SUMMARY OF THE PROJECT

1. Proposal (Title)	:	**Scheduling algorithm to minimize the worst case execution time of Safety Critical applications on Multicore processors**
2. Broad Area	:	Computer Science and Engineering
3. Duration of the grant:		3 years

4. Principal Investigator:

Name	: Dr.Mrs.P.Chitra	Designation	: Assistant Professor

Qualification	: Ph.D	Department	: CSE	Date of Birth	: 04.09.1974

5. PI's address: Dr.Mrs.P.Chitra,

Assistant Professor, CSE Department,

Thiagarajar College of Engineering, Madurai. Pin code - 625 015

Phone: 04522482240, Fax:04522483427

Email : pccse@tce.edu

5. UGC MRP details

File No	: F.No.42-130/2013(SR),26.8.2014
Date	: 15/09/2014
Amount Released	: Rs.4,93,500/-
Amount Spent	: Rs.4, 98,466/-

6. Objectives of the proposal:

- To investigate and develop a method for measuring the worst case execution time of safety critical application on Multicore processor.

- To schedule the task on the multicore processor to minimize the worst case execution time.

7. Abstract of the research:

Multi-core computer architectures are on the forefront in consumer electronics and adaptation in safety-critical applications such as avionics could be beneficial due to their potential increased performance. There are challenges to deploy cutting edge multi-core architectures for safety-critical applications. New computing architectures are more integrated and optimized for average cases. One the other side, safety critical applications need to be designed for the worst case. For example, the impact of integrating critical applications is not fully understood yet, especially with respect to execution times of critical paths. Hence, there is a need for measuring the WCET to calculate the execution times of real time

applications. There are variations that exist in the Microarchitectures of the new multicore processor and it is necessary to build a time-predictable system for critical applications on these architectures.

Network on Chip (NoC) was derived from multicore platform offers better communication between the processing elements such as Core, Cache, internal GPU,APU's, internal RAM module etc. The communication path plays a major role in estimating the execution time of the applications. Hence, the research concluded by comparing the WCET performance on 2d and 3D space on NoC.

Paper publications:

Paper title: Evaluation of Worst Case Execution Time of Tasks on Multi-core Processor

Published in: Advances in Intelligent Systems and Computing, Springer, Vol. 325, 2015

Abstract :

In hard real time systems it is required to compute the Worst Case Execution Time (WCET) of each task that has become a difficult problem. The increasing complexity of modern processor architectures makes achieving this objective more and more challenging. To measure the worst case execution time of a task, it is necessary to develop a framework on multi-core platform considering the nature of a program and shared resources. Path Analysis, one of the static based approaches is used. It is important to take into account the analysis of paths as it calculates the execution time of each basic block with respect to the cache behaviour. The WCET estimates of this analysis are performed using simplescalar simulator.

The overall design of the system for estimating the WCET is simplified and shown in Figure2. Given the task i.e., program, it is complied to get an assembly code and executable code. The executable code is disassembled using disassembler program to form an intermediate code. This intermediate code is used for representation of basic blocks. To view the process of the system Control Flow Graphs are generated. Finally, to calculate the WCET, the parameters used are basic blocks, loops, hits and misses. It should be noted that the whole process depends on the basic blocks which is represented using intermediate codes.In essence the method works in eight steps:

1. The task is given as an input. The input is in the form of C program.
2. The given C program is converted to an assembly code, object code and an executable code. These conversions are done by the SimpleScalar GCC compiler.

3. The executable code is converted to an intermediate code by using disassembler program.
4. Combining intermediate code and assembly code forms the basic blocks of a given tasks.
5. When the conversions made from the executable code to an intermediate code a TCFG is generated. Trace Control Flow Graph (TCFG) generated gives the information about the connection between the nodes.
6. With the help of TCFG, Control Flow Graphs can be formed.
7. The basic blocks obtained are analyzed using static based approach.
8. The execution time of the each basic blocks are calculated using MIPS instructions.

After getting above work done WCET calculated using the help of equation (1),(2).

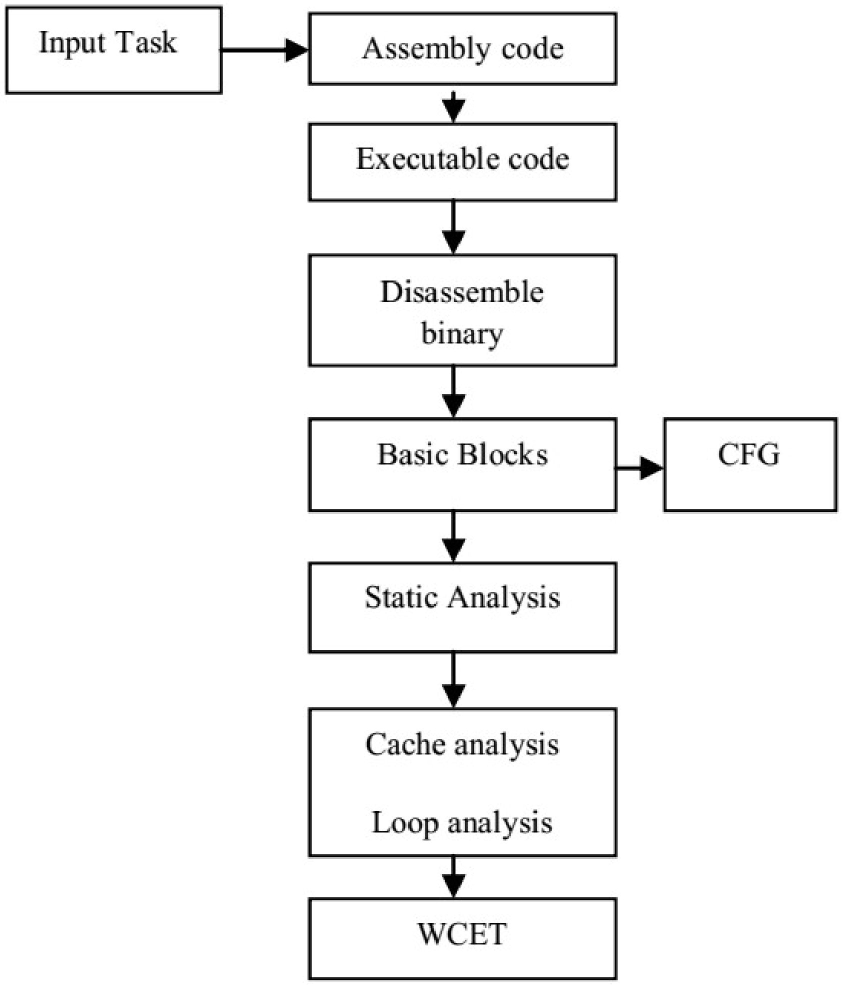

Figure 1: Overall Design of the System for WCET analysis

A. Measuring Execution Times

There are several approaches to measuring execution time, each has its own advantages and disadvantages. The two widely known approaches are using clock cycle counters and using timers.

A clock cycle counter counts clock cycles from the CPU cores. The advantages are that this approach is exact; time can be measured with the help of fine granularity. It does not disturb the execution of software in any way.

Timers can be used in many systems. This can be implemented in different ways: fully dependent in hardware or combination of software.

Real-time clocks (RTCs) are implemented with the hardware solutions, and is accessed similar as clock cycle counters, with the difference of having a lower resolution. In this paper, clock cycle counters are used.

B. Measuring Basic Blocks

To measure the total time taken for the execution of basic blocks, first it is necessary to find the cost of each basic block using equation 1. The costs of basic blocks are found using cycles. The cycles are calculated using MIPS instructions.

$$BT_i = nhit_i + nmiss_i + (\text{Total number of iterations in loop} * BC_i) \qquad (1)$$

Where i = 1,2,..........n, number of basic blocks

BT_i		total time takes for each block to execute
BC_i		Block Cost
$nhit_i$		number of hits occurred on i^{th} basic
block	-	number of misses occurred on i^{th} basic

C. Calculating WCET

To estimate Worst-Case Execution Time, sum of execution times of all the basic blocks (BT) is computed. To calculate the worst-case execution time the formula used is given in equation 2.

$$WCET = \sum_{i=1}^{n} BT_i \qquad (2)$$

The sum of all the execution time of all basic blocks is calculated by considering the iterations (if any), block cost, cache hits and misses.

To estimate the worst-case execution time of a task, static based approach is used. This approach helps to analyze the behaviour of cache and path. Simplescalar simulator is used to evaluate the precision of the analysis. Experimental results indicate that the path analysis can be used to obtain the worst case execution of a task, and thus it is preferable for real-time applications.

Paper title: Cube NoC based on Hybrid Topology

Publisher : Australian Journal of Basic and Applied Sciences, Vol. 8(15), pp. 216-225, 2014

A number of research studies have demonstrated the feasibility and advantages of Network-on-Chip (NoC) over traditional bus-based architectures such as multicore SoC. The nature of communication bottleneck come on the routing of the signals on the BUS. BUS type communication has a limitation such as single signal flow, inteference etc. So we have developed a model on NoC, NoC replaces buses with interconnect(packet transfer) by which the communication hazard was overcome and the WCET value was reduced.

Rolling into modern processor technology, developers are increasing the number of transistors exponentially. NoC is a proficient on-chip communication platform for SoC architecture; partitioning a die into segments and stacking them in 3D fashion significantly reduce latency and energy consumption. A new Cube Network-on-Chip (NoC) based architecture is proposed, which takes the advantage of this exponential increase. In this model, the number of processing elements can be increased exponentially, while reducing the space complexity. WCET estimated to meet the need of realtime process.

Why to Cube Model

Model offers regular structure and non congested floorplaning. For enterprise processing solution, increasing the processing elements linearly will not make much difference. This approach allows the exponential increase of Processing Elements (PEs) and has an effective routing strategy and offers deadlock aware minimal hop network communication, failsafe and thermal aware execution. This type of failsafe link and better routing minimize the WCET and offers a realtime task output.

The proposed cube based NoC model offers a regular structure and non-congested floor planning. For enterprise processing solution, increasing the processing elements linearly will not make much difference. This approach allows the exponential increase of Processing Elements (PEs) and has an effective routing strategy and offers deadlock aware minimal hop network communication, failsafe and thermal aware execution.

In traditional NoC architecture, the router will be a separate element and attached to a switch in PEs. In the proposed cube model, each PE is equipped with switching and routing logic. The processing element with the switching and routing logic is shown in Figure 2. The router is placed within the processing element itself. In case of a 2D NoC layer, the PE can be implemented on one of the physical planes of the system, consisting of n*n PEs in each layer. For the 3-D system, n such layers are stacked and hence the total number of PEs will be n*(n*n) which is n^3.

The router contains control logic to monitor the power and thermal impact, it also maintain an access control list which include the neighbor node's power and thermal details. As flit traffic is forwarded across the router the consumption of power is high and this further increases the thermal state of processing element that is the temperature. So to avoid the bad impact of traffic on the nodes the routing logic should be consistent over all traffic conditions.

In the Cube based NoC every element is arranged in a 3D matrix form and every elements of the processing blocks are designed and developed in the form of cubes. Fig. 2 shows the processing element cube and Fig. 3 shows the proposed 3D cube NoC architecture model. This way the final architecture of the network will have cube properties. The structure of NoC will be n*n*n matrix model, which grows in order of 3. Such cube based floor planning offers average equal length of communication path and avoids congested wiring.

The router has control logic to monitor the power and thermal impact, it also maintain a access control list which include the neighbor node power and thermal details. As flit traffic across the router increases the usage of power and subside it increase the thermal of processing element. So to avoid the bad impact of traffic the routing logic should be consistent over all sort of nature. Cube model offers a intelligent routing logic which helps to act according to the nature of traffic and failure.

Floor Planning

In Cube based NoC every element is arranged in 3D matrix form, every elements of processing block designed and developed as cubes[Figure 2,3]. Final production of NoC offers cube properties. Structure of NoC will be n*n*n matrix model, which grows in order of 3.

Processing Element with Router

In cube based model, router will be embedded with the processing element. Router has seven input and output ports with selector and arbiter. This model avoid unnecessary space and wire delay as its close to the IP of the processing Element.

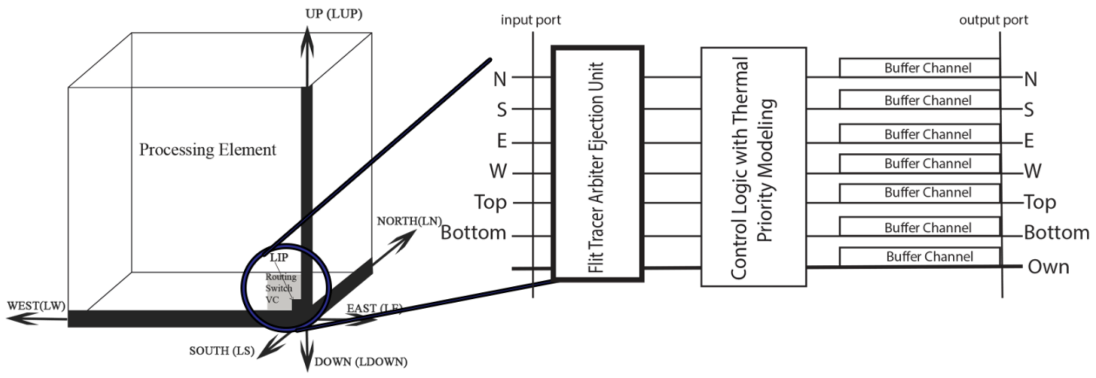

Figure 2. Processing element with Router

Topology

Topology is developed for reliable flit transfer between the PEs. The proposed topology takes the advantages of mesh with a custom topology and it tends to be a hybrid one.

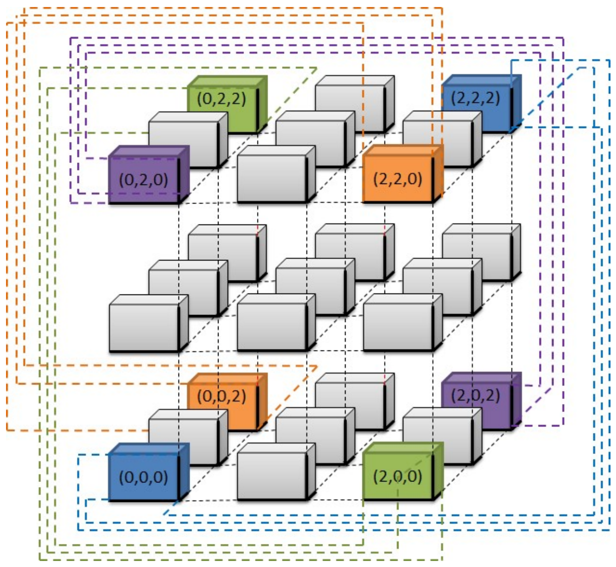

Figure. 3: illustration of 3*3*3 cube model with topology

Mesh is the primary topology over the chip and there are diagonal fail safe link outer of the model . Figure 6 shows the diagonal connection established for cube model.

The topology switching used has seven links port in each PEs(Figure 5), one to connect the IP and the router, and six other to connect to adjacent PEs within the layer or if the router is in edge then in six, three ports connect to its diagonal PE router port.

Routing Logic

Packet transfer can take place either within the same layer or across the layers. According to the Figure 5 there are seven links denoted by LIP, LE, LW, LN, LS, LUP, and LDOWN. The routing algorithm has to determine the communication is within the same layer or up/down layer.

LIP – link between the IP and the switching port.

LE, LW, LN, LS - act as the port for communicating within the layer.

LUP, LDOWN - act as the port for communicating among the layers.

LE,LW,LN,LS,LUP,LDOWN- also used to communicate with diagonal nodes

While doing the routing the WCET value was calculated and the routing decided to go through the mesh or to the failsafe link.

XY routing is used, if the communication is within the same layer. If it's to other layer, then the routing logic determines the shortest failsafe link and completes the transfer considering the traffic and thermal impact.

In moderated NoC environment due to flit traffic, power and thermal issues, router in PEs may fail to carry on the flit transfer across network. The proposed model can operate without interruption even under such situations. As it has diagonal link between bottom layer corner to top layer corner (Figure 6) even their some failures in intermediate node over mesh interconnect, diagonal link act as a failsafe communication for the model and offers many dedicated paths to proceed the operation. Let us have 'n' nodes in L number of layers.

In Cube model, 4 PE's have failsafe diagonal links

$$L(0,0,0) - L(n,n,n) , \; L(0,0,n) - L(n,0,0) , L(0,n,n) - L(n,0,0) \text{ and } L(0,n,0) - L(n,n,0)$$

Each corner PE contribute 3 diagonal links over the input and output port. Considering our 3*3*3 model(Figure 6), model exhibits 12 (4 PE*3 link) failsafe links for flit transfer.

If the local IP core at the bottom layer, wants to communicate to the IP core at the top layer, it does not need do a multiple hop by jumping layer by layer, instead jump to the top via diagonal link and do the needful action. For packets routed across layers, the router makes routing decision based on flit traffic and latency on port. When a packet is to be delivered between the routers in the same layer then, the router uses XY routing and finds the minimal hop path between them with the WCET.

If the source and destination are in the different layer, then uses the hybrid cube routing for efficient transfer of flit. If the up/down link of the router is busy or not accessible, it will try to find an intermediate free route with a healthy vertical link at the same layer, which has a minimal distance to the destination, based on the routing table and the WCET. An example for the routing is given below.

3D NoC technology reduces the interconnect delays by stacking multiple layers on top of each by providing shorter vertical interconnect. A new Cube based NoC model is proposed, that provide shorter interconnects with hybrid topology. This also offers better scalability, with respect to processing elements. The interconnection proposed here, offers failsafe communication through the shortest link via optimized routing logic which helps to reduce the WCET value to negligible.