FUTURE AND EMERGING TECHNOLOGIES
PROJECT N. 249013


SEVENTH FRAMEWORK PROGRAMME THEME
FET proactive 1 (ICT-2009.8.1)
Concurrent Tera-Device Computing

# TERA$^{F}$LUX

## Exploiting dataflow parallelism in Teradevice Computing

## Reliability Work Package

Theo Ungerer- University of Augsburg (UAU),
Avi Mendelson - Microsoft Israel (MSFT)

CASTNESS 2012

# Overall Objectives

- VLSI circuits for 1000 core processors will be less reliable
- Permanent, intermittent and transient faults may occur
- Techniques to establish a reliable overall multi-/many-core system out of unreliable components such as cores and interconnects.

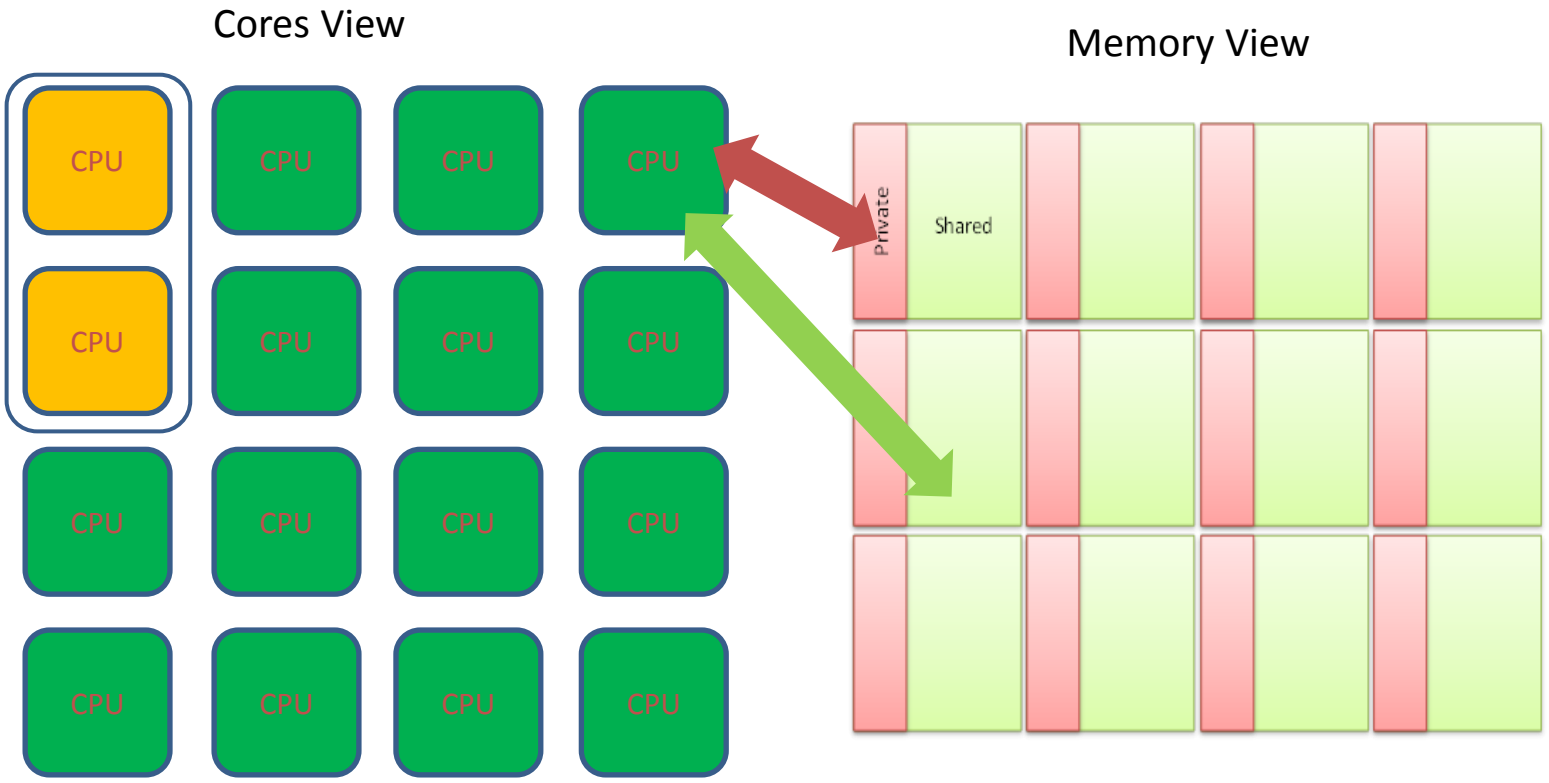
# Reliability – Overall Goal

- Fault detection and recovery mechanisms required on all levels
  - **OS level**, the system must guarantee that with high probability no observable failure (to the application) can occur.
  - **NoC level**, the system needs to be able to detect and recover from any link or router fault.
  - **Node/cluster level**, heartbeats to detect core faults within node
  - **Core level,** the system needs to be able to detect and recover from “stuck-at” or “soft-error” faults.
- Additional requirements: prevent overheating and wear-out, load balancing and power-aware scheduling

# Operating System Level

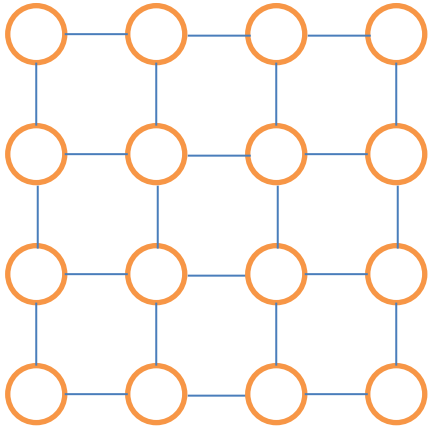
- Operating System designed as Linux OS running on the Service Node of TERAFLUX and being connected to L4 kernels running on nodes.
- We start defining the mechanisms to handle faults based on different fault models.
- Work on resource allocation and OS related issues.

# Operating System
# Low Level Core/Memory Map

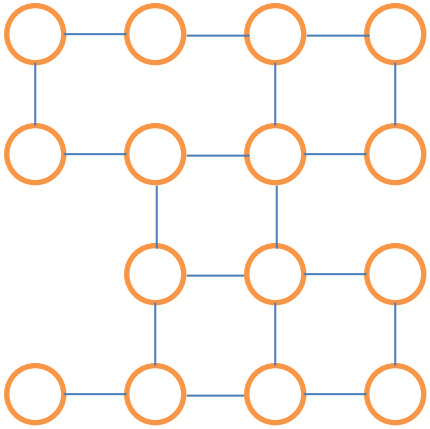



Each CPU can access it's private memory
All shared memories can be accessed (as one virtually contiguous address space) by the DMA for coping data in and out

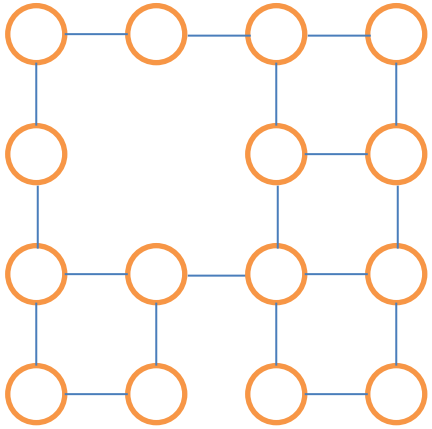
# Reliability at (Clustered) NoC Level



Basic mesh



chip1



chip2

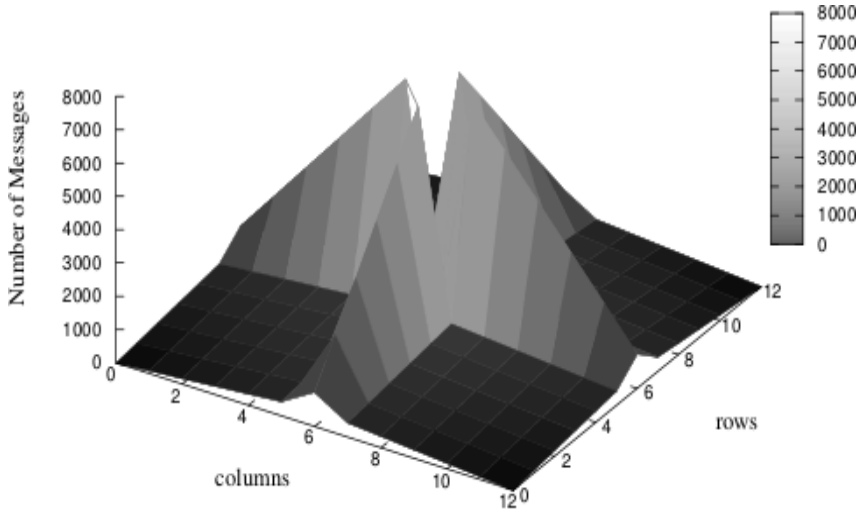
- Fault detection by Heatbeat messages to FDUs (Fault Detection Units)
- Faulty elements are avoided by re-routing using turn-model
- Target of software remains always the basic mesh
  (OS and Node Manager keep this illusion)

# Implication of Heartbeats for the NoC Level

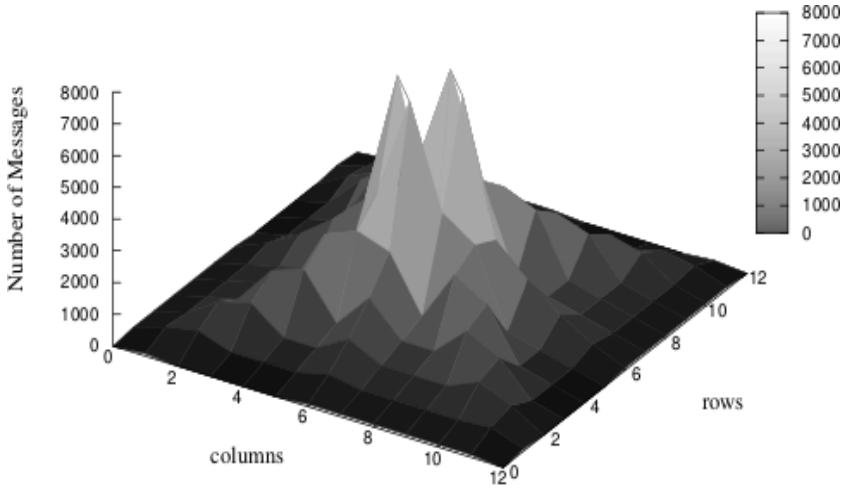
- Heartbeat based fault detection has an influence on the NoC
  - Timing constraints require heartbeat message prioritization over application messages.
  - Timing pattern of heartbeat sending to prevent heartbeat message interference.
  - Staircase vs. XY routing of heartbeats can relax the application message delays (depends on used traffic pattern).

# Core to FDU Routing at NoC Level

## Network delays induced by heartbeat messages

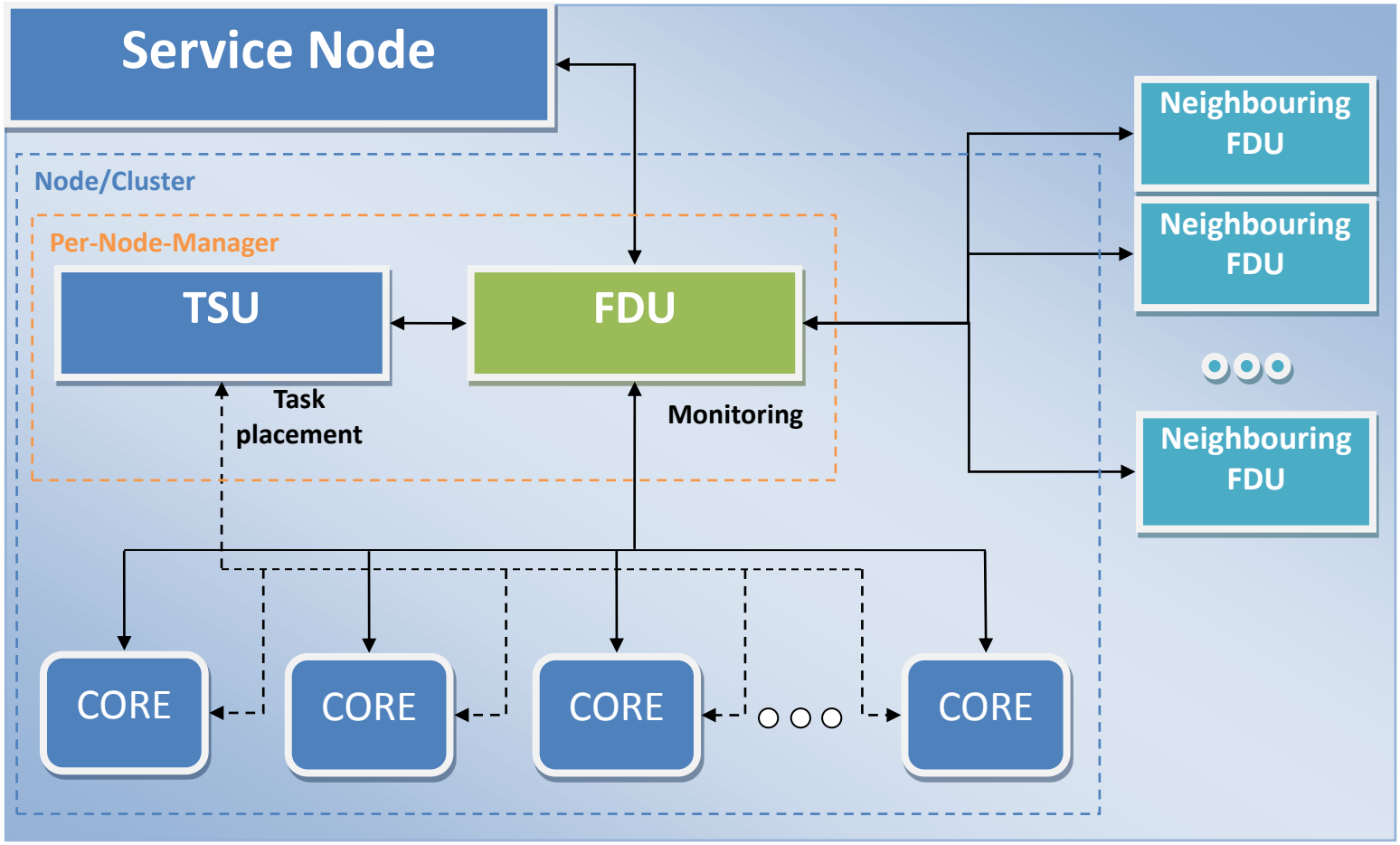



using XY routing




using Staircase routing

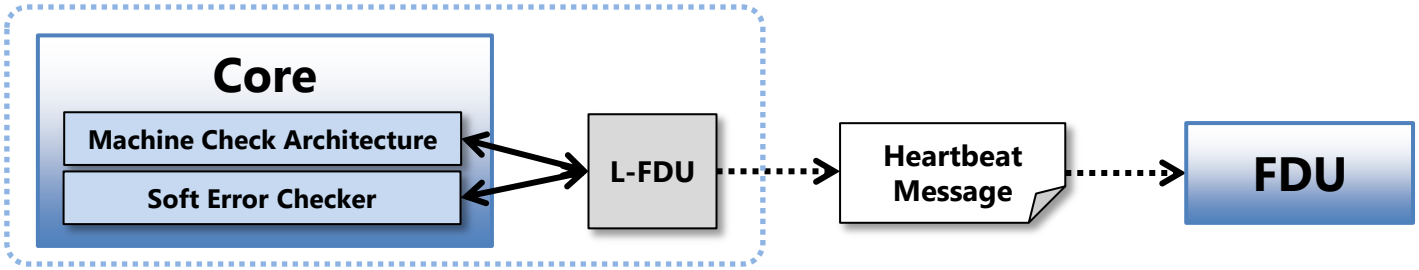
# Fault Detection Unit (FDU) Concept at Node/Cluster Level



# Fault Detection at Node/Cluster Level

- FDU functions and message formats defined and implemented
  - Heartbeat messages carry information on core fault rate, temperature, ...
- Core to FDU heartbeat communication is operational in Noxim NoC Simulator and attached to COTSon Simulator
- Algorithms for Clustering of cores to an FDU in a pure NoC-based Teraflux system
  - In a fixed node/cluster-based Teraflux system the core-FDU mapping is static
- Algorithms for mutual fault detection of faulty FDUs or whole nodes/clusters; also extended to Faulty I/O and peripheral devices

# Fault Detection at Core Level




- Machine Check Architecture (MCA) of x86-cores
  - detects and corrects certain faults in the processor logic
  - provides information on core status by registers
- Temperature information from Digital Thermal Sensor
- Registers of MCA and MSR are provided to FDU by the cores via alert/heartbeat messages
- Special Target: Soft error detection and recovery

# Soft Error Detection and Recovery

- Control flow error checking techniques
  I. Instrumentation and checker concept developed
  II. Double execution for control flow and data error detection (alternatively)
- The recovery mechanism is incorporated as part of the data-flow execution model
  - Verify that no error occurred before commit of dataflow thread
- Assume ECC for memory cells and busses
  - Not covered in this project (→ TRAMS project)

# Control Flow Error Checking

- Detect control flow errors in dataflow thread execution by temporal and logical control flow monitoring
  - Code instrumentation on basic block level


  - Checker hardware extension to cores
  - Dataflow thread execution commitment should be deferred until all checks are done.
  - Checker alerts FDU&TSU in case of time out or wrong control flow.

# Double Execution

- Detects control flow AND data errors
- Run each thread twice, once as a leading and second time as a trailing thread.
- Each execution generates signature of output results.
- At completion compare the two signatures, if consistent, the primary write its results to main memory.
- If not, no commitment and reschedule both threads.
- The duplicated threads can run on the same core or on different cores of the same node/clusters.

# Planning for 2012

- OS level:
  - Impact of the selected memory model and other alternatives of memory models on the reliability of the system
  - Scheduling and resource allocation algorithms that can efficiently handle dynamic fault conditions of the system
- NoC level:
  - Faulty link: Impact of re-routing decisions on Heartbeat timing
  - Exploiting Heartbeat timings for fault localization within the NoC
- Node/cluster level:
  - Adaption of FDU technique to Teraflux dataflow architecture
  - Dynamic adaption techniques within FDU (MAPE cycle)
  - Task distribution between FDU and TSU for fault recovery

FUTURE AND EMERGING TECHNOLOGIES
PROJECT N. 249013


SEVENTH FRAMEWORK PROGRAMME THEME
FET proactive 1 (ICT-2009.8.1)
Concurrent Tera-Device Computing

# TERA$^{F}$LUX

## Exploiting dataflow parallelism in Teradevice Computing

PROJECT NUMBER: 249013

http://teraflux.eu