

Data-Driven Multithreading

Pedro Trancoso, Skevos Evripidou

Computer Architecture Lab
University of Cyprus

Data-Driven Multithreading (DDM)

- Data-Flow is a formal and elegant model for handling concurrency
 - Functional/Side-effect free
 - Easy programmability
 - An operation is scheduled for execution only after all its input data have been produced.
 - Tolerance to Memory, Synchronization, and Network latencies

- **Program** is a set of **threads**
- **Thread**: code, data-in, data-out, consumers, producers
- **Threads scheduled** based on data availability: **Dataflow** at the thread level
- Efficient **data prefetching**
- **DDM implemented** as a **Programming model** for **Multi-core** systems:
 - **Homogeneous** such as Dual-Quad core
 - **Heterogeneous** such as the IBM CELL/BE

DDM Programming

Sequential program

```
#include <stdio.h>

int main()
{
    int i=0;
    int a[10000];

    for(i=0;i<100;i++)
    {
        a[i]=i*i+i;
    }

    printf("DONE");
}
```



User

Code with directives

```
#include <stdio.h>

int main()
{
    int i=0;
    int a[10000];

    #pragma ddm par for
    for(i=0;i<100;i++)
    {
        a[i]=i*i+i;
    }
    #pragma ddm end for

    printf("DONE");
}
```



Preprocessor

DDM Parallel Code

DDM
parallel
Code

High Level
Code

Compiler



Parallel Binary

Defines	Pragma Directive	Also Supports
Block	#pragma ddm block BLOCKID #pragma ddm endblock	Importing and exporting data needed by its threads
Thread	#pragma ddm thread THREADID kernel KERNELID #pragma ddm endthread	Shared data importing and exporting, Re-entering, and Dependence on threads and loops
Loop	#pragma ddm for thread THREADID #pragma ddm endfor	Unrolling, Reduction, and loop/loop and loop/threads dependence

DDM Execution

