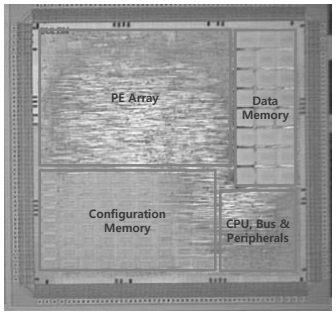
Mapping Control Flows onto CGRA

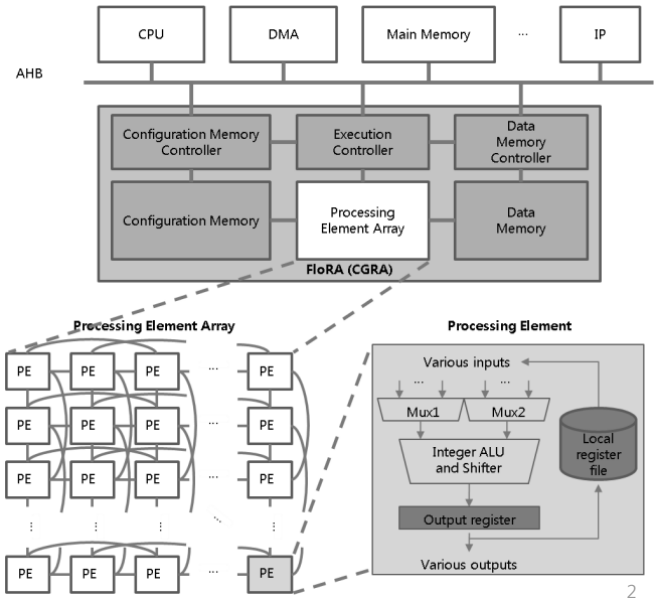
Kyuseung Han and Kiyoung Choi

Design Automation Lab
Seoul National University

Introduction

- FloRA





Introduction

- Predicated execution

```
for(i=0; i<4; i++)
{
  if(c[i]==1)
    x = 0;
  else
    x = 1;
}
```

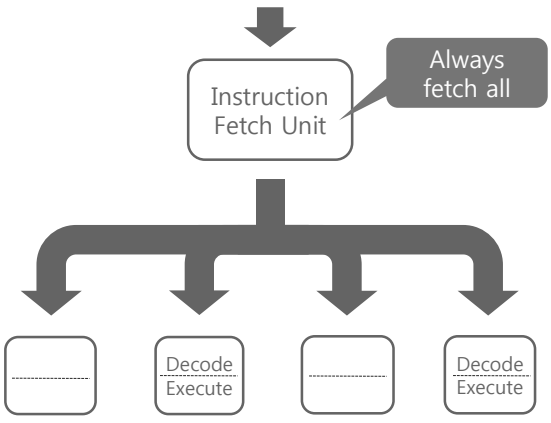
```
UC  LOAD r0, c[i]
UC  CMP r0, #1
EQ  MOV r1, #0
NE  MOV r1, #1
UC  ...
```


Introduction

- Predicated execution

```
for(i=0; i<4; i++)
{
  if(c[i]==1)
    x = 0;
  else
    x = 1;
}
```

```
UC  LOAD r0, c[i]
UC  CMP r0, #1
EQ  MOV r1, #0
NE  MOV r1, #1
UC  ...
```

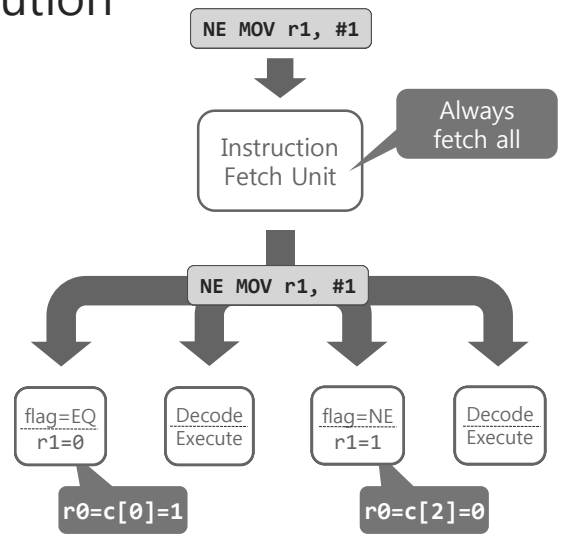


Introduction

- Predicated execution

```
for(i=0; i<4; i++)
{
  if(c[i]==1)
    x = 0;
  else
    x = 1;
}
```

```
UC  LOAD r0, c[i]
UC  CMP r0, #1
EQ  MOV r1, #0
NE  MOV r1, #1
UC  ...
```



Introduction

- Problems of predicated execution
  - Nested if-else structure
    - Most predicated execution techniques cannot handle
  - Long execution time
    - Always issue both if- and else- paths

State-Based Full Predication

- State-based Full Predication (SFP) using sleep counter (cf. SFP using tag)
- If branch is to be taken,
  - Sleep instruction is executed (conditional execution)
  - Sleep counter is set
- When sleep counter>0 (sleep state)
  - All instructions are nullified
  - Sleep counter is decremented
- When sleep counter becomes 0
  - The PE wakes up at the next cycle

Example

```
for(i=0; i<8; i++)
{
  if( cond0[i] )
    if( cond1[i] ){
      x[i] = a;
      y[i] = a;
      z[i] = a;
    }
    else{
      x[i] = b;
      y[i] = b;
    }
  else
    x[i] = c;
}
```

(a)

The loop code with nested if-else structure written in C language

```
load reg0 cond0[i]
cmp reg0 #1
b neq pc+11
load reg1 cond1[i]
cmp reg1 #1
b neq pc+5
store a x[i]
store a y[i]
store a z[i]
b uc pc+3
store b x[i]
store b y[i]
b uc pc+2
store c x[i]
```

(b)

The assembly-level pseudo code of loop body for a scalar processor

```
load reg0 cond0[i]
cmp reg0 #1
sleep neq tag0
load reg1 cond1[i]
cmp reg1 #1
sleep neq tag1
store a x[i]
store a y[i]
store a z[i]
sleep uc tag2
awake tag1
store b x[i]
store b y[i]
store a y[i]
awake tag2
sleep uc tag3
awake tag0
store c x[i]
awake tag3
```

(c)

The assembly-level pseudo code for SIMD machines when adopting tag SFP

```
load reg0 cond0[i]
cmp reg0 #1
sleep neq #9
load reg1 cond1[i]
cmp reg1 #1
sleep neq #3
store a x[i]
store a y[i]
store a z[i]
sleep uc #1
store b x[i]
store b y[i]
sleep uc #0
store c x[i]
```

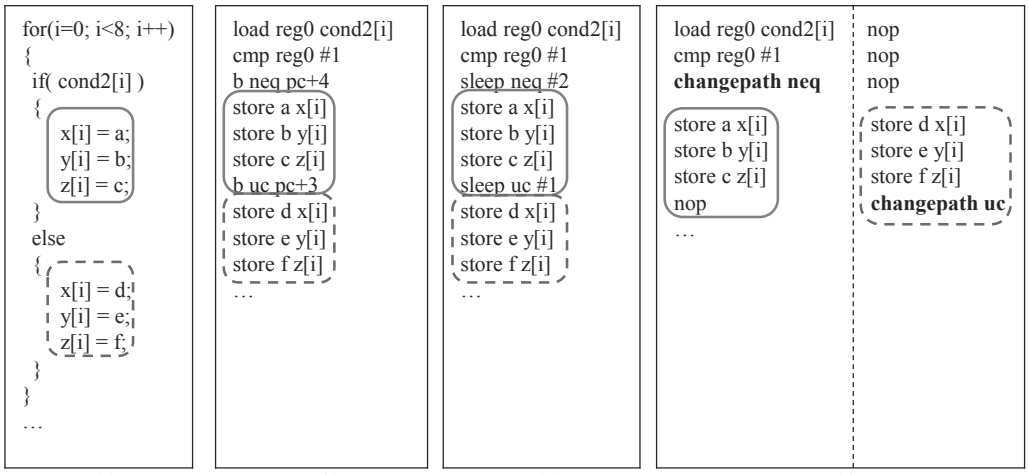
(d)

The assembly-level pseudo code for SIMD machines when adopting counter SFP

DISE

- Dual-Issue Single-Execution
- Both if-code and else-code are issued at the same time
- A PE executes only one code depending on the condition
- If-code and else-code can be overlapped to enhance performance

Example



(a) the loop code with *if-else* structure

(b) the assembly code for a scalar processor

(c) when the counter SFP is used

(d) when the DISE technique is used

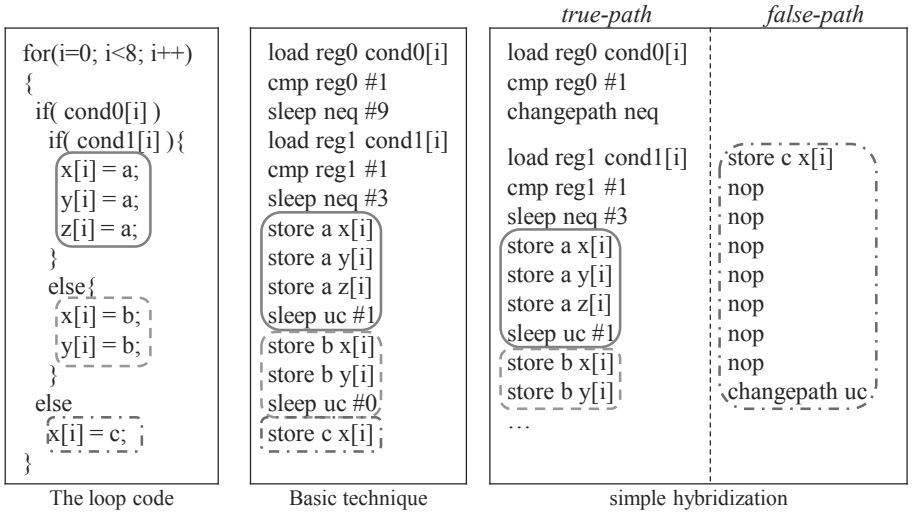
Hybridization

- Drawback of DISE
  - Cannot handle nested structure

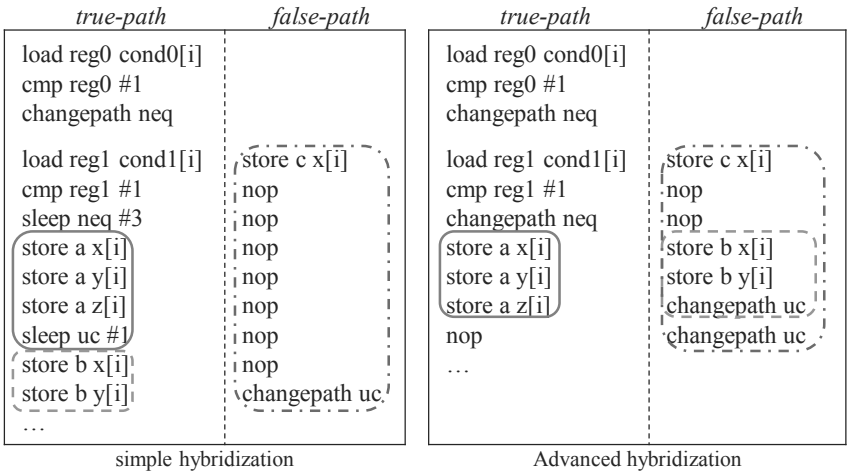
-->

- Hybridization with counter SFP
  - Simple if-else structures --> DISE
  - The rest --> counter SFP

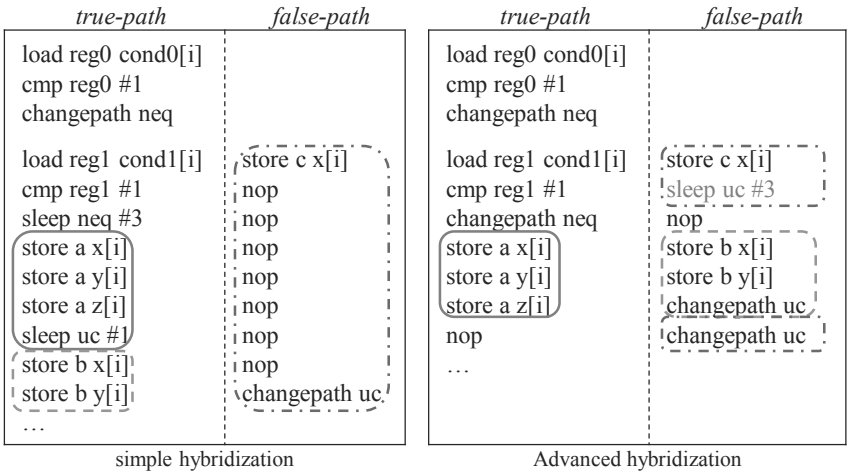
Example



Example



Example



Experimental Results

- H.264 deblocking filter

		Tag SFP (PseudoBranch)	Hybrid	Improvement
Luma Filtering	# cycles	1,040	792	24%
	code size (bytes)	2,944	2,752	7%
Chroma Filtering	# cycles	190	166	13%
	code size (bytes)	1,824	1,824	0%

- Benchmark examples
  - 13.9% delay reduction
  - 14.2% energy reduction

Summary

- Proposed Hybrid SFP technique
  - Counter SFP + DISE
  - Can handle nested control-flow structures
  - Improvement in performance, energy, and code size
    - 13.9% speedup
    - 14.2% energy reduction
  - Compiler-friendly
  - Only 2% hardware cost