An interactive multimedia learning environment for VLSI built with COSMOS

Marios C. Angelides*, Harry W. Agius

Department of Information Systems and Computing, Brunel University,

Uxbridge, Middlesex, UB8 3PH, UK

E-mail: {angelidesm, harryagius}@acm.org

Abstract: This paper presents Bigger Bits, an interactive multimedia learning environment that teaches students about VLSI within the context of computer electronics. The system was built with COSMOS (Content Oriented Semantic Modelling Overlay Scheme), which is a modelling scheme that we developed for enabling the semantic content of multimedia to be used within interactive systems.

Keywords: multimedia/hypermedia systems; interactive learning environments; intelligent tutoring systems.

---

* Corresponding author. Tel/fax: +44 1895 203393.

1. Multimedia for VLSI education

The teaching of VLSI (very large scale integration)[1], with its emphasis on both products and processes, can benefit enormously from the use of educational systems, particularly where such systems take advantage of the use of multimedia to enhance positively the teaching-learning experience. Exploiting multimedia, such as animated graphics, video, and audio, in engineering education goes back a long way (e.g. Bailey & Thornton, 1992/93; El-Sharkawy, 1993; Iksander et al., 1993) and the advantages are numerous (Ranky et al., 1997). In terms of VLSI education, multimedia offers significant benefits because it enables the various products and processes to be demonstrated 'in action', thereby affording better understanding on the part of the student. For example, photographic images may be used to demonstrate different applications of VLSI (e.g. memory chips, microprocessors) and video segments may be used to demonstrate the fabrication process for making the wafer of monolithic circuits. The use of text alone, while useful, often adds little value over the information provided in textbooks, whereas multimedia can provide a pragmatic alternative for real-life observation (as the above examples illustrate).

This paper presents the Bigger Bits system, which uses multimedia to teach VLSI within the context of a fictitious computer electronics manufacturer called Bigger Bits Ltd. Bigger Bits was built with COSMOS, which is a modelling scheme that we developed to enable the semantic content of multimedia to be used within interactive systems, and which has proved successful in the development of other educational systems.

[1] We use the term VLSI broadly to encompass all integration that is beyond LSI (large scale integration), including ULSI (ultra large scale integration).

This paper is now structured as follows. In Section 2, we present an overview of COSMOS. Section 3 then presents the Bigger Bits system and discusses how COSMOS was used to build its architecture. Section 4 closes the paper with some concluding remarks.

2. An overview of COSMOS

Our content-modelling scheme, COSMOS (Content Oriented Semantic Modelling Overlay Scheme) (Agius & Angelides, 1999a), has been used to develop a number of educational multimedia systems (Agius & Angelides, 1999b; Tong & Agius, 1999; Agius & Angelides, 2000). The *multimedia frame* (*m-frame*) forms the basis of COSMOS. The m-frame serves as the conceptual representational structure in which semantic multimedia content is modelled. As its name suggests, the m-frame is a 'slot-and-filler' type structure, consisting of a set of *perspectives* on the semantic content and a set of more specific *instances* of that content. Each m-frame includes an `mt` perspective to denote the m-frame type, an `st` perspective to denote the sub-type of the m-frame, and an `id` perspective to denote an identifier for the m-frame.

COSMOS uses three types of m-frames:

1. **Syntactic m-frames (*SYM*s):** A *SYM* models *spatial relationships between objects* for an individual frame of a time-based visual media segment (e.g. video or animated graphics). Thus, there is a strict one-to-one relationship between the frames of segments and the *SYM*s within COSMOS. Figure 1 illustrates the structure of a *SYM*. The `id` takes the form of the name of the video segment (`"videoname"`) followed by the frame number (denoted by a # in the figure). As the figure illustrates, a *SYM* has two perspectives: (1) `OBJECTS` which store the object names and their co-ordinates (the co-ordinates are a

series of points representing a polygon), and (2) SPATIALRELS to store the spatial relationships between the objects. We use the term 'object' to refer to any visible or hidden object depicted within a video frame at any necessary level of detail, e.g. it may be as entire as a chicken or as decomposed as a chicken's beak. Each spatial relationship (indicated by SR in Figure 1) is represented using one or more of the primitives detailed in Table 1.

[Figure 1 near here **]**

[Table 1 near here **]**

2. **Semantic m-frames (*SEM*s):** SEMs model *events* (occurrences within the media), *actions* (shorter sub-segments of the events), or *object properties* over an arbitrary number of frames of a visual and/or aural media segment (e.g. video, animated graphics, or audio). Figure 2 depicts the structure of these three *SEM*s, known as *Objects SEMs*, *Events SEMs*, and *Actions SEMs*. As the figure indicates, the three *SEM*s are treated together and represent the semantic content for a particular object. Shots are shown for the instances in the figure through the use of shot identifiers located in brackets after the instance. Multiple shots may be associated with each instance, and there are no restrictions on the number of times that the same shot may be used with the same *SEM* or other *SEM*s. In this way, *SEM*s may be used to reflect multiple perspectives on (i.e. different points of view of) the same semantic content. Non-content-based information may also be included within *SEM*s, and this is shown in the figure through sets of empty brackets next to the instances. The perspectives of the Objects and Events *SEM*s are able to be freely defined by the developer, and are not restricted to a pre-defined set, as with the *SYM*s. Thus, the

`SPECIALISATION OF` perspective given in the Objects *SEM* in Figure 2 is not obligatory, but has been given by way of an example as to how relationships may be defined between objects. However, the perspectives of Actions *SEM*s are indeed restricted, namely to the event names given in associated Events *SEM*s.

[Figure 2 near here **]**

3. **Temporal m-frames (*TEM*s):** *TEM*s model the *temporal relationships between events and actions* modelled in the *SEM*s. There are two types of *TEM*s: (1) an *Event TEM* models the temporal relationships of one event to all other events defined within COSMOS, (2) an *Action TEM* models the temporal relationships between one action within one event, and all other actions within the same event and all other events. COSMOS uses the 13 temporal relationship primitives derived by Allen (1983) within the *TEM*s, and these form the perspectives of the *TEM*. In this way, one Event (Action) *TEM* exists for each and every event (action) modelled within the Events (Actions) *SEM*s, and each perspective contains all the events (actions) that have the defined temporal relationship with the specified event (action). Allen's temporal relationship primitives are detailed in Table 2. The structure of an Action *TEM* is shown in Figure 3. The structure of the Event *TEM* is similar, but the instances of the `st` and `id` perspectives differ accordingly, and the instances for each of the temporal relationship perspectives are event rather than action identifiers.

[Table 2 near here **]**

[Figure 3 near here **]**

We have also developed a formal definition language (COSMOS-DL) and a formal query language (COSMOS-QL) to support the creation of the structure, population, and querying of the model (Agius & Angelides, 1999b). To reduce the need to use the languages directly, we have developed three front-end software tools that facilitate the creation of *SYM*s, *SEM*s, and *TEM*s, known as the SYMulator, SEMulator, and TEMulator respectively (Agius & Angelides, 2000). The relationship between these supplementary components, their relationship to COSMOS, and their relationship to the final developed interactive multimedia system is illustrated in Figure 4.

[Figure 4 near here **]**

In the following section, we present Bigger Bits, an interactive multimedia learning environment that we developed using our modelling scheme and which teaches VLSI within the context of computer electronics.

3. Bigger Bits: an interactive multimedia learning environment for VLSI

Bigger Bits is an interactive multimedia learning environment that was built with COSMOS. The system teaches about VLSI within the context of a fictitious computer electronics manufacturer called Bigger Bits Ltd. Our objective in developing the system was to provide an environment whereby a student would be able to see the application of the theoretical knowledge of product and process that they had learned in the classroom and through reading their textbooks. We therefore based our domain knowledge on information provided in textbooks like those of Streetman (1995) and Shur (1996) and that provided in more advanced handbooks such as that of Sze (1988). We then looked for media that exemplified

this knowledge, and drew on a variety of educational videos, footage from electronics and computer companies' promotional videos, and our own footage that we filmed especially for the system.

Bigger Bits uses an intelligent tutoring system (ITS) architecture, composed of domain, remedial, tutoring, and student knowledge (Tong & Angelides, 2000). COSMOS forms the domain knowledge of the system. However, since *SEM*s may also be used to represent non-content-based information, we use them to form the basis of the rest of the architecture, i.e. for the remedial, tutoring, and student knowledge. This is an approach that we have used in other multimedia ITSs that we have developed (Agius & Angelides, 1999b; Agius & Angelides, 1999a; Tong & Agius, 1999), since it yields a uniform, consistent, and flexible 'component-based' architecture, that may represent both procedural and non-procedural, and content-based and non-content-based information within a consistent framework.

Although the *SEM*s used in the remedial, tutoring, and student knowledge maintain the perspective and instances structure of their COSMOS counterparts, the nature of the perspectives and instances differs. Since the structure differs throughout the architecture, we discuss each type of *SEM* in the appropriate sub-sections that follow. We begin, however, by discussing the domain knowledge in the system.

3.1. Domain knowledge in Bigger Bits

The domain knowledge in Bigger Bits is COSMOS, as it was detailed in Section 2. Examples of the *SYM*s, *SEM*s, and *TEM*s that are used in the system are given in Figures 5-7. As is illustrated by Figure 5, the objects that are modelled in the *SYM*s are the detailed parts and sub-parts of computer electronic devices such as integrated circuits, transistors, CCDs (charge coupled devices), wafers, packages, package posts, leads connecting pads on chip

peripheries to posts, etc. We also use objects to delineate the functional areas of the microprocessor within the system's video footage of the architecture of the chip. In the case of the *SEM*s, we use objects to represent the computer electronic devices. These are related together through the use of a `PART OF` perspective. An example of this is given in Figure 6, whereby a microprocessor is part of a motherboard. The Objects *SEM*s model information regarding the properties of the various devices, while the Events and Actions *SEM*s model information concerned with how these devices are manufactured and how they operate and are connected with other devices. The *TEM*s therefore represent the temporal order of these events and actions.

[Figure 5 near here **]**

[Figure 6 near here **]**

[Figure 7 near here **]**

3.2. Remedial knowledge in Bigger Bits

The remedial knowledge in Bigger Bits consists of a set of *remedial SEMs*. These remedial *SEM*s contain remedial information that is used in conjunction with remedial strategies to remedy the student. The remedial *SEM*s mirror the structure and the relationships of the COSMOS *SEM*s (i.e. the *SEM*s constituting the domain knowledge). The major difference is that remedial information rather than shots are adjacent to the instances. In addition, the m-frame sub-type (`st`) includes an additional (`R`) to denote that it is a remedial *SEM*. The remedial information provided is intended to guide the student towards the remedial goals, and thus the tutoring goals, of the system. The set of remedial *SEM*s for an object are used in

conjunction with the *SEM*s for the same object within the domain knowledge so that specific misconceptions may be illustrated to the student through the use of suitable media segments. For example, if the student were having difficulty identifying a pin grid array (PGA) package, then the multimedia-based remedial strategies (contained in the tutoring knowledge, which is discussed in the next section) would show the student a video segment depicting a PGA as given in the package Object *SEM* in the domain knowledge. The set of remedial *SEM*s for the microprocessor object in Bigger Bits are shown in Figure 8.

[Figure 8 near here **]**

3.3. Tutoring knowledge in Bigger Bits

Bigger Bits' tutoring knowledge is divided into four main components, each of which has its own form of *SEM*s: (1) tutoring goals, (2) tutoring strategies, (3) remedial goals, and (4) remedial strategies. Portions of Bigger Bits tutoring knowledge are illustrated in Figure 9. This figure will be referred to in the discussion which follows.

[Figure 9 near here **]**

Tutoring goals. The tutoring goals determine what the individual student should be instructed on. Goal attainment by the student is achieved through the use of associated tutoring strategies. Bigger Bits has one *tutoring goals SEM* for every object within the domain knowledge. Figure 9(a) shows Bigger Bits' microprocessor tutoring goals *SEM*. As the figure illustrates, tutoring goals *SEM*s have an appropriate sub-type name, and the perspectives serve to specify the order in which the goals are to be achieved. Each goal within the *SEM* consists of the name of the perspective that the goal is concerned with, the

minimum number of instances that a student must name in order to satisfy the goal, and a number of tutoring strategies that may be used to try to achieve the goal (identified using a tutoring strategy identifier, e.g. 1, 2, M1, or M2, each of which refers to a particular tutoring strategy *SEM*). Where the perspective named for the goal is event-oriented, e.g. `MANUFACTURED BY`, the goal is achieved if the student names at least the number of events specified by the goal and all of the constituent actions of these events (an individual tutoring strategy is therefore used both for tutoring about the events and their actions).

Tutoring strategies. Each tutoring strategy encompasses one way in which a student may be taught a particular subset of knowledge. Tutoring strategies present material that will allow students to master a particular tutoring goal (teaching) while also evaluating the student's reaction to the instruction (testing). To this end, the use of tutoring strategies is guided by the student's individual needs as reflected by the student knowledge. Bigger Bits uses both non-multimedia-based and multimedia-based tutoring strategies. Multimedia-based tutoring strategies use media within the teaching-learning interaction and each tutoring strategy provides a different way in which a particular subset of knowledge may be taught appropriately using the media. For example, tutoring how and where a microprocessor is inserted onto the motherboard by showing a video of this. Most of the tutoring in Bigger Bits uses multimedia-based tutoring strategies. However, the additional inclusion of non-multimedia-based tutoring strategies enables the student to be taught when media cannot be used (e.g. because media segments are not available) or the use of media is unnecessary (e.g. it is sufficient to teach about the number of transistors on a microprocessor using text only). A tutoring strategy *SEM* has four perspectives: (1) `TEACHING TACTICS`, which provides the procedures for presenting the tutoring goal to the student; (2) `CHECK TEACHING TACTICS`, which provides the procedure for checking the teaching that took place previously for missing concepts and misconceptions; (3) `TESTING TACTICS`, which provides the

procedures for testing the student according to the tutorial goals; and (4) OPERATIONS, whose instances evaluate the responses of the student and model them as student knowledge. A portion of a tutoring strategy *SEM* from Bigger Bits is given in Figure 9(b). Bigger Bits has a number of different tutoring strategies including 'question and answer', 'multiple choice', and 'true or false'. These different tutoring strategies (through their use of the semantic content embodied in COSMOS) enable the system to use media in a variety of interactive ways — e.g. referring to objects by their relative spatial locations by exploiting the spatial relationships in the *SYM*s, highlighting events as they occur within the media by exploiting the temporal relationships in the *TEM*s — and enable the student to make their responses in a variety of ways — e.g. clicking on an object in the video using the mouse, typing answers into an input line, selecting from multiple-choice options. Figure 10 shows a tutoring strategy being used to teach (top screen shot) and test (bottom screen shot) the first tutoring goal that was shown in Figure 9(a).

[Figure 10 near here **]**

Remedial goals. The remedial goals are used to provide remedial assistance to the student with the aim of correcting students' misconceptions. There is one *remedial goals SEM* for each and every tutoring goals *SEM* within Bigger Bits. The microprocessor remedial goals *SEM* is depicted in Figure 9(c). As can be seen, the structure of a remedial goals *SEM* is similar to that of a tutoring goals *SEM*. Each remedial goal in a remedial goals *SEM* consists of a number of sub-goals, each of which has a number of associated remedial strategies, which are used to carry out the remediation. Typical remedial sub-goals are illustrated in Figure 9(c) and are explained as follows:

- `ME` remedies the student by using the remedial information in the associated remedial *SEM* in conjunction with the information tutored to the student initially,
- `SIMILAR` remedies the student by present similar objects, object properties, events, or actions, depending on the associated tutoring goal, to enable the student to make comparisons,
- `BLANKOUT` remedies the student by presenting the student with the information tutored initially in conjunction with a 'blanking out' of certain parts of the required answer(s) so that the student can fill in the blanks.

Remedial strategies. Remedial strategies reflect ways in which a student may be remedied against a particular tutoring goal. As with tutoring strategies, Bigger Bits uses both multimedia-based and non-multimedia-based remedial strategies. A *remedial strategy SEM* has only one perspective, `TACTICS`, which provides the procedures for presenting a remedial goal to the student. A portion of a remedial strategy *SEM* from Bigger Bits is given in Figure 9(d).

3.4. Student knowledge in Bigger Bits

The student knowledge provides valuable information during tutoring about the status of the student so that tutoring processes may be altered accordingly, i.e. changing the tutoring and/or remedial strategy to one that may prove more successful in tutoring the student. Unlike the domain and tutoring knowledge, the majority of the student knowledge is constructed during the course of the student's interaction with the system as student overlay knowledge of the domain knowledge and diagnosed student misconceptions. This knowledge is complemented by a bugs library.

Student overlay knowledge. This is a representation of the current status of the student in terms of the correct knowledge they have attained. It is represented as *student overlay SEMs*, whose structure and organisation mirrors that of the *SEM*s contained with the domain knowledge (i.e. COSMOS). The main difference between student overlay *SEM*s and domain knowledge *SEM*s is that the student overlay *SEM*s store additional information within their instances (a student overlay *SEM* also uses an `(S)` after the sub-type to denote that it models student knowledge). Each instance in a student overlay *SEM* records the instance name (which is equivalent to the correct answer given by the student during testing), the strength of the acquired knowledge during teaching and testing modes (rated between 0 and 1), the successful tutoring strategy used to elicit this response, and the media segments used if a multimedia-based tutoring strategy were used. Figure 11(a) shows a typical microprocessor student overlay Object *SEM* in Bigger Bits (the microprocessor Events and Actions *SEM*s are not shown in the figure, but would also exist within the student knowledge, assuming that the student had been taught and/or tested on the events and actions for the object).

[Figure 11 near here **]**

Student misconceptions. Student misconceptions are represented with *student misconception SEMs*. These are similar to the student overlay *SEM*s, except that the instances record the incorrect answers given by the student and the appended numbers indicate the seriousness of this bad knowledge (on a scale between -1 and 0).

Bugs library. The bugs library contains common student misconceptions about the domain. It is composed of *mal SEMs*, each of which records a number of common misconceptions for the equivalent *SEM* within the domain knowledge. Mal *SEM*s therefore

mirror the structure and organisation of the domain knowledge *SEM*s. This information is used during remediation to inform the student that their mistake is a common one and it is taken into account when judging the seriousness of the bad knowledge and when altering the tutoring and/or remedial strategies. Figure 11(b) depicts a portion of the microprocessor mal Object *SEM* from Bigger Bits (microprocessor Events and Actions mal *SEM*s also exist but are not shown in the figure). The sub-type of the *SEM* is assigned as `OBJECT (MAL)` to denote that this is an Object *SEM* that forms part of the bugs library. The perspectives mirror those of the domain knowledge Object *SEM*, but the instances differ since they reflect the common misconceptions.

4. Concluding remarks

This paper has presented the Bigger Bits interactive multimedia learning environment, which was built with COSMOS. The development of Bigger Bits, like our previous developments, serves two short-term purposes: (1) a testbed for experimentation on COSMOS, and (2) the use of COSMOS for the development of interactive multimedia environments in diverse domains. Our long-term goal is for end-users to develop their own applications using COSMOS, be they learning environments, business applications, medical systems, or any other applications that would benefit from having semantic multimedia content modelled using a scheme such as COSMOS.

5. References

Agius, H. W. & Angelides, M. C. (1999a). COSMOS – Content Oriented Semantic Modelling Overlay Scheme. *The Computer Journal, 42*(3), 153-176.

Agius, H. W. & Angelides, M. C. (1999b). Developing knowledge-based intelligent multimedia tutoring systems using semantic content-based modelling. *Artificial Intelligence Review*, *13*(1), 55-83.

Agius, H. W. & Angelides, M. C. (2000). A method for developing interactive multimedia from their semantic content. *Data & Knowledge Engineering*, *34*(2): 165-187.

Allen, J. F. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM*, *26*(11), November, 832-843.

Bailey, H. J. & Thornton, N. E. (1992/93). Interactive video: innovative episodes for enhancing education. *Computer Applications in Engineering Education*, *1*(1), 97-108.

El-Sharkawy, M. (1993). A multimedia laboratory. *Computer Applications in Engineering Education*, *1*(2), 129-140.

Iksander, M. F., Reed, T. & Breen, J., III (1993). Interactive video lessons for electromagnetic education. *Computer Applications in Engineering Education*, *1*(2), 147-158.

Ranky, P. G., Bengu, G. & Spak, G. T. (1997). The development and application of synchronous and asynchronous technology based learning aids for undergraduate engineering education. Paper presented at the *Engineering Education Innovators' Conference*, April 7-8, Arlington, VA (http://www.njit.edu/papers/EEIC/)

Shur, M. (1996). *Introduction to Electronic Devices*. Wiley, New York, NY.

Streetman, B. G. (1995). *Solid State Electronic Devices*, Fourth Edition. Prentice Hall, Englewood Cliffs, NJ.

Sze, S. M. (ed.) (1988). *VLSI Technology*, Second Edition. McGraw-Hill, New York, NY.

Tong, A. K. Y. & Agius, H. W. (1999). ARISTOTLE: a multimedia-based intelligent tutoring system for zoology. *Journal of Intelligent Systems*, *9*(2), 107-133.

Tong, A. K. Y. & Angelides, M. C. (2000). An empirical model for tutoring strategy selection in multimedia tutoring systems. *Decision Support Systems*, *29*(1), 31-45.

6. Vitae

Marios C. Angelides is Professor of Computing in the Department of Information Systems and Computing at Brunel University. He holds a BSc in Computing and a PhD in Information Systems both from The London School of Economics and Political Science where he began his academic career as a Lecturer in Information Systems. He has over 10 years of research experience in multimedia information systems where he has published extensively in journal and book format. He is the author of *Multimedia Information Systems* which was published by Kluwer. He is a member of the ACM, the IEEE Computer Society, and the British Computer Society. His home page is at *http://www.brunel.ac.uk/~csstmca/*.

Harry W. Agius is a Lecturer in Computing in the Department of Information Systems and Computing at Brunel University. His research interests lie in the area of digital media computing with a particular interest in digital media management and retrieval, which he has been actively researching and publishing in for a number of years. He holds a BSc in Computing and Information Systems (1994), an MSc in Analysis, Design and Management of Information Systems (1995), and a PhD in Information Systems (1997), all from the The London School of Economics and Political Science. He is a member of the ACM, the IEEE, the British Computer Society, and the UK Academy for Information Systems. He may be found on the Web at *http://www.brunel.ac.uk/~cssthwa/*.

Table captions

Table 1 Primitives for spatial relationships between objects.

Table 2 Allen's (1983) 13 temporal relationships.

Figure captions

Figure 1 Conceptual representation of the structure of a *SYM*.

Figure 2 Conceptual representation of the structure of an Object, an Events, and an Actions SEM.

Figure 3 Conceptual representation of the structure of a TEM.

Figure 4 Relationships between COSMOS, its supplementary components, and a COSMOS-based interactive multimedia system.

Figure 5 A portion of one of the *SYM*s from Bigger Bits.

Figure 6 Portions of the Object, Events, and Actions *SEM*s for the microprocessor object in Bigger Bits.

Figure 7 A portion of one of the *TEM*s from Bigger Bits for one of the microprocessor actions (only the actions from Figure 6 are shown).

Figure 8 A portion of the set of remedial *SEM*s for the microprocessor object in Bigger Bits.

Figure 9 A portion of the tutoring knowledge from Bigger Bits for the microprocessor object: (a) a tutoring goals *SEM*, (b) a tutoring strategy *SEM*; (c) a remedial goals *SEM*; (d) a remedial strategy *SEM*.

Figure 10 Teaching (top) and testing (bottom) in progress in Bigger Bits.

Figure 11 A portion of the student knowledge from Bigger Bits for the microprocessor object: (a) a student overlay Object *SEM*, (b) a mal Object *SEM*.

Table 1

Regular Spatial Relation	Notation	Inverse Spatial Relation	Notation	
O1 *touches* O2	O1 = O2	O2 *touches* O1	O2 = O1	2-D spatial primitives
O1 *above* O2	O1 $\uparrow$ O2	O2 *beneath* O1	O2 $\downarrow$ O1	
O1 *left* O2	O1 < O2	O2 *right* O1	O2 > O1	
O1 *inside* O2	O1 $\subseteq$ O2	O2 *encapsulates* O1	O2 $\supseteq$ O1	3-D spatial primitives
O1 *before* O2	O1 $\Uparrow$ O2	O2 *behind* O1	O2 $\Downarrow$ O1	

Table 2

Temporal relation	Notation	Inverse notation	Conceptual example
A *before* B	<	>	AAA BBB
A *equal* B	=	=	AAA BBB
A *meets* B	m	mi	AAABBB
A *overlaps* B	o	oi	AAA BBB
A *during* B	d	di	AAA BBBBBB
A *starts* B	s	si	AAA BBBBB
A *finishes* B	f	fi	AAA BBBBB

```
mt:           SYM
id:           "videoname":#
OBJECTS:      object1 (x1,y1,x2,y2, ..., xn,yn)
              object2 (x1,y1,x2,y2, ..., xn,yn)
              ...
SPATIALRELS:  object1 SR object2
              ...
```

Figure 1

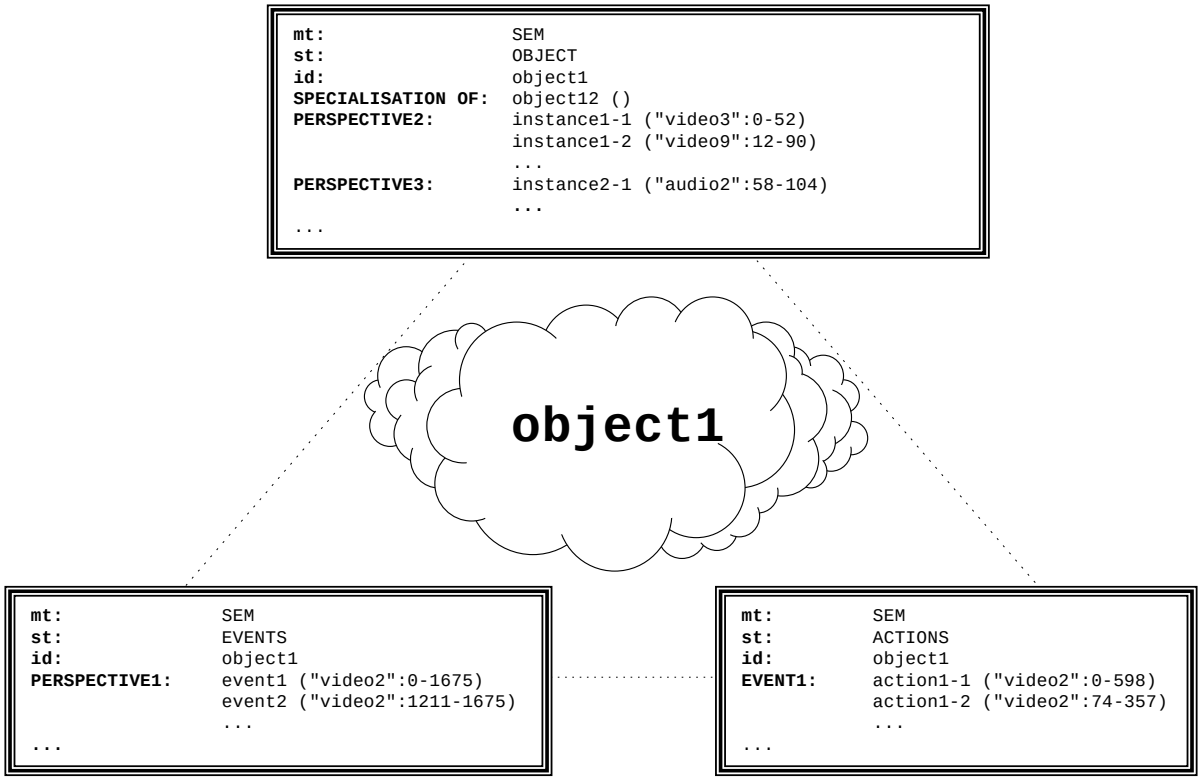
mt: SEM
st: OBJECT
id: object1
SPECIALISATION OF: object12 ()
PERSPECTIVE2: instance1-1 ("video3":0-52)
instance1-2 ("video9":12-90)
...
PERSPECTIVE3: instance2-1 ("audio2":58-104)
...
...
object1
mt: SEM
st: EVENTS
id: object1
PERSPECTIVE1: event1 ("video2":0-1675)
event2 ("video2":1211-1675)
...
...
mt: SEM
st: ACTIONS
id: object1
EVENT1: action1-1 ("video2":0-598)
action1-2 ("video2":74-357)
...
...

Figure 2

```
mt:       TEM
st:       ACTION
id:       object1:event1:action1-1
<:        ...
>:        ...
=:        ...
m:        ...
mi:       ...
o:        ...
oi:       ...
d:        object1:event1:action1-2
          ...
di:       ...
s:        ...
si:       ...
f:        ...
fi:       ...
```

Figure 3

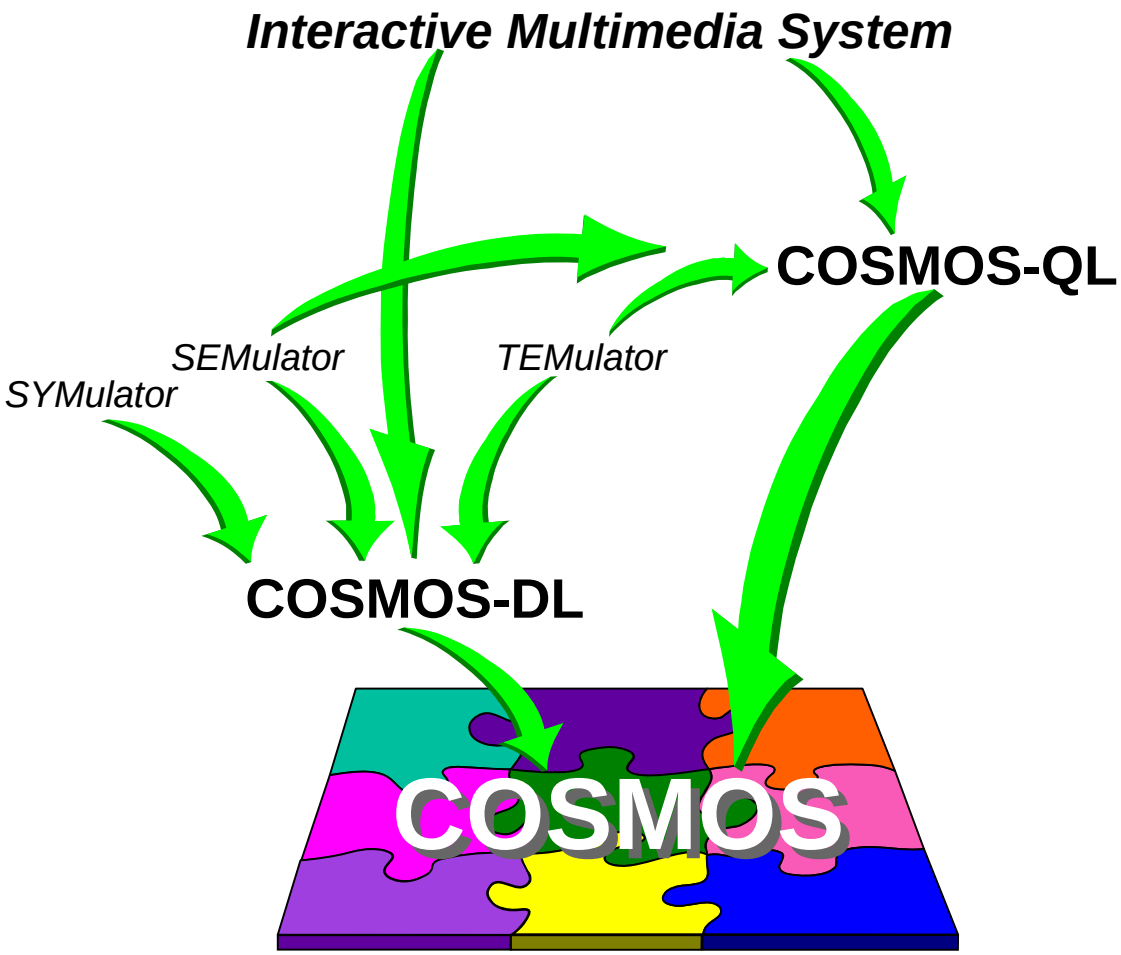
Interactive Multimedia System
COSMOS-QL
SEMulator
TEMulator
SYMulator
COSMOS-DL
COSMOS

Figure 4

```
mt:              SYM
id:              "assembly":7624
OBJECTS:         signal-trace-1 (44,13,52,13,52,48,44,48)
                 signal-trace-2 (63,13,71,13,71,48,63,48)
                 ...
                 package (20,4,20,240,305,240,305,4)
                 lead-1 (85,36,103,75)
                 ...
SPATIALRELS:     signal-trace-1 < signal-trace-2
                 signal-trace-1 ⇑= package
                 ...
```

Figure 5

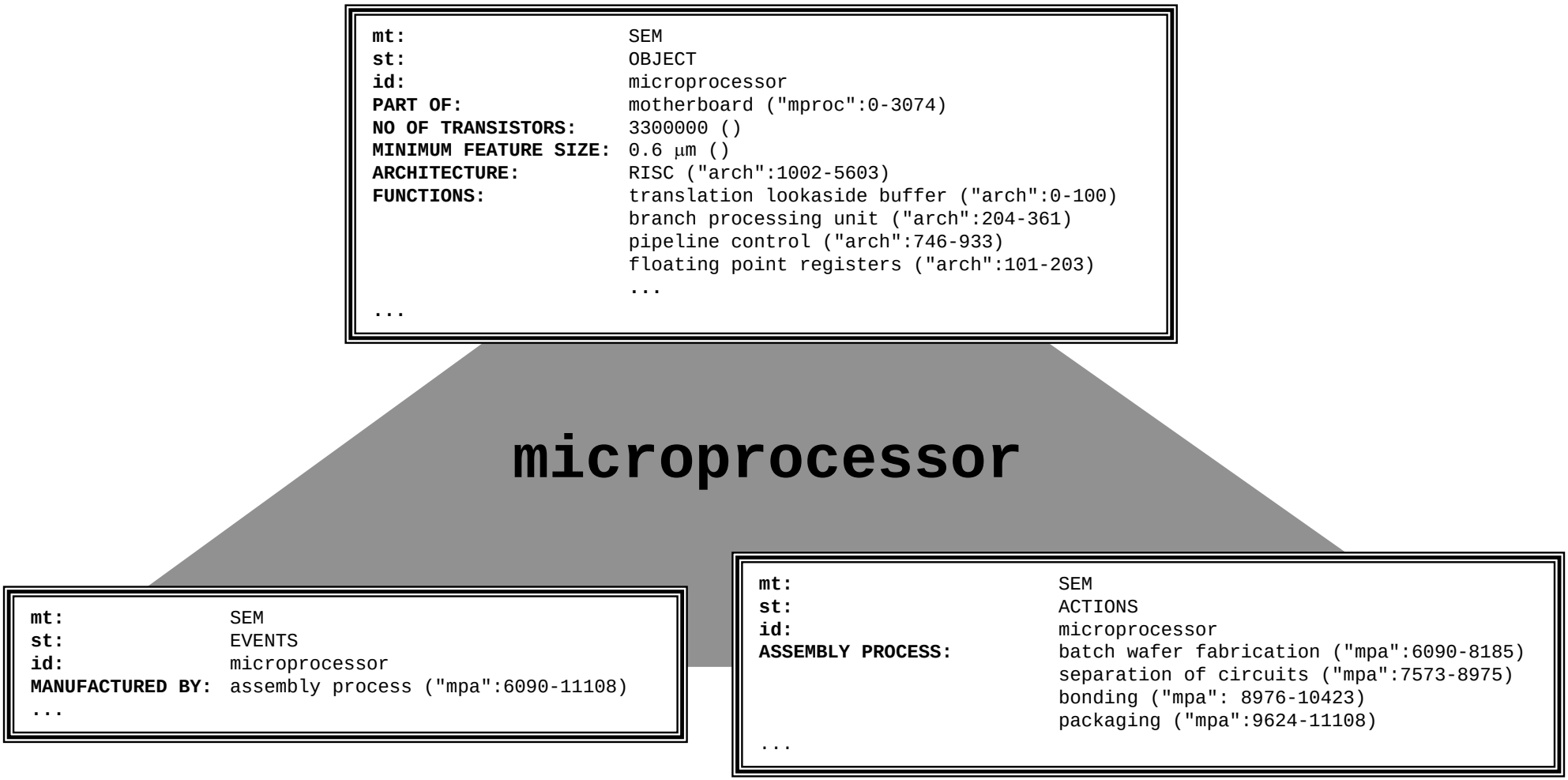
mt: SEM
st: OBJECT
id: microprocessor
PART OF: motherboard ("mproc":0-3074)
NO OF TRANSISTORS: 3300000 ()
MINIMUM FEATURE SIZE: 0.6 μm ()
ARCHITECTURE: RISC ("arch":1002-5603)
FUNCTIONS: translation lookaside buffer ("arch":0-100)
branch processing unit ("arch":204-361)
pipeline control ("arch":746-933)
floating point registers ("arch":101-203)
...
...
microprocessor
mt: SEM
st: EVENTS
id: microprocessor
MANUFACTURED BY: assembly process ("mpa":6090-11108)
...
mt: SEM
st: ACTIONS
id: microprocessor
ASSEMBLY PROCESS: batch wafer fabrication ("mpa":6090-8185)
separation of circuits ("mpa":7573-8975)
bonding ("mpa": 8976-10423)
packaging ("mpa":9624-11108)
...

Figure 6

```
mt:    TEM
st:    ACTION
id:    microprocessor:assembly process:separation of circuits
<:     microprocessor:assembly process:packaging
       ...
>:     ...
=:     ...
m:     microprocessor:assembly process:bonding
       ...
mi:    microprocessor:assembly process:batch wafer fabrication
       ...
o:     ...
oi:    ...
d:     ...
di:    ...
s:     ...
si:    ...
f:     ...
fi:    ...
```

Figure 7

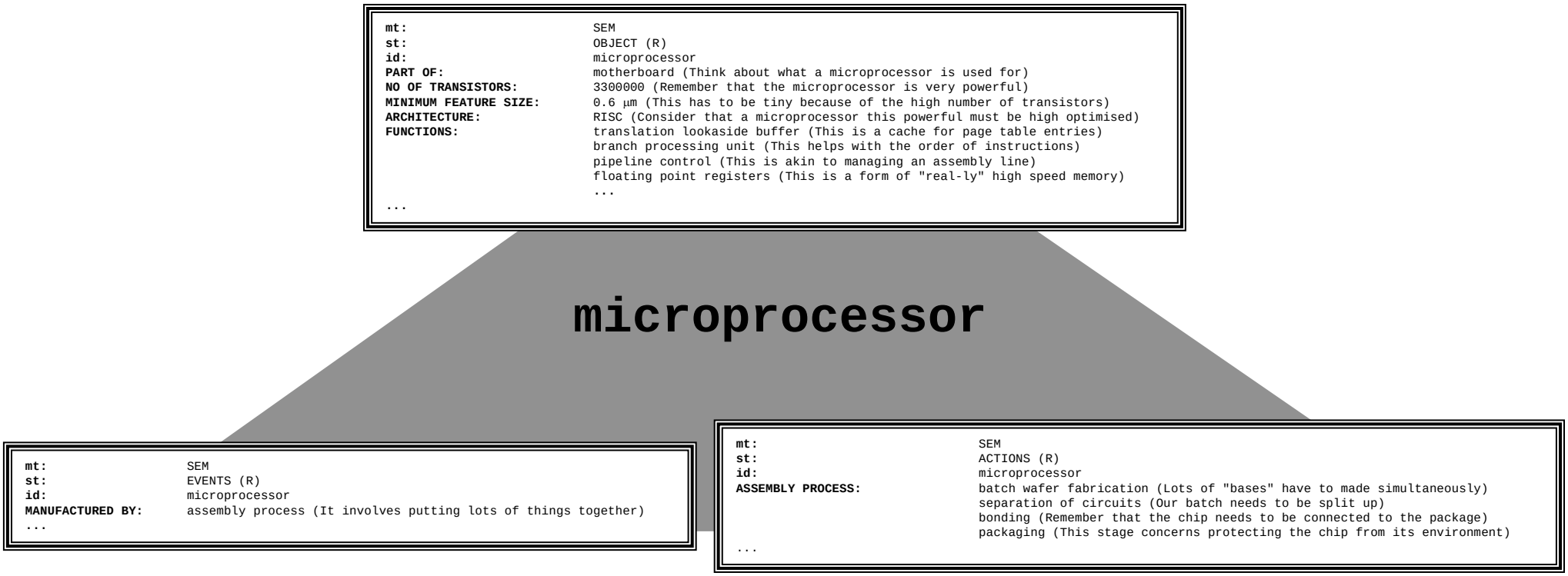
mt: SEM
st: OBJECT (R)
id: microprocessor
PART OF: motherboard (Think about what a microprocessor is used for)
NO OF TRANSISTORS: 3300000 (Remember that the microprocessor is very powerful)
MINIMUM FEATURE SIZE: 0.6 μm (This has to be tiny because of the high number of transistors)
ARCHITECTURE: RISC (Consider that a microprocessor this powerful must be high optimised)
FUNCTIONS: translation lookaside buffer (This is a cache for page table entries)
branch processing unit (This helps with the order of instructions)
pipeline control (This is akin to managing an assembly line)
floating point registers (This is a form of "real-ly" high speed memory)
...
...
microprocessor
mt: SEM
st: EVENTS (R)
id: microprocessor
MANUFACTURED BY: assembly process (It involves putting lots of things together)
...
mt: SEM
st: ACTIONS (R)
id: microprocessor
ASSEMBLY PROCESS: batch wafer fabrication (Lots of "bases" have to made simultaneously)
separation of circuits (Our batch needs to be split up)
bonding (Remember that the chip needs to be connected to the package)
packaging (This stage concerns protecting the chip from its environment)
...

Figure 8

(a)

```
mt:      SEM
st:      TUTORING GOALS
id:      microprocessor
GOAL 1:  PART OF 1 (M1,M2,M3)
GOAL 2:  NO OF TRANSISTORS 1 (1,2,3)
GOAL 3:  MINIMUM FEATURE SIZE 1 (1,2,3)
GOAL 4:  MANUFACTURED BY 1 (M1,M2,M3,M4)
GOAL 5:  ARCHITECTURE 1 (M1,M2,M3,M4)
GOAL 6:  FUNCTIONS 3 (M1,M2,M3,M4)
...
```

(b)

```
mt:                     SEM
st:                     TUTORING STRATEGY
id:                     M1
TEACHING TACTICS:       sCurrentInstanceNo of this book = 0
                        send doNextMultimediaInstance
CHECK TEACHING TACTICS: send askQuestion_Multimedia_1 FALSE
TESTING TACTICS:        send askQuestion_Multimedia_1 TRUE
OPERATIONS:             if pClicking = false then
                           send NonMultimediaBasedTutoringStrategy_1_Operations
                           else
                              sysCursor = 4
                              sNoOfRightAnswers of this book = 0
                              if ASYM_ItemOffset(sCurrentPerspective of this book,sEvents of this book) <> 0\
                                 OR ASYM_ItemOffset(sCurrentPerspective of this book,sObjectProps of this\
                                 book) <> 0 then
                                 clear sEventsNamed of this book
                              end if
                              increment sNoOfAttempts of this book
                              clear vSoundsLikeTextlineNos
                              clear vTooManyWordsTextlineNos
                              clear vWrongTextlineNos
                              clear vCorrectTextlineNos
                              clear vCorrectJTextlineNos
                              vWhatStudentShouldMatchTo = sCurrentValidInstances of this book
                              vActualStudentAnswer = whatStudentClickedOn (pFrameNo,pWhereClicked) of page\
                              "Domain Knowledge"
                        ...
```

(c)

```
mt:      SEM
st:      REMEDIAL GOALS
id:      microprocessor
GOAL 1:  ME (M1,M2)
         SIMILAR (1,2)
         BLANKOUT (1,2)
GOAL 2:  ME (1,2,3)
         SIMILAR (1,2,3)
         BLANKOUT (1,2,3)
...
```

(d)

```
mt:       SEM
st:       REMEDIAL STRATEGY
id:       M1
TACTICS:  sysCursor = 4
          clear vRemedialInfo
          conditions
             when sCurrentRemedialSubgoal of this book = "ME"
                vRemedialInfo = getMe()
                conditions
                   when sCurrentTutoringStrategy of this book = "1" OR sCurrentTutoringStrategy of this book = "M1"
                      vTextlineNoToUse = random(textlineCount(vRemedialInfo))
                      if (vTextlineNoToUse mod 2) = 0 then
                          decrement vTextlineNoToUse
                      end if
          ...
```

Figure 9

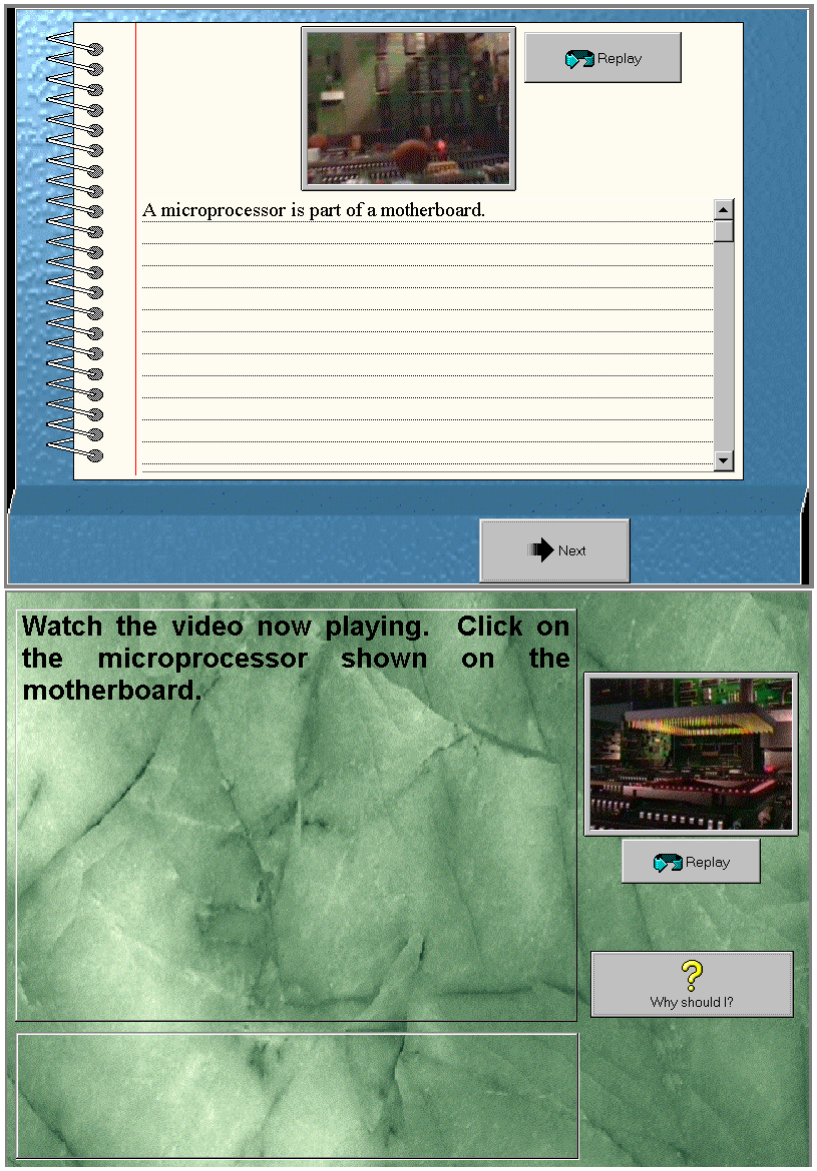
Replay
A microprocessor is part of a motherboard.
Next
Watch the video now playing. Click on the microprocessor shown on the motherboard.
Replay
Why should I?

Figure 10

(a)

```
mt:                     SEM
st:                     OBJECT (S)
id:                     microprocessor
PART OF:                motherboard TEACH 0.94 M3("mproc":0-3074)
NO OF TRANSISTORS:      3300000 TEACH 0.94 3()
MINIMUM FEATURE SIZE:   0.6 μm TEACH 0.94 3()
ARCHITECTURE:           RISC TEACH 0.94 M4("arch":1002-5603)
                        ...
...
```

(b)

```
mt:                     SEM
st:                     OBJECT (MAL)
id:                     microprocessor
PART OF:                IC
                        CPU
                        register
                        ...
NO OF TRANSISTORS:      <= 3000000
                        >= 3600000
...
```

Figure 11