AT VLSI High Level Synthesis Laws: Theory and Practice

AT VLSI High Level Synthesis Laws: Theory and Practice

Miodrag Potkonjak
Computer Science Department
University of California
Los, Angeles, Ca 90024

Jan Rabaey
University of California at Berkeley
Dept. of EECS
Berkeley, CA 94720

Abstract

As high level synthesis matures in a viable alternative for design of industrial ASIC and rapid scanning of the design solution space, the importance of development of firm and sound theoretical foundations is well recognized. Simultaneously a need for rapid and accurate prediction methodology and tools is becoming widely acknowledged. We attack those two problem by developing AT (area-time) high level synthesis laws.

We first establish three AT high level synthesis laws that relate different components of the area of ASIC implementation cost, namely foreground memory, execution units and interconnect, to the sampling period (available time). The laws state that: $A = const$, $AT = const$, and $AT^2 = const$ for the area of registers, execution units, and interconnect respectively. We validate the correctness of the AT laws using case studies and statistical analysis of synthesis results of 80 real life designs.

Several important applications of the *AT* laws for development of high level synthesis tools are presented. For example, algorithms for area optimization using module selection, power optimization using clock selection, and hierarchical allocation and scheduling are introduced. Use of the AT high level synthesis laws as an effective method for encapsulation of high level synthesis knowledge is also studied. In this framework, the AT laws are used to guide design process and to reason about properties of high level synthesis approaches and algorithms. The effectiveness of the AT laws applications is documented on numerous real life designs.

1.0 Introduction

1.1 Motivation: Design Space Exploration

We study the following fundamental design space exploration problem in high level synthesis of numerically intensive applications. An arbitrary ASIC computation (computation which is executed on semi-infinite stream of data on dedicated hardware) is given. When this computation is implemented under a timing constraint that the available time (sampling period, initiation time) is *T1*, the area of implementation is *A1*. The question which we are interested in is, if the available time is changed to *T2*, what we can say about the area of implementation under the new timing constraint. We assume that both *T1* and *T2* are, of course, longer than the critical path of the computation. Furthermore, we are interested not only in the relationship between the total areas of implementations under different sampling rates, but also in the ratios of area requirements for specific components of design: memory elements in data paths, functional elements and interconnect.

The just outlined problem is a backbone of crucially important high level synthesis and system level synthesis design exploration task. Design space exploration has been recognized from the early beginnings of high level synthesis as one of the key problems whose solution will provide a designer extremely valuable information [Bar73]. Since then, it constantly has been receiving a great deal of attention from a number of high level researchers [McF90a, Wal91, Rab91] but only recently, with the advent of hardware-software codesign [DeM93] it was promoted to the position of the key high level synthesis and system level synthesis task.

We will study the design space exploration problem from two distinct, but deeply interrelated points of view. The first one is macroscopic theoretical model and associated methodology; the second one is extensive well organized experimentation with large number of real life practically sized examples. Both points of view (sound theory and practical usefulness) are widely recognized as mandatory prerequisite for advancement of the state of the art in high level synthesis [McF90a, DeM92, Gaj92].

1.2 Motivational Example: Efficiency vs. Throughput

Widespread and well established common belief, not just in high level synthesis but also in many other areas which address parallelism exploration, is that efficiency and throughput are inversely correlated requirements. However, the following small and simple experimental study and literature survey indicates a surprising evidence: that resource utilization and throughput are strongly positively correlated in high level synthesis ASIC designs, and that the AT product is continuously improved as timing constraints are becoming stricter.

AVAILABLE TIME	50	40	30	25	20
EXU (mm^2)	0.52	0.77	0.89	1.09	1.40
EXU AT	26.00	30.8	26.7	27.25	28.00
Registers (mm^2)	2.68	2.68	2.68	2.68	2.68

Table 1: The size parameters of the 8th order IIR Avenhaus cascade filter at the RT level

Interconnects	18	24	36	44	60
INT $AT^{1.35}$	3539	3491	3552	2294	3424
Active Area	3.71	4.17	5.24	5.70	6.17
Active Area AT	26.23	26.37	27.46	26.35	27.59

Table 1: The size parameters of the 8th order IIR Avenhaus cascade filter at the RT level

Table 1 shows the relationship between the AT product and available time of the 8th order Avenhaus cascade filter [Ave72] obtained using the HYPER [Rab91] high level synthesis system. The data is measured by the size of the datapath at the RT level. AT product is the product of area of implementation and available time. Even without elaborate statistical study, the following conclusion is unavoidable: **As the sampling period (available time *T*) decreases the AT product significantly improves**.

One can argue that "counter-intuitive" experimental results are actually consequence of peculiarities of used high level synthesis tools, which for unspecified reasons perform significantly better under strict constraints. In order to dismiss that objection, Table 2 presents the relationship between min-bounds and available time. The min-bounds are again provided by the HYPER system, and they indicate lower bounds on the amount of the required resources [Rab91]. Again, the experimental data strongly supports the previous conclusion.

TIME	50	40	30	25	20
EXU (mm^2)	0.52	0.72	0.89	1.09	1.40
EXU AT	26.00	28.8	26.7	27.25	28.00
REG	27	27	27	27	27

Table 2: The lower bounds on the size parameters of the 8th order IIR Avenhaus cascade filter at the RT level

Yet another objection can be raised, that the indicated AT relationship is valid only at the RT level, and that at the layout level the trade-off will be "intuitive". However, comparing the three layouts in Figure 1 from Table 3 is again strong confirmation of the claimed relationship. The layouts were automatically (without any human intervention) produced using HYPER and LAGER tools.

So, the only other remaining pending final objections can be that actually some peculiarities of the used example, 8th order IIR Avenhaus filter or the fact that high effort manual optimization is not applied, dictate relationship between efficiency and throughput. However, a short survey of the high level synthesis literature clearly indicates that the claimed AT trend is not accidental. For example, Rabaey et al. [Rab91]

TIME	50	40	25
AREA	8.19	8.99	12.65
AT product	57.91	56.86	63.25

Table 3: The size parameters of the 8th order IIR Avenhaus filter at the physical layout level

(a) (b) (c)

Figure 1: Layouts of the 8th order Avenhaus filter for three different available times: (a) 25 control cycles; (b) 40 control cycles; (c) 50 control cycles

present the implementation of the 7th order IIR filter whose layouts are heavily manually optimized. The comparison of the area under three different sampling rates (see Table 4), clearly indicates that the AT product does not increase when the available time is reduced. Several other important properties of this type of the AT relationship, as can be seen from Tables 1, 2 and 3, state that the relationship between the area of execution units, registers and interconnect and the sampling rate is respectively following laws $AT = const$; $A = const$ and $AT^2 = const$. The evidence for those relationships is indeed strong. For example there is no difference at all among the register numbers as indicated by either full implementation or min-bounds. The difference for execution units is less than 19% percent, and even the number of interconnect

TIME	20	16	10
AREA	13	18.9	27.95
AT product	260	302.4	279.5

Table 4: Physical characteristics of the 7th order IIR filter [Rab91]

is following the claimed trade-off line within less than 5%. Even the area of the layouts indicates just slightly more than 10% discrepancy. The analysis of the high level literature again confirms the claimed relationship. For example, the results presented by Parker at al. [Par91] fully support thesis that the number of registers in the implementation does not change with the available time.

In the rest of the paper we will demonstrate that the area-time laws, detected in the introductory example, are not just fascinating high level synthesis paradox, but also surprisingly powerful starting point for the development of tools and methodologies in not just design space exploration and many other high level synthesis tools, but also in general design methodology.

1.3 Previous work

We will survey two different classes of research which are most relevant to AT high level synthesis laws: (i) the most influential laws in computer science and engineering research and (ii) the most notable laws and AT models in CAD research, and in particular in high level synthesis.

Probably the two best known computer science and engineering laws, next to the famous Brooks law [Bro75] stating that for the last 10% percent of any software project one has to spend 90% of the overall effort, are Grosch's law and Amdahl's law. Grosh law is expressed by the formula

$$p = k \times c^2$$

where p is computing power of general purpose computer, k is a constant, and c is system cost, or more precisely purchase price. Note that, in some sense, Grosh's law is, at least superficially, similar in spirit to what we observed in the 8th order Avenhaus filter trade-offs, where computing power corresponds to throughput, and price to the area of implementation. Grosh established his law [Gro53] by considering cost-performance ratios on unprecedented variety of computers, which included hand calculations, log tables, desk calculators, and a couple of generations of electronic (relay and vacuum tubes) computers.

Amdahl's law states that the performance improvement of a general purpose computer is limited by the fraction of the time where faster mode which employs parallelism exploitation can be used [Amd64]. It was used for the major strategic decision during design of the popular IBM 360 and 370 series of computers [Cas78] and is discussed in a number of computer architecture books in detail.

Due to the relative maturity of physical design in comparison with other CAD fields, it is not surprising that the largest number of fundamental relationships was established in that area. The most well known law in physical design is the Rent's rule [Han88]. The law relates the number of IO pins to the average size (AS) of the blocks and the chip and their number in the equation

$$IO = AS \times BK^r \tag{EQ 1}$$

r was denoted as the Rent exponent and takes several different values in accordance with the design style.

Later on, several more elaborate attempts were undertaken to investigate experimentally observed data about distribution of wire lengths. Sastry and Parker [Sas86] statistically proved that the empirical data of wire length distribution fits well the Weibull model significantly better than Pareto-Levy's distribution proposed earlier by Donath [Don81].

The design space exploration was considered by a number of research groups in high level synthesis [Wal91, Gaj92]. McFarland and Kowalski used both AT and AT^2 measures to forcefully stress the importance of including the explicit cost of datapath memory elements and interconnect [McF90b]. Jain [Jai89] established the lower-bound area-delay trade-off law which says that the AT product is constant. However, his assumptions about designs were extremely strong: 100% utilization rate of all resources is assumed and only the area of execution units was taken into account. We will show that for the area of execution units this law is valid for a large class of designs, regardless of the resource utilization rate. Potkonjak and Rabaey recently showed a surprising fact that by using transformations the throughput of any linear computation can be improved to the arbitrarily high level while maintaining the AT product [Pot92].

Finally, we would like to mention VLSI theory and its AT laws. The VLSI theory got the initial impetus in the Ph.D. work by C.D. Thompson at the Carnegie-Mellon University [Tho79]. Soon, it was rapidly developed within theoretical science community [Ull84]. Recently, it has been recognized as an important methodology for parallelism exploration [Sei90]. For excellent, in detail treatments of the VLSI AT laws we refer to [Ull84]. The VLSI AT laws establish bounds on what can be achieved in a given silicon area assuming 100% utilization of all available hardware resources.

2.0 High Level Synthesis AT Models and their Formal Analysis

2.1 AT laws and Assumptions under which they are exact

This section is devoted to the development of three fundamental high level synthesis laws. They are:

1. **Register AT law:** The area of registers in a datapath is not dependent on the throughput of the design. So $A = const$ for registers.
2. **Execution Unit AT Law:** The product of the area of execution units and throughput is constant, regardless of a targeted throughput. So $AT = const$ for execution units.
3. **Interconnection AT Law:** The product of the square of the number of interconnects in design and throughput is constant for any throughput. So, $AT^2 = const$ for the number of interconnect.

The underlying assumptions behind the laws are the following. All designs have in its behavioral description a very large number of operations. Furthermore for each type of operations present in behavioral description there are many instance.

The available time is significantly larger than the critical path of computation, but at the same time it is short enough that for all types of operations a large number of execution units is required. There is no any structure in behavioral description of computation and in particular there is no restrictions and regularity in the interconnect pattern of communications (i.e. essentially all communication is global).

Although the presented restriction appears very strict and limiting, the experimental studies show that all, with an exception of the last one (that all communication among operations is such that implies the need for global interconnect), can be drastically relaxed to allow the application on surprisingly small designs.

2.2 Derivation of High Level Synthesis AT Laws

2.2.1 Control Data Flow Graph CDFG Gas Model

For derivation of the high level synthesis laws we will use control data flow graph (CDFG) Gas model. We will first analyze forces which act on the operations of the computation under timing and hardware allocation constraints. Scheduling, assignment (binding) and allocation [Wal91] in high level synthesis is often modeled as a placement problem of computation operation in two dimensions, those of space and time [e.g. Dev89]. Each operation has to be scheduled to a particular time position and assigned to a particular time position. Another commonly used notion in high level synthesis is one of forces, which indicate an expected likelihood that a particular operation is bound to be scheduled at particular time/

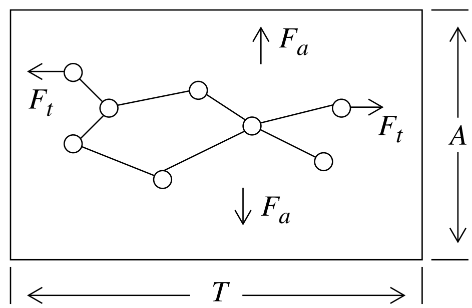


Figure 2: CDFG Gas Model

space position. We will combine and generalize those two representations to form the CDFG gas model. Figure 2 shows the forces which act between the structure of CDFG and timing and resource constraints or goals. Note that F_t and F_a forces differ from the forces which have been recognized in a number of high level synthesis approaches, most notably in force directed scheduling [Pau89]. Force directed scheduling, as it will be explained in Section 5, takes into account only forces with respect to available time, because it assigns the probability that a particular node is bound to be scheduled in a particular time step by analyzing only *ASAP* and *ALAP* times.

The F_a and F_t forces act between the CDFG and available time and allocated resources and their intensity changes as temporal and hardware constraints are changing. Of course, there are also forces which CDFG nodes exercise on each other through data and control edge dependences. Note that even nodes which are not in transitive fan-out or transitive fan-in of each other exercise forces on each other indirectly through either "walls" (timing and hardware constraints) or other nodes.

Since we assumed that the CDFG is large, there is an apparent analogy with Gas Models. CDFG nodes correspond to gas molecules and hardware and timing constraints to a container in which gas is trapped. The volume of the container corresponds to the product of area of execution units (each CDFG node (molecule of gas) has to be assigned to some execution unit) and time step. The structure of the graph dictates the "density" of the CDFG gas. Using this analogy we can directly transfer gas laws to the high level synthesis domain. We will do that without explaining elaborate and involved derivation of gas laws which can be found in the excellent physics textbooks [Fey65] or gas models studies [Hir65].

We now ask ourselves the following question: What will happen with the area of execution units when we increase the available time T by some amount ΔT? The corresponding analysis in the gas model domain indicates that any segment of $\Delta\tau$ will expand by $\Delta T \times (\tau / T)$ under reduced pressure from remaining gas molecules and the walls of the container. This will reduce the pressure on A correspondingly so that the AT product remains the same (See Figure 3).

2.2.2 High Level synthesis AT Laws

We are now ready to analyze the consequences on specific components of the area cost due to the alternation in pressure on the internal structure of the considered CDFG after the relaxation of the available time constraints.

The execution unit requirements alternations under the changed timing constraints are the easiest to ana-

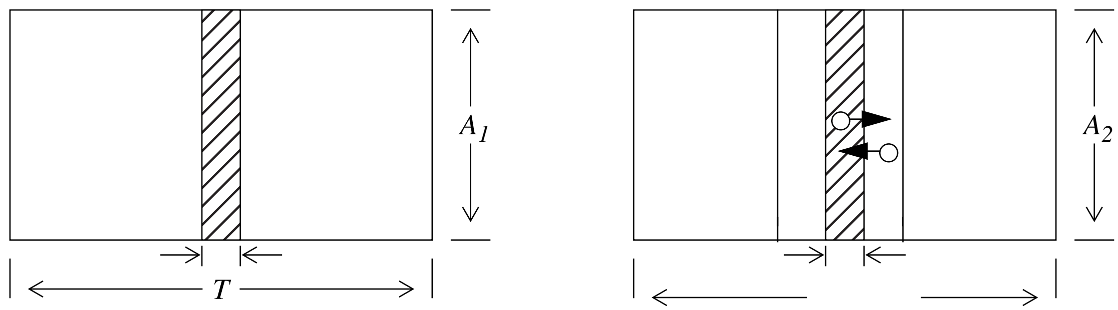


Figure 3: Derivation of High Level Synthesis AT Laws

lyze, and they directly lead to derivation of both register and interconnect laws. Since the gas is unchanged (it has the same internal forces as dictated by the corresponding CDFG structure and exercise the same force on the environment), the total pressure on all walls is still the same. This implies that the area on which CDFG gas acts with the pressure has to be the same. This area of the container corresponds in the high level synthesis domain to the AT product of execution units whose quantity therefore stays unaltered.

The consequences on register requirements is also easy to analyze. After the incremental increase in available time, all variables will have the corresponding increase in their life time. Those increases are the same, and therefore there is no change in density of simultaneously alive variables, which implies that area of register does not change with time alternation.

Finally, since there is no structure in interconnect pattern, and n units can communicate in $O(n^2)$ ways, the linear change in the number of execution units implies the quadratic change in the number of interconnects, establishing $AT^2 = const$ law for the number of interconnect.

3.0 Experimental Verification

We did comprehensive experimental study on the 80 DSP, video, image, telecommunication, control and information theory applications examples in order to empirically and statistically validated high level synthesis AT laws. The example were including three versions of the low pass 5th order elliptical wave digital filter, 7th and 9th order wave digital filters, Winograd Convolution, several different forms (direct, cascade, parallel, continued fraction and Gray-Markel ladder) 8th order IIR bandpass Avenhaus filters, several different Winograd transformations, several 1D and 2D DCT algorithms, VHS transform, NTSC coder, G^2 blend surface calculation using quintic polynomials, LMS adaptive filter, Hilbert Transform, 2nd order Volterra filter, 3rd order truncated Volterra filter, 28th, 50th, 60th, 100th, 126th and 133rd order FIR filters with a variety of transfer function and applied windowing techniques, PCM DA converter, image histogram, two GE 5th order linear controllers, steam controller, power controller, 3 VSTOL airplane controllers, distillation plant controller, Goppa, Reed-Muller, Fire, Hamming and BCH code.

A particular attention was paid to ensure a great variety of structural characteristics among considered examples. The diversity of the examples with respect to the number of important design parameters is shown in Table 4. On the half of examples a set of transformations for either speed, area or power optimization was applied in order to additionally diversify analyzed examples and more closely emulate high quality design practice to extensively optimize design at behavioral level.

All examples were studied using the following procedure. Each example was scheduled in at least 5 different available times, starting from the fastest possible (when the available time is set to the critical path time) to the most inexpensive possible implementation (when for each type of resources only one instance of each execution unit type is used). However, in the analysis of results presented in the reminder of this section, only implementations where the available time is at least 4 control steps an at least 10% longer than the critical path are considered, for the reasons explained in detail in the next section.

Parameter	min	max	average	median
Number of nodes	20	7,376,121	145,000	159
Number of node types	3	5	4.07	4
CP/ Available Time	0.01	1.0	0.43	0.44
Iteration bound	1	133	7.1	4
Resource Utilization	19%	88%	42%	41%
Area Reg/Exu	0.03	7.14	1.52	1.02

Table 5: Physical parameters of the analyzed examples

By far the most convincing experimental confirmation was obtained for the register AT laws. For 69 examples (86%) the greatest discrepancy in the number of registers at any two different available times was less than 10%. On the almost one quarter of examples (19) there was no discrepancy at all. On only 5 examples (6%) the discrepancy was larger than 15%, and the largest discrepancy was only 23% There were no example with more than 30 registers which had discrepancy large than 10%.

The results for the area of the execution units showed larger discrepancy, but still it was apparent that even for small examples the proposed AT law is exceptionally good approximation. For more than half of examples (45) the largest discrepancy was smaller than 10%. For only 8 examples the largest discrepancy was greater than 25%. The largest discrepancy was 50%. When example with more than 10 units were considered all examples were within 11% discrepancy. T For example with more than 20 units, the largest discrepancy was less than 5%. here are two reasons that explain why the register AT law was superior. The first reason is that in all examples, the number of registers was significantly larger than the number of execution units. Recall that the laws were derived under the assumption of the large number of instances of the resource. The second reason is that is several examples the area of the execution units was dominated by the area of the big multiplier, and a change of the number of multipliers between too close available time, was resulting in big difference in the corresponding AT product.

We did not find one single example for which the interconnect AT^2 law was valid. Obviously, in real life examples it does not happen that interconnect pattern is fully random and fully global. However, for all examples the number of interconnects was between the AT and AT^2. It is interesting to note that actually almost all examples were following curves dictated by the functions of the type AT^r, where r was varying between 1.2 and 1.5. Nevertheless, we decided to preserve the original AT^2 interconnect law in our presentation for the sake of simplicity of intuitive reasoning and because the CDFG gas model implies it. Also, the interconnect AT law convincingly states (and experimental study confirms) that the rate at which interconnect is changing with the available time is the fastest pace, which has a number of qualitative consequences as explained in the next two sections.

The additional experimental results which confirm validity of the high level synthesis AT laws are presented in the next section. For validation purpose the most relevant results are one which statistically validate prediction models (See Section 4.1.1).

4.0 Applications of High Level Synthesis AT Laws for Design Space Exploration and in ASIC Design Process

This section presents a brief description of a wide spectrum of applications of the AT high level synthesis laws in high level synthesis, retargatble compilation, system level synthesis, and as design methodology tool. The applications can be divided in two groups. The first group covers the application of the high level synthesis AT laws for CAD tool development. The second group includes application where the AT laws are used for efficient encapsulation of design knowledge, which consequently can be used either to guide design process or to reason about fundamental qualities of high level synthesis approaches and algorithm.

4.1 Using High Level Synthesis AT laws for Estimations and Development of High Level Synthesis Algorithms

The applications of the AT laws itself can be divided in two categories. The laws can be used to derive time efficient, accurate and consistent prediction tools. The tools can be used as objective function for conducting optimization on higher level of abstraction (Section 4.1.1). The AT laws can be also used to help development of the algorithms for high level synthesis. In particular, the AT laws describe well the solution space topology. So, they can indicate how efficient algorithms for high level synthesis tasks can be build or they can be used to detect that due to the complex structure of solution space, the development of efficient algorithm is precluded, and implicit enumeration methods are required for best result (Sections 4.1.2 - 4.1.4).

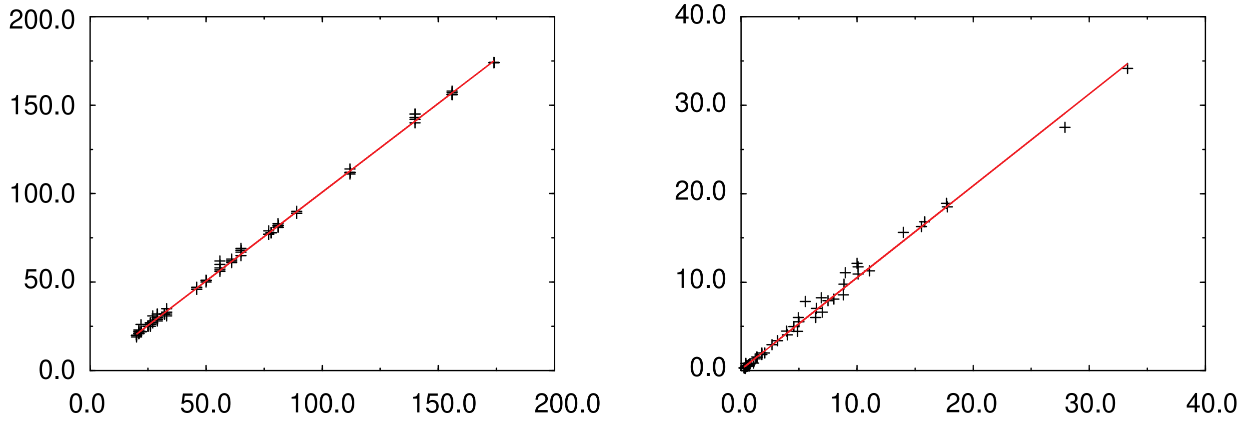
4.1.1 Estimation (Prediction) Tools Development

Fast and accurate prediction (estimation) tools are rapidly gaining recognition as an important integral component of high level synthesis [Jai89, Rab91, DeM93]. The AT high level synthesis laws provide a basis for development of prediction tools with a variety of accuracy/speed trade-offs.

For example, if the highest priority is rapid prediction, we can use the following two step procedure for the estimation of the area of execution units for some arbitrary available time *T1*.

(1) *Allocate for each type of execution unit resource only one instance.*

(2) *Schedule the given CDFG in the shortest possible time under the outlined resource allocation resources.* Note that now scheduling tasks can be done much more rapidly, than in general scheduling and assignment case, since the whole assignment is fixed and known.



(a) (b)

Figure 4: (a) Building prediction models using the AT laws theory: Predicting the number of registers; (b) Building prediction models using the AT laws theory: Predicting the area of execution units

(3) *Using that AT = const for execution units, calculate the area requirements for given T1.*

Similar procedure can be used for the number of interconnect (which can be done only in one way when only one instance of each type of execution units is available) and registers. However, note that for the case of interconnect it is mandatory to obtain the measure of globality of transfers in the corresponding CDFG. Figures 4a and 4b show the prediction models using linear regression. Exceptionally good prediction is apparent. For example, the correlation analysis shows that the correlation between the prediction variable and data obtained after scheduling is higher than 99% for both registers and execution units model build using the AT laws.

The AT laws prediction tool can be directly used in many high level synthesis tools to replace currently used objective functions. For example, it can be used for algorithm and architecture selection, partitioning, transformations, and allocation, assignment, and scheduling. For details of all mentioned applications of estimations tools, see [Rab91, Rab94]. In many instance the AT laws provide more time efficient, accurate and consistent tools. For example, the analysis for the prediction of register area using the Hyper prediction tools for the application of transformations and AT laws-based prediction tool show 88% and 99% correlation levels respectively. Taking into account that the AT estimation tool requires only one scheduling when assignment is given, it also has significant efficiency advantage.

4.1.2 Module Selection for Area Optimization

The module selection problem can be defined in the following way. Given a library of modules, so that for each functionality several different module are available, select the set of modules which minimizes the implementation area under a given throughput. The optimization trade-off is based on the following observation. If we select fast modules, they will have large areas. However, now we can select shorter cycle clock time, and therefore increase the number of clock cycles and the required number of units. Vice versa, if we select slower modules, they are smaller, but the smaller number of control cycles is available, so the number of units increases. The complexity of the problem results from the fact that with change of modules, the demand for interconnect also changes. The AT laws can be used to rapidly analyze trade-offs for different selections, and also to reduce search space.

We denote by CT_1 and CT_2 two different cycle times. The area of designs for CT_1 and CT_2 can be expressed using the following equations

$$A(CT_1) = A_R(CT_1) + A_E(CT_1) + A_I(CT_1) \quad \text{(EQ 2)}$$

$$A(CT_2) = A_R(CT_2) + A_E(CT_2) + A_I(CT_2) \quad \text{(EQ 3)}$$

A_R, A_E, and A_I denote respectively registers, execution units, and interconnect area. If we denote the number of different types of units by NE_j then

$$A_E = \sum_{i=1}^{j} A_{Ej} \cdot NE_j \quad . \quad \text{(EQ 4)}$$

It is easy to derive that the number of execution units, and interconnect area for cycle time CT_2, are:

$$NE_j(CT_2) = NE_j(CT_2) \cdot \frac{CT_2}{CT_1} \quad \text{(EQ 5)}$$

$$A_I(CT_2) = A_I(CT_1) \cdot \left(\frac{CT_2}{CT_1}\right)^r \quad \text{(EQ 6)}$$

So, now we can easily develop an **algorithm for area optimization using module selection**:

1. *Find $A_R(CT_1)$, $A_E(CT_1)$, and $A_I(CT_1)$*

2. *Propose new set of modules and find new CT_2 (time required to execute all modules)*

3. *Find $A_R(CT_2)$, $A_E(CT_2)$, and $A_I(CT_2)$, using EQ4, EQ5, and EQ6*

Note that, since the clock times for different modules is discontinuous function, implicit enumeration of all solutions is mandatory. Table 6 gives improvements over the initial selection for 4 examples.

Example	Initial Area [mm^2]	Area after Optimization [mm^2
steam controller	82.34	43.72
6th order IIR filter	10.90	6.67
aircraft controller	25.62	12.12
12th order IIR filter	25.73	15.28

Table 6: Area Optimization using AT-laws-based module selection algorithm

The average area reduction by 44.8%.

4.1.3 Clock Cycle Selection for Power Optimization

The power optimization is increasingly important optimization goal. Until now, in high level synthesis only influence of transformations on power reduction has been discussed [Cha92]. The AT laws provide a suitable platform for treating power minimization using several other high level synthesis tasks. In this subsection, we discuss an approach to minimize power using clock cycle selection. Clock cycle selection influence power consumption due to the following trade-off.

In the first approximation, the power minimization can be treated as the product of effective capacitance and the square of supply voltage [Cha92]. Both effective capacitance and voltage depend on cycle time. If clock cycle time is longer, we can reduce voltage, and therefore power. However, at the same time, for a fixed sampling rate, the number of control steps is reduced, and therefore more interconnect is needed, which effectively increases the effective capacitance.

Example	initial power [nJ/sample]	power after optimization [nJ/sample]
steam controller	39.2	12.3
6th order IIR filter	40.2	14.6
aircraft controller	117	30.6
12th order IIR filter	89.2	33.1

Table 7: Power reduction using the AT-laws based algorithm

The same algorithm as used for module selection for area minimization, can be used for the new task. However, now we can apply binary search on clock cycle time to find optimal solution. This algorithm is optimal due to the following observation. Capacitance is non-decreasing and voltage is non-increasing function as cycle time is increasing. The product of two monotic functions is unimodal function. For unimodal function binary search is the optimal polynomial complexity algorithm [Thi88].

Table 7 shows the application of the algorithm on 4 examples. The average power reduction was by factor of 3.11 times.

4.1.4 Hierarchical Allocation and Scheduling

With a steady increase in the size of ASIC designs, the need for hierarchical scheduling in high level synthesis is becoming more and more apparent [Pot89]. Also, as the exploration of the connection between retargetable compilation [Mar93] and high level synthesis has been gaining research momentum, the relevance of hierarchical treatment of scheduling is additionally emphasized. This is so, because a typical application of retargetable compilers is a large, and complex program. Hierarchical scheduling is scheduling done after partitioning of the complete computations in a number of blocks. Each block is scheduled using standard scheduling techniques. The goals is that after all block schedules are combined, the global schedule is done so that required amount of hardware is minimized.

The complexity of hierarchical scheduling is the consequence of not only a need to allocate hardware resources in a such way that they are suitable for implementation of all blocks, but also due to need for an additional level of optimization to distribute available time between all the blocks. The hierarchical scheduling problem can be defined in more formal way as follows. The computation has n blocks. For block b_i available time t_i is allocated. The area of design for block i is given $A_i(T_i) = A_R(T_i) + A_E(T_i) + A_I(T_i)$, where Ai denotes to total area of datapath, and A_R, A_E, and A_I denote respectively registers, execution units, and interconnect area. T_i is time allocated for i-th block. The throughput constraint imposes $T = \sum_{i=1}^{n} T_i$. The goal is to distribute the available time to blocks in a such way that the area of design is minimized. It is easy to see that the area of design can be expressed as

$$A(T) = \max_{i=1,n}(A_{Ri}(T_i)) + \max_{i=1,n}(A_{Ei}(T_i)) + \max_{i=1,n}(A_{Ii}(T_i))$$

The key idea to effective solution of the problem is to solve the dual of the problem. For a given amount of resource, using the AT laws it is to calculate the required time for each block. If sum of all times is higher than the sampling rate, the proposed hardware is not sufficient, otherwise we can either reduce amount of hardware, or start with scheduling individual blocks. We tried the proposed algorithm in the framework provided by Hyper hierarchical scheduling tool. The run time of scheduling was improved by more than two order of magnitude, with no decrease in the quality of solution.

4.2 High Level Synthesis Knowledge Encoding

As high level synthesis matures, a growing need for development of its sound theoretical foundation arises. Use of high level synthesis tools at system level design [Gup94] additionally emphasis this need. The AT laws are an effective way to encapsulate a part of high level synthesis knowledge. In this subsection we demonstrate how AT laws can be used to guide design process and to enable reasoning about high level synthesis approaches and algorithms. This subsection illustrates both mentioned types of applications of the AT high level synthesis laws.

4.2.1 Throughput-Efficiency Relationship

We start by identifying criteria for positive correlation between throughput and efficiency. There are at least two situations when the AT product can be significantly improved with simultaneous increase of the throughput of the implementation:

1. When the cost of design is dominated by the cost of registers. It can happen, when a high level of hardware sharing is employed, or when the cost of registers is relatively high to the cost of execution units and interconnect (e.g. for fixed point computation without multiplication which uses relatively few bits in their word-length).
2. The second situation can be obtained by analyzing when an assumption used in the derivation of the high level synthesis the AT laws are not fulfilled. For example, when some type of nodes (in particular with high implementation cost, e.g. multiplications) have very few instances in the considered CDFG. For example, if we have only 2 multiplications in the CDFG and use 10 control steps, the design is bound to be inefficient. Changing the available time to very few control steps most likely will dramatically improve resource utilization, by a virtue of avoiding mandatory idle multipliers in the high percentage of control steps.

Table 8 shows 5 examples which clearly confirm the first claim at the RT level. The reported area is the

Example	T = 20	T = 40	T = 60	20AT	40AT	60AT
FIR Hamming	11.26	6.84	6.17	225	274	370
DS FIR	8.90	7.55	6.50	178	302	390
iir6	4.44	3.26	2.00	89	130	120
gm	2.03	1.68	1.18	41	67	71
wdf9	-	2.14	1.86	-	86	112

Table 8: Improving the AT products due to constant register requirements. T is the available time.

active area which included the area of execution units, registers, and multiplexers. Table 9 reports data at

TIME	1	2	3	4	6	10	15
AREA	73.58	84.90	75.90	73.83	62.40	23.83	7.44
AT	74	170	228	295	374.40	238	112

Table 9: Improving the AT products - Physical layout Data - COLOR transform example

the physical layout data for the COLOR transform example. The AT product has improved more than twice as time is reduced from 6 control steps to 2 control steps. The implementation with no hardware sharing (when the available time is 1 control step) is even smaller, but here the analysis indicates that the

TIME	EXU AREA	AREA	AT product
15	13.44	23.42	351
20	13.19	22.65	453
30	12.69	21.22	637

Table 10: The active area of the 2nd order pipelined Volterra filter

improvement is coming mainly from the absence of the complex hardware sharing control logic. The AT product of the two slowest implementations is reduced due to drastically reduced interconnect. The typical example which supports the second claim is the 2nd order pipelined Volterra filter where we see only slightly more than 10% change in the area as the throughput is doubled as shown in Table 10. This is due to the fact that for all implementations only one multiplier is sufficient and dominates the total area

4.2.2 Interconnect importance

Next, the AT laws reaffirm well known folklore observations that the importance of local interconnect can not be overstated in large high performance designs. Since interconnects grows as the second order polynomial with the throughput improvement, while register requirements are constant and execution unit cost grows only linearly with increased throughput, it is apparent that interconnect cost will eventually dominate cost of high performance designs. The results from Table 11 strongly support this conclusion.

Example	T = 20		T = 40		T = 60	
	Mux	Tot	Mux	Tot	Mux	Tot
FIR32	3.02	7.41	0.85	4.73	0.52	4.47
Conv5	1.54	3.45	0.37	1.51	0.22	1.36
ellip	3.13	6.69	0.84	3.36	0.35	2.15
126FIR	10.85	14.27	2.61	6.68	0.62	3.38

Table 11: Importance of optimizing Interconnect at high speed. T is the available time. The area of multiplexers (Mux) is rapidly becoming the dominant part of the overall implementation even when only associated control logic is considered

For example, we see that in the NEC FIR filter with 126 taps the percentage of the multiplexers as the part of the active area increases from less than 20% when 60 control cycles are available to more than 75% when the available time is reduced to 20 control steps.

4.2.3 Benchmark and Driver Example Selection

Finally, there is one more very important and on the first view paradox consequence of the high level synthesis AT laws: If high resource utilization is targeted, implementation in the available time equal to or close to the critical path should be vigorously avoided. For this type of implementation, frequently much higher additional cost has to be paid for very small increase in performance. The correct globally optimal high level synthesis approach in this situation is, of course, to employ transformations or other high level synthesis tools (e.g. chaining of operations) and make at least some discrepancy between the longest path in the CDFG and the sample rate.

Scanning the high level synthesis literature it is evident that almost all published benchmark examples are done so that available time is either set to the critical path or just slightly longer. The most likely reason is that faced with the lack of the precise throughput requirements, researchers in high level synthesis were considering this choice of available time “natural”. Also, almost all early benchmark designs were small and if higher discrepancy between sample rate and critical path was assumed, CDFG can be easily scheduled using only 1 unit of each type, making differentiation among proposed schedulers impossible.

But even without insights provided by the high level synthesis AT laws, there is an abound (unnoticed publicly until now) evidence that critical path implementation has as a direct consequence very low resource utilization. For example, a number of high level synthesis systems during implementation of the 5th order elliptical wave digital filter in 17 control step requires 3 adders and 3 multipliers, which is proven to be the minimum hardware configuration. When 21 control steps are available, only 2 adders and 1 multiplier are sufficient. So, we see that for 19% increase in throughput, almost 3 times higher price (multiplier is significantly more expensive than adder in fixed point computation used in filters) in the number of execution units has to be paid. The difference in required interconnect is even more dramatic. Analysis of the results reported in Tables 12 and 13 once again convincingly demonstrates the correctness of the consequence of the AT laws.

Name	CP	CP Area	Time T1	Area at T1	CP/T1	CP Area/ Area at T1
iir11Cheb	28	88.67	40	31.83	0.70	2.79
wave1	16	8.06	20	3.01	0.80	2.68
wave5	15	8.57	20	4.50	0.75	1.90
IIR-df	18	67.50	20	23.40	0.90	2.88

Table 12: Exceptionally high cost when scheduling in the critical path time is used: The RT level data. The reported area is the active area

High level synthesis researchers and developers can also use the developed AT laws in many ways. Due to space limitation, we will mention just few of them which are either the most apparent or the most important.

Name	CP	CP Area	TimeT1	AreaT1	CP/T1	CPArea/ AreaT1
cascade	10	67.54	25	8.19	0.40	8.28
GM	21	13.22	25	8.84	0.84	1.50
GM	21	13.22	30	5.15	0.70	2.57
CF	28	36.94	30	18.41	0.93	2.01

Table 13: Exceptionally high cost in scheduling in the critical path time Physical Layout Data. All area are in mm^2

4.2.4 Transformation Selection

The AT laws indicate very strongly and clearly which component of the overall implementation cost (registers, execution units or interconnect) should be targeted during transformations. For example, as we already noticed even when the goal is resource utilization, if the available time is equal or very close to the critical path, the most promising way to reduce the implementation cost in time shared designs is to reduce the critical path, and significantly higher efficiency will be direct consequence of the AT laws. We can say even more precisely when reducing critical path is a very efficient approach for improving the area of design. It is in designs where the dominating component of the area cost are either in interconnect or execution units [Pau89].

On the other hand, when we have a significant discrepancy between the available time and the critical path and the cost of design is register dominated, pipelining should be avoided. This is so because, the improvement in execution units cost is not sufficient to offset the additional cost in registers. Once again the analysis of the results from high level synthesis literature provides an ample confirmation of this claim [Pot92].

4.2.5 Theoretical Evaluation of CAD Approaches and Algorithms

We will finish this brief outline of the potential applications of the AT laws in high level synthesis and design methods by pointing out their surprising and powerful ability to evaluate not just particular set of tools, but also generic methodology. For example, it can be used to identify conceptual flaws in the basic

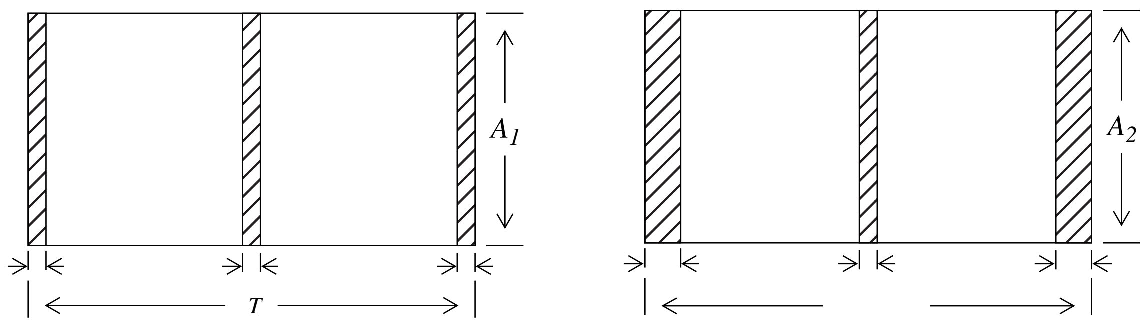


Figure 5: Deriving high level synthesis AT laws using the uniform probability assumption

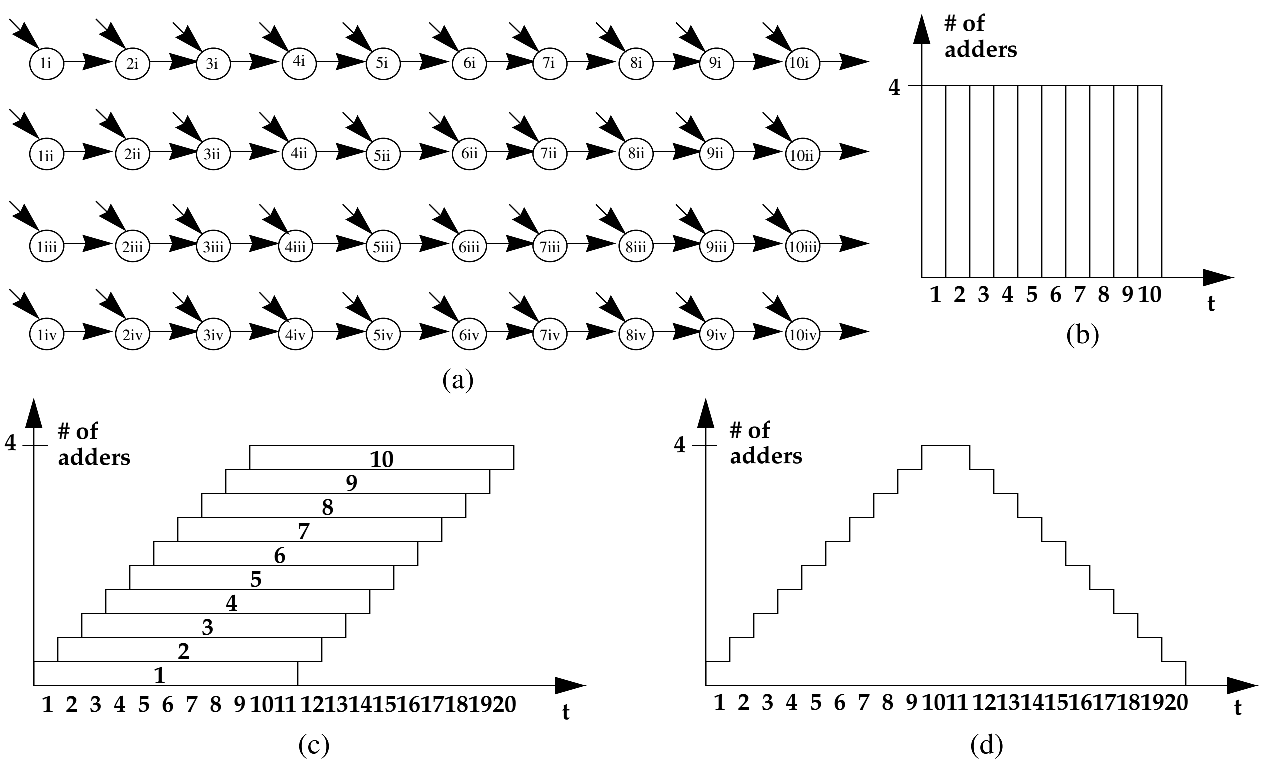


Figure 6: Why uniform distribution is providing incorrect prediction: (a) The explanatory example; (b) Density of operation when the available time is 10 control cycles; (c) and (d) density of operations when the available time is 20 control cycles

assumptions on which one or more high level synthesis tools are build. We will use the popular and intuitively very appealing force directed scheduling as a driver example. We will derive again the high level synthesis AT laws, this time starting from the assumption, used in the force directed scheduling that the probability that a particular operation will be used in a given control step is equal to $1/(ALAP - ASAP + 1)$ for the control steps in the range [*ALAP*, *ASAP*], and zero otherwise. (For definitions of *ASAP* and *ALAP* times and excellent exposition of force directed methodology see [Pau89]). This implies that the most likely control step when an operation is bound to be scheduled is $(ALAP + ASAP)/2$. This has as a consequence that when the available time is increased by ΔT, each operation most likely is bound to be scheduled in $(ALAP + ASAP)/2 + \Delta T/2$ step. Although, the range in which each operation can be scheduled will increase for ΔT, and the density of probability of each operation will proportionally decrease sum of densities of all operations for control steps which are close to boundary will not change. This is so because each range τ will export to adjacent ranges the same amount of density, as it will receive as the consequence of the fact that the adjacent strips are also exporting. Therefore, only densities which are close to the boundary of the CDFG will experience decreased density, and therefore the area of the used units will not be altered, as shown in Figure 5. This leads to counter-intuitive $A = const$ law for execution units, which incorrectness can be easily verified on all examples presented in high level synthesis.

To shed more intuition why the just outlined reasoning is indeed correct consider the example in Figure 6. It shows the example which consists of four parallel chains of 10 additions. If the available time is 10 control steps and if addition takes one control step then the uniform distribution implies that we need 4 adders. This is correct conclusion. However, if the available time is 20 control cycles then each operation can be scheduling with the equal probability in one of eleven cycles. For example, the first operations in the chains can be scheduled between the first and the eleventh control step. Figure 6c shows the density which corresponds to the probability of scheduling in some particular step. When we sum those probabilities as shown in Figure 6d, we see that the maximum density is still implying need for 4 adders, although the available time is doubled and obviously only 2 adders are sufficient. Therefore, we see that uniform probability is leading again to the incorrect conclusion and does not depict design process appropriately.

It is very important to carefully analyze all consequences of the just presented reasoning and prevent dangerous wrong conclusions. The analysis, by no means, does not imply that the force directed high level synthesis approaches are conceptually incorrect and bound to produce pure results on large examples. Already, there is a vast experimental evidence and a clear intuitive understanding that backbone optimization structure which first targets steps when the density of operation is the highest is powerful heuristics and that the method is effective. Actually, this backbone is so effective that even serious inaccuracies in some of used calculations for probabilities when some operation is most likely to be scheduled often do not prevent the generation of very high quality solutions.

The only component which is actually inappropriate in the force directed methods is the way how the probability that a particular operation will happen in a particular step is calculated. More precisely, obviously, the uniform distribution of this probability in *ASAP-ALAP* interval is often very far from correct. Incorporation of CDFG gas model or some other more adequate probability distribution in the powerful optimization framework where timing and resource constraints forces are balanced will further enhance level of competitiveness of this already in practice proven approach.

5.0 Conclusion

We established, using a novel CDFG gas model, the following three fundamental AT laws in high level synthesis: $A = const$ for registers, $AT = const$ for execution units, and $AT^2 = const$ for interconnect. The laws are experimentally verified using the detailed statistical study on a large number of real life examples.

Using the AT laws, the estimation tools and the number of high level synthesis algorithms are build, including tools for area optimization using module selection, power optimization using clock selection, hierarchical scheduling, and an instance of hardware-software codesign cost optimization. It has been also demonstrated that the high level synthesis AT laws can be used a key basic building block for development of theoretical foundation for high level synthesis, retargetable compilers and system level synthesis.

6.0 References

[Amd64] G.M. Amdahl, G.A. Blaauw, F.P. Brooks: "Architecture of the IBM System/360", IBM Journal of Research and Development, Vol. 8, No. 2, pp. 87-101, 1964.

[Ave72]E. Avenhaus: "On the design of digital filters with coefficients of limited word length", IEEE Trans. on Audio and Electroacoustics, Vol. 20, pp. 206-212, 1972.

[Bar73] M. Barbacci: "Automatic Exploration of the Design Space for Register Transfer (RT) Systems, Ph.D. Thesis, Department of Computer Science, Carnegie-Mellon University, November 1973.

[Bro92]. R.W. Brodersen, ed.: "Anatomy of a Silicon Compiler", Kluwer Academic Publishers, Boston, MA, 1992.

[Bro75] F.P. Brooks: "The mythical man-month: essays on software engineering", Addison-Wesley, Reading, MA, 1975.

[Cas78] R.P. Case, A. Padges: "The architecture of IBM System/370", Comm. of ACM, Vol. 21, No.1, pp. 73-96, 1978.

[Cha92] A.P. Chandrakasan, M. Potkonjak, J. Rabaey, R. Brodersen: "Hyper-LP: A Design System for Power Minimization using Architectural Transformations", IEEE International Conference on Computer-Aided Design, Santa Clara, CA, 300-303, November 1992.

[DeM92] H. De Man: "Design Technology Research for the Nineties: More of the Same?", Euro-DAC '92, pp. 592-596, Hamburg, Germany, 1992.

[DeM93] G. De Micheli: "Extending CAD tools and techniques", IEEE Computer, Vol. 26, No. 1, pp. 85-87, 1993.

[Dev89] S. Devadas, A.R. Newton: "Algorithm for Hardware Allocation in Data Path Synthesis", IEEE Trans. on CAD, Vol. 8, No. 7, pp. 768-781, 1989.

[Don81] W.E. Donath: "Wire length distribution for placement of computer logic", IBM Journal of Research and Development, Vol. 25, No. 3, pp. 152-155, 1981.

[Fey65] R.P. Feynman, et al.: "The Feynman Lectures on Physics", Addison-Wesley, Redwood City, CA, 1965.

[Gaj92] D. Gajski, N. Dutt, A. Wu, S. Lin: "High-Level Synthesis: Introduction to Chip and System Design", Kluwer, Norwell, MA, 1992.

[Gro53] H.R.J. Grosch: "High Speed Arithmetic: The Digital Computer as a Research Tool", Journal of the Optical Society of America, Vol. 43, No. 3, pp. 306-310, 1953.

[Gup94] R.K. Gupta, C.N. Coehlo, G. De Micheli: "Program Implementation Schemes for Hardware-Software System", IEEE Computer, Vol. 27, No. 1, pp. 48-55, 1994.

[Han88] D.L. Hanson: "Interconnection Analysis", pp. 31-64, in "Physical Design Automation of Electronic Systems", ed. by B.T. Preas, M.L. Lorenzetti, Benjamin/Cummings, Menlo Park, CA, 1988.

[Hir65] J.O. Hirschfelder, at al.: "Molecular Theory of Gases and Liquids", John Wiley, New York, NY, 1965.

[Jai89] R. Jain: "High-Level Area-Delay Prediction with Application to Behavioral Synthesis", Technical Report CENG 89-23, Electrical Engineering- Systems Department, University of Southern California, Los Angeles, CA.

[Mar93] P. Marwedel: "Tree-Based Mapping of Algorithms to Predefined Structures" IEEE ICCAD-93, pp. 586-593, 1993

[McF90a] M.C. McFarland, A.C. Parker, R. Camposano: “The High-Level Synthesis of Digital Systems”, Proc. of the IEEE, Vol. 78, No. 2, pp. 301-308, 1990.

[McF90b] M.C. McFarland, T.J. Kowalski: “Incorporating Bottom-Up Design into Hardware Synthesis”, IEEE Trans. on CAD, Vol. 9, No. 9, pp. 938-950, 1990.

[Par91] A.C. Parker, P. Gupta, A. Hussain: “The Effect of Physical Design Characteristics on the Area-Performance Trade-off Curve”, 28th ACM/IEEE Design Automation Conference, pp. 530-534, 1991.

[Pau89] P.G. Paulin, J.P. Knight: “Force-Directed Scheduling for the Behavioral Synthesis of ASIC’s”, IEEE Trans. on CAD, Vol. 8, No. 6, pp. 661-679, 1989.

[Pot89] M. Potkonjak, J.Rabaey “A Scheduling and Resource Allocation Algorithm for Hierarchical Signal Flow Graphs”, 26th ACM/IEEE Design Automation Conference, Las Vegas, NV, pp. 7-12, June 1989.

[Pot92] M. Potkonjak, J. Rabaey: “Pipelining: Just Another Transformation”, Proc. 1992 Application Specific Array Processors”, pp. 163-175, IEEE Computer Society Press, 1992.

[Rab91] J.M. Rabaey, at al. “Fast Prototyping of Datapath-Intensive Architectures”, IEEE Design and Test, pp. 40-51, 1991.

[Rab94] J.M. Rabaey, M. Potkonjak: “Estimating Implementation Bounds for Real Time DSP Application Specific Circuits”, IEEE Trans. on CAD, Vol. 13, No. 6, to be published, 1994.

[Sas86] S. Sastry, A.C. Parker: “Stochastic models for wirability analysis of gate arrays”, IEEE Trans. on CAd, Vol. 5, No. 1, pp. 52-65, 1986.

[Sei90] C.L. Seitz: “Concurrent Architecture”, in “VLSI and Parallel Computation”, ed. by R. Suaya, G. Birtwistle, Morgan Kaufmann, San Mateo, CA, 1990.

[Thi88] R.A. Thisted: “Elements of Statistical Computing”, Chapman, New York, NY, 1988.

[Tho79] C.D. Thompson: “Area-time complexity for VLSI”, Symposium on the Theory of Computing”, pp. 81-88, 1979.

[Ull84] J.D. Ullman: “Computational Aspects of VLSI”, Computer Science Press, Rockville, MD, 1984.

[Wal91] R.A. Walker, R. Camposano: “A Survey of High-Level Synthesis Systems”, Kluwer, Boston, Ma, 1991.