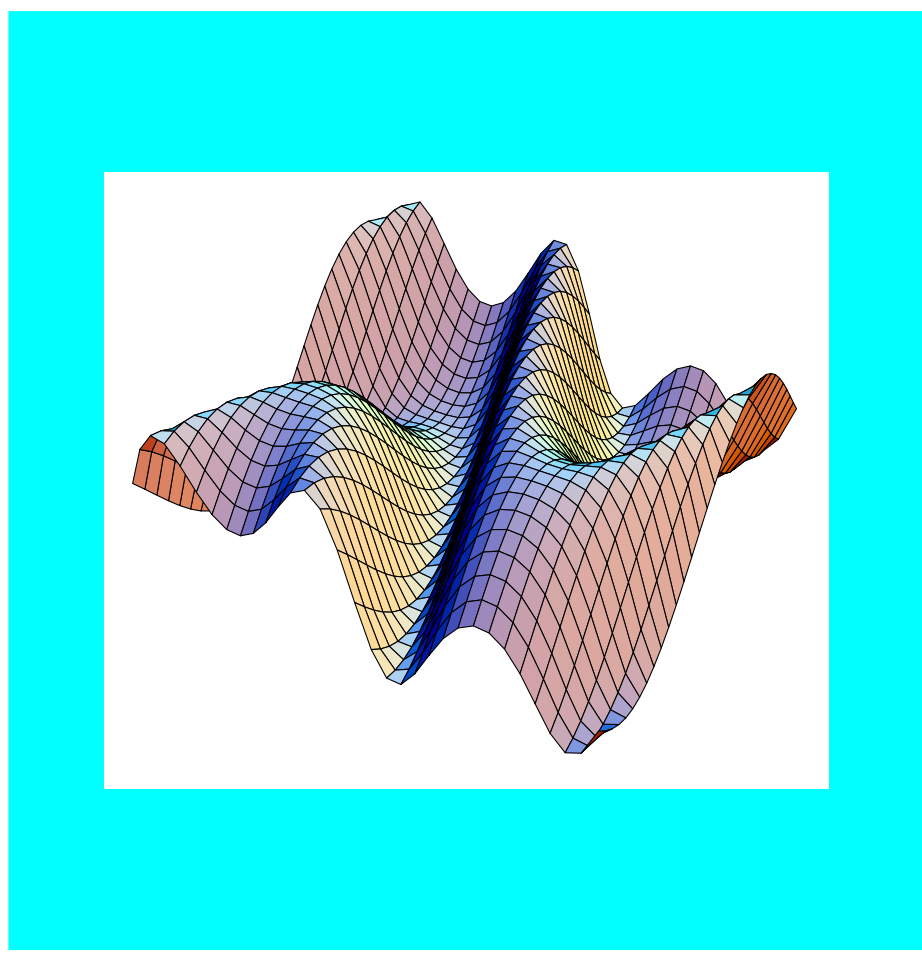


Area and power consumption of multiprocessors / Gunnar Carlstedt. - Nösslinge : Vaftrudner, 2001-12-19
ISBN 91-89546-12-1
ISSN 1650-2027, no 12

Area and power consumption of multi-processors

by Gunnar Carlstedt



1 Abstract

This report calculates the physical size and the power dissipation for a particular application implemented on a multiprocessor. All parts except the control part as instruction decoding and addressing are treated.

It is found that the power dissipation is approximately equal to twice the intrinsic logic gate power, i.e. the best theoretically possible. For large problems an additional power being proportional to the mean vertex length of a DAG has to be added. Power due to memories may be ignored.

The size being the area of microcircuits corresponds to the area of the pure memory cells for a theoretical minimum amount of memory needed.

It is also found that the implementation of the arithmetic, memories, wiring and clock synchronisation depart heavily from the traditional way to implement them. The techniques used, is however not new.

It is also found that there is no need of long wires on a chip. Thus the emerging problem with transmission delays on such wires is eliminated.

2 Summary - introduction¹, method and result

Computer architecture as a science has existed since the 1950's. In the beginning it was a path of trial and errors in order to do the best to find what a processor was. The result was structure.

In the 70's the research turned to be more results oriented. New and old structures were researched. By using different test suites measures from simulated and also actual measures from tests were used to compare different implementations. This type of research has continued until now.

In the respect of multiprocessors the research have mainly been on implementation details and structures. Little measures have been reported.

It is the author's belief that computer architecture could be turned into a real science where the various characteristics of a computer could be described in analytic form by mathematical expressions. In an earlier paper [3] it has been shown that the main computer architectural measures are **latency time**, **memory size** and **communication speed**. An application could be characterised by a number of measures. By using these application measures the architectural measures could be calculated for a particular multiprocessor.

Such a (multi)processor has a general structure consisting of arithmetic units and memory modules all being connected by a communication network.

The research in this report is a continuation where the physical measures should be calculated. The physical measures are the latency **time delay**, **area**, and **power dissipation** based on resistance and capacitances of actual physical devices as transistors and wires.

This report does not differ between processor and multiprocessor. The general understanding of a processor is a special case of a multiprocessor. The words processor and multiprocessor are therefore freely used

¹ This report is not written as conventional research reports. Instead a top down breadth first description is used. Thus the main paragraph is the abstract followed by a summary containing an introduction, problem formulation, result and description of research method. A discussion of the result follows and then all paragraphs according to the research method.

and should be considered the same, but when referring to traditional research the traditional meanings are used.

2.1 Problem formulation

A processor is a machine consisting of arithmetic units, memories and a communication network. The parts have a very general meaning, thus

- an *arithmetic unit* is some type of device using binary state to implement expressions with physical devices as gates. No states are stored. The entire unit is combinatorial.
- a *memory* is a type of device storing words of binary information. The simplest form is a register, and more complex forms store words in lists or arrays.
- a *communication network* consists of switching arithmetic and communication links. The switching arithmetic moves depending on control binary information from one point to another point. The communication links are the necessary devices and the transmission medium performing the actual physical transport.

This report assumes that the implementations of these units are based on CMOS integrated circuit technology. The communication is mainly on metal wires, either on-chip or off-chip. Optical transmission links are also treated.

The physical implementation has, except for the communication links, a **size** being measured as area. Communication links have a physical size on-chip, but off-chip they just have a length. They all consume **energy** for a particular operation. If this energy is repeated with a particular frequency, a **power dissipation** could be estimated. Each unit has a throughput defined by the **scheduling interval**. It is the time between two subsequent operations performed by the particular unit.

The problem to be researched in this report is

- find appropriate ways to abstract the processor units
- define the important characteristics of the units
- give analytic expressions on the characteristics for the particular units and from them give the physical measures for a processor.

2.2 Results

If the application is too small there is no result.

The application is executing either one task or repeated tasks with real-time cycles. One such task is considered. It could be described by a *data access graph* **DAG**. It consists of operations connected by directed vertices. The result of an operation is a node.

area

Allocation and scheduling methods, not described in this report [3], may allocate and schedule the nodes to a number (less than the number of nodes) of memory words. The **size of an optimum processor** is almost the size of the memory cells for these words. The size of wiring, communication switches and arithmetic units could generally be ignored. However, contemporary implementations are far from this.

energy

In the DAG there are operators. If each of them is substituted by a DAG for the operator implementation, a complete DAG is the result. Such a DAG contains a certain amount of nodes, equivalent to the amount of gate nodes. The technology used has a gate capacitance. When it is

charged to the power supply voltage, it contains a particular energy. This energy is the reference for energy. The energy of an optimum processor dissipates about 4 such reference energies per gate node. The energy for communication links and memories could generally be ignored. However, contemporary implementations are far from this.

For some classes of problems the communication may increase the energy consumption by a factor 10.

2.2.1 Indicated results

As a by-product to this research there is another result, however just indicated and not proved:

A multiprocessor chip could be assumed to consist of a number of regions. Generally such a region is a set of arithmetic units, memories etc or parts of them. From circuit point of view these units do only have one global electrical feature - the power distribution.

All wiring is short and only within a region or between neighbour regions. This is for all types of wires, including data, control and clocks.

2.2.2 Other research

As a part of the result it is found that there are some areas needing more research

- a bidirectional **graph of clocks** are assumed. The clocks are to be controlled to have the same phase. Each clock is generated by an oscillator "clock generator". Each such one has a connection according to the graph. Thus each clock generator has as input time references from other clock generators. They are used in the regulator loop. The goal is to have picosecond phase difference.
- **high trough put low power memory**. Contemporary fast memories are generally designed for fast access time at an adequate power dissipation level.
- **low power short distance communication links**. The goal is to reduce the energy consumption for a transferred bit as much as possible.
- **pipe-lined arithmetic**. The goal is to increase frequency, reduce area and energy. Critical problems to solve is clock distribution and register design. This may result in restrictions on the DAG for operators.

There is also one more fundamental problem to research, namely the control mechanisms of parallel hardware (see paragraph 3.7 on page 10).

2.3 Method

A straight forward research method is used:

- 1 operators and nodes are allocated and scheduled to arithmetic units and memory words, respectively.
- 2 The main performance issues for the different types of units are described and analysed.
- 3 For each type of unit one or several good circuit types are described. Each of them are implemented using CMOS technology or appropriate hardware.

- 4 The hardware characteristics are described with mathematical expression. The scheduling interval, the area and the energy dissipation are estimated.

The units are described as general units with size parameters. The parameters are optimised for best performance. The hardware characteristics are estimated using these optimised parameters.

- 5 The physical measures are related to programming concepts, i.e. the DAG. The different units and the processor are discussed in that respect.

3 Discussion

The main assumption for this research is that the execution could be described by a DAG [3]. It has to be derived by some means from the programs specifying the application. This report does not describe this conversion nor does it show that it is possible.

It is clear that many signal processing applications can easily be converted to DAGs, however sometimes rather large. There are other applications that must be reformulated or translated in order not to be sequential. An example is scanning of text strings.

A first step in the design procedure is to supply specifications for the arithmetic. This specification is also a DAG. It substitutes each operator in its place in the DAG of the application. This resulting DAG has all the properties needed for an implementation in a multiprocessor. The specification for the operator is an implementation. There is a trade off between several parameters when selecting such an implementation (see paragraph 3.2 on page 7). The result is almost insensitive to the available choices.

There is also a main decision whether an implementation should be bit-serial, byte-serial or parallel. It could be shown that a parallel variant is better because it uses less amount of memory (see paragraph 3.3 on page 8).

This report uses standard static CMOS technology. There are a lot of circuit technologies available. Among them there are some modern or very high-performance types. The result is almost insensitive to the choice of technology (see paragraph 3.4 on page 8). It is not likely that area could be change significantly, however energy consumption may be slightly reduced.

Technology will continue to evolve. The circuit devices like wires and transistors are scaled. The various devices do not scale with the same amount. Wires are scaled much less, and the long ones almost not. In the long run this may change the possibility to make optimal design because wires will dominate the chip. Massive parallelism allows the introduction of pipe-lined wires. The effect will be less wires need. This could compensate the less scaling on wires (see paragraph 3.5 on page 9).

Optimisation is generally a complex problem. In this report there are only some few parameters to tune. However, there are a lot of trade-offs to consider. Most of them depend on properties of the DAG. It is likely that there is such an optimum (see paragraph 3.1 on page 7). The area is depicted by the amount of memory needed and the energy consumption is given by the DAG.

In order to facilitate a commercial concept there must be some building block. If not all applications would need a custom design that is expensive. In such a block things are generally not optimal instead there is a commercial trade off between various properties. There is such a building block (see paragraph 3.6 on page 10). It is probably based on memory. It would have adequate amount of communication network and arithmetic units. The good with this is that it is a very neutral concept that could be used and manufactured by almost anyone. The size of such a block in contemporary technology is around 1mm^2 . It could therefore be packaged in multiple units.

power dissipation

The operator is central. To an operator approximately none memory cell. Memories are accessed by three memory accesses. There are three vertices corresponding to paths in the communication network. The path is 1 to approximately 30 communication links deep. From energy point of view the energy dissipated by the arithmetic dominates in most cases. For large problems the dominating operators are floating point. With the contemporary technology of Table 1: 80pJ per operator. Executing 1 Tflop/s dissipates 80W. Power dissipation scales quadratic with scaling geometry of the technology.

area

The scheduling rate for the arithmetic unit is almost constant times the transistor characteristics. Using the same technology as above the scheduling interval is 60ps, i e a clock frequency of 16GHz. The specific area per executed operator is for an arithmetic unit $75\mu\text{m}^2/\text{MHz}$. The execution of 1 Tflop/s needs 75mm^2 chip area for the arithmetic.

The memory size is impossible to state. The theoretical minimum is very low, and less than the size of the arithmetic units. Probably the memory area dominates.

The arithmetic unit area scales cubically by technology and memory cells quadratic. Therefore it seems that the area of the arithmetic units will almost disappear by time. This is not the case, because the amount of threads could scale down with technology. The memory size is therefore scaling cubically.

difference to state of the art

The characteristics of such devices are extreme. The power dissipation is probably three orders of magnitude less than today's commercial processors. The memory area is dominating the chip and is application dependant. Due to the large amount of parallel threads this area is much larger than in conventional sequential processors. The performance, i e operators per time unit, of a multiprocessor is several magnitudes higher than the contemporary single processors.

There is a main question - why is this the case? why is the solution not found earlier? The answer is twofold -

- **this report does only consider one half of the problem.** Also the DAG has to be generated during run-time by the multiprocessor (see paragraph 3.7 on page 10).
- the traditional control by instructions in von Neuman computers is well adopted for sequential execution but not for parallel ones.

For problems where the DAG is simple there may be rather simple solutions that could be used to reach the reported performance. For other DAGs there is a need of more research in methods to control the hardware.

There will be a subsequent report on that subject by the author of this paper.

3.1 What is optimum?

In the sections for the various physical units, arithmetic (see paragraph 6 on page 14), memory (see paragraph 7 on page 21), communication (see paragraph 8 on page 30), and clock (see paragraph 9 on page 38), the units have each been optimised to both area and power. It has been shown that several units are needed of each type. Generally, it is not the individual unit that should be optimised rather the entire processor.

There are two inputs to the optimisation, the *number of nodes* and the *number of operators*, that are fixed. Starting with them

- 1 the size of the memory and the memory scheduling rate could be estimated. The number of memories are then given.

The number of arithmetic units could be estimated as shown in this report. However, there is a trade off between area and energy consumption. It is likely that the area of the memories are much larger than the area of the arithmetic units. A change of the area of the arithmetic units does not change the total area significantly, thus the arithmetic unit scheduling rate could be decreased in order to decrease the energy consumption. Thus,

- 2 find a proper trade off between area and energy consumption. The scheduling rate is set somewhat longer than the optimal one.

A change of the scheduling rate changes the amount of arithmetic units. The number of ports to the communication network is subsequently changed. Generally this change is marginal.

3.2 Implementation of cells like wires, memories and arithmetic

In early days wires, memories and arithmetic units were very simple. The reason to this was that chip area was very expensive. When time elapsed the application demands changed the units. Different trends and optimisations were used. It resulted in the types of units we see today.

For most electronics there has been a demand to reduce the cycle time and the latency time. As a consequence the propagation delays have been pushed down. An example is an adder: it started as an implementation with full adders consisting of a three input EXCLUSIVE OR, a two input OR and an AND/OR-gate. A 32 bit adder consisted of 896 transistors. Now adays a carry look-ahead adder is used containing 1708 transistors and with a much more complex wiring.

It is shown in this report that the throughput rather than the latency time is important in order to reduce chip area. In order to reduce energy, units with as few switching inputs as possible, i e simple circuits, should be used.

The effect of this is

- normal integer arithmetic should use simple adders with some proper trade off between latency time and throughput.
- parallel multipliers should be considered only if their energy consumption per operation and their ratio of area and throughput are favourable.
- floating point arithmetic could use simpler solutions than current ones

The increase of latency time has a penalty. It causes more threads in execution. Generally the memory size should increase logarithmically with

the size of a thread and proportional to the number of threads. The amount of memory may therefore increase.

There may be a trade off between latency time and area. Probably should latency time be short, but not pushed. This means that normal parallel methods should be used but probably with some simplification.

3.3 Serial or parallel?

Traditionally there has been two major implementations for massive parallel processing - serial or parallel arithmetic. Is it possible to derive their merits?

The allocation and scheduling use a DAG. It does not indicate whether serial or parallel implementation is to be used. However, the operators are substituted with DAGs using gate operators. The substitution may use different algorithms, e.g. an adder may use a simple carry chain or a look-ahead carry. The carry chain has low area and the look-ahead carry has short latency time. Because latency-time is traded by memory cells, there could be a slight increase in area when using a simple carry chain.

An execution is said to be sequential if the bits of a word for one operation are allocated to the same arithmetic unit. It is likely that during subsequent periods use bits from the same word.

Whether bits are allocated parallel or sequential it does not change the amount of work to do. Thus there is no difference from arithmetic unit or communication link point of view. However, memories were introduced as an area effective device. It was shown that the address decoder was common for several bits of a word. Depending on the memory size and the amount of memories needed, it is likely that bit sequential access is power wasting and often also area wasting.

To conclude, bits sequential is not as effective as parallel.

3.4 Circuit technology

A typical standard static CMOS technology has been used when deriving the characteristics of the units. There are more circuit types to be considered. However, it was not the purpose to find the best circuit technology and then use it in the most optimum way, rather it was to find the physical bounds. There may be better circuits, but it is not easy to derive whether they are better or not.

single phase dynamic logic

This logic uses a dynamic load [15][35]. Each gate acts as a register. The pipe-line register is therefore not an overhead. There is also a related dynamic technology [11].

In each gate there is two transistors switching with the clock frequency. The capacitive load on a gate output is as for conventional CMOS. The effect is that the register consists of one switched invert with one inverter load. Compared to static CMOS the clock load is only 2 gates compared with 6. The energy reduction is considerable for shallow pipe-line stages.

The register corresponds to 4 additional transistors over conventional static CMOS. Wiring for the gates are much simplified because only one transistor has to be driven for each clock input. The effect is probably a considerable area reduction.

Because the chip area is generally dominated by memory, the effect on the chip area is small. The effect on energy dissipation is high, it influences memory access circuits, wires, communication networks and arithmetic almost proportional.

domino logic

A modern technology used is some type of domino logic [9][30]. It is used in high-performance arithmetic units [7]. It is a clocked logic. Therefore the pipe-line is free, no additional registers are needed. On the contrary, the clock switches have to be included in all gates, probably also in nodes between transistors. It is not clear whether the amount of transistors driven by clocks is greater or less. Probably there are more. The same applies to the total transistor count. Clear is however that more nodes are switched and therefore the energy dissipation is higher.

The effect may be that domino logic is faster and therefore fewer arithmetic units are needed. Thus the area of the arithmetic units decreases. However, they are not significant for the total area.

Effect of circuit technology

Circuit technology can reduce the time delays in registers. Therefore the optimal scheduling rate of arithmetic units may be reduced to one gate delay. This will reduce the amount of arithmetic units and reduce the total area because of this. Generally this are is marginal.

On the contrary, the latency time of an arithmetic unit may be reduced. This will reduce the amount of memory. The reduction is almost proportional to the latency time. Therefore a slight reduction in total area is expected.

3.5 Technology

Technology scales. A main problem is that everything does not scale equally. The trend during the last 10 years is that transistor length and with scale. Wire density does not scale fully. The main wiring layers are the two first the polysilicon and metal layers. The uppermost layers do almost not scale.

The effect of scaling is:

- the units become smaller.
- memory cells are scaled more than other units. Designs dominating by memory will probably scale best.
- long wires are not scaled as much as the units. There is a time when the area of a chip will be dominating by wires. As wiring and its topology is fundamental to the DAG it could not be replaced.

However, transistors scale much. The introduction of registers in long wires would help to increase the speed, i.e. reducing the scheduling interval.

The relative size of registers including receiver and transmitters would decrease by scaling. Therefore the distance between the registers could be decreased. The communication speed could be further increased and the amount of communication links will decrease.

The relative area of wires for a particular DAG may therefore decrease when scaling.

- power distribution wires are almost not scaled. This is good, because power density increases and therefore wires need to have lower resistance. It will not influence the multiprocessor design.

The wire width does not scale as good as transistor width. Therefore the ratio of wiring capacitance to gate capacitance will increase. The relative energy consumption for transports will therefore increase. In the long run the power consumption from the wires will be dominating.

3.6 Is there a processor?

Assume two types of arithmetic units, L and S, having 10000 and 2000 gates, respectively. They correspond to 32 bit arithmetic for floating point [1] [6] [13] [14] [18] [31] [32] [33] [34] and fix point, respectively. A typical implementation of them will use 40000 and 8000 transistors. Assume that there is a area ratio between the transistor density in arithmetic units and memories being 10. They use an area equal to 520 and 104 memory words, respectively.

The scheduling rate of the arithmetic units sets the throughput of the communication network. More than 96 communication links are needed. In a worst case when one operand is stored locally and the other more globally the long distance communications needs 20-30 communication paths containing several communication links.

In traditional implementations there have been a processor module. With the general approach used here there are no such unit. However, there may be a good package containing a part of the processor. Such a package could be named "**processor module**".

If an application use much more area of one type of unit, this type of unit may set the size of the processor module. Other units may be added with an overhead.

The L and S arithmetic units are expected to have 60 and 15 pipe-line stages, respectively. Each such stage executes one thread in the DAG. A thread stores long-life nodes. The amount is application dependant. However, using a depth first scheduling policy it is likely that there is a need of more than 10 memory words per thread. Thus the amount of words near an arithmetic units is greater than 600 and 150, respectively. As shown above, the area of these words are somewhat more than the size of the arithmetic units.

Most applications use far more memory. If so a processor module may be based on one arithmetic unit and memories using the dominating part of the area. The communication needs as shown above need a limited amount of communication links. The number of links are so few that they could be implemented as pads or on-chip wires.

The answer to the heading is yes, and it is based on memory as resource. There is an overhead of arithmetic and communication. A module containing 8192 words, i e 32kbytes, may be proper. The memory dominates such a module. The size of the memory is in contemporary technology 0.5mm^2 . It needs as a chip 60 pads for communication and probably 10 for power giving a sum of 70 pads. This indicates that the pad distance is around $35\mu\text{m}$.

3.7 The control is large and complex

The control part is not included in figures discussed. It is regarded that the control part is the unit needing most research in order to find a proper solution. It is however not in the scope of this report.

To give the reader some understanding of the complexity and the problem with the control, some few words should be said.

control data dependency

The DAG may contain if-then clauses. They could be treated as operators, where the parameters are the expressions to be read. A lazy evaluation would eliminate the execution of unnecessary parts of the DAG.

indexing

Indexing could be treated as an operator having the entire array as input. Because of the way the DAG is treated in this report, there is an increase of the communication much more than necessary.

NIMD

A DAG is a graph. Each operator and node is allocated to a physical unit and scheduled to a time period. If the DAG is not data dependent this binding is constant. This is the case for several signal processing applications. Thus the control is a constant array indexed by time period for each physical unit. There is no instruction. The control is **No Instruction Multiple Data**.

Programs

A program could be considered an execution dynamically creating a DAG. The classical von Neuman technique using instruction streams is one of several control methods. A general control method includes the following steps:

- 1 *duplication* a template of a function.
- 2 *delaying* parts being lazy driven. A part could be eagerly or lazily driven.
- 3 *decoding* of each variable reference to an identifier (something similar to an address)
- 4 *mimic the allocation mechanism* and calculate the actual physical unit to be controlled or accessed.
- 5 *substitute* the identifiers with the actual value for each variable. This mechanism is the communication mechanism described elsewhere in this report.

This mechanism is to be allocated and scheduled on a processor. For a typical von Neuman processor these steps corresponds to 1. instruction memory read, 2. branch decision, 3. instruction decoding, 4. base address add and indexing, 5. memory access.

As shown, each operator or vertex in the DAG has five execution steps just for creating the DAG. Contemporary used techniques is complex. There may be other more effective methods. These are to be discussed in other reports.

The complexity is confirmed by contemporary processors. The control of the pipe-line, branch prediction, dynamic register allocation and scheduling, TLB arithmetic and address arithmetic units are large. They may occupy the major part of the processor chip area.

4 Allocation and scheduling

The entire application is considered as a sequence of real-time states R_n , inputs I_n and outputs O_n . There may be several real-time state sequences being asynchronous. It complicates the figure but does not change the method. Therefore only one real-time cycle is assumed.

During one real time cycle n the input I_n and the real-time state R_n is used as input to a function F having a result to the next real-time state R_{n+1} and output O_{n+1} . The function F could be expanded to a directed graph DAG containing operators and nodes. An operator has an output being a node. A node feeds other operators.

The DAG is the structure used for the analysis in this report. It is also used by allocation and scheduling algorithms.

Allocation and scheduling algorithms time space multiplexes the DAG onto physical hardware:

- operators are implemented in arithmetic units
- nodes are stored in register (or memory cells)
- vertices are placed on communication links

4.1 Operators

Operators are generally from a set of used operators. Each operator may be expanded further into gate-like operators, e g AND, OR, case, if. Each such gate operator is to be implemented by a physical gate.

Different operators may be combined into one larger controlled operator. The control selects the particular operator. The combined operator is to be implemented in an arithmetic unit.

It is outside the scope of this report, what synthesis performed to form combined operators with gate like operators. It is assumed that there is a set of combined operators, each specified using gate operators.

For each type of combined operator there is a corresponding physical arithmetic unit. It is implemented according to the detailed description using gate operators.

- 1 There is a set of arithmetic units, at least one for each type of combined operator used in the DAG. These operators are instances alu_i .
- 2 There are time periods t_j .

The allocation and synthesis algorithms together allocate an operator op_i on an instance alu_k during the time period t_j . No other operators are allocated simultaneously on that instance.

4.2 Storage - registers and memory

Input and output are special states. They are ignored here. They could be treated as nodes.

Because there is a time space multiplex, the node values need to be stored after an allocated time period in an instance. It is performed in node instances. They are stored in registers. As will be shown later memories are optimised versions for a network and a set of registers.

In the same way as operators are allocated and scheduled on instances the nodes are allocated and scheduled on registers.

4.3 Communication

All vertices carry state from one point to another. The points are generally nodes and operators.

In the same way as operators are allocated and scheduled on instances, the vertices are allocated and scheduled on communication links. Further down the concept network is defined. In order to connect to points several communication links have to be passed. Therefore allocation and scheduling is performed on a sequence of communication links along a path through a communication network.

5 Physical performance

An implementation of an application is a multiprocessor. It consists of various parts. They all together exhibits the following physical performance.

- **latency time** is the time from a clock cycle, during which the first state of the real-time cycle is used, until the end of the clock cycle, during which the last part of the new real-time state is clocked into a register.
- **area** is the total physical area for integrated circuit implementation on a single chip or several chips, ignoring pads and external drivers if not stated explicitly. Receivers and drivers for communication links are included.
- **energy dissipation** is the energy used for a unit during one real-time cycle. It includes energy for communication links of reasonable size. The ratio energy divided by latency time is the **power dissipation**.

The latency time should be equal to or less than the period of the real-time cycle of the application. In the analysis here, they are assumed to be equal. The implementation analysed is the data path including memories, registers, arithmetic and data paths. **Control structures are not included** (see paragraph 3.7 on page 10).

Physical cells

In order to evaluate the performance, physical cells are needed. They are all described in subsequent sections of this chapter. There are many different types of cells available to use. Here there are references to some few of them and only one is analysed here.

A cell is characterised by the parameters described above. It is implemented in a particular technology. The goal in this report is to use a total technology independent analysis using a switch model. It is almost possible, but some few adjustment must be included.

Technology

The basic technology is based on CMOS. In the switching model it is assumed to be n- and p-channel devices. They are always of minimal size except for where explicitly specified scales S .

A transistor is assumed to have a resistance r/S and a gate capacitance $c*S$. All other capacitances are neglected.

The analysis is generally base on a symmetrical p- and n-channel device. However, actual current of the devices differ by approximately a factor of two. This difference can generally be neglected by assuming the resistance from the p-channel devices and the capacitance as the mean capacitance of p- and n-channel devices.

The model is of course wrong because it neglects important parameters as stray capacitances etc. The model is however used by many simulators where the r and c values are adjusted to compensate for such differences. The accuracy of such simulators are assumed to be good and adequate.

Transistors

MOS-transistors are controlled by charge in the channel under the gate oxide. The electrical field between drain and source creates a current between drain and source. The speed is proportional to the electric field V_{ds}/L_{chan} . The speed is limited to v_{th} by a saturation speed caused by the thermal action of the electrons. The saturation is present fro channel length less than some micro-meters. The drain current I_d is:

$$I_d = v_{th} \cdot w_{tr} \cdot c_{ox} \cdot V_{gs} \quad 1$$

Here w_{tr} and l_{tr} are the width and the length of the transistor and c_{ox} is the gate capacitance per unit area. The threshold voltage has been neglected. The time delay for this current to charge a capacitance corresponding to the gate capacitance is:

$$\tau = w_{tr} \cdot l_{tr} \cdot c_{ox} \cdot V_{gs} / I_d = l_{tr} / v_{th} \quad 2$$

This speed will be used as the base for the analysis in this report. The speed is almost only dependent on physical constants, v_{th} is approximately equal to 100km/s. For the channel length 100nm the speed is 1 ps.

The charging a gate is the main cause of power dissipation. In this report the charge on one transistor is fed from one power supply V_{dd} . The gate oxide of the transistor has the dielectric constant ϵ_{ox} and the thickness t_{ox} . Thus the energy is:

$$e = (w_{tr} \cdot l_{tr} \cdot \epsilon_{ox} / t_{ox}) \cdot V_{dd}^2 \quad 3$$

For a 1 V power supply 3nm oxide thickness and 100nm gate length and 400nm gate width the energy is 45aJ.

Contemporary technology

In this report it is not the goal to get an absolutely correct figures but figures to be compared with each others. As such the model is assumed to be fine.

State of the art technology is fast changing. Because this report should be technology independent there should be no need of data for a particular technology. A technology is assumed. Figures should be at state of the art, but are assumed. Below is a contemporary technology specified:

measure		p-channel	n-channel	
dielectric constant	ϵ_{ox}	3.85	3.85	
electron saturation speed	v_{th}	100	100	km/s
gate oxide thickness	t_{ox}	3.0	3.0	nm
gate length	l	0.15	0.15	μm
gate width	w	0.4	0.4	μm
gate capacitance	c	1.0	1.0	fF
transistor current	I_d	680	680	μA
transistor resistance	r	1,5	1,5	kohm
drain diffusion capacitance	c_d	100	100	aF
transistor time constant	τ	1,5	1,5	ps
power voltage	V_{dd}	1.0	1.0	V
switching energy	e	1.0	1.0	fJ
wire capacitance	c_w	200	200	pF/m
chip area per transistor	A_{tr}	12	12	μm^2

Table 1: A reference technology.

6 Operators - arithmetic units

As shown above (see paragraph 4.1 on page 12) the arithmetic is a net of arithmetic operators connected by a DAG. The allocation and scheduling mechanism time space multiplexes these operators on a number of *arithmetic units*.

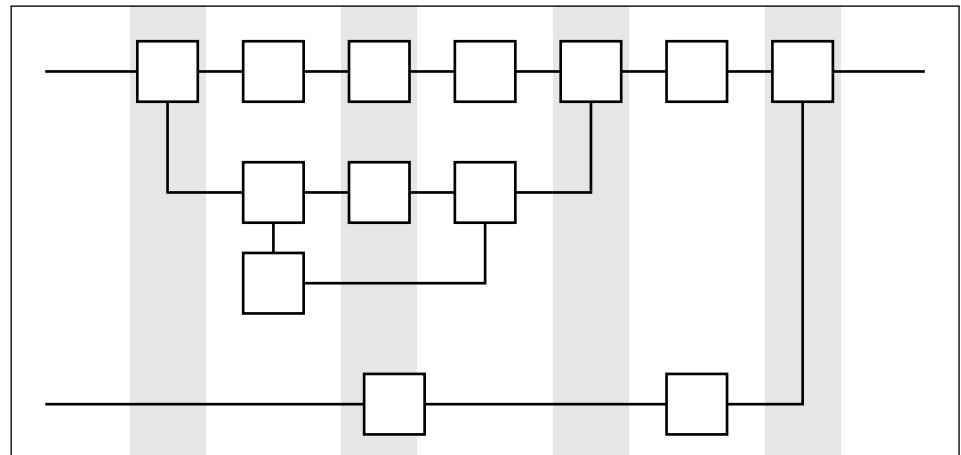
The arithmetic unit is generally a rather large unit. It could however be divided into a number of units resulting in an even smaller unit for multiplexing. Because most operations are add and multiply-like operations, the finer units are more or less parts of the full unit and all such units are needed in a typical operation. The various units needs to be implemented. They could be pipe-lined. This technique is further described below.

The structure of the arithmetic

The arithmetic of an arithmetic unit is assumed to be built by gates using boolean algebra. The unit consists of a net of such gates directed towards an output. There are paths from the output to the input. It is said to have a *depth* equalling the number of gates passed. Different paths may have different depth.

Assume a drawing starting from left continuing to the right where there is one column for one level of depth. The deepest path has one gate placed in each column. The other paths have gates placed in proper columns. For such paths gates may be moved left/right without any change in function.

Figure 1 Pipe-lining within arithmetic units.



This is the general view of an arithmetic units. Typical arithmetic units may have between 10 and 200 columns.

The gates

Each gate has a *fan-in* being the number of wires connected to the gate as inputs and a *fan-out* being the number of gates connected to an output.

In CMOS technology each gate could be considered one switch network connected to V_{dd} using p-channel devices and one network connected to *ground* using n-channel devices. Each such network may have the state *closed* or *open* depending on the input. It is assumed that the two networks are not closed at the same time. However they may be open at the same time.

If they are open at the same time a dynamic state is introduced. In large VLSI-circuits they are generally not used because of test problems. This analysis considers both cases.

Area is consumed by each implemented gate. The analysis counts area by counting scaled transistors.

Energy is consumed by charging a node to V_{dd} . Subsequently this energy is discharged to ground by another state change. From the charge through a discharge until the next charge there is a time period. The longer this period is the less power is consumed (energy is the same). A node having a capacitance c_{node} consumes the energy

$$E = c_{node} \cdot V_{dd}^2 \quad 4$$

during the mentioned time. The time is at least two clock cycles. The mean power dissipation is depending of the mentioned time. In a particular arithmetic unit there is a probability of being in a 1 or 0 state. For arbitrary nodes the probability is 50%, but may differ for arithmetic nodes (e g an overflow node in an adder may not at all switch). Energy is therefore assumed to be switched with the relative frequency η . The energy consumption of a node with the capacitance c and the power voltage V_{dd} is

$$E = c_{node} \cdot V_{dd}^2 \cdot \eta \quad 5$$

Race conditions may cause a node to fluctuate until a stable state is reached. In this analysis such fluctuations are ignored.

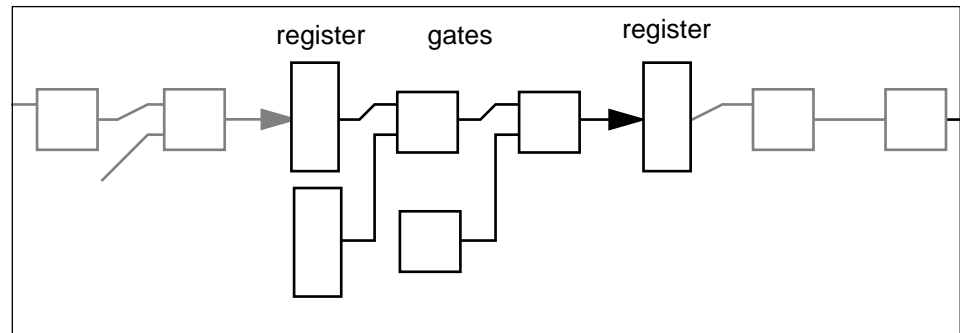
If the input signal has a slow rise-time there could be a considerable current through simultaneously conducting p- and n-channel devices. If the rise-time almost equals the rise-time of a flip-flop the switching behaviour could be normalised. The energy consumption due to concurrent conduction could be estimated as a particular fraction of the energy due to capacitances. Thus it could be included in the model by adjusting the node capacitances.

Modern technology using low voltage transistors may have transistors that always conducts. Thus there is a continuous conduction through the p- and n-channel devices, This conduction is general a magnitude or more less than the normal switching conduction. Below the switching frequencies are very high. Therefore the normal conduction is dominating and the effect could be neglected.

6.1 Implementation of optimal arithmetic

The multiprocessor is assumed to consist of a network, memories and arithmetic units. The arithmetic units are considered to consist of a pipe-line of stages. Between stages and at the input and the output of an arithmetic unit there are registers.

Figure 2 An arithmetic unit is implemented with a combinatorial set of gates. Adding pipe-line within the gates increases utilisation and speed. A stage from one register to another one is shown. It consists of a number of gates and a register.



There may be one or several types of arithmetic units within the set of operators OP depending on the DAG. Of the particular arithmetic operator op there are N_{op} arithmetic units. Thus the total amount of arithmetic units is

$$N_{alu} = \sum_{op \in OP} N_{op} \quad 6$$

The area of the arithmetic unit for operator op is A_{op} . The total area of all arithmetic units is

$$A_{alu} = \sum_{op \in OP} N_{op} \cdot A_{op} \quad 7$$

The multiprocessor executes a DAG with a real time cycle T_{cyc} . The DAG contains n_{op} instances of the operator op resulting in an execution frequency:

$$f_{op} = n_{op}/T_{cyc} \quad 8$$

An arithmetic unit starts new arithmetic operations with a **scheduling interval** t_{op} . By assuming that the arithmetic units are loaded during the entire real time cycle the resulting area of all arithmetic units is estimated to

$$A_{alu} = \sum_{op \in OP} f_{op} \cdot A_{op} \cdot t_{op} \quad 9$$

This execution indicates that the area of the arithmetic units are directly proportional to the operand execution frequency. It is also proportional to the scheduling interval of each arithmetic unit.

The DAG is given and thus f_{op} . In order to reduce cost the area A_{alu} should be reduced. Therefore either the area of a particular type of arithmetic unit should be decreased or the scheduling interval should be decreased.

6.1.1 Choice of implementation of operator

An operator has its definition. There may be several implementations of this operator. Therefore the choice of implementation is important. The area time product $A_{op} \cdot t_{op}$ should be reduced. Alternatives could be units with small area and long time or large area and short times. Thus the consideration of using a parallel multiplier or several sequential ones should be decided upon the area time product (see paragraph 3.2 on page 7).

For complex arithmetic as floating point there are many alternatives to be investigated. Such complex operators could be broken down into some common operations as normalization, alignment, add, multiply, scale etc.

It is not within the scope of this report to find a proper arithmetic operator implementation. It is just assumed that there exists such a one.

6.1.2 Increasing the scheduling rate

For a given operator the scheduling interval could be reduced by using pipe-lined arithmetic. The arithmetic unit is implemented as a number of stages. Between each stage a register is inserted. The width of such a register corresponds to the width of the DAG at that particular point.

In the analysis below it is assumed that each stage has a gate depth of d_{stage} . Because of the introduction of registers the affective area of the arithmetic unit increase by the registers (see paragraph 3.4 on page 8). The scheduling interval is decreased to the sum of the gate delays of the stage and the propagation delay and setup time of the register. As a consequence the energy consumption is increased because of the registers.

Assuming that the fan-in and fan-out are n_{fanin} and n_{fanout} , the scheduling interval can be estimated to

$$t_{op} = [2 + n_{fanout} + (d_{stage} - 1) \cdot n_{fanin} \cdot n_{fanout} + (n_{fanin} + 1)] \cdot 2 \cdot \tau \quad 10$$

Here the first term is due to the delay in the register, the second term for all intermediate gate nodes and the last term is due to the last gate and the clock gate of the input stage of the register. This function is plotted in Figure 3 where n_{fanin} and n_{fanout} are equal to 2.

The register is assumed to have an area A_{reg} . The arithmetic unit for operation op is assumed to have the area A_{op} . It has a gate depth d_{op} . The number of stages are

$$n_{stage} = d_{op}/d_{stage} \quad 11$$

and the total area is

$$A_{op,pipe} = n_{stage} \cdot w_{stage} \cdot A_{reg} + A_{op} \quad 12$$

The first term is due to the registers and the second term the arithmetic. w_{stage} is the width of the register. Inserting this in equation 9 results in

$$A_{alu} = \sum_{op \in OP} f_{op} \cdot (d_{op}/d_{stage} \cdot w_{stage} \cdot A_{reg} + A_{op}) \cdot t_{op} \quad 13$$

In this equation t_{op} is taken from equation 10. For a particular DAG all parameters are constant except d_{stage} . By varying d_{stage} the total area for an operator op may be varied. By subsequently reducing d_{stage} from d_{op} down to 1 the scheduling interval is decreased and the register area is increased. There is a rather flat optimum near 2 as shown in Figure 3.

The implementation of a pipe-line uses the same gate structure as the DAG but inserts register. The switching energy of one stage is

$$E_{stage} = [1 + n_{fanout} + (d_{stage} - 1) \cdot n_{fanout} + 1] \cdot \eta \cdot 2 \cdot e + n_{cl} \cdot e \quad 14$$

Here n_{cl} is the number of transistors in the clock controlled switches. The first two terms are due to the internal node of the register and the output stage of the register. The third term is due to intermediate gates and the fourth the last gate. The fifth term is for the clock controlled switches.

An operator op contains g_{op} nodes. Using 14 for pipe line results in the power dissipation for arithmetic:

$$P = (1/T_{cyc}) \cdot \sum_{op \in OP} n_{op} \cdot g_{op} \cdot E_{stage}/d_{stage} \quad 15$$

6.2 A reference implementation

In order to have a reference to which other implementations can be compare, an intuitive such should be defined here. It cannot be implemented.

It is based on a pipe-line with a stage depth d_{stage} being 1. All influences in the characteristics due to the registers are ignored. The scheduling interval is assumed to correspond to one gate delay. Equation 10 is therefore changed to

$$t_{op} = 2 \cdot \tau \cdot n_{fanin} \cdot n_{fanout} \quad 16$$

Equation 9 is therefore changed to

$$A_{ref} = \sum_{op \in OP} f_{op} \cdot A_{op} \cdot n_{fanin} \cdot n_{fanout} \cdot 2 \cdot \tau \quad 17$$

This area contains only parameters depending on the DAG and the technology. The area is much smaller than what is feasible. Generally this area is calculated for only one type of operator.

During a real time cycle η fraction of the nodes switches a complete cycle where each switch consumes the energy $2 \cdot e$. Thus the power dissipation of the arithmetic of a DAG is

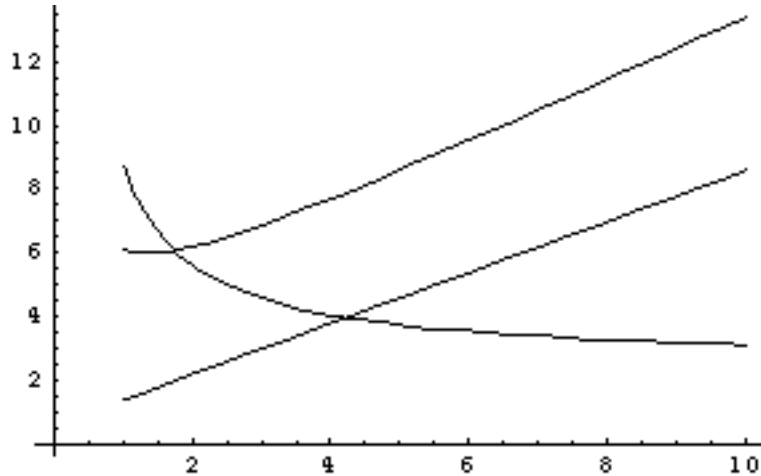
$$P_{ref} = (1/T_{cyc}) \cdot \sum_{op \in OP} n_{op} \cdot g_{op} \cdot \eta \cdot n_{fanout} \cdot 2 \cdot e \quad 18$$

This power consumption is reached if the implementation is performed by separate arithmetic units connected as the DAG. The same power dissipation is consumed by the reference implementation.

6.3 Selecting an optimal implementation

In Figure 3 the performance of area and energy dissipation is plotted as function of the stage depth. The figure shows continuous values of depth, but implementations could only be performed on integer values.

Figure 3 Clock cycle, energy dissipation and area of pipelined arithmetic relative a reference implementation as function of the stage depth d_{stage} . Based on $n_{fanout}=2$, $n_{fanin}=2$, $n_{cl}=6$, $A_{reg}=10$, $A_{op}/d_{op}=4$, $w_{stage}=1$. The clock cycle time is divided by 10.



By increasing stage depth power dissipation decreases and area increases. The area has an optimum between 1 and 4. The minima is rather flat. Assuming that $E \cdot A$ is a quality measure. This function is minimal for 4. The area is 7.7 and power dissipation 4.1 relative the reference.

The clock cycle is 38 gate time constants. In a contemporary technology it corresponds to 5-20 GHz clock frequency.

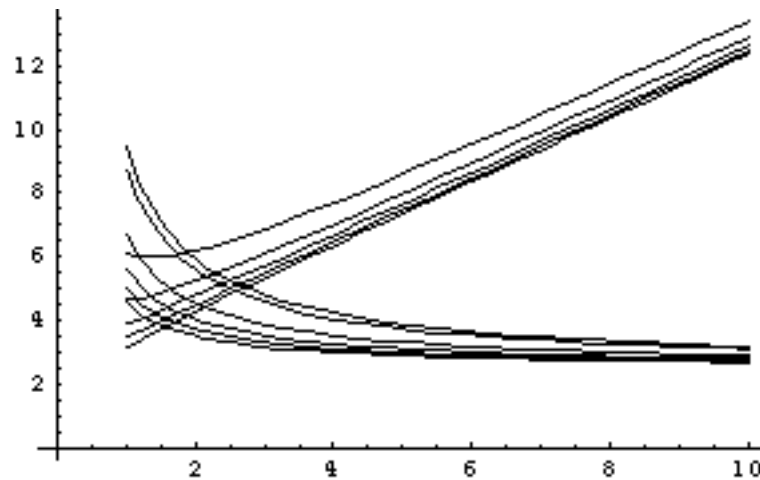
These figures are based on the parameters stated in the figure text. Actual operators may have other values.

6.3.1 Fan-in and fan-out

Fan-in and fan-out are parameters to the scheduling interval t_{op} and behaves similarly. An increasing fan-out or fan-in changes the reference implementation approximately the same. In Figure 4 the n_{fanout} is a parameter. A higher n_{fanout} reduces both area and power dissipation. The area optimum for d_{stage} decreases.

In arithmetic units fan-out is generally rather small. A typical look-ahead adder has the highest n_{fanin} being 5 and the mean value is 1.7. The corresponding figures for n_{fanout} are 5 and 1.8. Therefore the choices of 2 are typical.

Figure 4 Area and energy dissipation as function of d_{stage} . Parameter is n_{fanout} . Other parameters as in Figure 3.

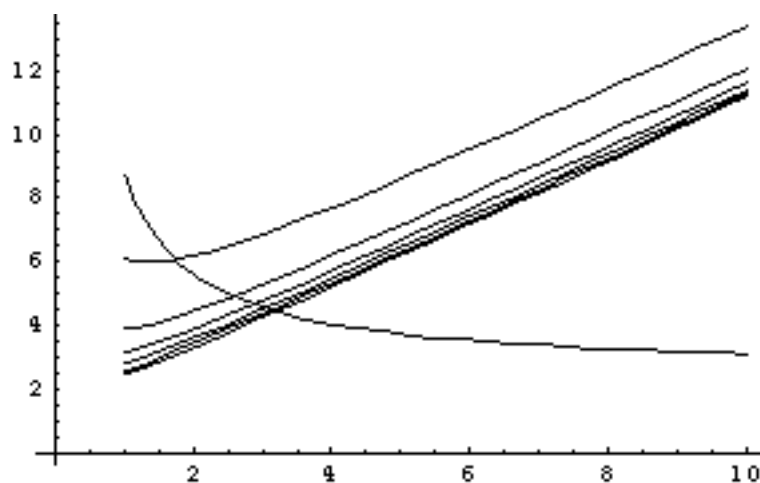


Area, depth and width of the operator

A_{op} , d_{op} and w_{stage} all influences one and the same term in equation 13. Together they specify the relative area of the register compared with the area of the stage arithmetic. They can all be analysed but only one of them need to be varied. In figure Figure 5 the area A_{op} is parameter. The energy dissipation is not depending on the parameter. The area decreases with an increasing area of the operator. The optimum for d_{stage} decreases.

At small values of d_{depth} it is not likely that the number of arithmetic gates would be much larger than 1. A higher value however increases relative performance. Thus the chosen value is typical.

Figure 5 Area and energy dissipation as function of d_{stage} . Parameter is A_{op} . Other parameters as in Figure 3. The relative area of A_{op} is increased from 1 to 6.



6.3.2 The register

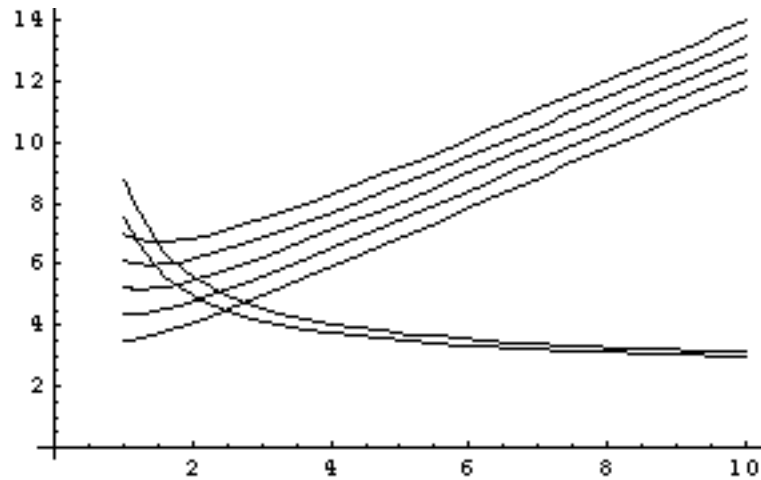
The register implementation is significant. In Figure 6 the area and power dissipation is drawn with the parameter A_{reg} . The register area is swept from 4 to 12 transistors.

An increasing amount of transistors increases both area and power dissipation. Testing methods may cause the number of transistors to be increased. Technology may decrease the number of transistors. Advanced switching methods may also decrease the number of transistors (see paragraph 3.4 on page 8).

It is not likely that the dual inversion could be eliminated, however the clock switches could be integrated into the gates. Thus the two inverters of the register could be eliminated. The clock gate transistors remain. If the registers are dynamic the number of transistors are 4.

The choice of 10 transistors is therefore conservative. Probably should the best alternative be feasible, reducing the relative size considerably.

Figure 6 Area and energy dissipation as function of d_{stage} . Parameter is A_{reg} . Other parameters as in Figure 3. The parameter is 4, 6, 8, 10, 12. In the case of 4 n_{cl} is 4.



In CMOS technology it is assumed that at least a pair of n- and p-channel devices is needed for a clock switch. The minimum number of phases in a switched logic is 2. Thus the minimum n_{cl} is 4. It is used in the best alternative in Figure 6.

6.4 Discussion

The arithmetic could be implemented very efficiently. Above it was shown that the area could be as low as 4.1 for an implementation with stage depth of 2 and smallest possible register. The power consumption is 5 times the reference implementation. Changing the stage depth to 4 gives 5 and 3.8, respectively.

By using a more conservative static register with 10 transistors and stage depth of 4 the area is 7.7 and power dissipation 4.1.

The pipe-line increases clock speed far up in the GHz range. It could be argued that this is a too high frequency. As shown, from power point it is not. From synchronisation point of view the circuit size is at least a magnitude smaller than current microprocessor circuits. Probably the area is much less than 1 mm².

The implementation of the register is crucial. The power dissipation would be around 4 and the area in the range from 4 to 8. These figures should be compared to those for a conventional implementation in microprocessors using the stage depth being the entire arithmetic unit. For such an implementation the area is 63 and power dissipation 2.6.

The massive implementation could be made a magnitude smaller than in contemporary microprocessors!

7 Storage - registers and memories

In a processor the real-time state has to be stored. It is assumed that there are much more nodes in a DAG than there are nodes as input and output in the DAG. therefore the real-time state is ignored.

Storage is only needed for input, output and intermediate nodes of the DAG. The primary storage is registers.

Input are read once during a real-time cycle. Output nodes are written once. Intermediate nodes are written once and read as many times as there is fan-out of the node. Typical fan-out are 1-2. there are however examples of nodes with fan-out greater than 1000.

A register contains as little as two internal circuit nodes and consumes only power when written! It is impossible to use less energy (see paragraph 3.4 on page 8). The register contains a4-10 transistors of a significant size. In order to reduce the area for an implementation of a DAG the amount of registers and the area of a register should be reduced. Scheduling can be used to minimize the amount of registers. Thus a physical implementation has to reduce the size of the register.

The usual way to solve this is to minimise the unique parts and share the remaining parts. The registers then results in a SIMD being a memory. It consists of a network (a bus), the bit lines of the memory, and memory cells. Thus the memory cells and memory peripheral could be considered resources to be scheduled together with the arithmetic and communication.

There may be memories of several types. Memory capacity, static power and energy consumption for the accesses are to be optimised. Generally there are some few memories storing many words and many small high throughput memories.

7.1 Memory structure and hierarchy

Let us assume that the memories are resources, bus and memory cells, connected to a network The interface between the bus and the network is always through a register. Thus all communication in the multiprocessor is through registers. The registers have to be schedule in conjunction with memory accesses, communication, and arithmetic operators.

A memory contains memory cells. Nodes in the DAG can be scheduled and allocated in several different schemes as:

- **direct**, a value of a memory cell is transferred to a register, communicated to a new register and then used as input to arithmetic.
- **indirect**, a value of a memory cell is transferred to a register, communicated to a new register and then written into another memory cell (generally of a more powerful memory). This access is used in a memory hierarchy and queues.
- **cache**, a node has a home place in a memory cell. Transfers are performed as in an indirect case, but the use of the value always results in a transfer from one memory to another and generally back again.

In the direct case there is only one write access and n_{fanout} read accesses to memories. The total amount of accesses is

$$n_{acc} = 1 + n_{fanout} \quad 19$$

In the indirect case there is generally only one write access when writing a node in the DAG. The read access consists of one read of an upper memory and one write to a lower memory. This procedure is repeated ($n_{hier}-1$ times) for each memory in the memory hierarchy. Finally, the lowest memory is read. Transfers from an upper to a lower memory may be limited to one transfer. The total amount of accesses is

$$n_{acc} = 1 + n_{fanout} \cdot (1 + (n_{hier} - 1) \cdot 2) \quad 20$$

In a cache memory the write operation causes a transfer from an upper memory to a lower, write in a lower, and a transfer back to an upper memory. A read access causes a transfer from an upper memory to a lower memory, and a read of a lower memory. The cache could have a hit ratio η_{hit} causing less transfers during the read operation. If the cache block length is w_{mem} .

$$n_{acc} = (1 + n_{hier} \cdot 4 \cdot w_{mem}) + n_{fanout} \cdot (1 + (n_{hier} - 1) \cdot (1 - \eta_{hit}) \cdot 2 \cdot w_{mem}) \quad 21$$

Caches and memory hierarchies introduces many more accesses to memories than necessary, especially a cache. Assume that the fan-out is 2. A node in the DAG then corresponds to 3 accesses for a simple memory, 7 accesses for a two level hierarchy and 7-12 for a cache. The low memory has 3, 5 and 5-7, respectively.

Memory hierarchies may reduce latency time but not the number of accesses. In a massive application where nodes can be scheduled and allocated a memory hierarchy does not help!

The allocation may allocate nodes with long life-time to memories with long latency time. The application as such, i.e. the DAG, defines the number of accesses and where it is possible to allocate the nodes.

7.2 Memories

Conventional memories use a micro architecture being inherited for generations. All characteristics have been tuned to standard microprocessors. The memories for massive multiprocessing must have other characteristics.

Memories are used in order to

- 1 reduce the number of arithmetic units. A memory cell stores a node of a DAG in order to time space multiplex the arithmetic units. This is done in order to reduce the size of an arithmetic unit to generally one memory word.
- 2 reduce the size of a register and its network to one memory cell.

From power point of view the use of memories are always worse than what they substitute for both cases 1 and 2. From area point of view they are superior. Area cannot be reduced below the size of the memory word. Scheduling may allocate several nodes of the DAG to one memory word. Therefore the properties of the DAG defines a minimal area.

7.3 Many storage banks

The arithmetic units have a considerable speed pacing the speed needed by the DAG. Equations 19-21 indicates that there are at least 3 memory access per arithmetic operation. The memory bandwidth must match this speed.

In the current technology the needed bandwidth may be 10th of GHz per arithmetic unit.

7.4 Memory micro architecture

There may be many types of memories in a multiprocessor. The significant memories are those on a lowest level. Therefore a low power high speed memory is discussed in this section.

7.4.1 The memory cell

The memory cell is the essential cell giving the performance. Because of the speed characteristics the only possible cell is a static register cell. There are various implementation of this cell. The most common is 4 transistors and two resistors.

Generally all these cells are very fast. Therefore the important characteristics is the physical size. These cells have all a negligible power consumption.

Reading

Because the cell should be very small, the available currents from the cell is rather small. The only available network devices are two transistors feeding the state to a bus. The currents available are so low that the voltage switching speed on the busses are almost not present. Of that reason the currents are measured.

The current goes from ground through one driving transistor of the memory cell, the coupling transistor, to the bit-line bus, charging the bus capacitance and to a sense amplifier reading the bus voltage.

The speed of this circuit is crucial.

Normal amplifiers are power hungry and slow. The best amplifiers are short circuited flip-flops that are allowed to swing free from a small unbalanced state to a saturated state (see paragraph 8.1.5 on page 34). Because the current is low, the energy dissipated is low.

Writing

Writing of a memory cell is to force a state onto the flip-flop. Because of the asymmetry of the flip-flop the V_{dd} voltage applied to the bit-lines do not switch the memory cell. Instead the partly short circuiting one node inside the memory cell to ground initiates the switching. This means that both bit-lines have to be switched, one to ground and one to V_{dd} .

The bit-lines have a large capacitance due to both wire capacitance and drain diffusion capacitance of the coupling transistors. Due to this large capacitance the writing dissipate much energy. The number of words of a bit-line has to be kept low in order to reduce the energy dissipation.

The capacitance of the internal nodes of the memory cell is small. The memory cell is therefore fast. The time delay of the writing operator is caused by switching the bit-lines.

Addressing

Addressing is performed by switching the voltage high on one the word-line controlling the coupling transistors of the memory cell. The capacitance is given for a memory cell, two gate loads. Rise-time should be at least moderate in order to get short read and write delays.

The word line has the capacitance of an entire word. Thus it is considerable. However it is twice the smallest capacitance possible. The energy consumption is therefore at most a factor of two from the theoretical minimum. It is in the same range as for a normal register.

The bit cell dimensioning

As described above all characteristics of the bit cell is good except the writing characteristics. It is heavily depending on the number of words and the technology for the drain region of the transistors. The number of words could be limited by architecture design. The drain capacitance could only be controlled by choosing an appropriate technology.

7.4.2 Pipe-lined memory

The pipe-line has to be designed to increase throughput as much as possible. The shortest scheduling interval is define by how short the read or write cycle can be done.

Registers may be placed on each word-line and bit-line. Writing cannot be made shorter. By placing the sense amplifier as a switch using the small offset at the clock boundary and swinging out during the next cycle would create an equally short read cycle.

The bit line registers must be controlled. Some type of control signals control them. They must be there and they will influence the power consumption. Because the word length is short, it is reasonable to have such registers.

Having a register on each word-line, however, causes a lot of energy to be dissipated in clock gates. In order to reduce the energy consumption the gate has to be moved before a decoder decoding all or parts of the total decoding.

Therefore the following structure of memory is suggested (see figure Figure 7 on page 26):

- 1 the address bits are latched in a register. It has normal and inverted outputs with little skew.
- 2 The address bits are divided into dim number of address parts. Each such part is decoded by a decoder. Each decoder has $adrl$ input bits and an output of ${}^2\log adrl$ bits. They are latched in normal registers.

Each bit-line has a latch for data to be written.

- 3 each word line consists of one address decoder and a driver stage feeding the word line. The decoder is an AND-gate with dim inputs. The driver stage may consist of 0, 1, or 2 cascaded inverters.

a number of bit-lines are grouped to a bank. There is a switch between the bit-lines and a bit-line bank bus

each bit-line bank has a driver to the intermediate bit-line bank bus.

the intermediate bit-line bank bus connects to the input stage of a sense amplifier.

- 4 the sense amplifier consists of a short circuited flip flop. The output is latched in registers for the read data.

A write operation uses stage 1 to 3 and a read operation all stages.

There are probably many ways to design the memory, however it is not in the scope of this report to evaluate them or find the best one, instead a good choice is to be used. Below this design should be analysed.

stage 1 - address register

This stage consists of a normal register with inverted outputs and feeding a number of AND-gates.

stage 2 - decoders and register

The address is typically divided into three parts, one decoding the banks of the memory and the other two for a two dimensional addressing of a word-line.

The decoder has an additional input being an enable signal for access. The decoder decodes all states for each address part.

The output is to a register.

The number of AND-gates are of course much less than there are word-lines. However, the number is great and therefore skew should be as little as possible not causing the gate outputs to flicker. Therefore the register of the earlier stage should have outputs with little skew!

Figure 7 The circuit diagram of the memory. Not complete.

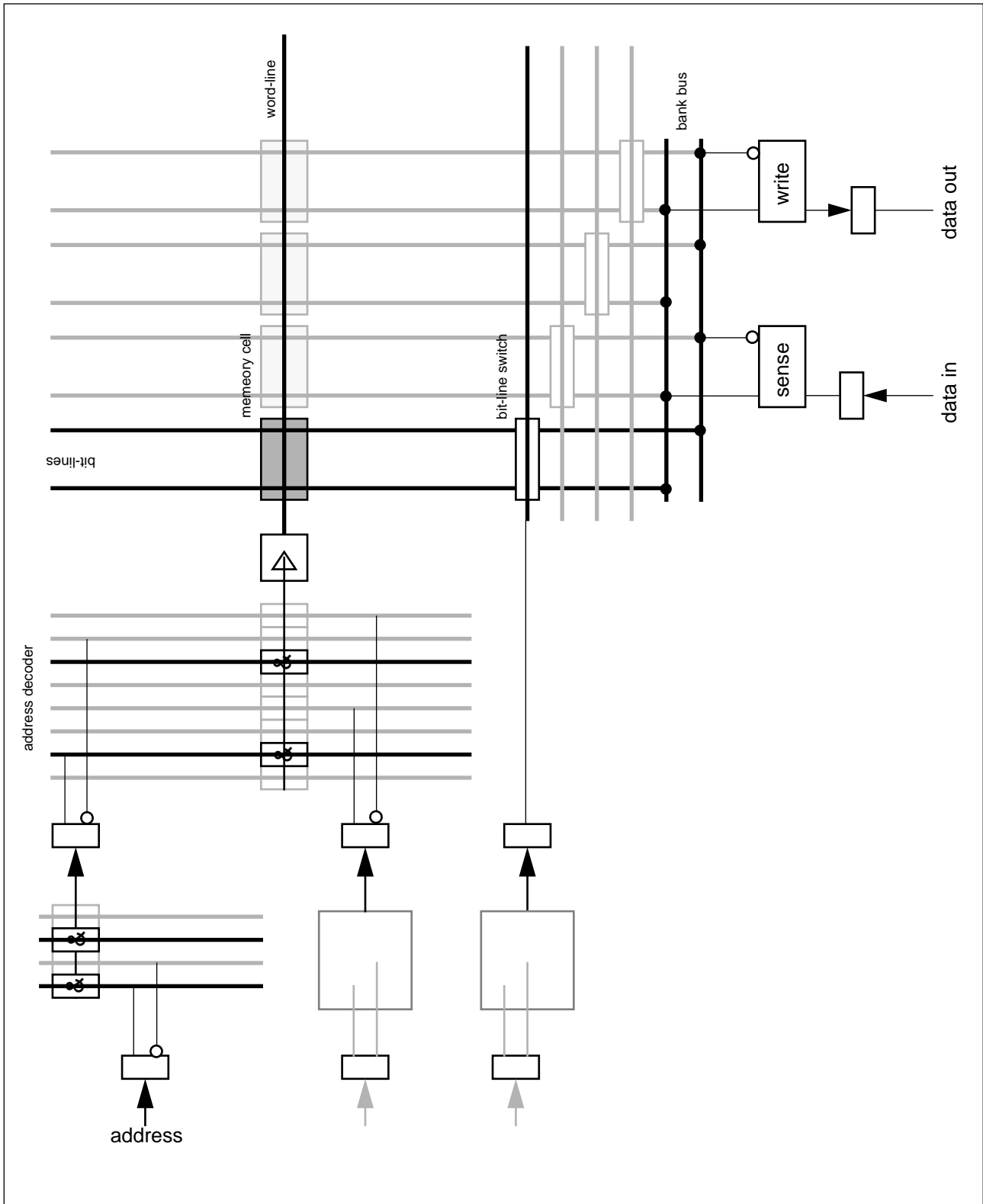
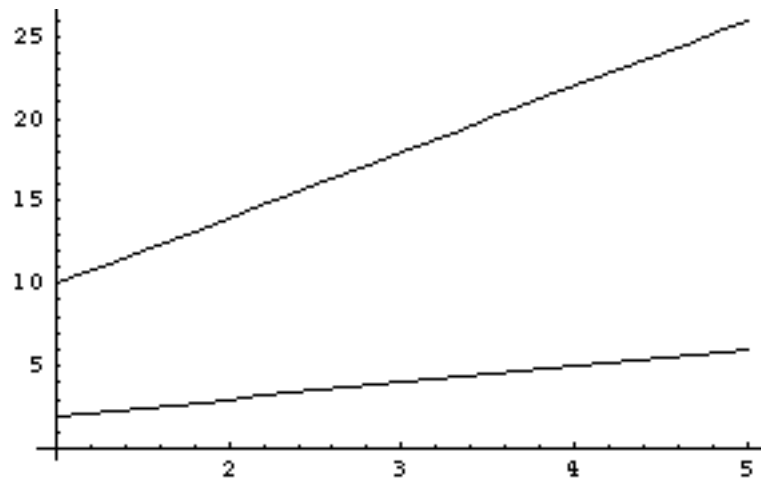


Figure 8 The energy dissipation and latency time of stage 2. It includes output from the address register, the decoder and the input to the next register. On the horizontal axis the length of the part address n_{bit} shown.



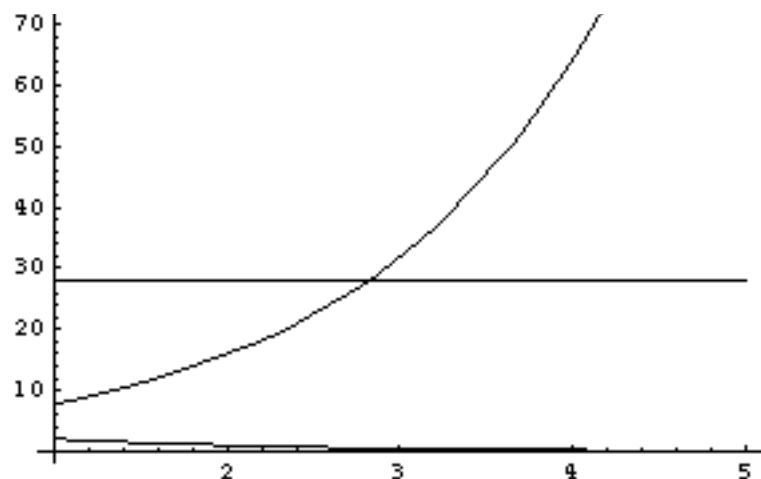
stage 3 - address decoder

In this section only the address decoder part of stage 3 is discussed. It consists of a number of word rows. In each row there is one AND-gate and one driver.

In conventional memories the address decoding is binary using n address bits with $2n$ address wires, each having 2^{n-1} transistors connected. Total amount of switching transistors is $2 \cdot n \cdot 2^{n-1}$. The power dissipation is proportional to this figure. In order to reduce the power dissipation the decoding is made two dimensional. In such a decoder there are $2 \cdot 2^{(n/2)}$ switching transistors, which is much less.

The address decoder is analysed in Figure 8. The latency time is almost not depending on the number of words. The energy dissipation increases by the number of words but is low. The energy dissipation per word falls. The energy per word is generally much less than 1.

Figure 9 The energy dissipation and latency time of stage 3. On the horizontal axis the length of the part address n_{bit} shown.

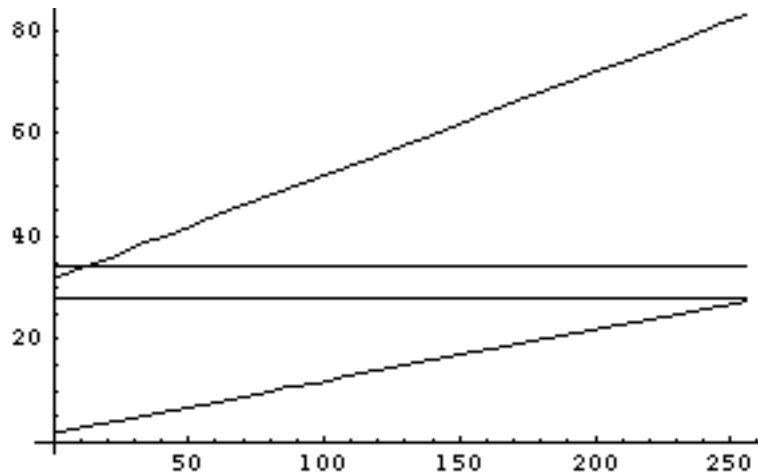


stage 3 - writing/reading

The bit-line circuits are used for reading and writing. The circuits are almost size independent, only the transistors of the bank switch and the write transistors may be scaled.

In Figure 10 the reading and writing characteristics are shown. The read time and the energy dissipation is constant. For write the write time and the energy dissipation is proportional to the number of words.

Figure 10 The energy dissipation and latency time of stage 3 for reading and writing. On the horizontal axis the number of words in the memory is shown.



7.5 Discussion - memory characteristics

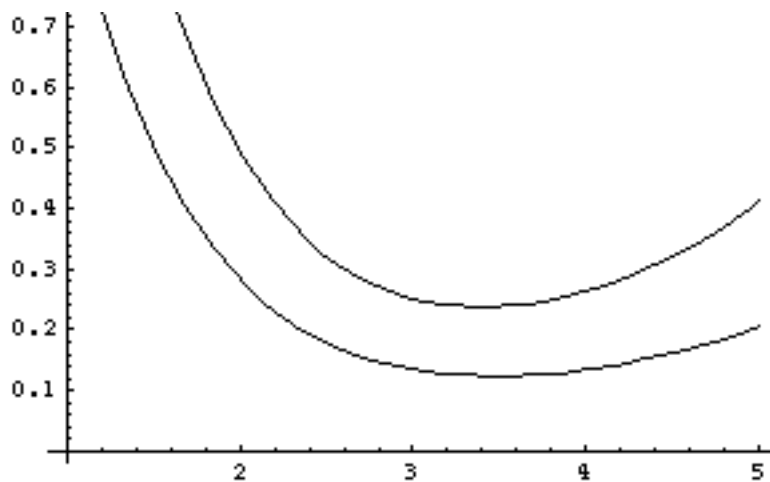
area

The memory contains memory cells and other circuits. The most important characteristics of a memory is how small the memory could be made. The memory cells are given by the technology. Therefore the fraction of non memory cells should be low. This fraction is plotted in Figure 11 as a function of the memory size for the memory described earlier (see paragraph 7.4 on page 23).

The diagram shows implementations from 4 words to 2048 words, each being 32 bits. Above 16 words the overhead is very low. The figure depicts ratio of transistor counts. The memory cell is denser than the peripheral logic, but not more than 1.5 times. The two curves shows memories with 1 or 2 banks.

The physical area of the largest of these memories is in the range of 0.1mm^2 . It is small and uses areas so efficiently. Many such memories could be used to implement the multiprocessor memory system.

Figure 11 The relative area of non bit cells in the memory compared to all memory cells as function of the number of stored words. The memory is described in the text.



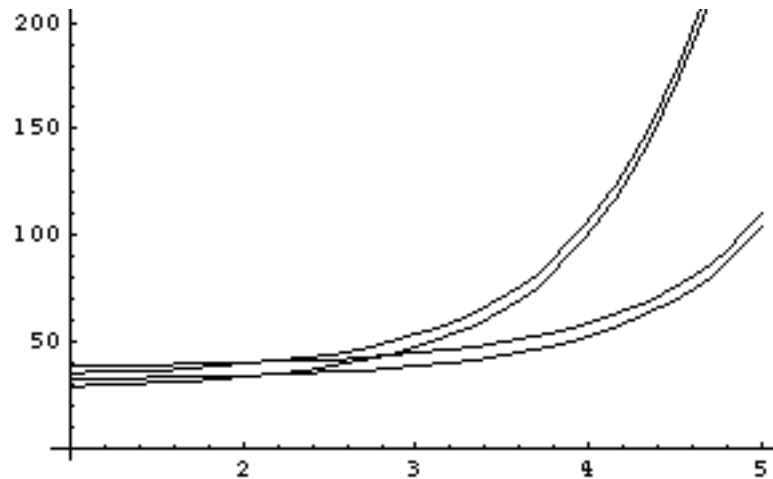
energy dissipation

The memory is accessed about 3 times per node of the DAG (see figure 7.1 on page 22), 1 write and 2 read operations. In Figure 12 the energy consumption is plotted as function of the memory size. As seen from the figure large memories consume more energy per access than smaller ones. Therefore it is questionable whether memories should be larger than 512 words. This figure is heavily dependant of the technology for drain diffusion of the memory cells.

Such a memory consumes 225 energy equivalents per bit for $1w+2r$ accesses.

This figure should be compared to the energy dissipation of an arithmetic unit bit. Earlier it was shown that the typical arithmetic energy consumption is in the range of 8 energy equivalents per gate. The figure above corresponds to 28 gates. The bit slice of an arithmetic unit is much larger. The memory is therefore not dominating the power consumption.

Figure 12 The energy dissipation for reading and writing of one bit as function of the number of stored words. The memory is described in the text.

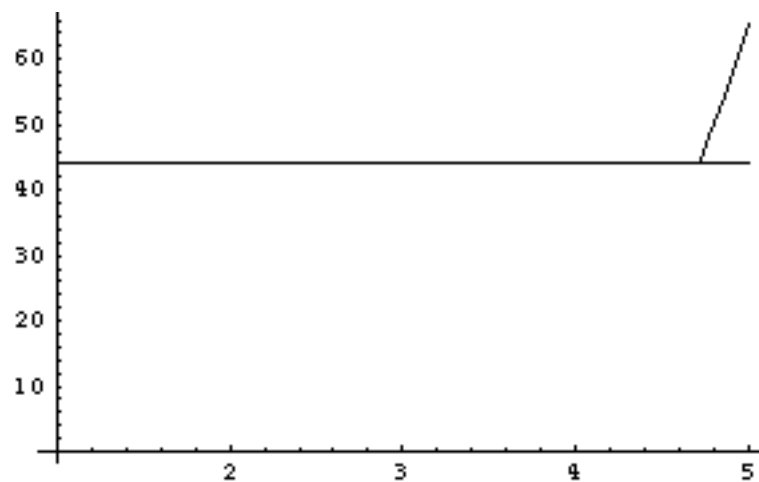


scheduling interval

The scheduling interval for the memory system should pace the speed of the arithmetic units. As shown earlier the memory system needs a throughput of more than 3 times the arithmetic units.

In Figure 13 the scheduling interval is plotted for the memory as a function of its size. As shown the read access is almost constant. However the write operation increase at large memories. For a 512 word large memory this interval is 44 time constants. It is approximately the same figure as the arithmetic unit (see paragraph 6.3 on page 19). Therefore at least 3 times more memories must be used than arithmetic units. Typical DAGs may need a much larger capacity in order to store applicable nodes. Thus the throughput is not an issue.

Figure 13 The scheduling interval of a memory as function of the number of stored words. The memory is described in the text.



Conclusion

In order to keep energy dissipation low the number of words in a memory should be limited. The amount is low. In order to cover the storage needs there are many memories, much more than 3 per arithmetic unit. Therefore the scheduling rate for the memories are adequate.

Memories could not be made to small because then the memory peripheral circuits will dominate. Therefore there is a window in storage capacity proper for the memory units. This window is probably 32-512 words.

In this section a “special design” was used. Other designs would exhibit the same type of characteristics, however with different figures. The energy for write is too large in contemporary memories.

8 Communication - network and links

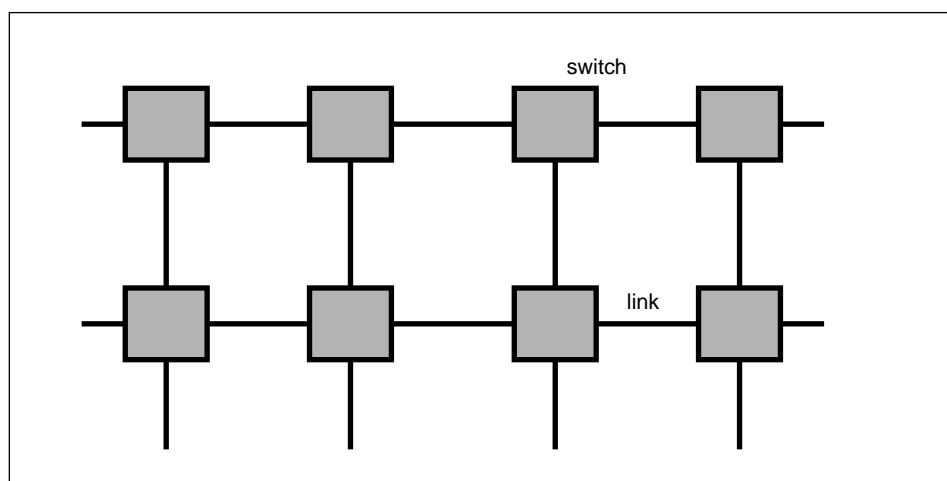
A processor consists of a number of units like memories and arithmetic units. They are connected by a network. This network may pass edges of chips. It is not necessary in this analysis to divide between chip local transport or inter chip transports. However, it is important to understand the physical links.

The network has a topology. Generally this topology must be rich in order to cope with several types of DAGs. It is not within the scope of this report to design the topology. Here it is just necessary to understand that the network consists of a number of communication nodes connected by communications links.

The network consists of communication **switches** and communication **links**. They may, as in the case of a bus, be integrated to one unit. Because the bus is a very special case of a communication link it is not treated here. Instead it is assumed that it has the same characteristics as a corresponding communication link.

Allocation and scheduling mechanisms allocates a bit to a particular communication link during a period. There are numerous schemes to do this: two traditional schemes are either bit serial and bit parallel. However, it is not at all necessary to define this transport here, just to understand that a bit is transferred.

Figure 14 A communication network. It consists of switches and links.



communication switch

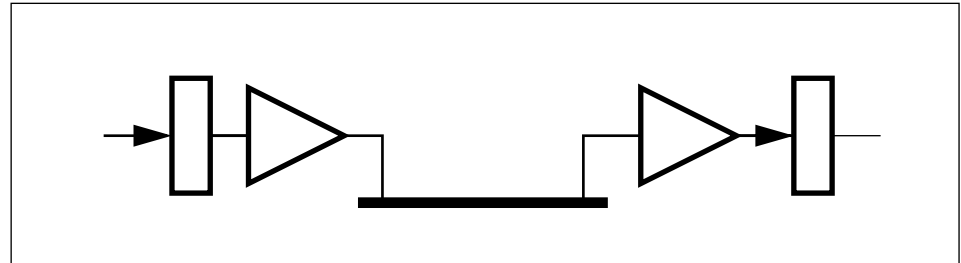
The communication switch is a device consisting of a permutation network moving an incoming bit on a communication link to an outgoing bit on another communication link. The communication switch could be regarded as a special arithmetic unit performing switch operations. Of that reason the communication link is not treated in detail.

communication link

The communication link consists of two *communication nodes*, the A and B node. They are the points connected. In each of them there is a register or a register pair, one for a received bit and one for a bit to send. Between the nodes there is a **transport medium** (in this report an opto fibre/opti-

cal system or a metal wire complex). A node has a driver, reading a register and applying a physical stimuli on the corresponding position of the transmission medium. It also has a receiver reading the transmission medium and transferring the read value to a register. The transmission may be simplex or duplex. A bus has several nodes along the transmission medium.

Figure 15 A communication link. Each node has one register for the transport, transmit and receive, a driver and receiver. Between the nodes there are wires.



Each node has its own clock. They are all synchronised for the transport. The synchronisation mechanism is discussed elsewhere (see paragraph 9.1 on page 38). During one clock cycle a transmitter transfers information from the register to the wire. During the corresponding clock cycle in the other end a receiver transfers this information from the wire to the register. The clocks have the same frequency but with a particular phase difference t_{phase} . The information has a propagation delay t_{pd} from the input to the transmitter to the output of the receiver.

The receiver is assumed to have a three phase clock. During phase 1-3 the transmitter sends data. During phase 2 the slave part of the receiver register reads the wire and during the other phases it stores the value. The master register of the receiver reads the slave during phase 1. Phase 3 is used to compensate for jitter and phase difference. It may be in some cases be skipped.

8.1 Communication links

The following links are analysed

- full signal swing on metal wire
- reduced signal swing on metal wire
- reduced signal swing on electrical transmission lines
- optical medium with current to photon and photon to current devices

The following has to be considered for each link:

- registers for single bits in the transmitter and receiver,
- an electrical driver and transmitter between the register and the transmission medium
- a receiver and a sense amplifier between the transmission medium and a registers
- clock generators in the transmitter and the receiver

Dividing the problem into these parts simplifies the analysis. The registers, driver, sense amplifier and clock generators have the same performance for all communication links. Thus they could be treated separately.

8.1.1 Full signal, metal

This type of link is the conventional wire used on chip. Because of the wire capacitance and resistance the energy dissipation is high and the scheduling interval is long at long distances [10][16][17].

The driver is a normal CMOS inverter [10]. It could be scaled to the capacitance of the wire [5][7]. A number of cascaded inverters are used [12]. It is assumed that energy is dissipated in the inverters and registers. The final inverter dissipates the wire energy. The optimum inverter scale is approximately 3.7 [5]. The energy dissipation is:

$$E = 2 \cdot E_{reg} + \eta \cdot V_{dd}^2 \cdot c_w \cdot l \quad 22$$

Where η is the fraction of switch transitions in a transmission (probably near 0.5), V_{dd} is the power supply voltage, c_w is the wire capacitance per unit length and l the wire length. E_{reg} is the energy in one register, (see figure Figure 19 on page 37).

The maximum switching speed is assumed to correspond to either the time constant of the wire or the propagation delay of the driver, (see figure Figure 20 on page 37).

8.1.2 Reduced signal, metal

This type of link is a low voltage current driven wire used on chip. Because of the wire capacitance the wire voltage is low, generally in the range of 10-100's of mV. In order to cope with this low voltage a pair of wires is used, but only one is necessary. below one of several possible designs are analysed.

The driver consists of an inverter and two p-channel drivers being clocked. The drivers feed one of the wires with a current during phase 2. There are also two n-channel transistors used to discharge the wires to ground during phase 1.

One n-channel transistor is used at the receiver end to short circuit the two wires during phase 1.

There are two n-channel transistors use as transmission gates to feed the sense amplifier input at phase 2. This value is kept during the next phase.

A sense amplifier consists of a short circuited flip-flop. During phase 2 it is unpowered and the internal nodes are short circuited. During phase 1 the flip-flop is powered. In an initial state the internal nodes have the same potential, however there is a small offset on the nodes due to the input to the sense amplifiers. The small offset swings out exponentially to the saturated value V_{dd} . The output from the flip-flop is buffered.

The energy dissipation is:

$$E = 2 \cdot E_{reg} + E_{sense} + 2 \cdot \eta \cdot V_{dd} \cdot V_{th} \cdot c_w \cdot l \quad 23$$

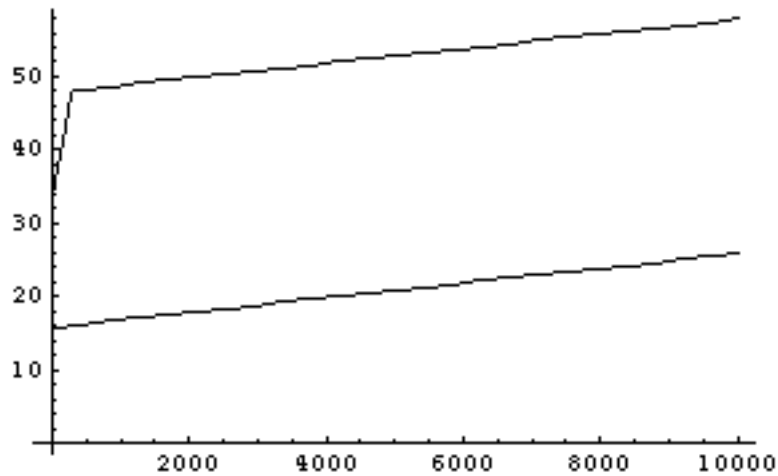
Where η is the fraction of switch transitions in a transmission (probably near 0.5), V_{dd} is the power supply voltage, c_w is the wire capacitance per unit length and l the wire length. E_{reg} and E_{sense} are the energies in a register and a sense amplifier, respectively.

The maximum switching speed is assumed to correspond to either the time constant of the wire or the propagation delay of the driver and the sense amplifier.

In Figure 16 the performance is shown for one link. The propagation delay time is short, about 16 time equivalents. For a wire corresponding to 10000 gate lengths (1.5mm) the delay is 26 time equivalents. Because of a lot of switching within the registers and the sense amplifier the power consumption is rather high, 50 power equivalents, at a short wire. Energy consumption increases slowly by the length of the wire due to

longer switch times in the sense amplifier (see paragraph 8.1.5 on page 34).

Figure 16 The energy dissipation and scheduling interval as a function of the wire length.



8.1.3 Reduced signal, transmission line

This type of link is a low voltage current driven wire used external chip. Because of the characteristic impedance the wire voltage is low, generally in the range of 10's of mV.

The circuit resembles the preceding one except for the transmission medium. The short circuit transistors are not used and the driver is single output.

The energy dissipation is:

$$E = 2 \cdot E_{reg} + E_{sense} + \eta \cdot t_{bit} \cdot V_{dd} \cdot V_{sense} \cdot 2/Z_o \quad 24$$

Here t_{bit} is the time period for one bit, Z_o is the characteristic impedance of the transmission line. t_{bit} is the maximum value depending on transmission quality on the transmission line and the delay in the sense amplifier. There is a problem with the inductance from driver output to transmission line (see paragraph 8.1.6 on page 35).

There is a dynamic energy consumed at a bit transfer due the registers, driver and sense amplifier on a level corresponding to the preceding type. However, there is also a static current through the driver and the transmission line to ground. If the scheduling interval is increased energy is also increased, see equation 24.

Transmission lines are generally either cables or microstrips on substrates. The high-frequency characteristics depend on both the physical size of the wire and on the dielectricum. Some cofired ceramics have high dielectric losses and may limit the transmission frequency to 10GHz and about 20cm. The thickness of the conductor generally used will limit the speed and wire length to approximately the same length.

Of that reason transmission lines should be considered to be used only inside a computer with small dimensions.

8.1.4 Optical

This type of link is an optical link used external chip. There is a photo diode or laser used as transmitter and a photo diode used as receiver. The transmission media is some type of optical system, among which the normal optical fibre is found.

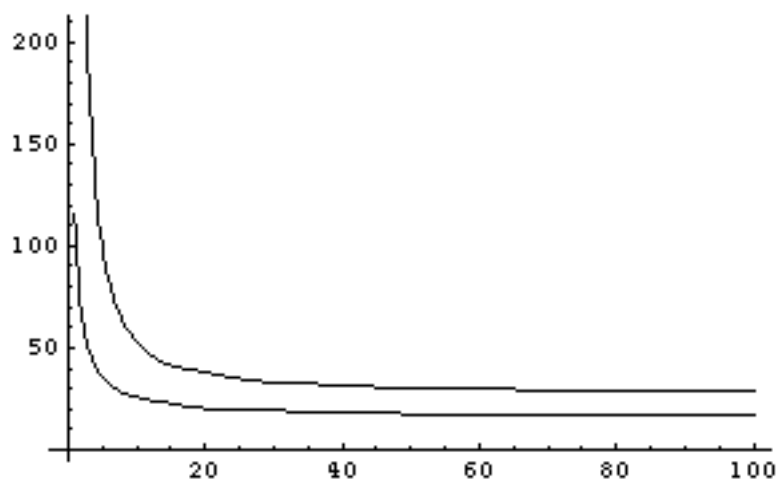
From circuit point of view the transmitter, the transmission medium and the photo diode could be assumed to be an *opto coupler* with a current gain α . The transmitter is a diode of some sort having a forward voltage of 1.2 - 1.5 V. The photo diode is a back biased diode having a leakage current being the photo current output. The circuit around these devices is therefore almost similar to the preceding type. The difference is that the sense amplifier has current instead of voltage on input.

The energy dissipation is:

$$E = 2 \cdot E_{reg} + E_{sense} + \eta \cdot t_{bit} \cdot V_{dd} \cdot V_{th}/r \cdot 1/\alpha \quad 25$$

Here r is the resistance of the transistors in the sense amplifier. There is a problem with the inductance from driver output to transmitter diode (see paragraph 8.1.6 on page 35) and the capacitance of the photo diode and the package/pad capacitance of the input stage to the sense amplifier.

Figure 17 The energy dissipation and scheduling interval as a function of the current gain α for an optical link.



There is a dynamic energy consumed at a bit transfer due the registers, driver and sense amplifier on a level corresponding to the preceding type. However, there is also a static current through the driver and the transmitter to ground. If the scheduling interval is increased energy is also increased.

If $r \cdot \alpha$ is less than Z_0 of the transmission line link the optical link is more power consuming. r is a characteristics of sense amplifier. A typical value is 10kohm if input capacitances are neglected. However if the photo diode has a considerable capacitance the transistors have to be scale reducing r . α is generally less than 1%. Therefore the transmission line and the optical one are almost the same if the capacitance of the photo diodes are very small.

If photo diodes are not placed on-chip it is likely that the sense amplifier transistors have to be scaled. Thus the power consumption is higher.

In some research it assumed that sophisticated optic devices are used to increase connectivity. Some of these devices have large optical losses. Therefore α is reduced and the power dissipation increases.

The optical system has diffraction as the main attenuation component. It is generally very low for opto fibres. Therefore optical links are the only transmission medium for long distances (>1m).

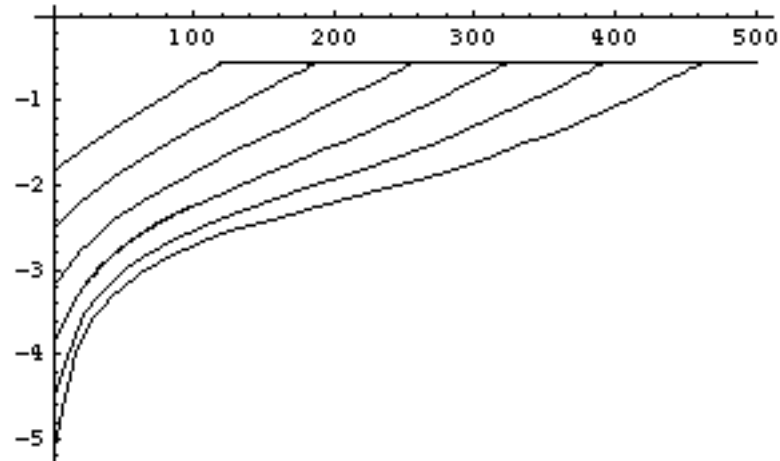
8.1.5 Sense amplifier

There are many types of sense amplifiers. If the threshold voltage is moderate the short circuited flip-flop has good characteristics [21][27]. Class

Analog amplifiers consume more power and are more complex. Therefore the first type is analysed here.

The sense amplifier consists of a flip-flop and some short-circuiting gates and current switch gates. The flip-flop is initially short-circuited between the two internal electrical nodes. By some means the nodes are set to have somewhat different voltages depending on the sense input. When the short circuit is opened, the flip-flop starts to swing against the nearest stable value. The course is almost logarithmic, i.e. very fast. Figure 18.

Figure 18 The switching behaviour for a switched sense amplifier. The y-axis shows the logarithm of the swing from the short-circuited value. The x-axis is the time. The course is almost logarithmic until the stable value. 6 curves for the initial offset 0.01, 0.02, 0.04, 0.08, 0.16 and 0.32 are shown.



The time delay is approximately:

$$t_{sense} = 0.5 \cdot r \cdot c \cdot \log(V_{dd}/V_{in}) \quad 26$$

When adding an input and output buffer, the delay is for a 4% input offset in a typical sense amplifier is 1.4 time equivalents.

The lowest input voltage is determined by the offset voltage of the input transistors. It is very dependent on the technology used. Voltages in the range of 10-40mV should generally be possible to detect [20][21][28].

8.1.6 Chip edge problems

At the chip edge there is a connection from a drain diffusion through a pad and an external connection to the external device. The external connection has an equivalent circuit being an inductance followed by a capacitance.

In the case of a transmission line or a laser/photo diode the capacitance could be ignored. The inductance is however a problem. The current should have fast rise time and the inductance limits this rise time. There is a delay that could be estimated to:

$$t_{pad} = L_{pin} \cdot I_{pin} / (V_{dd} - V_{pin}) \quad 27$$

Here L_{pin} is the inductance from transmitter to transmission line or laser/photo diode, I_{pin} is the nominal current used to the transmission medium, V_{dd} the power supply voltage and V_{pin} the voltage over the external device.

V_{pin} is about 50mV for a transmission line and 1.5V for a laser diode. V_{dd} is around 2V. L_{pin} depends heavily on the package and mounting. A pretty good value is 10nH. The transmission line would have a delay of 10ps and the laser diode 40ps at 2mA. If the current has to be increased to 10mA the delay is 200ps.

Therefore opto links have a reduced transmission speed if the laser diode current has to be increased. This could be necessary if the optical losses are too large or inefficient diodes are used.

Some modern epitaxy technology integrates the laser on the same chip as the electronics. The inductance is then reduced considerably.

8.2 Physical wires

The on-chip physical wires are lossy transmission lines. For high-speed circuits the transmission line effects are essential, especially for clock wires. The main obstacle is the RC-effect of the wire resistance and the wire capacitance. It both delays the signal and changes the wave form.

The effect could not be changed by using wider wires or tapering the wires. The technology may use Cu instead of Al as wire material. The difference is within a factor of 2. Other dielectrics may be used and the insulator thickness may be increased, however the effect is within a factor of 2. Therefore technology is not the solution.

The wire may be altered to a wire with a number of amplifiers placed along the wire [4][7][10]. An amplifier may be an inverter or a receiver/transmitter pair. The change of the wire introduces delay and increases the skew between wires.

As shown above the units (memory and arithmetic) are generally small. Within a unit there is generally no problem with the wire length, even when scaled.

A multiprocessor may see a wire as a transmission line with several bits. The scheduling mechanism may allocate several transmissions simultaneously on such a wire. The delay could be considerable without effecting the overall performance. Therefore a long wire could be replaced by a number of shorter wires separated by registers. The distance could be chosen such that the RC-effect of the wire has no significance.

8.3 Discussion

In Figure 19 and Figure 20 the performance of the four treated transmission media are shown. In these figures the energy dissipation and the scheduling interval are plotted against the physical length of the link. As expected the full swing variant is only superior below 250 (approximately 30 μ m). There is a difference within a factor of 2 for the others. There are design parameters such that this difference could be compensated for. By sure the low voltage pair could be halved.

scheduling interval

The figures for the scheduling interval shows that full swing is superior as expected. The others are equal within a factor 1.4. In this case it is not likely that there could be a compensation.

Figure 19 The energy dissipation for the four transmission media. The values at zero length are for full swing = 19, low voltage = 33, transmission line = 27, and opto = 53.

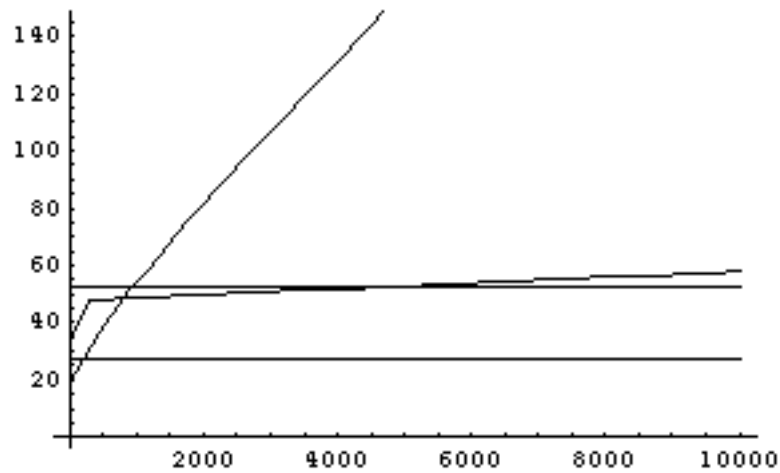
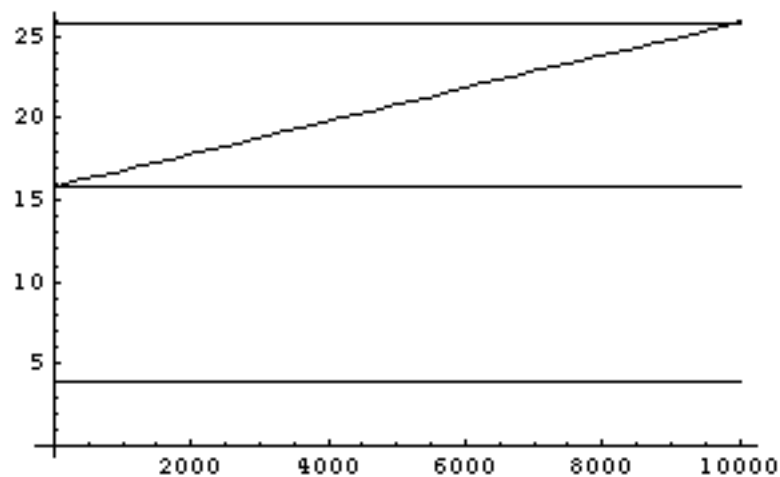


Figure 20 The scheduling interval for the four transmission media. The values at zero length are for full swing = 4, low voltage = 16, transmission line = 16, and opto = 26.



The scheduling interval for all of them should be compared to the scheduling intervals of arithmetic and memory. They are shorter, thus the network could fulfil the necessary transports.

energy dissipation

The energy dissipation is less than in the arithmetic unit and the memories. Thus single link transports do not significantly increase the energy dissipation. However in a multi stage link the power dissipation of the links may be worse.

It is also fine that on-chip as well as off chip links are equal good. Thus the chip-edge is no limiting border. However, an arithmetic unit needs approximately 100 communication links in order to pace its speed. Contemporary technology allows for several hundreds of these wires. Therefore the communication may be hindered when more than 10 arithmetic units are present on a chip!

The topology for a network could from transport point of view be analysed as a number of communication switches separated by communication links. The communication switches have almost negligible energy dissipation and scheduling intervals. The throughput of these links are independent on how many links being cascaded. However, the energy dissipation is proportional.

From execution point of view each node of a DAG could be analysed. The value of a node to transport is a result of an arithmetic operation consuming an implementation independent amount of energy. The result is written and then read *fan-out* (typical 2) times from the memory. Thus there are some few $1+fanout$ transports. Assuming that there are n_{hop}

communication links passed for each transport, then the energy consumed is

$$E_{net} = n_{hop} \cdot (1 + n_{fanout}) \cdot E_{link} \quad 28$$

This energy could be considerable. An allocation method may place node near to be local. The communication network may be rich, however the number of hops have to be limited. In order to understand how important this is, assume that there are two operators L and S using 10000 and 2000 gates, respectively, for 32 bit arithmetic. They correspond to the size of a floating point arithmetic and fix point arithmetic units. Each bit of them consumes approximately 600 and 120 energy equivalents. Assuming that a link dissipates 40 energy equivalents, the energy for the arithmetic corresponds to 10 and 1 hops, respectively.

An optimal communication network connecting n_{term} points may have ${}^2\log(n_{term})$ hops. For foreseeable sizes this figure is less than 30. Thus the worst case communication consumes about 6 and 30 times, respectively, more than the arithmetic.

9 Clock generation

There are some asynchronous self-timed logic. It is not well understood nor well-used. For other types of systems there is a need of a clock. It should as an abstract feature be understood as a physical action indicating a particular time.

The clock system consists of clock generators and clock distribution devices. The clock generator is some type of oscillator, free-running or controlled. It produces some type of cyclic signal, generally a square wave. The clock distribution communicates this signal to (all) registers.

Clocks are always needed if not self-timed circuits are used. Self timed circuit either use edges as indicator of state and time or combinations of logical levels and hand-shaking signals. It is not likely that units using a number of peripheral signal should be able to handle such timing safely [24]. Probably it is of that reason that this techniques is not well used.

Traditional communication links have used hand-shaking. On long wires this is both slow and less reliable. Generally older communication standards used hand-shaking. Most, but not all, modern communication standards are synchronous in some respect. As an example on the contrary is the popular PCI-bus.

9.1 Where to use clock generation

Clock generation is a must of some type of processor or parts of processors. The units need clock signals for their registers. This is the conventional way to use clock generators. However, it is a result of a long lasting evaluation and the question is whether there could be other types:

- a clock generator could be unique for each register bit. Because of the size of the clock generator compared to the register bit this is a costly implementation. There are few places where this type of application could be used.
- a clock generator could be unique for a register. A common application is the receiving or transmitting register in a communication link.

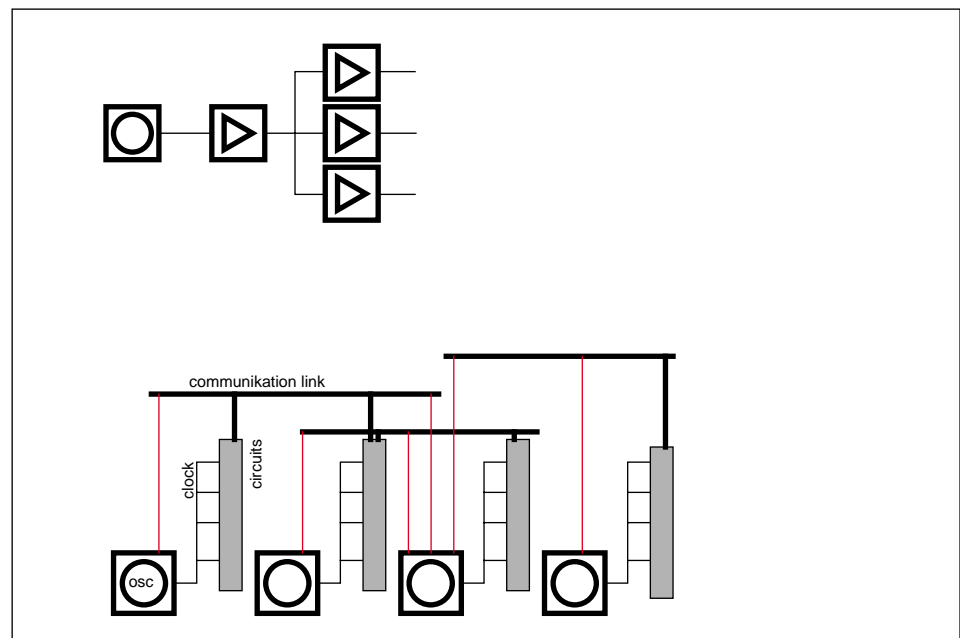
- a clock generator could be used for a group of registers. They may be part of an arithmetic unit, a part of a processor or a full processor.

In all these cases clock frequency and phase has to be controlled. The technique to phase lock an oscillator to a signal is well developed. This technique could be used for the first application. The latter two applications may let separate clock regions to be asynchronous or being phase locked to each other.

The technique using asynchronous clock regions requires communication between them to be self-timed. One problem with such logic is that it is not reliable because of meta-stable state. If the clocks in two communicating units are almost in phase or have only a slight frequency difference it is likely that such meta-stable state would occur. As a multiprocessor may use large amounts of separate clock regions the possibility to enter such states is even more likely.

Another type of clock scheme is to have all clock generators phase locked to each other.

Figure 21 Clock systems. a: is a central system and b: a register per unit. Clocks are distributed between the clock generators in order to synchronize



9.2 Phase locking many clock generators

Assume that there is a set of oscillators. They have all some state specifying a nominal clock frequency and means to control an oscillator to run at this frequency within some tolerance.

The clock signals should be controlled in two ways

- their frequency should be within a frequency range near a nominal frequency f_0 . In some systems this frequency needs to be very accurate, in other less or just adequate to control the registers.
- the phase difference between all oscillators should be zero.

The frequency could be adjusted by giving priorities to the clock or/and each oscillator has a priority for each value of the state space of the nominal frequency.

The phase difference can only be measured if clock edges from other clocks are compared to a clock's own frequency. A network of such clock

transmissions is necessary. However, the same network as the communication network may be used.

All clocks are therefore connected in a bidirectional graph. The entire control system is a complex regulator system. One such system has been demonstrated¹.

9.3 Clock distribution

The clock distribution consists of wires and drivers. An arithmetic unit may contain some thousand registers. Therefore the fan-out is very high resulting in a tree of drivers. Optimum amplification is around 3. Therefore 5-6 levels are needed.

The delay of such a driver is 6. Thus the total delay is 36. This value should be compared to the delay of one pipe-line stage in the arithmetic unit being in the range of 50 (see paragraph 6.3 on page 19).

The different branches of the tree have to be matched. Mismatch may be compensated for by increasing the scheduling interval. A match of the drivers within 10% is likely.

A normal wire has a resistance in the range of 1kohm/mm, depending on the technology. The time constant of the wire is in the range of 10ps/mm². The size of an arithmetic unit is expected to be small (see paragraph 6.4 on page 21). Therefore the delay in the wires should be negligible.

The main problem in the distribution network is to keep fast rise and fall times. Introduction of negative resistance as schmidt triggers may help.

Clock distribution is probably one of the most important parts to be researched. However, going from non pipe-lined arithmetic to pipe-lined increases the amount of registers with a factor 10-30. It corresponds to only 2-3 more levels in the cascade of clock drivers. It is therefore likely that the problem could be solved.

9.4 Discussion

Clock distribution has been considered a main problem. Contemporary research focuses on synchronisation very large chips. The first attempt has been to match the distribution of the clock [26]. A next step is to include programmable delay lines [29][36].

One of the misconceptions is that chips are assumed to be large centralised devices. By building up both communication and clock systems as a number of separate units, each small unit may use classical techniques. Between units artificial transmission lines being a shift register can be used in order to eliminate the RC-delay problem. Clocks within each small unit are to be synchronised with neighbour clocks.

By using the edges of the transmitted data as synchronisers perfect clock phase matches may be created. There is a clock generator component having inputs from some few communication links producing a clock signal. This type of device is frequently used on a chip. Such a component does not exist today.

In such a system the phase difference between distant units could be large but low between neighbour units.

¹ In the rp8601 system these types of clock generators was researched. The system was simulated but not built. Nor research report was published.

10 Acknowledgements

The documentation of this research has partly been supported by the IDE section of Halmstad University, Sweden. The main result was researched during the earlier phases of rp8601-project that was financed by Incentive AB (now Gambro).

11 References

- 1 B. J. Benschneider, W. J. Bowhill, E. M. Cooper, M. N. Gavrielov, P. E. Gronowski, V. K. Maheshwari, V. Peng, J. D. Pickholtz, and S. Samudrala, "A pipelined 50-MHz CMOS 64-bit floating-point arithmetic processor," *IEEE Journal of Solid-State Circuits*, vol. 24, pp. 1317 - 1323, October 1989.
- 2 Cardarilli G. C., Salmeri M., Salsano A., and Simonelli O.: *Bus Architecture for Low-Power VLSI Digital Circuits*, *IEEE Proc. ISCAS-96*, Vol. 4, pp. 21-24, 1996, Atlanta, USA.
- 3 Carlstedt Gunnar, "Resource theory for multiprocessors", Carlstedt Elektronik AB, 1996, ISBN 91-89546-07-5.
- 4 J.-D. Cho and M. Sarrafzadeh, "A buffer distribution algorithm for high-performance clock net optimization," *IEEE Trans. VLSI Syst.*, vol. 3, pp. 84-97, Mar. 1995.
- 5 J. S. Choi and K. Lee, "Design of CMOS tapered buffer for minimum power-delay product," *IEEE J. Solid-State Circuits*, vol. 29, pp. 1142-1145, Sept. 1994.
- 6 J. Clouser, M. Matson, R. Badeau, R. Dupcak, S. Samudrala, R. Allmon, and N. Fairbanks, "A 600-MHz superscalar floating-point processor," *IEEE Journal of Solid-State Circuits*, vol. 34, pp. 1026 - 1029, July 1999.
- 7 J. Cong and C.-K. Kok, "Simultaneous driver and wire sizing for performance and power optimization," *IEEE Trans. VLSI Syst.*, vol. 2, pp. 408-425, Dec. 1994.
- 8 F. Dartu, N. Menezes, and L. T. Pileggi, "Performance computation for precharacterized CMOS gates with RC loads," *IEEE Trans. Computer-Aided Design*, vol. 15, pp. 544-553, May 1996.
- 9 G. De Micheli, "Performance-oriented synthesis of large-scale domino CMOS circuits," *IEEE Trans. Computer-Aided Des.*, vol. CAD-6, pp. 751-765, 1987.
- 10 S. Dhar and M. A. Franklin, "Optimum buffer circuits for driving long uniform lines," *IEEE J. Solid-State Circuits*, vol. 26, no. 1, pp. 32-40, Jan. 1991.
- 11 N. Goncatves and H. J. De Man, "NORA: A racefree dynamic CMOS technique for pipelined logic structures," *IEEE J. Solid-State Circuits*, vol. SC-18, pp. 261-266, 1983.
- 12 N. Hedenstierna and K. O. Jeppson, "CMOS circuit speed and buffer optimization," *IEEE Trans. Computer-Aided Design*, vol. 6, no. 2, pp 270-281, Mar. 1987.
- 13 N. Ide, H. Fukuhisa, Y. Kondo, T. Yoshida, M. Nagamatsu, J. Mori, I. Yamazaki, and K. Ueno, "A 320-MFLOPS CMOS floating-point

- processing unit for superscalar processors," *IEEE Journal of Solid-State Circuits*, vol. 28, pp. 352 - 361, March 1993.
- 14 N. Ide, M. Hirano, Y. Endo, S. Yoshioka, H. Murakami, A. Kunimatsu, T. Sato, T. Kamei, T. Okada, and M. Suzuoki, "2.44-GFLOPS 300-MHz floating-point vector-processing unit for high-performance 3-D graphics computing," *IEEE Journal of Solid-State Circuits*, vol. 35, pp. 1025 - 1033, July 2000
 - 15 Y. Jiren, I. Karlsson, and C. Svensson, "A true single phase clock dynamic CMOS circuit technique," *IEEE J. Solid-State Circuits*, vol. SC-22, pp. 899-901, 1987.
 - 16 A. B. Kahng and S. Muddu, "New analyzes of distributed RC interconnections," in *Int. Symp. Circuits Systems*, vol. 4, pp. 241-244, 1996.
 - 17 A. B. Kahng, K. Masuko, and S. Muddu, "Analytical delay models for VLSI interconnections under ramp input," in *IEEE/ACM ICCAD 1996*, 1996, pp. 30-36.
 - 18 S. Komori, H. Takata, T. Tamura, F. Asai, T. Ohno, O. Tomisawa, T. Yamasaki, K. Shima, H. Nishikawa, and H. Terada, "A 40-MFLOPS 32-bit floating-point processor with elastic pipeline scheme," *IEEE Journal of Solid-State Circuits*, vol. 24, pp. 1341 - 1347, October 1989.
 - 19 E. T. Lewis, "Optimization of device area and overall delay for CMOS VLSI designs," *Proc. IEEE*, vol. 72, pp. 670-689, 1984.
 - 20 S. J. Lovett, M. Welten, A. Mathewson, and B. Mason, "Optimizing MOS transistor mismatch," *IEEE Journal of Solid-State Circuits*, vol. 33, pp. 147 - 150, January 1998.
 - 21 S. J. Lovett, G. A. Gibbs, and A. Pancholy, "Yield and matching implications for static RAM memory array sense-amplifier design," *IEEE Journal of Solid-State Circuits*, vol. 35, pp. 1200 - 1204, August 2000.
 - 22 D. A. B. Miller and H. M. Ozaktas, "Limit to the bit-rate capacity of electrical interconnects from the aspect ratio of the system architecture," *Journal of Parallel & Distributed Computing*, vol. 41, pp. 42-52, 1997.
 - 23 J. Mori, M. Nagamatsu, M. Hirano, S. Tanaka, M. Noda, Y. Toyoshima, K. Hashimoto, H. Hayashida, and K. Maeguchi, "A 10 ns 54*54-b parallel structured full array multiplier with 0.5- μ m CMOS technology," *IEEE Journal of Solid-State Circuits*, vol. 26, pp. 600 - 606, April 1991.
 - 24 F. Mu and C. Svensson, "A 750Mb/s 0.6 μ m CMOS two-phase input port using self-tested self-synchronization," *IEEE International Solid-State Circuits Conference*, vol. XLII, pp. 178 - 179, February 1999.
 - 25 F. Mu, A. Edman, and C. Svensson, "Digital multiphase clock/pattern generator," *IEEE Journal of Solid-State Circuits*, vol. 34, pp. 182 - 191, February 1999.

-
- 26 M. Nekili, G. Bois, and Y. Savaria, "Pipelined H-trees for high-speed clocking of large integrated systems in presence of process variations," *IEEE Trans. VLSI Syst.*, vol. 5, pp. 161–174, June 1997.
 - 27 B. Nikolic, V. G. Oklobdzija, V. Stojanovic, W. Jia, J. K. Chiu, and M. M. Leung, "Improved sense-amplifier-based flip-flop: Design and measurements," *IEEE Journal of Solid-State Circuits*, vol. 35, pp. 876 - 884, June 2000.
 - 28 M. J. M. Pelgrom, A. C. J. Duinmaijer, and A. P. G. Welbers, "Matching properties of MOS transistors," *IEEE Journal of Solid-State Circuits*, vol. 24, pp. 1433 - 1439, October 1989.
 - 29 P. Ramanathan, "Clock distribution in general VLSI circuits," *IEEE Trans. VLSI Circuits Syst.*, vol. 41, pp. 395–404, May, 1994.
 - 30 M. Shoji. "FET scaling in domino CMOS gates." *IEEE Journal of Solid-State Circuits*, vol. SC-20, pp. 1067-1071, 1985.
 - 31 K. Takeda, F. Ishino, Y. Ito, R. Kasai, and T. Nakashima, "A single-chip 80-bit floating point processor," *IEEE Journal of Solid-State Circuits*, vol. 20, pp. 986 - 992, October 1985.
 - 32 N. Takla and M. Hecker, "A monolithic 64 bit floating-point coprocessor," *IEEE Journal of Solid-State Circuits*, vol. 19, pp. 538 - 539, August 1984.
 - 33 M. Uya, K. Kaneko, and J. Yasui, "A CMOS floating point multiplier," *IEEE Journal of Solid-State Circuits*, vol. 19, pp. 697 - 702, October 1984.
 - 34 G. Wolrich, E. McLellan, L. Harada, J. Montanaro, and R. A. J. Yodlowski, "A high performance floating point coprocessor," *IEEE Journal of Solid-State Circuits*, vol. 19, pp. 690 - 696, October 1984.
 - 35 J. Yuan and C. Svensson, "High-speed CMOS circuit technique," *IEEE Journal of Solid-State Circuits*, vol. 24, pp. 62 - 70, February 1989.
 - 36 Q. Zhu and W. W. M. Dai, "High-speed clock network sizing optimization based on distributed RC and lossy RLC interconnect models," *IEEE Trans. Computer-Aided Design*, vol. 15, no. 9, pp. 1106–1118, Sept. 1996.

Symbols

α	current gain in an optical system consisting of transmitter, medium and receiver	n_{acc}	number of memory accesses per node of a DAG
ϵ_{ox}	dielectric constant of gate oxide	N_{alu}	number of arithmetic units
η	relative switching frequency of a node. When switching every cycle the value is 0.5	n_{cl}	number of gates controlled by the clock signal in one stage of a pipe-line in an arithmetic unit
η_{hit}	hit ratio in a cache memory	n_{fanin}	fan-in of a gate, i.e. the number of inputs
τ	intrinsic circuit delay of a technology, i.e. $c \cdot r$	n_{fanout}	fan-out of a gate, i.e. the number of inputs connected to the output
A_{alu}	total chip area for all arithmetic units	n_{hier}	number of memory levels in a memory hierarchy
A_{op}	chip area for an arithmetic unit implementing operator op	n_{hop}	number of communication links along a path implementing a vertex of the DAG
$A_{op,pipe}$	chip area of the pipe-line in an arithmetic unit	N_{op}	number of arithmetic units implementing the operator op
A_{ref}	a reference area of an intuitive implementation of the arithmetic of an application	n_{op}	number of instances of operator op in the DAG
A_{reg}	chip area of a register bit	n_{stage}	number of stages in a pipe-line of an arithmetic unit
c_{node}	capacitance of a node	P	power
c_{ox}	gate capacitance per unit area	P_{ref}	a reference power dissipation of an intuitive implementation of the arithmetic of an application
c_w	wire capacitance per unit length	r	transistor resistance
d_{op}	gate depth of an arithmetic unit implementing operator op, i.e. along longest path from input to output	t_{bit}	scheduling rate for a transport
d_{stage}	gate depth of a stage in a pipelined arithmetic unit	T_{cyc}	period for the real-time cycle of the application
E	energy	t_{op}	scheduling interval for arithmetic units
e	the elementary energy stored in one gate capacitance, $c \cdot V_{dd}^2$	t_{ox}	thickness of gate oxide
E_{link}	energy dissipation for one transport over a communication link	t_{pad}	time delay for a pad
E_{net}	energy dissipation for the transport along a path for a vertex of the DAG	V_{dd}	power supply voltage
E_{reg}	energy dissipation of a register	V_{in}	
E_{sense}	energy dissipation of a sense amplifier	v_{th}	saturation velocity for electrons under gate oxide
E_{stage}	switching energy of one stage of a pipe-line in an arithmetic unit	V_{pin}	voltage on a pad
f_{op}	execution frequency for an operator op of an application	V_{sense}	voltage threshold for a sense amplifier
g_{op}	the number of gates in an arithmetic unit implementing the operator op	w_{stage}	bit width of a stage of a pipe-line in an arithmetic unit
I_d	drain current	Z_o	characteristic impedance of a wire
I_{pin}	current through a pad	w_{mem}	block length in the cache memory hierarchy
l	wire length	w_{tr}	width of transistor
l_{tr}	channel length of a transistor		
L_{pin}	inductance of a wire from a pad		

12 The author



Gunnar Carlstedt has 30 years of experience in computer architecture. Most of his time has been devoted to research and development in this field, on consultancy bases, mainly to the Swedish industry, but also to other European and US based companies. He has designed 23 different computer architectures, of which about every second have been produced. Among them, computers have been designed for Ericsson APZ telephone switching systems, the Swedish Defence air crafts JA37 Viggen and JAS 39 Gripen.

Among his interests were hardware-near languages, including various general-purpose languages and graphical languages of imperative and applicative type. His main concern has been the semantics and its implementation.

His interests also include the processor micro-architecture, its formal definition, way of mapping language interpreters onto this structure and the performance estimating of such interpreters. He probably designed the first RISC processor the year 1974.

His interests also include the hardware components of processors as memories, buses and arithmetic units of various kind. He has a special interest in associative memories.

His interests also include the methods for synthesis of computer architecture and processors, including hardware description languages, automatic synthesis on high and low level, and special silicon compilers.

He was very early (1975) interested in distributed parallel computer systems, with an emphasis on languages, problem allocation, protocols, memory systems, and computer topology and this interest remains.

Gunnar Carlstedt has also been researching the Swedish JAS 39 Gripen project on the behalf of the board to find the performance of the programs, computers, tests, tools and project.

Gunnar Carlstedt is a direct ancestor to

- king Karl VIII Knutsson in Sweden
king Erik den helige,
king Inge den äldre,
king Stenkil
- king Valdemar II Seir,
king Valdemar den store,
king Erik Plovpenning,
king Erik Ejegod,
king Knut IV Svensson,
king Svend Estridsen,
king Svend Tveskægg,
king Harald Blåtand and
king Gorm den gamle of Denmark.

Sven Estridsson one of the earliest and believed to be one of the most important kings in the Nordic history

- king Håkan V Magnusson of Norway,
king Magnus 6 Lagabøte,
king Håkon IV Håkonsson,
king Håkon III Sverresson,

king Sverre Sigurdsson,
king Olav Tryggvarsson,
king Harald Hårfager,
fylkesking Halvdan Svarte

- king Sigurd II Munn from the Fare Isles

He also has relatives near many middle age kings. Especially Ulf Jonsson the assistant of Engelbrekt, but also Måns Bengtsson who assassinated Engelbrekt on the lake Mälaren.

Some people have sacrificed the lives for their beliefs

- One of king Gustav Vasas nearest men, Måns Bryntesson (Lilliehöök af Fårdala, nr 1). He got to much power and was assassinated.
- Erik Abrahamsson, was decapitated during Stockholm's blood bath.
- Nils Bosson was killed by his peasants.
- Erik Eriksson the younger was killed by the Västgöta peasantry.
- Brynte Bertilsson died at in the battle at Örskelljunga.
- king Erik Ploppenning was captured and assassinated in Slien by the men of duke Abel of Sønderjylland.
- king Knut IV Svensson was killed
- Knut den store killed Ulf Jarl.

Bo Jonsson Grip is one of the richest persons of Sweden ever. The holy Saint Birgitta was near the relatives.