清華大學
Tsinghua University

Energy-Aware Loops Mapping on Multi-Vdd CGRAs Without Performance Degradation

Speaker: Jiangyuan Gu

Date: January 16th, 2016

From Tsinghua University

Content

1 Introduction and Background

2 Motivation of Our Problem

3 Problem Formulation

4 Solution Implementation

5 Experimental Results

Content

1 Introduction and Background

2 Motivation of Our Problem

3 Problem Formulation

4 Solution Implementation

5 Experimental Results

Introduction and Background

◆ Nowadays, the embedded systems, such as the mobile phones and video processing devices, are required to meet tighter constraints on both high performance and high energy-efficiency;

◆ Coarse-grained reconfigurable architectures (CGRAs) have been received a huge attention due to their high energy efficiency and performance [5][6][9] ;

◆ Since loops are the major portion of applications and dominate the total execution time, our work is focused on mapping loop applications onto CGRA .

◆ There are many works being studied on CGRAs, but they are usually either focused on performance optimization or aimed at energy reduction alone, so that they are rarely to consider both performance and energy optimization;

Introduction and Background

- Modulo scheduling [1][2][3][11] is one of the pipelining techniques and widely used in loops mapping on CGRAs to achieve high performance, which overlaps the successive iterations to reduce ***II*** (*Initiation Interval*). Here, ***II*** is the number of cycles between the start of two consecutive iterations of loops.
  - In [3], a simulated annealing based on modulo scheduling approach is proposed to find loop mappings on CGRAs;
  - In [2], EPIMap formalizes loops mapping problem as graph epimorphism problem using routing and re-computations and attempts to find a maximal common subgraph (MCS) [8] between DFG (Data Flow Graph) and TEC (Time-Extended CGRA);
  - In [11], a force-directed scheduling and mapping approach was employed to solve DFG scheduling and mapping problems on CGRAs;
  - There are also many other works, but all of them are mainly focused on the performance optimization to achieve smaller ***II*** only;

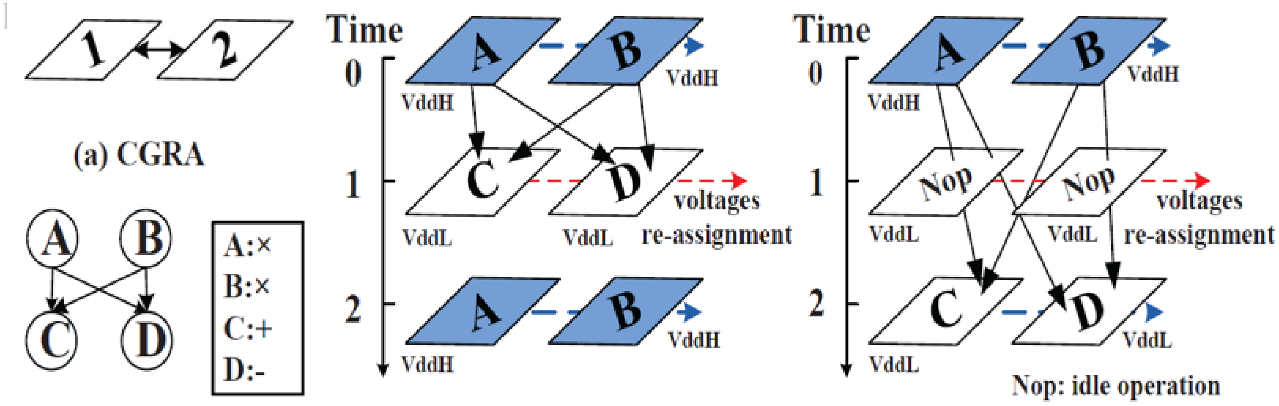
Introduction and Background

- Multi-Vdd assignment technique is widely employed to CGRAs for reducing both dynamic and static power consumptions [4] [5] [6] ;
  - Higher voltage (VddH) is assigned to the processing elements (PEs) to execute the long-delay operations , (e.g. multiplication);
  - Lower voltage (VddL) is assigned to PEs executing the short-delay operations, (e.g. addition);
  - In [4], a low-power CGRA utilizing variable dual-Vdd is proposed in [4], achieving an average power reduction by 15% with area penalty less than 3% with VddL fixed at 0.6-0.75V;
  - In [9], a dynamic dual Vdd switching and mapping method is proposed in which can reduce energy dissipation by 12.5% with the area penalty less than 1%;

Introduction and Background

- According to Vdd assignment ways, dual-Vdd technique are divided into two categories, **Temporal Vdd assignment** and **Spatial Vdd assignment**;

- **Temporal Multi-Vdd Assignment:** the higher or lower supply voltages are ***dynamically configured*** for each PE by context in [6];
  - executing operations and Vdd switching between lower voltage and higher voltage can’t be done in one cycle simultaneously
  - Extra cycle for voltage conversion and settlement;



(a) CGRA

(b) Loop DFG

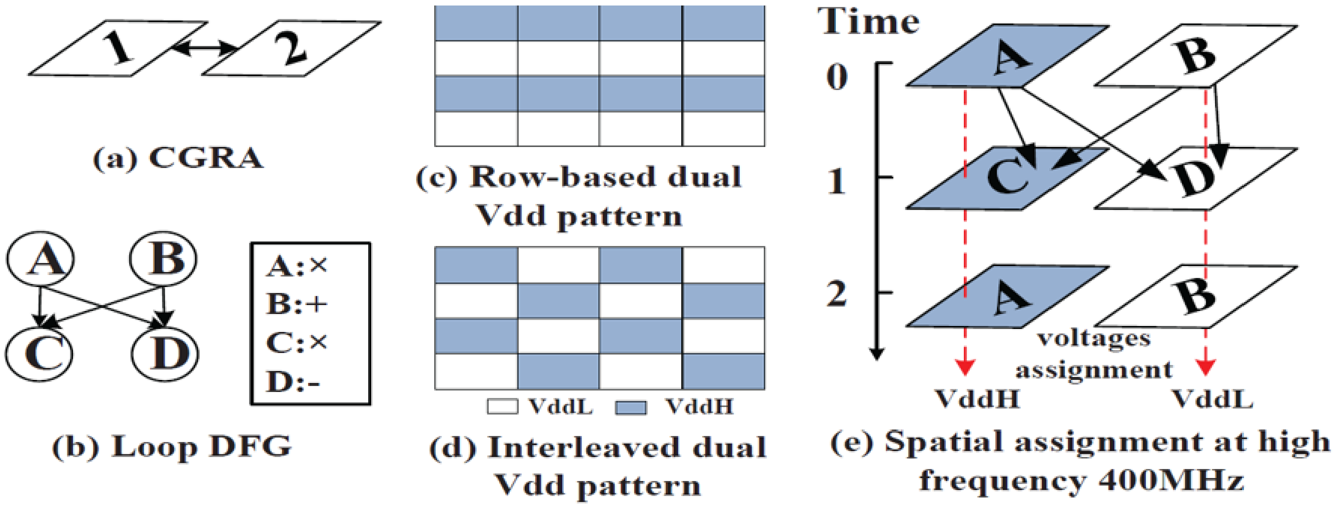
(c) Temporal assignment at low frequency 100MHz

(d) Temporal assignment at high frequency 400MHz

Introduction and Background

◆ **Spatial multi-Vdd Assignment**: to avoid influencing performance, in the spatial multi-Vdd assignment, the voltages on each PEs don't change during loops execution once configured;

- ➢ Pre-mapping approach [4]: multi-Vdd assignment before schedule and mapping procedures;
- ➢ Post-mapping approach [7]: multi-Vdd assignment after schedule and mapping procedures;



(a) CGRA

(b) Loop DFG

(c) Row-based dual Vdd pattern

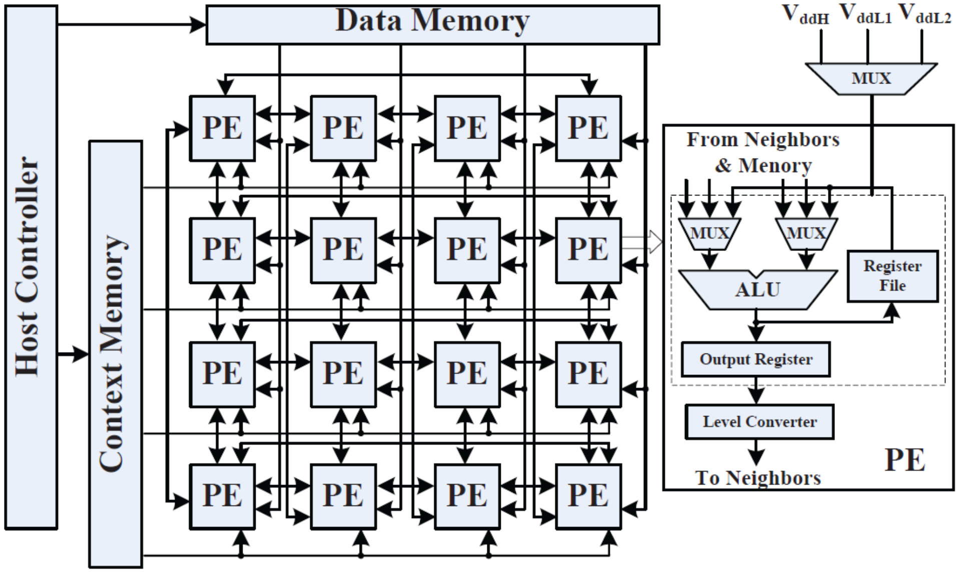
(d) Interleaved dual Vdd pattern

(e) Spatial assignment at high frequency 400MHz

Background and Related Works

- ◆ A joint loops mapping approach is proposed in our work, which integrates the pipelining technique of modulo scheduling and the spatial multi-Vdd assignment technique;

- ◆ Multi-Vdd CGRA Architecture: 2-D PE array (PEA), host controller, context and data memory, etc; Each PE：ALU, MUX, Registers, power switch and level converter , etc;



Content

1 Introduction and Background

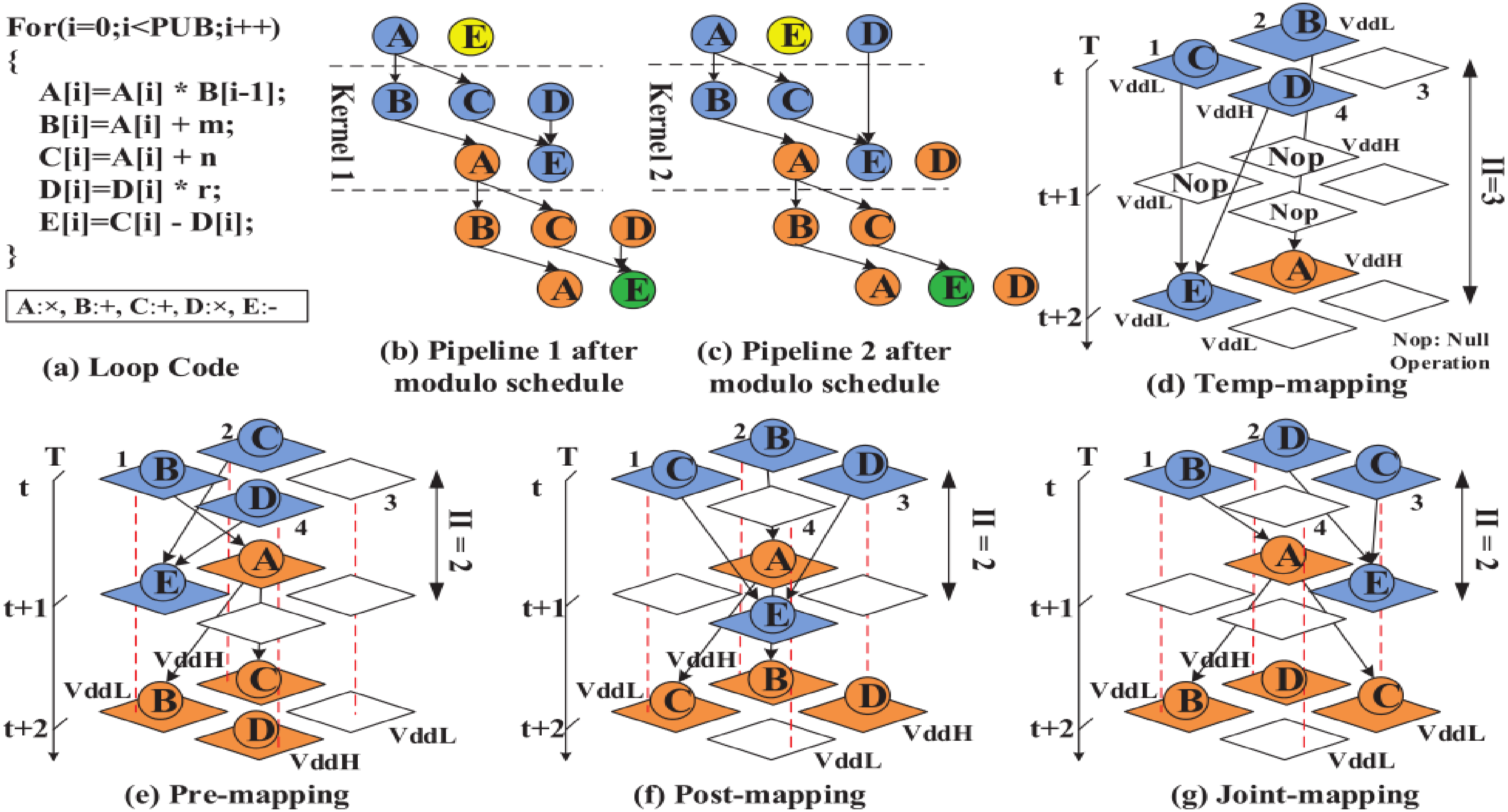
2 Motivation of Our Problem

3 Problem Formulation

4 Solution Implementation

5 Experimental Results

Motivation of Our Problem



```
For(i=0;i<PUB;i++)
{
   A[i]=A[i] * B[i-1];
   B[i]=A[i] + m;
   C[i]=A[i] + n
   D[i]=D[i] * r;
   E[i]=C[i] - D[i];
}
```

A:×, B:+, C:+, D:×, E:-

(a) Loop Code

(b) Pipeline 1 after modulo schedule

(c) Pipeline 2 after modulo schedule

(d) Temp-mapping

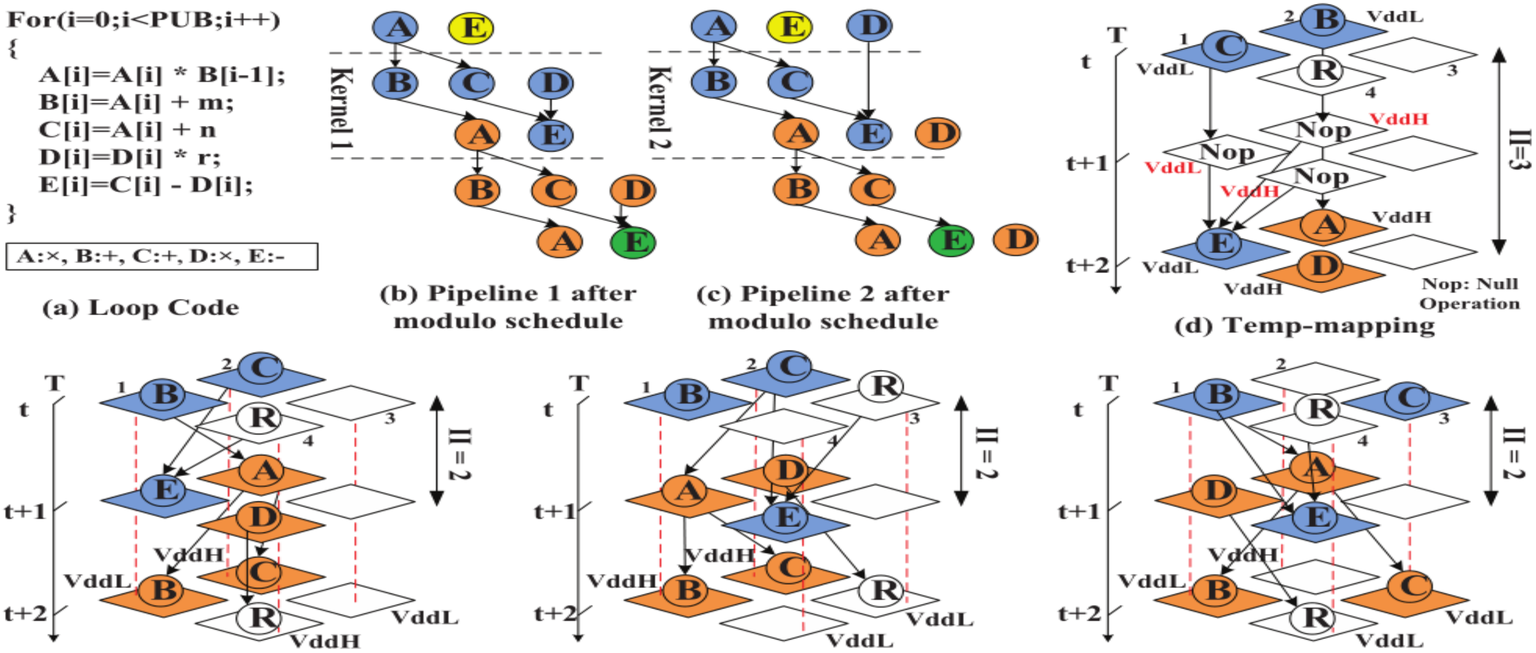
(e) Pre-mapping

(f) Post-mapping

(g) Joint-mapping

Figure 2. A motivation example.

Motivation of Our Problem



```
For(i=0;i<PUB;i++)
{
   A[i]=A[i] * B[i-1];
   B[i]=A[i] + m;
   C[i]=A[i] + n
   D[i]=D[i] * r;
   E[i]=C[i] - D[i];
}
```

A:×, B:+, C:+, D:×, E:-

(a) Loop Code

(b) Pipeline 1 after modulo schedule

(c) Pipeline 2 after modulo schedule

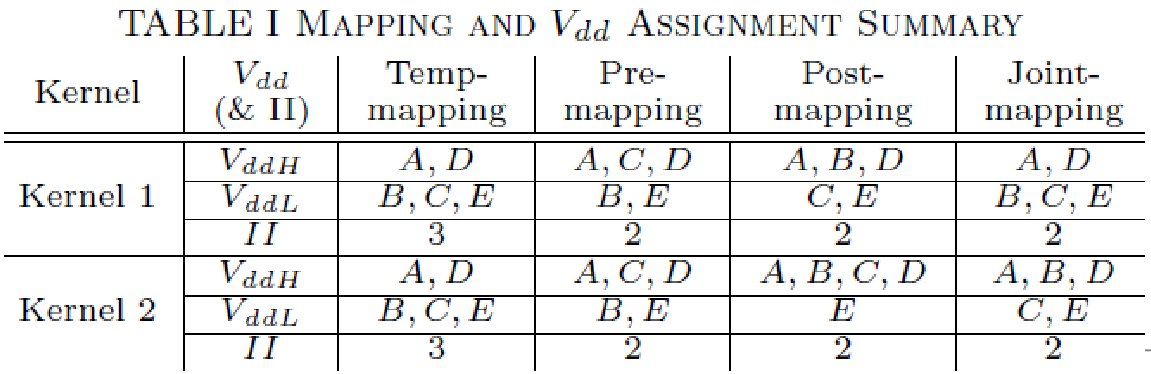
(d) Temp-mapping

(e) Pre-mapping

(f) Post-mapping

(g) Joint mapping

TABLE I Mapping and V_{dd} Assignment Summary



Kernel	V_{dd} (& II)	Temp-mapping	Pre-mapping	Post-mapping	Joint-mapping
Kernel 1	V_{ddH}	A, D	A, C, D	A, B, D	A, D
	V_{ddL}	B, C, E	B, E	C, E	B, C, E
	II	3	2	2	2
Kernel 2	V_{ddH}	A, D	A, C, D	A, B, C, D	A, B, D
	V_{ddL}	B, C, E	B, E	E	C, E
	II	3	2	2	2

Motivation of Our Problem

◆ In ***Temp-Mapping*** approach, extra clock cycles are needed for the PEs to perform voltage conversion and reconfiguration. Although achieving a lower energy consumption, it causes a larger ***II*** and degraded performance;

◆ In ***Pre-Mapping*** approach, in order to not enlarge ***II***, the pre-defined Vdd pattern usually limits the loop operations mapping and causes a worse Vdd assignment scheme and a larger energy consumption;

◆ In ***Post-Mapping*** approach, since modulo scheduling is first performed without any consideration of affect of voltage assignment, loop operations are mapped on PEs arbitrarily and blindly, which also limits the power optimization space and causes a larger energy consumption;

Motivation of Our Problem

- In our ***Joint-Mapping*** approach, the loops mapping and Vdd assignment are integrated together, which increases the potential of the lower voltage VddL assignment. So it can help to reduce the number of operations assigned VddH and bring a lower energy consumption;

- From those two pipelined kernels of two different schedules above, the schedules with more balanced operation distribution along the time-axe, not only increases the probability of successful mapping but also helps to achieve lower energy consumptions;

- In a word, the Vdd assignment is closely correlated to the loops scheduling and mapping procedures, so it is significant to jointly model and resolve the multi-Vdd assignment and loops scheduling and mapping together.

Content

1 Introduction and Background

2 Motivation of Our Problem

3 Problem Formulation

4 Solution Implementation

5 Experimental Results

Problem Formulation

- To formulate and model our joint loops mapping optimization problem, we extend the compatibility concept proposed in [2][8][10];

- Our mapping includes two mappings:
  - $OP \rightarrow \mathrm{PE}_j^t$
  - $\mathrm{PE}_j^t \rightarrow V_{\mathrm{dd}}$

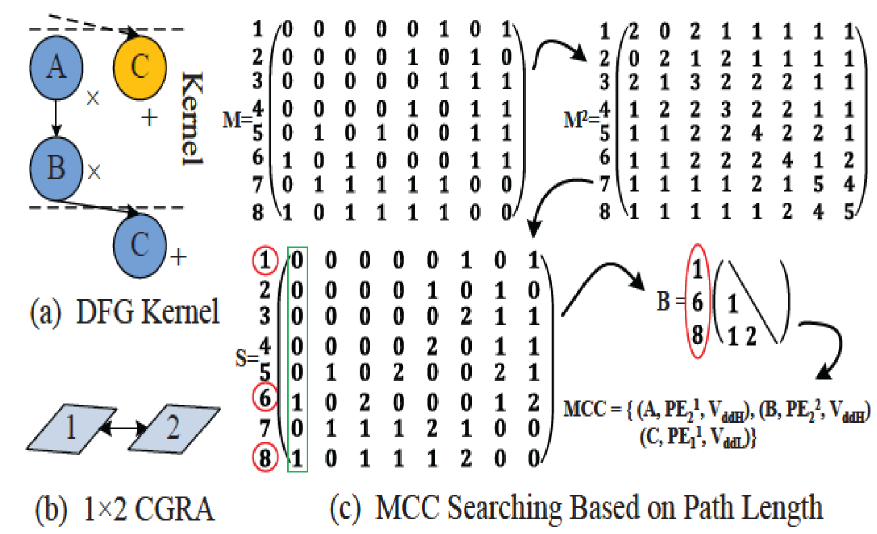
- For the compatibility pair (PE_j^t, OP), we add another attribute Vdd into the triple $(\mathrm{PE}_j^t, OP, V_{\mathrm{dd}})$;

- A valid loop mapping and multi-Vdd assignment scheme is equivalent to find a maximal compatibility class (MCC) from compatibility matrix (CM) generated from the DFG and TEC (Time-Extended CGRA) [10][2][8].

Problem Formulation

- There are there are totally 8 compatibility triples that can be generated according to the definitions above, such as:

$$tr_1 = (C, PE_1^1, V_{ddL}), \quad tr_5 = (A, PE_1^1, V_{ddH}), \quad tr_7 = (B, PE_1^2, V_{ddH})$$



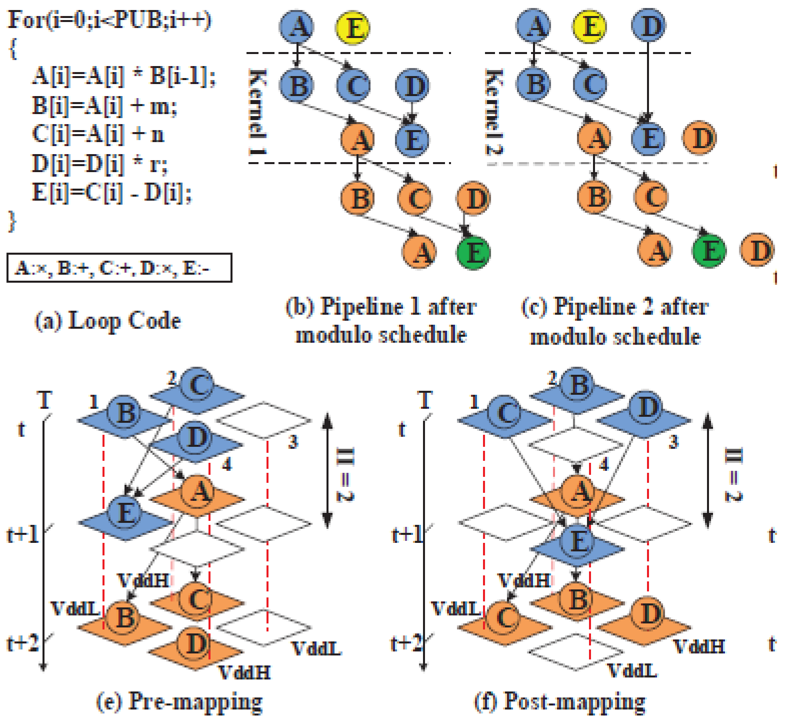
$$M=\begin{array}{c}1\\2\\3\\4\\5\\6\\7\\8\end{array}\begin{pmatrix}0&0&0&0&0&1&0&1\\0&0&0&0&1&0&1&0\\0&0&0&0&0&1&1&1\\0&0&0&0&1&0&1&1\\0&1&0&1&0&0&1&1\\1&0&1&0&0&0&1&1\\0&1&1&1&1&1&0&0\\1&0&1&1&1&1&0&0\end{pmatrix} \qquad M^2=\begin{array}{c}1\\2\\3\\4\\5\\6\\7\\8\end{array}\begin{pmatrix}2&0&2&1&1&1&1&1\\0&2&1&2&1&1&1&1\\2&1&3&2&2&2&1&1\\1&2&2&3&2&2&1&1\\1&1&2&2&4&2&2&1\\1&1&2&2&2&4&1&2\\1&1&1&1&2&1&5&4\\1&1&1&1&1&2&4&5\end{pmatrix}$$

$$S=\begin{array}{c}1\\2\\3\\4\\5\\6\\7\\8\end{array}\begin{pmatrix}0&0&0&0&0&1&0&1\\0&0&0&0&1&0&1&0\\0&0&0&0&0&2&1&1\\0&0&0&0&2&0&1&1\\0&1&0&2&0&0&2&1\\1&0&2&0&0&0&1&2\\0&1&1&1&2&1&0&0\\1&0&1&1&1&2&0&0\end{pmatrix} \qquad B=\begin{array}{c}1\\6\\8\end{array}\begin{pmatrix}&&\\1&&\\1&2&\end{pmatrix}$$

$$MCC = \{(A, PE_2^1, V_{ddH}), (B, PE_2^2, V_{ddH}) \; (C, PE_1^1, V_{ddL})\}$$

(c) MCC Searching Based on Path Length

Figure 4. MCC Searching Example.



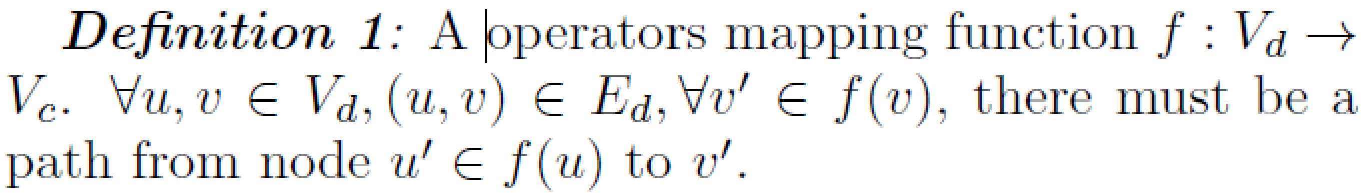
```
For(i=0;i<PUB;i++)
{
   A[i]=A[i] * B[i-1];
   B[i]=A[i] + m;
   C[i]=A[i] + n
   D[i]=D[i] * r;
   E[i]=C[i] - D[i];
}
```

A:×, B:+, C:+, D:×, E:-

Problem Formulation

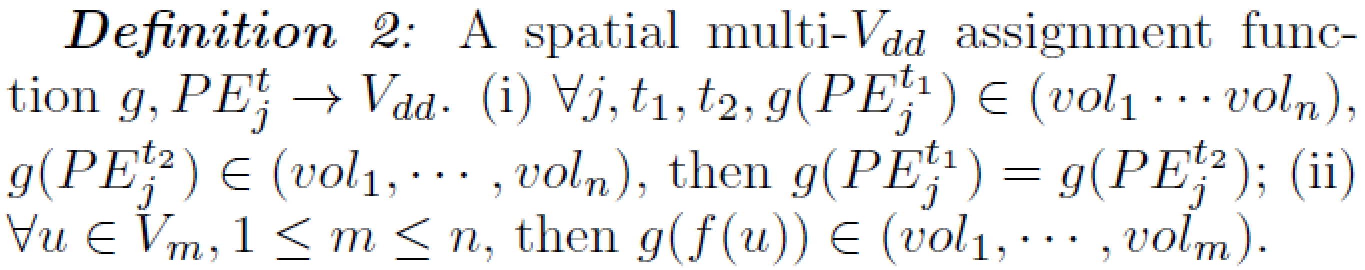
- ◆ For a valid loops mapping and multi Vdd assignment scheme, there are two definition constraints as follows:

➢ **Definition 1: Valid Mapping**



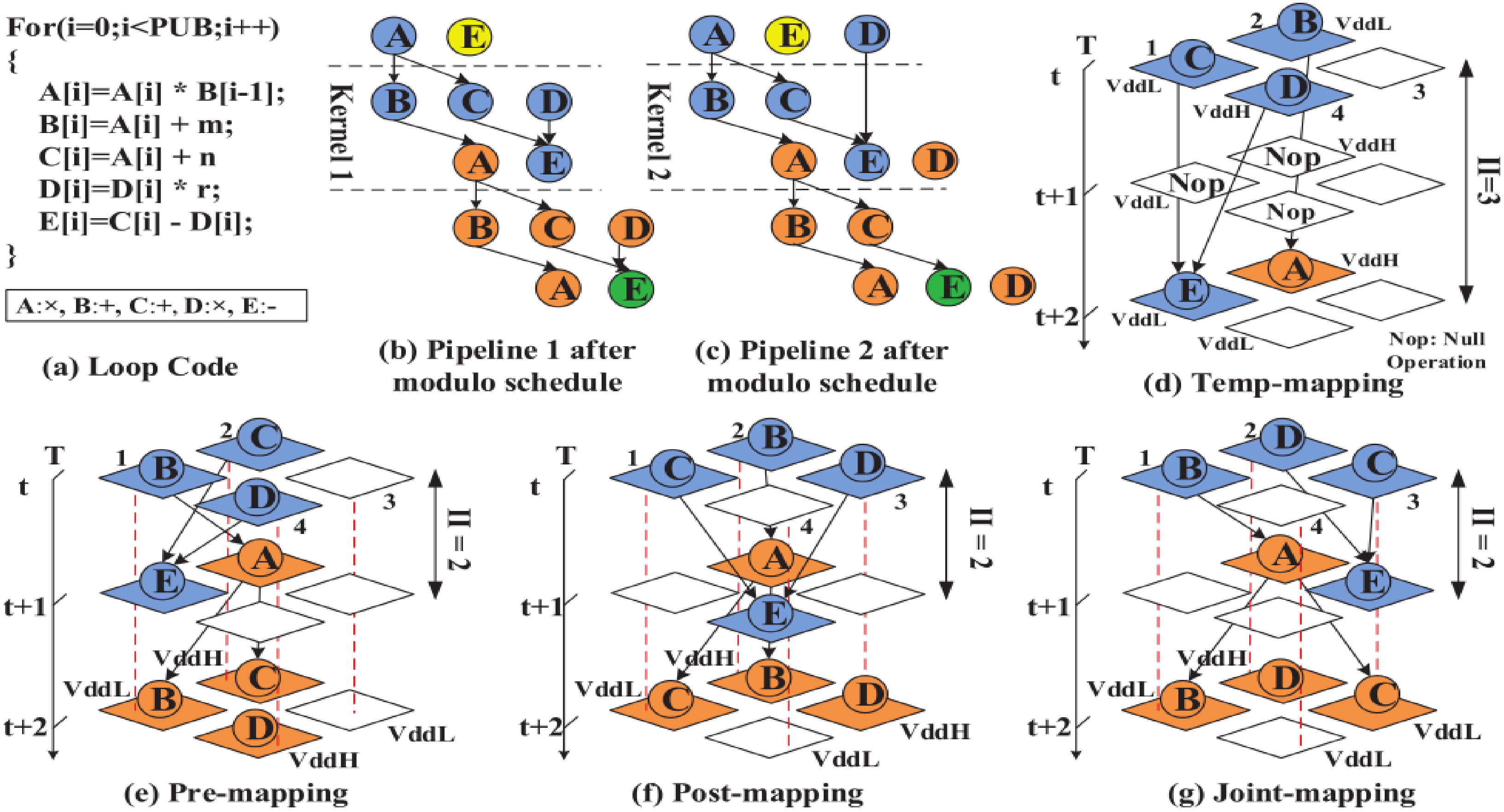
Definition 1: A operators mapping function $f : V_d \to V_c$. $\forall u, v \in V_d, (u, v) \in E_d, \forall v' \in f(v)$, there must be a path from node $u' \in f(u)$ to v'.

➢ **Definition 2: Valid Spatial Multi-Vdd Assignment**



Definition 2: A spatial multi-V_{dd} assignment function $g, PE_j^t \to V_{dd}$. (i) $\forall j, t_1, t_2, g(PE_j^{t_1}) \in (vol_1 \cdots vol_n)$, $g(PE_j^{t_2}) \in (vol_1, \cdots, vol_n)$, then $g(PE_j^{t_1}) = g(PE_j^{t_2})$; (ii) $\forall u \in V_m, 1 \leq m \leq n$, then $g(f(u)) \in (vol_1, \cdots, vol_m)$.

Problem Formulation



```
For(i=0;i<PUB;i++)
{
    A[i]=A[i] * B[i-1];
    B[i]=A[i] + m;
    C[i]=A[i] + n
    D[i]=D[i] * r;
    E[i]=C[i] - D[i];
}
```

A:×, B:+, C:+, D:×, E:-

(a) Loop Code

(b) Pipeline 1 after modulo schedule

(c) Pipeline 2 after modulo schedule

(d) Temp-mapping

(e) Pre-mapping

(f) Post-mapping

(g) Joint-mapping

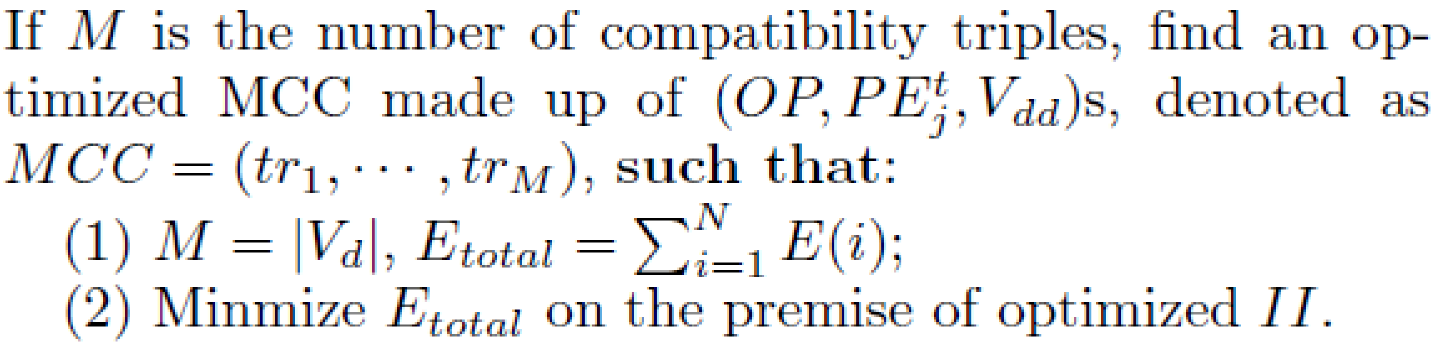
Figure 2. A motivation example.

Problem Formulation

- Since performance is still of most importance in most applications, the first objective is minimizing ***II*** in our work;

- Minimizing the number of PE with assigned the higher voltage VddH (***Nvddh***) helps to reducing the energy consumption ;
  - Under the achieved best ***II***, we try to minimize ***Nvddh*** to reduce energy;

- Our whole problem is formulated as follows:



If M is the number of compatibility triples, find an optimized MCC made up of (OP, PE_j^t, V_{dd})s, denoted as $MCC = (tr_1, \cdots, tr_M)$, such that:

(1) $M = |V_d|$, $E_{total} = \sum_{i=1}^{N} E(i)$;

(2) Minmize E_{total} on the premise of optimized II.

Content

1 Introduction and Background

2 Motivation of Our Problem

3 Problem Formulation

4 Solution Implementation

5 Experimental Results

Solution Implementation

- In order to achieve high performance and energy efficiency simultaneously, our proposed energy-aware joint loops mapping approach can resolve such bi-objective optimization problem effectively for **three main reasons** below:

  - It begins searching valid mappings from a smaller ***II*** (***MII***) iteratively to guarantee obtaining smaller ***II***s and better performance;

  - An energy-aware FDS algorithm (eFDS) [12] is applied to find the more balanced schedules to map. It helps to increase the probability of mapping successfully and achieve a lower-energy scheme finally;

  - A rapid MCC searching method based on matrix path length [10] is successfully applied to find the optimized mapping schemes. It searches out all the potentially feasible compatibility triples to add into MCC at once and then removes the invalid ones that cannot co-exist out of them;

Solution Implementation

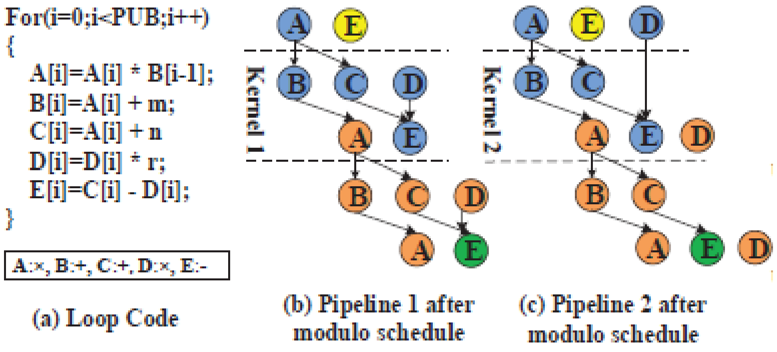
- **Energy-Aware Scheduling by the eFDS**:
  - eFDS tries to make the same type operations, especially these needed to be configured with VddH, scheduled at different timeslot in TEC, which can avoid PEs assigned VddH;
  - First, DFG is pipelined by folding modulo the current ***II***, which can produce different pipelined kernels due to operation mobility [2][3];
  - Next, construct eDGs (Energy Probability Distribution Graphs) and calculate *forces* when operation scheduled to level *j* by Equation 1;

$$
\begin{aligned}
eDG_n(i) &= \sum_k [P_n(k,i) \times E(k)] \\
Force_k(j) &= \sum_{i=1}^{II} [eDG_n(i) * x_k(i)]
\end{aligned}
\qquad (1)
$$

  - Finally, select the lowest *force* for this operation schedule, and repeat these steps until all operations are scheduled.

Solution Implementation

- **Energy-Aware Scheduling by the eFDS**:
  - We assume that the normalized energy consumption for all multiplier, adder and substractor are 1, 0.5 and 0.5, respectively;
  - The final calculated forces of operation ***D*** for ***kernel 1*** and ***kernel 2*** are represented by ***Force(1)*** and ***Force(2)*, ** respectively;
  - Since ***Force(1)*** is smaller than ***Force(2)*,** ***kernel 1*** rather than ker***nel 2*** is selected, which has more balanced energy distribution. This is also consistent with the analysis in the **Motivation** Part above.



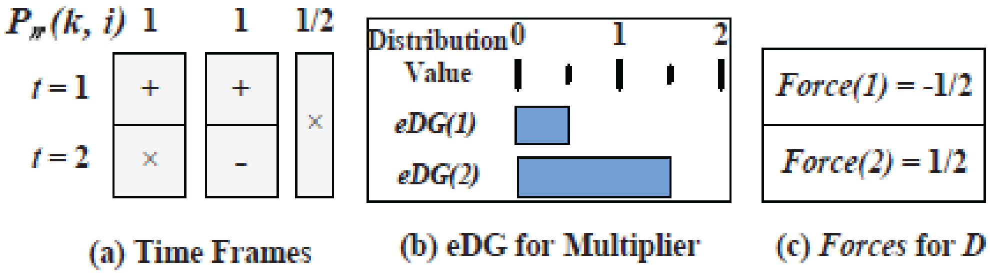
```
For(i=0;i<PUB;i++)
{
   A[i]=A[i] * B[i-1];
   B[i]=A[i] + m;
   C[i]=A[i] + n
   D[i]=D[i] * r;
   E[i]=C[i] - D[i];
}
```

A:×, B:+, C:+, D:×, E:-

(a) Loop Code

(b) Pipeline 1 after modulo schedule

(c) Pipeline 2 after modulo schedule



Force(1) = -1/2

Force(2) = 1/2

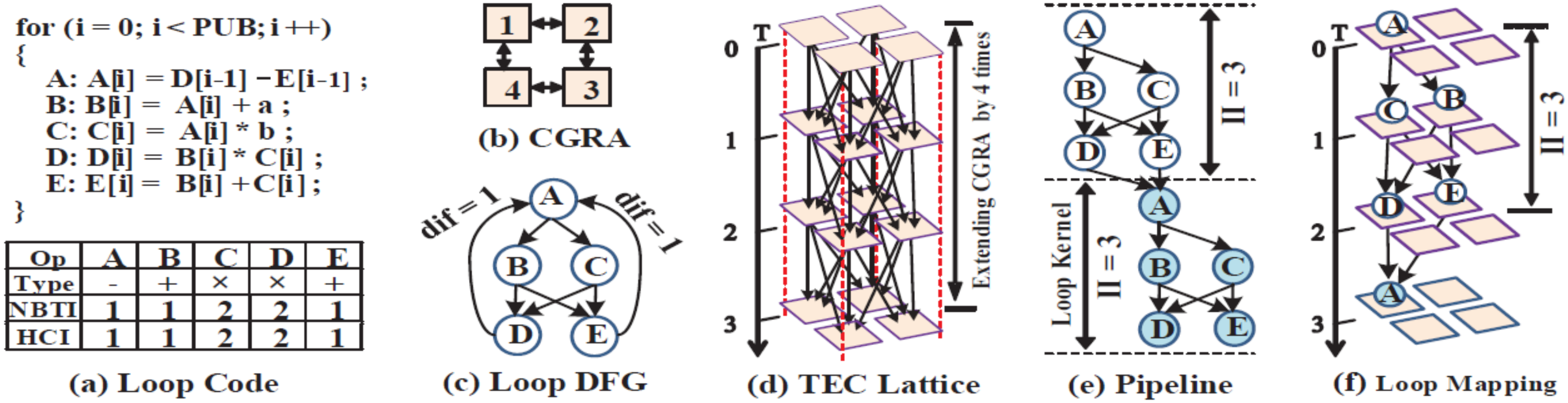
(a) Time Frames

(b) eDG for Multiplier

(c) *Forces for D*

Solution Implementation

◆ Finding A valid mapping and Vdd assignment scheme on CGRA is similar to a graph-to-graph matching problem ;



```
for (i = 0; i < PUB; i ++)
{
   A: A[i] = D[i-1] − E[i-1] ;
   B: B[i] =  A[i] + a ;
   C: C[i] =  A[i] * b ;
   D: D[i] =  B[i] * C[i] ;
   E: E[i] =  B[i] + C[i] ;
}
```

Op	A	B	C	D	E
Type	-	+	×	×	+
NBTI	1	1	2	2	1
HCI	1	1	2	2	1

(a) Loop Code

(b) CGRA

(c) Loop DFG

(d) TEC Lattice

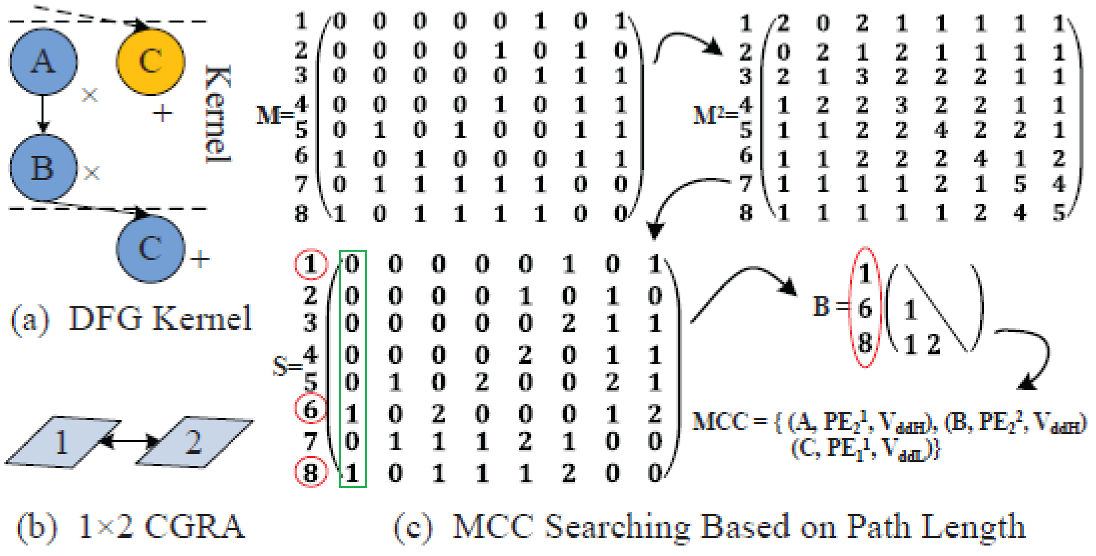
(e) Pipeline

(f) Loop Mapping

Fig. 3: Example of Loops mapping on CGRA, which is similar to a graph-to-graph matching problem and equivalent to find a maximal compatible mapping part between DFG of loop kernel and TEC lattice.

Solution Implementation

- ◆ And finding a valid loop mapping scheme is equivalent to find a maximal compatibility class (MCC) in the compatibility matrix (CM)



$$M=\begin{array}{c}1\\2\\3\\4\\5\\6\\7\\8\end{array}\begin{pmatrix}0&0&0&0&0&1&0&1\\0&0&0&0&1&0&1&0\\0&0&0&0&0&1&1&1\\0&0&0&0&1&0&1&1\\0&1&0&1&0&0&1&1\\1&0&1&0&0&0&1&1\\0&1&1&1&1&1&0&0\\1&0&1&1&1&1&0&0\end{pmatrix}$$

$$M^2=\begin{array}{c}1\\2\\3\\4\\5\\6\\7\\8\end{array}\begin{pmatrix}2&0&2&1&1&1&1&1\\0&2&1&2&1&1&1&1\\2&1&3&2&2&2&1&1\\1&2&2&3&2&2&1&1\\1&1&2&2&4&2&2&1\\1&1&2&2&2&4&1&2\\1&1&1&1&2&1&5&4\\1&1&1&1&1&2&4&5\end{pmatrix}$$

$$S=\begin{array}{c}1\\2\\3\\4\\5\\6\\7\\8\end{array}\begin{pmatrix}0&0&0&0&0&1&0&1\\0&0&0&0&1&0&1&0\\0&0&0&0&0&2&1&1\\0&0&0&0&2&0&1&1\\0&1&0&2&0&0&2&1\\1&0&2&0&0&0&1&2\\0&1&1&1&2&1&0&0\\1&0&1&1&1&2&0&0\end{pmatrix}$$

$$B=\begin{array}{c}1\\6\\8\end{array}\begin{pmatrix}\diagdown&&\\1&\diagdown&\\1&2&\diagdown\end{pmatrix}$$

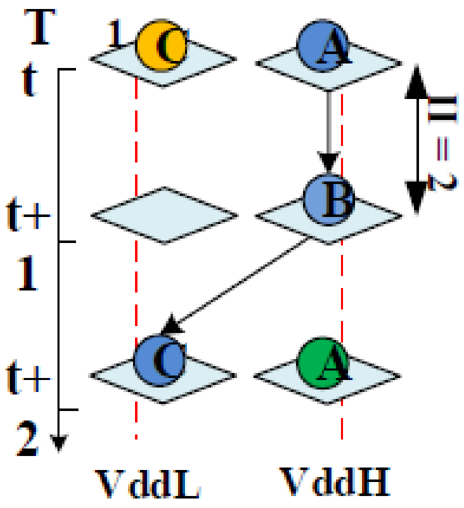
$MCC = \{ (A, PE_2^1, V_{ddH}), (B, PE_2^2, V_{ddH}) (C, PE_1^1, V_{ddL})\}$

(a) DFG Kernel

(b) 1×2 CGRA

(c) MCC Searching Based on Path Length

Figure 4. MCC Searching Example.



Valid Mapping

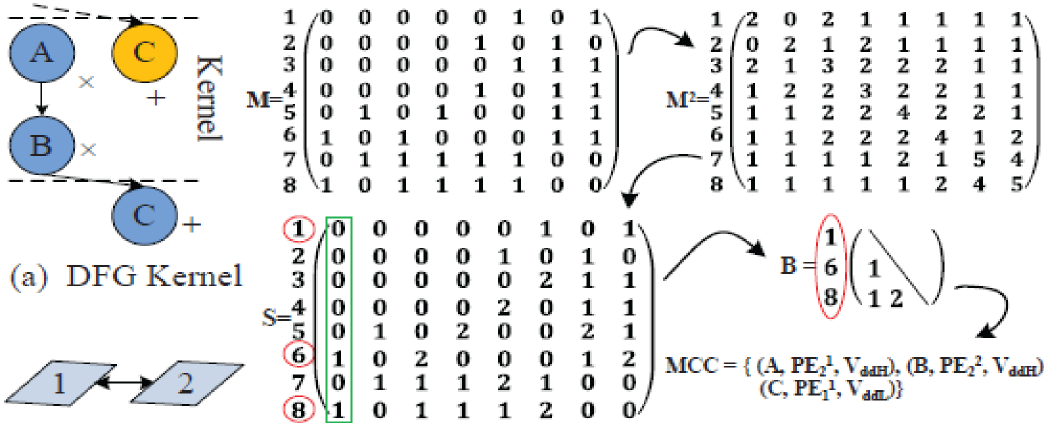
Solution Implementation

- Three conditions should be obeyed to construct the **Compatibility Table**:
  - The matrix entries in CM represent these compatibility mapping triples ($\mathrm{PE}_j^t, OP, V_{\mathrm{dd}}$) ;
  - i) The same PE_j^t must not execute two different operations, and the same OP not mapped to two different Pes;
  - ii) For two triples, if there is an edge between operators in DFG, there must be a communication path between their mapped PEs in TEC. Otherwise the two triples are incompatible;
  - iii) Due to the spatial multi-Vdd assignment constraints, if two triples are mapped to the same PE in CGRA, the Vdd of them must be the same. Otherwise they are incompatible;

Solution Implementation

- **Compatibility Matrix Construction**:
    - 8 compatibility triples totally;

1	(C, PE_1^1, V_{ddL})	2	(C, PE_2^1, V_{ddL})	3	(C, PE_1^1, V_{ddH})	4	(C, PE_2^1, V_{ddH})
5	(A, PE_1^1, V_{ddH})	6	(A, PE_2^1, V_{ddH})	7	(B, PE_1^2, V_{ddH})	8	(B, PE_2^2, V_{ddH})



(a) DFG Kernel

(b) 1×2 CGRA

$$M=\begin{array}{c} 1\\2\\3\\4\\5\\6\\7\\8 \end{array}\begin{pmatrix} 0&0&0&0&0&1&0&1\\ 0&0&0&0&1&0&1&0\\ 0&0&0&0&0&1&1&1\\ 0&0&0&0&1&0&1&1\\ 0&1&0&1&0&0&1&1\\ 1&0&1&0&0&0&1&1\\ 0&1&1&1&1&1&0&0\\ 1&0&1&1&1&1&0&0 \end{pmatrix}$$

$$M^2=\begin{array}{c} 1\\2\\3\\4\\5\\6\\7\\8 \end{array}\begin{pmatrix} 2&0&2&1&1&1&1&1\\ 0&2&1&2&1&1&1&1\\ 2&1&3&2&2&2&1&1\\ 1&2&2&3&2&2&1&1\\ 1&1&2&2&4&2&2&1\\ 1&1&2&2&2&4&1&2\\ 1&1&1&1&2&1&5&4\\ 1&1&1&1&1&2&4&5 \end{pmatrix}$$

$$S=\begin{array}{c} 1\\2\\3\\4\\5\\6\\7\\8 \end{array}\begin{pmatrix} 0&0&0&0&0&1&0&1\\ 0&0&0&0&1&0&1&0\\ 0&0&0&0&0&2&1&1\\ 0&0&0&0&2&0&1&1\\ 0&1&0&2&0&0&2&1\\ 1&0&2&0&0&0&1&2\\ 0&1&1&1&2&1&0&0\\ 1&0&1&1&1&2&0&0 \end{pmatrix}$$

$$B=\begin{array}{c} 1\\6\\8 \end{array}\begin{pmatrix} \diagdown & & \\ 1 & \diagdown & \\ 1 & 2 & \diagdown \end{pmatrix}$$

$MCC = \{ (A, PE_2^1, V_{ddH}), (B, PE_2^2, V_{ddH})$
$(C, PE_1^1, V_{ddL})\}$

(c) MCC Searching Based on Path Length

Figure 4. MCC Searching Example.

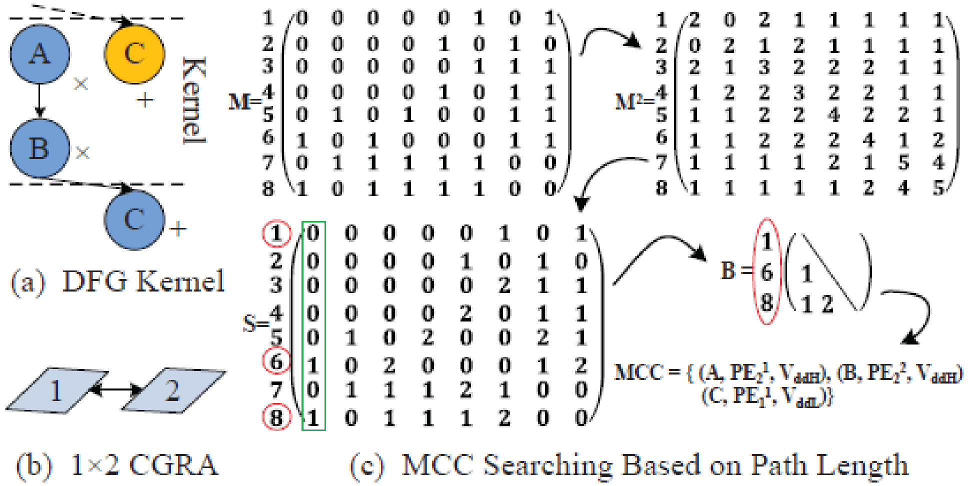
Solution Implementation

- **Find Valid Mappings by Searching MCC in CM:**
  - Finding a valid loop mapping scheme is equivalent to find a valid MCC in the compatibility matrix CM (Matrix ***M*** in the figure)
  - If the obtained number of compatibility triples in MCC is the same as the number of the DFG operations, this MCC represents a valid loops mapping and Vdd assignment scheme;
  - We search the column with more operations of the same types mapped on the same PEs (**help to assign the same Vdd**), calculate the energy consumption of the obtained MCC, and select the MCC with lowest energy consumption as the optimized mapping and Vdd assignment scheme (***Map****) on CGRA for the current loop kernel;

Solution Implementation

◆ Find Valid Mappings by Searching MCC in CM:

- ➢ Since it searches out all the potentially feasible compatibility triples at once and remove the invalid ones that cannot co-exist out them, rather than add the feasible triples into MCC one by one, this MCC searching method can find an optimal mapping and Vdd assignment scheme more rapidly and effectively.



(a) DFG Kernel

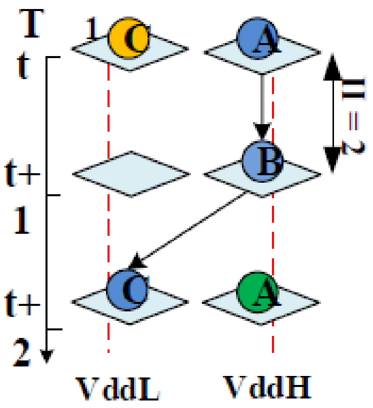
(b) 1×2 CGRA

$$M=\begin{array}{c}1\\2\\3\\4\\5\\6\\7\\8\end{array}\begin{pmatrix}0&0&0&0&0&1&0&1\\0&0&0&0&1&0&1&0\\0&0&0&0&0&1&1&1\\0&0&0&0&1&0&1&1\\0&1&0&1&0&0&1&1\\1&0&1&0&0&0&1&1\\0&1&1&1&1&1&0&0\\1&0&1&1&1&1&0&0\end{pmatrix}\qquad M^2=\begin{array}{c}1\\2\\3\\4\\5\\6\\7\\8\end{array}\begin{pmatrix}2&0&2&1&1&1&1&1\\0&2&1&2&1&1&1&1\\2&1&3&2&2&2&1&1\\1&2&2&3&2&2&1&1\\1&1&2&2&4&2&2&1\\1&1&2&2&2&4&1&2\\1&1&1&1&2&1&5&4\\1&1&1&1&1&2&4&5\end{pmatrix}$$

$$S=\begin{array}{c}1\\2\\3\\4\\5\\6\\7\\8\end{array}\begin{pmatrix}0&0&0&0&0&1&0&1\\0&0&0&0&1&0&1&0\\0&0&0&0&0&2&1&1\\0&0&0&0&2&0&1&1\\0&1&0&2&0&0&2&1\\1&0&2&0&0&0&1&2\\0&1&1&1&2&1&0&0\\1&0&1&1&1&2&0&0\end{pmatrix}\qquad B=\begin{array}{c}1\\6\\8\end{array}\begin{pmatrix}&&\\1&&\\1&2&\end{pmatrix}$$

$MCC = \{ (A, PE_2^1, V_{ddH}), (B, PE_2^2, V_{ddH})$
$(C, PE_1^1, V_{ddL})\}$

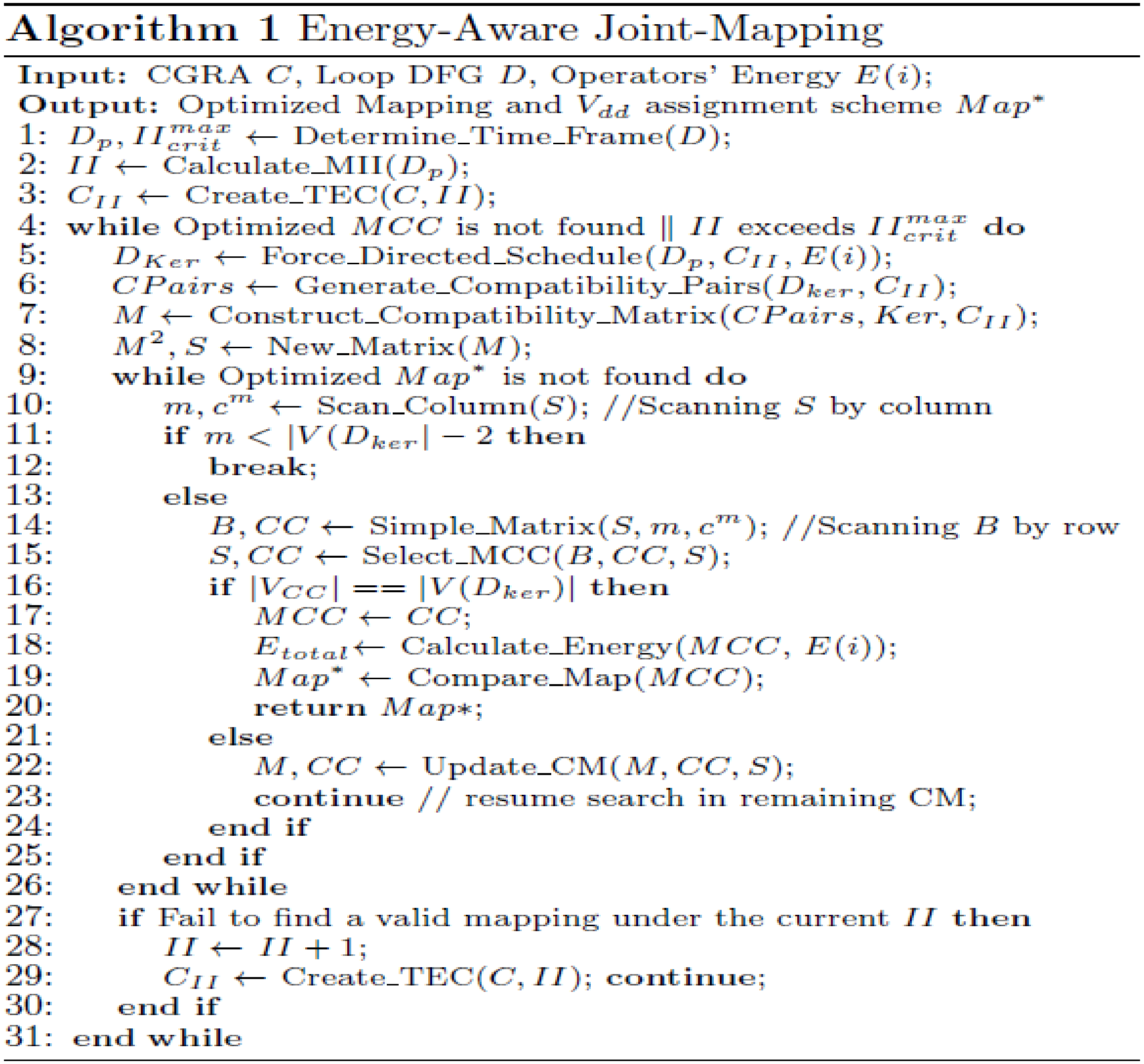
(c) MCC Searching Based on Path Length



Valid Mapping

Figure 4. MCC Searching Example.

Solution Implementation



Algorithm 1 Energy-Aware Joint-Mapping

```
Input: CGRA $C$, Loop DFG $D$, Operators' Energy $E(i)$;
Output: Optimized Mapping and $V_{dd}$ assignment scheme $Map^*$
$D_p, II_{crit}^{max} \leftarrow$ Determine_Time_Frame($D$);
$II \leftarrow$ Calculate_MII($D_p$);
$C_{II} \leftarrow$ Create_TEC($C, II$);
while Optimized $MCC$ is not found $\|$ $II$ exceeds $II_{crit}^{max}$ do
    $D_{Ker} \leftarrow$ Force_Directed_Schedule($D_p, C_{II}, E(i)$);
    $CPairs \leftarrow$ Generate_Compatibility_Pairs($D_{ker}, C_{II}$);
    $M \leftarrow$ Construct_Compatibility_Matrix($CPairs, Ker, C_{II}$);
    $M^2, S \leftarrow$ New_Matrix($M$);
    while Optimized $Map^*$ is not found do
        $m, c^m \leftarrow$ Scan_Column($S$); //Scanning $S$ by column
        if $m < |V(D_{ker}| - 2$ then
            break;
        else
            $B, CC \leftarrow$ Simple_Matrix($S, m, c^m$); //Scanning $B$ by row
            $S, CC \leftarrow$ Select_MCC($B, CC, S$);
            if $|V_{CC}| == |V(D_{ker})|$ then
                $MCC \leftarrow CC$;
                $E_{total} \leftarrow$ Calculate_Energy($MCC$, $E(i)$);
                $Map^* \leftarrow$ Compare_Map($MCC$);
                return $Map*$;
            else
                $M, CC \leftarrow$ Update_CM($M, CC, S$);
                continue // resume search in remaining CM;
            end if
        end if
    end while
    if Fail to find a valid mapping under the current $II$ then
        $II \leftarrow II + 1$;
        $C_{II} \leftarrow$ Create_TEC($C, II$); continue;
    end if
end while
```

Content

1 Introduction and Background

2 Motivation of Our Problem

3 Problem Formulation

4 Solution Implementation

5 Experimental Results

Experimental Results

◆ To evaluate our proposed approach, 10 representative loop benchmarks with different sizes, different data dependencies and different operator types are taken from Spec2006, PolyBench and MiBench;

◆ And the power and delay analysis are synthesized with Synopsys Designer Compiler and TSMC 65-nm library; The working frequency is 500MHz; Three voltages, VddH=1.1V, VddL1=0.9V and VddL2=0.7V, are available for every PE according to the operations' delays;

◆ To evaluate our joint mapping approach, we design 4 other loops mapping and multi-Vdd assignment approaches for comparison:

(1) VddH-Only: only one voltage VddH is assigned during mapping [11];

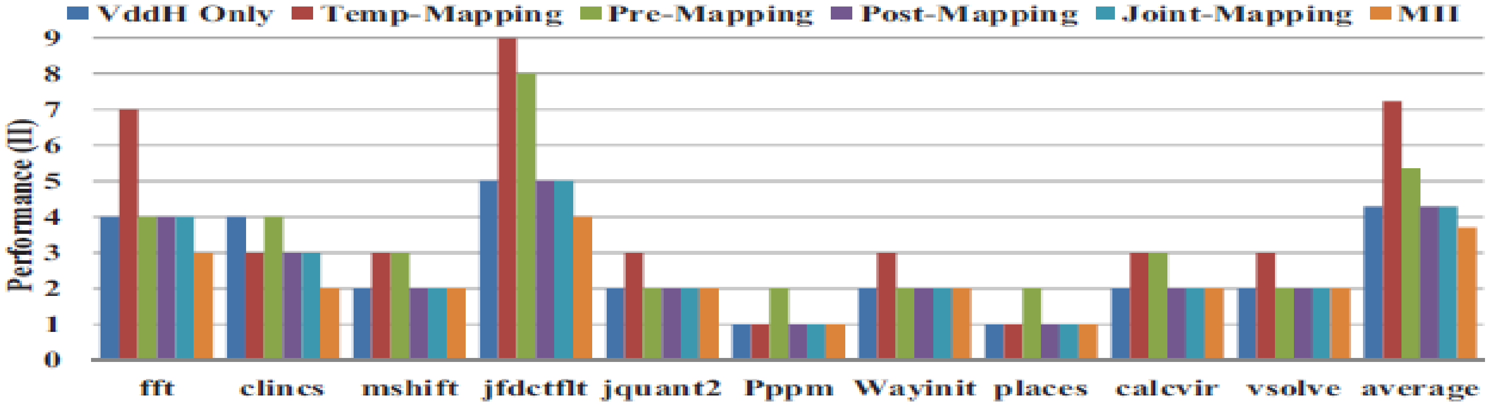
(2) Pre-Mapping: multi-Vdd assignment before mapping [4][11];

(3) Temp-Mapping: dynamical multi-Vdd assignment during mapping [6][11];

(4) Post-Mapping: multi-Vdd assignment after mapping [7][11];

Experimental Results

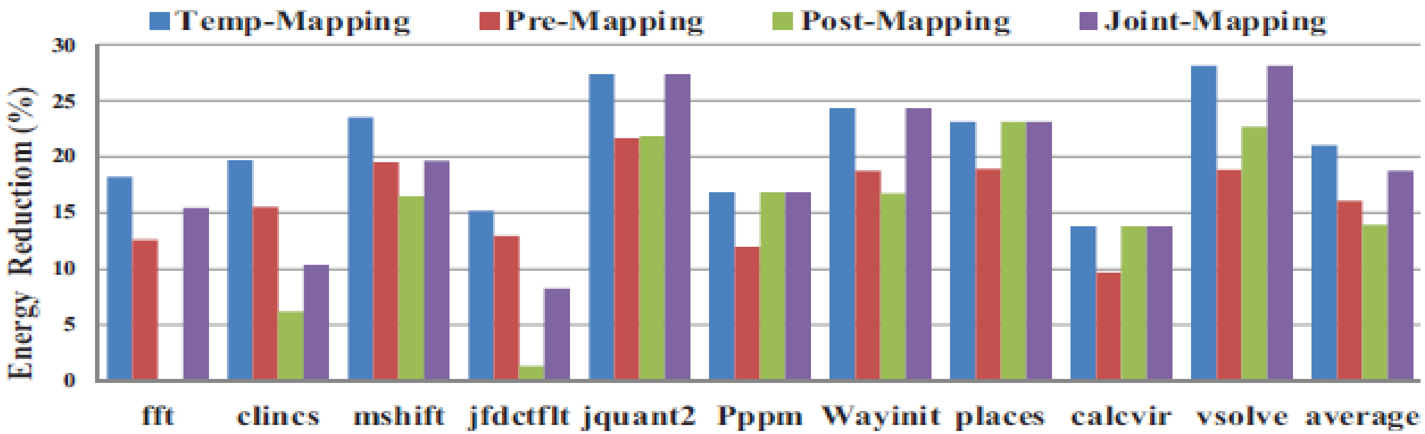
- **Performance Comparison**:
  - The ***II***s obtained by the Temp-Mapping approach are larger than those by our approach all the 10 loops;
  - The ***II***s handled by the Pre-Mapping approach are larger than those of our approach in 6 out of 10 loops;
  - The ***II***s achieved by the Post-Mapping approach are the same as those obtained by our approach among all the 10 loops;
  - Our proposed Joint-Mapping approach obtains the optimized ***II***s among all five approaches, and achieves *MII* in 7 out of 10 loops;



(a) Performance (*II*)

Experimental Results

- **Performance Comparison**:
  - The Temp-Mapping approach achieves the largest energy reduction but worse ***II***s, since it assigns Vdd for each PE each configuration context;
  - Both the Pre-Mapping and Post-Mapping approach achieves the worse energy reduction, because they are limited by either the pre-defined Vdd pattern or the blindly mapping results;
  - Our Joint-Mapping approach obtains an average energy reduction of 18.7%, improved by 16.7%, 34.7% compared to pre-mapping approach and post-mapping approach, respectively.



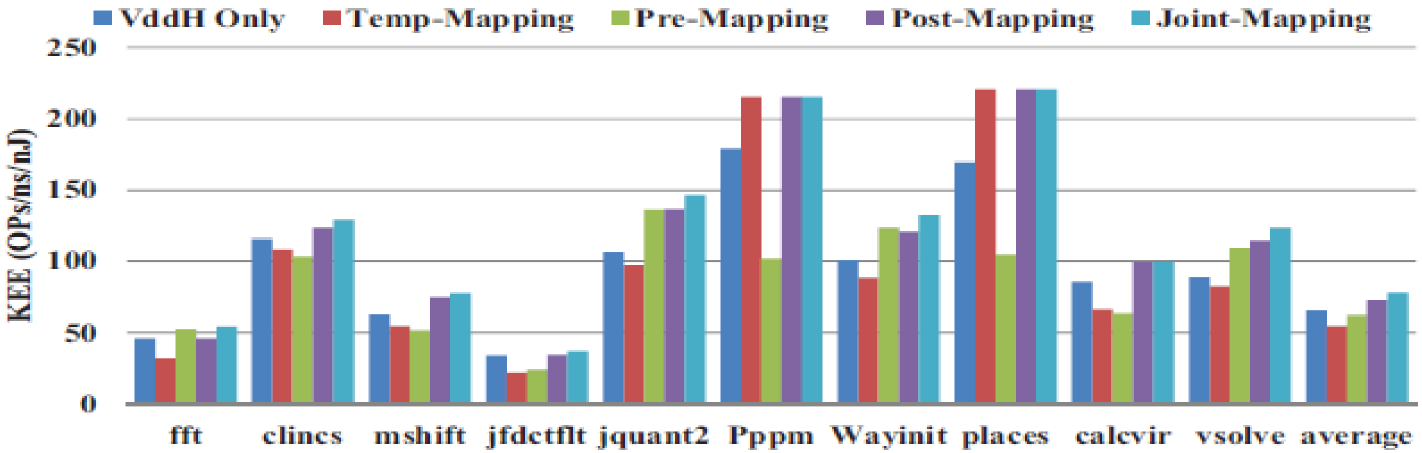
(b) Energy Reduction (%)

Experimental Results

- **Energy Efficiency Comparison**:
  - We define the kernel energy efficiency (KEE) as Equation 2 below. The unit of it is the number of executed Ops per *ns* per *nJ* (OPs/ns/nJ).

$$\mathrm{KEE} = \frac{\text{The Number of Kernel OPs}}{II \times \text{Cycle} \times \text{Kernel Energy}} \qquad (2)$$

  - Our joint-mapping approach outperforms other four mapping approaches and achieve an average KEE of 78.28 OPs/ns/nJ, which improves average 20%, 14%X, 15% and 8%, respectively, as compared to the VddH-Only Mapping, Temp-Mapping, Pre-Mapping and Post-Mapping approachesS;



(c) Kernel Energy Efficiency (OPs/ns/nJ)

Conclusion

◆ CGRA is a kind of promising architectures that can provide high performance and high power-efficiency simultaneously;

◆ Since loops usually dominate the total execution time, we focus on the mapping loop applications onto the multi-Vdd CGRAs;

◆ To achieve best performance and high energy efficiency simultaneously, we proposes a novel energy-aware loops mapping approach, which integrates the multi-Vdd assignment into the loops scheduling and mapping procedures;

◆ Experimental results indicates that our proposed joint-mapping approach can bring a 19% energy reduction and improves the kernel energy efficiency by 20% on average, while not degrading the loops performance;

Loops Mapping on Multi-Vdd CGRA

Thanks for Listening !

关注