Int. J. Elect. Enging. Educ., Vol. 31, pp. 334–341. Manchester U.P., 1994. Printed in Great Britain

BEHAVIORAL SIMULATION OF AN ARITHMETIC UNIT USING THE SPREADSHEET

ILYA LEVIN
School of Education, Tel Aviv University, Israel

1 INTRODUCTION

The digital circuit's design, based on an initial description of its working principles, is usually called *implementation*. The computer memory-based implementation of the circuit's operation can be called *simulation*.

The kind of simulation of the circuit where both the circuit and its behavior are described on the input–output level, will here be called the *behavioral* simulation. In this case the formulation of the designing problem is simulated, but not the contents and the structure of the unit.

There is a difference between the behavioral and the *structural* simulation, in which the structure of the discrete unit, components of the unit and connections between the components are simulated. The structure simulation is the simulation of the solution of the designing problem.

Both these kinds of simulation can be used in all the stages of designing for instance, the stage of simulation of structure automata, of logical circuits, of electric networks, etc.

In the case of the spreadsheet base simulation, two mentioned kinds of simulation can be defined as follows. If at least one coordinate of the spreadsheet represents the discrete time we call this simulation the *behavioral simulation within the spreadsheet*. When both coordinates of the spreadsheet are the space coordinates, we call this simulation the *structural simulation within the spreadsheet*.

Use of structural simulations of logical networks within a spreadsheet is known in educational practice at present[1,2]. Popular opinion[3] is, that behavioral simulations are inappropriate in the field of education. But there are some advantages of the behavioral simulations. Firstly, they are simple, testable, debuggable and constructible. Secondly, the behavioral simulations give users an opportunity to investigate the behavior of the system in time.

This paper deals with the use of the spreadsheet as a means for behavioral simulation of digital circuits. We shall say that the spreadsheet *simulates* the work of the circuit during the time interval $[0, T]$, if it displays all $T+1$ sequential state alternations of this circuit (this spreadsheet includes $T+1$ columns). The fragment of a spreadsheet which simulates the functioning of a circuit is the *spreadsheet-model* of the circuit.

Every cell (i, j), placed on intersection of the i-column and j-row of the spreadsheet-model M is programmed for the implementation of the particular function F_{ij}^{M}, oriented on this model. We shall call this function the *indicative function* of the spreadsheet-model M. Representation of the circuit in the form of the spreadsheet-model will be called the *simulation of the circuit*. We assume here, that horizontal coordinates of the spreadsheet represents the discrete time.

The aim of this paper is to illustrate the opportunities offered by behavioral simulation (for instance, by Arithmetic Unit (AU) simulation) using the spreadsheet. We shall illustrate the simulation via the example of simulation of *the binary multiplication circuit*. The spreadsheet EXCEL for the Macintosh computer is applied for this purpose.

2 SIMULATION OF THE OPERATION AUTOMATON

The AU consists of two parts: *an operation automaton* and *a control automaton* in accordance with the classical scheme (Fig. 1). The operation automaton in our example (Fig. 2) consists of two shift registers, an accumulator and a

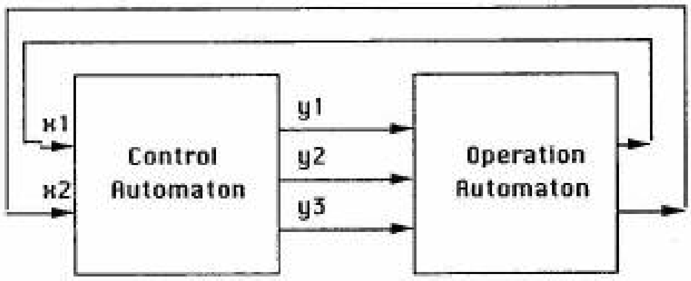

FIG. 1 The general structure of the AU.

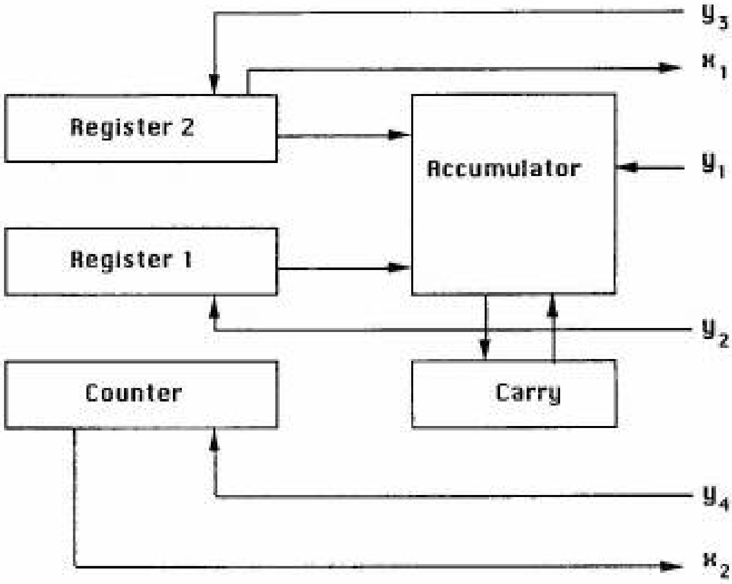

FIG. 2 Structure of the operation automaton.

counter. As we can see on the diagram the following input control signals appear in the scheme:

y_1 — initialization of the addition in the accumulator;
y_2 — shift initialization of register 1;
y_3 — shift initialization of register 2;
y_4 — decrement of the counter.

Two ouput (information) signals emanate from the operation automaton:

x_1 — younger bit of the factor equal 1;
x_2 — the content of the counter equal 0.

The algorithm of multiplication is implemented within the control automaton. The simulation of the control automaton is represented later.

Simulation of the shift register
The τ-column of the spreadsheet-model represents the contents of the register at the moment τ in time.

The indicative function of this behavioral model represents the dependence of the current bit of the register on the previous state of the register. This function value is equal to the highest bit (when left shift is used) or to the least-significant bit (when right shift is used). Formally this is represented as follows:

$$F_{ij}^{ls} = F_{i-1,j-1}^{ls} \quad \text{for a left shift;} \tag{1}$$

$$F_{ij}^{rs} = F_{i-1,j+1}^{rs} \quad \text{for a right shift.} \tag{2}$$

For implementation of these functions all cells of the spreadsheet-model have to be filled by spreasheet functions $=R[-1]C[-1]$ and $=R[1]C[-1]$. The spreadsheet-model shows a sequence of changes in the contents of the register. The spreadsheet-model of registers in our example is the fragment of the spreadsheet *R4C3:R10C12* (register Rg1) and *R13C3:R15C12* (register Rg2), displayed in Table 1. The spreadsheet-models of Rg1 and Rg2 are programmed for the implementation of the function (1). Namely the following spreadsheet function is written for our example in cells of these fragments: $=IF(R50C[-1]=1, R[1]C[-1], RC[-1])$.

Simulation of the subtracting counter
Let us analyze a certain bit B of the counter and express its contents as a function of the contents of the counter at the previous moment of time. The content of the bit B is defined by the values of the less-significant bits of the counter at the previous moment of time. Namely, if all less-significant bits' values are 0, then the value of the bit B will change to its inverse (1 to 0 or 0 to 1). If at least 1 less significant bit is equal 1, then the value of our bit B will not change. This simple logic of a subtracting counter can be expressed by the indicative function:

$$F_{ij}^{sc} = \begin{cases} F_{i-1,j}^{sc}, & \text{if} \quad SUM\,(F_{i-1,1}^{sc}, \ldots, F_{i-1,j-1}^{sc}) \neq 0; \\ (F_{i-1,j}^{sc}+1)_{\text{mod}2}, & \text{if} \quad SUM\,(F_{i-1,1}^{sc}, \ldots, F_{i-1,j-1}^{sc}) = 0. \end{cases} \tag{3}$$

TABLE 1 *Spreadsheet representation of the operation*

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1			OPERATION AUTOMATON															
2	t		1	2	3	4	5	6	7	8	9	10						
3			0	0	0	0	0	0	0	0	0	0	0	TRANSITION TABLE				
4			0	0	0	0	0	0	0	0	1	0	0	state	1	2	3	4
5	R		0	0	0	0	0	1	1	1	0	1	0	1	*3*	*3*	*2*	*2*
6	e		0	0	0	1	1	0	0	0	1	0	0	2	*3*	*3*	*3*	*3*
7	g		1	1	1	0	0	1	1	1	0	0	8	3	*3*	*1*	*2*	*1*
8			0	0	0	1	1	0	0	0	0	0	0	OUTPUT TABLE FOR y1				
9	1		1	1	1	0	0	0	0	0	0	0	2	y1	1	2	3	4
10			0	0	0	0	0	0	0	0	0	0	0	1	*0*	*0*	*1*	*1*
11			0	0	0	0	0	0	0	0	0	0	0	2	*0*	*0*	*0*	*0*
12	R												10	3	*0*	*0*	*1*	*0*
13	e		1	1	1	0	0	1	1	1	0	0	1	OUTPUT TABLE FOR y2				
14	g		0	0	0	1	1	0	0	0	0	0	0	y2	1	2	3	4
15	2		1	1	1	0	0	0	0	0	0	0	4	1	*1*	*1*	*0*	*0*
16													5	2	*1*	*1*	*1*	*1*
17	A		0	0	0	0	0	0	0	0	0	0	0	3	*1*	*0*	*0*	*0*
18	c		0	0	0	0	0	0	0	0	0	0	0	OUTPUT TABLE FOR y3				
19	c		0	0	0	0	0	0	0	1	1	1	32	y3	1	2	3	4
20	u		0	0	0	0	0	0	0	1	1	1	16	1	*1*	*1*	*0*	*0*
21	m		0	0	1	1	1	1	1	0	0	0	0	2	*1*	*1*	*1*	*1*
22	u		0	0	0	0	0	0	0	0	0	0	0	3	*1*	*0*	*0*	*0*
23	l		0	0	1	1	1	1	1	1	1	1	2	OUTPUT TABLE FOR y4				
24			0	0	0	0	0	0	0	0	0	0	0	y4	1	2	3	4
25													50	1	*1*	*1*	*0*	*0*
26			0	0	0	0	0	0	0	0	0	0		2	*1*	*1*	*1*	*1*
27	C		0	0	0	0	0	0	0	0	1	0		3	*1*	*0*	*0*	*0*
28	a		0	0	0	0	0	0	0	0	1	1						
29	r		0	0	0	0	0	0	0	0	1	0						
30	r		0	0	0	0	0	1	1	1	0	0						
31	y		0	0	0	0	0	0	0	0	0	0						
32			0	0	0	0	0	0	0	0	0	0						
33			0	0	0	0	0	0	0	0	0	0						
34			0	0	0	0	0	0	0	0	0	0						
35																		
36			0	0	0	0	0	0	0	0	0	0						
37	C		1	1	1	0	0	1	1	1	0	1						
38	o		1	1	1	1	1	0	0	0	0	1						
39	u		0	0	0	0	0	0	0	0	0	1						
40	n		0	0	0	0	0	0	0	0	0	1						
41	t		0	0	0	0	0	0	0	0	0	1						
42	e		0	0	0	0	0	0	0	0	0	1						
43	r		0	0	0	0	0	0	0	0	0	1						
44			CONTROL AUTOMATON															
45		x1	1	1	1	0	0	1	1	1	0	0						
46	C	x2	1	1	1	1	1	1	1	1	0	1						
47	o	z	4	4	4	2	2	4	4	4	1	2						
48	n	STATE	1	2	3	1	3	1	2	3	3	1						
49	t	y1	0	1	0	0	0	0	1	0	0	0						
50	r	y2	0	0	1	0	1	0	0	1	1	0						
51	o	y3	0	0	1	0	1	0	0	1	1	0						
52	l	y4	0	0	1	0	1	0	0	1	1	0						

The spreadsheet-model of the subtracting counter is shown in Table 1. The spreadsheet-model of the counter in our example is the fragment *R37C3:R43C12* of the spreadsheet. This spreadsheet-model is programmed for the implementation of the function (4).

For our example (Table 1) the following spreadsheet function is written in cells of this fragment:

$$= IF(AND(SUM(R36C[-1]:R[-1]C[-1]) = 0, R50C[-1] = 1),$$
$$MOD(RC[-1] + 1,2), RC[-1]).$$

Simulation of the accumulator

We have the spreadsheet-models simulating the work of two shift register Rg1. Let us develop a spreadsheet-model *A* simulating the work of the *accumulator*. This accumulator calculates the sum of two binary numbers which is being continuously saved in registers Rg1 and in the accumulator in previous moment of time.

The accumulator can be represented in the form of two tables: the spreadsheet *A* of a *sum* and the spreadsheet *C* of a *carry*. As usual, the contents of the accumulator and the carry can be expressed by the column of the spreadsheet. Indicative functions of these spreadsheet-models are:

$$F_{ij}^{A} = (F_{ij}^{R1} + F_{ij-1}^{A} + F_{ij}^{c})_{\text{mod}2}; \tag{4}$$

$$F_{ij}^{c} = \begin{cases} 1, & \text{if } F_{ij}^{R1} + F_{ij}^{R2} + F_{ij}^{c} > 1 \\ 0, & \text{if } F_{ij}^{R1} + F_{ij}^{R2} + F_{ij}^{c} \leq 1 \end{cases} \tag{5}$$

Both of these spreadsheets are displayed in Table 1. In our example the spreadsheet-model of the accumulator is fragment *R18C3:R24C12* of the spreadsheet, which is programmed for the implementation of function (4). The spreadsheet-model of the carry is the fragment *R27C3:R34C12* of the spreadsheet, which is programmed for the implementation of the function (5).

In our example (Table 1) the following spreadsheet functions are correspondingly written in cells of spreadsheet-models of accumulator and carry:

$$= IF(R49C[-1] = 1, MOD(RC[-1] + R[-14]C + R[10]C,2), RC[-1]),$$
$$= IF(R[-9]C[-1] + R[-23]C + R[1]C > 1,1,0).$$

3 SIMULATION OF CONTROL UNIT

All previously described units belong to operation units which are the units for transformation of information. However, a digital system (AU) cannot work without a control unit which defines the sequence of transformation of the information. This sequence is expressed in our example in the form of a flow-chart (Fig. 3). The control automaton can be obtained formally from the flow-chart[4].

The automaton is defined as vector $\{A, Z, W, \delta, \lambda, a_1\}$, where:

$A = \{a_1, \ldots, a_m, \ldots, a_M\}$ — state alphabet;

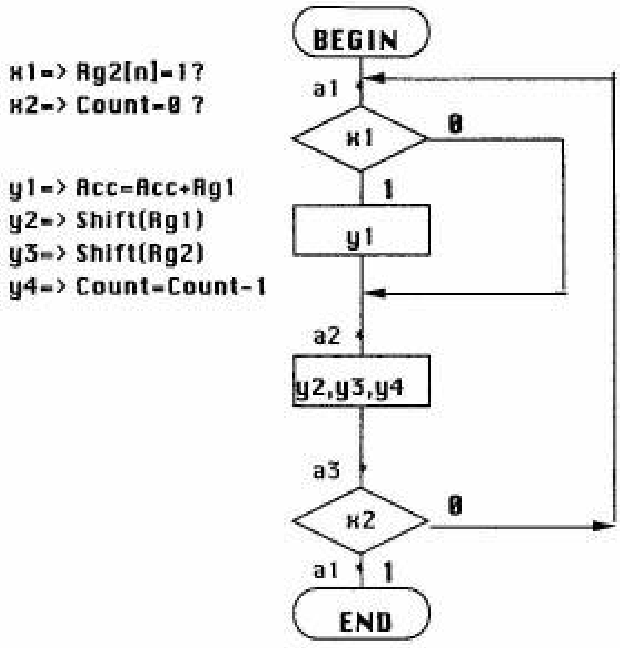

FIG. 3 The flow chart of the algorithm.

$Z = \{z_1, ..., z_f, ..., z_F\}$ — input alphabet;
$W = \{w_1, ..., w_n, ..., w_N\}$ — output alphabet;
$\delta \Rightarrow A^*X \rightarrow A$ — transition function;
$\lambda \Rightarrow A^*X \rightarrow Y$ — output function;
a_1 — initial state.

Usually automata are defined by *a transition table* and *an output table*. The present automaton, corresponding to the flow-chart of Fig. 3, is presented by the transition table (see Table 1), in the fragment *R4C15:R7C18* and the output tables in fragments *R9C15:R12C18*, *R14C15:R17C18*, *R19C15:R22C18*, *R24C15:R27C18*. In this automaton:

$A = \{a_1, a_2, a_3\}$;

$Z = \{z_1, z_2, z_3\}$; in our example every letter of alphabet Z is a vector of input binary variables x_1, x_2 of the control automaton as follows: $z_1 = \{x_1 = 0, x_2 = 0\} = 0$, $z_2 = \{x_1 = 0, x_2 = 1\} = 2$, $z_3 = \{x_1 = 1, x_2 = 0\} = 3$, $z_3 = \{x_1 = 1, x_2 = 1\} = 4$.

$W = \{w_1, w_2, w_3\}$; in our example every letter of alphabet W is a vector of output binary variables y_1, y_2, y_3, y_4 of the control automaton.

The spreadsheet-model of the automaton behavior brightly displays the successive changes of states, input and output signals of the automaton. It consists of three rows. The first of them represents an input signal, the second represents a state of the automaton, and the third represents an output signal. The current state and the output can be obtained using the following formulas:

$$a_i = \delta(a_{i-1}, z_i); \qquad w_i = \lambda(a_{i-1}, z_i).$$

The value of the δ-function on the transition from the state i to the state j can be found on the intersection of the i-line and the j-column of the transition table. The value of the λ-function on the transition from the state i to the state j can be found on the intersection of the i-line and j-column in the output table. Thus, indicative functions of the automaton's behavior spreadsheet-model can be defined using a spreadsheet function INDEX[5]. In our example these spreadsheet functions for state row (fragment *R48C3:R48C12*) are:

$$= INDEX(R5C15{:}R7C18,RC[-1],R[-1]C).$$

Rows of outputs y_1, y_2, y_3, y_4 (fragments *R49C3:R49C12, R50C3:R50C12, R51C3:R51C12, R52C3:R52C12*) will correspondingly be:

$$= INDEX(R10C15{:}R12C18,R[-1]C[-1],R[-2]C),$$

$$= INDEX(R15C15{:}R17C18,R[-2]C[-1],R[-3]C),$$

$$= INDEX(R20C15{:}R22C18,R[-3]C[-1],R[-4]C),$$

$$= INDEX(R25C15{:}R27C18,R[-4]C[-1],R[-5]C).$$

One can see that the spreadsheet-model which was constructed simulates the work of the control unit of our AU.

The set of simulated units available (registers, accumulator, counter, control unit) is enough for simulation of the whole AU on the spreadsheet. All these units appear in a vertical column so, that all τ-columns of spreadsheet-models, indicating the state of the unit at the τ-moment of time, are on the same vertical.

To incorporate the operation of AU into our example, it is necessary to connect between it and the control automaton. We update the references to output signals into control cells of the operation automaton in order to achieve this purpose. Accordingly, we update into the input cells of the control automaton the output signals from the operation automaton. In our example, sequences of inputs x_1, x_2 are written in rows *R45, R46*. These sequences were received from the operation automaton. The cells *R49, R50, R51, R52* sequences of inputs y_1, y_2, y_3, y_4 are written. These sequences are transmitted from the control automaton to the operation automaton. Namely, the cells of the rows *R45* include the reference to the corresponding cells of row $R13 = R[-32]C$, the cells of the rows *R46* include the reference to the corresponding cells: $= IF(SUM(R[-9]C{:}R[-3]C) = 0,0,1)$.

Simulation should be started with:

(i) updating the values of factors into the registers,
(ii) updating the number of bits of the second factor into the counter, and
(iii) setting up the control automaton in the initial state.

After that, the spreadsheet will show the 'history' of data processing in all parts of our AU.

4 CONCLUSION

Simulation of the operation of the digital system (AU) as described above, enables students the possibility to study the behavior of a system without concentration on the structure of its elements. Each of the simulated units was implemented in a very simple, obvious way. Actually, the behavior simulation is the representation of the initial description of the digital system's operation in terms of a spreadsheet.

REFERENCES

[1] El-Hajj, A. and Kabalan, K. Y., 'A spreadsheet simulation of logic networks', *IEEE Trans. Educ.*, **34**, No. 1, pp. 43–46 (1991)

[2] Levin, I., 'Matrix model of logical simulator within spreadsheet', *International Journal of Electrical Engineering Education*, **30**, No. 3, pp. 216–223 (1993)

[3] Williams, R. D., 'A logic simulator to support design instruction', *IEEE Trans. Educ.*, **E-30**, pp. 244–247 (1987)

[4] Baranov, S., *Synthèse des atomates microprogammés. Moscou* (in French) (1979)

[5] Cobb, D., *EXCELin Business.*, Bellevue, WA: Microsoft Press (1985)

ABSTRACTS – ENGLISH, FRENCH, GERMAN, SPANISH

Behavioral simulation and arithmetic unit using the spreadsheet

This paper examines the use of spreadsheets for the construction of the behavioral simulation of Arithmetic Units. It is shown that the spreadsheet is appropriate both for teaching and learning of different types of Arithmetic Units. These units can be simulated very simply by students during the lesson.

Simulation de comportement et unité arithmétique utilisant un tableur

Cet article examine l'utilisation de tableurs pour la construction d'un simulateur du comportement d'unités arithmétiques. On montre que le tableur convient pour l'enseignement et l'étude de différents types d'unités arithmétiques. Ces unités peuvent être simulées très simplement par les étudiants pendant les leçons.

Verhaltenssimulierung und arithmetische Einheit mit Hilfe der Tabellenkalkulation

Diese Arbeit untersucht den Einsatz von Tabellenkalkulationen für die Konstruktion von Verhaltenssimulierung arithmetischer Einheiten. Es wird gezeigt, daß die Tabellenkalkulation sowohl für das Lehren als auch das Lernen verschiedener Arten arithmetischer Einheiten angemessen ist. Diese Einheiten können sehr einfach von Studenten während der Vorlesung simuliert werden.

Simulaciones de funcionamiento y unidades aritméticas empleando hojas de cálculo

Este artículo examina el empleo de hojas de cálculo para la construcción de simulaciones de funcionamiento de Unidades Aritméticas. Se muestra como la hoja de cálculo es adecuada tanto para la enseñanza como para el aprendizaje de diferentes tipos de Unidades Aritméticas. Estas unidades pueden ser simuladas muy simplemente por los estudiantes durante las lecciones.