韓國産業應用學會論文誌 제2권 제2호 109-114, 1999.10
J. of Korea Society of Industrial Application, Vol.2, No.2, 109-114, October. 1999

시스템 분할과 합성을 이용한 신호처리기의 비용예측에 관한 사례연구
Case Study of a Cost Estimation for the Signal Processor through System Partitioning and Synthesis

김 종 태*
Jong-Tae Kim*

<요 약>

본 논문에서는 응용 주문형 집적회로 (ASICs)로 구현되는 신호처리기의 비용 예측 방법을 소개한다. 비용 예측은 디자인의 초기 단계에서 다양한 설계 사양들을 비교하여 성능과 비용 면에서 최적의 설계를 찾는데 도움을 준다. 본 비용 예측 방법은 Computer-Aided Design도구들을 이용하여 시스템 동작 표현으로부터 시작하여 시스템 분할과 상위 수준 합성을 거쳐 레지스터 전송 수준 단계에서 비용 예측을 실행한다. 사례 연구로 SWIR focal plane으로부터 생성되는 신호를 처리하는 신호처리기의 비용 예측을 실험한다. IBM 1.0 마이크론 기술의 CMOS 표준 셀을 적용하여 실험을 한 결과 각 채널로부터 전달되는 데이터를 실행하기 위해서는 3개의 칩이 필요했다.

Key Words : Size Estimation, High-level Synthesis

1. Introduction

In early stage of signal processing subsystem design it is required to accurately estimate signal processor cost to determine size and weight, impact on related subsystems such as the power supply, and to help evaluate the desirability and feasibility of various tradeoffs. The objective of processor cost estimation is to determine the hardware requirements of a processor configuration that implements the correct behavior, meets real-time constraints, and ideally has the minimum amount of hardware. Unfortunately, there is no known analytic solution to find the size of the optimal signal processor configuration. It is necessary to actually create a design, evaluate it, and determine if it meets requirements. Because the number of possible designs is very large, and in fact is exponential with respect to the many variables involved, a search for the optimal design is computationally intractable. In spite of this complexity, we are frequently called upon to estimate the size, weight, and

* 정회원, 성균관대학교 전기전자컴퓨터공학부, 부교수, 工博
경기도 수원시 장안구 천천동 300
email : jtkim@yurim.skku.ac.kr
Associate Professor, School of Electrical and Computer Engineering
본 연구는 성균관대학교 성균학술연구비의 지원에 의하여 수행되었습니다.

power of the final configuration given only the abstract behavior. Cost estimation techniques used at industry include extrapolation from similar programs and derivation of gate counts directly from behavioral specifications. These techniques have the advantage of simplicity and speed. Unfortunately they are grossly inaccurate since they cannot account for placement and routing effects. The approach we take is the use of top-down design method and Computer-Aided Design tools to automate estimation tasks. Our cost estimation task begins with behavior (e. g., multiply, shift), progresses to register transfer structure (e. g., adders, registers, multiplexors), and estimate chip size and performance without completing the register-transfer level(RTL) synthesis, logic synthesis, and layout. It is difficult to predict the low-level characteristics of a design from higher levels, since there are many different ways of performing the transformation at each step. To the best of knowledge, there is no reported work on the cost estimation which begins with behavior in algorithm level.

We use a CAD tool, LAST, for area and delay estimation. LAST[1] which uses models for standard cells and macros can predict area based on empirically determined formulas and then reconstructs the design, estimating the effects of routing and placement. Most of the reported area estimation models assumed that layout are done on gate arrays[2]. Grue et al.[3] looked into the problem of predicting the wire length distribution and estimating the average wire length of IC's. Standard cell area estimation[4] was proposed by Kurdahi and Parker. New area and delay estimation[5] from RTL description was reported. Also, Nemani[6] and Najm studied the area and power estimation from RTL description.

2. Processor Cost Estimation Method from a Behavioral Specification

2. 1 Inputs to the Processor Cost Estimation Task

The inputs needed for processor cost estimation may include: system behavior and constraints such as data rates, latency from input to output, area and power constraint.

The system behavior description specifies what manipulations must be performed on the data to accomplish the signal processing task; it identifies what must be done, but not how it is implemented in hardware or software. The system behavior can be specified in various formats, including mathematical equations, data flow graphs, software code, or the VHSIC Hardware Description Language (VHDL). The preferred level of description is RTL, or approximately that of software languages such as C or Fortran. Primitives at this level include arithmetic operations such as addition and multiplication, logic functions, and control operations such as loops and branches. Common higher level mathematical operations, such as trigonometric functions and differential equations, may be included, but more esoteric operations (e. g., centroiding) should be described using RTL primitives or other common mathematical operations.

Signal processing unit is a continuously operating system, and real-time response is required. Therefore the data rate defines the performance constraint on the implementation. Other implicit constraints on the implementation include the minimization of risk and production cost. The design goal is to find the cheapest implementation that can handle the data rates and has the required reliability and lifetime.

2.2 Size Estimation Method from a Behavioral Specification

Figure 1 shows the steps of our approach. The major steps are partitioning, RTL. If the entire design cannot fit on a single chip, the behavior must be partitioned into smaller regions, each of which will fit on a single chip. Architectural synthesis is then performed, producing RTL structure from the behavior in each partitioned region. From

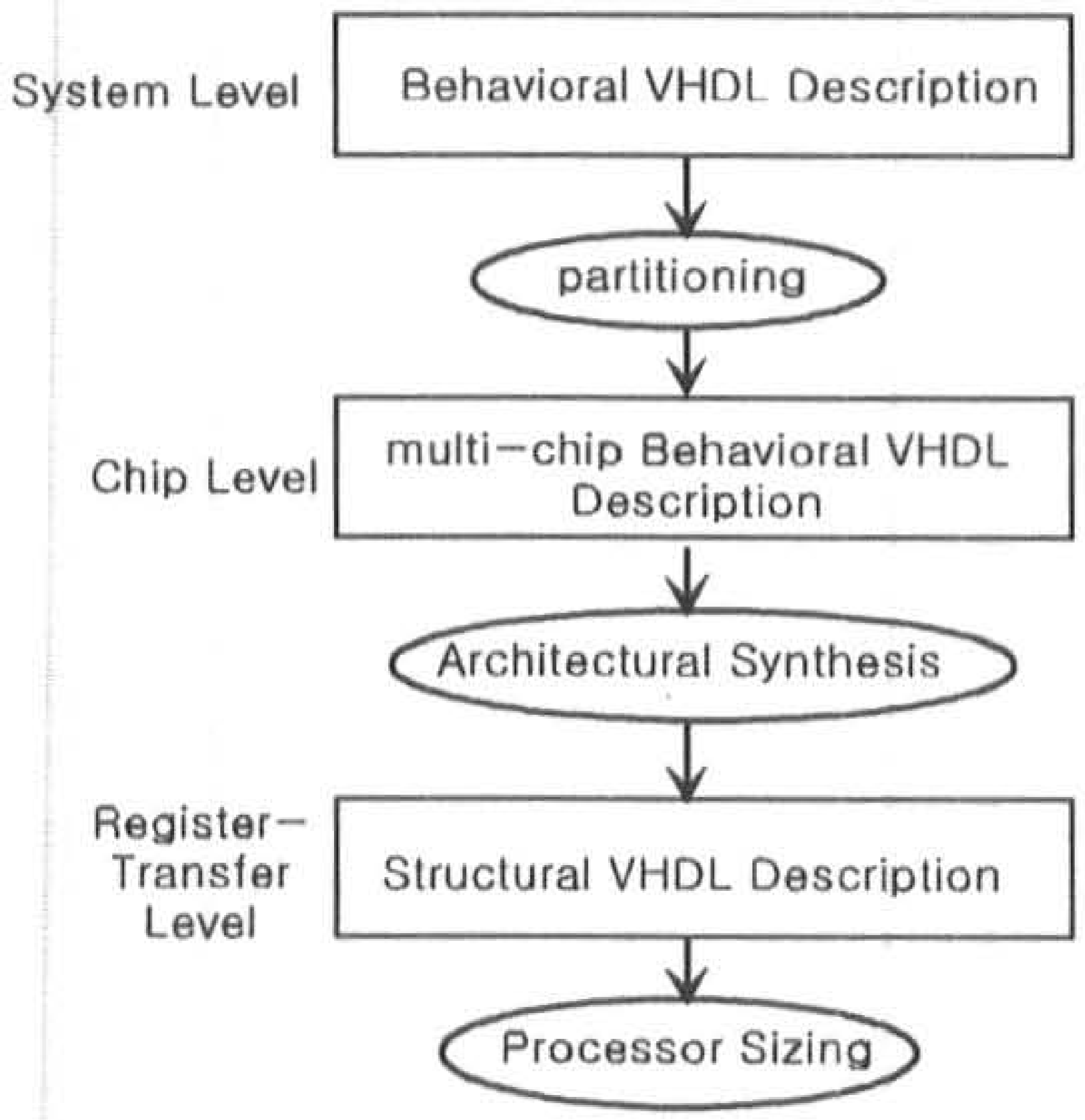

Figure 1 Estimation Steps

the RTL structure we estimate chip size and performance without completing the last steps of design, which are RTL synthesis, logic synthesis, and layout. Feedback on area and timing estimates can be used if necessary to improve the partition and synthesis results. For example, if a circuit is smaller than expected the design can be partitioned into fewer, larger regions to reduce the number of chip types required (or a smaller chip size can be used). If the areas of the partitioned regions turn out to be greater than expected, the design can be repartitioned into a greater number of smaller regions (alternatively a larger chip size can be used). If the performance estimate exceeds requirements we may be able to reduce the area by substituting smaller, slower RTL modules or to serialize the design and reduce the number of operators. The steps involved in cost estimation with the example of signal processing algorithm shown in Figure 1 are described in more detail in the following section.

3. Case Study

3.1 Overview of Signal Processing Subsystem

Signal processing subsystem for surveillance systems performs on-board target identification, classification, and impact point prediction. Because of high data rates and real-time constraints it requires the development of high performance signal and data processors[7]. Typical operations for signal processor are the suppression of gamma spikes, background clutter to increase the signal to noise ration, and identification of potential targets. It passes the processed data into data processing unit for target classification and tracking. Figure 2 shows a block diagram of signal processing operation example.

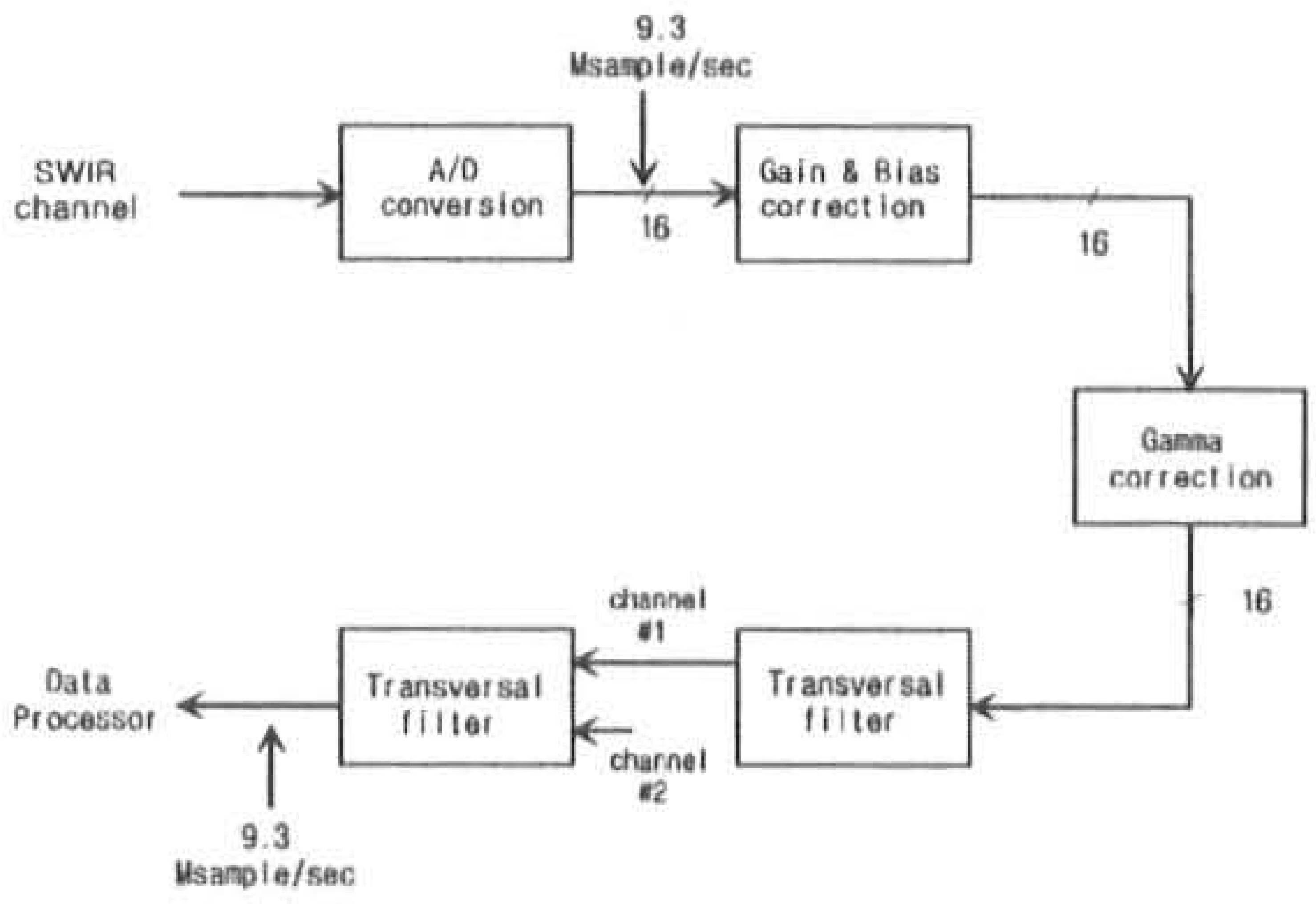

Figure 2 Signal Processing Subsystems Block Diagram

3.2 Processor Cost Estimation Experiment

We describe each block using VHDL to get the behavioral specification and we use 9.3 million-samples-per-second(MSPS) date rates. The design goal for this experiment is to find the cheapest implementation that can handle the data rates.

3.2.1 Partitioning

We performed partitioning on the signal processing algorithm shown in Fig 2. The algorithms are linear, involving few conditional branches and no internal loops. The partitioning was guided by the structure of the algorithms themselves (for example, all circuits for the in-scan transversal filter should ideally be placed on the same chip) and by estimates of how much circuitry will fit on a chip of the chosen size. For this exercise we placed all operations up to and including the in-scan filter into a single chip, P1, which accepts all data coming from a single A/D converter. The cross-scan transversal filter and all subsequent signal processing operations are placed on a second chip, P2. Three processing chips are required to process the data coming from each channel, two of type P1 and one of type P2.

3.2.2 Architectural Synthesis

Architectural synthesis involves several steps: clocking scheme synthesis, RTL module selection, operation scheduling, and module binding. To meet the 9.3 MSPS data rate we need to accept data once every 110 ns. The critical path of our algorithm (the longest path from input to output as defined by data dependencies) is too long to be executed in 110 ns using the fastest modules in the IBM library[8) (for example, the multiplies take 58 ns). For this reason we must pipeline the design, that is, break up the computations into stages separated by stage latches. New data can be input to the

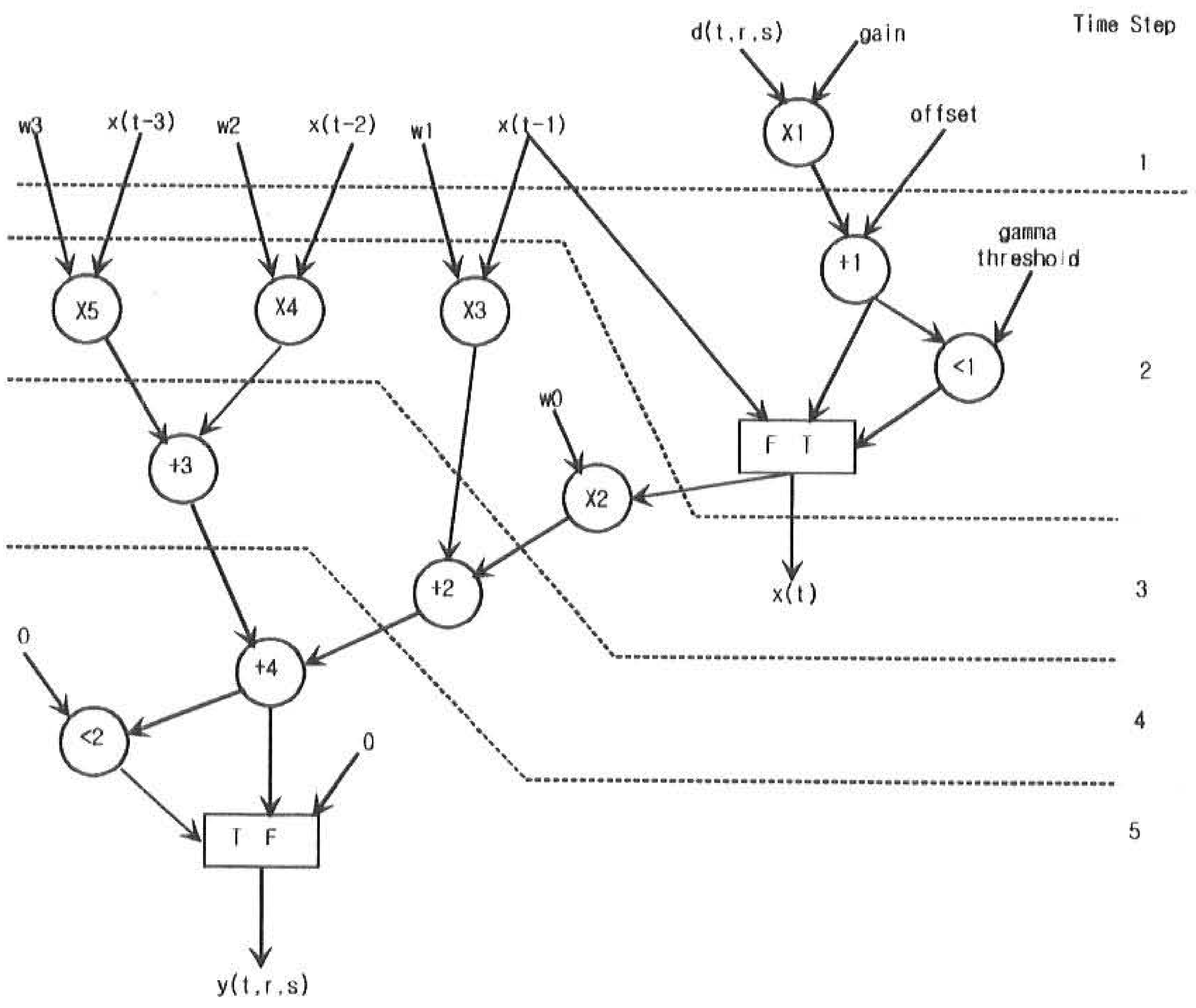

Figure 3 Schedule for the In-scan Filter

Table 1 Experiment Result

Chip type	P1	P2
Functionality	Gain correction, Gamma suppression, In-scan filter	Cross-scan filter, Auto-threshold
Estimated size	9.1 mm x 9.2 mm	9.0 mm x 8.0 mm
Estimated pins	100	145
Estimated power	443 mW	613 mW

first stage of the pipe while previous data is still being processed in later stages. Assuming that data is input once every clock period each stage of the pipeline should take no longer than 110 ns.

RTL module selection is the choice of a physical module type from the technology library for every operation type in the specification. For example, the IBM library contains both carry-select and carry-lookahead adders, either of which may be chosen to implement an addition. Assuming that every operation must complete in a single clock period, for each operation we chose the smallest module in the library whose execution time was less than the clock period.

The pipelined operation scheduling was performed using an high-level synthesis tool[9)10)]. It schedules operations into time slots according to data dependencies, clock period length, and estimated module delays; multiple operations may be cascaded within a single clock period. The schedule produced for one portion of the SWIR algorithms, the in-scan filter, is shown in Figure 3 as an example. There are five stages, so the latency of the pipeline is five times the clock period, or 550 ns.

Module binding is the assignment of physical modules to all scheduled operations. For some designs it is possible to share modules, that is, to assign a single physical module to operations that execute during different clock periods. For pipelined designs such as ours that have inputs every clock period, however, sharing is not possible, so a separate module is bound to each operation. In addition, stage latches are assigned to all data values that are produced and consumed in different clock periods and multiplexors are assigned where needed. The output of module binding is a structural netlist.

3.2.3 Prediction of Chip Area, Performance, and Power

During architectural synthesis we make use of area and delay estimates that must be provided for every module in the library. The area and delay of a chip cannot be calculated directly from module values, however, because of placement and routing effect. To estimate the area of the chip we therefore use a CAD tool, LAST, which predicts chip size and delay from a structural netlist. LAST partitions the design down to a level at which it can predict area based on empirically determined formulas and then reconstructs the design, estimating the effects of routing and placement. The size estimates produced by LAST are shown in Table 1. Circuits for both partitions are comfortably within the limits of the chosen chip size. Pin count is also at an acceptable level.

IBM provides a formula which was used to predict chip power for the 1.0 micron library, as follows;

$$Power = 1/2 \times C \times V^2 \times F \times SF$$

where C is the estimated load capacitance. V is the power supply voltage, F is the clock frequency, and SF is the switching factor.

The capacitance was assumed to be 0.5 pF for every two cells, with a worst-case power supply voltage of 5 volts, a clock frequency of 9.3 MHz, and a macro input switching factor of one-third. The resulting power figures for both chips are shown in Table 1.

4. Conclusion

In this paper we present the method of signal processor cost estimation for surveillance satellite system. The top-down design method and CAD tools used for estimation provide more accurate estimates than previous cost estimation techniques used in Aerospace industry. But our approach has certain limitations. We assume a bit-slice standard cell organization with a small number of macros that are not bit-sliced. Based on this study, we can go further to automate the design and estimation process for signal processing algorithms in early stage of time.

References

1) F. J. Kurdahi and C. Ramachandran, "LAST: a layout area and shape function estimator for high level applications," in Proc. 2nd European Conf. on Design Automation, pp. 351-355, 1991.

2) W. Heller et al., "Prediction of wiring space requirements for LSI," in Proc. of the 14th Design Automation Conf., pp. 20-22, IEEE/ACM, June 1977.

3) C. Gura and J. Abraham, "Average interconnection length and interconnection distribution based on Rent's rule," in Proc. of the 26th Design Automation Conf., pp. 574-577, IEEE/ACM, June 1989.

4) F. J. Kurdahi and A. C. Parker, "Techniques for area estimation of VLSI layouts," IEEE Trans. on CAD, vol. 8, pp. 81-92, January 1989.

5) A. Srinivasan, G. Huber, and D. LaPotin, "Accurate area and delay estimation from RTL descriptions," IEEE Trans. VLSI System, vol. 6. No. 1, pp. 168-172, March 1998.

6) M. Nemani and F. Najm, "High-level area and power estimation for VLSI circuits," IEEE Trans. on CAD, vol. 18, No. 6, June 1999.

7) Y. Bar-Shalom and T. E. Fortmann, Tracking and Data Association, Orlando, FL, Academic Press, 1988.

8) IBM 1.0 micron Radiation Harden CMOS Standard Cell Library, IBM, 1992.

9) N. Park and A. Parker, "Sehwa: a software package for synthesis of pipelines from behavioral specifications," IEEE Trans. on CAD, vol. 7, no. 3, pp. 356-370, 1988.

10) J. Kim, "Automated synthesis of time-stationary controllers for pipelined data path of application specific integrated circuits," 한국정보처리학회논문지, 제4권 제8호, pp. 2152-2162, 1997년 8월.

(1999년 7월8일 접수, 1999년 10월18일 채택)