Università di Roma La Sapienza
Ingegneria dell'informazione, informatica e statistica
Dipartimento di Informatica

Modelica2GPU

From Modelica code to C++ MPGOS Translator

Riccardo La Marca 1795030
riccardo.lamarca98@gmail.com
lamarca.1795030@studenti.uniroma1.it

Ottobre, 2020

Indice

Introduzione

Sempre più, nella nostra realtà, sta diventando importante modellare e simulare *sistemi cyber-fisici*, come droni o sistemi IoT, e *sistemi biologici*, come il ciclo cellulare o ancora lo sviluppo e la propagazione di un virus. Sebbene questi due siano argomenti diversi, hanno principalmente una cosa in comune: entrambi possono essere espressi come un sistemi di equazioni differenziali (o *ODE*, *Ordinary Differential Equation*) il quale può essere integrato sul tempo al fine di dare un "vita" al modello stesso il quale risulterà in un dataset con cui si andrà a descrivere l'andamento, in termini di valori sugli stati, quindi la **traiettoria** del sistema stesso. Questa fase di integrazione viene chiamata *simulazione* del modello, durante la quale vengono ad essere scoperti i valori sugli stati, calcolati a seguito del processo di integrazione delle derivate e di valutazione di particolari situazioni che vengono dette **eventi**.

Gli *eventi* sono delle situazioni alternative che deviano il normale corso della traiettoria del sistema facendole toccare punti che altrimenti non avrebbero mai considerato. Esistono quindi due tipologie di eventi: di *stato* e di *tempo*. I primi sorgono da condizioni sugli stati, ad esempio $x > 10$ con x stato del sistema, mentre i secondi da condizioni sul tempo. Su quest'ultimi il discorso diventa più ampio però un classico evento di questo tipo scaturisce inserendo nel sistema una componente detta *Sample-And-Hold*, che agisce creando una condizione sul tempo della simulazione. Una componente di *Sample-And-Hold*, letteralmente "campionamento e mantenimento" è una classica componente che viene inserita all'intero di sistemi continui al fine di discretizzarne una parte di comportamento. Il nome stesso deriva dalle due azioni che questo svolge: (1) campiona il tempo della simulazione in base a dei valori che per semplicità chiameremo *start* ed *interval*, e (2) mantiene fino al prossimo sample il valore di uno stato. Un tipico esempio è l'**Analog to Digital Converter** (ADC) il quale implementa internamente il sottosistema di s&h[1]. Gli eventi, quindi, servono principalmente per immettere una discretizzazione all'interno di un sistema che altrimenti sarebbe continuo.

```
parameter Real sample_time(unit="s")=0.1251231;
when sample(0, sample_time) then
  omega1_measured = omega1; // omega1 è uno stato
end when;
```

Figure 1.1: Esempio di componente SH in Modelica.

[1] Sample-And-Hold

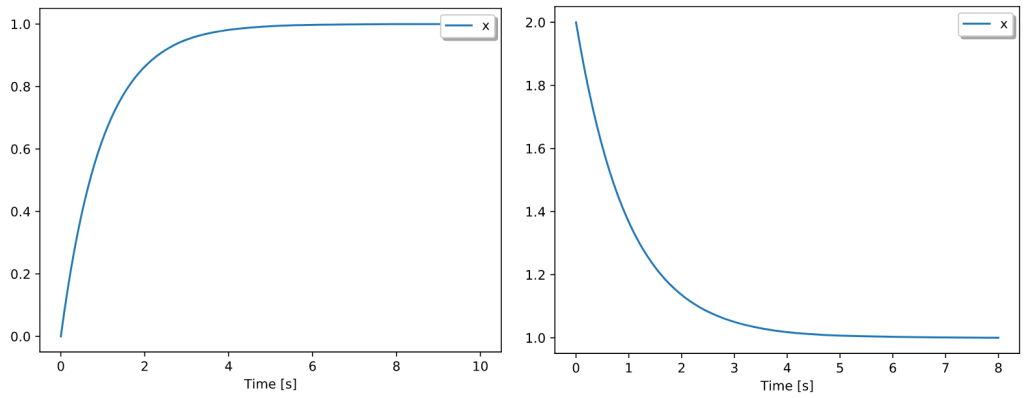
Esempio1. First order system with sample-and-hold

Si consideri il semplice sistema descritto da una sola equazione differenziale[2].

$$\dot{x} = (1 - x) \tag{1.1}$$

Vediamo come questo sistema descrive semplicemente l'andamento dello stato x fino al punto 1, sul quale troverà l'equilibrio dal momento che per tutti i successivi istanti di tempo non cambierà mai di valore. Difatti il grafico della traiettoria sarà il seguente

Se volessimo aggiungere una componente SH all'interno del sistema per vedere il valore di x a



(a) Traiettoria del sistema con valore iniziale $x = 0$

(b) Traiettoria del sistema con valore iniziale $x = 2$

determinati istanti di tempo, decidiamo prima il tempo di sampling (es. 0.3) e implementiamo tale evento, ottenendo

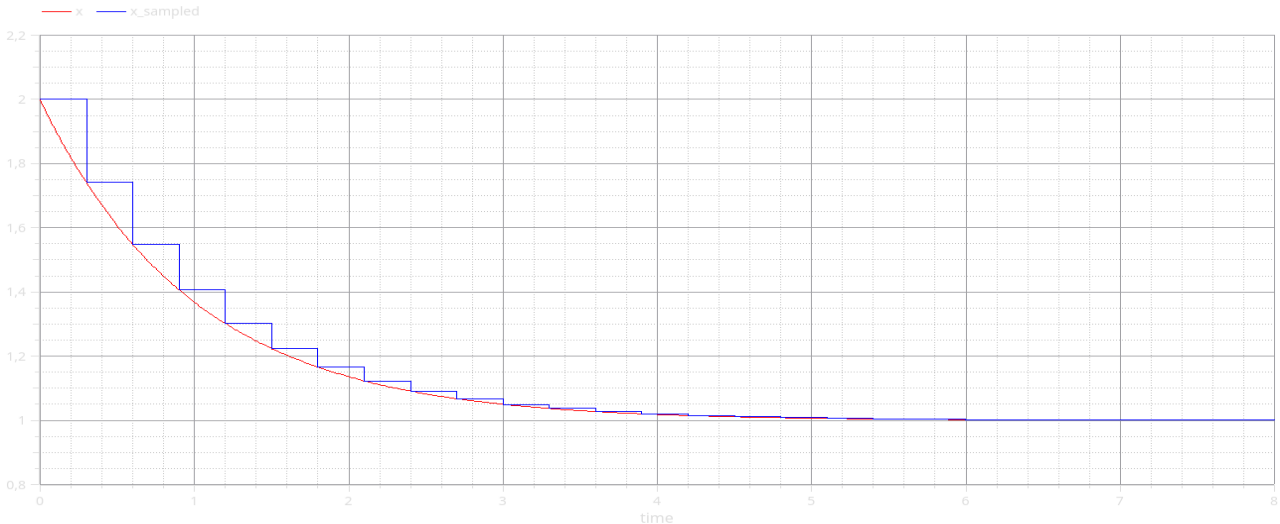


Figure 1.3: FirstOrderSystem con sample-and-hold e sample time di 0.3

[2]Scriveremo $\dot{x}$ al posto di $\frac{\mathrm{d}}{\mathrm{d}t}x(t)$