ECE382N.23: Embedded System Design and Modeling

Lecture 4 – State-Based MoCs

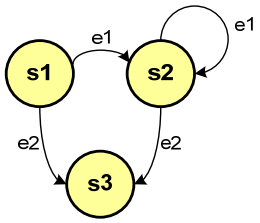
Andreas Gerstlauer
Electrical and Computer Engineering
University of Texas at Austin
gerstl@ece.utexas.edu

Lecture 4: Outline

- **State-based Models of Computation (MoCs)**
  - Finite state machines
  - Hierarchical, concurrent state machines
  - Process state machines
  - Hybrid models

State-Based Models

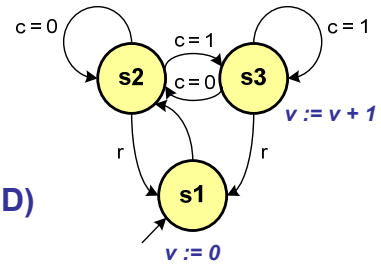
- **Control flow, automata**
  - State and event-driven reactivity
- **Explicit enumeration of computational states**
  - State represents captured history
- **Explicit flow of control**
  - Transitions in reaction to events



- **Stepwise operation of a machine**
  - Cycle-by-cycle hardware behavior
  - Finite number of states
    - Not really Turing complete
- **State-oriented imperative representation**
  - Imperative models as control/data flow graphs (CDFGs)
- **Formal analysis**
  - Reachability, equivalence, …

Finite State Machines

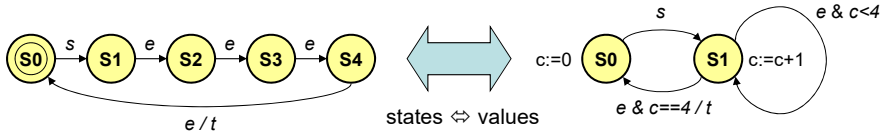
- **Finite State Machine (FSM) model**
  - Synchronous semantics, zero-delay assumption
    - Operational cycle: detect inputs, transition, emit outputs in zero time
  - States *S,* inputs/outputs *I/O,* and state transitions
    - FSM: <*S*, *I*, *O*, *f*, *h*>
    - Next state function *f*: $S \times I \rightarrow S$
    - Non-deterministic: *f* is multi-valued
  - Output function *h*
    - Mealy-type (input-based), *h*: $S \times I \rightarrow O$
    - Moore-type (state-based), *h*: $S \rightarrow O$
      - Convert Mealy to Moore by splitting states per output



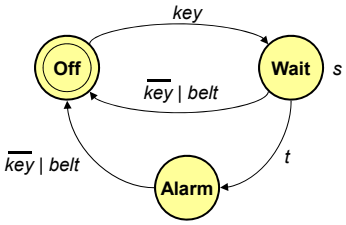
- **Finite State Machine with Data (FSMD)**
  - FSM plus variables *V*
    - FSMD: <*S*, *I*, *O*, *V*, *f*, *h*>
    - Next state function *f*: $S \times V \times I \rightarrow S \times V$
    - Output function *h*: $S \times V \times I \rightarrow O$
  - Computation as control and expressions
    - Controller and datapath of RTL processors

Reducing Complexity

- **FSM with Data (FSMD) / Extended FSM (EFSM)**
  - Mealy counter
    - Implicit self-loops on ē (absence), $f: S \times V \times 2^I \rightarrow S \times V$, $h: S \times V \times 2^I \rightarrow 2^O$

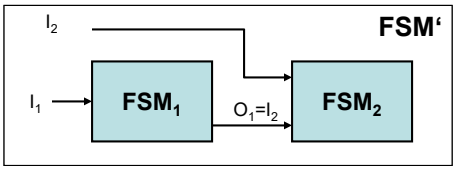


- **Non-Deterministic FSM (NFSM)**
  - Choice in control
    - Implicit self-loops for unspecified conditions? Usually!
    - **Wait**: *belt* & *t* ?
    - Multiple arcs for same condition?
    - Incomplete specification (undecided), unknown behavior (don't care)

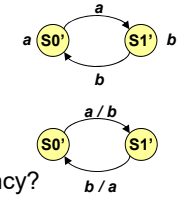


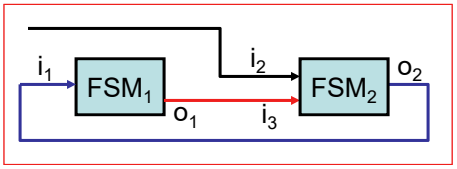
Communicating FSMs

- **FSM composition**
  - FSM': <S',I',O',f',h'>
    - $S' = S_1 \times S_2 = \{\ldots,(\mathbf{Sn_1},\mathbf{Sm_2}),\ldots\}$
    - $I' \subseteq I_1 \cup I_2$
    - $O' \subseteq O_1 \cup O_2$
    - $f': S_1 \times S_2 \times I' \rightarrow S_1 \times S_2$, s.t. $f' \in f_1 \times f_2$
    - $h': S_1 \times S_2 \times I' \rightarrow O'$, s.t. $h' \in h_1 \times h_2$
  - Connectivity constraints
    - Mapping of outputs to inputs: $f_i(s_i, \ldots, h_j(s_j, i_j), \ldots)$, $h_i(s_i, \ldots, h_j(s_j, i_j), \ldots)$
  - Synchronous hypothesis
    - Simultaneous, instantaneous: lock step, zero delay [Synchronous-Reactive]



- **Composability**
  - Moore
    - Delayed
    - Well-defined
  - Mealy
    - Instantaneous
    - Cycles: consistency?

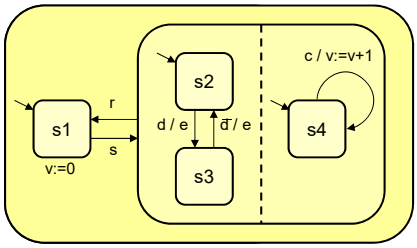




Source: M. Jacome, UT Austin.

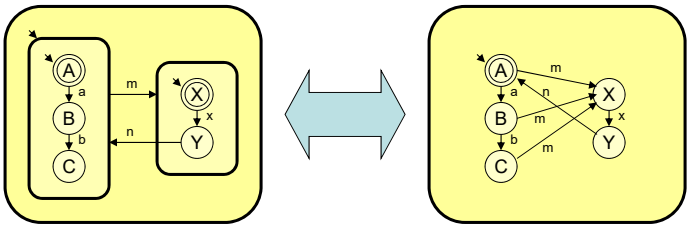
Hierarchical & Concurrent State Machines

- **Superstate FSM with Data (SFSMD)**
  - Hierarchy to organize and reduce complexity
    - Superstates that contain complete state machines each
    - Enter into one and exit from any substate
- **Hierarchical Concurrent FSM (HCFSM) [Harel'87]**
  - Hierarchical and parallel state composition
    - Lock-step concurrent composition and execution
  - Communication through global variables, signals and events
  - ➢ Graphical notation [StateCharts]

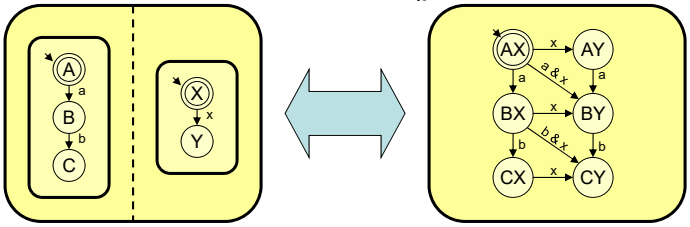


Managing Complexity and State Explosion

- **Hierarchy (OR state)**
  - Reduce the number of transitions (reset behavior)

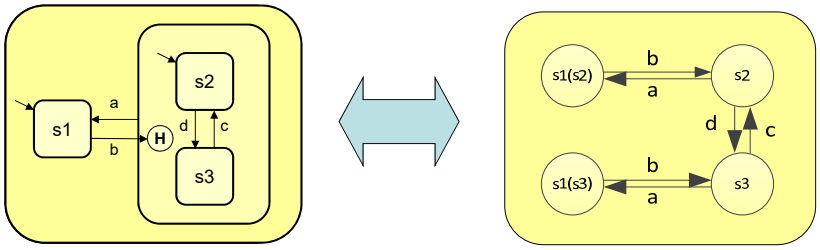


- **Concurrency (AND state)**
  - Reduce the number of states (product state machine)

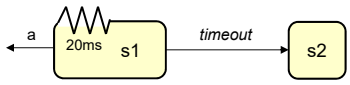


Additional HCFSM Concepts

- **History**
  - Interrupt behavior

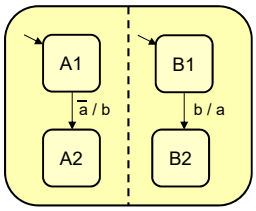


- **Timers**
  - Timeout if no other event after a certain time



HCFSM Semantics (1)

- **Communication between state machines?**
  - ➢ Simultaneous, instantaneous: zero time, broadcast (Mealy)



- **Cycles?**
  - Grandfather paradox (inconsistency)
  - Multiple choices (non-determinism)
- ➢ **Synchronous reactive (SR) model**
  - ➢ Reject all cycles [Argos]
  - ➢ Require unique fixed-point [SyncCharts]

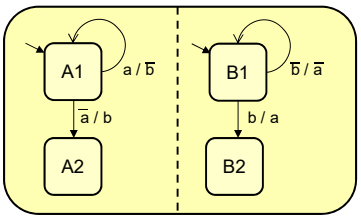
➢ **Two-level model of "time" [StateCharts/Statemate]**

- *N* micro-steps (internal signals) per macro-step (transition)
  - Events posted in next and only in next micro step (values in next step)
  - ➢ "Synchronous": micro = macro step: transition in every step/event (Moore)
  - ➢ "Asynchronous": internal event propagations (Mealy-like, but not truly)
- ➢ Not compositional, but deterministic
  - ➢ Together with other rules, e.g. priority of conflicting transitions/accesses

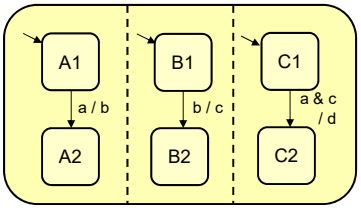
Source: "Statemate Course," K. Baukus

HCFSM Semantics (2)

- **Synchronous-Reactive vs. Statemate (async. micro-steps)**



- SR: reject model, error
- Statemate: endless micro-step loop

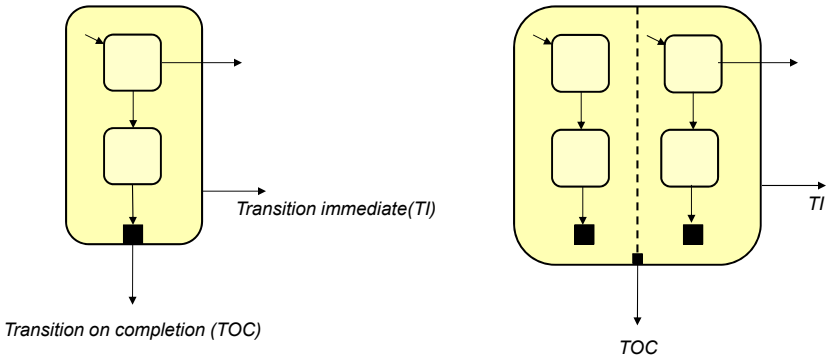


- SR: output *d*
- Statemate: event validity?
  - One micro-step only: no output
  - Globally in macro-step: output d

Source: "Statemate Course," K. Baukus

HCFSM Extensions

- **Transition on completion [UML, SpecCharts]**
  - Final state (completion) of a superstate
  - Transition when *all* final states have been reached



The Esterel Language

- **Synchronous-reactive, imperative language [Berry'83]**
  - Structural hierarchy
    - **module** <*name*> <*b*> **end module**
    - **input** <*ports*>, **output** <*ports*>
  - Behavior
    - Sequential: <*p*> **;** <*q*>
    - Concurrent: <*p*> || <*q*>
  - Multiform time
    - Sequence of logical instants (cycles)
    - Statements take zero time (same instant) or de
    - Delay for one cycle: **pause**
  - Broadcast signals (valued or unvalued)
    - Present or absent in each instant (default to absent in each new cycle)
    - **emit** <*s*>**(**<v>**)**: Make signal <*s*> present in current instant (optional value)
  - Control (branch, loop, exceptions)
    - **present** <*s*> **then** <*p*> **else** <*q*> **end** (in current instant)
    - **loop** <*b*> **end** / **loop** <b> **each** <*s*> (restart when <*s*> is present)
    - **trap** <*s*> **in** <*b*> **end** (use **exit** <*s*> inside <*b*>), **suspend** <*b*> **when** <*s*>

```
module ABRO:
input A, B, R;
output O;

loop
  [ await A || await B ];
  emit O
each R

end
```

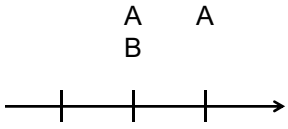
Shortcut for:

```
trap T in
  loop
    pause;
    present A then exit T end
  end loop
end
```


Esterel Semantics

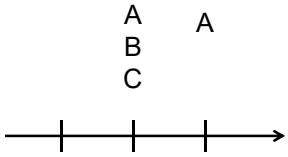
- **Instantaneous communication**
  - Signal emitted in a cycle is visible immediately

```
[
  pause; emit A; pause; emit A
||
  pause; present A then emit B end
]
```



- **Bi-directional communication**
  - Can communicate back and forth in same cycle

```
[
  pause; emit A; present B then emit C end;
  pause; emit A
||
  pause; present A then emit B end
]
```



Source: S. Edwards, Columbia

Esterel Semantics

- **Signal coherence**
  - Signals can not be both absent and present
  - Writers run before readers do

```
present A else emit A;
```

- **Causality**
  - No contradictory or non-deterministic programs (cycles)

```
present S1 else emit S2 end;
present S2 else emit S1 end;
```

- **Instantaneous loops**
  - Loops must have at least one statement with delay

```
loop emit A end
```

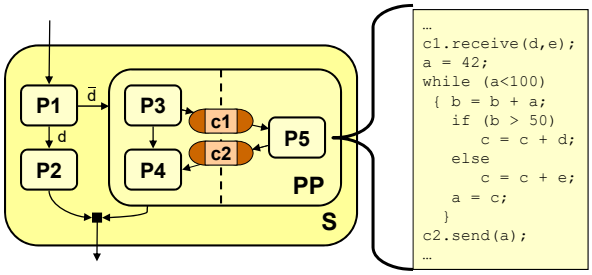
➢ **Erroneous programs, rejected by compiler**
  ➢ Statically verified, guaranteed deterministic

Source: S. Edwards, Columbia

Process and State Based Models

- **From synchronous to asynchronous compositions…**
  - Asynchronous concurrency in HCFSMs [UML]
    - Explicit event queues, deadlock analysis [PetriNet]
  - Processes are state machines
    - Globally asynchronous, locally synchronous (GALS) systems
    - Co-design Finite State Machines (CFSM) [Polis]: mailbox (1-deep buffer)
    - FSMs communicating via unbounded FIFOs [SDL]
  - States are processes
    - FSMs controlling continuous process networks [*Charts, FunState]
    - Imperative leaf states, Program State Machine (PSM) [SpecCharts]

➢ Arbitrary hierarchy
  ➢ Process State Machine (PSM) [SpecC]
  ➢ Heterogeneous MoCs [Ptolemy]



Lecture 4: Summary

- **Models of Computation (MoCs)**
  - Formally express behavior
- **State-based models: FSM(D), HCFSM**
  - States (history, storage) & transitions (reactions)
  - Synchronous concurrency & composition
  - Hierarchy to manage complexity
  - ➢ Control flow dominated, reactive
- ➢ **Hybrid models**
  - ➢ Combination of process and state (data and control flow)
  - ➢ Behavior from specification down to implementation