Крайни автомати Stateflow/Simulink, HIL системи

Практикум по приложение на графични програмни среди

Владимир Димитров

Катедра Силова Електроника
Факултет по Електронна Техника и Технологии

11 октомври 2018 г.

Съдържание

Крайни автомати

Крайни автомати

Основни дефиниции

Крайни автомати

Цел на дисциплината

Автомат (FSM - finite state machine) е за моделиране на системи, които могат да се опишат като комбинация от крайно множество състояния в които те могат да бъдат, както и условията за преход между тях. Такива системи се наричат реактивни.

Цел на дисциплината

Автомат (FSM - finite state machine) е за моделиране на системи, които могат да се опишат като комбинация от крайно множество състояния в които те могат да бъдат, както и условията за преход между тях. Такива системи се наричат реактивни.

- Използват се за моделиране на:

Крайни автомати

Цел на дисциплината

Автомат (FSM - finite state machine) е за моделиране на системи, които могат да се опишат като комбинация от крайно множество състояния в които те могат да бъдат, както и условията за преход между тях. Такива системи се наричат реактивни.

- Използват се за моделиране на:
  - Електрически вериги: променливите на състоянието са напрежение на кондензатор и ток през индуктивност или графове

Крайни автомати

Цел на дисциплината

Автомат (FSM - finite state machine) е за моделиране на системи, които могат да се опишат като комбинация от крайно множество състояния в които те могат да бъдат, както и условията за преход между тях. Такива системи се наричат реактивни.

- Използват се за моделиране на:
  - Електрически вериги: променливите на състоянието са напрежение на кондензатор и ток през индуктивност или графове
  - Софтуер: Чрез Universal Modeling Language (UML диаграма на състоянията).

Крайни автомати

Цел на дисциплината

Автомат (FSM - finite state machine) е за моделиране на системи, които могат да се опишат като комбинация от крайно множество състояния в които те могат да бъдат, както и условията за преход между тях. Такива системи се наричат реактивни.

- Използват се за моделиране на:
  - Електрически вериги: променливите на състоянието са напрежение на кондензатор и ток през индуктивност или графове
  - Софтуер: Чрез Universal Modeling Language (UML диаграма на състоянията).
  - Цифрови системи: Състоянията на веригата се помнят от последователности логически елементи (тригери, памети)

Видове Крайни автомати

- Автомат на Moore - изхода y на системата е функция f от сегашното състояние X.

$$y = f(X)$$

Видове Крайни автомати

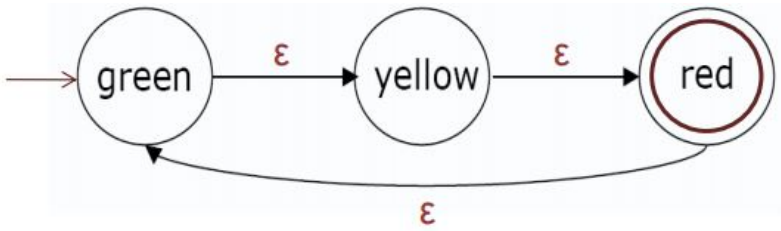
- Автомат на Moore - изхода y на системата е функция f от сегашното състояние X.

$$y = f(X)$$

- Автомат на Mealy - изхода y на системата е функция f от сегашното състояние X и сегашния вход u.

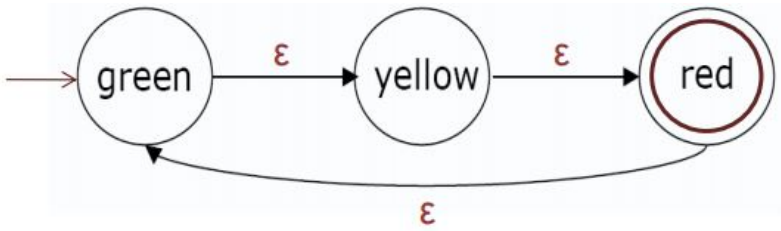
$$y = f(X, u)$$

Описание



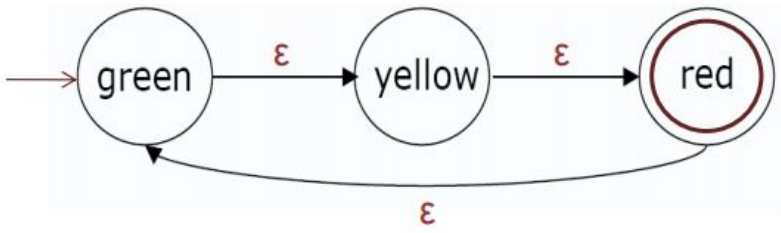
- За да се опише FSM е необходимо $(S, s_0, \delta, F, \sum)$

Описание



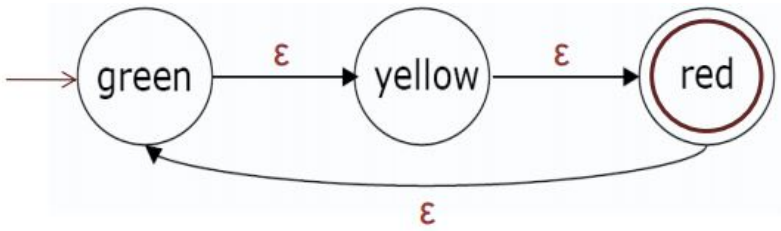
- За да се опише FSM е необходимо $(S, s_0, \delta, F, \sum)$
  - S- множество от състояния [red,yellow,green]

Описание



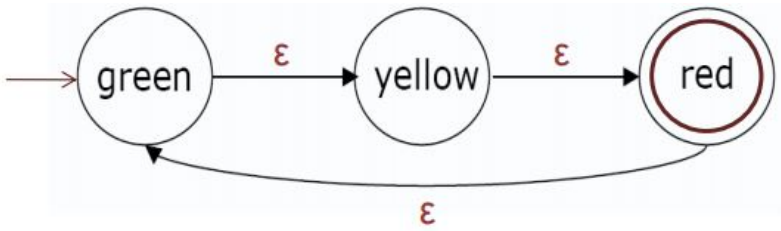
- За да се опише FSM е необходимо $(S, s_0, \delta, F, \sum)$
  - S- множество от състояния [red,yellow,green]
  - S_0 - първоначално състояние [green]

Описание



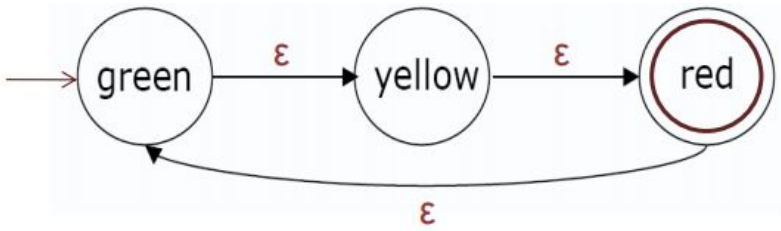
- За да се опише FSM е необходимо $(S, s_0, \delta, F, \sum)$
  - S- множество от състояния [red,yellow,green]
  - S_0 - първоначално състояние [green]
  - F - множество на крайните състояния [red]

Описание



- За да се опише FSM е необходимо $(S, s_0, \delta, F, \sum)$
  - S- множество от състояния [red,yellow,green]
  - S_0 - първоначално състояние [green]
  - F - множество на крайните състояния [red]
  - δ - множество от условия, които определят преминаването в отделните състояния
    $[green[\epsilon] -> yellow, yellow[\epsilon] -> red, red[\epsilon] -> green]$

Описание



- За да се опише FSM е необходимо $(S, s_0, \delta, F, \sum)$
  - S- множество от състояния [red,yellow,green]
  - S_0 - първоначално състояние [green]
  - F - множество на крайните състояния [red]
  - δ - множество от условия, които определят преминаването в отделните състояния
    $[green[\epsilon] -> yellow, yellow[\epsilon] -> red, red[\epsilon] -> green]$
  - $\sum$ множество на използваните символи за преходите $[\epsilon]$

Пример

Да се състави диаграма на състоянията на D тригер.

Пример

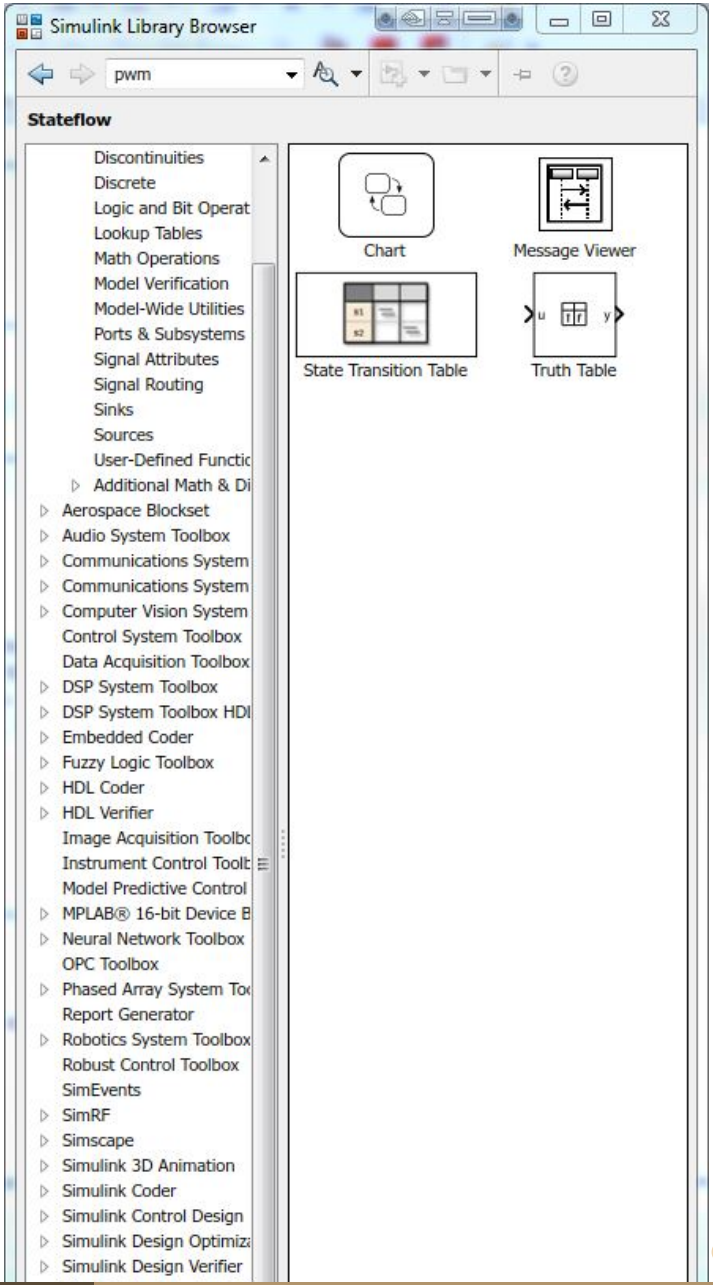
Да се състави цифрова схема и диаграма на състоянията по представената таблица на истинност

	X		
A, B	0	1	Z_1
00	00	01	1
01	01	10	1
10	10	11	0
11	11	00	0
	A^*, B^*		

Крайни автомати

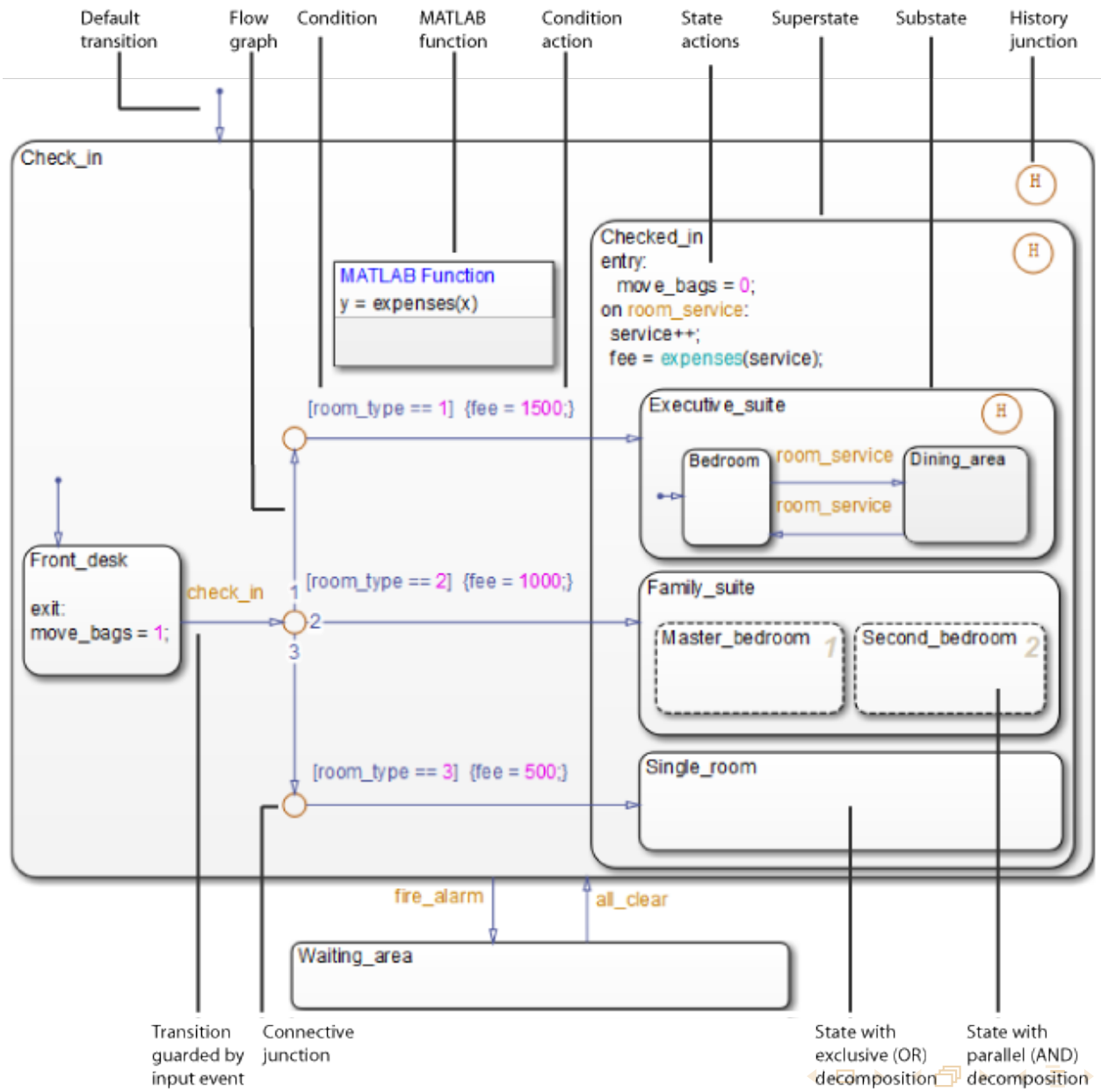
Stateflow

Основните елементи в библитека Simulink/Stateflow



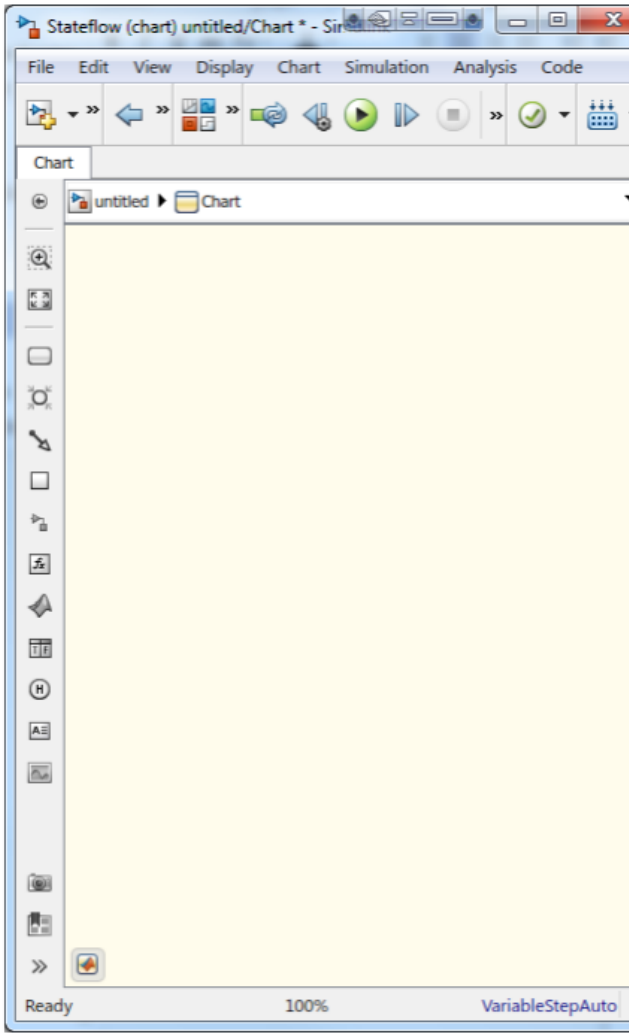
- Chart: За създаване на диаграма на състоянията, която се свързва към Simulink модел - основен метод
- Truth table: За задаване на таблица на истинност, която

Диаграма на състоянието

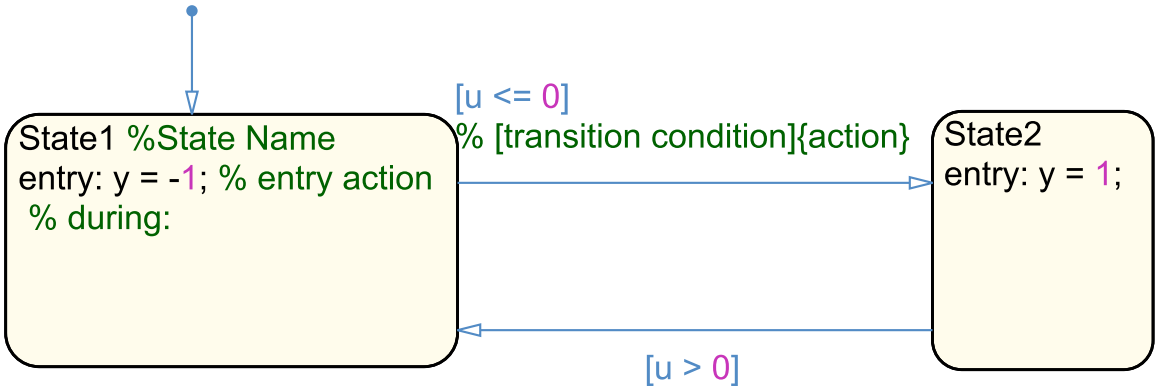


Основните елементи в Chart

- 1- State - добавя нови състояния
- 2- Junction - за създаване на конструкции if-then-else, for.
- 3- Задаване на начален преход към състояние
- 5,6,7,8 - Задаване по алтернативен начин - изпълняване от Matlab скрипт, Simulink файл, таблица на истинност или графична функция
- 9 - Запомняне на предишно състояние
- 10,11 - За украсяване с описание или картинка



Синтаксис на блок състояние

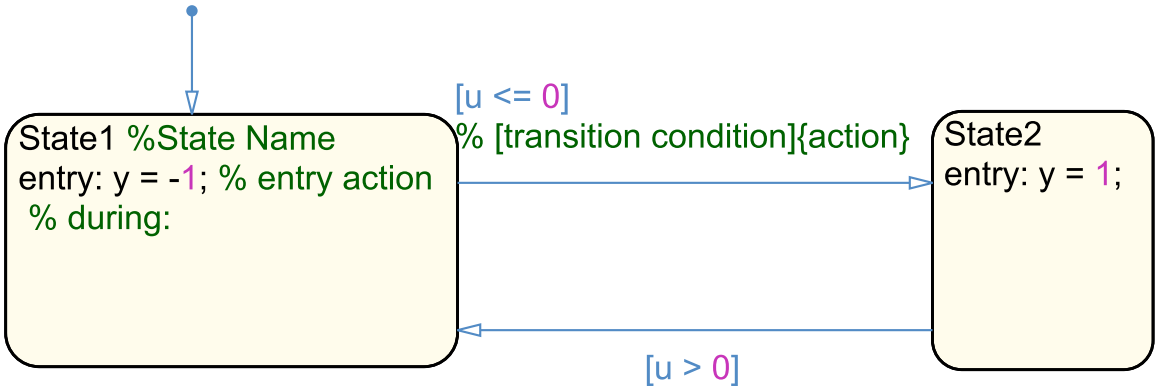


- entry - при влизане в състоянието

- on - при настъпване на събитие event_name или при постъпване на съобщение message_name
- bind - указва, че съответното състояние е единствения възможен източник на събитие x.

Състоянията могат да бъдат ексклузивно изпълняване (логическо

Синтаксис на блок състояние

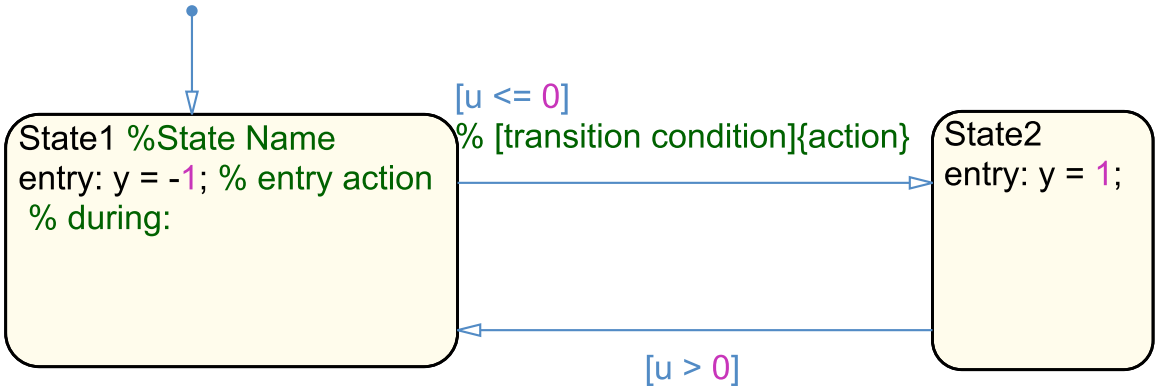


- entry - при влизане в състоянието
- during - докато сме в състоянието

- on - при настъпване на събитие event_name или при постъпване на съобщение message_name
- bind - указва, че съответното състояние е единствения възможен източник на събитие x.

Състоянията могат да бъдат ексклузивно изпълняване (логическо

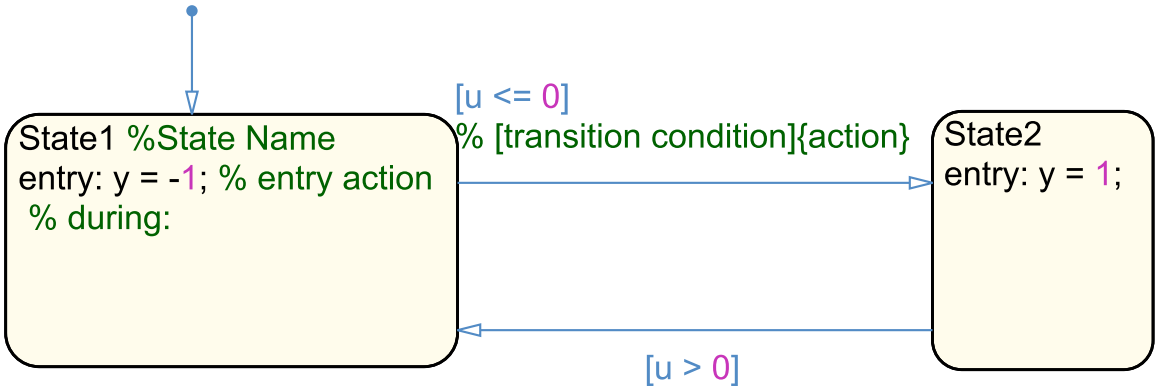
Синтаксис на блок състояние



- entry - при влизане в състоянието
- during - докато сме в състоянието
- exit - при излизане от състоянието
- on - при настъпване на събитие event_name или при постъпване на съобщение message_name
- bind - указва, че съответното състояние е единствения възможен източник на събитие x.

Състоянията могат да бъдат ексклузивно изпълняване (логическо

Преходи между състояния



Синтаксис:

event_or_message[condition]condition_action/transition_action.

- event_or_message - показва сигналът който би довел до преход
- [*condition*] - булев израз за сигнал, който ако е истина се прави прехода
- condition_action - Действие, което се изпълнява при условие, че [condition] е истина и преди да е изчислено следващото състояние.
- transition_action - Действие, което се изпълнява при условие, че [condition] е истина и след като е определено следващото

Крайни автомати в Stateflow

- При автоматите на Moore действията се извършват само в състоянията, докато при тези на Mealy само при преходи.

Крайни автомати в Stateflow

- При автоматите на Moore действията се извършват само в състоянията, докато при тези на Mealy само при преходи.
- Изхода на изчислението е достъпен по-рано при Mealy

Крайни автомати в Stateflow

- При автоматите на Moore действията се извършват само в състоянията, докато при тези на Mealy само при преходи.
- Изхода на изчислението е достъпен по-рано при Mealy
- Moore автоматите не създават алгебрични контури при използване на тяхните изходи в затворен контур

Пример 1

Да се състави диаграма на състоянията на компаратор. x,y да са съответно вход и изход. Прехода между състоянията да става със условие за x. На входа да се подаде синусоидална напрежение.

Пример 2

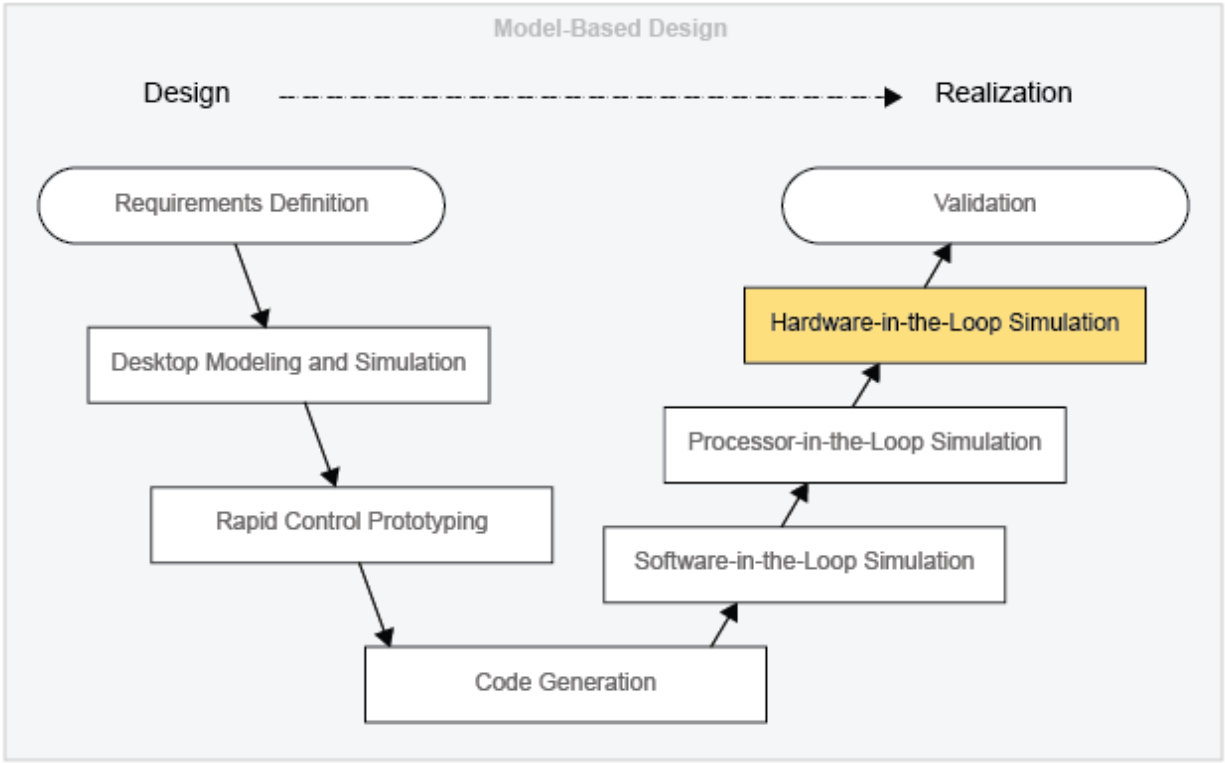
Да се състави диаграма на състоянията на D тригер чрез Stateflow. Входовете са clk, d, reset. Изходите са q и qinv. Прехода между състоянията да става при събитие clk.

Модел Базирано Проектиране

Модел Базирано Проектиране

Основни понятия

- Концепцията model-based design е процес, който позволява бърза разработка на динамични системи, включващи микроконтролери. При тези системи моделът на системата е в центъра на разработката, като той се реализира на няколко етапа.



Модел Базирано Проектиране

Етапи на моделирането

- MIL (Model in the Loop) - проектиране със предавателни характеристики или диаграма на състоянията чрез Simulink/Stateflow. Изясняване на концепцията

- MIL (Model in the Loop) - проектиране със предавателни характеристики или диаграма на състоянията чрез Simulink/Stateflow. Изясняване на концепцията
- SIL (Software in the Loop) - проектиране на софтуерната част и нейното тестване с помощта на Simulink. Възможност за генериране на C код от проектираната Stateflow диаграма

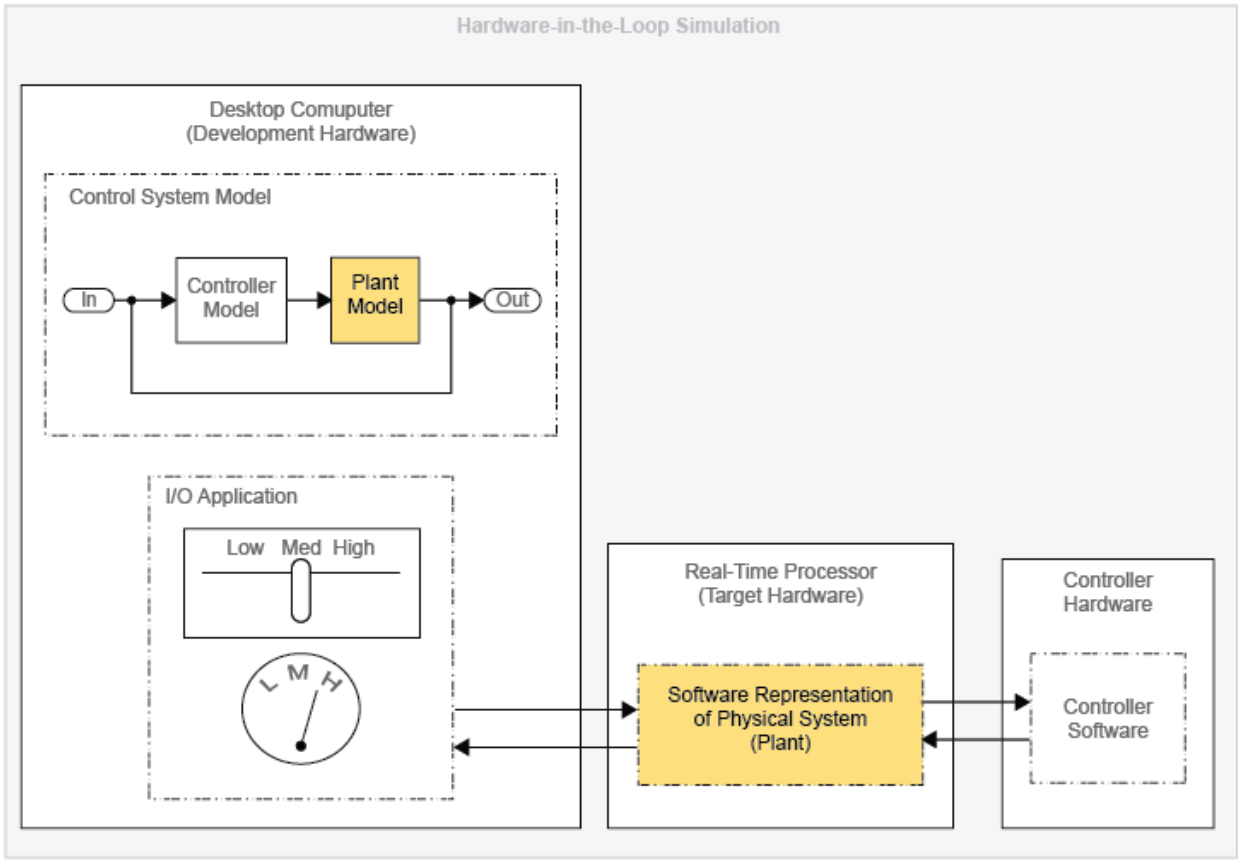
- MIL (Model in the Loop) - проектиране със предавателни характеристики или диаграма на състоянията чрез Simulink/Stateflow. Изясняване на концепцията
- SIL (Software in the Loop) - проектиране на софтуерната част и нейното тестване с помощта на Simulink. Възможност за генериране на C код от проектираната Stateflow диаграма
- PIL (Processor in the Loop) - тестване на софтуерната част в реален процесор, който комуникира с Matlab/Simulink.

- MIL (Model in the Loop) - проектиране със предавателни характеристики или диаграма на състоянията чрез Simulink/Stateflow. Изясняване на концепцията
- SIL (Software in the Loop) - проектиране на софтуерната част и нейното тестване с помощта на Simulink. Възможност за генериране на C код от проектираната Stateflow диаграма
- PIL (Processor in the Loop) - тестване на софтуерната част в реален процесор, който комуникира с Matlab/Simulink.
- HIL (Hardware in the Loop) - тестване на реален модул със симулиране на части от останалата периферия

Hardware-in-the-loop (HIL) е симулация в реално време. При нея се симулира реакцията на проектирания реагулатор в реално време, при реалистични входни сигнали.

- Компютърът (развойния хардуер) съдържа модела, работещ в реално време на обекта за регулиране.
- Интерфейс за връзка с процесора, работещ в реално време на който се съдържа реалния код.

Възможен е вариант компютъра само да визуализира, а процесора моделиращ обекта да е отделен.

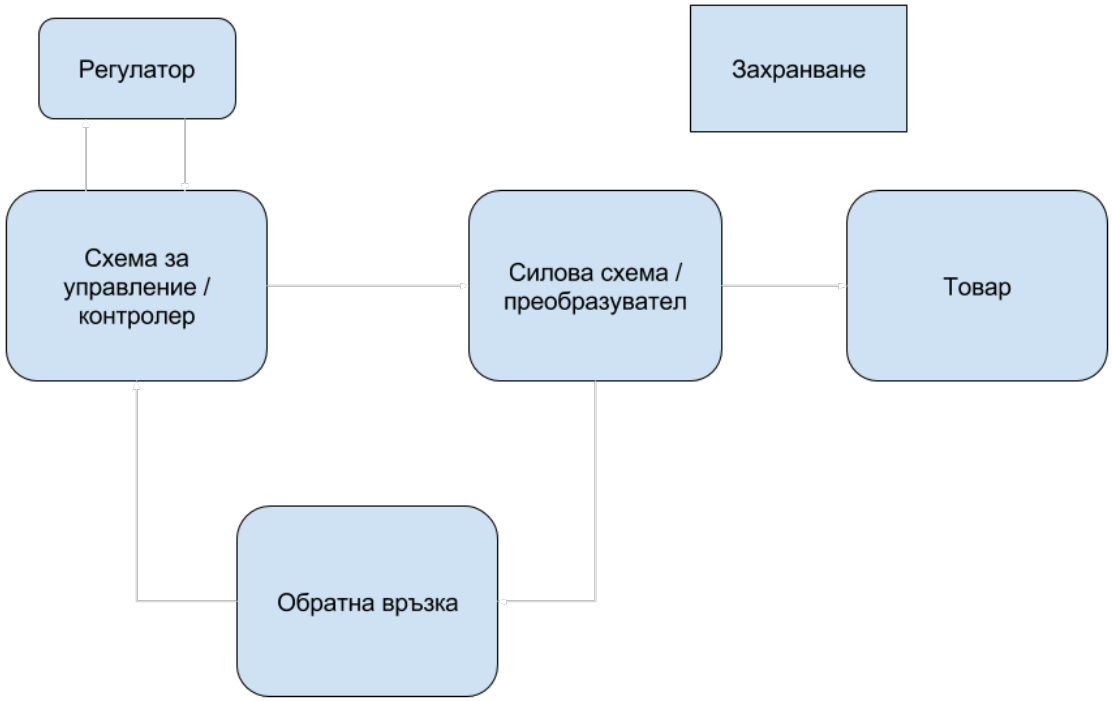


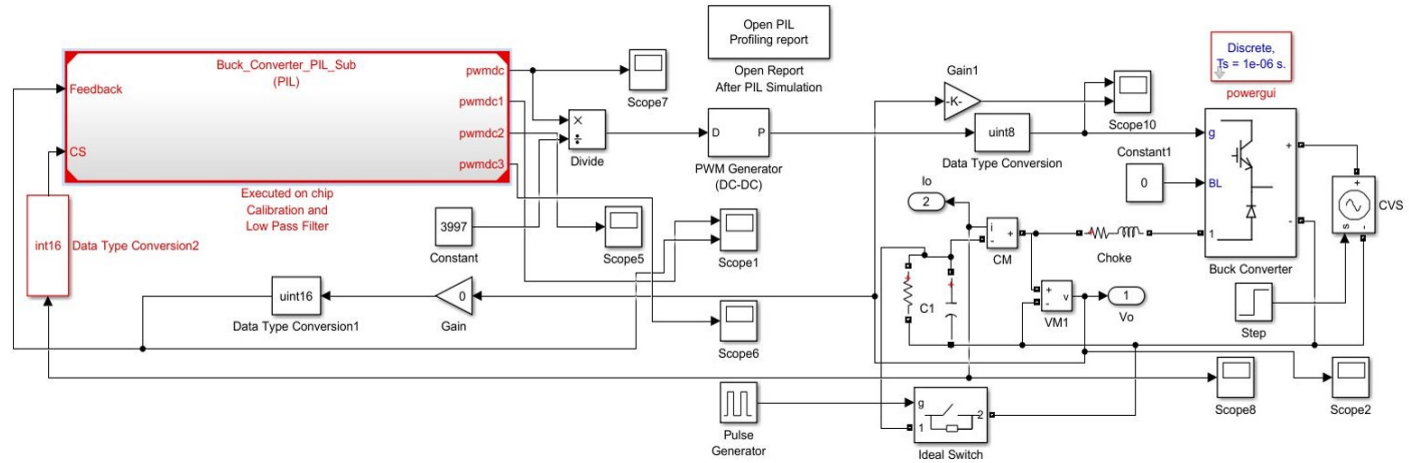
Съществуващи хардуерни решения

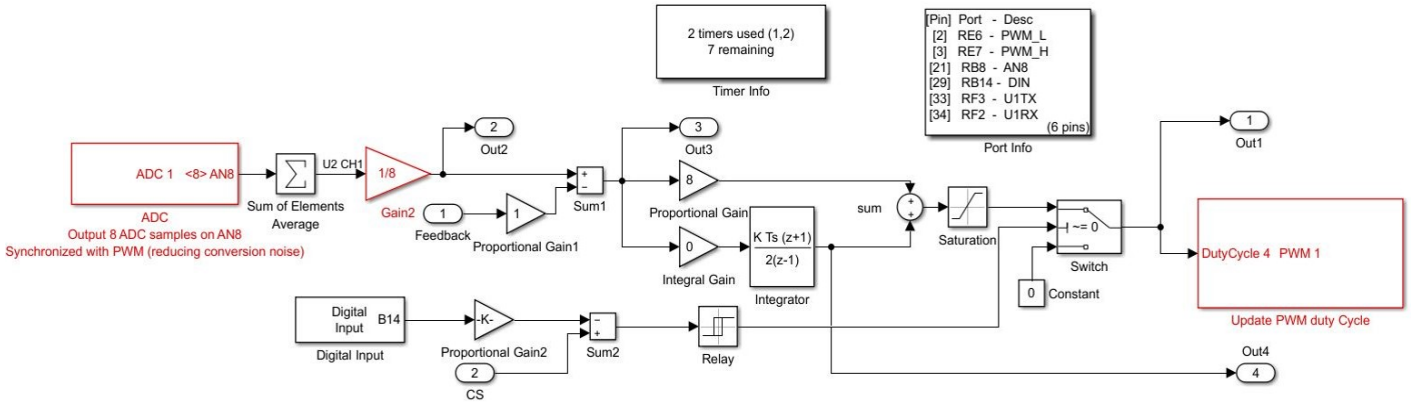
- https://www.opal-rt.com/
- https://www.typhoon-hil.com/
- http://www.plexim.com/products/rt_box
- http://www.ni.com/hil-simulators/

Пример

Да се даде пример за разработването на силов преобразувател.




Buck_Converter_PIL_Sub
(PIL)
Feedback
CS
pwmdc
pwmdc1
pwmdc2
pwmdc3
Executed on chip
Calibration and
Low Pass Filter
int16
Data Type Conversion2
Scope7
Divide
Open PIL
Profiling report
Open Report
After PIL Simulation
PWM Generator
(DC-DC)
3997
Constant
Scope5
Scope1
uint16
Data Type Conversion1
Gain
Scope6
Gain1
uint8
Data Type Conversion
Scope10
Discrete,
Ts = 1e-06 s.
powergui
Constant1
Io
CM
Choke
Buck Converter
CVS
C1
VM1
Vo
Step
Pulse
Generator
Ideal Switch
Scope8
Scope2

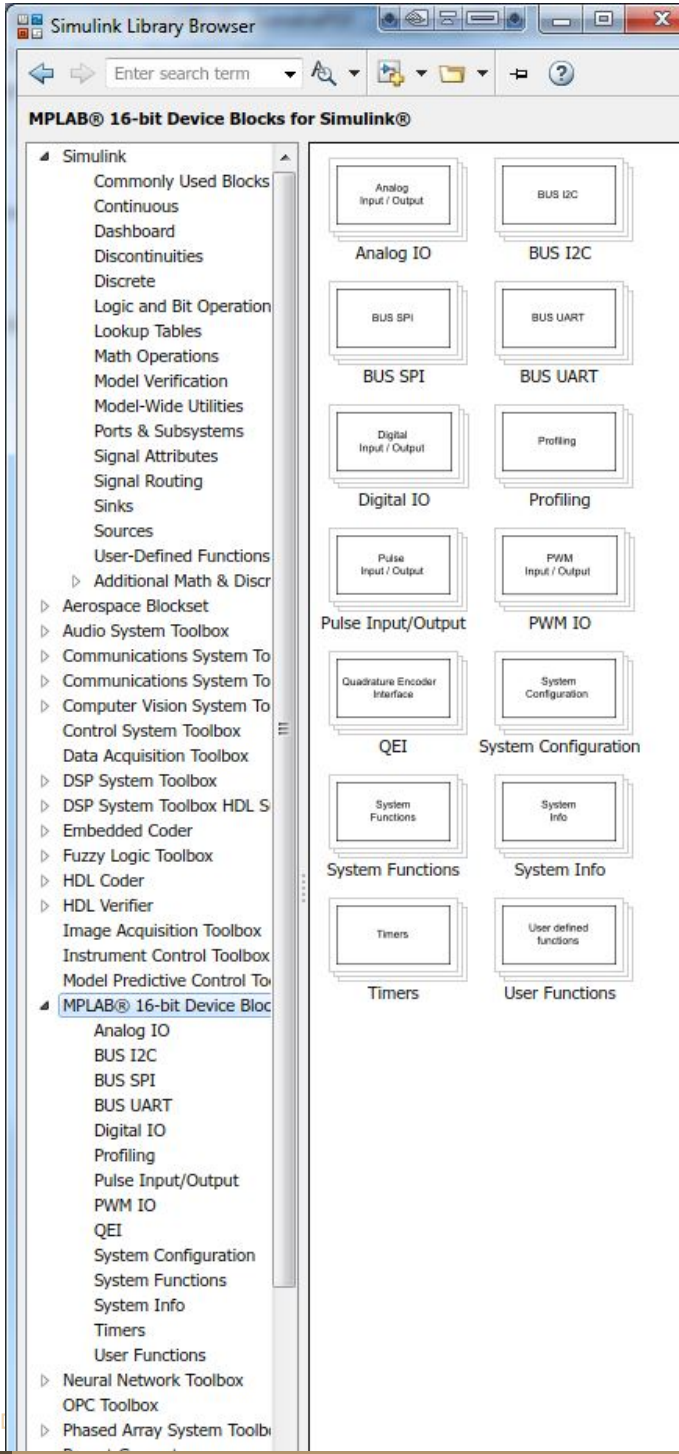

2 timers used (1,2)
7 remaining
Timer Info
[Pin] Port - Desc
[2] RE6 - PWM_L
[3] RE7 - PWM_H
[21] RB8 - AN8
[29] RB14 - DIN
[33] RF3 - U1TX
[34] RF2 - U1RX
(6 pins)
Port Info
ADC 1 <8> AN8
ADC
Output 8 ADC samples on AN8
Synchronized with PWM (reducing conversion noise)
Sum of Elements
Average
U2 CH1
1/8
Gain2
Feedback
Proportional Gain1
Out2
Sum1
Out3
Proportional Gain
Integral Gain
K Ts (z+1)
2(z-1)
Integrator
sum
Saturation
Switch
Constant
Out1
DutyCycle 4 PWM 1
Update PWM duty Cycle
Digital Input
B14
Digital Input
Proportional Gain2
CS
Sum2
Relay
Out4

Модел Базирано Проектиране

Възможности на Simulink на отделните етапи

- За MIL система - Simscape, Simulink, Stateflow библиотеки, които използвахме дотук
- За SIL система - Embedded Coder
- За PIL система - библиотеки на различните производители, например
  - Microchip PIC- Simulink Blocks
  - ARM
  - TI TMS320

- System Configuration: Основни блокове за използвания компилатор и настройки за използвания процесор
- Analog - за настройка на АЦП и компаратор
- BUS I2C,BUS SPI - за настройка на съотвения интерфейс
- BUS Uart - настойка на интерфейса, но и комуникацията с компютъра.
- Digital IO- входове и изходи, настройки
- Pulse Input/Output- За Input Capture или Output compare модула
- PWM - за ШИМ модула



Пример

Да се демонстрира за управлението на светодиод с ШИМ.

На когото му се чете вкъщи . . .

- A. Gilat.
  MATLAB: An Introduction with Applications.
  John Wiley & Sons, Incorporated, 2017.
- E.B. Magrab, S. Azarm, B. Balachandran, J. Duncan, K. Herold, and G. Walsh.
  An Engineers Guide to MATLAB.
  Pearson Education, 2011.