www.cnrs.fr/ www.ens-lyon.fr/ www.inria.fr/ www.ens-lyon.fr/LIP/ www.univ-lyon1.fr/ www.universite-lyon.fr/

Affine Multibanking for High-Level Synthesis

Focus: High-Level Synthesis (HLS) for FPGA, targetting Von Neumann architecture model

Challenge: Memory contention induced by concurrent accesses

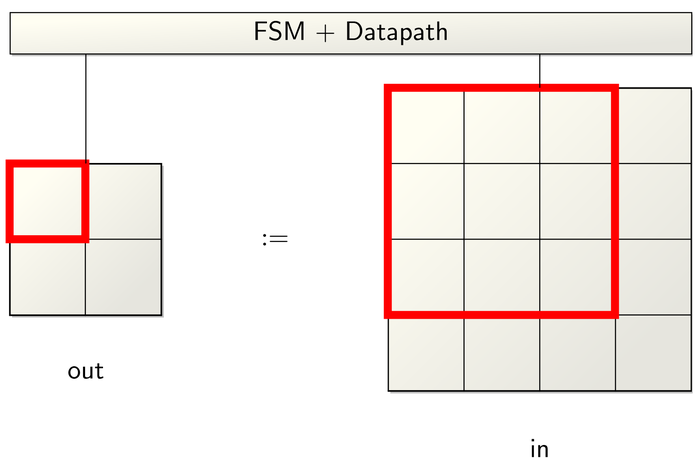
Contribution: **HLS algorithm** for array restructuring with **affine multibanking** using the **polyhedral model**

1) High-Level Synthesis at a glance

High-Level Synthesis (HLS): Program → Hardware

- Typically: compute-intensive kernel → hardware accelerator IP
- Target: ASIC or FPGA

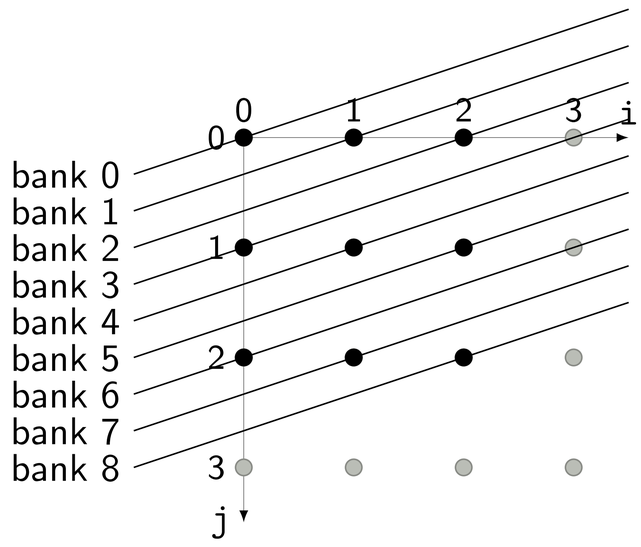
```
for(i=1; i<N-1; i++)
  for(j=1; j<N-1; j++)
    out[i,j] =
      in[i-1,j-1]+in[i-1,j]+in[i-1,j+1]+
      in[i,j-1]  +in[i,j]  +in[i,j+1]  +
      in[i+1,j-1]+in[i+1,j]+in[i+1,j+1]; //S
```



Features:

- **Von Neumann architecture model** (e.g. VivadoHLS)
- Parallelism hindered by **limited read ports**

2) Solution: Multibanking



(Affine) multibanking mappings:

For each array cell in[i,j]:

- **Bank number:** $\text{BANK}_{in}(i,j) = i + 3j \bmod 9$
- **Position in a bank:** $\text{OFFSET}_{in}(i,j) = j \bmod N$

Source-to-source compilation methodology:

- **Rewrite** each $in[u(\vec{i})]$ as $\hat{in}[\text{BANK}_{in}(u(\vec{i}))][\text{OFFSET}_{in}(u(\vec{i}))]$
- **Add HLS pragmas** to partition the bank dimensions:

```
#pragma HLS ARRAY_PARTITION variable=in_ cycle dim=1 ↩
 factor=9
```

3) Our Multibanking Approach

Banking constraints:

- **Correctness:** enforce distinct banks for concurrent access

$$a(\vec{i}) \parallel_\theta b(\vec{j}) \wedge (a,\vec{i}) \neq (b,\vec{j}) \Rightarrow \text{BANK}_a(\vec{i}) \neq \text{BANK}_b(\vec{j})$$

Relaxed as:

$$a(\vec{i}) \parallel_\theta b(\vec{j}) \wedge (a,\vec{i}) \neq (b,\vec{j}) \Rightarrow \phi_a(\vec{i}) \ll \phi_b(\vec{j})$$

- **Efficiency:** reduce the bank number for each dimension

$$\phi_b(\vec{j}) - \phi_a(\vec{i}) \leq \sigma(\vec{N})$$

Analogous to **affine scheduling:**

operation	array cell
dependence	concurrent access
latency	number of banks

Offset constraints (similar):

- **Correctness:** enforce distinct offsets for conflicting array cells

$$\text{BANK}_a(\vec{i}) = \text{BANK}_b(\vec{j}) \wedge a(\vec{i}) \bowtie_\theta b(\vec{j}) \wedge (a,\vec{i}) \neq (b,\vec{j}) \Rightarrow \text{OFFSET}_a(\vec{i}) \neq \text{OFFSET}_b(\vec{j})$$

Relaxed as:

$$\phi_a(\vec{i}) = \phi_b(\vec{j}) \wedge a(\vec{i}) \bowtie_\theta b(\vec{j}) \wedge (a,\vec{i}) \neq (b,\vec{j}) \Rightarrow \psi_a(\vec{i}) \ll \psi_b(\vec{j})$$

- **Efficiency:** minimize the number of offsets (into a same bank)

$$\phi_a(\vec{i}) = \phi_b(\vec{j}) \Rightarrow \psi_b(\vec{j}) - \psi_a(\vec{i}) \leq \tau(\vec{N})$$

4) Experimental Results

Kernel	Version	Latency	Interval	**Speed-up**	BRAM18K	DSP	FF	LUT	URAM
matvec	Baseline	532	533	**10.2**	0	320	4799	4423	0
	With banking	52	53		0	320	67618	13518	0
matmul	Baseline	1555	1556	**29.9**	0	10240	135581	123129	0
	With banking	52	53		0	10240	196648	152161	0
conv2d	Baseline	1442	1443	**29.4**	0	0	923	4290	0
	With banking	49	50		0	0	65562	33043	0
jacobi2d	Baseline	11011	11012	**1.6**	0	0	117140	96019	0
	With banking	6851	6852		0	0	192295	137499	0
seidel2d	Baseline	6914	6915	**2.0**	0	0	452	1280	0
	With banking	3458	3459		0	0	574	2903	0
canny	Baseline	10194	10195	**4.3**	0	0	669	1837	0
	With banking	2355	2356		0	0	6616	6085	0
gaussian	Baseline	3922	3923	**1.7**	0	0	449	1012	0
	With banking	2354	2355		0	0	2367	2811	0
median	Baseline	3362	3363	**1.3**	0	0	373	846	0
	With banking	2522	2523		0	0	2367	2501	0
prewitt	Baseline	3846	3847	**2.0**	0	0	371	906	0
	With banking	1924	1925		0	0	2249	2142	0

Experimental setup:

- VivadoHLS 2019.1
- Target: Kintex 7 FPGA (XC6K70T-FBV676-1)
- Preliminary prototyping using **fkcc**

Benchmarks:

- *Linear algebra:* matvec, matmul
- *Stencils:* jacobi2d, seidel2d
- *Convolutions:* conv2d, canny, gaussian, median, prewitt

Ilham Lasfar, Christophe Alias, Matthieu Moy, Rémy Neveu, Alexis Carré

INRIA, ENS de Lyon, France