Scicos et Modelica

Ramine Nikoukhah

C’est quoi Scicos ?

- **Editeur, simulateur et générateur du code pour les systèmes dynamiques hybrides**
- **Objectif : Utilisations industrielles mais aussi l’enseignement de l’automatique et la recherche**
- **Composant (Boîte à Outils) de Scilab**
  - **Inclus dans le package Scilab**
  - **Existe depuis 94 (premier release Scilab)**
  - **Financés par INRIA, des projets RNTL (Simpa, Metisse, Eclipse, Simpa2) et des contrats industriels (Renault, EDF)**

Sur quoi est-il basé ?

- **Un formalisme ouvert et documenté**
  - **Inspiré des langages synchrones**
    - **Extension à temps-continu**
  - **Permet la modélisation des systèmes hybrides par une intégration harmonieuse des composants temps-discrets, temps-continus et événementiels**
  - **Permet une gestion efficace de l'utilisation du solveur numérique**

- **Des solveurs numériques ODE et DAE**
  - **Lsodar et Daskr**
  - **Modifiés et interfacés au simulateur**

Qui sont les « autres » ?

- **Simulink**
  - **Produit Mathworks**
  - **Boîte à outils de Matlab**
  - **Quasi monopole**

- **SystemBuild**
  - **Initialement développé par Wind Rivers faisant partie du logiciel MATRIXx**
  - **Point faible : pas d'environnement de support comparable à Matlab**
  - **Acheté par Mathworks mais vendu sur une décision du DOJ (position du monopole)**
  - **Commercialisé par NI; intégration LabView**

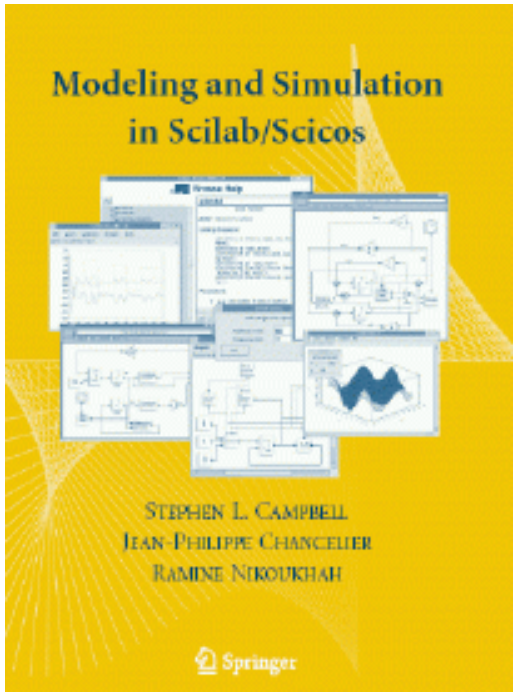
- **Dymola (Modelica)**
  - **Produit Dynasim**

- **Autres: plus spécialisés**

Ses composants

- **Editeur**
  - Ecrit en Scilab (code, IHM et graphique)
  - Facile à customiser
- **Compilateur**
  - Scilab et C
- **Simulateur**
  - C et Fortran (solveurs numériques)
- **Librairie des blocs**
  - Palettes de blocs élémentaires
  - Scilab (fonctions d'interfaçages) et C (fonctions de simulation, quelques vieux blocs en Fortran)
- **Compilateur Modelica**
  - Développé en Caml
- **Générateur du code C**

Etat actuel



- **Documentations disponibles**
  - Site Web www.scicos.org
  - Exemples, documentations
  - Aide en ligne
  - Livre
- **Formalisme de base**
  - Bien adapté pour les besoins actuels
  - Extension Modelica en cours de développement (Simpa, Simpa2, collaboration avec Peter Fritzen)
  - Permet de remplacer la plupart des schémas Simulink (sans Stateflow) et SystemBuild
- **Compilateur**
  - Code assez fiable refait en 2004
  - Algorithme plus efficace mais développé en partie dans Scilab => problème de vitesse pour très grand schéma

Etat actuel

- **Simulateur**
  - Code C **bien testé et efficace**
  - Extensions récentes pour l'introduction de Modelica
    - Modification et interfaçage de DASKR
    - Gestion de redémarrage de DAE
- **Générateur du code**
  - Code C monoprocesseur
    - Prise en compte du dynamique temps-continu (implantation par solveurs à pas fixe)
    - Génération pour Linux **RTAI** (R. Bucher)
  - Code **SynDEx** (RNTL ECLIPSE)
- **Editeur**
  - IHM style Windows en cours de développement

Scicos et Scilab

- **Scicos est une boîte à outils et fonctionne dans l'environnement Scilab.**

- **L'intégration Scicos/Scilab est importante pour fournir les fonctionnalités de Matlab/Simulink.**

- **Utilisation du langage Scilab pour batch processing**
  - **Post-traitement des résultats de simulation**
  - **Validation de modèle**
  - **Affichage graphique**

- **Utilisation des outils Scilab dans la construction de modèles :**
  - **Identification de modèle à partir de données numériques**
  - **Construction de filtres et de contrôleurs (automatique et traitement de signal)**

Développement de Scicos dans Scilab Avantages et Inconvénients

- **Editeur facilement adaptable : addition de menus et fonctionnalités,...**
- **Flexibilité dans la définition de forme et icones de blocs et liens (fonctions graphiques standard de Scilab)**
- **Facilité de développement et de débogage**
- **Portage de Scicos sur nouveaux systèmes avec Scilab**
- **Les structures de données des modèles Scicos sont des listes Scilab : facilité de manipulation et utilisation de fonctions Scilab pour l'interaction.**

- **IHM limité par IHM de Scilab.**
- **Scilab est un langage interprété : manipulation des schémas de très grande taille peut être lente.**

Scicos : Formalisme

Scicos fournit un environnement pour la construction des systèmes réactifs.

Les modèles Scicos sont construits en utilisant un éditeur schéma-blocs mais un langage déclaraltif sous-jacent existe basé sur un formalisme bien défini.

Le formalisme de base est simple car il ne traite que la partie réactive ; il ne s'agit pas d'un langage de programmation complet.

Les blocs sont des atomes dans Scicos : le simulateur les considère comme des boîtes noires. Seul, quelques propriétes sont utilisées par le compilateur.

Scicos : Formalisme

Le code réalisant le comportement du bloc (fonction de simulation) peut être du C, Fortran ou Scilab

L'exécution des fonctions de simulations est supposée instantanée : Scicos est un langage synchrone étendu au temps continu.

L'existence d'un temps unique et universel est supposée.

Le formalisme Scicos est très proche du langage Modelica (mélange continu-discret, notion d'événement, rémanence des variables,…)

Scicos : Bloc

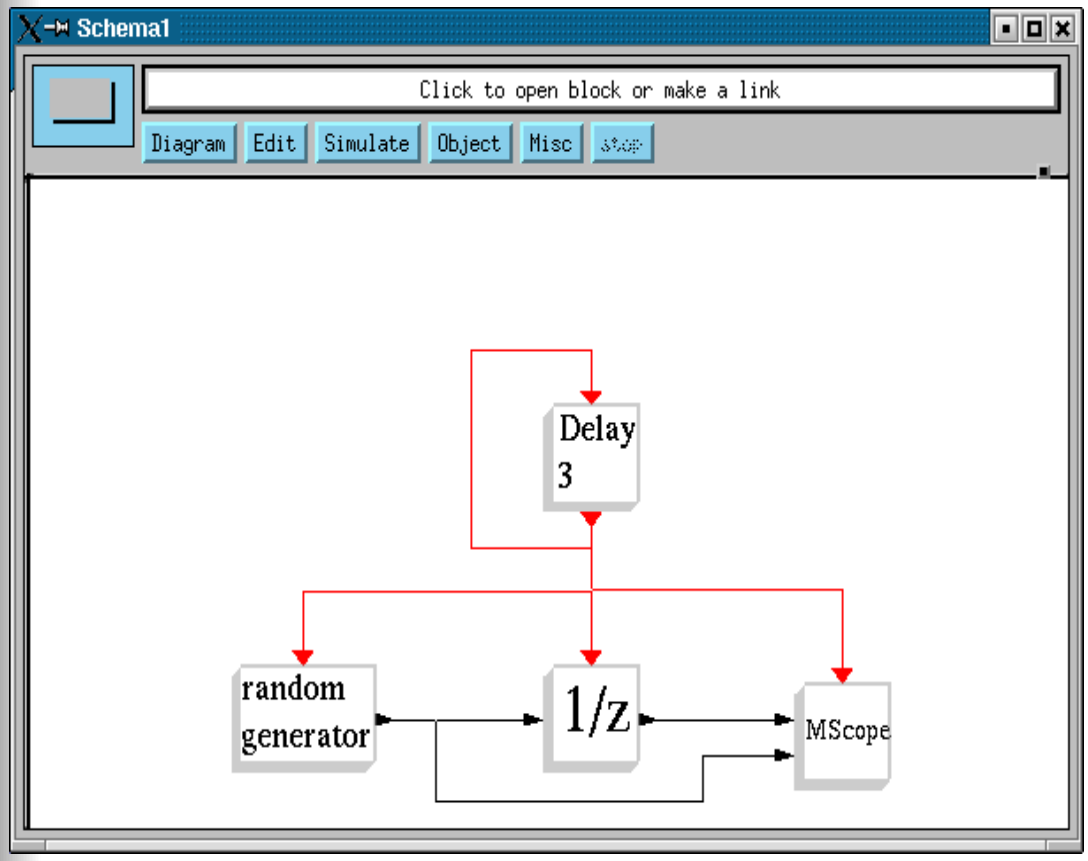
- **Bloc Scicos peut avoir deux types d'entrées et de sorties :**

  *** entrée régulière (souvent placée sur les cotés)**
  *** sortie régulière (aussi sur les cotés)**
  *** entrée d'activation (souvent en haut)**
  *** sortie d'activation (souvent en bas)**

- **Les entrées sorties régulières sont utilisées pour communiquer des données de bloc à bloc par des liens réguliers.**

- **Les entrées sorties d'activation connectées par des liens d'activation transmettent des informations de contrôle.**

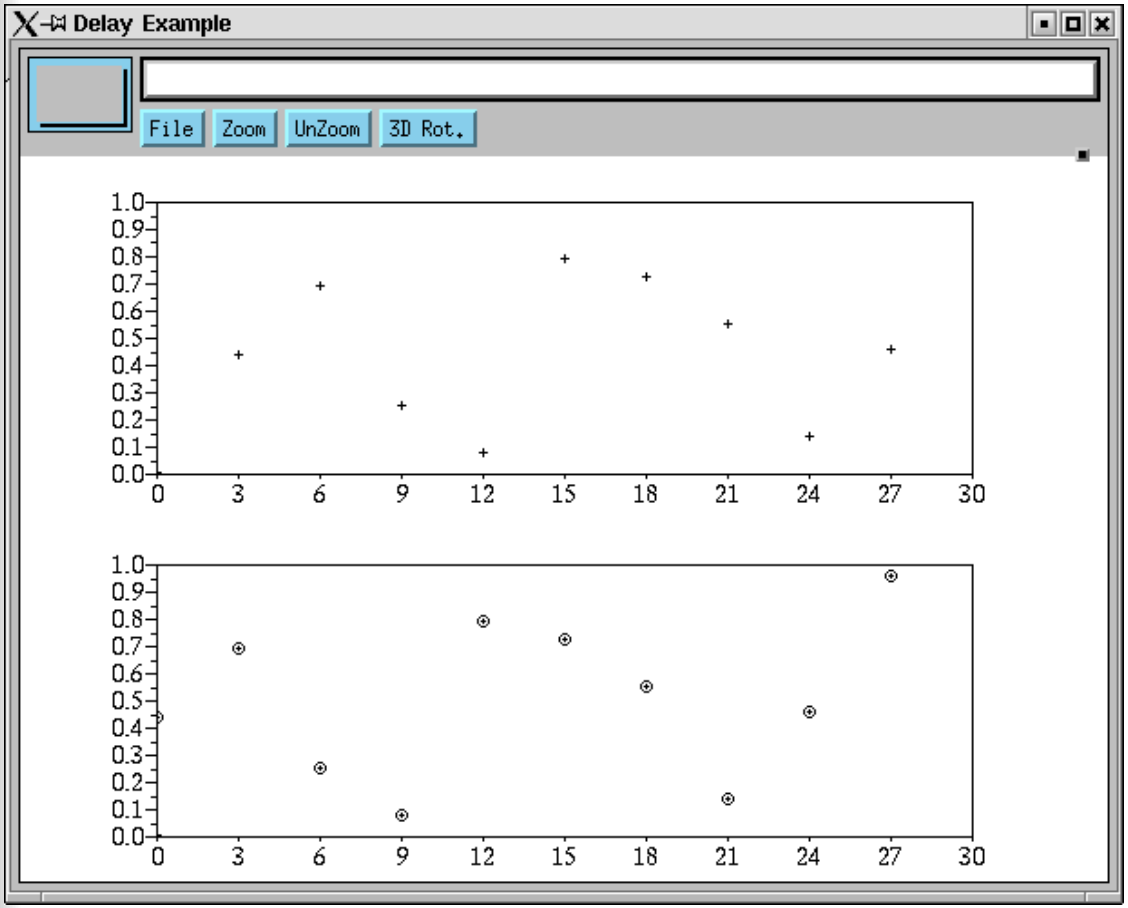
Exemple



Le bloc MScope affiche la sortie du générateur de séquence aléatoire et une version retardée.

Ce schéma contient une seule source d'activation. Tous les blocs sont donc activés de façon synchrone.

Simulation



Le deuxième signal est une réplique du premier avec un retard de 3

Sous-échantillonnage

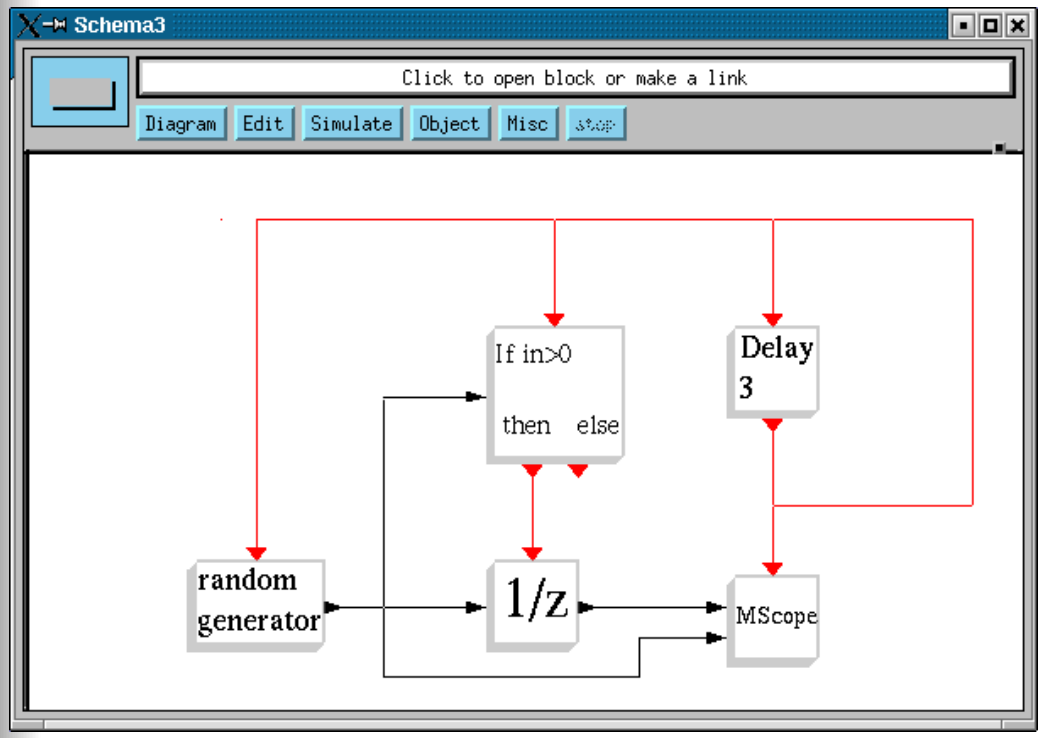
- **Deux blocs dans Scicos réalisent du sous-échantillonnage :**
  - **Bloc If-Then-Else**
  - **Bloc Eselect**

- **Pas des vrais blocs**
  - **une facilité d'édition**
  - **pas de fonction de simulation**
  - **gérés en phase de compilation**

- **Les sorties ne constituent pas des sources d'activation indépendantes : Evénements de sortie synchrones avec l'entrée.**

- **L'analogue (dans un contexte différent) du conditionnement en C de If-Then-Else et Switch-case.**

Sous-échantillonnage



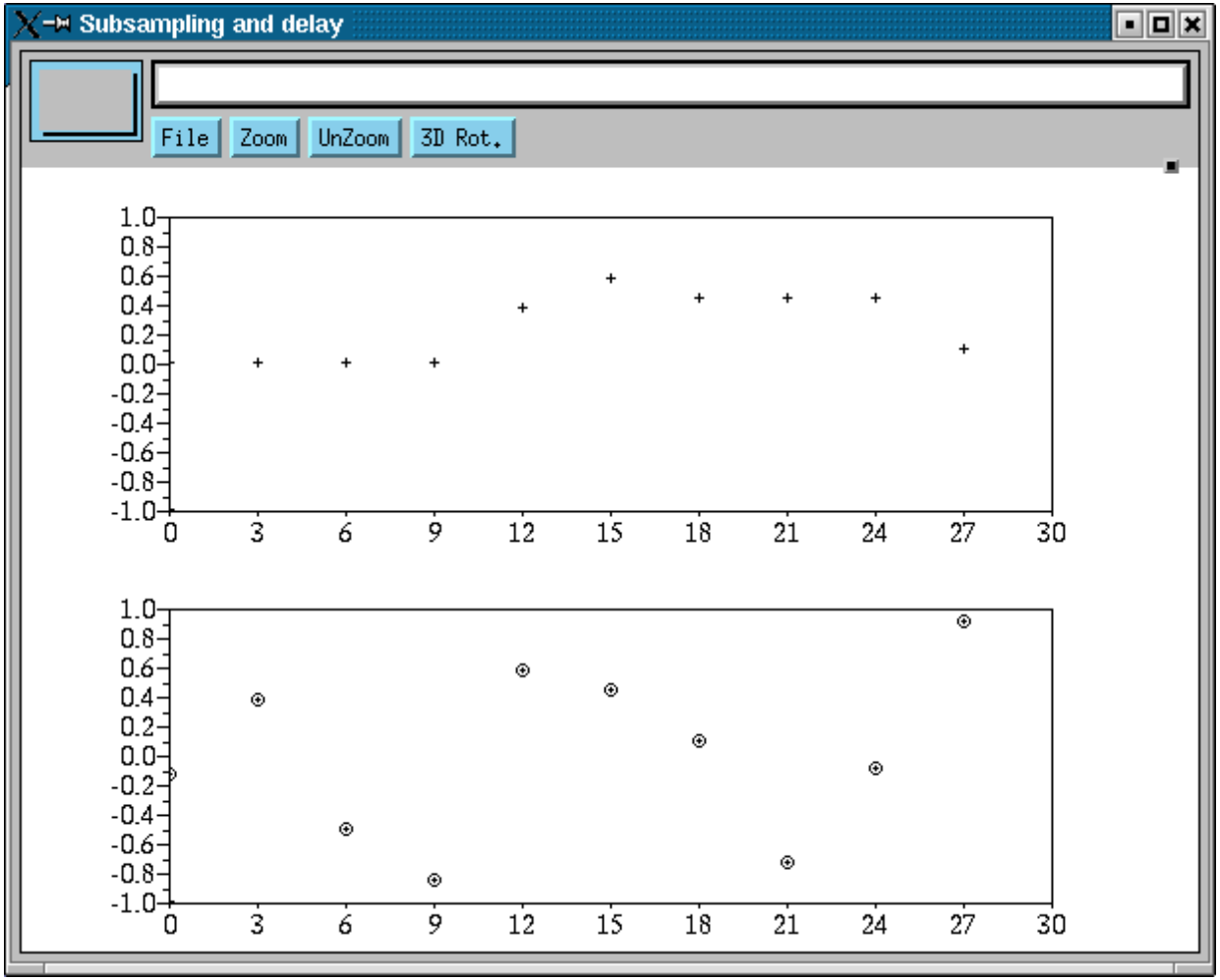
La multifréquence est réalisée dans un cadre synchrone utilisant le sous-échantillonnage ; cet exemple présente un cas de multifréquence conditionnelle.

Le bloc 1/z n'est activé que si la sortie aléatoire est positive.

Ce diagramme est synchrone (une seule source d'activation indépendante).

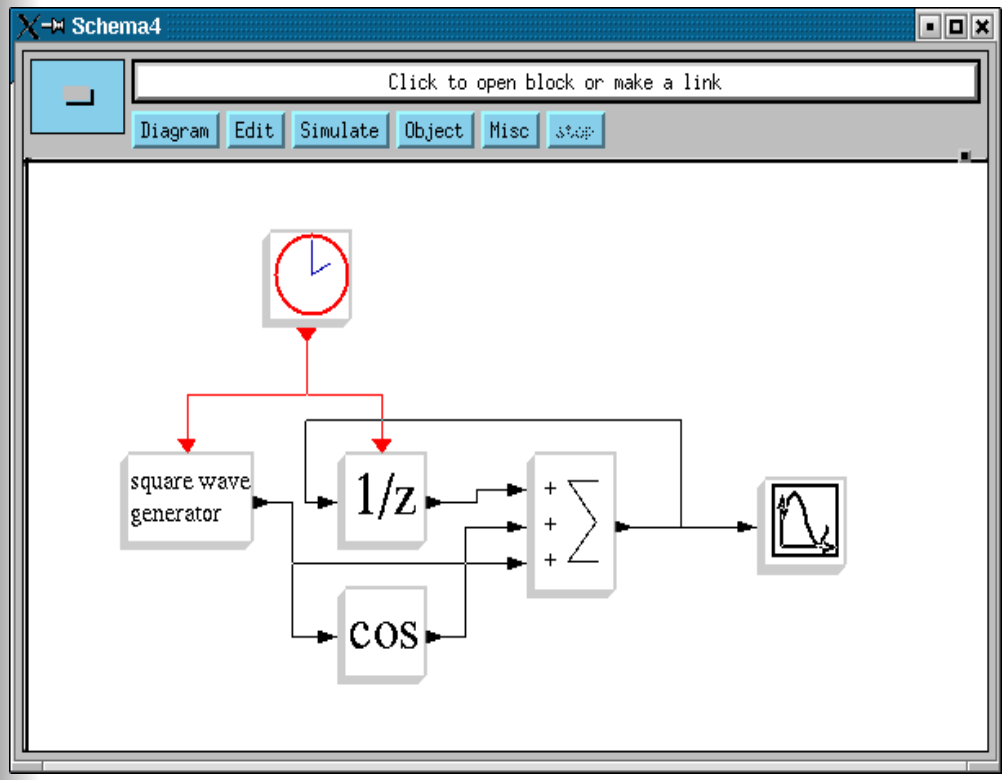
Le "bloc" If-Then-Else redirige les événements d'entrée vers l'une des sorties.

Simulation


Subsampling and delay
File
Zoom
UnZoom
3D Rot.
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-1.0
0
3
6
9
12
15
18
21
24
27
30

Event Driven vs Data Flow

- **Le formalisme Scicos est « event driven »: l'activation de chaque bloc est dû à un événement d'activation.**
- **Mais le mécanisme d'héritage produit un comportement de type "data flow" dans certains cas.**

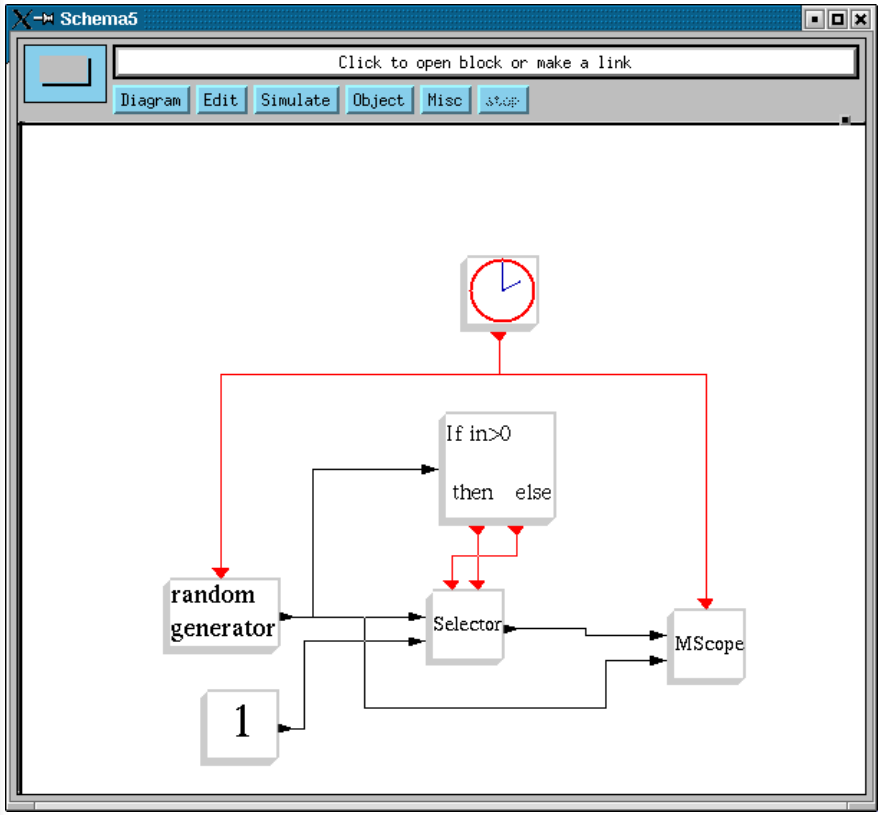


Un bloc sans entrée d'activation (et pas toujours-actif) hérite son activation à travers ses entrées régulières

Héritage et multifréquence

- Le mécanisme d'héritage est simple en présence d'une seule activation. Mais l'héritage marche dans le cas général (conditionnement, asynchronisme) suivant des règles précises.

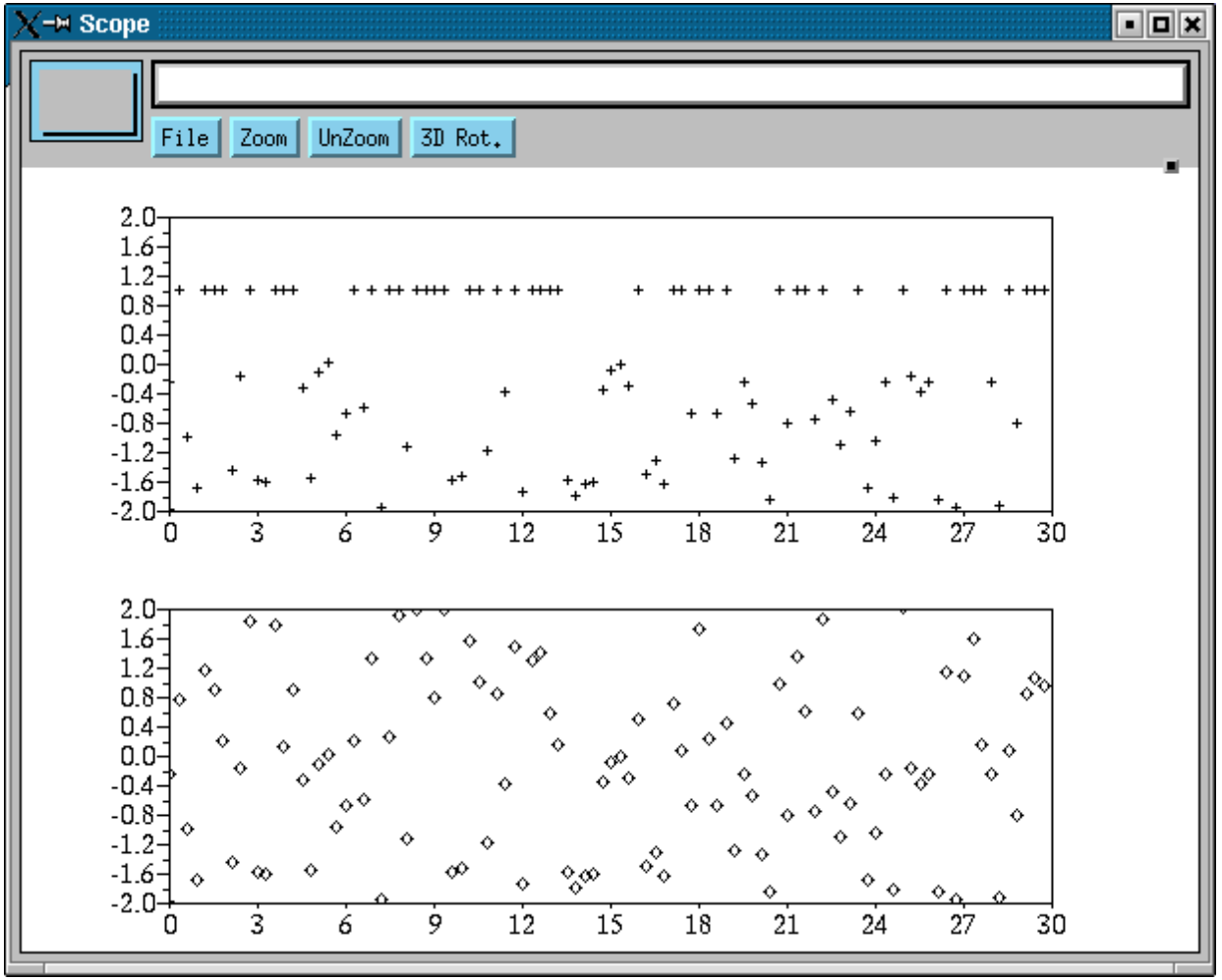
- Noter qu'en Scicos, un bloc peut avoir plusieurs entrées d'activation.



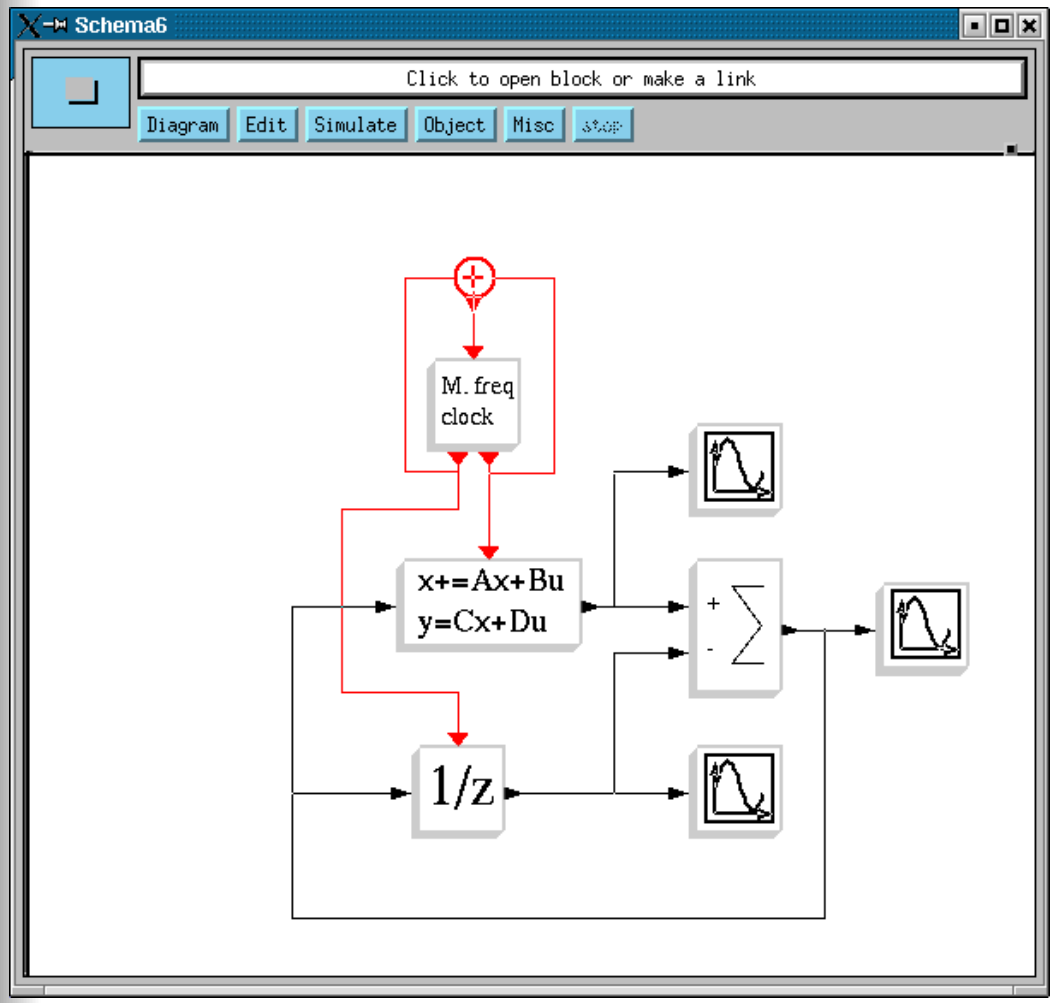
Le bloc Selector a 2 entrées d'activation.
Le bloc connait la voie par laquelle il a été activé (1, 2 ou 1-2) et choisit l'entrée à placer sur la sortie.
Donc la sortie du Selector est aléatoire si elle est négative sinon elle est égale à 1

Le bloc "1" correspond à une constante. Ce bloc n'est pas activé et n'hérite pas. Il est donc activé seulement une fois à l'initialisation.

Simulation


Scope
File
Zoom
UnZoom
3D Rot.
2.0
1.6
1.2
0.8
0.4
0.0
-0.4
-0.8
-1.2
-1.6
-2.0
0
3
6
9
12
15
18
21
24
27
30

Exemple asynchrone (héritage)



Ici la somme et les oscilloscopes marchent par héritage.

Les oscilloscopes héritent des activations différentes.

L'héritage des oscillos est simple.
La somme hérite de deux sources d'activation différentes.

Dans ce cas le mécanisme d'héritage crée deux ports d'entrée d'activation.

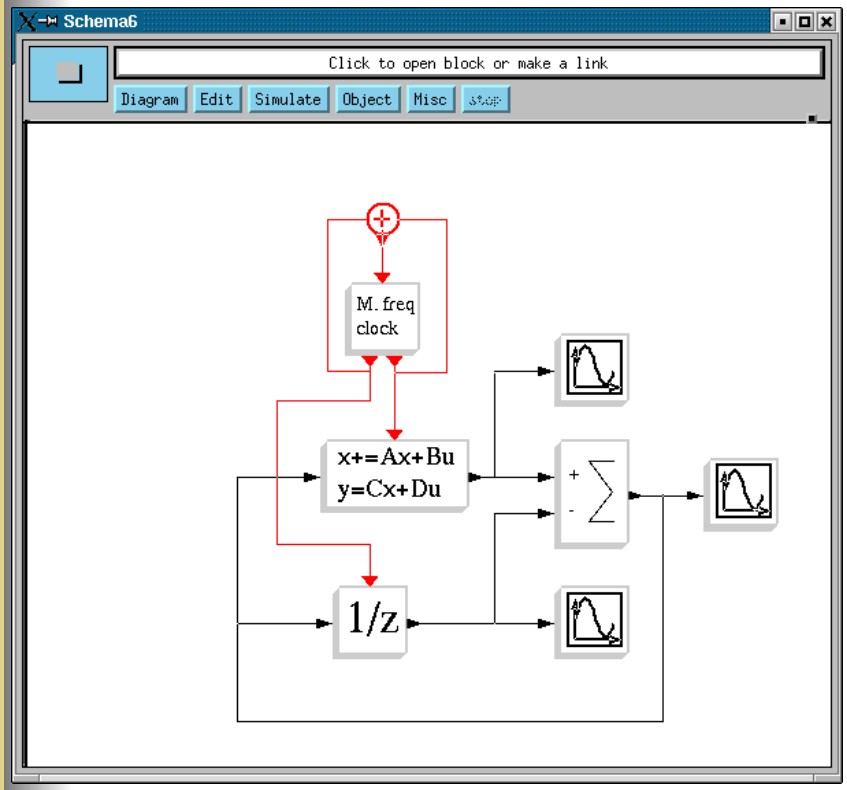
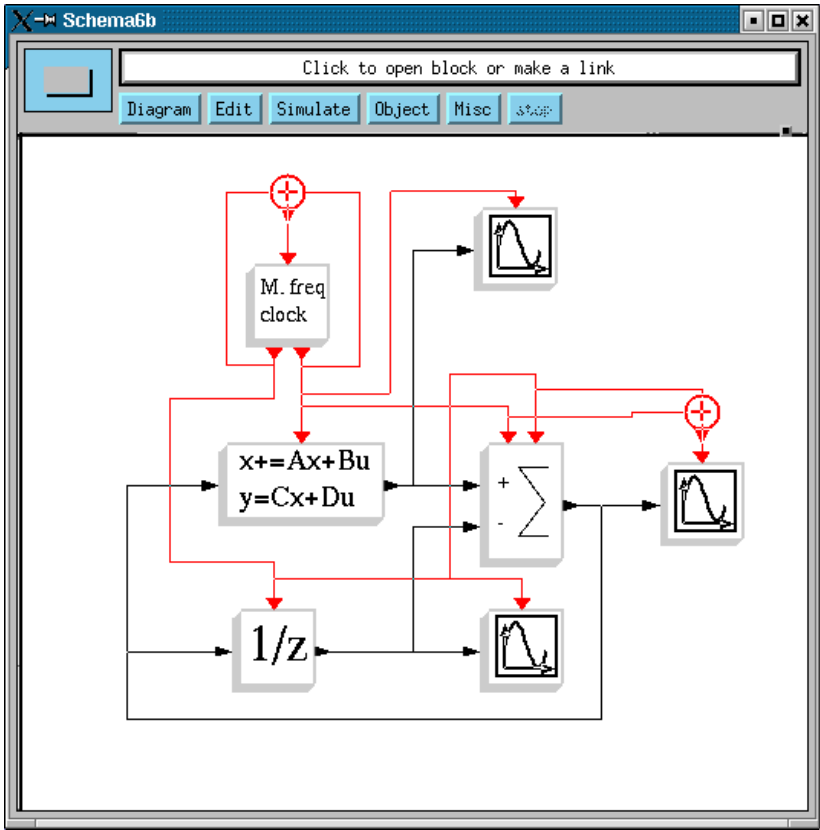
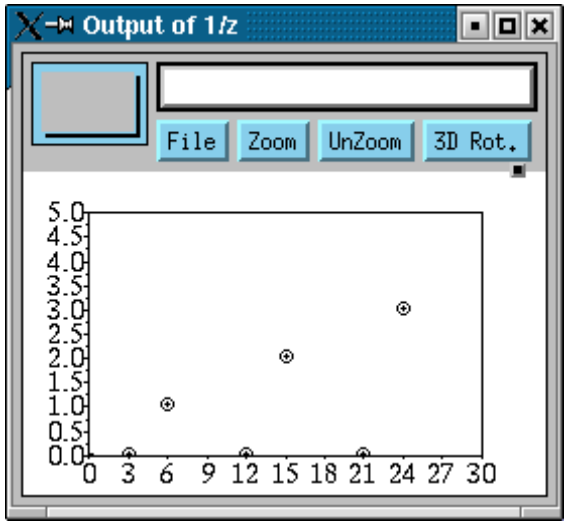


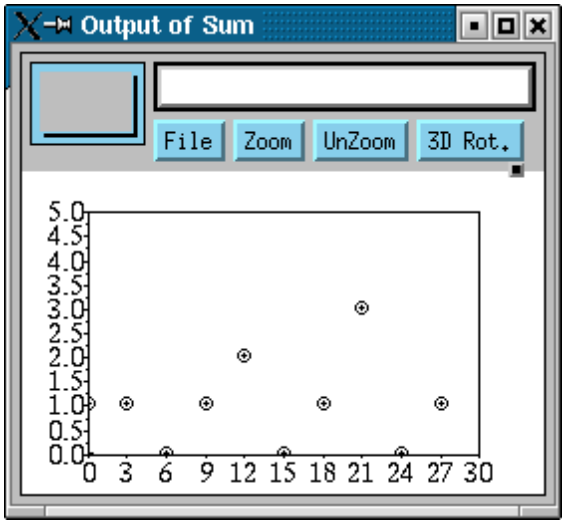
Schéma d'origine

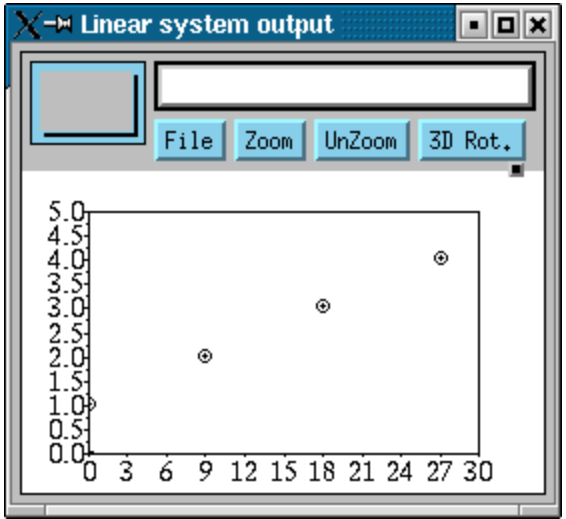
Prise en compte De l'héritage



Simulation (mécanisme d'héritage)



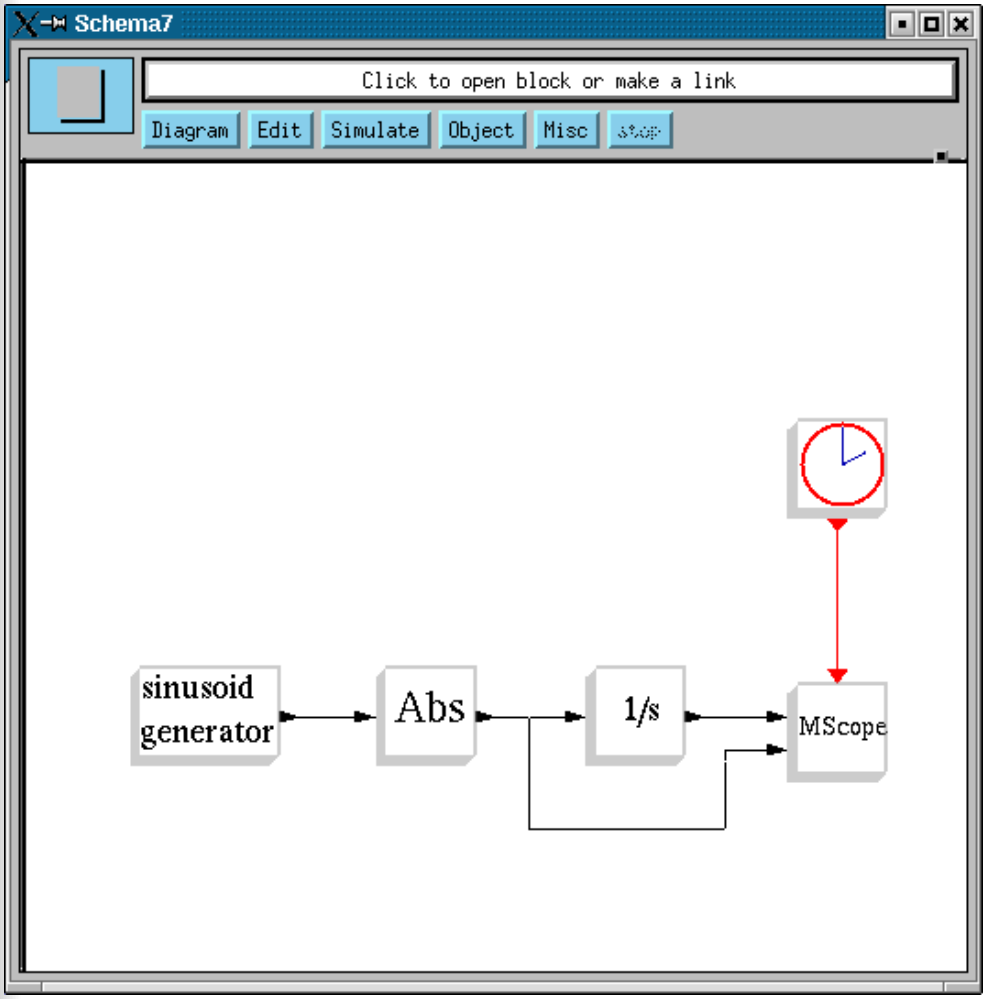




Temps-continu : toujours-actif

- **Un bloc peut être déclaré “toujours-actif”.**
- **Un bloc toujours-actif génère des sorties actives en temps continu.**
- **L’activation “toujours-actif” doit normalement être traitée comme toute autre activation. Mais pour simplifier l’édition du schéma, les blocs ainsi activés sont codés par un paramètre interne du bloc.**
- **Ils existent des blocs toujours-actifs dans les palettes Scicos (Sinusoid Generator, 1/s,…).**

Exemple



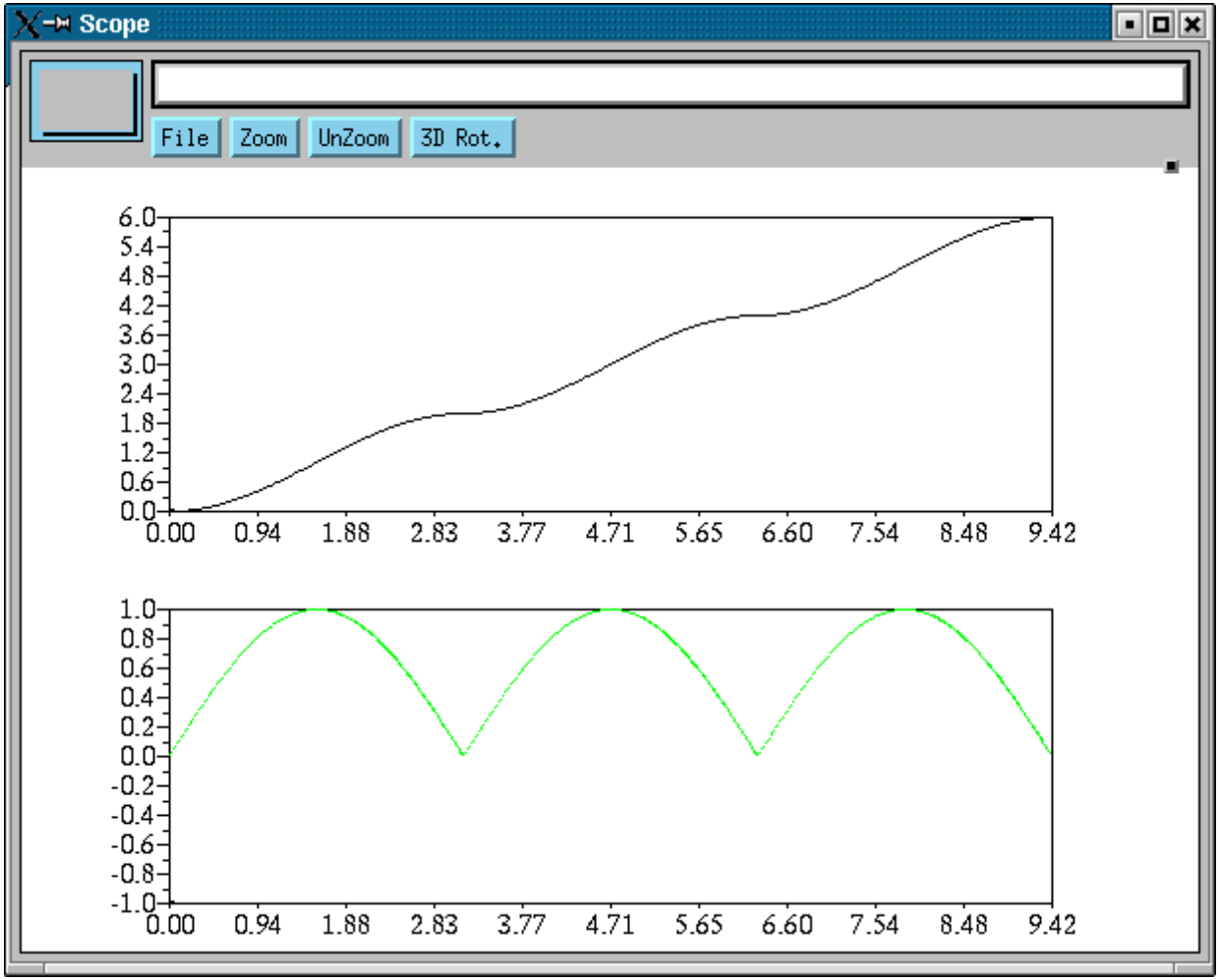
Sinusoid Generator et 1/s sont toujours-actifs.

Le bloc Abs l'est aussi par héritage.

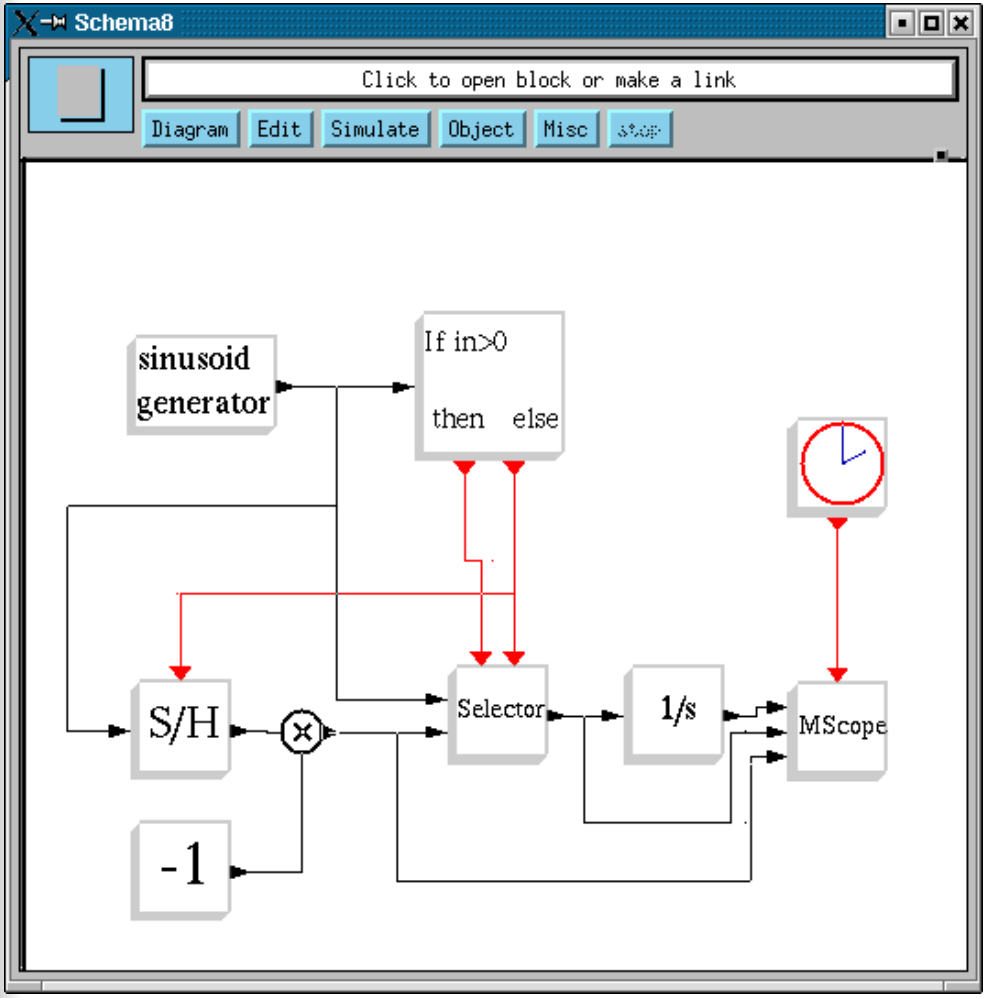
On aurait pu aussi ne pas déclarer 1/s toujours-actif dans ce cas. Mais pas en général car si l'entrée de 1/s est une constante, la sortie n'évolue pas.

Clock est utilisé ici juste pour rythmer l'affichage de l'oscilloscope.

Simulation


Scope
File
Zoom
UnZoom
3D Rot.
6.0
5.4
4.8
4.2
3.6
3.0
2.4
1.8
1.2
0.6
0.0
0.00
0.94
1.88
2.83
3.77
4.71
5.65
6.60
7.54
8.48
9.42
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6
-0.8
-1.0
0.00
0.94
1.88
2.83
3.77
4.71
5.65
6.60
7.54
8.48
9.42

Temps continu
Sous-échantillonnage



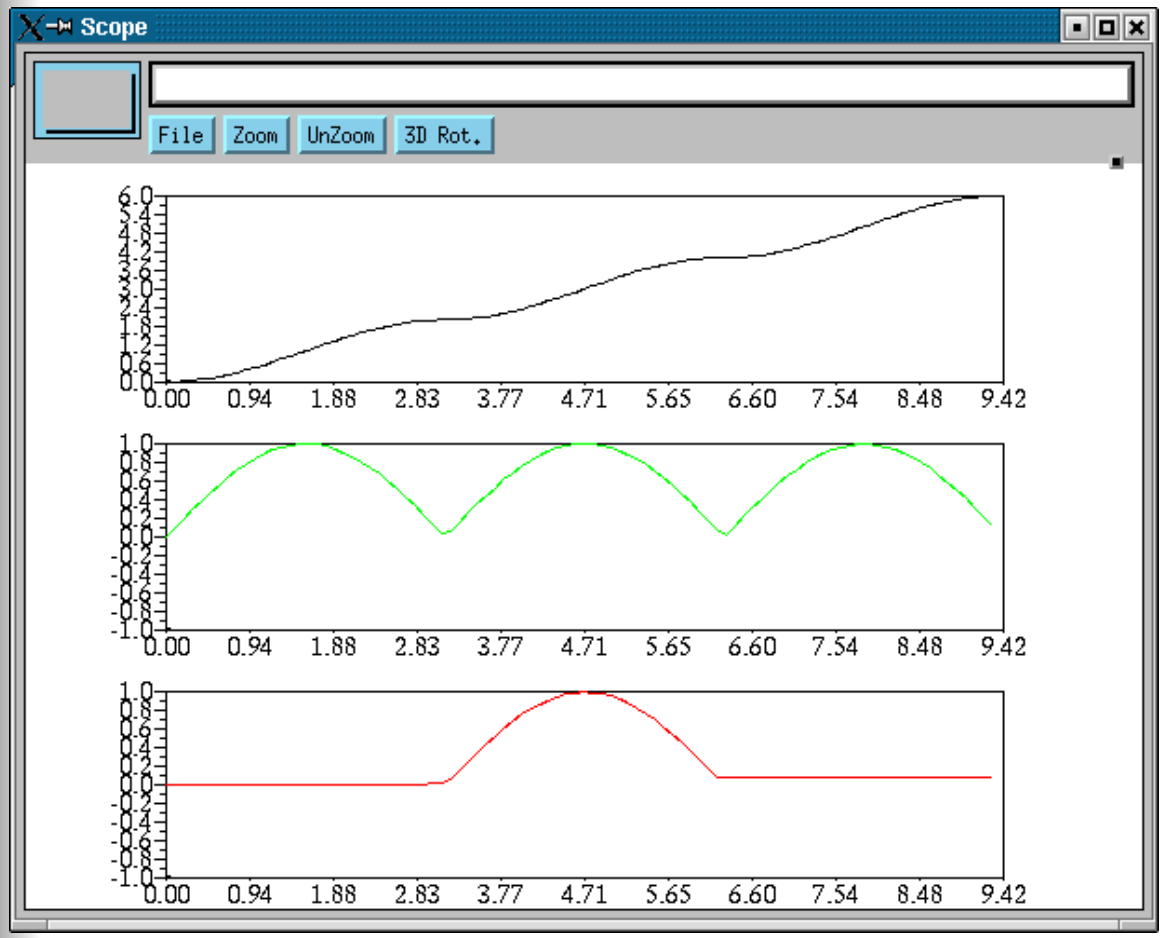
Le sous-échantillonnage marche aussi pour l'activation continue.

Sans le bloc S/H, le bloc multiplication serait toujours actif (par héritage de Sinusoid Generator).

Mais maintenant il hérite du "Else" du bloc If-Then_Else. Alors il est activé seulement quand sin(t) est négatif.

Economie à faire dans les cas complexes.

Simulation



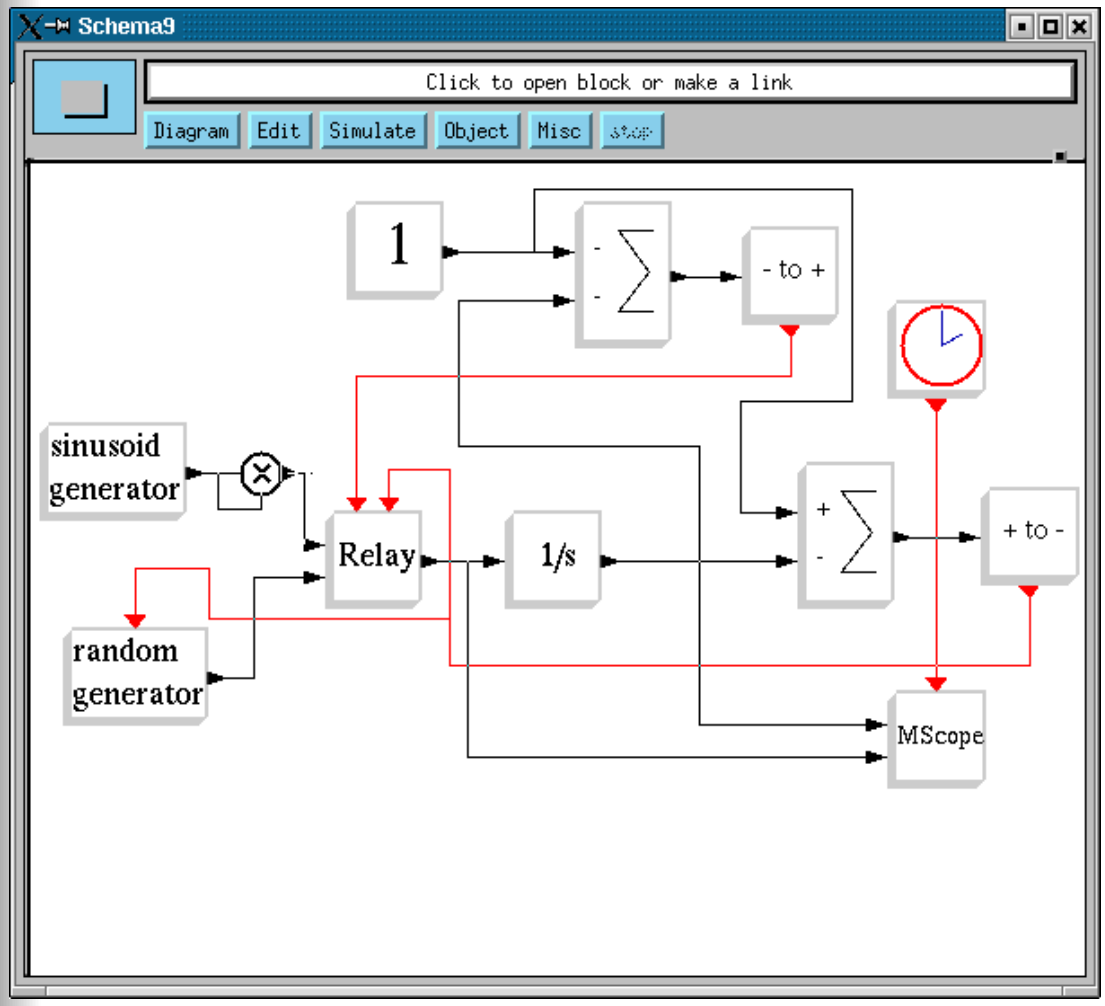
La simulation montre l'inactivité de la multiplication.

Temps continu/discret : interaction

Les opérations temps continu et les événements discrets interagissent :

- **Les activations continues et discretes peuvent activer le même bloc générant des signaux d'activations hybrides.**
- **Les signaux continus peuvent générer des événements par des blocs "zero-crossing".**
- **Un événement peut générer un saut dans un signal continu.**

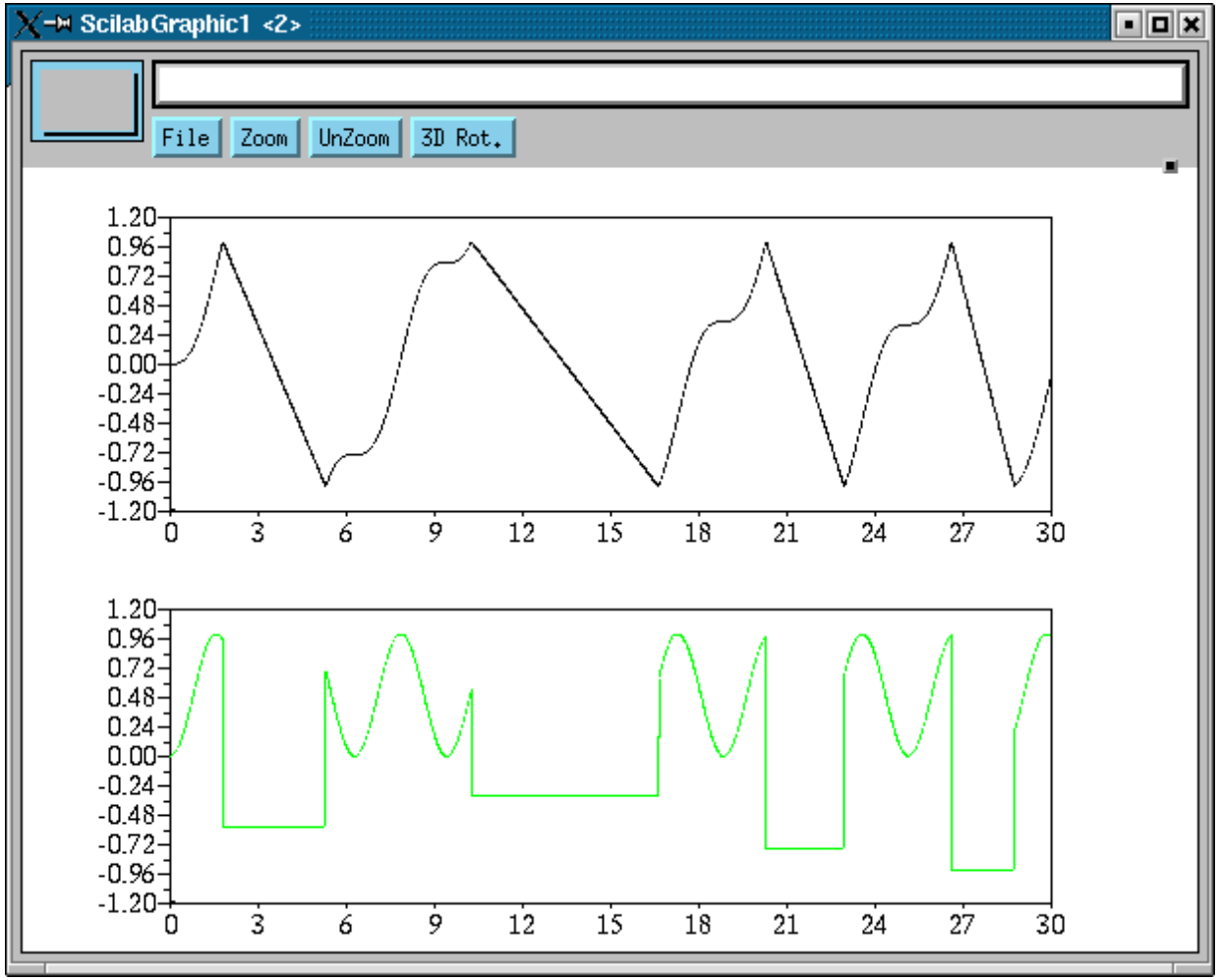
Exemple



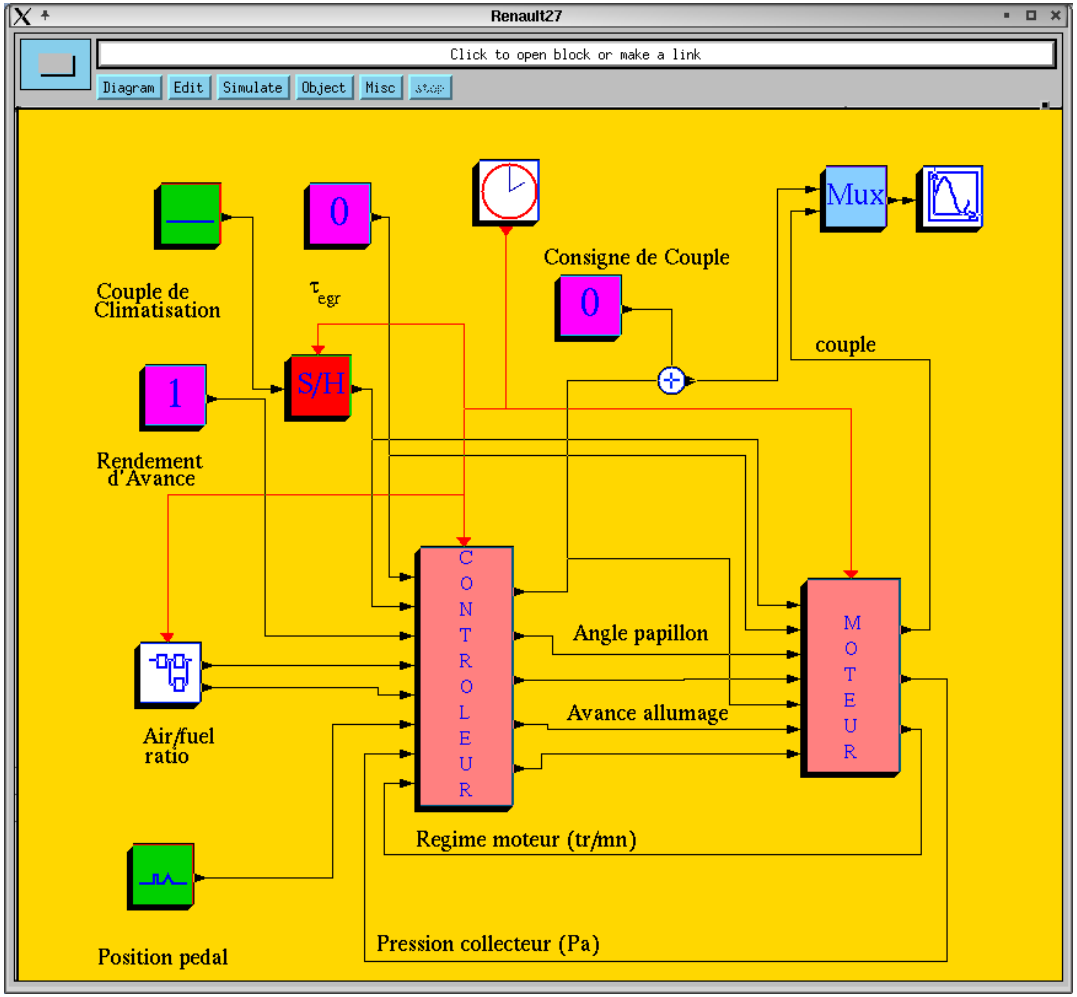
Les -to+ et +to- sont des blocs zero-crossing.

Ici l'événement zero-crossing est utilisé pour activer le bloc Relay dans un sens ou dans l'autre.

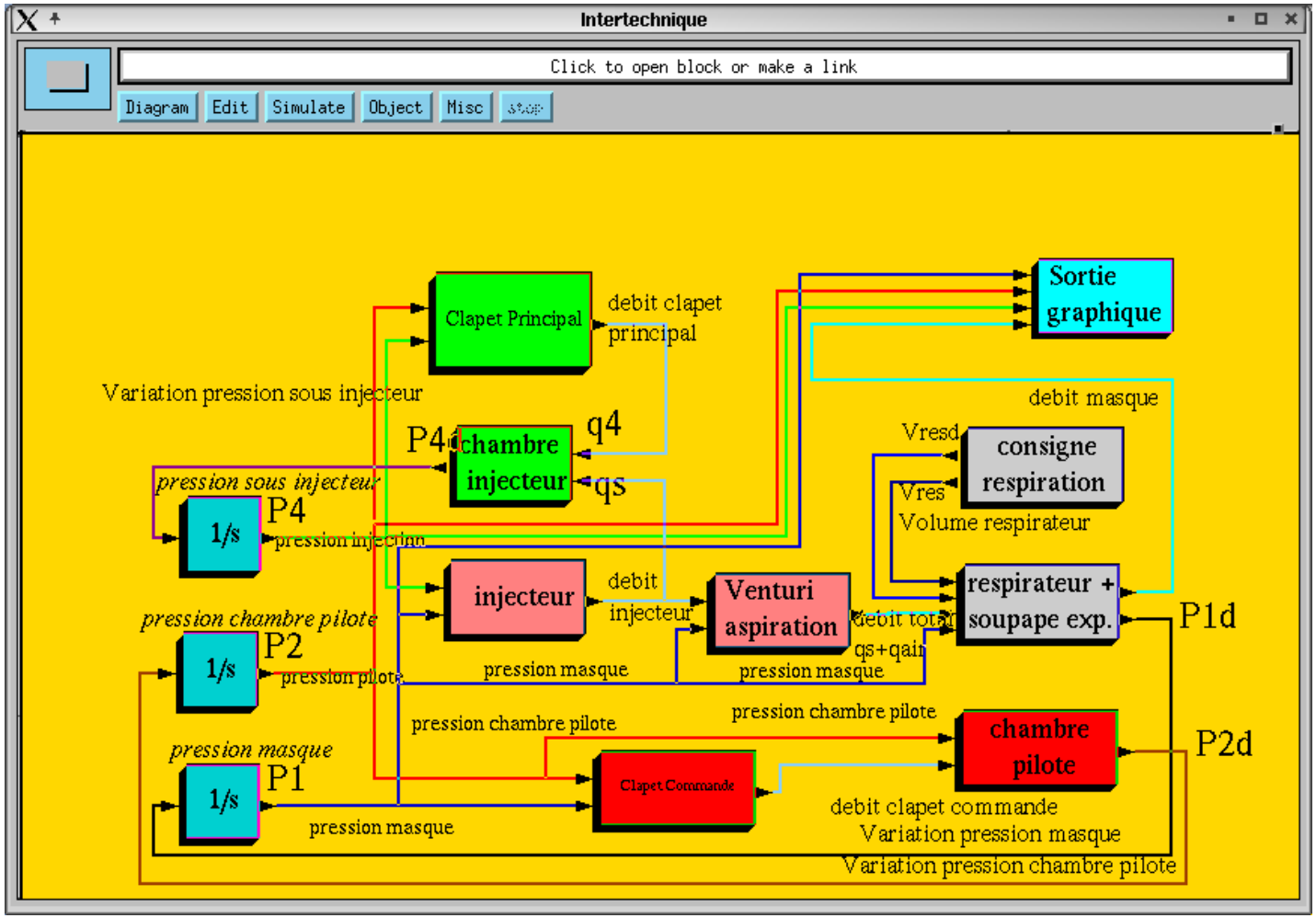
Simulation


ScilabGraphic1 <2>
File
Zoom
UnZoom
3D Rot.
1.20
0.96
0.72
0.48
0.24
0.00
-0.24
-0.48
-0.72
-0.96
-1.20
0
3
6
9
12
15
18
21
24
27
30

Application (Renault) moteur à injection directe



Application (Intertechnique) masque à oxygène



Modelica (Langage de modélisation)

- **Langage déclarative**
  - **Avec des équations et des fonctions mathématiques**
  - **Permet la modélisation non causal**
  - **Spécification haut niveau**
- **Modélisation multi-domaine**
  - **électrique, mécanique, hydraulique,…**
  - **Discret (commande), événementiel**
- **Orienté objets**
  - **Fortement typé**
- **Non propriétaire**
  - **Pas directement associé à un produit commercial**
  - **Définition du langage par « Modelica design group »**
  - **Existe depuis 1996**

Scicos ou Modelica : Avantages et Inconvénients

Avantages de Modelica

- **Modélisation au niveau des composants**
- **Modélisation formelle : Optimisation du code, calcul de Jacobien, réduction d'index**
- **Langage unique**

Inconvénients

- **Langage compliqué**
- **Utilisation des fonctions externes limitée aux fonctions statiques**
- **Loin du solveur et son implantation numérique : directives pour le solveur numérique (no-event, smooth,…) pas claires et insuffisantes**

Scicos ou Modelica : Avantages et Inconvénients

Avantages de Scicos

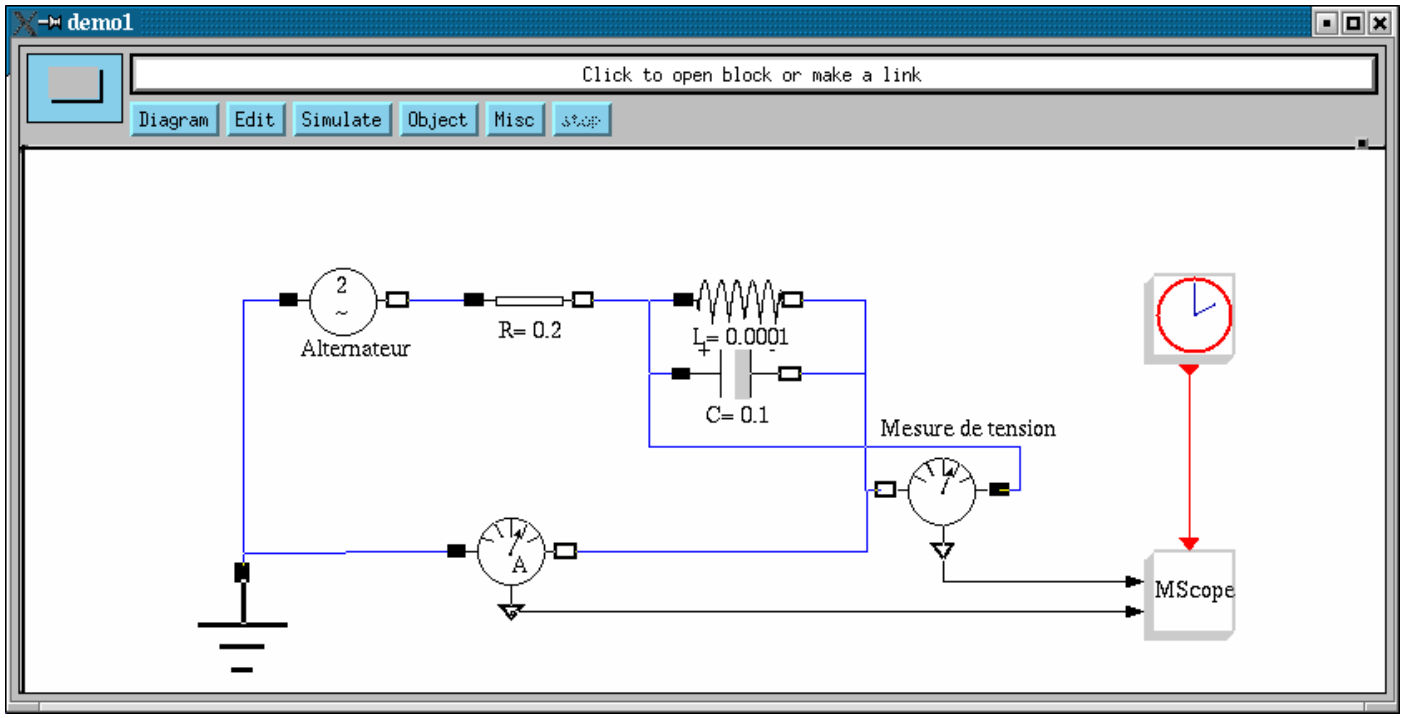
- **Formalisme simple et précis (peu de primitives)**
- **Utilisation des blocs à la simulink**
- **Bloc en C, C++,… Possible d'intervenir au bas niveau pour gérer le solveur numérique (solveur à pas variable,…)**

Inconvénients

- **Pas de « bloc » non-causal**
- **Pas de simplification formelle ni calcul de Jacobien**

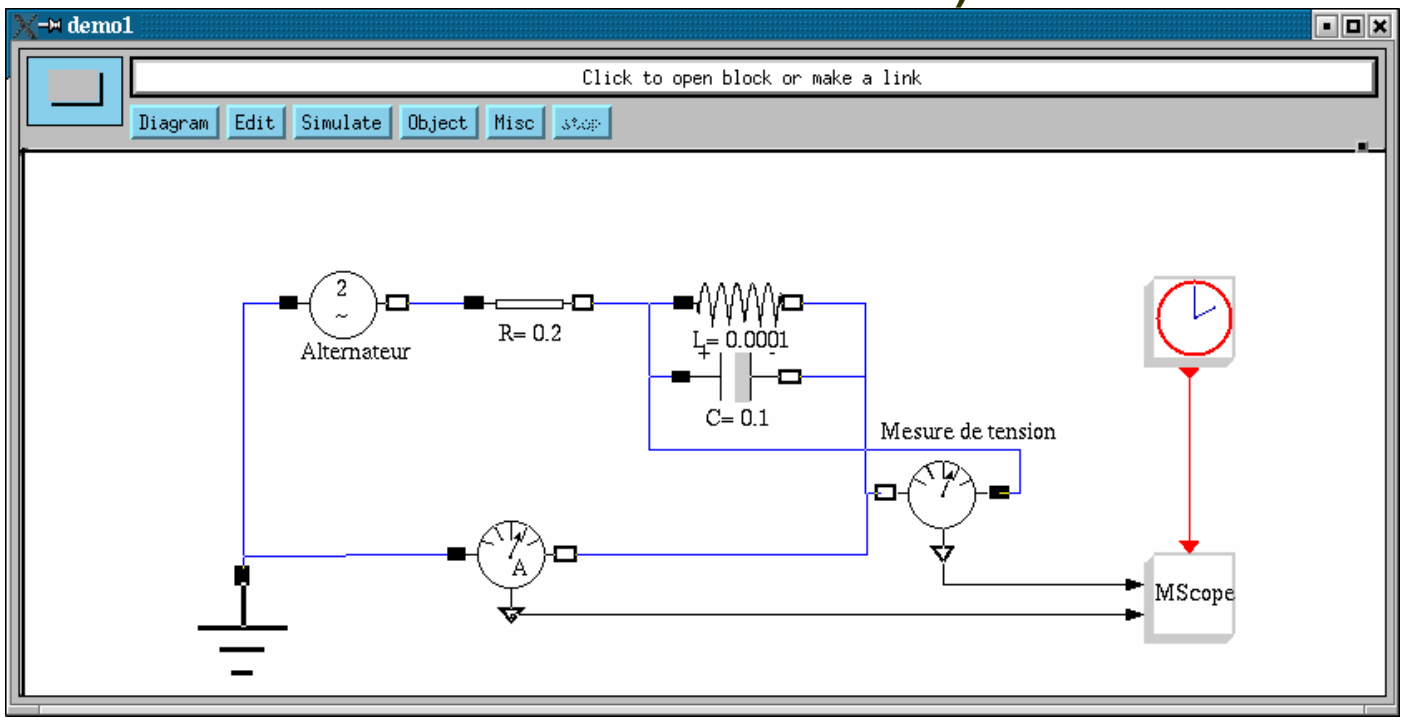
Scicos et Modelica : une première intégration

- **Extension du Scicos (RNTL Simpa)**
- **Editeur Scicos permet de mélanger les blocs Scicos et Modelica au sein du même schéma**
- **Précompilation : regroupement des blocs Modelica => programme Modelica => code C => bloc Scicos**



Scicos et Modelica : une première intégration

- **Similaire à l'intégration Dymola/Simulink, AMESim/Simulink et AMESim/Scicos**
- **Modelica utilisé pour construire sous-modèle temps-continu (peu de support pour le discret)**
- **Partie Modelica supposée toujours active (pas d'échange d'événement avec le reste du schéma)**



Scicos et Modelica : l'intégration complète

- **Projet RNTL Simpa2**
- **Formalisme hybride de Scicos et Modelica compatible :**
  - **« when », « edge » Modelica ⇔ activation Scicos**
  - **Notion d'événement**
  - **Réinitialisation d'état temps-continu par événement, …**
- **Objectif : chaque bloc peut être en Modelica ou en C (ou Scilab)**
- **Scilab/Scicos/Modelica : Environnement complet de simulation, ouvert et gratuit**

Perspectives

- **A court terme**
  - **Intégration complète du Modelica**
    - **Compilateur du langage complet**
    - **Inclusion des « Modelica libraries »**
  - **Plus de facilité pour la génération du code et des applications « hardware in the loop »**
  - **Extensions pour faciliter la migration Simulink vers Scicos**
  - **Amélioration de l'éditeur**
  - **Optimisation du compilateur**

Perspectives

- **A long terme**
  - **Transfert du développement et maintenance de l'éditeur à l'équipe Scilab**
  - **Traducteur Simulink vers Scicos**
  - **Liens avec DSpace, NI, …, pour pouvoir proposer une chaine complète**
  - **Documentation professionnelle**
  - **Intégration du Synchart (ou autre logiciel similaire)**