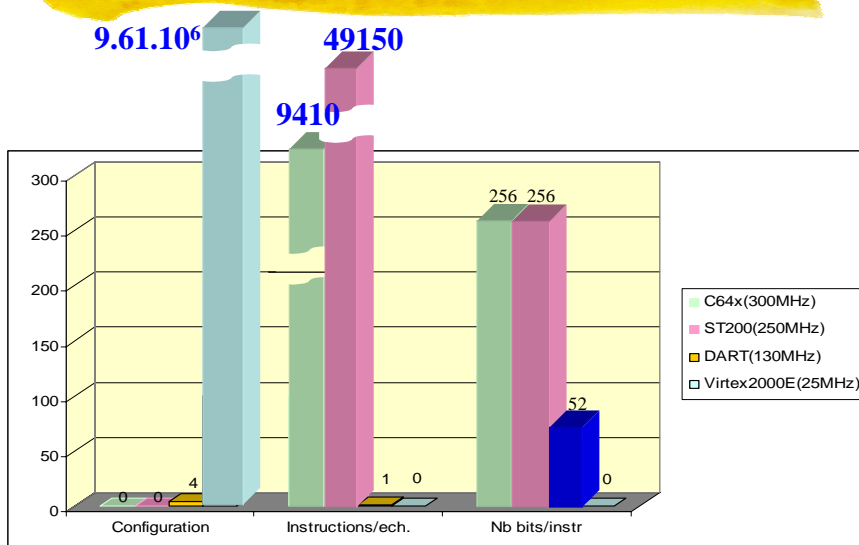
Résultats d'implémentation pour le Rake receiver



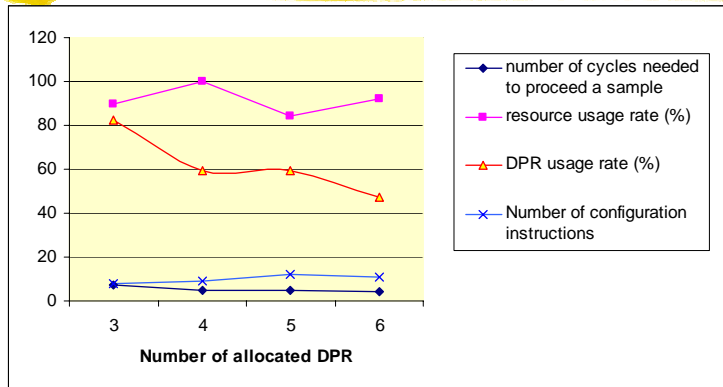
DART : 33,2 MOPS/mW

17

Volume des configurations pour le Rake receiver

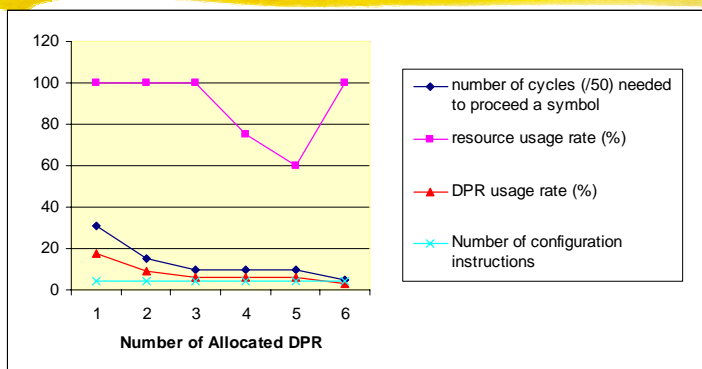


Implémentation du FIR W-CDMA sur un cluster de DART en mode SWP



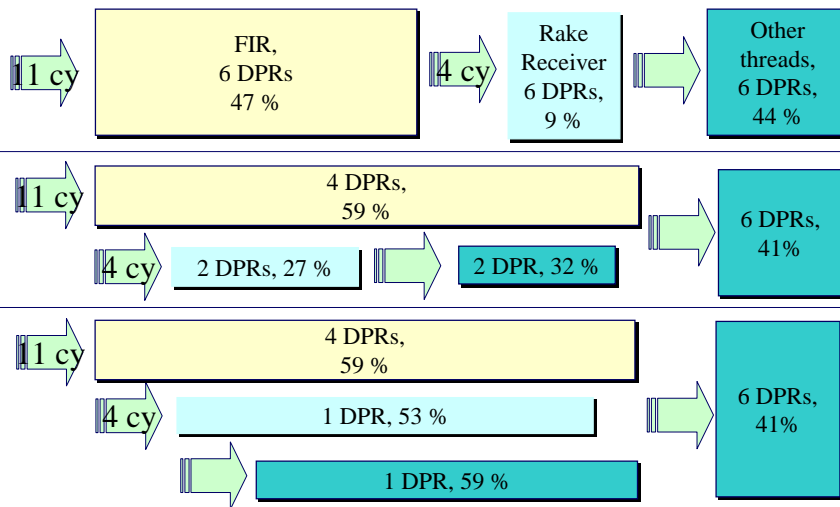
Nb DPR	nb cy/sample	resources usage rate	DPRs usage rate	Nb configuration instructions
3	7	90	82.7	8
4	5	100	59.1	9
5	5	84	59.1	12
6	4	92	47.3	11

Implémentation du rake receiver sur un cluster de DART en mode SWP



Nb DPR	nb cy/symbol (x100)	resources usage rate	DPRs usage rate	Nb configuration instructions
1	15,51	100	17,8	4
2	7,78	100	8,9	4
3	5,21	100	5,9	4
4	5,21	75	5,9	4
5	5,21	60	5,9	4
6	2,63	100	3	4

Exemple de compromis entre parallélisme d'opération et de tâche



21

Plan

3. Développer une application pour DART

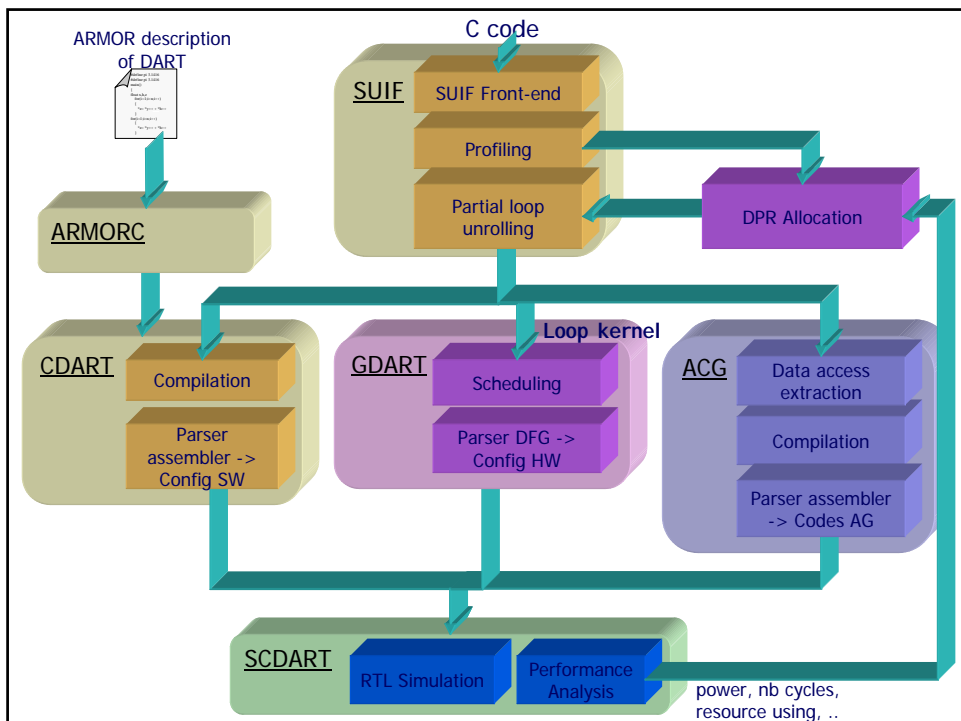
- ✦ Le flot de conception
- ✦ Les différents outils
 - Le front-end
 - cADRT
 - gDART
 - ACG
 - SCDART

22

Flot de développement

- ☐ Trois types de traitements doivent être distingués :
 - Les codes irréguliers
 - Les manipulations de données
 - Les calculs réguliers
- ☐ Les codes irréguliers et les manipulations de données sont traduits en codes binaires exécutables via des passes classiques de compilation issues de CALI FE
 - Génération des instructions SW : *cDART*
 - Génération des instructions de manipulation de données : *ACG*
- ☐ Les traitements réguliers sont transformés en reconfigurations HW via une extension de BSS
 - Génération des configurations HW : *gDART*

23



Plan



4. Conclusions et perspectives

25

Conclusion



DART supporte les principales contraintes des T3G

- ☺ Variété de grains de calcul
- ☺ Variété de motifs de calcul
- ☺ Variété de tailles de données
- ☺ Traitements concurrents
- ☺ Hautes performances
- ☺ Faible consommation
- ☺ Outils de développement de haut niveau

26

Etat d'avancement (1)

Au niveau Architectural

- Conception d'un modèle RTL des DPRs
 - ↳ VHDL
 - ↳ Synthèse logique et optimisation des opérateurs critiques (UFs, AGUs)
 - ↳ Caractérisation en consommation des opérateurs
 - ↳ SystemC
 - ↳ Simulation fonctionnelle
 - ↳ Estimation de la consommation d'énergie
 - ↳ => Le modèle RTL des DPRs est validé

27

Etat d'avancement (2)

Au niveau logiciel

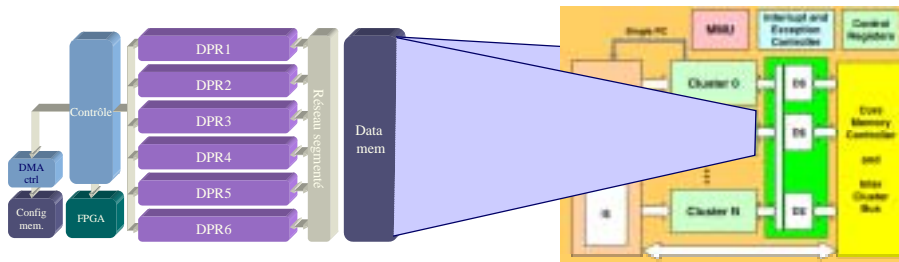
- Définition de certains blocs de base de notre flot
 - ↳ gDART
 - ↳ Front-end SUI F
 - ↳ Modélisation de DART en Armor
 - ↳ Assembleur pour les générations d'adresse

28

Perspectives (1)

Au niveau Architectural

- Définition d'un modèle fonctionnel du FPGA enfoui
- Intégration d'un cluster de DART dans une architecture système



29

Perspectives (2)

- Conception VLSI de DART
- Mise à disposition des outils, du simulateur et du modèle RTL

30