Simulation of Mixed Bond Graphs and Block Diagrams on Personal Computers Using TUTSIM

by J. J. A. J. BEUKEBOOM, J. J. VAN DIXHOORN *and* J. W. MEERMAN

Twente University of Technology, Department of Electrical Engineering, P.O. Box 217, 7500 AE Enschede, The Netherlands

ABSTRACT: *The TUTSIM simulation program for continuous dynamic systems accepts (nonlinear) block diagrams, bond graphs or a free mix of both. The simulation is "hands on" interactive, providing a direct contact with the model. The implementation of the program on existing personal computers (Apple II, IBM PC) requires small memory size and has a high computational speed, due to its assembler source code. A slower FORTRAN CP/M version is available. It is shown how bond graphs can be used as an input language. An example using bond graphs as a modelling tool is presented.*

I. Introduction

An increasing need for simulation tools exists, because simulation is recognized to be a very powerful aid in the design of a product, in the analysis of physical and technical systems and in education. Until recently, it was not possible to have a small, low-cost but still powerful simulation unit. The generally accepted simulation packages like CSMP (ENPORT and CAMP for bond graphs) are written in FORTRAN and require rather big computer systems. Another disadvantage of many packages running on mainframes is that the user does not have an immediate contact with the model and its computation. Running a slightly different model, e.g. change of a parameter value, may take a long time. On the other hand, most existing simulation languages for small computers comparable with ACES cannot contain large models consisting of a few hundred blocks or require an excessive computation time. It is therefore clear that "hands-on" interactivity in simulation packages on stand alone mini- and microcomputers is an important property.

TUTSIM (Twente University of Technology SIMulator) is a simulation language that makes it possible to run relatively large models on a small computer. During the seventies the THTSIM program was developed for the PDP11 and LSI11 series of DEC (1). It was written in MACRO 11, while a version in ANSI FORTRAN IV is also available, in order to make it transportable to non DEC computers. With the rise of the micro- and personal computers, the performance/cost ratio in terms of computational speed and memory space is greatly increased. These small computers make it possible to obtain a complete simulation unit within the range of U.S. $400 to U.S. $3000. TUTSIM, which will be described here, is the personal computer version of THTSIM. While THTSIM is extended with FORTRAN callable parameter estimation (SIMPLEX) and frequency analysis (FFT) packages (**2**), these facilities are not yet included in the various TUTSIM versions, due to lack of memory space in the available PC's.

In this contribution two subjects are tied together. One is simulation on a low cost base with personal computers, the other is the concept of bond graph modelling. Bond graphs are a clear and systematic tool for performing the difficult step of setting up a model of dynamical physical systems (**3**). The bond graph gives the possibility to gain fast insight in the physical and computational properties of a system. From the bond graph model the system state equations, ordinary differential equations or transfer functions can be derived in an algorithmic way. It is much more convenient if the bond graph itself can be a direct input to the simulation program. TUTSIM has this facility, thus creating a possibility to compute the dynamic system behaviour directly from the bond graph or from the block diagram on a low cost personal computer.

II. TUTSIM on Personal Computers

TUTSIM is an interactive simulation language for continuous dynamical systems. Its input has to be given in block diagram form (function blocks) or in bond graphs, the latter being discussed in a following paragraph. The block diagram may for instance be derived from the differential equations describing the system. The user of TUTSIM has an immediate contact with the model and with its results, which can be displayed on a screen and plotted. The user can interrupt the simulation, change the structure or the parameters of the model, the timing or the plot ranges and restart immediately. After the experiment a hard copy of the listing and results is made and the model is stored on floppy disk or cassette. On returning to the machine, the stored model can be reexecuted within seconds. The calculated results can be shown either graphically on a monitor screen or plotter, or numerically in tables on display (screen) or printer, depending on the system configuration being used.

The main characteristics of TUTSIM are:

(i) fixed integration stepsize,
(ii) dynamic memory allocation,
(iii) over 50 algebraic, logic, dynamic and bond graph function blocks, non-linear functions, PID-controller, noise generator etc.,
(iv) each integrator can be chosen to be Euler or Adams Bashforth II, with or without output limitation,
(v) possibility of real time simulation for slow processes (time constant >0.1 s, although this depends on the model size),
(vi) calculation interrupt,
(vii) memory space about 4 kW for every 400 function blocks used in the model and
(viii) floating point arithmetic (no scaling required).

The commands to control the simulation (about 30 in total) can be divided in three more or less separated classes:

(1) Change commands, to alter the simulation data.

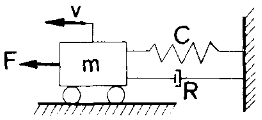


FIG. 1. Ideal physical model.

(2) Simulation commands, to start or to proceed the calculations.
(3) Utility commands, to list data, save the model on floppy disc, etc.

TUTSIM can be seen as a digital equivalent of an analog computer, with many extra facilities, e.g. all the nonlinear and logical functions. Figure 2 shows a model in block diagram form of the physical configuration shown in Fig. 1, a second order mass-spring-damper system. The structure of the model is given in the structure table (Fig. 3). This is equivalent with the form in which TUTSIM accepts the model.

We will discuss the 6502 assembler version for the Apple II, firstly because it has a good performance and secondly because this computer is widely spread and available [which is probably the main reason why so many simulation languages for microcomputers can only run on this machine (5)]. A minimum set to run TUTSIM on the Apple II effectively, costs about U.S. $800. Also some specific properties of other versions (8088 assembler for the IBM PC and CP/M FORTRAN) will be dealt with. Finally, a comparison between the versions is presented.

II.1. *The 6502 assembler version*

To make a program as fast and compact as possible, it should be written in assembler language. Starting from the PDP11 MACRO 11 assembler version, this has been converted into 6502 assembler source code, the 6502 being the microprocessor of the well known Apple II personal computer. To run TUTSIM on other 6502 based 8 bit personal computers like the Commodore 64, several alterations have to be made to cope with the specific I/O requirements. A minimum amount of 32 kbyte RAM, consisting of 16 kbyte for the program, 6 kbyte for the storage of the model and 10 kbyte for the graphical facilities and variable storage, is sufficient for the 6502 assembler version. A cassette tape may be used to load the program into the memory of the Apple II, but when floppy discs are used, an operating system like DOS has to be installed, which makes the total amount of required memory 48 kbyte RAM.

Some other necessary equipment to run TUTSIM effectively is a TV monitor to show the conversations and the simulation results (provided the PC has "in-

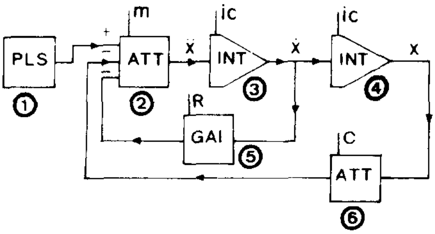


FIG. 2. Block diagram of model.

```
output  ,  block  ,  inputs
  1     ,   PLS
  2     ,   ATT   ,  1,-6,-5
  3     ,   INT   ,  2
  4     ,   INT   ,  3
  5     ,   GAI   ,  3
  6     ,   ATT   ,  4
```

FIG. 3. Structure table and TUTSIM input.

computer" graphics), and a printer. The standard software of the Apple II is capable of handling the C.Itoh (8500 and 8510) or the Epson printers (MX80 with graphic extension and MX82). These printers can also give a reasonable graphical output of the desired data. The TUTSIM version for the Apple II contains a floating point package, exchangeable with a hardware floating point arithmetic processor board. When installed with an AM9511 based floating point hardware board or an Accelerator 11 board (the latter gives all other applications of the Apple II and also a very fast calculating unit), the calculation speed is even better than the LSI11 minicomputer equipped with a KEV11 hardware floating point instruction set. In Fig. 4 a possible Apple II configuration is displayed.

TUTSIM for Apple II will run without adaptation on almost every Apple II equivalent, e.g. the Franklin Ace and the Microprofessor III.

II.2. *The 8086 or 8088 assembler version*

Because another very popular and wide spread personal computer exists, the IBM PC based on the 8086 equivalent 8088 microprocessor, an 8086/8 assembler version has been realized. Whereas TUTSIM on the Apple II and some other versions allows models of "only" 600 to 1000 blocks, depending on the available memory space, the IBM PC can handle models of up to 10,000 blocks. It is possible to have a multi-colour output, which makes the interpretation of TUTSIM results easier. The relatively large memory space also gives the possibility to make the program more "user-friendly", or even to include the THTSIM minicomputer extensions (FFT and SIMPLEX).

A remarkable disadvantage of the IBM PC is its computational speed, as will be shown in the next section. The computing speed however can be improved by

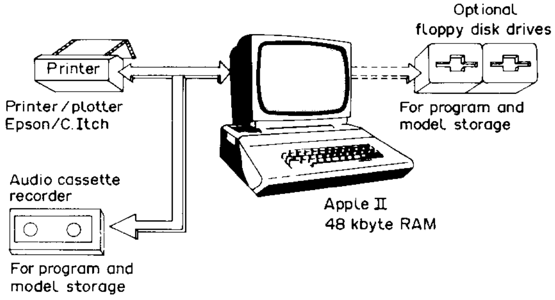


FIG. 4. Apple II simulation configuration.

plugging in the 8087 hardware floating point co-processor, although this is a rather expensive extension.

There are a number of IBM PC equivalents, that will run TUTSIM with minor or no adaptation. The low cost Sanyo MBC550 runs the 8086/8 assembler version of TUTSIM very well.

II.3. *The CP/M version*

A more transportable version is realized in FORTRAN 80 under the CP/M (Control Program for Microcomputers) operating system, while CP/M is used on a wide variety of PC's and a CP/M FORTRAN 80 compiler is readily available. Two disadvantages of the FORTRAN version are its relatively slow computational speed and relatively large required memory size (48 kbyte). Since personal computers have standard 64 kbyte of memory, this is generally no problem, as CP/M and TUTSIM together need 54 kbyte. (For the Exidy with only 48 kbyte a stripped version of TUTSIM has been realized.)

It has to be mentioned that in general there is no graphics output with a CP/M computer. By means of a standard RS232 serial I/O port connected to an external graphical device this can be solved.

II.4. *Computational speed comparison*

Since in a simulation program many multiplications (divisions) and additions (subtractions) have to be performed, calculation time is of crucial importance. Especially large models can be very time consuming when computation is slow and this has to be avoided. Therefore it is necessary to make a comparison between all the different versions of TUTSIM on mini- and microcomputers. The results are shown below in Table I.

The test has been performed with 2000 time steps on the second order system of

TABLE I
Computational speed TUTSIM

Computer	Language	Time (s)	Remarks
PDP11	MACRO 11	5.7...30	
LSI11	MACRO 11	16.5	FIS
LSI11/23	MACRO 11	9.5	FPU
LSI11	FORTRAN	42	
CP/M	FORTRAN	58	osborne
CP/M	FORTRAN	51...80	bigboards
IBM PC	assembler	59	
IBM PC	assembler	18	8087
Apple II	assembler	35	
Apple II	assembler	15	AM9511
Apple II	assembler	11	Accel.11
Apple II	BASIC	1540	ACES
Ohio Sup.	BASIC	1680	TUTSIM

Fig. 1, without any intermediate plotting or printing, in order to measure the real computing time. The time range of the PDP11 and the bigboards corresponds with different types that have been tested. The result clearly shows the fast comptuational speed of TUTSIM on the Apple II with hardware calculation unit compared with the others, including the IBM PC with a 8087 hardware co-processor and the FORTRAN versions. It can easily withstand the comparison with the PDP11 and LSI11 minicomputers. The last two rows in Table I show a comparison between the BASIC written simulation package for microcomputers ACES (All-purpose Continuous Equation Simulator) and the TUTSIM BASIC version on the Ohio Superboard computer, with comparable slow results. It might be concluded that BASIC on a interpreter base is not an acceptable simulation language. When time is of great importance, assembler source code should be used, while in case of transportation requirements, a FORTRAN version might be more convenient.

III. From Bond Graphs to Computer Input

Only the most important features and characteristics of bond graphs regarding TUTSIM will be treated. For more extensive coverage of bond graphs and their computability, the reader is referred to (3) and (4).

Once one is acquainted with the "bondgraph way of thinking", setting up a model of a physical system can be done more easily and in a more systematic way than it can be done with the block diagram description. Mechanical, electrical, hydraulic, acoustical, chemical or thermodynamical systems can be described as a physical network, with (nonlinear) transformations and coupling between the mentioned domains.

The system equations can be derived from the bond graph before actually performing the calculations of the dynamical behaviour (simulation) of the system. TUTSIM provides a means to present the bond graph directly as input to the simulation program. Also, other block-oriented simulation languages can be simply adapted to accept bond graphs as input freely mixed with function blocks (4). This extension to bond graphs greatly facilitates the use of TUTSIM and the insight of the user in physical or technical systems.

Figure 5 shows how a bond graph C-element in integral causality has been defined by a C-block in the TUTSIM program. This C-block is a composition of the three standard blocks shown; the same applies to the I-block. The other bond graph blocks in TUTSIM and their equivalent function blocks are listed in Table II. Most of them are standard blocks renamed with bond graph names. From the definitions follows that free mixing of bond graphs and function blocks is possible.

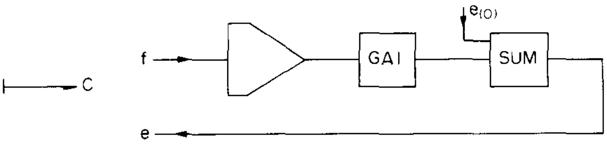
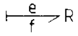
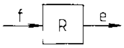
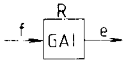
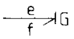
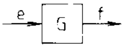
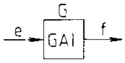
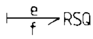
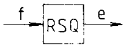
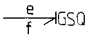
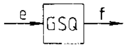
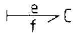
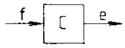
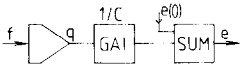
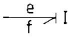
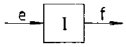
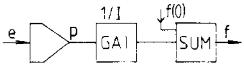
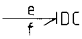
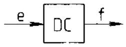
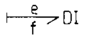
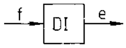
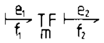
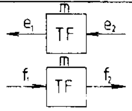
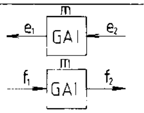
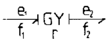
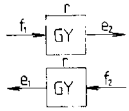
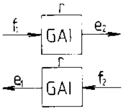
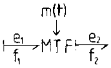
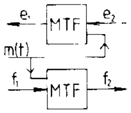
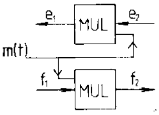
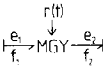
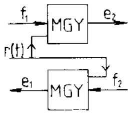
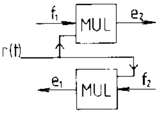
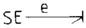
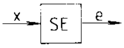
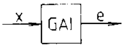
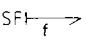
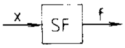
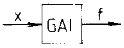
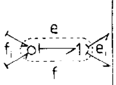
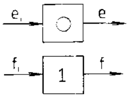
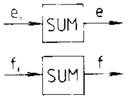


FIG. 5. C-element and associate function blocks.

TABLE II
Definition of bond graph blocks

b.g. element	b.g. block	Equivalent function block(s)
		
		
		$e = f \cdot \|f\| \cdot R$
		$f = \mathrm{sign}(e) \cdot \sqrt{\|e\|} \cdot G$
		
		
		$f_n = \frac{1}{3} f_{n-1} + \frac{2C}{3T}(e_n - e_{n-1})$
		$e_n = \frac{1}{3} e_{n-1} + \frac{2I}{3T}(f_n - f_{n-1})$
		
		
		
		
		
		
		

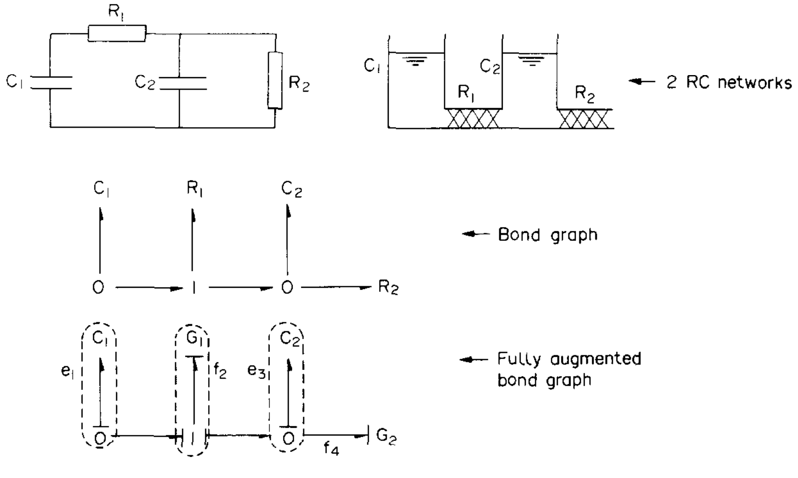
N.B. All inputs can be multiple (Σe or Σf)

A simple example of their use is shown in Fig. 6, in which two analogous RC networks are modelled by a bond graph. Causal strokes are added after which at each element the output variable is indicated by *e* or *f* and a number. The numbers have to be unique as the TUTSIM variables are numbers only. The number of variables can be strongly reduced by not assigning an output variable to each junction. Junctions, causally governed by an element like those in the example, do not need an individual output variable. They can be considered as a part of the element and encircled together. The input variables and their signs can be directly found by looking at the in- and outgoing bonds of the encircled complex. Junctions, causally governed by another junction, always occur in pairs ("causally independent junction pair"). In such cases both junctions require an individual output variable.

Nonlinear bond graph elements can be assembled from functional blocks: a nonlinear C is made from an integrator and a function table, a modulated R from a multiplier, a 2-port C from two integrators and two weighted sum blocks (linear case) and/or multipliers or function tables (nonlinear). Algebraic loops are dynamically iterated by the introduction of a unit delay element in the loop. They can also be removed by the addition of a small storage element, which changes causality; this method may lead to a stiff system however (4).

IV. Example: Force Control System

The example in Fig. 7 shows a control system that is represented by a mix of bond graphs and functional blocks, thus illustrating the ease of use of TUTSIM.



STRUCTURE TABLE

output	elem	inputs
e_1	C	$-f_2$
f_2	G	$e_1, -e_3$
e_3	C	$f_2, -f_4$
f_4	G	e_3

TUTSIM input

```
1,C,-2
2,G,1,-3
3,C,2,-4
4,G,3
```

FIG. 6. Bond graph example showing ideal physical systems, bond graph, fully augmented bond graph (with output variables) and structure table and TUTSIM input.

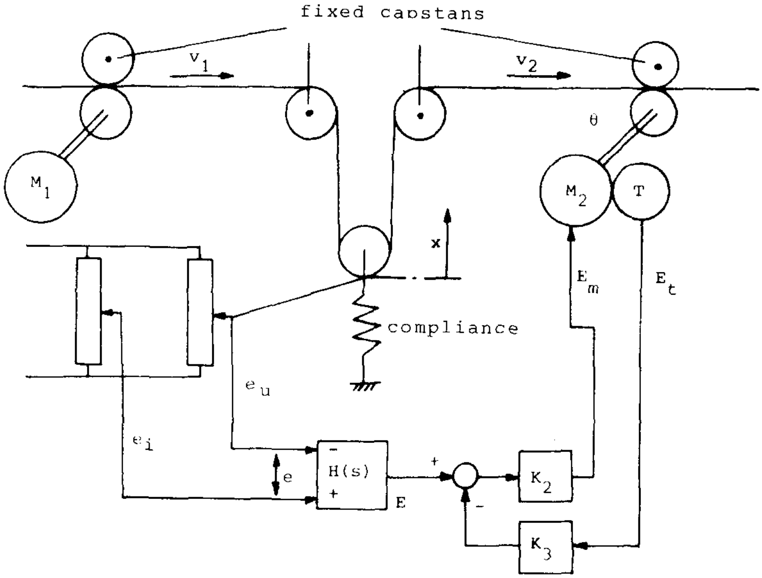


FIG. 7. Force control system.

Some flexible material (cloth, thread or the like) is transported by capstans and motors M_1 and M_2. M_2 is servo-controlled to stabilize the stress in the material, that is dictated by the spring with compliance C_v. The voltage e_u is proportional to x (and so with the stress in the material) as $e_u = C_px$. The voltage e_i is a measure for the desired stress, the difference e_i–e_u being the control error. The transfer function of the controller $H(s)$ is K_1/s, the resulting controller output voltage E drives the motor

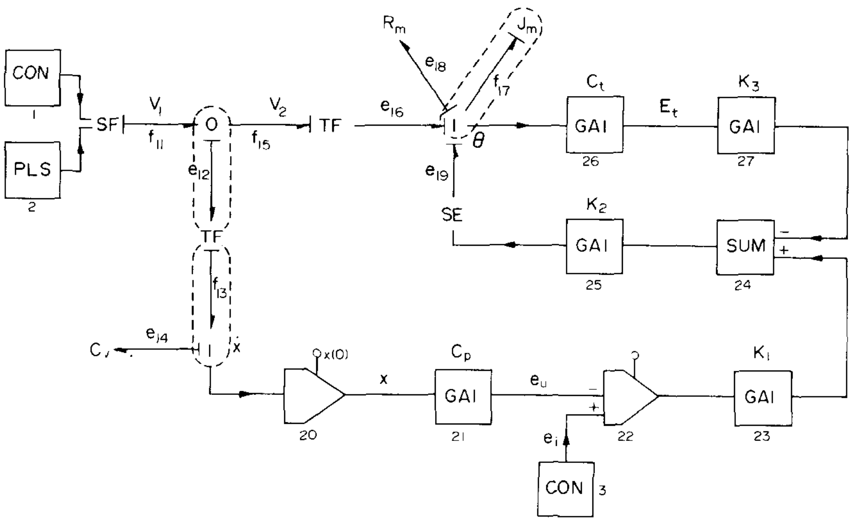
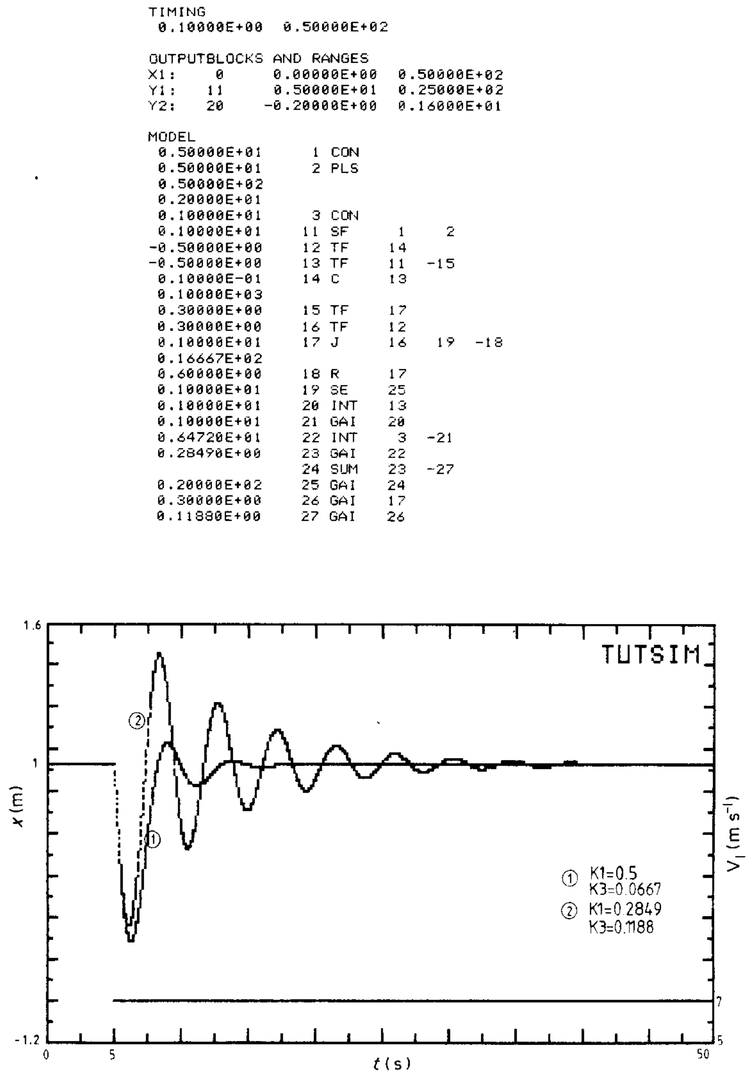


FIG. 8. Simulation scheme of the model.

M_2. This motor, with a linear characteristic, has a total on the motorshaft reduced inertance and friction of J_m resp R_m. The servo feedback is done with the tacho, $E_t = C_t\dot{\theta}$. The speed v_1 is an input, independent from material stress.

In Fig. 8 the combined bond graph–block diagram is shown. The coupled physical system (inertia, motors, spring and the translational to rotational transformations) is described in bond graph form, and the control loop with its associated signals in function blocks. This separation is very natural. The listing of the model, simulated with TUTSIM on the Apple II, is shown in Fig. 9. In the same figure the simulation results are shown. It gives an idea of the effect on x, after a



```
TIMING
 0.10000E+00  0.50000E+02

OUTPUTBLOCKS AND RANGES
X1:    0     0.00000E+00  0.50000E+02
Y1:   11     0.50000E+01  0.25000E+02
Y2:   20    -0.20000E+00  0.16000E+01

MODEL
 0.50000E+01      1 CON
 0.50000E+01      2 PLS
 0.50000E+02
 0.20000E+01
 0.10000E+01      3 CON
 0.10000E+01     11 SF       1     2
-0.50000E+00     12 TF      14
-0.50000E+00     13 TF      11   -15
 0.10000E-01     14 C       13
 0.10000E+03
 0.30000E+00     15 TF      17
 0.30000E+00     16 TF      12
 0.10000E+01     17 J       16    19   -18
 0.16667E+02
 0.60000E+00     18 R       17
 0.10000E+01     19 SE      25
 0.10000E+01     20 INT     13
 0.10000E+01     21 GAI     20
 0.64720E+01     22 INT      3   -21
 0.28490E+00     23 GAI     22
                 24 SUM     23   -27
 0.20000E+02     25 GAI     24
 0.30000E+00     26 GAI     17
 0.11880E+00     27 GAI     26
```

FIG. 9. Listing and plot-output on the Apple II with TUTSIM and C.Itoh printer.

sudden change in v_1, with two different sets of control parameters K_1 and K_3. This "plot" is realized with the C.Itoh printer, connected to the Apple II. Of course, a real plotter (connected to the Apple or a CP/M machine) would give a much higher resolution result.

V. Conclusion and Discussion

It is now possible to realise a fast and efficient simulation unit, capable of running large models, for a low price. The introduction of bond graphs in a block oriented simulation language like TUTSIM establishes the link between a powerful modelling concept for continuous dynamic systems and a low-cost simulation unit with good performance.

For many years, the majority of mechanical and electrical engineering students at Twente University of Technology have been trained in modelling and simulation using bond graphs as a modelling tool, and TUTSIM as a simulation language apart from CSMP. Also, the industry successfully uses TUTSIM. New versions for the 6502 based Commodore 64 and the very powerful MC-68000 (Apple Lisa and Mackintosh) are in the process of being written, the last processor giving the possibility to install the extensions of THTSIM (FFT and SIMPLEX) on a personal computer. This will create a remarkable versatile and powerful means for analysis and design. Present research is directed towards a general simulation package, which can handle multibond graphs for the simulation of 3-dimensional mechanical structures.

References

(1) R. A. Kraan, "THTSIM, a conversational simulation program on a small digital computer", *J. A*, pp. 186–190, 1974.

(2) J. W. Meerman, "THTSIM software for the simulation of continuous dynamic systems on small and very small computer systems", *Int. J. Modelling Simulation*, Vol. 1, No. 1, pp. 52–56, 1981.

(3) D. C. Karnopp and R. C. Rosenberg, "System Dynamics: A Unified Approach", Wiley, New York, 1975.

(4) J. J. van Dixhoorn, "Simulation of bond graphs on minicomputers", *Trans. ASME J. Dynamic Syst. Measure. Control*, Vol. 99, pp. 9–14, 1977.

(5) "Catalog of Simulation Software", SCS, Simulation, Vol. 41, No. 4, pp. 156–160, Oct. 1983.

(6) J. W. Meerman "Apple TUTSIM, Users Manual", Meerman Automation, P.O. Box 154, 7160 AC Neede, The Netherlands, 1983.