Special Correspondence

Data Processing Section for Microprocessor-Like Integrated Circuits

ALTAMIRO AMADEU SUZIM

Abstract—**A methodology for the design of complex logic VLSI circuits is presented. The target application is described by an algorithm from which the structures of the control section and the data processing section of the integrated circuit are inferred. The architectural aspects are discussed and a model is proposed. A set of functional cells is then presented which implements the data section.**

I. Introduction

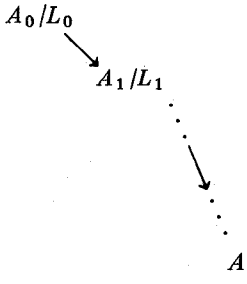
The design of VLSI integrated circuits is a problem that polarizes a large number of people. In fact, as technological advances allow a larger and larger number of elements on a single chip, the work needed to design these circuits grows at an even faster rate. Figures of 50 man years to design and 50 man years to debug commercial circuits have been mentioned [1]. New methodologies to design complex circuits (the highly repetitive ones excluded) must be searched in order to take advantage of the possibility of integration of 100 000 transistors and more [2].

The Capri project [3]–[6] investigates all the phases of the integrated circuits' development activity from the application description and algorithm transformation to architecture definition and layout, trying to automatize what is possible, and to standardize the intermediate phases. This work presents a methodology for the automatic realization of the operatives of microprocessor-like circuits. A microprocessor-like circuit is one that executes a somewhat complex algorithm over a data structure, the operative being the data handling unit.

Physically, the operative holds the data buses, memory elements, arithmetic and logic operators, switches, amplifiers, etc. Besides the microprocessors themselves, this class of circuits contains the I/O processors, arithmetic processors, complex peripheral controllers, etc.

II. Basic Principles

The very starting point of the process is the description of the selected application by the algorithm that it must execute. Let us call this algorithm A_0 and the language L_0. In a top-down design methodology, L_i will be interpreted by another language of lower level L_{i+1} and A_i will be transformed in an equivalent algorithm A_{i+1}, giving the chain



Manuscript received October 22, 1980; revised January 20, 1981.

The author is with the Computer Architecture Group, IMAG Laboratory, Grenoble, France, on leave from Universidade Federal, Rio Grande do Sul, Brazil.

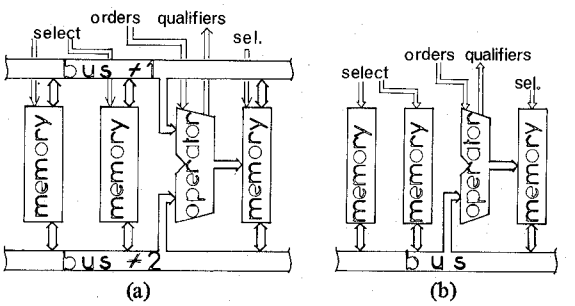


Fig. 1. Models of the operatives. (a) Dyadic operative. (b) Monadic operative.

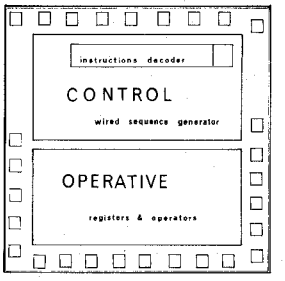


Fig. 2. Main blocks of the M6800 microprocessor.

L_n is the phase language of the physical machine [7]. Vertical arrows represent the transformation process that can be executed either automatically or manually. The resources of L_n (primitives and address space) correspond to the physical resources of the machine. The aim of this work is to realize the resources of level n for integrated machines. Two classes of operatives, called monadic operatives and dyadic operatives, are proposed. Fig. 1 sketches the structure of these operatives. In fact, they are more general allowing partitioning, commutation, and repetition of the elements to fit control signals topological constraints.

To drive the operatives, activating signals are generated by the control. Qualifiers are generated by the operatives to inform the control about the characteristics of some operations' result or data patterns detected on the buses. All this cooperates to execute the target algorithm; the flow of the algorithm corresponds to the state transition sequence of the control and the variables of the algorithm map into the address space of the operative(s), which also executes the data transformation and routing. To illustrate what is meant by control and operative in an integrated circuit the global structure of the M6800 is recalled. Fig. 2 shows the main blocks: instruction decoder and wired logic for the control and registers and operators for the operative.

The control contains the instruction register, instruction decoder, timing circuits, sequencing automata, etc., which all cooperate to generate the activating signals for the operative. The operative contains the registers (for the programmers' use and internal ones), the operators, the data buses, etc. The operative of this microprocessor is organized as 8 one-bit slices. These two blocks communicate via interface circuits, providing appropriate routing and buffering of activating and qualifiers signals.

The observation of several microprocessors shows a clear trend towards the use of regular structures. The desired

0018-9200/81/0600-0233$00.75

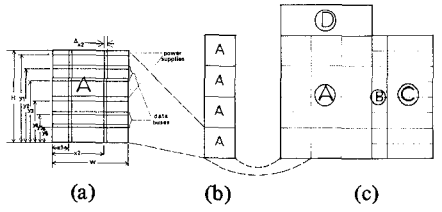


Fig. 3. General form of (a) a cell, (b) an element, and (c) an operative.

modularity based on specialized functional blocks does not have clear criteria nowadays. We can find a circuit with the instruction and status registers in the operative and another with its status register scattered all over the control.

III. Realization of the Operatives

The operatives are implemented on a cell based system. The cells are completely designed before use and are maintained in a cell library. The principal characteristics of these cells are (8):

The cells are completely defined in the cell library and do not change their shape or function at assembling time.

They are functionally defined, that is to say that for each function needed there must exist one cell.

For each family, the bus structure is standardized.

They constitute families and the cells of a family can be directly assembled together. All the cells of a family have the same height module. There may exist several versions of the same function in a given family. Also, one cell may be used in two different families if they differ only by characteristics that are meaningless for that cell (precharging of the bus for the shift cell, for example).

The cells are hand drawn to optimize surface utilization. An automatic layout system which accepts restrictions on the position of some elements will make the tasks of cell library extension and transformation easier.

Fig. 3(a) presents some characteristics of a cell (and consequently of the family to which it belongs). The cell has a height H and a width W. It is traversed in the horizontal direction by metal lines at ordinates $y1, y2, \cdots, y6$ for power supplies and data buses. There are two vertical polysilicon lines at abscissas $x1$ and $x2$ carrying commands and/or qualifiers. This cell (and more generally, all the cells) may be stacked to form an element as in Fig. 3(b). The elements are arranged side by side contiguously to form an operative [Fig. 3(c)]. Fig. 3(c) also shows a special cell "D" that goes over a set of elements. This category of cells, called hat cells because of their position, are intended to execute standard interface functions, like command amplification and are not discussed here.

The Cell Library

An initial set of cells for a given family provides the elements to implement an operative as previously defined. They are:

memory elements: registers, memory (RAM), and immediate values (ROM);
operators: logic unit, shifting unit, arithmetic unit, inc/dec unit;
bus oriented circuits: bus sensing/driving amplifiers, bus precharging, and baising circuits.

New cells are added to the family as they are needed by a particular application. A standard form is given to these new cells, whenever possible, and they enter the cells' catalog. That is the case, for example, of a content addressable memory in our application.

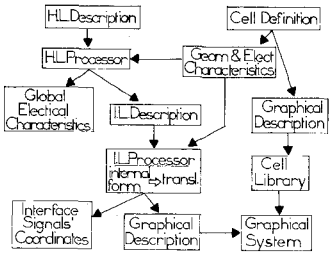


Fig. 4. System overview.

IV. System Overview

There are three levels of description in the system: the high-level description given by the user (just a list of the required elements), which is transformed by the high-level processor into an intermediate-level description (a position relative description). The intermediate-level processor then calculates the absolute coordinates of the cells and generates a textual description for the graphical system (Fig. 4).

The High Level Description

In a problem oriented strategy, a great number of conventions may be assumed. The user will ask for an operative of known structure: that's why a straightforward and simple description like a list of type and number of cells provides a sufficient information. The cells are known by name by the high-level processor. An example of a list would be

```
Operative user_name1 type dyadic width 8 bits
        8  ROMD (ϕ, FF, 66, . . .)
        4  RAMD
           ALUX
           REGX(A)
           AMPLID
           SWITCHD(A, B)

Operative user_name2 type monadic width 16 bits
           INC/DECS
           REGY
           AMPLIS
        5  RAMS
       10  ROMS (ϕ, FFFF, . . .)
           END
```

The two operatives communicate via the interface cell SWITCHD. The high-level processor also evaluates the global electrical characteristics of the operatives.

The Intermediate Level Description

The intermediate form is based on a simple language representing the relative position of the cells and may be described as follows:

⟨*fig*⟩ ::= ⟨*el*⟩ | ⟨*fig*⟩ *op* ⟨*el*⟩
⟨*el*⟩ ::= ⟨*var*⟩ *! parameter-list !*
⟨*var*⟩ ::= ⟨*cell-name*⟩ | *(*⟨*fig*⟩*)* | ⟨*rec*⟩
⟨*op*⟩ ::= ↑ | →
⟨*parameter-list*⟩ ::= *free format list of geometrical functions*
⟨*rec*⟩ ::= *empty rectangular cell of dimensions !dx, dy!*

The parameters have, at present, a very restrained use such as setting absolute coordinates.

↑ and → mean, respectively, up and right juxtaposition.

V. Conclusion

This correspondence presents an automatic design methodology for the data processing section of microprocessors and similar integrated circuits. Instead of trying to solve all the

problems, a problem oriented method that gives good results for the greatest number of cases was preferred. The proposed architecture is sufficiently general and appropriate to be controlled by known regular structures such as PLA and microprogramming. These two control techniques may also directly implement different levels of interpretation and can be generated by automatic systems; more, these systems may take advantage of the possibility of changing the form of a PLA (5) or a ROM to fit the operative's structure.

The limitation of the system is the necessity of redesigning the cells, either manually or automatically, when the technology changes. The topology will be conserved, in general. The advantage is the possibility of obtaining, with little work, the set of registers and logic and arithmetic elements of a given size and word length with cells of proved correctness.

If a circuit is to be designed, why not do it with cells and get the cell library for future circuits?

Presently, we have a basic set of a HMOS1 cell's family; the kernel of the intermediate processor works properly and we are working on the high-level processor. The application is a communication controller and we would like to report the results in our next paper.

REFERENCES

[1] B. Lattin, "VLSI design methodology—The problem of the 80's for microprocessor design," in *Proc. 16th Design Automat. Conf.*, San Diego, CA, June 1979, pp. 548–549.

[2] W. Wiemann, "CAD system for VLSI," in *Proc. 16th Design Automat. Conf.*, San Diego, CA, June 1979, p. 550.

[3] A. A. Suzim, "Parties opératives à éléments modulaires," DEA, ENSIMAG, Grenoble, France, Tech. Rep., 1979.

[4] A. A. Suzim, "Modular data processing units," in *Proc. 6th Europ. Solid-State Circuits Conf.*, Grenoble, France, Sept. 1980, pp. 287–289.

[5] T. Perez Segovia, "Optimisation de PLA's," ENSIMAG, Grenoble, France, Tech. Rep., 216, 1980.

[6] F. Anceau and R. Etienne, "Project Capri," ENSIMAG, Grenoble, France, Int. Rep., 1979.

[7] J. P. Shoellkopf, "Machine PASC-HLL: Définition d'une architecture pipe-line pour une unité centrale adaptée au langage PASCAL," Docteur de 3° cycle thesis, Grenoble, France, 1977.

[8] D. Johannsen, "Bristle blocks: A silicon compiler," in *Proc. 16th Design Automat. Conf.*, San Diego, CA, June 1979, pp. 310–313.

[9] J. M. C. Alves Marques, "A multiprocessor architecture adapted to VLSI custom design," in *Proc. EUROMICRO Symp.*, London, England, Sept. 1980, pp. 321–327.

The Development of CCD Analog TV Line Stores

J. N. GOODING, P. S. PADDAN, AND V. A. BROWNE

Abstract—**This correspondence discusses the development of analog TV line stores with very low spatial noise and wide signal bandwidth. These devices are fabricated on a buried n-channel silicon gate process which exhibits low thermal leakage and low leakage spike density. A 768-stage and an 850-stage store have been designed. The problems which have been encountered during the development and their solutions are described.**

Manuscript received October 27, 1980; revised January 5, 1981. This work was supported in part by the Procurement Executive, D.C.V.D., Ministry of Defence, England.

The authors are with Plessey Research (Caswell) Limited, Allen Clark Research Centre, Caswell, Towcester, Northamptonshire, England.

I. INTRODUCTION

Charge coupled devices (CCD's) are attractive in signal processing applications which require the temporary storage of analog video information. This correspondence describes the development of such devices that have been designed primarily for use in a thermal imaging system, but are equally applicable to visible TV systems. The principal performance parameters for these devices are wide signal bandwidth and low spatial noise. Excessive spatial noise may cause fixed pattern noise while poor signal bandwidth would result in loss of picture resolution.

Under this development, a 768-stage device and an 850-stage device have been designed so far. The 850-stage device offers greater resolution and circuit improvements; these will be discussed in detail later. Extensive process development work has been carried out to reduce the thermal leakage currents and leakage spike density to a very low level. The results of this work are presented in Section II. The solutions to the primary and secondary problems which were encountered during the device development are then discussed.

II. SPATIAL NOISE—PROCESSING ASPECTS

The devices have been fabricated on a coplanar two-level polysilicon, buried n-channel process using a total of seven photomasks. (For full description, see [1].) Due to the dynamic storage nature of CCD's, leakage of unwanted charge into the device storage cells represents a major technical problem to be overcome in achieving a satisfactory level of signal to spatial noise ratio. Therefore, in order to obtain adequate performance levels, it is necessary to understand the nature of the potential causes of leakage currents in the NMOS process. These can be grouped in three categories, viz.

1) thermal generation of electron-hole pairs,
2) electron injection,
3) avalanche generation.

All three mechanisms can, and indeed do, represent limitations to the performance of CCD analog line stores. While the first is largely a matter of improved processing techniques, great care must be exercised in the design of these stores to ensure that the latter two mechanisms do not become the dominant causes of excessive spatial noise.

A. Thermal Generation

Thermal carrier generation in the bulk depletion region and via surface states at the oxide–silicon interface gives rise to a continuous leakage of charge into the device depletion wells. This current I_L is strongly influenced by the carrier lifetime τ_o in the depleted bulk material and by the surface recombination velocity s_o such that

$$I_L \approx \frac{qn_iA}{2}\left(\frac{x_d}{\tau_o} + s_o\right) \tag{1}$$

where

n_i = intrinsic carrier concentration
q = electronic charge
x_d = depletion region depth
A = electrode area.

It has been well established [2] that imperfections in the silicon lattice can give rise to anomalously low values of τ_o in localized regions owing to the segregation of impurities in these regions. The resultant areas of high leakage current (leakage "spikes") have catastrophic consequences for leakage current uniformity. So the following discussion assumes that such imperfections have been largely eliminated from the slice fabrica-

0018-9200/81/0600-0235$00.75