Journal of Engineering and Development, Vol. ١٧, No.١, Mar. ٢٠١٣, ISSN ١٨١٣- ٧٨٢٢

Parallel Processing-Parallel Memory Approach for Super Fast Design of Future Microprocessor

Yaakob Karomy. Hanna
Al-Mustansiriyah University, College of Engineering
Department of Computer & Software Engineering
Baghdad, Iraq

Abstract

The early design of the microprocessor (μP) used a single ALU with a single unit of memory. The development of the microprocessor design generates a multi-ALUs microprocessor that is a parallel processing with multi-units of memory.

The parallel processing approach will increase the speed of the processing but this speed up is non-linear with increasing the number of processors that are used in the system (μP). However, the efficiency of the parallel processing is non-linear and depends on some factors such as the parallel processing type, the overall design, the programming approach and the applications, yet in general the parallel processing efficiency will decrease by the increase of the number of processors in the system.

The history of the Intel μP's will be used as an example to trace and analyze the growth of the μP. This tracing will disclose that the main future problem in the μP is the storage not the processing of the data. This problem is generated because the shared memory in the parallel processing will capture the processors in the system. The processor in this parallel processing system is not free to use the memory but it shares a single memory with other processors in this system.

This paper propose a novel approach designing a parallel memory that gives the processors in the parallel processing system a higher freedom to use the memory and eventually increases the efficiency of each processor, that end result will increase the total speed of the parallel processing system because it will become a parallel processing parallel memory (PPPM) system.

This approach will apply to the Intel processor P٤, which will show that it is able to increase the speed of the P٤ processor for more than four times. These results are used to propose a future design strategy as a first step to implement a super fast processor and then a super fast PC.

The proposed processor is PPPM system with ٢٥٦ ALUs, this processor is expected to enhance the strategy of the management and control units to become a successful super fast processor with speed up to ٦٠ times over the Intel P٤.

Keyword: *Intel μP, parallel processing, parallel memory, cache memory, DMA, memory cycle, P ٤, prediction, fetching, Net Burst, shared memory, delay, latency.*

الخلاصة

التصاميم الاولية للمعالجات الدقيقة استخدمت وحدة واحدة فقط من وحدات الحساب والمنطق (ALU) مع ذاكرة واحدة. التطوير في تصميم المعالج الدقيق خلق معالجات دقيقة بعدة وحدات من الـ (ALU) تعمل بصورة متوازية مع بعضها البعض مع استخدام عدة وحدات من الذاكرة (multi-units of memory).

طريقة المعالجة المتوازية تزيد من سرعة المعالجة ولكن هذه الزيادة لاتكون زيادة خطية مع الزيادة في عدد المعالجات المستخدمة في النظام. ان كفاءة نظام المعالجة المتوازية ليست خطية بسبب اعتمادها على عوامل اخرى مثل ، نوع المعالجة المتوازية، التصميم النهائي للنظام، نوع البرمجيات مع التطبيقات،لكن بصورة عامة ان كفاءة نظام المعاجة المتوازية تقل كلما ازداد عدد المعالجات في النظام.

ان تاريخ تطور المعالجات من نوع (Intel) ستستخدم كمثال لبحث وتحليل نمووتطور المعالج الدقيق. هذا البحث سيكشف ان المشكلة المستقبلية الرئيسية في المعالج الدقيق ستكون في خزن المعلومات وليس في معالجتها. هذه المشكلة تظهر بسبب ان الذاكرة المشتركة في نظام المعالجة المتوازية ستحدد عمل المعالجات في النظام حيث ان المعالج في هذا النظام سيكون ليس حرا في استخدام الذاكرة ولكنه سيتقاسم تلك الذاكرة مع المعالجات الاخرى في النظام.

البحث يقترح نهجا جديدا لتصميم الذاكرة المتوازية التي تعطي المعالجات في نظام المعالجة المتوازية حرية اكبر في استخدام الذاكرة مما يزيد من كفاءة كل معالج ، ونتيجة لذلك ستزداد السرعة الكلية للنظام بسبب نظام المعالجة المتوازية والذاكرة المتوازية (PPPM).

هذا النهج سيطبق على المعالج الدقيق نوع (Intel P ٤)، والذي سوف يزيد سرعة المعالج P ٤ لأكثر من أربع مرات. هذه النتائج تستخدم لاقتراح استراتيجية تصميم مستقبلية كخطوة أولى لتنفيذ معالج بسرعة فائقة ومن ثم حاسوب شخصي (PC) بمثل هذه السرعة.

المعالج المقترح يستخدم نظام معالجة متوازية مع ذاكرة متوازية (PPPM) بـ ٢٥٦ ALUs وحدة حساب ومنطق، ومن المتوقع ان هذا المعالج سيعزز استراتيجية وحدات الادارة والسيطرة لتصبح بنجاح معالج فائق السرعة تفوق سرعة معالج (Intel P ٤) بـ (٦٠) مرة.

١. Introduction

What are the requirements to implement a fast PC? The basic answer is the implementation of a fast PC requires a fast μP; but this answer is not complete because the fast μP requires to a suitable components to help it. The fast PC required at least to:

١. Fast μP.
٢. Large size of memory with suitable heretically implementation.
٣. Flexible I/O controller.

This answer arise a new question, what is the fast μP? It's a μP that has at least:

١. Fast clock.
٢. Wide data bus (ALU).
٣. Wide address bus.

٤. Suitable control signals.
٥. Large set of instructions.
٦. Good control circuits.

The application of all the above points in the design of the μP is not enough to implement a fast μP; the key of the next step to increase the μP speed is the parallel processing with extra help units such as fetching unit, MMX, SSE, Net Burst,….. [١]

This paper will trace the steps of the processor growing from the history of the Intel processors Section ٢.

Section ٣ shows the types of the parallel processing while the efficiency measurement of the shared parallel processing system will be summarized in section ٤.

Section ٥ shows the design of the P٤ processor as suitable example for Intel μP that uses the parallel processing approach.

Section ٦ will give the key of the novel method of the parallel memory, while the effect of the PPPM approach will present in examples in section ٧ that will estimate the time delays using a simple basic equations then simulate the simple parallel processing system for much better results.

Section ٨ gives the main lines to design a super processor in the future, the conclusions and a simple future work of this paper are in sections ٩ and ١٠.

٢. Growing History of Intel microprocessors

The history of the Intel microprocessors will be summarized in tables (١, ٢ and ٣); these tables show how the growing of the μP is joined with the other factors in the PC. The light points for this history are noted in bold lines in the tables like the use of DMA in ٨٠٠٨ μP.

The first row in table (١) shows the ٤٠٠٤ μP in ١٩٧١ which is the first general IC (the early form of μP) used to implement micro-controllers such as the simple calculator. this IC mange ٦٤٠ Byte of memory and uses the ٤٠٠٩ IC as I/O manager. The expansion of the memory size to ١٦ KB in ٨٠٠٨ will require a special memory management and I/O terminals. This unit is called Direct Memory Access (DMA). The ٨٠٠٨ IC has the same architecture of the ٤٠٠٤ IC with twice number of bytes for data bus and address bus. The ١٦ bit address bus was presented in ٨٠٨٠ that expands the memory to ٦٤ KB. The ٨٠٨٥ in ١٩٧٦ is a glaring point in the Intel processor history; this IC is a real μP that has a complex set of instructions ١٦ bit address bus with new input (maskable interrupt) controller, so it has three internally data-buses with all necessary registers set. The weakness in this μP is the limited memory (٦٤ KB) that is not suitable to save the real programs and the data for the real problems.

Expansion of the address bus to ٢٠ bit expands the memory size to ١ MB. The new size for memory requires a large size memory word and registers to capture the addresses of the instructions and the data in the memory, this problem is solved by using the *segmentation* approach. The electronic growth helps to implement fast and large ICs, the designer use this

advantage to increase the data bus to ١٦ bit, and hence to implement the first element for the successful family of processor that called ٨٠٨٦ in ١٩٧٨. The ٨٠٨٦ µP has speed up to ٨ MHz, maskable interrupt and management ١ MB memory will be used to realize the dream to implement the early PC that called IBM PS/٢.

Table (١). The early generations Intel µPs.

Year	µP	Data bus	Address bus	Clock (Hz)	MIPS	Memory (Byte)	I/O	Used
١٩٧١	٤٠٠٤	٤	٤	**٧٤٠K**	٠,٠٧	٦٤٠	٤٠٠٩	Calculator
١٩٧١	٤٠٤٠	٤	٤	٧٤٠K	٠,٠٧	١٠٢٤	٤٢٠٧	Calculator
١٩٧٢	٨٠٠٨	٨	٨	٥٠٠K	٠,٠٥	١٦K	**DMA**	-----
١٩٧٤	٨٠٨٠	٨	١٦	٢M	٠,٦٤	٦٤K	DMA	Traffic light
١٩٧٦	٨٠٨٥	٨	١٦	٢M	٠,٣٧	٦٤K	**maskable interrupt**	
١٩٧٨	٨٠٨٦	١٦	٢٠	٨M	٠,٦٦	**١M segmentation**	maskable interrupt	IBM PS/٢

The implementation of µP which is suitable to implement the practical PCs starts with the ٨٠١٨٦ µP in ١٩٨٢ as in table (٢). The real challenge in that time was the software manager not the hardware, this problem remained about ٣ years then the first real software manger for the PC that called Microsoft Software Disk Operating System (MSDOS) has been born, this software with the DMA and maskable interrupt input built a successful strategy for PC, that will be growing with the µP later.

The ٨٠٢٨٦ µP that has ٢٤ bit in address bus and ١٦ MB memory with MSDOS is able to implement a useful PC that can be used for long time. The new problem in that time was the limited memory size in comparison with the requirements for solved problems in PC. The solution comes with ٨٠٣٨٦DX in ١٩٨٥ that had ٣٢ bit for data bus and ٣٢ bits for address bus to expand the memory to ٤ GB with additional feature called *virtual memory* that expanded the external memory of the PC (hard disk) to ٦٤ TB.

The large memory size in ٨٠٣٨٦DX required a long read and write times. this problem was solved by adding a small (٨ KB) intermediate memory called *cache* memory in ٨٠٤٨٦DX.

The Pentium technology opens a new gate to implement a very large and very high speed ICs (٣,١ million transistors with ٢٧٣ pin), the data bus became ٦٤ bit with ١٦ KB cache in the first generation of these processors which called P٥ or PI. This processor with all the next generations uses two new technologies, the first is the *prediction* unit which represents the early class of the parallel processing, while the second is the imbedded processor to increase the speed independent of the memory and I/O terminals manger. The new technology (Pentium) with the

two new features in the processor gives the PC a high speed that makes the PCs able to run the advances operating systems (AOS) programs such as windows.

The ٢ ALUs with two levels cache memory in Pentium Processor represent the early attempt to use the *parallel* processing approach that increases the power of processor. The additional important unit emerged in ١٩٩٧ with PII used MMX internally unit that made the processor able to run a large set of instructions that called MMX set instruction, therefore sometimes this processor is called Pentium MMX. The PIII expands to run a new set of instructions called SSE also it used ٤ ALUs.

Table (٢). The practical generations Intel µPs for PC.

Year	µP	Data bus	Address bus	Clock (Hz)	Cache (Byte)	Memory (Byte)	I/O	Other factors
١٩٨٢	٨٠١٨٦	١٦	٢٠	٨M	no	١ M seg.	**DMA Interrupt**	**MSDOS**
١٩٨٢	٨٠٢٨٦	١٦	**٢٤**	١٠M	no	١٦ M seg.	DMA Interrupt	MSDOS
١٩٨٥	٨٠٣٨٦DX	**٣٢**	**٣٢**	١٦M	no	**٤ G virtual ٦٤ T paged**	DMA Interrupt	MSDOS
١٩٨٩	٨٠٤٨٦DX	٣٢	٣٢	٢٥M	**L١ ٨K**	٤ G ٦٤ T	DMA Interrupt	MSDOS
١٩٩٣	P٥ (PI)	**٦٤**	٣٢	٦٠M	L١ ١٦K	٤ G ٦٤ T	**Imbedded processor**	**Predict +AOS**
١٩٩٥	Pentium Processor	٦٤	٣٢	١٥٠M	**L١ ١٦K L٢ ٢٥٦K**	٤ G ٦٤ T	Imbedded processor	**٢ ALUs**
١٩٩٧	PII	٦٤	٣٢	٣٠٠M	L١ ٣٢K L٢ ٥١٢K	٤ G ٦٤ T	Imbedded processor	**MMX**
١٩٩٩	PIII	٦٤	٣٢	٥٠٠M	L١ ٢٥٦K L٢ ٢M	٤ G ٦٤ T	Imbedded processor	**٤ ALUs SSE**

The ALUs in Pentium Processor., PII and PIII are half parallel processing because they are connected as a parallel input serial output, the real parallel processing starts with *Net Burst* in P٤ as in table (٣) it has ٤ ALUs work as parallel input and parallel output units.

Table (٣) shows how the micro-architecture of the classic µP reaches saturation and the improve start to use the other factors as in the last column in table (٣). P٤ satisfies the parallel processing using Net Burst and enhances the set of instructions using *SSE/SSE* ٢ sets.

Itanum ٢ (Dual core) grouped two P٤ µPs internally and expands its instruction set to new set called SSE٣. The growing generates a new technology called Intel ٦٤ that is used to implement the Intel Core ٢ type Kentfield which has two dual cores internally and expands it to the Intel Core ٢ type Yorkfield that has ٦ cores internally with additional instructions set called SSSE٣.

Core i٥ technology (Nehalem type Lynnfield as example) used ٤- physical cores with ٢-channel DDR٣ memory also its designer separates the cache memory L١ to two parts one for data

and the other for instructions, expand L٢ to ٤ sections each one is ٢٥٦ KB adding the third common cache level L٣ that has ٨ MB capacity.

The Dual core, Two Dual core, Core ٢ Quad, ٤-physical cores with ٢-channel DDR٣ and in the last the ٦-physical cores with ٤-channel DDR٣ and the *Turbo Boost Technology* in Pentium Core™ i٧ (Westmere type Gulftown) µP strategies in March ٢٠١٠ enhanced the parallel processing but all the new technologies will not add high features for the total power of the µP because the limitations of the parallel processing that will be discussed later. For more detail about the history of microprocessor see Ref [١-٣].

Table (٣). The modern generations of Intel µPs.

Year	µP	Data bus	Address bus	Clock (Hz)	Cache (Byte)	Memory (Byte)	Other factors
٢٠٠٢	P٤	٦٤	٣٢	٢,٤G	L١ ١٦ K L٢ ٥١٢ K	٤ G ٦٤ T	**Net Burst** **SSE/SSE٢**
٢٠٠٢	Itanum ٢	٦٤	٣٢	٢,٤G	L١ ١٦ K L٢ ٤ M	٤ G ٦٤ T	**Dual core** **SSE٣**
٢٠٠٦	Intel ٦٤, Intel Core ٢, Kentfield	٦٤	٣٢	٣G	L١ ١٦ K L٢ ٢x٤ M	٤ G ٦٤ T	**Two Dual Core** **SSSE٣**
٢٠٠٨	Intel ٦٤, Intel Core ٢, Yorkfield	٦٤	٣٢	٣G	L١ ٣٢K L٢ ٢x٦M	٤ G ٦٤ T	**Core ٢ Quad** **SSSE٣**
٢٠٠٩	Nehalem , Core i٥, Lynnfield	٦٤	٣٢	٣,٢G	**L١ ٤x٣٢+٣٢ K** **L٢ ٤x٢٥٦ K** **L٣ ٨ M**	٤ G ٦٤ T	**٤- physical cores** **٢-channel** **DDR٣**
٢٠١٠	Westmere, Core™ i٧, Gulftown	٦٤	٣٢	**٦,٤G**	**L١ ٤x٣٢+٣٢ K** **L٢ ٨x٢٥٦ K** **L٣ ٢٤ M**	٤ G ٦٤ T	**٦- physical cores** **٤ × DDR٣** **Turbo Boost Technology**

٣. Parallel Processing

There are two main types of the parallel processing, each one has advantages and disadvantages over the other type. These two types are the Distributed Memory and the Shared Memory. Also these types are improved to become hybrid to generate a large amount of parallel processing systems. For more detail about the parallel processing see Ref [٤, ٥].

٣.١. Distributed Memory

Distributed memory systems vary widely but share a common characteristic. It requires a communication network to connect inter-processor memory as in Fig (١).

The processors have their own local memory. Memory addresses in one processor do not map to another processor, so there is no concept of global address space across all processors. Because each processor has its own local memory, it operates independently. Changes that made to any local memory have no effect on the memory of other processors. Hence, the concept of cache coherency does not apply. [٤]

When a processor needs access to a data in another processor, it is usually the task of the programmer to explicitly define how and when data is communicated. Synchronization between tasks is likewise the programmer's responsibility. The network "fabric" used for data transfer varies widely, though it can be as simple as Ethernet.

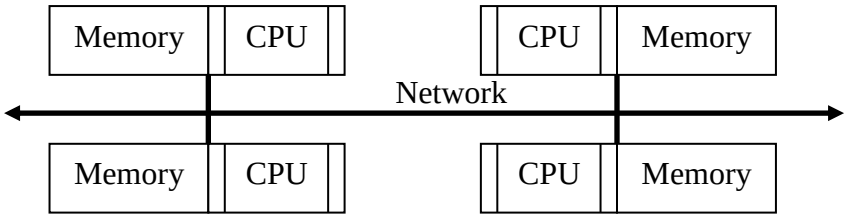

Fig (١): Distributed memory systems parallel processing.

The advantage of this method is each processor can rapidly access its own memory without interference and without the overhead incurred with trying to maintain cache coherency. While the disadvantage of this method, it is used for networks and for independent processors. [٤]

٣,٢. Shared Memory

The shared memory parallel computers Fig (٢) vary widely, but generally have in common the ability for all processors to access all memory as global address space. Multiple processors can operate independently but share the same memory resources. Changes in a memory location effected by one processor are visible to all other processors.

The advantages of this method are Global address space provides a user-friendly programming perspective to memory and the data Sharing between tasks is both fast and uniform due to the proximity of memory to CPUs.

The disadvantages of this method are the lack of scalability between memory and CPUs. Adding more CPUs can geometrically increase traffic on the shared memory-CPU path and for cache coherent systems and geometrically increase traffic associated with cache/memory management, programmer responsibility for synchronization constructs that insure "correct" access of global memory. [٤]

This paper will use the shared memory technique for all the next sections and call for simplicity “parallel processing”.

٣,٣. Multi Levels Parallel Processing

The shared memory method is active for a small number of processors (max. ٨ or ١٦ processors). The system with a large number of processors will use a multi levels parallel processing as shown in Fig (٣). The total result is like a parallel inside parallel processing. [٥]

The number of processors in each group is m; this number in default is equal for all groups because that gives optimal results and simple control. These (n) groups will be connected to a common memory in form of shared memory, this system is indicated to as m/n. Theoretically there is no condition that limits the number of levels or the number of groups in each level or the number of processors in the higher level, but practically the symmetry saves the best case.

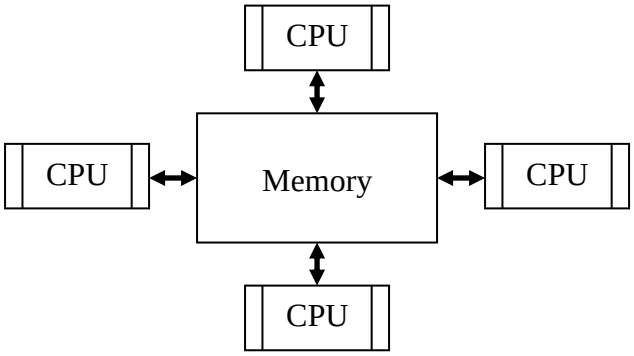

Fig (٢): Shared memory systems parallel processing.

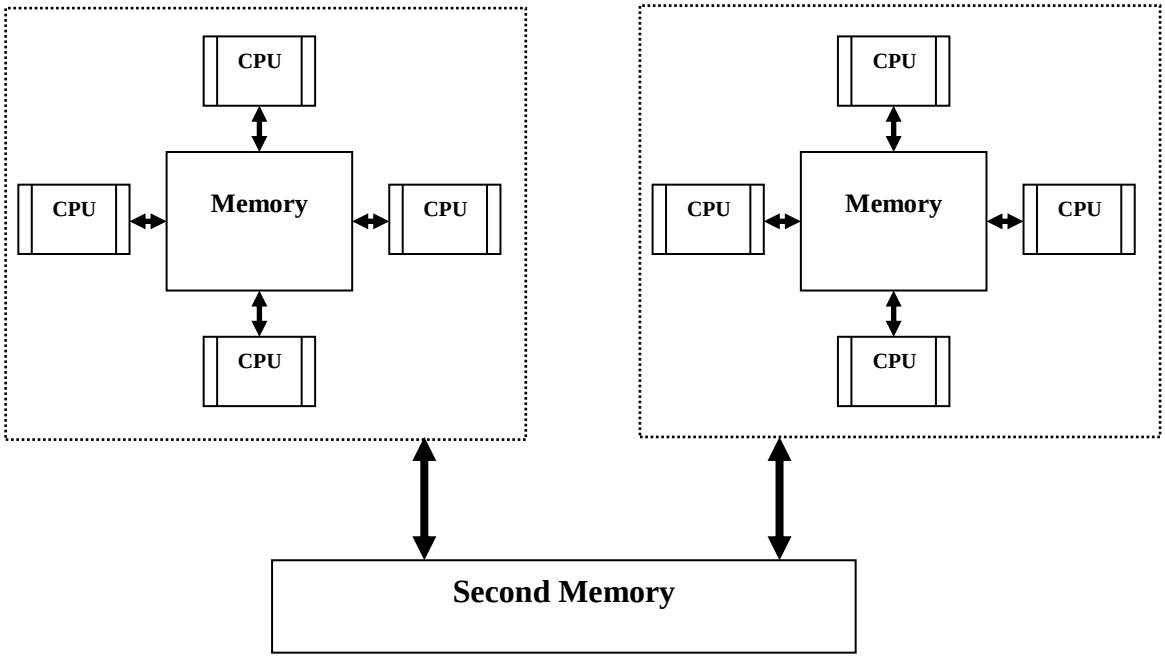

Fig (٣): Two levels (٤/٢) shared memory parallel processing system.

٣,٤. Virtual Processor

Some types of processors such as P٤ use multiple levels of memories (L١ and L٢) to interchange the data. Hence, the relation between them is not memory to memory but it is like the

relations between processor and the memory, therefore L٢ will regard the L١ as a processor; hence, the memory unit L١ in this case will be called a *virtual processor*.

٤. Efficiency Measurement of the Parallel Processing System

There are many factors that affect the efficiency of the parallel processing system like the number of processors in the system, the distribution form of these processors, the design strategy, type of connection, the processor characteristics, the ratio of used memory, memory speed, the size of the memory and other different factors that depend on the system and problems. [٦, ٧]

However, for simplicity let:

١. All the processors are of the same type.

٢. All processors and memory have same speed (equal clocks time).

٣. All processors have same memory used ratio (P_{mi}).

٤. The value of P_{mi} is less than ١/n where n is the number of processors in the system.

٥. Neglect all other factors.

Under these conditions the total speed of the system can be written as in the following equation (١). [٦, ٧]

$$S_s \le S_p \times \frac{n+1}{2} \qquad if \ \sum P_{mi} \le 1 \qquad (١)$$

Where S_s is the speed of the system, Sp is the speed of a single processor.

Define a new factor Rs as the maximum rational system speed that is given in equation (٢) as:

$$R_s = \frac{n+1}{2} \qquad if \ \sum P_{mi} \le 1 \qquad (٢)$$

The solutions for equation (١) show that the use of two parallel processors will give a system of a single processor with virtual speed less than ١,٥ times of the original speed of a single processor (S_p). Table (٤) shows some results of a simple simulation (enhanced with some practical results) for low rate data systems (P_{mi} < ٠,١).

From table (٤) we can note that, the efficient case is the system of two parallel processors with low traffic rate in memory. Another important note is the increase of the number of parallel processor will decrease the efficiency of the system for all cases and forms. The maximum practical number of processor is ١٦ processor and maximum acceptable levels is ٣ levels. So any specifications over this range will reduce the efficiency of the processor to not useful range.

The practical results are lower than these values except for some special problems that accept the parallel computation. For example the efficiency of a ٢ parallel processors full in rang from ٧٠٪ to ٨٥% and the efficiency of a ٤ parallel processors full in rang from ٤٥٪ to ٦٠%. [٧]

٥. P٤ Intel Processor

The designs of Intel processors are very different as we noted in section ٢, which means that there is no common design for these processors. Hence, we will select the design of the P٤ processor as a good example to show the simple design of a modern parallel processor.

The architecture of P٤ and its working strategy is very complex but we will simplify the main important points of the architecture and focus on the parallel processing features. Then we discuss its main algorithm of the memory cycle that will help to estimate the speed and efficiency of the processor.

Table (٤). Rational system speeds for parallel processing (shared memory).

Number of processor (shared memory)	Form	Rational speed	Efficiency Rs/n*١٠٠
٢	Single level	١,٨٥	٩٣٪
٤	Single level	٢,٧٥	٦٩٪
٦	Single level	٣,٥	٥٨٪
٨	Single level	٤	٥٠٪
٨	٤/٢ or ٢/٤	٤,٦	٥٨٪
١٦	Single level	٤,٩	٣١٪
١٦	٨/٢	٦,٣	٤٠٪
١٦	٢/٨	٦,٣	٤٠٪
١٦	٤/٢/٢	٧,٦	٤٨٪
١٦	٤/٤	٧,١	٤٤٪
٦٤	٨/٨	١٤.٤	٢٣٪
٦٤	٤/٤/٤	١٦.٩	٢٦٪

٥.١. The Architecture of P٤ Processor

The architecture of the Intel P٤ processor [٨] is very complex as in Fig (٤) but the main important advantages are:

١. It has ٤ ALU's, ٤ ways allocation registers and Net Burst unit to work as ٤ parallel processors.

٢. The four ALU's are not symmetric but each one has special properties as indicated in Fig (٤).

٣. The allocation and the ٣٢ entry station unit garnet the parallel in the input side of the ٤ ALU's.

٤. The Net Burst (impeded circuit in the control unit) garnet the parallel in the output side of the ٤ ALU's.

٥. There are three auxiliary ALU's used for auxiliary missions such as memory load and store, these ALU's do not appear in the calculation of the efficiency.

٦. It has two levels of internally caches, the first L١ has ٨-wayes and the second L٢ has ١٦-wayes. The size of these caches is different from type to another, so the sizes of their blocks are different in these types. But the default values for these factors are that used in table (٢) and in the example later.

٧. These four processors are connected as a shared memory with L١. The L١ memory connected with L٢ (as virtual processor) the final form is like *two levels (٤/١) shared memory systems parallel processing.*

٨. The processor will be connected with a single main memory in PC. The result will become like *three levels (٤/١/ ١) shared memory systems parallel processing.*

٥,٢. The Memory Cycle of P٤ Processor

The traditional memory cycle [٩] is as in Fig (٥) summarized in four main steps:

١. Start with the data request from instructions fetch unit to L١ the hit in this point will return the successful order to read/write order in step ٢, while the miss will freeze the processing operations and go to step ٣.

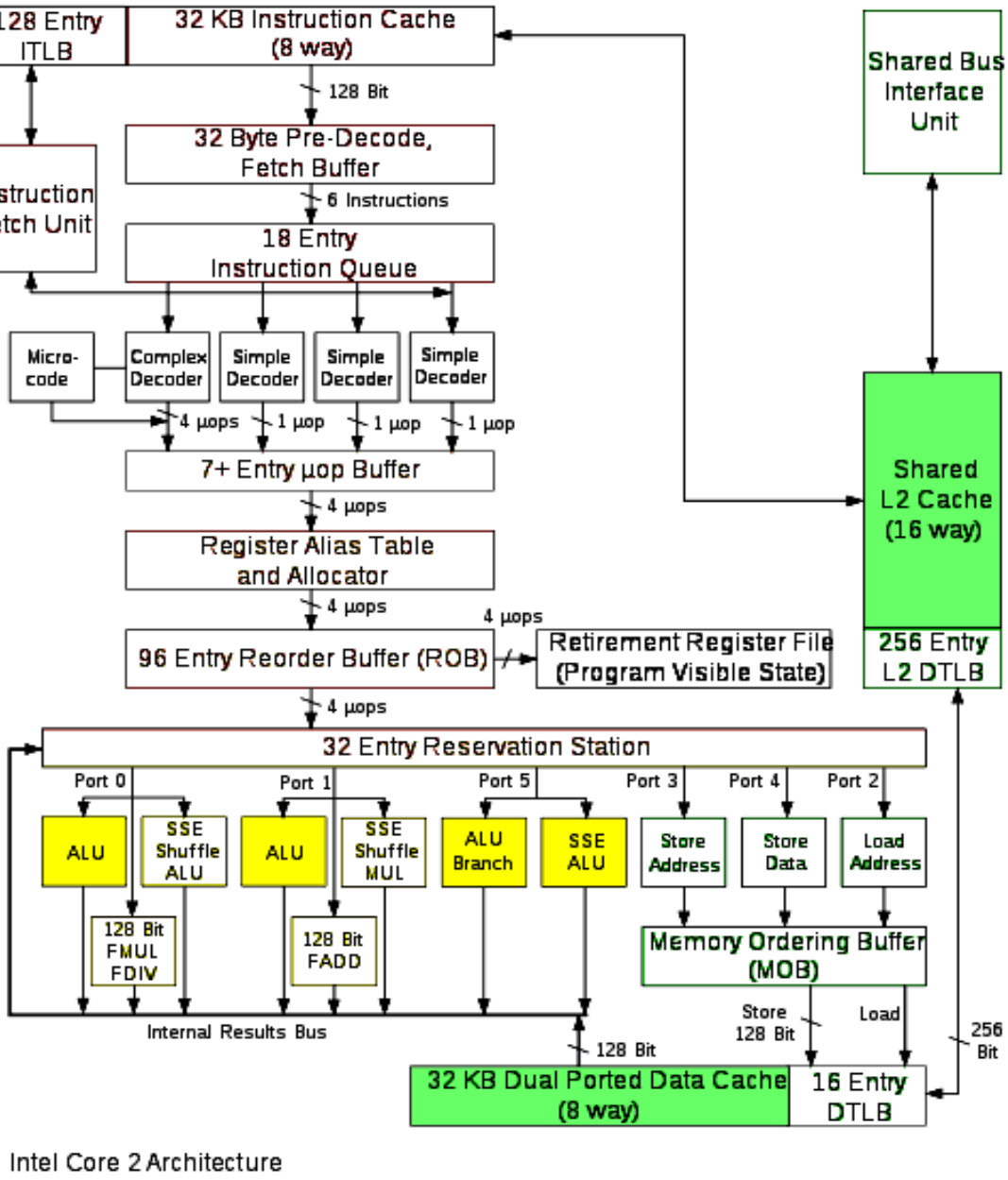

Fig (٤): micro architecture of Intel P٤ processor.

٢. Cheek the write instruction algorithm to move the new data in L١ to L٢ and later to main memory in the ideal memory time.
٣. Start the search in L٢ cache memory, if L٢ hit move the wanted data block from L٢ to L١ and go to step ١, while the miss sends processor to step ٤.
٤. Start the search in the main memory, if the main hit moves the data from the main to L٢ to L١ and go to step ١, while the miss sends processor to step ٥.
٥. Defreeze the processor to start processing with the next instruction and send order to DMA unit to start the search in the hard disk of PC.
٦. When DMA find the wanted data start to move the data from the hard disk to the main memory and indicate the processor.
٧. Refreeze the processor and move the data from main to L٢ to L١ and go to step ١.

Note that the write instruction algorithm to move the new data from L١ to L٢ and to main memory will add a latency time because it works in the ideal memory time.

The miss in L١ and L٢ causes a delay in the processor, because of the freeze in processing in this time. The freeze order is compulsory because the two caches are busy to move the wanted data to L١.

The miss in the main memory is latency because the defreeze order, but this time will become delay if the next instruction depends on the present instruction or if it requires data at the same block as we will discuss in section ٧.

٦. Proposed Parallel Memory Technique

The deep analysis for the P٤ processor shows that, the weakness point in this system is the *crowding* of the virtual processors (ALU's and L٢) on the L١ cache. This problem is solved partially in P٤ by using ٨ way cache L١.

The ٨ way cache L١ [٨] means the L١ is divided into ٨ independent parts but this solution will reduce the problem by factor less than ٢٠٪ because the stream of instructions is stored in the same block so their data are also stored in the same block generally.

At this time we need to *crumble* the single block to independent bytes form, this solution (no blocking) is very complex and impractical for large memories such as in P٤.

This paper proposes a simple solution with a small additional cost and a small additional complexity for the control circuit in the µP but eliminates the problem of crowding in L١ in about ٩٠٪.

The basic idea depends on the use of *multi independent internally channels*, each channel has *data bus and address bus* inside the memory as in Fig (٦ a) [١٠] that shows the architecture for the control of the single cell in the present memory while Fig (٦ b) shows the architecture for the control of the single cell in the proposed memory with two channels. The channel that uses

the memory doesn't care to the other channels. Hence, each channel is free to use the memory, so that each processor will become *free* to use the memory.

The collision is found when two channels need the same byte, this problem has very low probability so it will be solved using simple controller inside the memory giving a priority for channels. The controller will delay the lower priority in one cycle (very short time). The low probability and very short time delay means the effect of this case can be neglected in the calculations of the time processing.

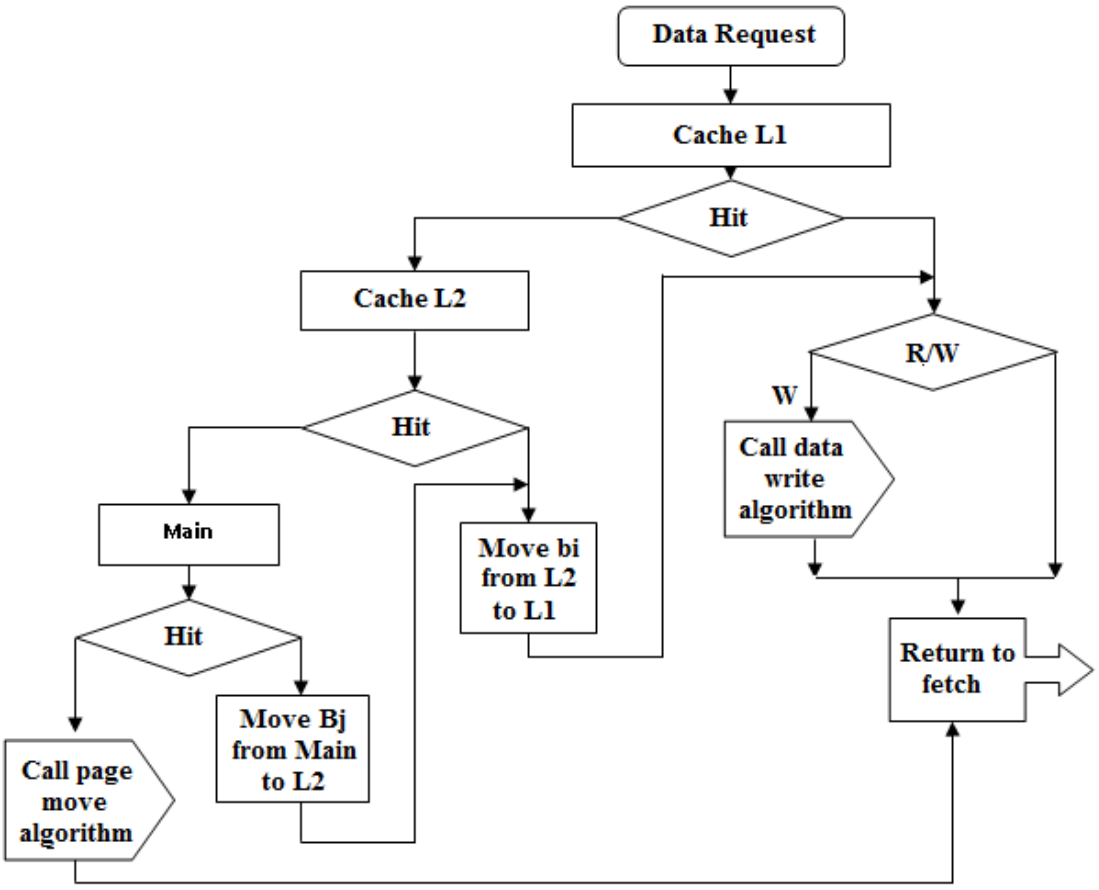

Fig (٥). The memory cycle in two levels caches Intel processor.

Hence, this approach will be used in all the memories of the µP and PC. The effect of this idea will show in the processor example in the next section.

٧. Time Delay Calculations and Simulation

٧,١. Delay and Latency

If we have a queue of instructions under execution in a µP, so any frozen for an instruction in this queue because, generally; it requires data as the miss in L١ case in memory

cycle. This freeze will be solved by stopping all the next instructions. This stopping time is called *Delay* and all the next instructions results will be delayed. Sometime this delay is large as in case of miss in the main memory, the solution to illuminate this delay is by putting aside this instruction and executing the next instructions until the required data is prepared which is necessary to run this instruction, this time delay is called *Latency*. The latency delays the result of

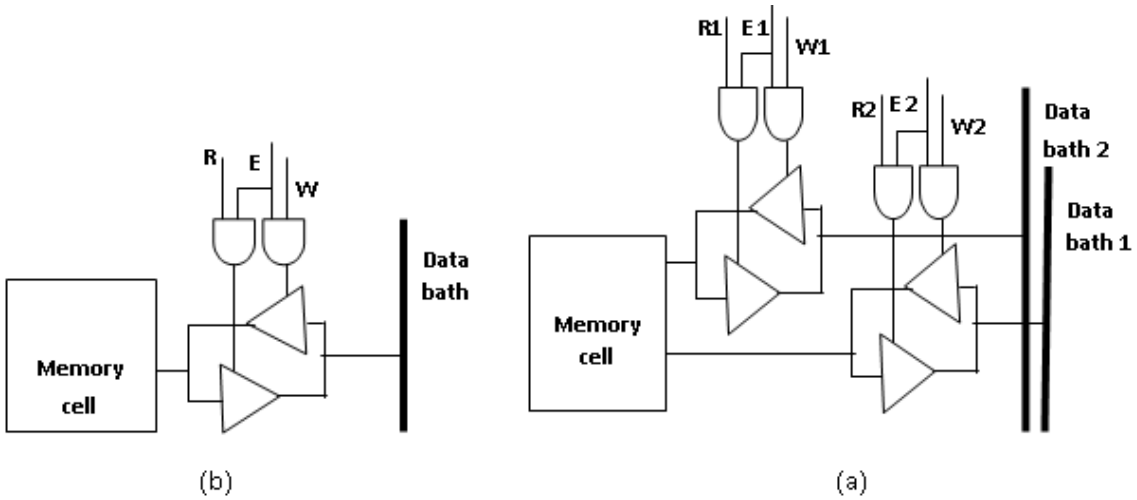

Fig (٦). Memory cell architecture, (a) single channel (b) proposed two channels.

the present instruction while it passes the results of the next instructions. Some times one (or more) instruction from the next instructions queue is depends on the aside instruction or sometimes it share the same block with it. In theses two cases all the latency of these instruction will become a delay.

٧,٢- Processor Example

The practical speed calculations of the processor with and without the parallel memory require practical values. These proposed values are the default values of P٤ processor [٨, ١١] in documents and statistics. They will be classified for simplicity into three groups as follows:

Group A: memory specifications

١. The block size in L١ (bi) is ٦٤ B.
٢. The block size in L٢ (Bj) is ٢ kB.
٣. The page size in main memory (Pk) is ٦٤ kB.

Group B: processor times

١. t is the clock time = the time of executing one instruction in fast ALU in the processor and let t=١.
٢. The time to read or write any data from L١ is ٢ (٢t).
٣. The time to read or write any data from L٢ is ٥ (٥t).
٤. The time to read or write any data from main memory is ٢٠ (٢٠t).
٥. The time to transfer one block from (to) L١ to (from) L٢ is Tb=٢٠ (٢٠t).

٦. The time to transfer one block from (to) L٢ to (from) main memory is TB=١٠٠٠ (١٠٠٠t).

٧. The time to transfer one page from (to) hard disk to (from) main memory is Tp=١٥٠،٠٠٠ (١٥٠٠٠٠t).

Group C: probabilities of the work

١. The number of run instruction in the CPU is ١٠٠٠،٠٠٠.

٢. See table (٥) for the used probabilities. These values are the most used in P٤ processor.

Table (٥). Definition of the probabilities set and their default values in the example.

Symbol	Value	No. of instructions	Definition	Note
Pm	٠,٢	٢٠٠،٠٠٠	Processor need to external data	Pm + Pp = ١
Pp	٠,٨	٨٠٠،٠٠٠	Processor not need to data	
Ph	٠,٩	١٨٠,٠٠٠	Hit in L١	Ph + Pmh + Pmmh + Pmmm = ١
Pmh	٠,٠٩	١٨،٠٠٠	Miss in L١ Hit in L٢	
Pmmh	٠,٠٠٩	١،٨٠٠	Miss in L١ Miss in L٢ Hit in Main	
Pmmm	٠,٠٠١	٢٠٠	Miss in L١ Miss in L٢ Miss in Main	
Pr	٠,٥	١٠٠،٠٠٠	Read data	Pr + Pw = ١
Pw	٠,٥	١٠٠،٠٠٠	Write data	
Pd	٠,٠٠١	٢٠٠	Dependency*	
Pwo١	٠,٠٢	--	Over flow in L١	
Pwo٢	٠,٠٠١	--	Over flow in L٢	
Prw	[٠،١]		Run in the waiting time	Practical value

* Dependency probability means the run of the next instruction depends on the results of the present instruction or the data of the two instructions are inserted in the same block (bi).

The application of the values in table (٥) over the memory cycle in Fig (٥) will result to calculate the average required time to run a single instruction in P٤ processor Tav. These calculations of the average time bellow will shows that the approximated time without parallel-memory approach required is about ٢.٨٧ cycles while the required time using parallel-memory approach is about ٠.٦٣ cycles. In other words the parallel-memory approach will increase the speed of the processor in more than ٤ times.

a. The calculations of the average required time in the example without parallel-memory approach.

$$Tav = Pp \times 0.25 \times (1 - \Pr w) + Pm \times \{[Ph \times 2 + Pmh \times (2 + Tb) + Pmmh \times (2 + Tb + TB) + Pmmm \times (2 + Pmmm \times Tp)] + Pw \times [Pwo1 \times Tb + Pwo2 \times TB]\}$$

$$Tav = 0.8\times 0.25\times(1-\Pr w)+0.2\times\{[\,0.9\times 2+\ 0.09\times 22\ +0.009 \times 1022+0.001\times(\,2+0.001\times 150{,}000\,)]+0.5\times[\,0.02\times 20 +0.001\times 1000\,]\}$$
$$= 0.2\times(1-\Pr w)+2.626\ +0.14$$
$$Tav = 2.866t$$
$$= 0.2*(1-0.5)+\ 2.626\ +0.14$$

b. The calculations of the average required time in the example using parallel-memory approach.

$$Tav = Pp\times 0.25\times(1-\Pr w)+Pm\times\{Ph\times 2+\ Pmh\times(\,2+Pmh\times Tb)+Pmmh \times(\,2+Pmh\times Tb+Pmmh\times TB)+Pmmm\times(\,2+Pmmm\times Tp)\}$$

$$Tav = 0.8\times 0.25\times(1-\Pr w)+0.2\times\{\,0.9\times 2+\ 0.09\times(\,2+0.09\times 20\,)+0.009 \times(\,2+0.09\times 20+0.009\times 1000\,)+0.001\times(\,2+0.001\times 150{,}000\,)\}$$
$$= \ 0.2\times(1-\Pr w)+0.12184 = \ 0.2\times(1-0.1)+0.449764$$
$$Tav = 0.629764t$$

٧,٣. The Results of the Simulation

This sub-section will summarize the results of a simple simulation for the memory cycle of the Intel processors. This simulation will use the default values in the previous sub-section and the information of processors in table (٣). The simulation will run over a proposal case (multi-channels parallel memory) to show the effects of the parallel-memory approach. Hence, the number of channels will be indicated in two arcs {} in table (٦).

The times in table (٦) measured for a universal clock for all processors that write in CPO (clock per operation) while the inverse term OPC (operation per clock) will be use to measure the speeds, the results in the last column is the percentage ratio for the ideal case.

The first row in table (٦) shows the properties of the present P٤ processor that describe as ٤{١}/١{١}/١{١} that's mean it has ٤ ALUs connected to L١ which is connected to L٢ and with L٢ connected to main memory. Hence, all these memories (L١, L٢ and main) have a single internally channel.

The row ١ (row ٤) satisfies the ideal case for the processor that has an ideal time ٠,٢٥ (٠,٠٤٢) for the P٤ (Gulftown) processor but the ALUs are not symmetrical and not all of them are general but each one is specialized for a type of instructions as in Fig (٤) and these instructions cannot be balance for all applications, therefore some ALUs in processor will be idle for some time. The idle time in the case of no miss means this case is not ideal case.

٨. Design of Super Future Processor

The design of a super future processor requires a deep study of the results in table (٦), and then tries some designs under simulation to select the most practical one.

row ٨ in the table shows the form of the imminent future processor that has ٦٤ ALUs and speed ١٦ time over P٤ (for the same clock) ideally but the simulation shows its practical speed is less than ٦ times over the practical speed of P٤, with very low efficiency which is about ٧٪. The enhancement of the imminent future processor using parallel-memory approach (row ٩) will jump up to ٢٤٪ efficiency and the speed to ١٥ OPC.

The try and error for simulation shows the maximum acceptable number for ALUs in processor is ٢٥٦. The row ١٠ shows a processor ٥٥ times faster than P٤ while the row ١١ shows a processor ٦٢ times faster than P٤. The two proposed super processors have ٢٥٦ ALUs but the second design has lesser memory layers with more complexity control circuits.

Table (٦): The results of a simple simulation for the memory cycle of Intel processors.

No.	Type of µP	Parallel form	Time range CPO	Tav CPO	Speed OPC	Efficiency
١	P٤ without miss.	٤{١}/١{١}/١{١}	٠,٣٢*	٠,٣٢	٣,١٢٥	٧٩٪
٢	P٤ with single channel	٤{١}/١{١}/١{١}	٠,٧٦ - ١,٨	١,١١	٠,٩	٢٣٪
٣	P٤ with two channels	٤{٢}/١{٢}/١{٢}	٠,٣٤ - ٠,٧٩	٠,٣٩	٢,٦	٦٥٪
٤	Gulftown without miss.	٤{١}/١{١}/٦{١}/١{١}	٠,٠٩١*	٠,٠٩١	١١	٤٦٪
٥	Gulftown with single channel	٤{١}/١{١}/٦{١}/١{١}	٠,١٩ - ٠,٧١	٠,٣٧	٢,٧	١١٪
٦	Gulftown with two channels	٤{٢}/١{٢}/٦{٢}/١{٢}	٠,١٢ - ٠,٢٧	٠,١٩	٥.٣	٢٢٪
٧	Gulftown with two channels and ٤ channels in L٣	٤{٢}/١{٢}/٦{٤}/١{٢}	٠,٠٩٦ - ٠,٢٣	٠,١٦٥	٦	٢٥٪
٨	Future processor	٨{١}/١{١}/٨{١}/١{١}	٠,١٥ - ٠,٣٧	٠,٢١	٤,٨	٧٪
٩	Proposed processor ١	٨{٤}/١{٢}/٨{٦}/١{٢}	٠,٠٣٢ - ٠,١٢	٠,٠٦٦	١٥	٢٤٪
١٠	Proposed processor ٢	١٦{١٦}/٤{٨}/٤{٨}/١{٢}	٠,٠١٤ - ٠,٠٥	٠,٠٢	٥٠	١٩٪
١١	Proposed processor ٣	٣٢{٦٤}/٨{٨}/١{٢}	٠,٠١١ - ٠,٠٦	٠,٠١٨	٥٦	٢٢٪

The large number of processor (٢٥٦ ALUs) in PPPM approach cannot be successful without the good additional units. This process requires at least to:

١. An enhanced prediction unit that predicates more than one instruction as example set prediction unit.

٢. Extra flexible allocate unit.

٣. Enhanced Net Burst circuit.

٤. Suitable classification and selection for ALUs with suitable distribution.

٩. Conclusions

Having done this work we can conclude some points that help the designer to indicate the optimal way to improve the microprocessor design in the future.

١. Increase the number of parallel processing will be useful for limited number of processors; maximum number is ٨ processors for single level shared memory systems parallel processing.

٢. Increase the number of levels also be useful for a limited number of levels; maximum number is ٤.

٣- The crowded ALUs over L١ in the µP is the main problem in the design of present processors.

٤- The PPPM approach is a very useful technique that will increase the speed of processor to about ٤ times.

٥- The design of the super processors in the future will not be satisfied by using more ALUs and more cores but it requires radical new ideas.

٦- The success of the PPPM approach required to enhance and expand the other units in the µP such as prediction unit, allocate unit and Net Burst circuit.

١٠. Future Work

The key implementation of the super fast processor that covers the future requirements and applications can be summarized in three main layers:

١. Use the PPPM proposed approach in this paper with enhancement and expand the other units in the processors such as the prediction unit, the allocation unit and the Net Burst unit.

٢. Use the assistant special processors; this idea is like the use of the math co. processor with the ٨٠٢٨٦ in some PCs. The proposed assistant processors are coding/decoding processor, ٣D image processor and complex function processor.

٣. Discuss and deduce new ideas for the data processing, which are suitable with burst data processing. These ideas must be left the classic idea of the ALU that depends on the using a single point (ALU) to execute a complete single instruction and use algorithms able to execute a large number of instructions partially at the same time. Hint, the burst data processing approach may burn a complex and slow processors but it is able to grow faster than the classical ideas.

References

[١] List of Intel microprocessors, Wikipedia, the free encyclopedia, "http://en.wikipedia.org/wiki/List_of_Intel_microprocessors".

[٢] Intel Consumer Desktop Pc Microprocessor History Timeline, http://www.intel.com/intel/intelis/museum/exhibit/.

[٣] Intel® ٦٤ and IA-٣٢ Architectures Optimization Reference Manual, Order Number: ٢٤٨٩٦٦-٠٢٠, November ٢٠٠٩.

[٤] Blaise Barney," Introduction to Parallel Computing", Lawrence Livermore National Laboratoy, "http://www /Introduction to Parallel Computing.htm".

[٥] Trupti Patil, "Evaluation Of Multi-Core Architectures For Image Processing Algorithms", Thesis Presented to the Graduate School of Clemson University August ٢٠٠٩.

[٦] B. L. Buzbee, "The Efficiency of Parallel Processing", Frontiers of Supercomputing, Los Alamos Science Fall ١٩٨٣, pp ٧١.

[٧] Quentin Smith, "Efficiency of Parallel Processing with JCilk", under the direction of Professor Charles E. Leiserson Massachusetts Institute of Technology Research Science Institute, August ٢, ٢٠٠٥.

[٨] Glenn Hinton, Dave Sager, Mike Upton, Darrell Boggs, Doug Carmean, Alan Kyker, Patrice Roussel, "The Microarchitecture of the Pentium ٤ Processor ", Intel Technology Journal Q١, ٢٠٠١, pp ١.

[٩] Mostafa Abd-El-Barr,Hesham El-Rewini, "Fundamentals Of Computer organization And Architecture", Copyright # ٢٠٠٥ by John Wiley & Sons, Inc. All rights reserved.

[١٠] Samuel Naffziger, Blaine Stackhouse, Tom Grutkowski, Doug Josephson, Jayen Desai, Elad Alon, and Mark Horowitz," The Implementation of a ٢-Core, Multi-Threaded Itanium Family Processor " IEEE journal of solid-state circuits, vol. ٤١, no. ١, January ٢٠٠٦, pp ١٩٧.

[١١] Trent Rolf, "Cache Organization and Memory Management of the Intel Nehalem Computer Architecture", University of Utah Computer Engineering, CS ٦٨١٠ Final Project, December ٢٠٠٩.