Department of Electronics Engineering
National Chiao Tung University
Hsinchu, Taiwan

A Multicycle Communication Architecture and Synthesis Flow for Global Interconnect Resource Sharing

Wei-Sheng Huang, Yu-Ru Hong,
Juinn-Dar Huang, and Ya-Shih Huang

Advanced
Design
Automation
Research

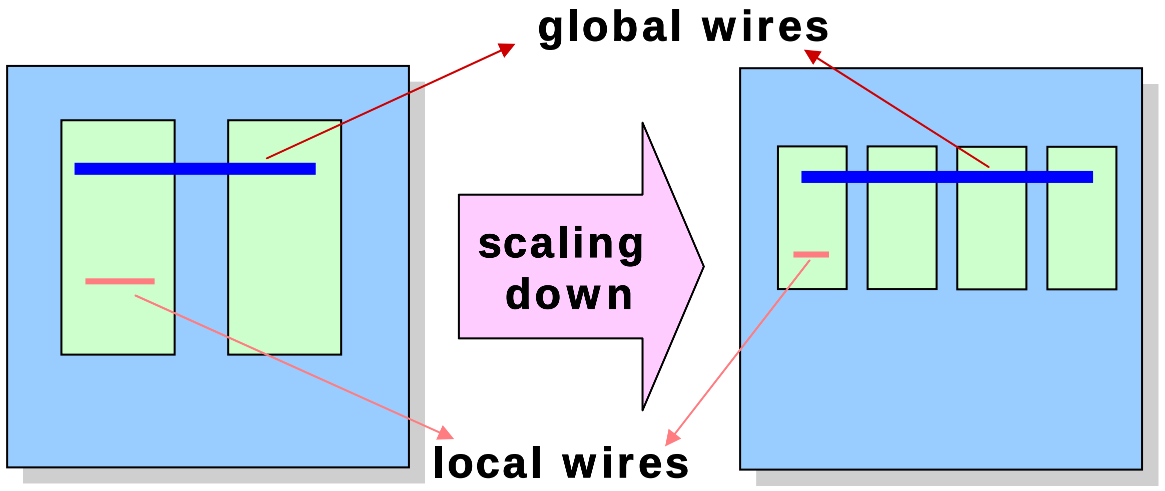
ASP-DAC, Seoul, Korea
Jan. 22, 2008

Outlines

- Introduction
- Motivation
- Problem Formulation
- Experiments
- Conclusions & Future Works

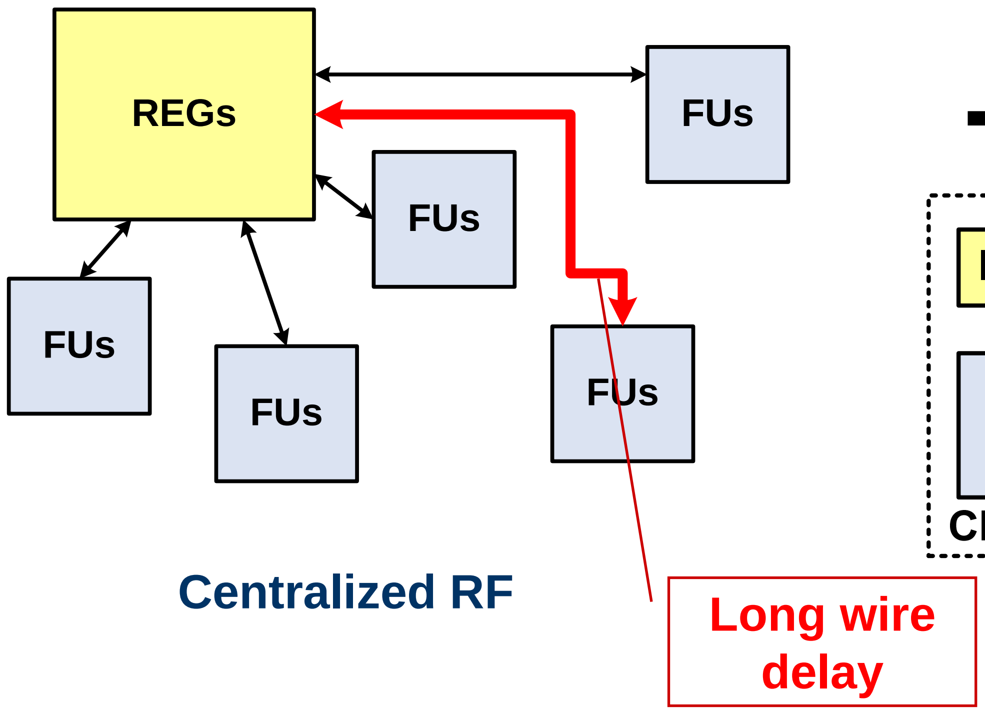
Introduction (1/2)

- In DSM, improvement of wire delay is relatively smaller than improvement of gate delay
  – long wire delay becomes the performance bottleneck

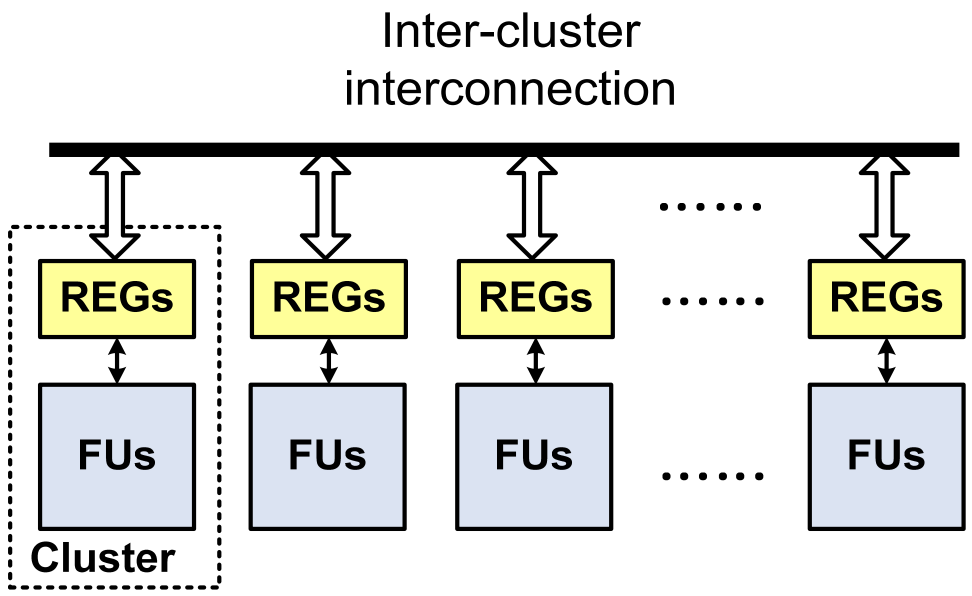


Introduction (2/2)

- Single-cycle communication + centralized RF
  ➔ Multi-cycle communication + distributed RF



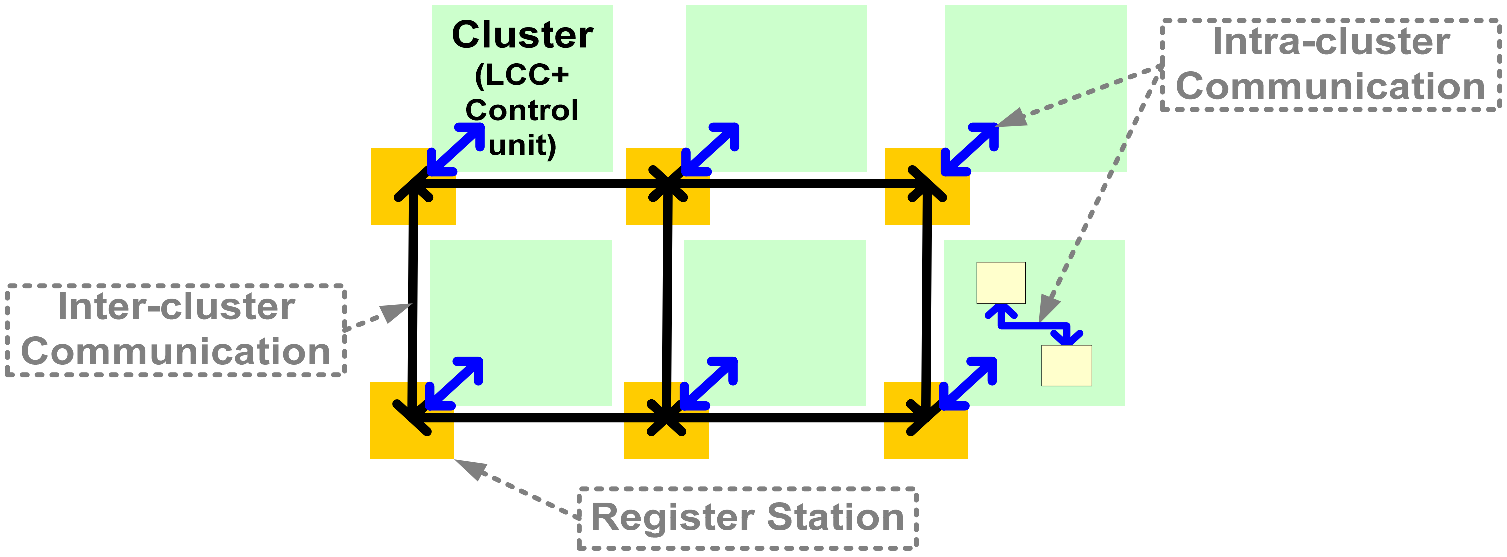
Centralized RF



Distributed RF

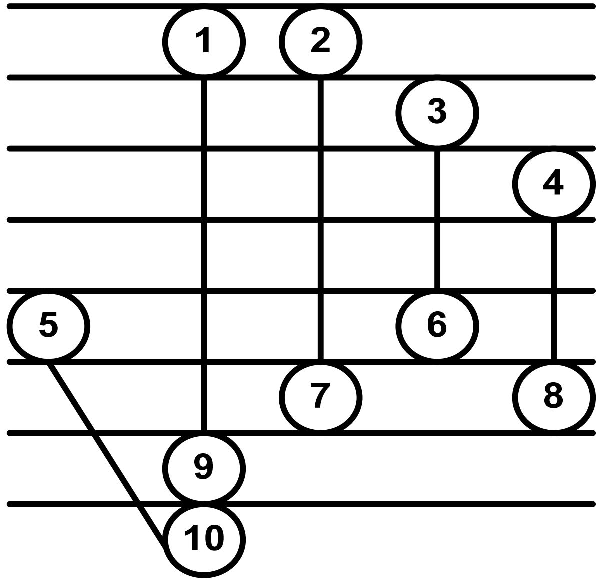
Regular Distributed Register (RDR)

- [Cong, et al., TCAD, 2004]
- High regularity
  - array of logic clusters
- Support multi-cycle communication

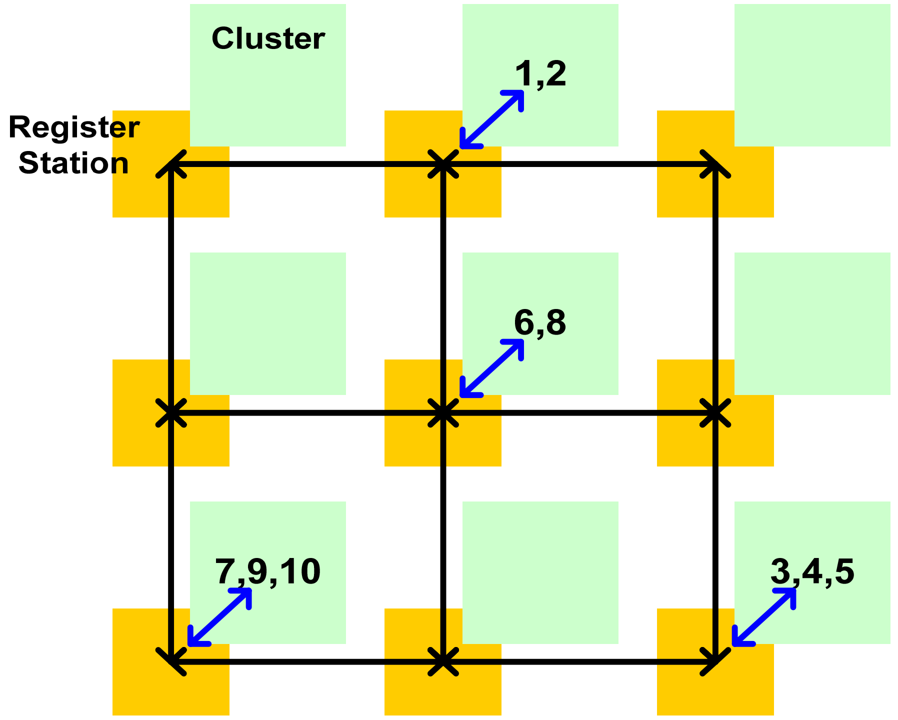


Motivational Example

- Channel and register allocation problem
- Given:
  - scheduled and bound DFG
  - FU placement in RDR



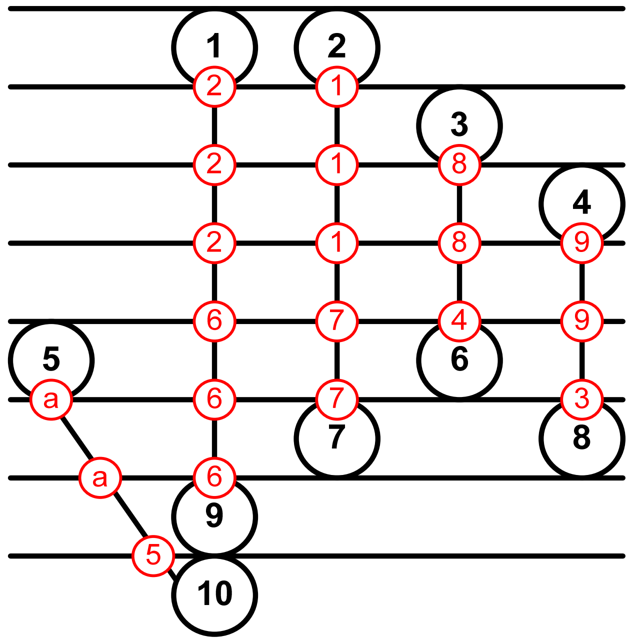
Scheduled and bound DFG

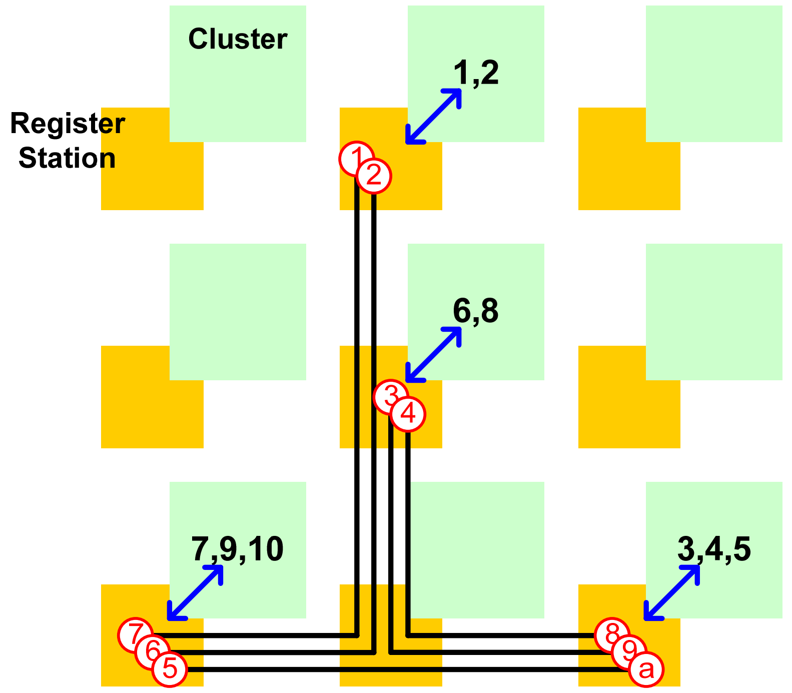


Placed FUs in RDR

RDR/MCAS [Cong et al., TCAD, 2004]

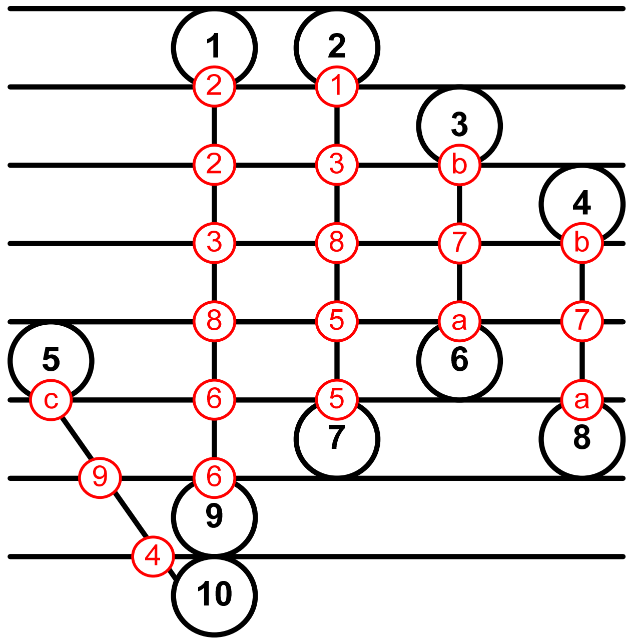
- Dedicated interconnections without pipelining
  - number of wire segments: 12
  - number of registers: 10
- Large amount of dedicated wires are needed
  - severe global **wiring overhead**

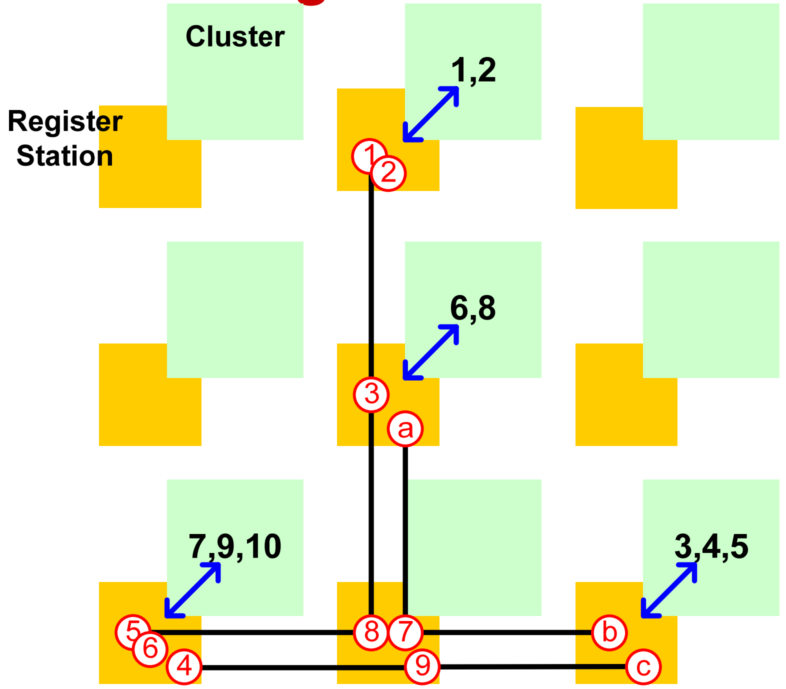




RDR-pipe/MCAS-pipe [Cong et al., DAC, 2004]

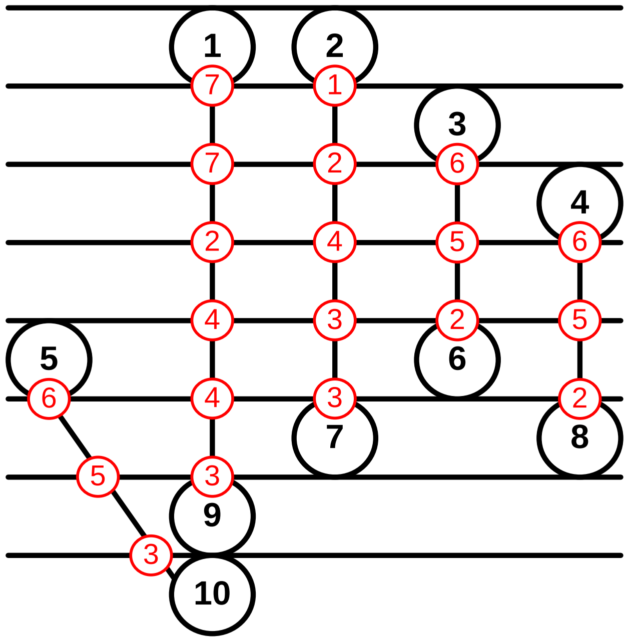
- Dedicated interconnections with pipelining
  - number of wire segments: 12➔7
  - number of registers: 10➔12
- Wires are shared among data transfers
  - **fewer global wires, BUT more registers**

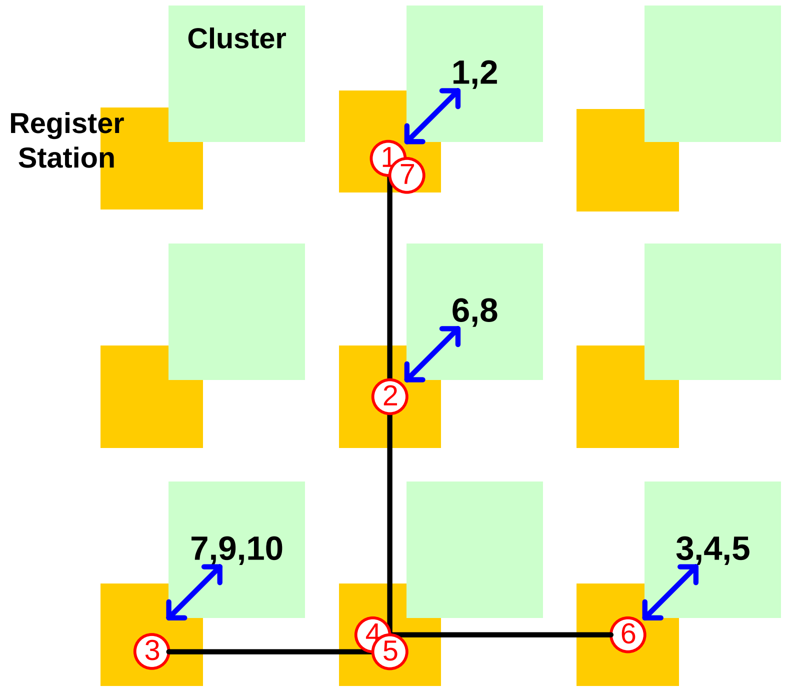




RDR-GRS

- Globally **share the wires and registers** among all clusters ➔ **RDR-GRS**
  - number of wire segments: 12➔7➔**4**
  - number of registers: 10➔12➔**7**





Outlines

- Introduction
- Motivation
- Problem Formulation
- Experiments
- Conclusions & Future Works

Problem Description

- **Given**
  - a data transfer set and associated RDR architecture specification
- **Constraint**
  - all data **MUST** arrive their destinations before the corresponding deadlines
- **Objective**
  - perform data transfer scheduling such that the required wires and registers are minimized

Problem Formulation

- Model the channel and register allocation problem using **integer linear programming (ILP)**
- Objective function

minimize

$$\sum_{\forall i: rst_i \in R_{st}} \alpha_i \times qr_i + \sum_{\forall j: ch_j \in E_{ch}} \beta_j \times qw_j$$

# of wires

# of registers

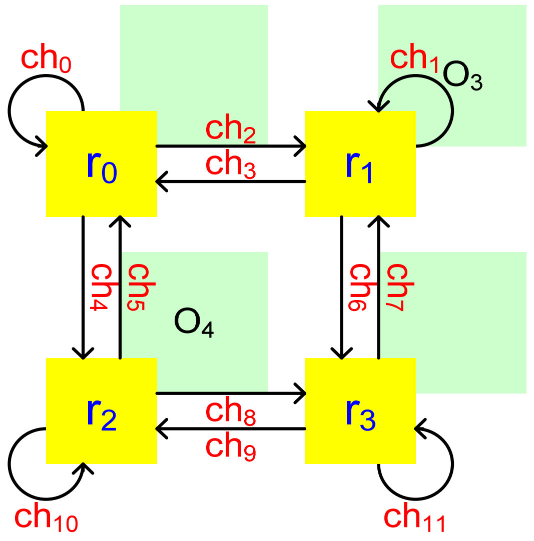
- Variable definition of $\mathbf{x_{i,j,k}}$
  - $\mathbf{x_{i,j,k}}$ is **1** if the data transfer $\mathbf{e_i}$ at cycle $\mathbf{t_j}$ is allocated to the channel $\mathbf{ch_k}$

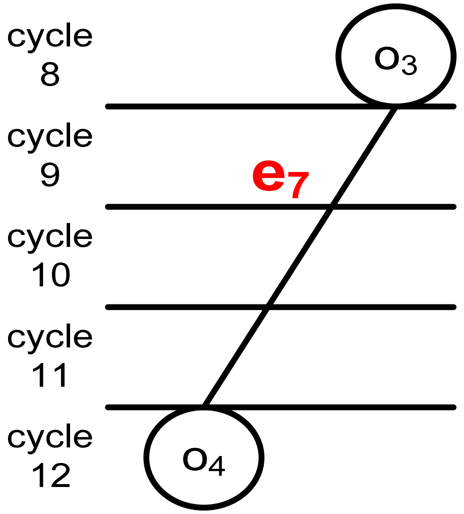
Uniqueness Constraint

- Ensure any data transfer can only occupy **a single channel** at a single cycle

$$\sum_{\forall k, ch_k \in X_{i,j}} x_{i,j,k} = 1, \forall i, j : tr_i \in T_r, req_t(tr_i) > j \geq gen_t(tr_i)$$

e.g., for e_7 at cycle 9 $x_{7,9,1} + x_{7,9,2} + \ldots + x_{7,9,12} = 1$



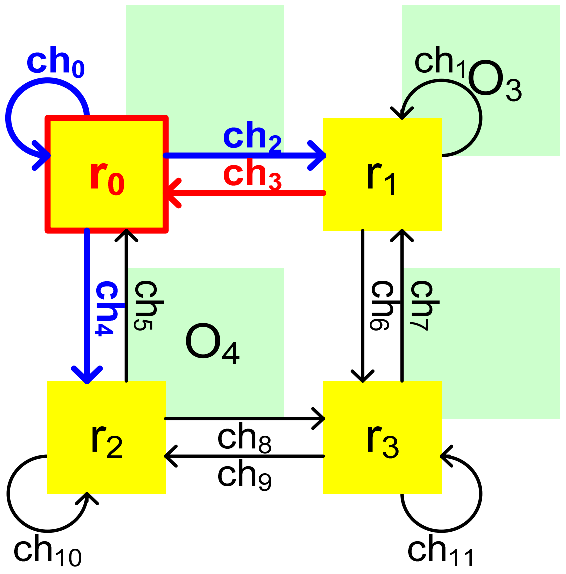


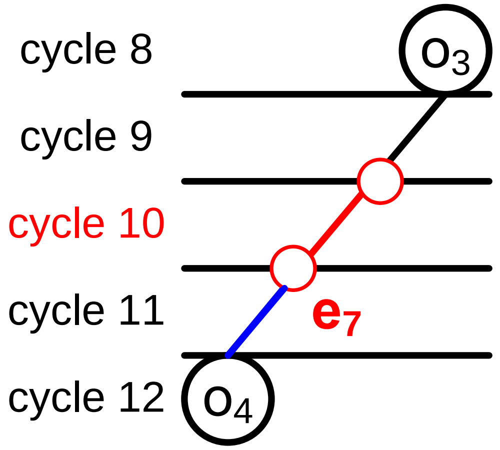
Continuity Constraint

- Ensure any data transfer is continuous in both **temporal and spatial domain**

$$-x_{i,j,k} + \sum_{\substack{\forall k', ch_{k'} \in X_{i,j+1} \\ \bullet ch_{k'} = ch_k \bullet}} x_{i,j+1,k'} \geq 0, \qquad \forall i,j,k: tr_i \in T_r, \; req_t(tr_i)\text{-}1 > j \geq gen_t(tr_i), ch_k \in X_{i,j}$$

e.g., for $x_{7,10,3}$ of e_7 $-x_{7,10,3} + x_{7,11,0} + x_{7,11,2} + x_{7,11,4} \geq 0$





Resource Constraint

- Ensure the number of allocated registers must not be fewer than the number of incoming data transfers at each cycle

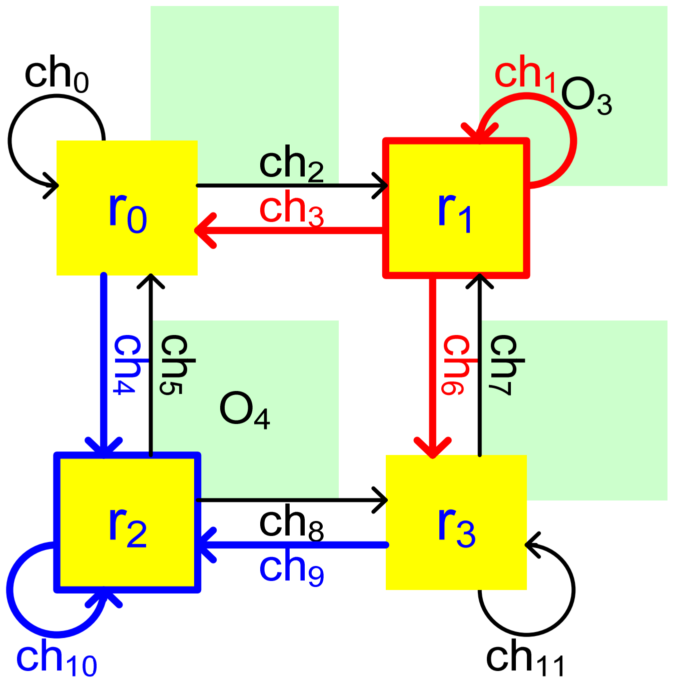
$$\boxed{qr_y} - \boxed{\sum_{\substack{\forall i: req_t(tr_i) > j,\\ j \ge gen_t(tr_i)}} \; \sum_{\substack{\forall k: ch_k \in X_{i,j}\\ ch_k \bullet = rst_y}} x_{i,j,k}} \ge 0, \forall y, j : rst_y \in R_{st}, t_j \in T$$

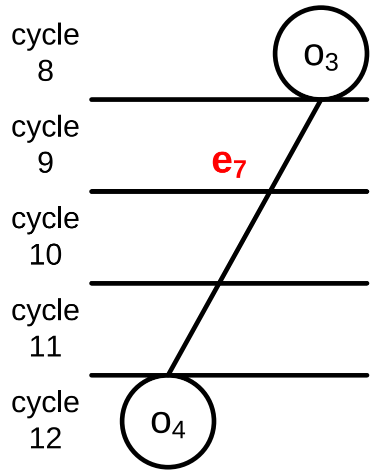
- Ensure the number of allocated wires must not be fewer than the number of transfers using this channel at each cycle

$$\boxed{qw_y} - \boxed{\sum_{\substack{\forall i: req_t(tr_i) > j,\\ j \ge gen_t(tr_i)}} \; \sum_{\forall k: ch_k \in X_{i,j}} x_{i,j,k}} \ge 0, \forall y, j : ch_y \in E_{ch}, t_j \in T$$

Variable Reduction

- Reduce required allocation variables





J (cycle)	$CS_{7,j}$	$CD_{7,j}$	$CS_{7,j} \cap CD_{7,j}$
9	{1,3,6}	{0~11}	{1,3,6}
10	{0,1,2,3,5, 6,7,8,11}	{0,3,4,5,6,8, 9,10,11}	{0,3,5,6, 8,11}
11	{0~11}	{4,9,10}	{4,9,10}

e.g., for e_7 at cycle 11

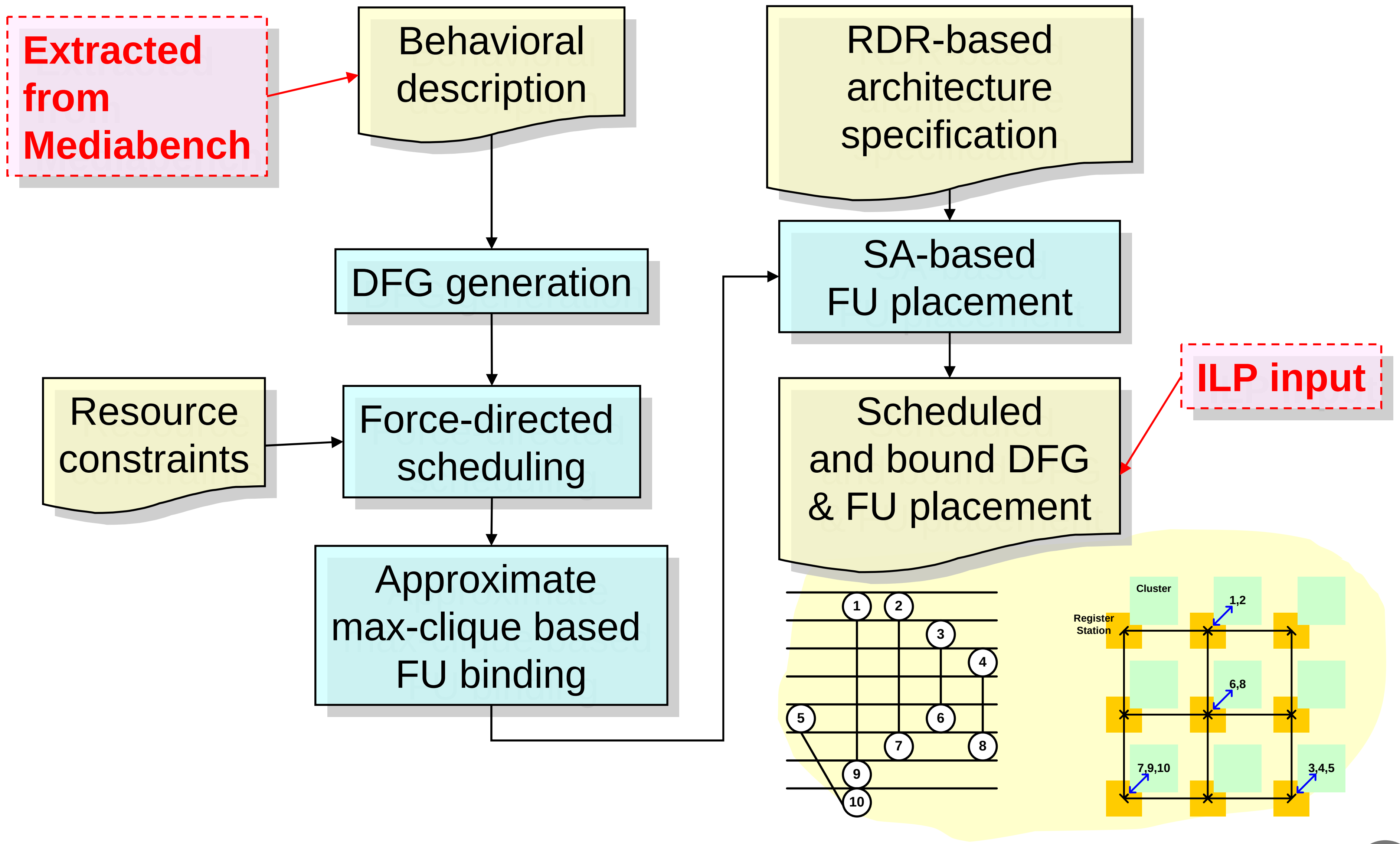
~~$X_{i,j} = \{x_{7,11,1}, x_{7,11,2}, \dots x_{7,11,12}\}$~~

$X_{i,j} = \{x_{7,11,4}, x_{7,11,9}, x_{7,11,10}\}$

Outlines

- Introduction
- Motivation
- Problem Description
- Problem Formulation
- Experiments
- Conclusions & Future Works

Input Generation


Extracted from Mediabench
Behavioral description
DFG generation
Resource constraints
Force-directed scheduling
Approximate max-clique based FU binding
RDR-based architecture specification
SA-based FU placement
Scheduled and bound DFG & FU placement
ILP input
1
2
3
4
5
6
7
8
9
10
Cluster
Register Station
1,2
6,8
7,9,10
3,4,5

Information of Input DFGs

	# node	FU resource		cycle count
		#ALU	#MUL	
(1) mpeg2enc1	66	5	2	23
(2) mpeg2enc2	101	9	4	23
(3) mpeg2enc3	196	18	8	18
(4) jpeg1	93	9	2	35
(5) jpeg2	109	9	2	33
(6) jpeg3	140	11	6	23
(7) rasta	119	7	5	33

Experiment Settings

- Objective function

$$\sum_{\forall i: rst_i \in R_{st}} 1 \times qr_i + \sum_{\substack{\forall j: ch_j \in E_{ch}, \\ \bullet ch_j \neq ch_j \bullet}} 5 \times qw_j$$

- ILP solver
  – lpsolve 5.5.0.0

- Compared with 2 previous works
  – RDR/MCAS
  – RDR-pipe/MCAS-pipe

Experimental Results

	RDR/MCAS		RDR-pipe/ MCAS-pipe		RDR-GRS/GRS-ILP		
	#wire	#reg	#wire	#reg	#wire	#reg	runtime (sec)
(1)	42	28	37	45	25	25	0.2
(2)	76	53	60	76	42	38	102.6
(3)	130	92	109	133	68	61	6.26
(4)	81	50	64	78	24	28	8.9
(5)	78	44	59	68	28	28	0.98
(6)	75	72	58	87	24	48	235.5
(7)	74	56	57	75	25	27	340.1
avg.	79.4	56.4	63.4	80.3	33.7	36.4	
	1	1	0.80	1.42	0.42	0.65	
	1.25	0.70	1	1	0.53	0.45	

Conclusions

- RDR-GRS architecture
  - **globally share the wires and registers**

- GRS-ILP
  - formulate channel and register allocation as an ILP problem
  - guarantee the optimal solution
  - 58% and 35% reduction in wires and registers compared to RDR/MCAS
  - 47% and 55% reduction in wires and registers compared to RDR-pipe/MCAS-pipe

Future Works

- An efficient heuristic is under development
  - deal with large-scale problems
- Take more design factors into consideration
  - e.g., MUX area

Thank you!