APPLICATION OF TIMED PETRI NETS TO ANALYSIS OF MULTIPROCESSOR REALIZATIONS OF DIGITAL FILTERS

Wlodzimierz M. Zuberek
Department of Electrical and Computer Engineering
McMaster University, Hamilton, Ont. L8S 4L7, Canada

Abstract

A rather restricted class of timed Petri nets is presented, and some of their basic properties are derived. Subsequently, the nets are applied to the analysis of a fifth order wave digital filter in the Jouan structure. The maximum sampling rate of the filter is derived as a function of the times required to perform arithmetic operations involved in the filtering process. A multiprocessor realization of the filter is discussed and some generalizations are indicated. The approach can easily be adapted to many other problems arising in hardware realizations of parallel and concurrent algorithms.

1. INTRODUCTION

In the hardware implementation of high-speed real-time digital signal processing algorithms there is often possibility of using several slower processing units instead of a single fast one. If a very high-speed is required, parallel processing may appear to be the only way to satisfy the requirements. The realization of digital filters using multiple processing units [2,3,6] is a special case of parallel processing because the digital filtering algorithms do not contain data-depending branching, and the detection of parallelism is relatively easy. The input data to digital filtering is a continuous stream of numbers for which the same operations are performed during each sampling period. The execution time of any sequence of operations is thus equal, and the parallelism is a deterministic feature of the algorithms.

In digital filtering algorithms the computations during each sampling period involve the evaluation of a set of arithmetic expressions which determine the output and the new delay of inputs in terms of the input and the previous delay. There are several methods to find maximal parallelism of such computations, for example the precedence graph methods, the critical paths methods, and the methods based on Petri nets [1,4,5]. Timed Petri nets seem to be especially convenient for the representation and the analysis of systems that exhibit interactions of multiple communicating units because they take into account not only precedence relationships of such interactions, but can express the time required by each step of interaction as well.

The aim of this paper is to present an application of a rather restricted class of timed Petri nets to the analysis of a multiprocessor realization of wave digital filters. Several basic properties of timed Petri nets are derived and then some implementations of a fifth order wave digital filter in the Jouan structure are discussed, their maximum sampling rates are evaluated, and processor utilization factors are given.

2. MARKED PETRI NETS

A Petri net N is a triple N = (P,T,E), where
P is a finite nonempty set of places,
T is a finite nonempty set of transitions,
E is a set of directed edges, E ⊆ P x T x P,
such that \( \forall (t \in T) \exists (p_i,p_j \in P) (p_i,t) \in E \land (t,p_j) \in E. \)

A place p is an input place of a transition t if there exists an edge (p,t) in the set E. The set of all input places of a transition t is denoted as Inp(t)

\[ \text{Inp}(t) = \{ p \in P | (p,t) \in E \}. \]

Similarly,

\[ \text{Inp}(p) = \{ t \in T | (t,p) \in E \}. \]
Out(t) = |p ∈ P | (t, p) ∈ E |
Out(p) = \{ t ∈ T | (p, t) ∈ E \}.

Moreover, if P_i is a subset of P, and T_j is a subset of T, then
Inp(P_i) = \{ t ∈ T | \exists (p ∈ P_i, t, p) ∈ E \}
Inp(T_j) = \{ p ∈ P | \exists (p ∈ T_j, p, t) ∈ E \}
Out(P_i) = \{ t ∈ T | \exists (p ∈ P_i, t, p) ∈ E \}
Out(T_j) = \{ p ∈ P | \exists (p ∈ T_j, p, t) ∈ E \}.

A Petri net N = (P, T, E) is simple if for each place p ∈ P there exists exactly one input and one output transition. Only simple Petri nets are considered in this paper.

A marked Petri net M is a pair M = (N, m_0), where N is a Petri net, N = (P, T, E), m_0 is an initial marking function that assigns a nonnegative integer to each place of the net m_0 = P × {0, 1, 2, ...}.

Let any function m : P × {0, 1, 2, ...} be called a marking of a net N = (P, T, E).

A transition t is enabled by a marking m if every input place of t contains at least one token. The set of all transitions enabled by a marking m is denoted as T(m).

Every transition enabled by a marking m can fire. When a transition fires, a token is removed from each of its input places and a token is added to each of its output places. This determines a new marking in a net, a new set of enabled transitions and so on.

A marking m_1 is directly reachable from a marking m_2 in a net N, m_1 → m_2, if there exists a transition t enabled by the marking m_2, t ∈ T(m_2), such that
\[ W(p ∈ P) m_2(p) = \begin{cases} m_1(p) - 1, & \text{if } p ∈ \text{Inp(t)} \setminus \text{Out(t)} \\ m_1(p) + 1, & \text{if } p ∈ \text{Out(t)} \setminus \text{Inp(t)} \\ m_1(p), & \text{otherwise}. \end{cases} \]

A marking m_1 is reachable from a marking m_k in a net N, m_1 → m_k, if there exists a sequence of markings \( m_{1:k} = m_1, m_2, ..., m_k \) such that \( m_1 = m_1, m_k = m_k \), and each marking \( m_i \) is directly reachable from \( m_{i-1} \), for \( i = 1, 2, ..., k \).

A set M(M) of reachable markings of a marked Petri net M = (N, m_0) is the set of all markings reachable from the initial marking m_0
\[ M(M) = \{ m | m \sim m_0 \}. \]

A marked net M is bounded if there exists a positive integer k such that each marking in the set M(M) assigns at most k tokens to each place of the net
\[ \forall (m ∈ M(M)) \exists \ (p ∈ P) m(p) \leq k. \]

For each bounded marked net M its set M(M) of reachable markings is finite. Only bounded nets are considered in this paper.

3. TIMED PETRI NETS

A timed Petri net T is a pair T = (M, f), where M is a marked Petri net, M = (N, m_0), N = (P, T, E),
f is a firing time function that assigns a positive rational number to each transition of the net
\[ f : T → R^+ \]
and \( R^+ \) denotes the set of positive rational numbers.

In a timed net each transition t takes a positive time f(t) to fire. When a transition t is enabled, a firing can be initiated by removing a token from each of the t's input places. This token remains in the transition t for the time f(t) and then the firing terminates by adding a token to each of t's output places. Each of the firings is initiated in the same instant of time in which it is enabled, unless a transition is enabled while it fires (in such a case a "delayed" firing is initiated immediately after termination of the current one.

The operation of a timed net can thus be considered as taking place in real time and it is assumed that it starts at the time t_0 = 0. At this moment the firings of the transitions enabled by the marking m_0 are initiated and the tokens are removed from their input places. Then, after the time determined by the smallest firing time of the transitions which initiated firings, the tokens are deposited in appropriate output places creating a new marking, a new set of enabled transitions, and so on.

The behaviour of a simple timed Petri net can be described by a sequence \( t_0, t_1, t_2, ... \) of those moments in which firings initiate and terminate. Let A_i, B_i, C_i and D_i, i = 1, 2, ..., denote the sets of transitions which are active, which initiate firing, which continue firing, and which terminate firing at the moment t_i, respectively. Let A_i = B_i ∪ C_i, i = 1, ..., k. Moreover, let a family of partial functions r_i : T → R^+ describe the remaining firing times for transitions which are active at the moment t_i. Then:
\[ \tau_0 = 0, \]
\[ \tau_1 = \min \{ f(t), \quad t \in T(m_0) \}, \]
\[ D_1 = \{ t \in T(m_0) \mid f(t) = \tau_1 \}, \]
\[ C_1 = T(m_0) - D_1, \]
\[ \psi(p_c P) m_1(p) = \begin{cases} m_0(p) + 1, & \text{if } p \in P_c(T(m_0)) - \text{Out}(D_1), \\ m_0(p), & \text{otherwise}, \end{cases} \]
\[ B_1 = T(m_1), \]
\[ A_1 = B_1 \cup C_1, \]
\[ \psi(t \in A_1) \tau_1(t) = \begin{cases} f(t), & \text{if } t \in B_1, \\ f(t) - \tau_1, & \text{otherwise}, \end{cases} \]
\[ \tau_2 = \tau_1 + \min \{ \tau_j(t) \}, \quad t \in A_1 \]
and, generally:
\[ \tau_{j+1} = \tau_j + \min \{ \tau_j(t) \}, \quad j \in A_j \]
\[ D_{j+1} = \{ t \in A_j \mid \tau_j(t) = \tau_{j+1} - \tau_j \}, \]
\[ C_{j+1} = A_j - D_{j+1}, \]
\[ \psi(p_c P) m_{j+1}(p) = \begin{cases} m_j(p) + 1, & \text{if } p \in P_c(T(m_j)) - \text{Out}(D_{j+1}), \\ m_j(p), & \text{otherwise}, \end{cases} \]
\[ B_{j+1} = T(m_{j+1}), \]
\[ A_{j+1} = C_{j+1} \cup B_{j+1}, \]
\[ \psi(t \in A_{j+1}) \tau_{j+1}(t) = \begin{cases} f(t), & \text{if } t \in B_{j+1}, \\ \tau_{j+1}(t) - \tau_{j+1} - \tau_j, & \text{otherwise}. \end{cases} \]

The sequence \( \tau_0, \tau_1, \tau_2, \ldots \) is thus deterministic and can be either finite (in such a case a net contains a so-called deadlock), or infinite but periodic [7], i.e., there exist integers \( j \) and \( k \) such that
\[ \psi(i \geq 1) \tau_{i+k} = \tau_1 = \tau_{i+2k} - \tau_{i+k}. \]

The minimal value of \( k \) determines the so-called cycle time \( \tau_0 \) of a net which is one of the basic characteristics of a timed Petri net. In the context of digital filters, the cycle time corresponds to the maximum sampling rate of the filter.

4. NET TRANSFORMATIONS

For large nets the straightforward derivation of the cycle time \( \tau_0 \) can be quite complex and therefore some transformations which preserve the cycle time may be very helpful. A set of such transformations is shown in Fig. 1:

(a) serial reduction:
\[ f(t') = f(t_1) + f(t_2), \]
\[ m_0(p_{j+1}) = m_0(p_j) + m_0(p_{j+1}), \quad i = 1, \ldots, n; \]
(b) forward splitting:
\[ f(t') = f(t_1), \]
\[ m_0(p_{j+1}) = m_0(p_j); \]

Fig. 1 Basic net transformations
(c) backward splitting:
\[ f(t_2^b) = f(t_2) = f(t_2^o), \]
\[ m_0(p_1^b) = m_0(p_1^o) = m_0(p_1); \]

(d) parallel reduction:
\[ f(t_4) = \max(f(t_2), f(t_3)), \]
\[ m_0(p_2^b) = 0, m_0(p_2^o) = \min(m_0(p_1) + m_0(p_2), \]
\[ m_0(p_3) + m_0(p_3)); \]

(e) serial splitting:
\[ f(t_2^s) = f(t_1), f(t_2^s) > 0, f(t_2^s) > 0, \]
\[ m_0(p_1^b) = 0; \]

(f) forward joining, provided \( f(t_2) = f(t_3) \):
\[ f(t_4) = f(t_2) = f(t_3), \]
\[ m_0(p_1^s) = 0, m_0(p_3) = m_0(p_1) + m_0(p_2), \]
\[ m_0(p_3^b) = m_0(p_2) + m_0(p_3); \]

(g) backward joining, provided \( f(t_1) = f(t_2) \):
\[ f(t_4) = f(t_1) = f(t_2), \]
\[ m_0(p_1^s) = 0, m_0(p_1) = m_0(p_2) + m_0(p_3), \]
\[ m_0(p_3^b) = m_0(p_2) + m_0(p_3); \]

(h) bypass reduction, provided \( m_0(p_1) = m_0(p_2) + \]
\[ m_0(p_3). \]

An example of a more complex transformation is shown in Fig. 2 for a "chain" net composed of three cells (Fig. 2(a)). Assuming that \( f(t_3^s) > f(t_3) \) the net can be transformed to that of Fig. 2(b) by serial splitting the transition \( t_1 \) into \( t_1^s \) and \( t_1^o \) in such a way that \( f(t_2^s) = f(t_3), \) then by forward joining the transitions \( t_2 \) and \( t_3, \) then by backward splitting the transition \( t_2 \) into \( t_2^s \) and \( t_2^o \) and finally by serial reductions of transitions \( t_4 \) and \( t_4^b \) into \( t_4^s \) and \( t_4^o \) into \( t_4. \) Now, if \( f(t_4) = f(t_1) + f(t_2) = f(t_3) > f(t_4), \) the similar series of basic transformations converts the net shown in Fig. 2(b) into that of Fig. 2(c), where \( f(t_4) = f(t_1) + f(t_2) - f(t_3) \) and \( f(t_4) = f(t_2) + f(t_3) - f(t_4). \) The net from Fig. 2(c), after parallel and serial reductions is shown in Fig. 2(c), where

\[ f(t_0) = \max(f(t_1), f(t_2), f(t_3) + f(t_4) + f(t_5) + f(t_6)). \]

Alternatively, if for the net shown in Fig. 2(b), \( f(t_2^o) < f(t_2), \) another series of basic transformations results in the net shown in Fig. 2(d) where \( f(t_1) = f(t_1), f(t_2^o) = f(t_5) + f(t_6) - f(t_3), f(t_3) = f(t_4), \) and \( f(t_4) = f(t_2) + f(t_3) + f(t_4) + f(t_5) - f(t_6). \) The nets in Fig. 2(c) and Fig. 2(d) are structurally identical and the same transformations may be applied to obtain the net shown in Fig. 2(e).

5. REALIZATIONS OF DIGITAL FILTERS

Wave digital filters in the Jahan model and structure have been discussed in [3], and a fifth order low pass wave digital filter in the same structure is
Fig. 3 Information flow diagram

presented in [4] in greater detail. Its information flow diagram containing all the arithmetic operations involved in the filtering process is shown in Fig. 3. The information diagram can easily be converted into an equivalent timed Petri net (Fig. 4), in which transitions represent arithmetic operations of the filter (they are described by + and x signs, respectively), and places correspond to arguments and results of these operations. By removing all bypass connections the net from Fig. 4 can be transformed to the net shown in Fig. 5, and then by serial reductions to the net shown in Fig. 6, where

\[ f(t_1) = 2a + m \quad f(t_5) = 4a + m \]
\[ f(t_2) = 2a + m \quad f(t_6) = a \]
\[ f(t_3) = 2a + m \quad f(t_7) = a \]
\[ f(t_4) = 2a + m \quad f(t_8) = d \]

and a is the time required to perform addition, m is the time required to perform multiplication, and d is the time required to transfer the input data. To obtain the cycle time corresponding to the maximum sampling rate of the filter, its output should be recycled to the input, which results in the net shown in Fig. 7, where \( f(t_9) = f(t_8) + f(t_{12}) \). The cycle time of the net is thus equal to

\[ t_0 = \max \left( f(t_1) + f(t_2) + f(t_3) + f(t_6) + f(t_8), f(t_4) + f(t_9), f(t_5) + f(t_{12}) \right) \]

and \( t_0 \) is the time required to perform the operations involved in the filtering without any particular realization of these operations. If all the operations are to be performed by one (sequential) processor (in some order, preserving the precedence relations), the cycle time will be equal to the sum of all operation times

\[ t_0^{(1)} = f(t_1) + f(t_2) + f(t_3) + f(t_4) + f(t_6) + f(t_8) + f(t_9) = 67 \text{~ms} \]

This is, however, the maximum sampling rate which takes into account only precedence of operations involved in the filtering and without any particular realization of these operations. If all the operations are to be performed by one (sequential) processor (in some order, preserving the precedence relations), the cycle time will be equal to the sum of all operation times

\[ t_0^{(1)} = f(t_1) + f(t_2) + f(t_3) + f(t_4) + f(t_6) + f(t_8) + f(t_9) = 67 \text{~ms} \]

For multiprocessor realizations of filtering algorithms, the computations should be distributed among the processors as uniformly as possible, preserving the required precedence of operations and minimizing the number of interactions between processors (interactions are regarded as additional operations with appropriate execution times). A 3-processor realization of the net from Fig. 7 is shown in Fig. 8 with explicit
Fig. 8 Petri net for 3-processor realization

representation of interactions in the form of additional transitions. It should be observed that
the net of Fig. 8 is structurally identical with that of Fig. 7. Assuming that the
interaction time is equal to 1 μs, its cycle time is
\[ \tau_{(3)} = \max(28, 28, 19, 18, 28) = 28 \mu s. \]

The utilization factor \( u_i \) of a processor is usually defined as a ratio of the time in which the processor is active to the total time elapsed. For the 3-processor realization (with the maximum sampling rate) the utilization factors \( u_i, i = 1, 2, 3 \), are thus equal to
\[ u_1 = \frac{28}{28} = 1.0 = 100 \%, \]
\[ u_2 = \frac{19}{28} = 0.68 = 68 \%, \]
\[ u_3 = \frac{28}{28} = 1.0 = 100 \%. \]

It can be observed that the processors 1 and 3 which are utilized in 100% perform computations that are strictly sequential (see Fig. 5) and therefore no better realization for the filter can be obtained for the assumed operation times. However, if the multiplication time can be reduced to 5 μs, the corresponding cycle time is equal to
\[ \tau_{(3)} = \max(18, 18, 14, 13, 18) = 18 \mu s, \]
which is 36% less than the previous one. Many other results can be derived in a very similar way.

6. CONCLUDING REMARKS

A rather restricted class of timed Petri nets is considered in the paper, however, the
restrictions are imposed on the nets first of all to simplify net descriptions and to derive some
properties of timed nets without elaborated formalism. Some of the restrictions can be
removed if a more powerful description is used [7].

The results obtained for a digital filter are given in a very general form (in fact, an
analytical solution has been derived for the maximum sampling rate), and therefore subsequent
analysis of realizations and selection of optimal realization parameters (for example, the
operation times) is rather simple and straightforward. It can be observed that the same
approach can be used for analysis of heterogeneous realizations, i.e., for multiprocessor
realizations in which different processors are used, and the time required for the same
operation (addition, multiplication) may differ from processor to processor.

Finally, the approach presented here is rather systematic and can be implemented in the form of
computer programs to analyze large digital structures. In such a case, however, a more
detailed study of the properties of timed Petri nets is probably required.

REFERENCES


Wlodzimierz M. Zuberek received the M.Sc. degree in Electronics and the Ph.D. degree in Computer Science, both from the Warsaw Technical University, Warsaw, Poland. Since 1966 he has
been with the Institute of Computer Science of the Warsaw Technical University. In 1981 he received a Postdoctoral Fellowship from McMaster University, Hamilton, Canada, where he joined the Group on Simulation, Optimization and Control and the Department of Electrical and Computer Engineering.