F.J.Rammig
Federal Republik of Germany

AN UNIVERSAL HARDWARE EDITOR BASED ON AN ALGEBRAIC MODEL OF THE TIME-BEHAVIOUR OF DIGITAL CIRCUITS

1. The basic model

Having in mind, to build a system for the editing of hardware, you are forced to have available an axact model of both the logic and time-behaviour of digital circuits. (It's nice to have such a model in any case).
Besides more global models /1,2,3,4/ such a model is given by the so called "Quasi Real Boolean Functions" /5/. Those functions allow a very precise description of the logic and time-behaviour of digital circuits.
In this paper a slightly modified approach will be presented. The modification has been done to allow an easier implementation of a hardware-editing-system based upon this model. On the other hand the modelling-power is very little decreased by the modification.
As we want to describe the time-behaviour as well as the logic behaviour we must treat the set of functions

$\Phi := \{f: (\sigma_T)^n \rightarrow (\sigma_T)^m \mid n,m \in \mathbb{N}\}$

mapping a n-tuple of signal-sets σ_T into a m-tuple of these signal sets instead of treating Boolean Functions

$b : \{0,1\}^n \rightarrow \{0,1\}^m$.

By a signal-set we mean the set

$[0,1]^T$, where T is the set $\mathbb{R}$ of real numbers or a denumberable set with the ordering and metric adapted from $\mathbb{R}$.

In the following we restrict ourselves on signalsets $\sigma_{T'}$ with

a) T' denumerable

b) $|t_{i+1}-t_i| = |t_i-t_{i-1}|$ for all $i \in \mathbb{Z}$ (equidistancy)

c) ${}^a t_i \in \{0,1\}$ for all $i \in \mathbb{Z}$.

and an functions mapping those sets.

Def 1 A function $h : (\sigma_{T'})^n \rightarrow (\sigma_{T'})^m$ is called Ideal Boolean Function iff

$h' : \{0,1\}^n \rightarrow \{0,1\}^m$ $\quad x \in (\sigma_{T'})^n$ $\quad t_o \in T'$, $\quad pr_{t_o}(h(x)) = h'(pr_{t_o}(x))$

when pr_{t_o} is the projection at the time t_o. With h'_j we will denote the j-th image-component.

By an inertial-delay-specification we mean a tripel of finite integers, ID := {up,dn,in}, "up" denoting the delay of an "up"-transition, "dn" of a "down"-transition and "in" specifying the minimum number of consecutive points of time a signal must have an identic constant value to be accepted (inertia). In addition "in" must be less than the minimum of "up" and "down".

Def 2 Let ID := {up,dn,in} be an inertialdelay-specification, md := {up,dn}. With respect to ID we call idf : $\sigma_{T'} \to \sigma_{T'}$ inertial-delay-function iff

$$\text{idf}' : \{0,1\}^{\text{md+in}} \to \{0,1\} \;\bigwedge_{x \in \sigma_T} \;\bigwedge_{t_o \in T'} \; pr_{t_o}(\text{idf}(x)) = \text{idf}'(pr_{t_o}(x),\ldots,pr_{t_{o-(md+in)}}(x))$$

where

$$\text{idf}'(pr_{t_o}(x),\ldots,pr_{t_{o-(md+in)}}(x)) = \begin{cases} pr'_{t_{o-up}}(x) & \text{if } pr_{t_o}(x) = 1 \\ pr_{t_{o-dn}}(x) & \text{if } pr_{t_o}(x) = 0 \end{cases}$$

and $pr'_{t_o-\{up\atop dn\}}(x) \neq pr_{t_o-\{up\atop dn\}} \iff$

$$t_o > t_k > t_{o-\{up\atop dn\}} \;\wedge\; t_{o-\{up\atop dn\}} > t_l > t_{o-\{up\atop dn\}+in} \begin{cases} pr_{t_k}(x) \neq pr_{t_o-\{up\atop dn\}} \\ \wedge \\ pr_{t_o}(x) \neq pr_{t_o-\{up\atop dn\}} \end{cases}$$

An intertial-delay-specification is called pure-delay-specification if "in" has the value 0. An intertial-delay-function defined with respect to a pure-delay-specification is called pure-delay-function.

Def 3 Let $Id_i = \{iup,idn,in\}_i =: \{iup_i,idn_i,in_i\}$, i=1,...,n be inertial-delay-specifications and $PD_i = \{oup,odn,0\}_j =: \{oup_j,odn_j,0\}$, j=1,...,m pure-delay-specifications.

Let be udi := $\max_{1\le i\le n} \{iup_i,idn_i\}$,

udo := $\max_{1\le j\le m} \{oup_j,odn_j\}$,

uin := $\max_{1\le i\le n} \{in_i\}$,

his := udi + udo + uin.

A function $q : (\sigma_{T'})^n \to (\sigma_{T'})^m$ is called a Simplified Quasi Real Boolean Function (SQRBF) iff

$$\bigvee q' : \{\{0,1\}^n\}^{his} \to \{0,1\}^m \quad \bigwedge_{x \in (\sigma_{T'})^n} \bigwedge_{t_o \in T'} \; pr_{t_o}(q(x)) = q'(pr_{t_o}(x),\ldots,pr_{t_o-his}(x)).$$

q' is defined by $(q'_1,q'_2,\ldots,q'_m)$ where $q'_j(pr_{t_o}(x),\ldots,pr_{t_o-his}(x)) =$

$$odf_j(h'_j(idf_1(pr_{t_o}(x),\ldots,pr_{t_o-(ud_i+in_i)}(x)),\ldots,idf_n(pr_{t_o}(x),\ldots,pr_{t_o-(ud_i+in_i)}(x)))$$
$$,\ldots,h'_j(idf_1(pr_{t_o-udo}(x),\ldots,pr_{t_o-his}(x)),\ldots,idf_n(pr_{t_o-udo}(x),\ldots,pr_{t_o-his}(x))))$$

h' with the image-componentes $h'_1,\ldots,h'_m$ being the function used above to define Ideal Boolean Functions, idf_i being the inertial-delay-function defined with respect to PD_j. If we want to describe the behaviour of a module only relative to the inputvalues at single points of time, we have to replace the function by an automaton.

It can easyly be, seen, that the following automaton describes the behaviour of a SQRBF: $A := (X,Y,S,\delta,\lambda)$ with

$X := \{0,1\}^n$

$Y := \{0,1\}^m$

$S := si_{1\rho}si_{1,1},\ldots,si_{1\mu di}si_{2\rho},\ldots,si_{n\mu di}\; so_{1o},\ldots,so_{1\mu do}\; so_{2\rho},\ldots,so_{m\mu do}$

is defined by:

1) $\bigwedge_{1\le j\le m} \bigwedge_{o\le k\le udo} \delta(so_{jk}) = so_{jk-1}$

2) $\bigwedge_{1\le i\le n} \bigwedge_{o\le k\le in_i} \delta(si_{ik}) = si_{ik-1}$

3) $\bigwedge_{1\le i\le n} \delta(si_{in_i+1}) = \begin{cases} \overline{si_{i,in_i}} & \text{if } \left(\bigvee_{o<k\le in_i} si_{ik} \ne si_{ik-1}\right) \wedge si_{i,in_i+1} \ne si_{i,in_i} \\ si_{i,in_i} & \text{otherwise} \end{cases}$

4) $\bigwedge_{i\le i\le n} \bigwedge_{in_i\le k\le udi} \delta(si_{ik}) = si_{ik-1}$

5) $\bigwedge_{1\le i\le n} \delta(si_{i\rho}) = x_i$

6) $\bigwedge_{1\le j\le m} (so_{j\mu}) = f'_j(a_1,\ldots,a_n)$ with $\bigwedge_{1\le i\le n} a_i = \begin{cases} si_{upi_i} & \text{if } x_i=1 \\ si_{dni_i} & \text{if } x_i=o \end{cases}$

λ is defined by

$$\bigwedge_{1\le j\le m} \lambda|_{Y_j} = \begin{cases} so_{j,upo_j} & \text{if } so_{j,1} = 1 \\ so_{j,dno_i} & \text{if } so_{j,1} = 0 \end{cases}$$

Note that the behaviour of this automaton is undefined with respect to word-prefixes of a certain length, as we allow each state to be an initial one. One sees immidiately, that this automaton has an extremly simple structure which can be realized with a set of shift-registers and very simple additional logic.
This implies the idea to build an universal hardware-editor which combines the flexibility and transparency of software-implemented simulators with the speed and the relevance of result provided by prototypes.
The basic idea is, to offer to the user a fixed set of digital moduls (IC's or larger units) and the possibility to make connections between these modules automaticly i.e. without manual interference. That means, that besides the modules, the user must have an automatic patch-panel or crossbar-switch. As module-outputs must not be connected together, only a set of selectors is necessary instead of a full crossbar-switch.
This solution whould be impossible, if the time-behaviour of the modules whould be represented by the time-behaviour of the real moduls, because of the distortion of the time-behaviour implied by the circuitry of the "cross-bar". Therefore the time-behaviour must be modelled by the editing system. This is done by providing circuitry realizing an Inertial-delay-function for every module-input (="crossbar"-output) and circuitry realizing a pure delay-function for every module-output (=crossbar"-input). In addition, by this, time is under full control of the system, allowing to stretch, distort and even to stop and restart time without any falsification of the modelling result. Our system therefore is not only a hardware-editor but also includes an "All channel State Analyser" without any time-limit.

2. The META-46 "GOLDLAC", a realization of the above concept

A the University of Dortmund, we decided to build a system realizing the above concepts in standard TTL-techn. We restricted ourselves on a system allowing to connect automaticly a digital circuit out of a given but changable set of modules where the sum of module-outputs must not be greater than 128 and the sum of the module-inputs not greater one 192.
From the above discussion follows, that for the basic system, we need

- a "crossbar":"128 x 192"
- 128 times a realization of a universal pure delay function
- 192 times a realization of a universal inertial delay function.

2.1 The "crossbar"

As outputs of digital modules must not be connected together, we can restrict ourselves on 192 selectors "1 out of 128" instead of being forced to build a completete crossbar. A selector "1 out of 128" can be build very easyly out of standard TTL-chips using a two level realization.
In addition a 7-bit register is needed to store the address of the actual selected line. The actual data within these 192 7-bit-registers specifies relative to the set of modules plugged in at that moment the circuit being build by the system.
From this follows, that based on a given set of modules up to 2^{1344} different circuits may be build without any manual interference. Our "crossbar" consists out of 24 PC-boards, each certaining 8 seletors "1 out of 128" with their registers.

2.2 The realization of an universal pure delay function (see fig. 1)

We want to be able to assign a delay to every module by the system. This delay should possibly be different for up and down transitions. We restricted ourselves on a possible range of delays between 1 and 16 time-units. A time unit may stand for one nano-second (for example) and is represented by one clock cycle within the system. Thus, an universal pure delay function can be build very easyly out of a parallel readable 16-bit shift-register, a selector "1 out of 16", two 4-bit-registers to store the actual "up"-and "down"-delay and some additional circuitry.(This circuitry consits only out of 4 EXOR-gates, as we store the bitwise boolean difference between the "up"-delay and the "down"-delay in the register for the "up"-delay). In our system the set of 128 pure delay-functions is distributed on 8 PC boards, each containing 16 pure delay functions.

2.3 The realization of an universal inertial-delay function (see fig. 2)

As we want to be able to assign an inertial delay to each module-input we had to construct a realization of an universal inertial delay function. This has been done by adding some circuitry for an universal inertia representation to the realization of a pure delay function. As this complicates the circuit, we restricted ourselves on a rarge for delay and inertia between 1 and 8 time-units.
The inertia is implemented by adding to a pure delay function the ability to store into the n-th cell of the shift-register not only the contence of the n-1-th cell but also the complemented value of this cell.
The latter is done iff

1) the n-th output of a inertia specifying demultiplexor carries a "high",
2) the n-th and the n-1-th cell carry different values,
3) any two other neighboured cells i, i-1 with i<n-1 carry different values.

In our system the set of 192 universal inertial delay functions is distributed on 24 PC-borads, each containing 8 such functions.

2.3 Additional components of the META-46 "GOLDLAC"

As in general circuits of interest are not autonomous ones the inputs into a circuit under test must be representable.
We do this by an additional 128 bit register. Each of the lines going into the "crossbar" may be connected with the corresponding cell of this register intead of being connected with a module-output. This is done by selectes "1out of 2" controlled by another 128-bit register.
Secondly, oberservation of the automaticly built circuit must be able, of course. For this pupose an arbitrary subset of the 128 lines going into the "crossbar" can be selected as "signalset of interest". Any transition on one of these selected lines causes an interrupt signal, that stops the machine, preserving its actual state. This state may then be read by a controlling minicomputer, which is used also to fill all the registers of the system.
For our system we use a PDP-11 as "mother-computer", coupled with the META-46 "GOLDLAC" by a parallel interface.
On this PDP-11 is running a small software-package that allows an interactive processing of the system.
The user defines relative to a given set of modules (if he supplies his own set he has first to describe it) a circuit by specifying the connections and assigning delays and inertias to lines. In addition he defines some circuit-inputs, and a set of lines of interest. He also may change all his specification interactively like in usual text-editors.
Having specified his circuit he loads and starts the system by a simple command. At any time when one of his "signals of interest" changes its value the values of all "signals of interest" are outputted on the user's terminal together with the system-time (simulated-time). The user now may simply registrate the new values or react by changing the input-values or the circuit itself.
The complete dialoque is protocolled on a logfile.
Future software-packagaes will immidiately process circuit-descriptions in a Computer Hardware Description Language (DIGITEST) /6/, will allow single user time-sharing for the simulation of larger circuits and multi-user time-sharing for a more economic use of the system.

References

/1/ Beister,J., A Unified approach to combinational hazards, IEEE TOC C-23 (1974), pp 566-575

/2/ Brzozowski, J.A. & Yoely,M., "Digital Networks" (Prentice-Hall, 1976)

/3/ Miller, R.E., "Switching Theory", Vol. II (J.Wiley & Sons, New York, 1965)

/4/ Noguchi, A. et al., "Mathematical Theory of Asynchronous Circuits I-III" (Waseda University Press, Tokyo 1963-1966)

/5/ Rammig, F.J., Quasi Reale Boolesche Funktionen: Ein Versuch, zeitliche Effekte physikalisch realisierter Bauteile zu algebraisieren, Digital Processes, 2 (1976) pp. 27-45

/6/ Rammig, F.J., Der Übersetzer DIGITEST (Version 2.1) Bericht Nr. 8 der Abteilung Informatik der Universität Dortmund (1975)

Fig. 1 Draft of the realization of a pure delay function

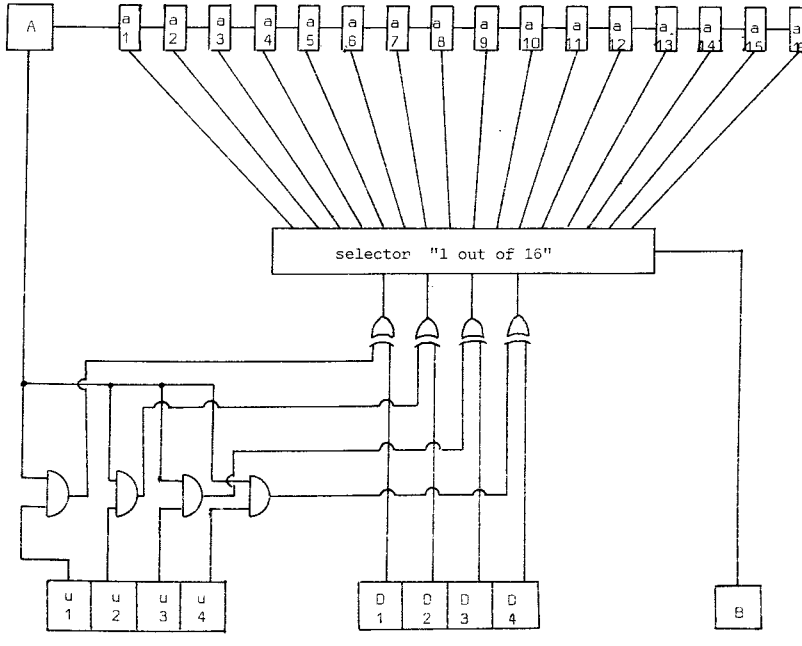

Fig. 2 Draft of the realization of an inertial delay function

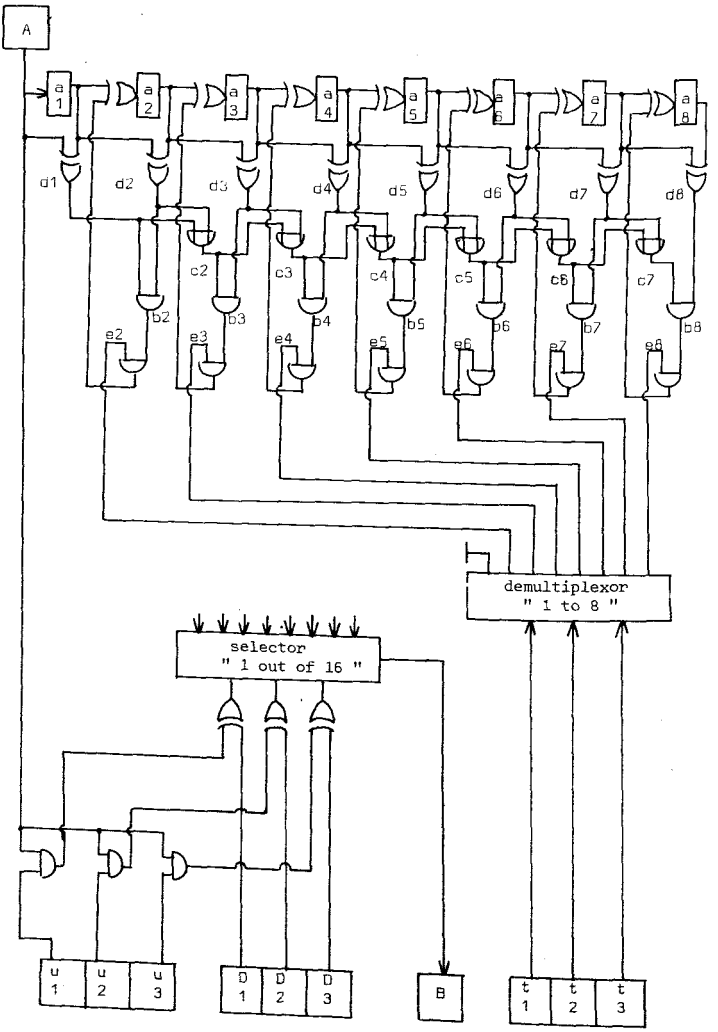

Fig. 3 Draft of the total system (only basic components)