– A TEMPORAL INFERENCE ACCELERATOR UNIT –

ESPRIT 2434

Contribution to:

Activity 2.5

- System Concept for Advanced CIM/AI Controller -

Title:

A TEMPORAL INFERENCE ACCELERATOR UNIT

Author/Company/Copyright: Jörg - Ingo Jakob

Achievements:

A novel microelectronics hardware unit suitable for integration into digital integrated circuits was invented. The domain of this temporal inference accelerator unit (TIU for short) is the field of qualitative temporal logic amongst time intervals. A key idea is the representation of compound temporal relations as binary words, in order that the transitive function of two arbitrary compound relations can be computed in parallel, i. e. in constant time. A speedup factor of one to four orders of magnitude compared to a software implementation of the transitivity table is expected, depending on the design of the software driver program and on the computer hardware, and the type of execution parallelism involved. The TIU can be easily integrated onto a conventional PCB inside a computer workstation; as part of a larger VLSI system, or as a standalone circuit. A patent for the TIU was applied for.

Summary:

The TIU is a hardware implementation of the transitivity table central to Allen's qualitative, interval-based temporal logic. In the TIU, temporal relations amongst two time intervals are represented as an n-bit binary word. Two such words, which describe part of the temporal relations amongst three connected time intervals, allow the computation of the third, missing relation. Thus, a TIU maps two binary words of length n onto one binary word of length n. [Typical size for n: n = 13]. Two possible instantiations of the TIU are shown in detail. Both instantiations demonstrate the general concept of a TIU.

Overview:

1. Motivation
2. Review of Work
3. Definition of Terms
4. Prototype Structure and Performance
5. Realization
6. Example 1: Six-Valued Temporal Logic
7. Example 2: Thirteen-Valued Temporal Logic (Allen's Logic)
8. Literature
9. Appendix

1. Motivation

Temporal logic answers questions such as: "When A happens before or parallel to B, and when B starts C or when B is before C, in which relation is A standing to C ?" These questions can be asked on complex issues, e. g. job scheduling on the factory floor, aircraft scheduling in airlines, etc.

2. Review of Work

Allen's temporal logic [Allen 1983, 1984] is a successful example of an algebraic system describing (qualitative) relations amongst time intervals. Several computer programs in the area of artificial intelligence have been based on this particular temporal logic [Allen 1983, Vilain 1982, Rit 1986, Becker 1988, Huber 1988]. However, these programs run on general purpose computers and are not supported by special hardware. Thus, they are inherently slow.

3. Definition of Terms

"Temporal logic" and "qualitative temporal logic" are defined in scientific literature. Cf. [Rescher, Urquhart 1971], [Allen 1983], [Huber, Becker 1988].

4. Prototype Structure and Performance

A simple prototype system may be implemented using a standard bus based system (e. g. PC, PS, EISA or VMEBus based workstation). The hardware may be implemented as a desktop-programmable gate array, or as programmable logic. The TIU will occupy one macro cell (or several identical macro cells, in a parallel processing system).

Using this new design, the simplest conceivable system which uses TIU hardware may consist of:

(1) a user interface;
(2) an expert system component capable of describing (temporal) situations by objects and relations amongst objects, and capable of connecting to other expert systems and data bases;
(3) a driver program that updates a temporal net in the computer's storage, and interfaces to the TIU;
(4) the TIU which executes Allen's transitivity table in hardware.

This could be mapped upon a conventional computer equipped with a bus system: components (1) and (2) are executed by the host, component (3) uses the host memory (or memory connected by bus), and component (4) is situated on a separate board connected to the host via bus.

The immediate gain in computational speed is impressive. Compared to a conventional software solution on a Lisp machine [Becker 1988], which consumes (on a small example) 407 seconds of cpu time doing 730,000 temporal inferences, or approximately 1,800 temporal inferences per second, we expect the hardware to perform 10 million or more temporal inferences per seconds (TIPS), limited only by the bandwidth of the host system bus, the wait states of the hardware for

the cpu, or a slow driver program. This is a speedup of nearly four orders of magnitude.

5. Realization

The TIU hardware device can be implemented as dedicated electronics hardware. This may be a gate array, a standard cell array, full customs VLSI hardware, programmable logic, and parts of any of these. Even a ROM-based solution is feasible, though not economical.

6. Example 1: Six-Valued Temporal Logic

We assume the existence of a qualitative six-valued temporal interval logic. The six relations (values) amongst intervals are:

forward relation	inverse relation
B (before)	A (after)
D (during)	C (contains)
O (overlaps)	OI (overlapped by)

If we want to express a statement like "interval P is before interval Q or contains interval Q", we simply write "P (B,C) Q".

Pictoral examples:

(1a) before:	X (B) Y	:	XXX YYY
(1b) before:	X (B) Y	:	XXXYYY
(2a) during:	X (D) Y	:	XXX YYYYY
(2b) during:	X (D) Y	:	XXX YYYYY
(2c) during:	X (D) Y	:	XXX YYYYY
(3) overlaps:	X (O) Y	:	XXX YYY
(4a) after:	X (A) Y	:	YYY XXX
(4b) after:	X (A) Y	:	YYYXXX
(5a) contains:	X (C) Y	:	YYY XXXXX
(5b) contains:	X (C) Y	:	YYY XXXXX
(5c) contains:	X (C) Y	:	YYY XXXXX

(6) overlapped by: X (OI) Y : YYY
XXX

[Note: the equality of two time intervals A, B can be enforced by specifying that (A (D) B) & (A (C) B)].

How to answer a more complex question like "When A happens before or contains B, and when B contains C or when B is overlapped by C, in which relation is A standing to C ?" As a formula: (A (B,C) B) & (B (C,OI) C) ?

Step 1: This can be done by the introduction of a transitivity table. A transitivity table answers all questions of the form "(A R1 B) & (B R2 C) ==> (A R3 C)", with R1, R2, R3 being arbitrary compound relations.

R3=?	B	O	D	A	OI	C
B	B	B	B,O,D	T	B,O,D	B
O	B	B,O	O,D	A,OI,C	O,OI,D,C	B,O,C
D	B	B,O,D	D	A	A,OI,D	T
A	T	A,OI,D	A,OI,D	A	A	A
OI	B,O,C	O,OI,D,C	OI,D	A	A,OI	A,OI,C
C	B,O,C	O,C	O,OI,D,C	A,OI,C	OI,C	C

[Note: let T = (B,O,D,A,OI,C)].

Step 2: To compute R3, one looks up each simple relation of R1 (as a row index) by each simple relation of R2 (as a column index). The union of all entries is the desired result R3.

Example: (A (B,C) B) & (B (C,OI) C) ==> R1 = (B,C), R2 = (C,OI). We look up

row	column	result
B	C	B
B	OI	B,D,O
C	C	C
C	OI	OI,C

and receive as a union of results R3 = (B,O,D,OI,C). Thus interval P may stand in any relation to interval Q, except that never P (A) Q.

Step 3: Such a transitivity table can be set up by systematically examining each simple combination of R1 and R2. By analysis of the transitivity table, one finds that there are exactly 16

compound or simple relations as instances of R3: (B), (B,O,D), T, (B,O), (O,D), (A,OI,C), (C,D,O,OI), (C), (D), (A), (A,OI,D), (B,O,C), (OI,D), (A,OI), (OI,C), (O,C). In the context of this example, we call each of these 16 relations a "state".

Step 4: Now we can specify a set of boolean equations for each of the 16 states. (The row values are taken as 6 independant boolean variables B1, O1, D1, A1, OI1, C1; and the column values are taken as 6 independant boolean variables B2, O2, D2, A2, OI2, C2).

$(B)^S = B1B2 + B1O2 + B1C2 + O1B2 + B1B2$

$(B,O,D)^S = B1D2 + B1OI1 + D1O2$

$T^S = B1A2 + D1C2 + A1B2$

$(B,O)^S = O1O2$

$(O,D)^S = O1D2$

$(A,OI,C)^S = O1A2 + O1C2 + C1A2$

$(C,D,O,OI)^S = O1OI2 + OI1O2 + C1D2$

$(C)^S = C1C2$

$(D)^S = D1D2$

$(A)^S = D1A2 + A1A2 + A1OI2 + A1C2 + OI1A2$

$(A,OI,D)^S = D1OI2 + A1O2 + A1D1$

$(B,O,C)^S = O1C2 + OI1B2 + C1B2$

$(OI,D)^S = OI1D2$

$(A,OI)^S = OI1OI2$

$(O,C)^S = C1O2$

$(OI,C)^S = C1OI2.$

Step 5: By mapping the 16 states upon the simple relations B, O, D, A, OI, C, we receive another 6 boolean equations:

$B^S = (B)^S + (B,O,D)^S + T^S + (B,O)^S + (B,O,C)^S$
$O^S = (B,O,D)^S + T^S + (B,O)^S + (O,D)^S + (O,OI,D,C)^S + (B,O,C)^S + (O,C)^S$
$D^S = (B,O,D)^S + T^S + (O,D)^S + (O,OI,D,C)^S + (D)^S + (A,OI,D)^S + (OI,D)^S$
$A^S = T^S + (A,OI,C)^S + (A)^S + (A,OI,D)^S + (A,OI)^S$
$OI^S = T^S + (A,OI,C)^S + (O,OI,D,C)^S + (A,OI,D)^S + (OI,D)^S + (OI,C)^S$
$C^S = T^S + (A,OI,C)^S + (O,OI,D,C)^S + (B,O,C)^S + (O,C)^S + (OI,C)^S + (C)^S$

Step 6: Now, a compound relation between two intervals can be represented as a 6-bit word, when we assign the first bit to B1 or B2, the second bit to O1 or O2, etc. E.g.: (A (B,C) B) & (B (C,OI) C) yields the words "100001" for R1 and "000011" for R2. By realizing the above boolean equations in hardware, we receive a hardware transitivity table (the TIU). The TIU of this particular, six-valued logic has two input words of 6 bits and one output word of six bits. The union of simple relations (as

described in step 2) is performed by the drafted hardware in parallel, that is, in constant time.

Step 7: Steps 4 and 5 can be summarized in one step, where the 16 "intermediate" states are NOT computed. Instead the 6 equations of step 5 are computed directly from the 6 x 6 = 36 table entries of the transitivity table. Example:

O^S = B1D2 + B1A2 + B1OI2 + O1O2 + O1D2 + O1OI2 + O1C1 + D1O2 + D1C2 + A1B2 + OI1B2 + OI1O2 + C1B2 + C1O2 + C1D2

Step 8: depending on the available technology for the dedicated VLSI hardware, the first scheme (steps 4 and 5) or the second scheme (step 7) may be more suitable for implementation. From the structure of the above equations it is clear that the equations can be easily simplified by application of the laws of boolean algebra. (This can be advantageously done under consideration of a certain VLSI technology). However, this tends to be a special procedure for a certain technology and will not be shown here.

7. Example 2: Thirteen-Valued Temporal Logic (Allen's logic)

In Artificial Intelligence literature, Allen's thirteen-valued qualitative temporal logic is a well-known example of time logic [Allen 1983]. The thirteen values of this logic are:

forward relation	inverse relation
b (before)	a (after)
d (during)	c (contains)
o (overlaps)	oi (overlapped-by)
m (meets)	mi (met-by)
s (starts)	si (started-by)
f (finishes)	fi (finished-by)
e (equals)	e (equals)

To distinguish from the relations of example 1, small letters are used. Conceptually, the relations are the same as in the six-valued logic of example 1, except that: relations "before" and "after" are subdivided into Allen's before, meets resp. after, met-by; and relations "during" and "contains" are subdivided into Allen's during, starts, finishes resp. contains, started-by, finished-by; and a relation equals is added.

Pictoral examples:

(1a)	before:	X (b) Y	:	XXX YYY
(2)	meets:	X (b) Y	:	XXXYYY
(3)	during:	X (d) Y	:	XXX YYYYY
(4)	starts:	X (d) Y	:	XXX YYYYY
(5)	finishes:	X (d) Y	:	XXX YYYYY

(6)	overlaps:	X (o) Y	: XXX YYY
(7)	after:	X (a) Y	: YYY XXX
(8)	met-by:	X (a) Y	: YYYXXX
(9)	contains:	X (c) Y	: YYY XXXXX
(10)	started-by:	X (c) Y	: YYY XXXXX
(11)	finshed-by:	X (c) Y	: YYY XXXXX
(12)	overlapped-by:	X (oi) Y	: YYY XXX
(13)	equals:	X (e) Y	: XXX YYY

The procedure for translating this logic into boolean equations and therefrom into a TIU is similar to example 1. The same steps can be identified:

Step 1: The transitivity table of this thirteen-valued logic is documented in [Allen 1983, p. 834] and is reprinted in the appendix. (The table is a matrix of 13 rows and 13 columns).

[Note: let T = (a,b,c,d,e,f,fi,m,mi,o,oi,s,si)].

Step 2: -- cf. example 1 --

Step 3: Again, the transitivity table can be set up by systematically examining each simple combination of R1 and R2. By analysis of Allen's transitivity table, one finds that there are exactly 27 compound or simple relations as instances of R3. Again, in the context of this example, we call each of these 27 relations a "state".

Step 4: It is possible to specify a set of boolean equations for each of the 27 states, similar to example 1. For brevity, we give only two examples:

$$(c,d,e,f,fi,o,oi,s,si)^S = oi1o2 + o1oi2 + c1d2$$

$$(c,oi,si)^S = fi1mi2 + fi1oi2 + oi1fi2 + o1mi2 + c1f2 + c1mi2 + c1oi2$$

The remaining 25 equations can easily be determined in a similar manner.

Step 5: By mapping the 27 states upon the simple 13 relations b, a, d, c, o, oi, m, mi, s, si, f, fi, we receive another 13 equations. For brevity, we give only one example:

$$(c)^S = (c,oi,si)^S + (c,fi,o)^S + (c)^S + (o,oi,d,s,f,c,si,fi,e)^S + (a,c,mi,oi,si)^S + (b,c,fi,m,o)^S + T^S$$

Step 6: Now, a compound relation between two intervals can be represented as a 13-bit word, when we assign the first bit to B1 or B2, the second bit to A1 or A2, etc., according to the enumeration sequence prescribed by Allen's transitivity table. E.g.: (A (B,C) B) & (B (C,OI) C) yields the words "0101000000000" for R1 and "0001010000000" for R2. By realizing the above boolean equations in hardware, we receive a hardware transitivity table (the TIU). The TIU of this particular, thirteen-valued logic has two input words of 13 bits and one output word of 13 bits. The union of simple relations (as described in step 2) is performed by the drafted hardware in parallel, that is, in constant time.

Step 7: Steps 4 and 5 can be summarized in one step, where the 27 "intermediate" states are NOT computed. Instead the 13 equations of step 5 are computed directly from the 13 x 13 = 169 table entries of the transitivity table. Example:

(f) = e1f2 + fi1f2 + fle2 + f1fi2 + f1f2 + sild2 + s1oi2 + mils2
+ milo2 + mild2 + mlmi2 + oils2 + oilo2 + oild2 + oloi2
+ c1d2 + dlsi2 + dloi2 + dlc2 + als2 + alm2 + alo2 + als2
+ alb2 + bla2

Step 8: -- cf. example 1 --

Appendix : in the appendix to this paper, the complete set of boolean equations for Allen's temporal logic is given. A "system 1" describes step 7, a "system 2" describes steps 4 and 5.

8. Literature

[Allen 1983] J. F. Allen, Maintaining Knowledge about Temporal Intervals; in: Comm. ACM, Vol. 26 No. 11, November 1983, pp. 832 - 843.

[Allen 1986] J. F. Allen, Towards a General Theory of Action and Time; in: Artificial Intelligence, Vol 23., pp. 123 - 154.

[Becker 1988] S. U. Becker, Konzeption und Implementation einer Temporalen Inferenzkomponente -- Anwendung auf Planungsprobleme in der Fließproduktion, Philips Forschungslaboratorium Hamburg, laboratory report no. 722/88.

[Huber, A., Becker, S., 1988] Production planning using a temporal planning component. Proceedings European Conference on Artificial Intelligence, Munich 1988, pp. 188-190.

[Rit 1986] J.-F. Rit, Propagating Temporal Constraints for Scheduling. Proc. Conf. American Assoc. for Artificial Intelligence 1986, pp. 383 - 388.

9. Appendices

Transitivity table for thirteen-valued temporal logic (Allen's temporal logic)

"System 1" of Example 2

"System 2" of Example 2

Appendix A:

Transitivity table for Allen's thirteen-valued temporal logic

B r2 C A r1 B	<	>	d	di	o	oi	m	mi	s	si	f	fi
"before" <	<	no info	< o m d s	<	<	< o m d s	<	< o m d s	<	<	< o m d s	<
"after" >	no info	>	> oi mi d f	>	> oi mi d f	>	> oi mi d f	>	> oi mi d f	>	>	>
"during" d	<	>	d	no info	< o m d s	> oi mi d f	<	>	d	> oi mi d f	d	< o m d s
"contains" di	< o m di fi	> oi di mi si	o oi dur con =	di	o di fi	oi di si	o di fi	oi di si	di fi o	di	di si oi	di
"overlaps" o	<	> oi di mi si	o d s	< o m di fi	< o m	o oi dur con =	<	oi di si	o	di fi o	d s o	< o m
"over-lapped-by" oi	< o m di fi	>	oi d f	> oi mi di si	o oi dur con =	> oi mi	o di fi	>	oi d f	oi > mi	oi	oi di si
"meets" m	<	> oi mi di si	o d s	<	<	o d s	<	f fi =	m	m	d s o	<
"met-by" mi	< o m di fi	>	oi d f	>	oi d f	>	s si =	>	d f oi	>	mi	mi
"starts" s	<	>	d	< o m di fi	< o m	oi d f	<	mi	s	s si =	d	< m o
"started by" si	< o m di fi	>	oi d f	di	o di fi	oi	o di fi	mi	s si =	si	oi	di
"finishes" f	<	>	d	> oi mi di si	o d s	> oi mi	m	>	d	> oi mi	f	f fi =
"finished-by" fi	<	> oi mi di si	o d s	di	o	oi di si	m	si oi di	o	di	f fi =	fi

Appendix B:

"System 1" of Example 2

(Note: & is logical AND,

is logical OR)

```
BEFORE =
                    ( BEFORE1 &          BEFORE2 )  #
                    ( BEFORE1 &           AFTER2 )  #
                    ( BEFORE1 &          DURING2 )  #
                    ( BEFORE1 &        CONTAINS2 )  #
                    ( BEFORE1 &        OVERLAPS2 )  #
                    ( BEFORE1 &   OVERLAPPED-BY2 )  #
                    ( BEFORE1 &          MEETS2 )  #
                    ( BEFORE1 &         MET-BY2 )  #
                    ( BEFORE1 &         STARTS2 )  #
                    ( BEFORE1 &      STARTED-BY2 )  #
                    ( BEFORE1 &        FINISHES2 )  #
                    ( BEFORE1 &     FINISHED-BY2 )  #
                    ( BEFORE1 &         EQUALS2 )  #
                    ( AFTER1 &         BEFORE2 )  #
                    ( DURING1 &         BEFORE2 )  #
                    ( DURING1 &       CONTAINS2 )  #
                    ( DURING1 &       OVERLAPS2 )  #
                    ( DURING1 &         MEETS2 )  #
                    ( DURING1 &     FINISHED-BY2 )  #
                    ( CONTAINS1 &         BEFORE2 )  #
                    ( OVERLAPS1 &         BEFORE2 )  #
                    ( OVERLAPS1 &       CONTAINS2 )  #
                    ( OVERLAPS1 &       OVERLAPS2 )  #
                    ( OVERLAPS1 &         MEETS2 )  #
                    ( OVERLAPS1 &     FINISHED-BY2 )  #
                    ( OVERLAPPED-BY1 &          BEFORE2 )  #
                    ( MEETS1 &          BEFORE2 )  #
                    ( MEETS1 &        CONTAINS2 )  #
                    ( MEETS1 &        OVERLAPS2 )  #
                    ( MEETS1 &          MEETS2 )  #
                    ( MEETS1 &     FINISHED-BY2 )  #
                    ( MET-BY1 &         BEFORE2 )  #
                    ( STARTS1 &         BEFORE2 )  #
                    ( STARTS1 &       CONTAINS2 )  #
                    ( STARTS1 &       OVERLAPS2 )  #
                    ( STARTS1 &         MEETS2 )  #
                    ( STARTS1 &     FINISHED-BY2 )  #
                    ( STARTED-BY1 &          BEFORE2 )  #
                    ( FINISHES1 &         BEFORE2 )  #
                    ( FINISHED-BY1 &          BEFORE2 )  #
                    ( EQUALS1 &         BEFORE2 ) ;

AFTER =
                    ( BEFORE1 &          AFTER2 )  #
                    ( AFTER1 &         BEFORE2 )  #
                    ( AFTER1 &          AFTER2 )  #
                    ( AFTER1 &         DURING2 )  #
                    ( AFTER1 &       CONTAINS2 )  #
                    ( AFTER1 &       OVERLAPS2 )  #
                    ( AFTER1 &  OVERLAPPED-BY2 )  #
                    ( AFTER1 &         MEETS2 )  #
                    ( AFTER1 &        MET-BY2 )  #
                    ( AFTER1 &        STARTS2 )  #
                    ( AFTER1 &     STARTED-BY2 )  #
                    ( AFTER1 &       FINISHES2 )  #
                    ( AFTER1 &    FINISHED-BY2 )  #
                    ( AFTER1 &        EQUALS2 )  #
                    ( DURING1 &         AFTER2 )  #
                    ( DURING1 &       CONTAINS2 )  #
                    ( DURING1 &  OVERLAPPED-BY2 )  #
                    ( DURING1 &         MET-BY2 )  #
                    ( DURING1 &      STARTED-BY2 )  #
                    ( CONTAINS1 &          AFTER2 )  #
```

```
                ( OVERLAPS1 &          AFTER2 )  #
                ( OVERLAPPED-BY1 &          AFTER2 )
                ( OVERLAPPED-BY1 &        CONTAINS2 )    #
                ( OVERLAPPED-BY1 &  OVERLAPPED-BY2 )    #
                ( OVERLAPPED-BY1 &         MET-BY2 )    #
                ( OVERLAPPED-BY1 &     STARTED-BY2 )    #
                ( MEETS1 &          AFTER2 )  #
                ( MET-BY1 &          AFTER2 )  #
                ( MET-BY1 &        CONTAINS2 )  #
                ( MET-BY1 &  OVERLAPPED-BY2 )  #
                ( MET-BY1 &         MET-BY2 )  #
                ( MET-BY1 &     STARTED-BY2 )  #
                ( STARTS1 &          AFTER2 )  #
                ( STARTED-BY1 &          AFTER2 )   #
                ( FINISHES1 &          AFTER2 )  #
                ( FINISHES1 &        CONTAINS2 )  #
                ( FINISHES1 &  OVERLAPPED-BY2 )  #
                ( FINISHES1 &         MET-BY2 )  #
                ( FINISHES1 &     STARTED-BY2 )  #
                ( FINISHED-BY1 &          AFTER2 )    #
                ( EQUALS1 &          AFTER2 ) ;

DURING =
                ( BEFORE1 &          AFTER2 )  #
                ( BEFORE1 &         DURING2 )  #
                ( BEFORE1 &  OVERLAPPED-BY2 )  #
                ( BEFORE1 &         MET-BY2 )  #
                ( BEFORE1 &       FINISHES2 )  #
                ( AFTER1 &         BEFORE2 )  #
                ( AFTER1 &         DURING2 )  #
                ( AFTER1 &        OVERLAPS2 )  #
                ( AFTER1 &          MEETS2 )  #
                ( AFTER1 &         STARTS2 )  #
                ( DURING1 &         DURING2 )  #
                ( DURING1 &       CONTAINS2 )  #
                ( DURING1 &        OVERLAPS2 )  #
                ( DURING1 &  OVERLAPPED-BY2 )  #
                ( DURING1 &         STARTS2 )  #
                ( DURING1 &     STARTED-BY2 )  #
                ( DURING1 &       FINISHES2 )  #
                ( DURING1 &    FINISHED-BY2 )  #
                ( DURING1 &         EQUALS2 )  #
                ( CONTAINS1 &         DURING2 )  #
                ( OVERLAPS1 &         DURING2 )  #
                ( OVERLAPS1 &  OVERLAPPED-BY2 )  #
                ( OVERLAPS1 &       FINISHES2 )  #
                ( OVERLAPPED-BY1 &          DURING2 )    #
                ( OVERLAPPED-BY1 &        OVERLAPS2 )    #
                ( OVERLAPPED-BY1 &         STARTS2 )    #
                ( MEETS1 &          DURING2 )  #
                ( MEETS1 &  OVERLAPPED-BY2 )  #
                ( MEETS1 &        FINISHES2 )  #
                ( MET-BY1 &         DURING2 )  #
                ( MET-BY1 &        OVERLAPS2 )  #
                ( MET-BY1 &         STARTS2 )  #
                ( STARTS1 &         DURING2 )  #
                ( STARTS1 &  OVERLAPPED-BY2 )  #
                ( STARTS1 &        FINISHES2 )  #
                ( STARTED-BY1 &          DURING2 )   #
                ( FINISHES1 &         DURING2 )  #
                ( FINISHES1 &        OVERLAPS2 )  #
                ( FINISHES1 &         STARTS2 )  #
                ( FINISHED-BY1 &          DURING2 )    #
                ( EQUALS1 &          DURING2 ) ;
```

```
CONTAINS =
                    ( BEFORE1 &              AFTER2 )  #
                    ( AFTER1 &             BEFORE2 )  #
                    ( DURING1 &            CONTAINS2 )  #
                    ( CONTAINS1 &              BEFORE2 )  #
                    ( CONTAINS1 &              AFTER2 )  #
                    ( CONTAINS1 &              DURING2 )  #
                    ( CONTAINS1 &            CONTAINS2 )  #
                    ( CONTAINS1 &            OVERLAPS2 )  #
                    ( CONTAINS1 &     OVERLAPPED-BY2 )  #
                    ( CONTAINS1 &              MEETS2 )  #
                    ( CONTAINS1 &             MET-BY2 )  #
                    ( CONTAINS1 &              STARTS2 )  #
                    ( CONTAINS1 &         STARTED-BY2 )  #
                    ( CONTAINS1 &          FINISHES2 )  #
                    ( CONTAINS1 &       FINISHED-BY2 )  #
                    ( CONTAINS1 &              EQUALS2 )  #
                    ( OVERLAPS1 &              AFTER2 )  #
                    ( OVERLAPS1 &            CONTAINS2 )  #
                    ( OVERLAPS1 &     OVERLAPPED-BY2 )  #
                    ( OVERLAPS1 &             MET-BY2 )  #
                    ( OVERLAPS1 &         STARTED-BY2 )  #
                    ( OVERLAPPED-BY1 &              BEFORE2 )  #
                    ( OVERLAPPED-BY1 &            CONTAINS2 )  #
                    ( OVERLAPPED-BY1 &            OVERLAPS2 )  #
                    ( OVERLAPPED-BY1 &              MEETS2 )  #
                    ( OVERLAPPED-BY1 &       FINISHED-BY2 )  #
                    ( MEETS1 &               AFTER2 )  #
                    ( MET-BY1 &              BEFORE2 )  #
                    ( STARTS1 &            CONTAINS2 )  #
                    ( STARTED-BY1 &              BEFORE2 )  #
                    ( STARTED-BY1 &            CONTAINS2 )  #
                    ( STARTED-BY1 &            OVERLAPS2 )  #
                    ( STARTED-BY1 &              MEETS2 )  #
                    ( STARTED-BY1 &       FINISHED-BY2 )  #
                    ( FINISHES1 &            CONTAINS2 )  #
                    ( FINISHED-BY1 &              AFTER2 )  #
                    ( FINISHED-BY1 &            CONTAINS2 )  #
                    ( FINISHED-BY1 &     OVERLAPPED-BY2 )  #
                    ( FINISHED-BY1 &              MET-BY2 )  #
                    ( FINISHED-BY1 &         STARTED-BY2 )  #
                    ( EQUALS1 &              CONTAINS2 ) ;

OVERLAPS =
                    ( BEFORE1 &              AFTER2 )  #
                    ( BEFORE1 &              DURING2 )  #
                    ( BEFORE1 &     OVERLAPPED-BY2 )  #
                    ( BEFORE1 &              MET-BY2 )  #
                    ( BEFORE1 &            FINISHES2 )  #
                    ( AFTER1 &             BEFORE2 )  #
                    ( DURING1 &            CONTAINS2 )  #
                    ( DURING1 &            OVERLAPS2 )  #
                    ( DURING1 &       FINISHED-BY2 )  #
                    ( CONTAINS1 &              BEFORE2 )  #
                    ( CONTAINS1 &              DURING2 )  #
                    ( CONTAINS1 &            OVERLAPS2 )  #
                    ( CONTAINS1 &              MEETS2 )  #
                    ( CONTAINS1 &              STARTS2 )  #
                    ( OVERLAPS1 &              DURING2 )  #
                    ( OVERLAPS1 &            CONTAINS2 )  #
                    ( OVERLAPS1 &            OVERLAPS2 )  #
                    ( OVERLAPS1 &     OVERLAPPED-BY2 )  #
                    ( OVERLAPS1 &              STARTS2 )  #
```

```
                    ( OVERLAPS1 &       STARTED-BY2 )  #
                    ( OVERLAPS1 &         FINISHES2 )  #
                    ( OVERLAPS1 &      FINISHED-BY2 )  #
                    ( OVERLAPS1 &           EQUALS2 )  #
                    ( OVERLAPPED-BY1 &          BEFORE2 )  #
                    ( OVERLAPPED-BY1 &        OVERLAPS2 )  #
                    ( OVERLAPPED-BY1 &           MEETS2 )  #
                    ( MEETS1 &            DURING2 )  #
                    ( MEETS1 &    OVERLAPPED-BY2 )  #
                    ( MEETS1 &         FINISHES2 )  #
                    ( MET-BY1 &          BEFORE2 )  #
                    ( STARTS1 &         CONTAINS2 )  #
                    ( STARTS1 &         OVERLAPS2 )  #
                    ( STARTS1 &      FINISHED-BY2 )  #
                    ( STARTED-BY1 &           BEFORE2 )  #
                    ( STARTED-BY1 &         OVERLAPS2 )  #
                    ( STARTED-BY1 &           MEETS2 )  #
                    ( FINISHES1 &         OVERLAPS2 )  #
                    ( FINISHED-BY1 &          DURING2 )  #
                    ( FINISHED-BY1 &        OVERLAPS2 )  #
                    ( FINISHED-BY1 &          STARTS2 )  #
                    ( EQUALS1 &         OVERLAPS2 ) ;

OVERLAPPED-BY =
                    ( BEFORE1 &           AFTER2 )  #
                    ( AFTER1 &          BEFORE2 )  #
                    ( AFTER1 &          DURING2 )  #
                    ( AFTER1 &         OVERLAPS2 )  #
                    ( AFTER1 &           MEETS2 )  #
                    ( AFTER1 &          STARTS2 )  #
                    ( DURING1 &        CONTAINS2 )  #
                    ( DURING1 &    OVERLAPPED-BY2 )  #
                    ( DURING1 &       STARTED-BY2 )  #
                    ( CONTAINS1 &           AFTER2 )  #
                    ( CONTAINS1 &          DURING2 )  #
                    ( CONTAINS1 &    OVERLAPPED-BY2 )  #
                    ( CONTAINS1 &           MET-BY2 )  #
                    ( CONTAINS1 &         FINISHES2 )  #
                    ( OVERLAPS1 &           AFTER2 )  #
                    ( OVERLAPS1 &    OVERLAPPED-BY2 )  #
                    ( OVERLAPS1 &           MET-BY2 )  #
                    ( OVERLAPPED-BY1 &          DURING2 )  #
                    ( OVERLAPPED-BY1 &        CONTAINS2 )  #
                    ( OVERLAPPED-BY1 &        OVERLAPS2 )  #
                    ( OVERLAPPED-BY1 &    OVERLAPPED-BY2 )  #
                    ( OVERLAPPED-BY1 &          STARTS2 )  #
                    ( OVERLAPPED-BY1 &       STARTED-BY2 )  #
                    ( OVERLAPPED-BY1 &         FINISHES2 )  #
                    ( OVERLAPPED-BY1 &      FINISHED-BY2 )  #
                    ( OVERLAPPED-BY1 &          EQUALS2 )  #
                    ( MEETS1 &           AFTER2 )  #
                    ( MET-BY1 &          DURING2 )  #
                    ( MET-BY1 &        OVERLAPS2 )  #
                    ( MET-BY1 &          STARTS2 )  #
                    ( STARTS1 &    OVERLAPPED-BY2 )  #
                    ( STARTED-BY1 &          DURING2 )  #
                    ( STARTED-BY1 &    OVERLAPPED-BY2 )  #
                    ( STARTED-BY1 &         FINISHES2 )  #
                    ( FINISHES1 &        CONTAINS2 )  #
                    ( FINISHES1 &    OVERLAPPED-BY2 )  #
                    ( FINISHES1 &       STARTED-BY2 )  #
                    ( FINISHED-BY1 &          AFTER2 )  #
                    ( FINISHED-BY1 &    OVERLAPPED-BY2 )  #
                    ( FINISHED-BY1 &          MET-BY2 )  #
```

```
                    ( EQUALS1 &  OVERLAPPED-BY2 ) ;

MEETS =
                    ( BEFORE1 &           AFTER2 )  #
                    ( BEFORE1 &          DURING2 )  #
                    ( BEFORE1 &   OVERLAPPED-BY2 )  #
                    ( BEFORE1 &          MET-BY2 )  #
                    ( BEFORE1 &        FINISHES2 )  #
                    ( AFTER1 &          BEFORE2 )  #
                    ( DURING1 &        CONTAINS2 )  #
                    ( DURING1 &        OVERLAPS2 )  #
                    ( DURING1 &    FINISHED-BY2 )  #
                    ( CONTAINS1 &         BEFORE2 )  #
                    ( OVERLAPS1 &       CONTAINS2 )  #
                    ( OVERLAPS1 &       OVERLAPS2 )  #
                    ( OVERLAPS1 &    FINISHED-BY2 )  #
                    ( OVERLAPPED-BY1 &          BEFORE2 )  #
                    ( MEETS1 &         STARTS2 )  #
                    ( MEETS1 &     STARTED-BY2 )  #
                    ( MEETS1 &         EQUALS2 )  #
                    ( MET-BY1 &         BEFORE2 )  #
                    ( STARTS1 &       CONTAINS2 )  #
                    ( STARTS1 &       OVERLAPS2 )  #
                    ( STARTS1 &    FINISHED-BY2 )  #
                    ( STARTED-BY1 &          BEFORE2 )  #
                    ( FINISHES1 &          MEETS2 )  #
                    ( FINISHED-BY1 &          MEETS2 )  #
                    ( EQUALS1 &          MEETS2 ) ;

MET-BY =
                    ( BEFORE1 &           AFTER2 )  #
                    ( AFTER1 &         BEFORE2 )  #
                    ( AFTER1 &         DURING2 )  #
                    ( AFTER1 &       OVERLAPS2 )  #
                    ( AFTER1 &         MEETS2 )  #
                    ( AFTER1 &         STARTS2 )  #
                    ( DURING1 &       CONTAINS2 )  #
                    ( DURING1 &  OVERLAPPED-BY2 )  #
                    ( DURING1 &     STARTED-BY2 )  #
                    ( CONTAINS1 &          AFTER2 )  #
                    ( OVERLAPS1 &          AFTER2 )  #
                    ( OVERLAPPED-BY1 &        CONTAINS2 )  #
                    ( OVERLAPPED-BY1 &  OVERLAPPED-BY2 )  #
                    ( OVERLAPPED-BY1 &     STARTED-BY2 )  #
                    ( MEETS1 &          AFTER2 )  #
                    ( MET-BY1 &       FINISHES2 )  #
                    ( MET-BY1 &    FINISHED-BY2 )  #
                    ( MET-BY1 &         EQUALS2 )  #
                    ( STARTS1 &         MET-BY2 )  #
                    ( STARTED-BY1 &          MET-BY2 )  #
                    ( FINISHES1 &       CONTAINS2 )  #
                    ( FINISHES1 &  OVERLAPPED-BY2 )  #
                    ( FINISHES1 &     STARTED-BY2 )  #
                    ( FINISHED-BY1 &          AFTER2 )  #
                    ( EQUALS1 &         MET-BY2 ) ;

STARTS =
                    ( BEFORE1 &           AFTER2 )  #
                    ( BEFORE1 &          DURING2 )  #
                    ( BEFORE1 &   OVERLAPPED-BY2 )  #
                    ( BEFORE1 &         MET-BY2 )  #
                    ( BEFORE1 &       FINISHES2 )  #
                    ( AFTER1 &         BEFORE2 )  #
                    ( DURING1 &       CONTAINS2 )  #
```

```
                    ( DURING1 &          OVERLAPS2 )  #
                    ( DURING1 &      FINISHED-BY2 )  #
                    ( CONTAINS1 &           DURING2 )  #
                    ( OVERLAPS1 &           DURING2 )  #
                    ( OVERLAPS1 &    OVERLAPPED-BY2 )  #
                    ( OVERLAPS1 &         FINISHES2 )  #
                    ( OVERLAPPED-BY1 &         OVERLAPS2 )  #
                    ( MEETS1 &           DURING2 )  #
                    ( MEETS1 &    OVERLAPPED-BY2 )  #
                    ( MEETS1 &         FINISHES2 )  #
                    ( MET-BY1 &           MEETS2 )  #
                    ( STARTS1 &          STARTS2 )  #
                    ( STARTS1 &       STARTED-BY2 )  #
                    ( STARTS1 &          EQUALS2 )  #
                    ( STARTED-BY1 &           STARTS2 )  #
                    ( FINISHES1 &         OVERLAPS2 )  #
                    ( FINISHED-BY1 &           DURING2 )  #
                    ( EQUALS1 &          STARTS2 ) ;

STARTED-BY =

                    ( BEFORE1 &           AFTER2 )  #
                    ( AFTER1 &          BEFORE2 )  #
                    ( DURING1 &         CONTAINS2 )  #
                    ( CONTAINS1 &           AFTER2 )  #
                    ( CONTAINS1 &          DURING2 )  #
                    ( CONTAINS1 &    OVERLAPPED-BY2 )  #
                    ( CONTAINS1 &          MET-BY2 )  #
                    ( CONTAINS1 &        FINISHES2 )  #
                    ( OVERLAPS1 &           AFTER2 )  #
                    ( OVERLAPS1 &    OVERLAPPED-BY2 )  #
                    ( OVERLAPS1 &          MET-BY2 )  #
                    ( OVERLAPPED-BY1 &        CONTAINS2 )  #
                    ( OVERLAPPED-BY1 &         OVERLAPS2 )  #
                    ( OVERLAPPED-BY1 &      FINISHED-BY2 )  #
                    ( MEETS1 &           AFTER2 )  #
                    ( MET-BY1 &          MEETS2 )  #
                    ( STARTS1 &       STARTED-BY2 )  #
                    ( STARTED-BY1 &          STARTS2 )  #
                    ( STARTED-BY1 &       STARTED-BY2 )  #
                    ( STARTED-BY1 &          EQUALS2 )  #
                    ( FINISHES1 &         CONTAINS2 )  #
                    ( FINISHED-BY1 &           AFTER2 )  #
                    ( FINISHED-BY1 &    OVERLAPPED-BY2 )  #
                    ( FINISHED-BY1 &          MET-BY2 )  #
                    ( EQUALS1 &       STARTED-BY2 ) ;

FINISHES =

                    ( BEFORE1 &           AFTER2 )  #
                    ( AFTER1 &          BEFORE2 )  #
                    ( AFTER1 &          DURING2 )  #
                    ( AFTER1 &         OVERLAPS2 )  #
                    ( AFTER1 &           MEETS2 )  #
                    ( AFTER1 &          STARTS2 )  #
                    ( DURING1 &         CONTAINS2 )  #
                    ( DURING1 &    OVERLAPPED-BY2 )  #
                    ( DURING1 &       STARTED-BY2 )  #
                    ( CONTAINS1 &          DURING2 )  #
                    ( OVERLAPS1 &    OVERLAPPED-BY2 )  #
                    ( OVERLAPPED-BY1 &          DURING2 )  #
                    ( OVERLAPPED-BY1 &         OVERLAPS2 )  #
                    ( OVERLAPPED-BY1 &          STARTS2 )  #
                    ( MEETS1 &          MET-BY2 )  #
                    ( MET-BY1 &          DURING2 )  #
                    ( MET-BY1 &         OVERLAPS2 )  #
```

```
                    ( MET-BY1 &          STARTS2 )  #
                    ( STARTS1 &   OVERLAPPED-BY2 )  #
                    ( STARTED-BY1 &        DURING2 )  #
                    ( FINISHES1 &       FINISHES2 )  #
                    ( FINISHES1 &    FINISHED-BY2 )  #
                    ( FINISHES1 &         EQUALS2 )  #
                    ( FINISHED-BY1 &     FINISHES2 )  #
                    ( EQUALS1 &       FINISHES2 ) ;

FINISHED-BY =
                    ( BEFORE1 &         AFTER2 )  #
                    ( AFTER1 &        BEFORE2 )  #
                    ( DURING1 &       CONTAINS2 )  #
                    ( CONTAINS1 &        BEFORE2 )  #
                    ( CONTAINS1 &        DURING2 )  #
                    ( CONTAINS1 &      OVERLAPS2 )  #
                    ( CONTAINS1 &        MEETS2 )  #
                    ( CONTAINS1 &        STARTS2 )  #
                    ( OVERLAPS1 &      CONTAINS2 )  #
                    ( OVERLAPS1 &  OVERLAPPED-BY2 )  #
                    ( OVERLAPS1 &    STARTED-BY2 )  #
                    ( OVERLAPPED-BY1 &         BEFORE2 )  #
                    ( OVERLAPPED-BY1 &       OVERLAPS2 )  #
                    ( OVERLAPPED-BY1 &         MEETS2 )  #
                    ( MEETS1 &         MET-BY2 )  #
                    ( MET-BY1 &        BEFORE2 )  #
                    ( STARTS1 &      CONTAINS2 )  #
                    ( STARTED-BY1 &        BEFORE2 )  #
                    ( STARTED-BY1 &      OVERLAPS2 )  #
                    ( STARTED-BY1 &        MEETS2 )  #
                    ( FINISHES1 &    FINISHED-BY2 )  #
                    ( FINISHED-BY1 &      FINISHES2 )  #
                    ( FINISHED-BY1 &    FINISHED-BY2 )  #
                    ( FINISHED-BY1 &        EQUALS2 )  #
                    ( EQUALS1 &    FINISHED-BY2 ) ;

      EQUALS =
                    ( BEFORE1 &         AFTER2 )  #
                    ( AFTER1 &        BEFORE2 )  #
                    ( DURING1 &       CONTAINS2 )  #
                    ( CONTAINS1 &        DURING2 )  #
                    ( OVERLAPS1 &  OVERLAPPED-BY2 )  #
                    ( OVERLAPPED-BY1 &       OVERLAPS2 )  #
                    ( MEETS1 &         MET-BY2 )  #
                    ( MET-BY1 &        MEETS2 )  #
                    ( STARTS1 &      STARTED-BY2 )  #
                    ( STARTED-BY1 &        STARTS2 )  #
```

Appendix C:

"System 2" of Example 2

(Note: & is logical AND,
is logical OR)

```
STATE1 =
                    ( BEFORE1 &          BEFORE2 )  #
                    ( BEFORE1 &        CONTAINS2 )  #
                    ( BEFORE1 &        OVERLAPS2 )  #
                    ( BEFORE1 &          MEETS2 )  #
                    ( BEFORE1 &         STARTS2 )  #
                    ( BEFORE1 &    STARTED-BY2 )  #
                    ( BEFORE1 &   FINISHED-BY2 )  #
                    ( BEFORE1 &         EQUALS2 )  #
                    ( DURING1 &         BEFORE2 )  #
                    ( DURING1 &          MEETS2 )  #
                    ( OVERLAPS1 &         BEFORE2 )  #
                    ( OVERLAPS1 &          MEETS2 )  #
                    ( MEETS1 &        BEFORE2 )  #
                    ( MEETS1 &      CONTAINS2 )  #
                    ( MEETS1 &      OVERLAPS2 )  #
                    ( MEETS1 &        MEETS2 )  #
                    ( MEETS1 &   FINISHED-BY2 )  #
                    ( STARTS1 &         BEFORE2 )  #
                    ( STARTS1 &          MEETS2 )  #
                    ( FINISHES1 &         BEFORE2 )  #
                    ( FINISHED-BY1 &         BEFORE2 )  #
                    ( EQUALS1 &         BEFORE2 ) ;

STATE2 =
                    ( AFTER1 &          AFTER2 )  #
                    ( AFTER1 &      CONTAINS2 )  #
                    ( AFTER1 &  OVERLAPPED-BY2 )  #
                    ( AFTER1 &        MET-BY2 )  #
                    ( AFTER1 &    STARTED-BY2 )  #
                    ( AFTER1 &      FINISHES2 )  #
                    ( AFTER1 &   FINISHED-BY2 )  #
                    ( AFTER1 &         EQUALS2 )  #
                    ( DURING1 &          AFTER2 )  #
                    ( DURING1 &        MET-BY2 )  #
                    ( OVERLAPPED-BY1 &          AFTER2 )  #
                    ( OVERLAPPED-BY1 &        MET-BY2 )  #
                    ( MET-BY1 &          AFTER2 )  #
                    ( MET-BY1 &      CONTAINS2 )  #
                    ( MET-BY1 &  OVERLAPPED-BY2 )  #
                    ( MET-BY1 &        MET-BY2 )  #
                    ( MET-BY1 &    STARTED-BY2 )  #
                    ( STARTS1 &         AFTER2 )  #
                    ( STARTED-BY1 &         AFTER2 )  #
                    ( FINISHES1 &        AFTER2 )  #
                    ( FINISHES1 &        MET-BY2 )  #
                    ( EQUALS1 &         AFTER2 ) ;

STATE3 =
                    ( DURING1 &         DURING2 )  #
                    ( DURING1 &         STARTS2 )  #
                    ( DURING1 &       FINISHES2 )  #
                    ( DURING1 &         EQUALS2 )  #
                    ( STARTS1 &         DURING2 )  #
                    ( STARTS1 &       FINISHES2 )  #
                    ( FINISHES1 &         DURING2 )  #
                    ( FINISHES1 &         STARTS2 )  #
                    ( EQUALS1 &         DURING2 ) ;

STATE4 =
                    ( CONTAINS1 &       CONTAINS2 )  #
                    ( CONTAINS1 &    STARTED-BY2 )  #
                    ( CONTAINS1 &   FINISHED-BY2 )  #
                    ( CONTAINS1 &         EQUALS2 )  #
```

```
                    ( STARTED-BY1 &        CONTAINS2 )  #
                    ( STARTED-BY1 &     FINISHED-BY2 )  #
                    ( FINISHED-BY1 &       CONTAINS2 )  #
                    ( FINISHED-BY1 &     STARTED-BY2 )  #
                    ( EQUALS1 &        CONTAINS2 ) ;

STATE5 =
                    ( OVERLAPS1 &          STARTS2 )  #
                    ( OVERLAPS1 &          EQUALS2 )  #
                    ( FINISHED-BY1 &       OVERLAPS2 )  #
                    ( FINISHED-BY1 &         STARTS2 )  #
                    ( EQUALS1 &        OVERLAPS2 ) ;

STATE6 =
                    ( OVERLAPPED-BY1 &       FINISHES2 )  #
                    ( OVERLAPPED-BY1 &         EQUALS2 )  #
                    ( STARTED-BY1 &  OVERLAPPED-BY2 )  #
                    ( STARTED-BY1 &       FINISHES2 )  #
                    ( EQUALS1 &  OVERLAPPED-BY2 ) ;

STATE7 =
                    ( MEETS1 &          STARTS2 )  #
                    ( MEETS1 &      STARTED-BY2 )  #
                    ( MEETS1 &          EQUALS2 )  #
                    ( FINISHES1 &         MEETS2 )  #
                    ( FINISHED-BY1 &        MEETS2 )  #
                    ( EQUALS1 &         MEETS2 ) ;

STATE8 =
                    ( MET-BY1 &       FINISHES2 )  #
                    ( MET-BY1 &    FINISHED-BY2 )  #
                    ( MET-BY1 &         EQUALS2 )  #
                    ( STARTS1 &         MET-BY2 )  #
                    ( STARTED-BY1 &       MET-BY2 )  #
                    ( EQUALS1 &         MET-BY2 ) ;

STATE9 =
                    ( STARTS1 &         STARTS2 )  #
                    ( STARTS1 &         EQUALS2 )  #
                    ( EQUALS1 &         STARTS2 ) ;

STATE10 =
                    ( STARTED-BY1 &     STARTED-BY2 )  #
                    ( STARTED-BY1 &       EQUALS2 )  #
                    ( EQUALS1 &    STARTED-BY2 ) ;

STATE11 =
                    ( FINISHES1 &       FINISHES2 )  #
                    ( FINISHES1 &       EQUALS2 )  #
                    ( EQUALS1 &      FINISHES2 ) ;

STATE12 =
                    ( FINISHED-BY1 &    FINISHED-BY2 )  #
                    ( FINISHED-BY1 &      EQUALS2 )  #
                    ( EQUALS1 &    FINISHED-BY2 ) ;

STATE13 =
                    ( EQUALS1 &         EQUALS2 ) ;

STATE14 =
                    ( BEFORE1 &         AFTER2 )  #
                    ( AFTER1 &        BEFORE2 )  #
                    ( DURING1 &       CONTAINS2 ) ;
```

```
STATE15 =
                    ( BEFORE1 &          DURING2 )  #
                    ( BEFORE1 &   OVERLAPPED-BY2 )  #
                    ( BEFORE1 &          MET-BY2 )  #
                    ( BEFORE1 &       FINISHES2 )  #
                    ( DURING1 &       OVERLAPS2 )  #
                    ( DURING1 &    FINISHED-BY2 ) ;

STATE16 =
                    ( AFTER1 &          DURING2 )  #
                    ( AFTER1 &        OVERLAPS2 )  #
                    ( AFTER1 &          MEETS2 )  #
                    ( AFTER1 &         STARTS2 )  #
                    ( DURING1 &  OVERLAPPED-BY2 )  #
                    ( DURING1 &     STARTED-BY2 ) ;

STATE17 =
                    ( CONTAINS1 &         BEFORE2 )  #
                    ( OVERLAPS1 &       CONTAINS2 )  #
                    ( OVERLAPPED-BY1 &          BEFORE2 )  #
                    ( MET-BY1 &         BEFORE2 )  #
                    ( STARTS1 &       CONTAINS2 )  #
                    ( STARTED-BY1 &         BEFORE2 ) ;

STATE18 =
                    ( CONTAINS1 &          AFTER2 )  #
                    ( OVERLAPS1 &          AFTER2 )  #
                    ( OVERLAPPED-BY1 &       CONTAINS2 )  #
                    ( MEETS1 &          AFTER2 )  #
                    ( FINISHES1 &       CONTAINS2 )  #
                    ( FINISHED-BY1 &          AFTER2 ) ;

STATE19 =
                    ( CONTAINS1 &         DURING2 )  #
                    ( OVERLAPS1 &  OVERLAPPED-BY2 )  #
                    ( OVERLAPPED-BY1 &       OVERLAPS2 ) ;

STATE20 =
                    ( CONTAINS1 &       OVERLAPS2 )  #
                    ( CONTAINS1 &         MEETS2 )  #
                    ( CONTAINS1 &        STARTS2 )  #
                    ( OVERLAPS1 &     STARTED-BY2 )  #
                    ( OVERLAPPED-BY1 &          MEETS2 )  #
                    ( STARTED-BY1 &       OVERLAPS2 )  #
                    ( STARTED-BY1 &         MEETS2 ) ;

STATE21 =
                    ( CONTAINS1 &  OVERLAPPED-BY2 )  #
                    ( CONTAINS1 &         MET-BY2 )  #
                    ( CONTAINS1 &       FINISHES2 )  #
                    ( OVERLAPS1 &         MET-BY2 )  #
                    ( OVERLAPPED-BY1 &    FINISHED-BY2 )  #
                    ( FINISHED-BY1 &  OVERLAPPED-BY2 )  #
                    ( FINISHED-BY1 &         MET-BY2 ) ;

STATE22 =
                    ( OVERLAPS1 &         DURING2 )  #
                    ( OVERLAPS1 &       FINISHES2 )  #
                    ( MEETS1 &          DURING2 )  #
                    ( MEETS1 &  OVERLAPPED-BY2 )  #
                    ( MEETS1 &       FINISHES2 )  #
                    ( FINISHES1 &       OVERLAPS2 )  #
                    ( FINISHED-BY1 &         DURING2 ) ;
```

```
STATE23 =
                ( OVERLAPS1 &          OVERLAPS2 )   #
                ( OVERLAPS1 &      FINISHED-BY2 )   #
                ( STARTS1 &         OVERLAPS2 )   #
                ( STARTS1 &     FINISHED-BY2 ) ;

STATE24 =
                ( OVERLAPPED-BY1 &           DURING2 )   #
                ( OVERLAPPED-BY1 &           STARTS2 )   #
                ( MET-BY1 &            DURING2 )   #
                ( MET-BY1 &          OVERLAPS2 )   #
                ( MET-BY1 &            STARTS2 )   #
                ( STARTS1 &   OVERLAPPED-BY2 )   #
                ( STARTED-BY1 &          DURING2 ) ;

STATE25 =
                ( OVERLAPPED-BY1 &   OVERLAPPED-BY2 )   #
                ( OVERLAPPED-BY1 &       STARTED-BY2 )   #
                ( FINISHES1 &   OVERLAPPED-BY2 )   #
                ( FINISHES1 &       STARTED-BY2 ) ;

STATE26 =
                ( MEETS1 &           MET-BY2 )   #
                ( FINISHES1 &     FINISHED-BY2 )   #
                ( FINISHED-BY1 &         FINISHES2 ) ;

STATE27 =
                ( MET-BY1 &            MEETS2 )   #
                ( STARTS1 &       STARTED-BY2 )   #
                ( STARTED-BY1 &          STARTS2 ) ;

BEFORE =
                STATE1 #
                STATE14 #
                STATE15 #
                STATE17 #
                STATE23 #
                STATE28;

AFTER =
                STATE2 #
                STATE14 #
                STATE16 #
                STATE18 #
                STATE25 #
                STATE29;

DURING =
                STATE3 #
                STATE14 #
                STATE15 #
                STATE16 #
                STATE19 #
                STATE22 #
                STATE24 #
                STATE28 #
                STATE29;

CONTAINS =
                STATE4 #
                STATE14 #
                STATE17 #
                STATE18 #
                STATE19 #
```

STATE20 #
STATE21 #
STATE28 #
STATE29;

OVERLAPS =

STATE5 #
STATE14 #
STATE15 #
STATE17 #
STATE19 #
STATE20 #
STATE22 #
STATE23 #
STATE28 #
STATE29;

OVERLAPPED-BY =

STATE6 #
STATE14 #
STATE16 #
STATE18 #
STATE19 #
STATE21 #
STATE24 #
STATE25 #
STATE28 #
STATE29;

MEETS =

STATE7 #
STATE14 #
STATE15 #
STATE17 #
STATE23 #
STATE28;

MET-BY =

STATE8 #
STATE14 #
STATE16 #
STATE18 #
STATE25 #
STATE29;

STARTS =

STATE9 #
STATE14 #
STATE15 #
STATE19 #
STATE22 #
STATE27 #
STATE28 #
STATE29;

STARTED-BY =

STATE10 #
STATE14 #
STATE18 #
STATE19 #
STATE21 #
STATE27 #
STATE28 #
STATE29;

```
   FINISHES =
                        STATE11 #
                        STATE14 #
                        STATE16 #
                        STATE19 #
                        STATE24 #
                        STATE26 #
                        STATE28 #
                        STATE29;

FINISHED-BY =
                        STATE12 #
                        STATE14 #
                        STATE17 #
                        STATE19 #
                        STATE20 #
                        STATE26 #
                        STATE28 #
                        STATE29;

      EQUALS =
                        STATE13 #
                        STATE14 #
                        STATE19 #
                        STATE26 #
                        STATE27 #
                        STATE28 #
                        STATE29;
```