BINDING, ALLOCATION AND FLOORPLANNING IN LOW POWER HIGH-LEVEL SYNTHESIS

A. Stammermann, D. Helms, M. Schulte

OFFIS Research Institute
D - 26121 Oldenburg, Germany
{stammermann, helms, schulte}@offis.de

A. Schulz, W. Nebel

Univ. of Oldenburg
D - 26111 Oldenburg, Germany
{arne.schulz, nebel}@uni-oldenburg.de

ABSTRACT

This work is a contribution to high level synthesis for low power systems. While device feature size decreases, interconnect power becomes a dominating factor. Thus it is important that accurate physical information is used during high-level synthesis [1]. We propose a new power optimisation algorithm for RT-level netlists. The optimisation performs simultaneously slicing-tree structure-based floorplanning and functional unit binding and allocation. Since floorplanning, binding and allocation can use the information generated by the other step, the algorithm can greatly optimise the interconnect power. Compared to interconnect unaware power optimised circuits, it shows that interconnect power can be reduced by an average of 41.2 %, while reducing overall power by 24.1 % on an average. The functional unit power remains nearly unchanged. These optimisations are not achieved at the expense of area.

1. INTRODUCTION

Recently, several research approaches have been reported taking physical information into account. Most of the proposed algorithms use floorplanning information in high-level synthesis to estimate area and performance more accurately [2, 3]. Similarly, a lot of techniques have already been proposed taking into account power consumption in high-level synthesis [4, 5, 6, 7]. Just a few of these contributions also consider interconnect power [8, 9, 10]. For high-level interconnect length estimation the well known Rent's rule is often used [11]. It states the relationship between the pin count (IO) and the block count (BK) of a chip $IO = AS * BK^r$. AS represents the average size of blocks within the chip, while r is a mystery quantity and is called the Rent's exponent. This model requires knowledge of empirical parameters that are computed from actual design instances. This limits the applicability and therefore we do not use Rent's rule.

This work evaluates an approach of simultaneous binding, allocation and floorplanning optimisation. Binding is the task of assigning compatible operations or variables to resources during the high-level synthesis. Allocation is the choice of the number of resources. In the following binding will denote the combination of binding and allocation. A low power binding is an assignment in which the power dissipation of the resources is minimal. Binding has a great influence on power dissipation, since different bindings lead to different input streams on the input of resources. Binding and allocation affect the area of the design, the netlist topology (beeing the basis of a floorplan) and the wire activity. In order to find a power optimal solution binding and floorplaning must be regarded simultaneously.

A precondition for combining binding and floorplanning is high estimation accuracy of the power consumption of RT-resources and interconnect. In order to determine the power consumption of resources power models describing the power consumption and area of the individual resources at RT level [12] are needed. Interconnect power primarily depends on the wire length of individual wires, the number of vias and the switching activity. We estimate the wire length by generating a slicing tree floorplan. Since a floorplan only affects wires connecting different RT-resources, only the global interconnect is considered. Wires within a resource are encapsulated by the power models.

We use a low power high-level optimisation tool, called ORINOCO [13, 14], to obtain the RTL circuits and the power consumption of the datapath. ORINOCO is interconnect unaware. It is amended by our new interconnect power estimation methodes detailed in section 4.

The paper is organized as follows: In section 2 we present a motivation example. In section 3 we discuss our RTL interconnect power estimation. The proposed optimisation methodology is described in section 4. An experimental evaluation is presented in section 5 and conclusions are drawn in section 6.

2. MOTIVATION

Fig. 1 illustrates the effect of different binding solutions on the interconnect length of a register-transfer level (RTL) design. A scheduled control data flow graph (CDFG) is given, which contains three generic types of operators: a, b and c and their corresponding functional units are A, B and C. Fig. 1 (a) shows a binding of the SDFG and the corresponding interconnect optimised floorplan. Operators within a grey bar are mapped on the same functional unit. In the floorplan the wires are annotated with their length. For simplification equal switching activity for all wires is assumed.

Fig. 1 (b) shows a different binding solution and the updated floorplan. b_3 is re-binded to B_1, b_4 to B_2 and c_1 to C_2. Thereby the total wire lenght decreases by 28 %. This clearly shows the importance of considering interconnect power during the binding step.

This work was supported by POET project, IST-2000-30125, funded by the European Union

ICCAD'03, November 11-13, 2003, San Jose, California, USA.

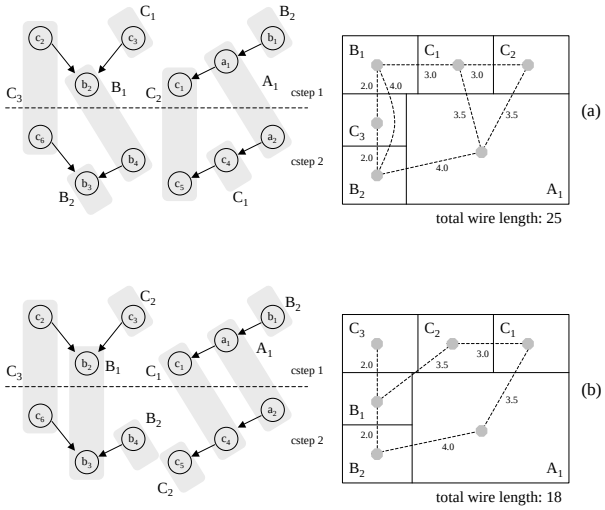

Figure 1: (a) Original binding (b) New binding and new floorplan

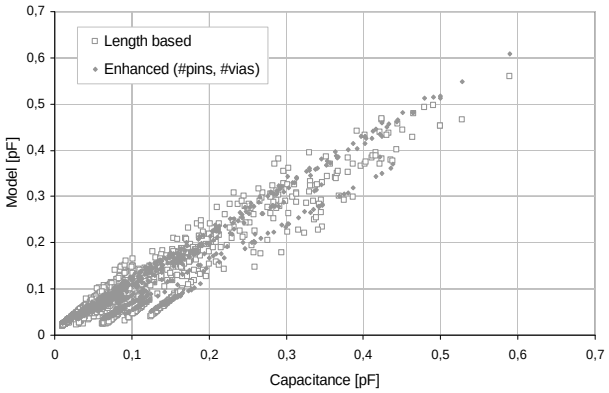

Figure 2: Capacitance model (fft benchmark)

3. RTL INTERCONNECT POWER ESTIMATION

Given is a scheduled CDFG and a set of allocated modules with specification of area and geometric information. Geometric information specifies the minimum and maximum aspect ratio of a module, representing the flexibility during floorplanning stage (cf. 4.1). To capture the physical meaning of data transfer the CDFG is transformed into a RT-netlist. The netlist is generated by an architecture extraction. Each functional unit is modeled as a 2-input 1-output combinational circuit and each register is modeled as a 1-input 1-output circuit. Every multiplexer is modeled as a n-input 1-output circuit with n greater than two (including control input). The netlist represents a multiplexer-based point-to-point interconnection.

The dynamic power dissipation of a VLSI interconnect with a capacitive load can be written as

$$P_{Inter} \approx \sum C_i \, D_i$$

where C_i and D_i are wire capacitance and switching activity for wire i. The switching activity extraction and the wire capacitance estimation used in our approach is discussed next.

3.1. Switching activity extraction

The paradigm of ORINOCO is to estimate the activity that is necessary to perform the functionality written in the source description. This means that for every point in time were an operation is executed, the current value of inputs is determined. These values and those gathered in the same way for the last operation are then used to compute the activity. In real systems additional activity, called spurious transitions, occurs at the input of FUs. Not all appearances of these transitions can be handled accurately at this high level of abstraction.

3.1.1. Glitches

Random logic introduces spurious transitions. These transitions cannot be effectively forecast. Due to this reason we assume that no chaining will be used. This is a sensible assumption for low power design, as otherwise the glitches introduced by the first unit will boost the power of the second [17]. This glitches also contribute significantly to output network power consumption. So far this effect is neglected.

3.1.2. Register Timing

Within the described paradigm it is implicitly assumed that both new values of an FU are applied at the same time. This is often not the case, as registers in the fanin of the FU are written in different cycles and values become visible immediately after writting. This effect is handled accurately. Due to variations in timing this phenomenon would even occur if both registers were written in the same cycle. This situation however can not be handled correctly.

3.1.3. Shared Registers

Shared registers output a merged data streams of all values mapped to them. Additional switching is produced. This effect is also handled accurately.

3.1.4. Input multiplexing

Input multiplexing occurs due to control structures and sharing. Let us consider two multiplexed data streams that go to one input of an FU
$b_1 \rightarrow b_2 \rightarrow \ldots \mid e_1 \rightarrow e_2 \rightarrow \ldots$.
The right order of these values would be
$b_1 \rightarrow e_1 \rightarrow b_2 \rightarrow e_2 \rightarrow \ldots$.
In this case only the necessary transitions would occur. For timing it may however be more convenient to execute
$b_1 \rightarrow e_1 \rightarrow b_1 \rightarrow b_2 \rightarrow e_2 \rightarrow \ldots$.
In this case b_1 is seen twice. This happens because the multiplexer switches back to the b data stream before the new value of b is produced. The problem is that the semantic of a behavioural description does not include the behaviour of the multiplexers at times where the results of the attached functional unit are not needed. It is therefore possible to extrapolate the behaviour at will. Our assumption here is a 'lazy' controller. That means that the controller switches the inputs as late as possible.

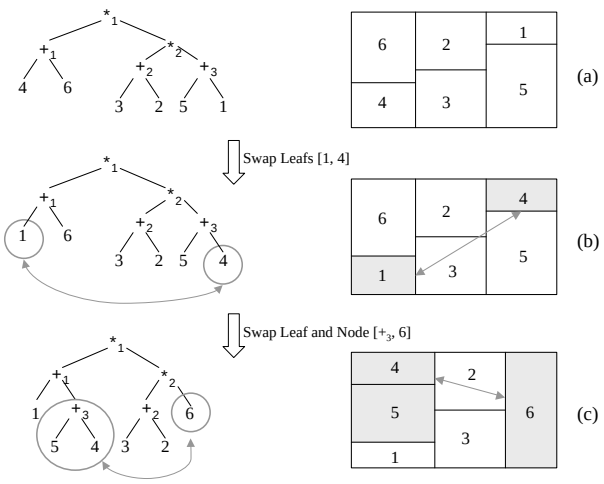

Figure 3: (a) Initial floorplan (b) Floorplan after F_1 (c) Floorplan after F_3

3.2. Wire capacitance estimation

We derive the wire capacitance by using a capacitance model. This model is based on wire length, number of pins and number of branch points. We use a linear regression technique to model the dependencies. *Pins* are the connecting points to RT-resources, e.g. a wire at the input of a multiplier is connected to about 6 gates, that is 6 pins. The number of pins depends on the RT-resource type and can be extracted from the corresponding RT-model. The number of branch points and the wire length is extracted from a floorplan (cf. 4.1).

In Fig. 2 the capacitance extracted from Cadence Silicon Ensemble® is plotted against the capacitance from our model. The unfilled dots are from a model based only on the wire length. It is observable that besides the wire length the number of pins and branch points is a second major contributor for the overall wire capacitance. This impact on the overall capacitance is due to the additional vias for further branches and pins. For the used 0.25 μm technology the enhanced model has an average std. deviation of 31.9 % and the length based model has an average std. deviation of 36.8 %.

4. INTERCONNECT DRIVEN HIGH-LEVEL SYNTHESIS

In this section we propose our high-level synthesis flow, which performs simultaneously slicing-tree structure-based floorplanning and functional unit binding.

4.1. Simulated annealing (SA) based floorplanner

For interconnect length estimation an extension of a well known SA based floorplanner by Wong and Liu [15] is used. Simulated annealing is an iterative technique for solving high-dimensional optimisation problems. These techniques switch from one solution (here: floorplan) to another solution in a well-defined way by using 'moves'. This algorithm considers slicing floorplans. A slicing floorplan is a floorplan that can be obtained by recursively cutting a rectangle by either a vertical line or a horizontal line into smaller rectangular regions. A slicing floorplan can be represented

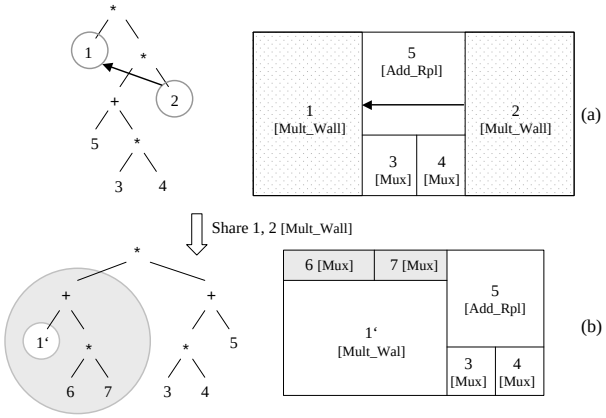

Figure 4: Share: Two resources are merged to one single resource

by an oriented rooted binary tree. Each internal node of the tree is labeled either $*$ or $+$, corresponding to either a vertical cut or a horizontal cut. Floorplan transformation is achieved by using five types of moves:

1. Swap Leafs F_1
2. Swap Nodes F_2
3. Swap Leaf and Node F_3
4. Shift Leaf or Node F_4
5. Switch direction F_5

Fig. 3 illustrates how the moves F_1 and F_3 affects the binary tree (left side) and shows the impact for the corresponding floorplan (right side). Each floorplan considered during SA process is evaluated based on area A and interconnect power P, using a cost function of the form $P + \lambda A$, where λA controls the relative importance of A and P. In [15] the interconnect length is estimated by calculating the Manhattan distance for two pin connections and the minimum spanning tree (MST) for connections with more than two pins. These technique does not suit real wiring because no branch points are considered. Instead, we use Steiner Trees for drawing data transfer wires. To treat the clock distribution network accurately an H-tree (balanced tree) is generated.

4.2. Extended approach

For our approach we modified the cost function and the SA process. The new cost function is of the form $P_{FU} + P_{wire} + \lambda A$. P_{FU} is the power consumption of the functional units, multiplexer and registers and P_{wire} is the power consumption of the interconnect. λA is the area's contribution to the cost function. The annealing process is amended by three new binding moves $B_1 - B_3$. In combination they are able to create every possible binding solution. Together with floorplan moves they allow a variation of the design architecture and corresponding floorplan simultaneously.

4.2.1. Share B_1

Share merges two resources res_1 and res_2 to one single resource. For such a move to be valid, res_1 and res_2 must be instances of the same type. Moreover, no operation performed by res_1 should have an overlapping lifetime with any operation of res_2. If the number

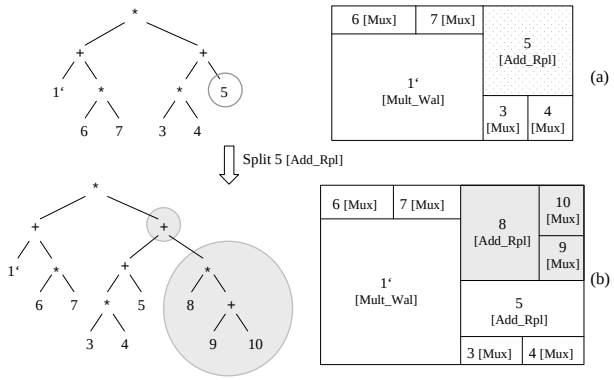

Figure 5: Split: A single resource is splitted into two resources

of sources at one input of a resource exceeds one, a new multiplexer is instantiated. If the number of sources decreases to one the corresponding multiplexer is dropped out. Sharing resources significantly affects both the switching activity in the data path and the network topology. In Fig. 4 the multiplier 1 and 2 are merged to one single multiplier $1'$. It is assumed that two new multiplexers, 6 and 7, had to be instantiated.

4.2.2. Split B_2

Split is the reverse of *share*. A single resource is splitted into two resources. Like in move B_1, multiplexers can vanish or appear. Splitting can be done without regarding the lifetime of operations. Apart from potentially reducing switched capacitance, these moves enlarge the avenues for applying other share moves. In Fig. 5 the adder 5 is splitted into the adders 5 and 8. It is assumed that two new multiplexers, 9 and 10, at the input of the new adder 8 had to be instantiated.

4.2.3. Swap B_3

Swap interchanges the inputs of commutative operations. Like in move B_1, multiplexers can vanish or appear. This move significantly affects the switching activity in the data path. The influence on the netlist is nearly negligible.

4.2.4. Balance point

New components are inserted at their balance point. The balance point is the point, where the new resource would produce the lowest interconnect power. In Fig. 6 (a) this point is inside the left half of leaf 4. Therefore leaf 4 is replaced by a new vertical node with the new leaf $5'$ placed on the left side and 4 on the right side. Our floorplanner supports soft macros, which means that leafs are flexible in their aspect-ratio. Therefore inserting or deleting a leaf does not destroy the floorplan. The unused area in Fig. 6 (b) only originates because we limited this ratio. This avoids unrealistic floorplans.

4.3. Optimisation algorithm

The algorithm itself consists of two nested simulated annealing (SA) processes (Fig. 7). The inner loop uses floorplan moves ($F_1 - F_5$) optimising the actual floorplan for interconnect power.

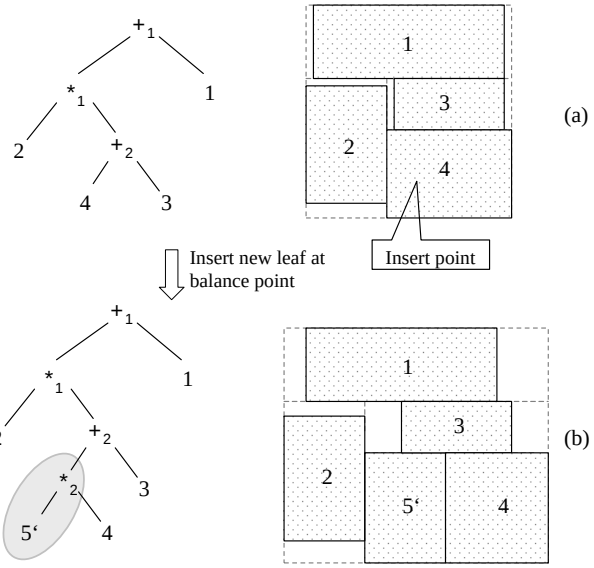

Figure 6: Inserting new leaf at balance point

The outer loop uses the binding moves ($B_1 - B_3$) optimising the actual architecture. By doing so every binding move is followed by a short floorplan annealing process, since binding moves can significantly affect the netlist topology and thus the interconnect power. The effect of a binding move can only be rated after a floorplan update. This can be done rapidly because of the modules' flexibility. Changes in the netlist topology are mended in the actual floorplan. In contrast to previous approaches, a time-consuming floorplan generation from scratch is not necessary.

In general an annealing move is chosen randomly. If a move leads to a decreased power consumption this move is accepted. If the power is not reduced the move may be acccepted on a probabilistic base. If a generated random number (0 - 1) is smaller than $e^{-\triangle\, Cost/T}$, where $-\triangle\, Cost$ is the power difference and T is the current temperature, the worse solution is accepted. This enables the SA to escape from local minima.

4.4. Annealing process acceleration

The algorithm keep on searching until some stopping criteria are met. Stopping criteria are: (1) the last k iterations did not identify a better solution and (2) some parameter have reached a threshold limit. Unfortunately for the most practical applications the runtime is out of scale. To cope with this we integrated some effective speedups (Fig. 8).

4.4.1. Constructive heuristic

Our iterative algorithm starts with a constructive heuristic to generate a pre-optimised solution (Fig. 8 (1)). The heuristic optimises the binding and floorplan separately. In the first step the architecture binding is optimised neglecting the interconnect power. The inner loop (update floorplan) is excluded. Now for this architecture the power optimal floorplan is chosen by executing only the inner loop.

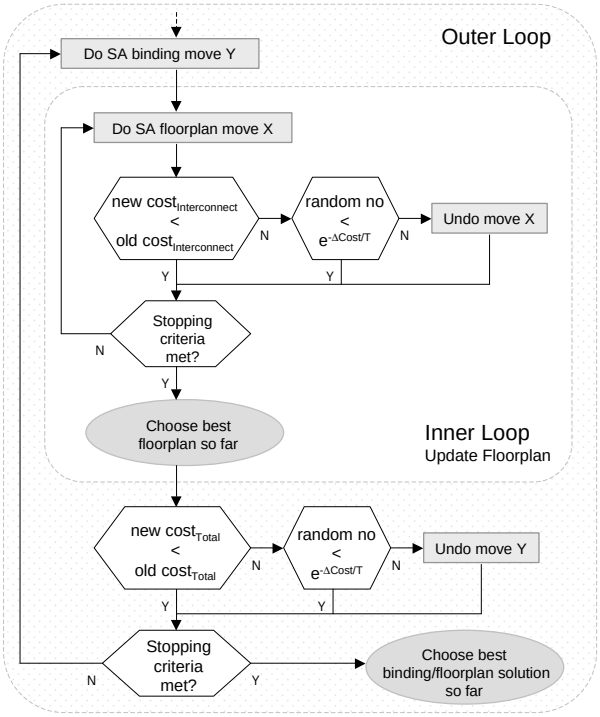

Figure 7: Flowchart of the optimisation algorithm

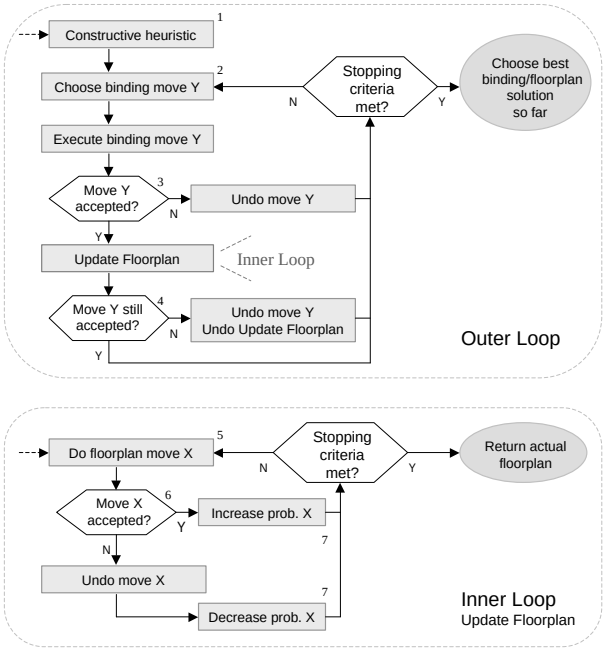

Figure 8: Flowchart of the advanced algorithm

4.4.2. *Floorplan driven binding*

The probability of choosing a binding move decreases or increases depending on the following factors (Fig. 8 (2)):

1. Through a binding move multiplexers can vanish or appear. An increasing number of multiplexers decreases the probability of choosing this move.
2. Sharing resources with a physical locality increases the probability.
3. Sharing resources that conduct data exchange increases the probability.
4. Each operation of a resource is assumed to be mapped exclusive on one resource. Then for this operation the balance point is determined. In this context the balance point is the position, where its exclusive resource would produce the shortest wire length. A high deviation between the balance point of a operation and its original resource increases the probability to split the operation.

4.4.3. *Avoiding unnecessary floorplan updates*

After each binding move and before executing the inner loop the effect of the move is evaluated (Fig. 8 (3)). Appearing or vanishing resources are inserted or deleted as supplied before. But before entering the inner loop a power pre-estimation is performed. Because of the sub-optimal floorplan the interconnect power is overestimated. Nevertheless the result is a good indicator for the impact of the binding move. If the power increases and the difference exceeds a threshold limit, the binding move is rejected. In this manner most unnecessary time consuming floorplan updates could be avoided.

4.4.4. *Floorplan update with zero temperature*

The available floorplan moves F_1 - F_5 are very different in their impact on the floorplan. E.g. the move F_2 (Swap Nodes) changes the complete floorplan. The other way round F_4 (Shift Leaf or Node) leeds to minor changes. F_2 is a effective move for a nearly un-optimised floorplan whereas F_4 produces good results for a nearly optimised floorplan. The availability of different powerful moves reduces the need of ”hill climbing” in SA. Hence the SA temperature T of the inner loop is set nearly to zero. That means that almost no deteriorate move is accepted (Fig. 8 (6)). In addition, the probability of choosing a move from the inner loop is decreased or increased depending on the moves acceptance rate (Fig. 8 (7)). This acceptance rate is pre-initialised through the constructive heuristic (see 4.4.1).

5. EXPERIMENTAL RESULTS

Our proposed technique is implemented on top of the low power high-level optimisation tool ORINOCO [13]. We use the wire capacitance from [16] and an industrial 0.25 μm RTL design library.

We evaluate eleven algorithmic level benchmarks. An *fdct* (fast discrete cosinus transfomation), an *fft* (fast fourier transform), a one-dimensional *wavelet* transform, two convolution filters *fir* (one-dimensional) and an *img_filter* (two-dimensional), *jpeg* (image compression codec from the independend JPEG group), *diffeq* (differentail equation solver), *matrix* (four-by-four matrix multiplication intended for 3D graphics), *overlapp_add* (windowing function used by ffts), *viterbi* and *turbo_decoder* (fault tolerant codecs).

Three different experiments are performed (cf. Table 1):

1. Full parallel (*FP*)
   Each operation is mapped on one single resource (no sharing of functional units). Only floorplan optimisation is ex-

		FP [nWs/mm^2]	IUO [nWs/mm^2]	SIO [nWs/mm^2]	IUO vs FP reduction [%]	SIO vs FP reduction [%]	SIO vs IUO reduction [%]
fdct	Data path	368.46	241.75	243.67	34.39	33.87	-0.79
	Interconnect	73.65	49.22	31.68	33.17	56.98	35.63
	Total	442.11	290.97	275.35	34.19	37.72	5.37
	Area	5.842	3.067	2.895	47.51	50.44	5.58
wavelet	Data path	464.70	448.98	571.29	3.38	-22.94	-27.24
	Interconnect	1514.47	1825.73	1224.44	-20.55	19.15	32.93
	Total	1979.17	2274.71	1795.73	-14.93	9.27	21.06
	Area	10.760	9.256	9.154	13.98	14.93	1.10
fir	Data path	674.14	355.40	355.64	47.28	47.25	-0.07
	Interconnect	520.16	466.77	290.39	10.26	44.17	37.79
	Total	1194.30	822.17	646.04	31.16	45.91	21.42
	Area	5.832	6.458	5.853	-10.73	-0.36	9.36
fft	Data path	486.53	480.42	480.57	1.26	1.23	-0.03
	Interconnect	149.49	147.99	89.96	1.00	39.82	39.21
	Total	636.02	628.41	570.53	1.20	10.30	9.21
	Area	9.184	6.695	5.434	27.11	40.84	18.83
jpeg	Data path	1154.44	1080.03	1185.92	6.45	-2.73	-9.80
	Interconnect	3943.79	3832.88	1783.88	2.81	54.77	53.46
	Total	5098.22	4912.90	2969.80	3.64	41.75	39.55
	Area	7.527	4.295	3.045	42.94	59.55	29.10
viterbi	Data path	156.20	155.30	166.70	0.58	-6.72	-7.34
	Interconnect	627.42	550.93	280.61	12.19	55.28	49.07
	Total	783.62	706.23	447.31	9.88	42.92	36.66
	Area	4.493	3.653	3.458	18.70	23.04	5.34
diffeq	Data path	250.42	199.65	199.68	20.27	20.26	-0.02
	Interconnect	219.17	154.58	109.68	29.47	49.96	29.05
	Total	469.59	354.23	309.35	24.57	34.12	12.67
	Area	3.237	3.062	3.291	5.42	-1.67	-7.50
matrix	Data path	937.78	670.12	739.63	28.54	21.13	-10.37
	Interconnect	1875.14	1889.72	935.72	-0.78	50.10	50.48
	Total	2812.92	2559.83	1675.35	9.00	40.44	34.55
	Area	10.757	4.504	3.707	58.13	65.54	17.69
img_ filter	Data path	1653.13	697.21	766.03	57.82	53.66	-9.87
	Interconnect	4664.97	3932.56	2631.19	15.70	43.60	33.09
	Total	6318.11	4629.77	3397.22	26.72	46.23	26.62
	Area	4.94	6.861	5.608	-38.89	-13.52	18.26
overlap_ add	Data path	658.00	655.25	655.67	0.42	0.35	-0.06
	Interconnect	575.91	643.48	339.81	-11.73	41.00	47.19
	Total	1233.91	1298.73	995.48	-5.25	19.32	23.35
	Area	4.918	7.581	5.411	-54.16	-10.04	28.62
turbo_ decoder	Data path	255.93	208.73	248.70	18.44	2.82	-19.15
	Interconnect	1165.73	1057.08	573.37	9.32	50.81	45.76
	Total	1421.66	1265.81	822.07	10.96	42.18	35.06
	Area	4.575	2.989	2.098	34.67	54.15	29.83
average	Data path				19.89	13.47	-7.70
	Interconnect				7.35	45.97	41.24
	Total				11.92	33.65	24.14
	Area				13.15	25.72	14.20

Table 1: Experimental results of the different experiments *FP*, *IUO* and *SIO* and the percentage energy and area reductions

ecuted. A parallel architecture is typically close to the lowest switched capacitance architecture, due to high temporal correlations.

2. Interconnect unaware optimisation(*IUO*)
   Binding optimisation and floorplan optimisation are executed consecutively. This interconnect unaware optimisation is the traditionally procedure in low power high-level synthesis.
3. Simultaneously optimisation (*SIO*)
   Binding and floorplanning is optimised simultaneously.

To achieve comparable results the total number of moves executed in each experiment are identical. The number of moves is determined depending on the benchmark size. The experiments were performed on a 1.0 GHz Athlon based PC with 256 MB memory. The CPU times vary from 6 seconds for *diffeq* to 138 seconds for *turbo_decoder*.

Fig. 9 shows the percentage power consumption of IUO and SIO compared to FP. The power of FP circuits is defined as 100 %. The bars are divided into data path power (lower part) and interconnect power (upper part). Please note that the total power of some interconnect unaware optimised benchmarks increase by 100 % (e.g. *wavelet*), which means that for these benchmarks the traditional optimisation fails. In Table 1 the exact values of the experiments are listed together with the percentage of energy and area reduction. Since scheduling and thus the timing is fix for each benchmark, energy reduction and power reduction are equivalent. Thus, we will further refer to power as energy. Compared to the traditionally procedure (*IUO*) our proposed technique (*SIO*) reduces the interconnect power for all benchmarks by an average of 41.2 %, while reducing overall power by 24.1 % on an average. The functional unit power just increases sensible for interconnect dominated designs (average of 7.7 %). Compared to *IUO* the area is also reduced by an average of 14.2 %.

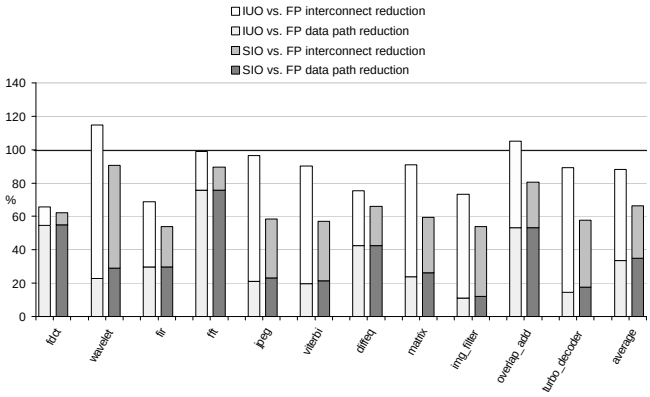

Figure 9: Total power and area reduction for different testcases

6. CONCLUSION

We showed that high-level synthesis has a significant impact on the interconnect power consumption. We proposed a new power optimisation algorithm which simultaneously performs floorplanning and functional unit binding. Experimental results demonstrate the benefit of incorporating interconnect in high-level synthesis for low power and the effectiveness of the proposed technique. Compared to interconnect unaware power optimised circuits, we have shown that interconnect power can be reduced by an average of 41.2 %, while reducing overall power by 24.1 % on an average. In fact, the energy consumption might even increase if the traditional optimisation flow is used. Our technique is implemented on top of the optimisation tool ORINOCO. Although our technique is general it can be easily incorporated into other high-level synthesis systems.

7. REFERENCES

[1] L. Scheffer, "A roadmap of CAD tool changes for sub-micron interconnect problems", in *International Symosium on Physical Design*, 1997.

[2] J.-P. Weng, and A. C. Parker, "3D scheduling: High-level synthesis with", in *Proc. of Design Automation Conference*, 1992.

[3] Y.-M. Fang, and D. F. Wong, "Simultaneous functional-unit binding and floorplanning", in *Proc. Int. Conf. Computer-Aided Design*, 1994.

[4] J. M. Chang, and M. Pedram, "Register allocation and binding for low power," in *Proc. of Design Automation Conference*, 1995.

[5] R. Mehra, L. M. Guerra, and J. M. Rabaey, "Low-power architectural synthesis and the impact of exploiting locality", in *J. VLSI Signal Processing*, 1996.

[6] A. Raghunathan, and N. K. Jha, "SCALP: An iterative-improvement-based low power data path synthesis system", in *Proc. Int. Conf. Computer-Aided Design*, 1997.

[7] L. Kruse, E. Schmidt, G. v. Cölln, A. Stammermann, A. Schulz, E. Macii, and W. Nebel, "Estimation of lower and upper bounds on the power consumption from scheduled data flow graphs", in *IEEE Trans. VLSI Systems*, 2001.

[8] K. Chao, and D. F. Wong "Floorplanning for Low Power Designs", in *IEEE Trans. VLSI Systems*, 1995.

[9] P. Prabhakaran, and P. Banerjee: "Simultaneous scheduling, binding and floorplanning in high-level synthesis", in *Proc. Int. Conf. VLSI Design*, 1998.

[10] L. Zhong, and N.K. Jha "Interconnect-aware High-level Synthesis for Low Power", in *ICCAD*, 2002.

[11] P. Christie, and D. Stroobandt "The Interpretation and Application of Rent's Rule", in *IEEE Trans. VLSI Systems*, 2000.

[12] G. Jochens, L. Kruse, E. Schmidt, and W. Nebel "A New Paramiterizable Power Macro-Model for Datapath Components", in *Proc. of Design, Automation and Test in Europe*, 1999.

[13] A. Stammermann, L. Kruse, W. Nebel, A. Pratsch, E. Schmidt, M. Schulte, and A. Schulz "System Level Optimization and Design Space Exploration for Low Power", in *14th International Symposium on System Synthesis*, Canada, 2001.

[14] W. Nebel, D. Helms, E. Schmidt, M. Schulte and A. Stammermann "Low Power Design for SoCs", in *Information Society in Broadband Europe*, Romania, 2002.

[15] D. F. Wong and C. L. Liu "A new algorithm for floorplan design", in *Proc. of Design Automation Conference*, 1986.

[16] Semiconductor Industry Association in *National Technology Roadmap for Semiconductors*, San Jose, CA: SIA, 1999.

[17] J. M. Rabaey "Digital Integrated Circuits", in *Prentice Hall*, 1996.