Synchronous Dataflow Programming

CS684: Embedded Systems
Topic 2

Paritosh Pandya

Indian Institute of Technology, Bombay

February 7, 2021

Uni-mode Controllers

- Controller defined by a single set of equations.
- Exactly one equation for every output and internal variable.
- Equations must be causal. They have deterministic behaviour.
- Each equation specifies the "reset" cycle behaviour and the non-reset cycle behaviour using the arrow operator. E.g.

```
count = (0 -> (pre(count) + 1))
```

0, 1, 2, 3

Programs with resets

```
node display() returns (o,p:int)
let
  reset
      o = 3 -> pre(5 -> (o + 1));
  every (0 -> pre(o)) = 9;
  p = 0 -> pre(p)+2;
tel
```

o	3	5	6	7	8	9	3	5	6	...
p	0	2	4	6	8	10	12	14	16	...

Resetting equations with external trigger

```
node display(rst:bool) returns (o:int)
let
  reset
     o = 3 -> pre(5 -> (o + 1));
  every rst;
tel
```

Multi-mode controllers

- Different sets of equations are applied in different circumstances.
- Conditions which decide which set of equations are chosen are called modes.
- At a time system can be in exactly one mode.
- In each mode each output and internal variable has exactly one equation governing it.
- Each mode works as a separate clock domain with separate memory.
- The logic of mode switching has to be programmed.
  Finite state automata with transitions.

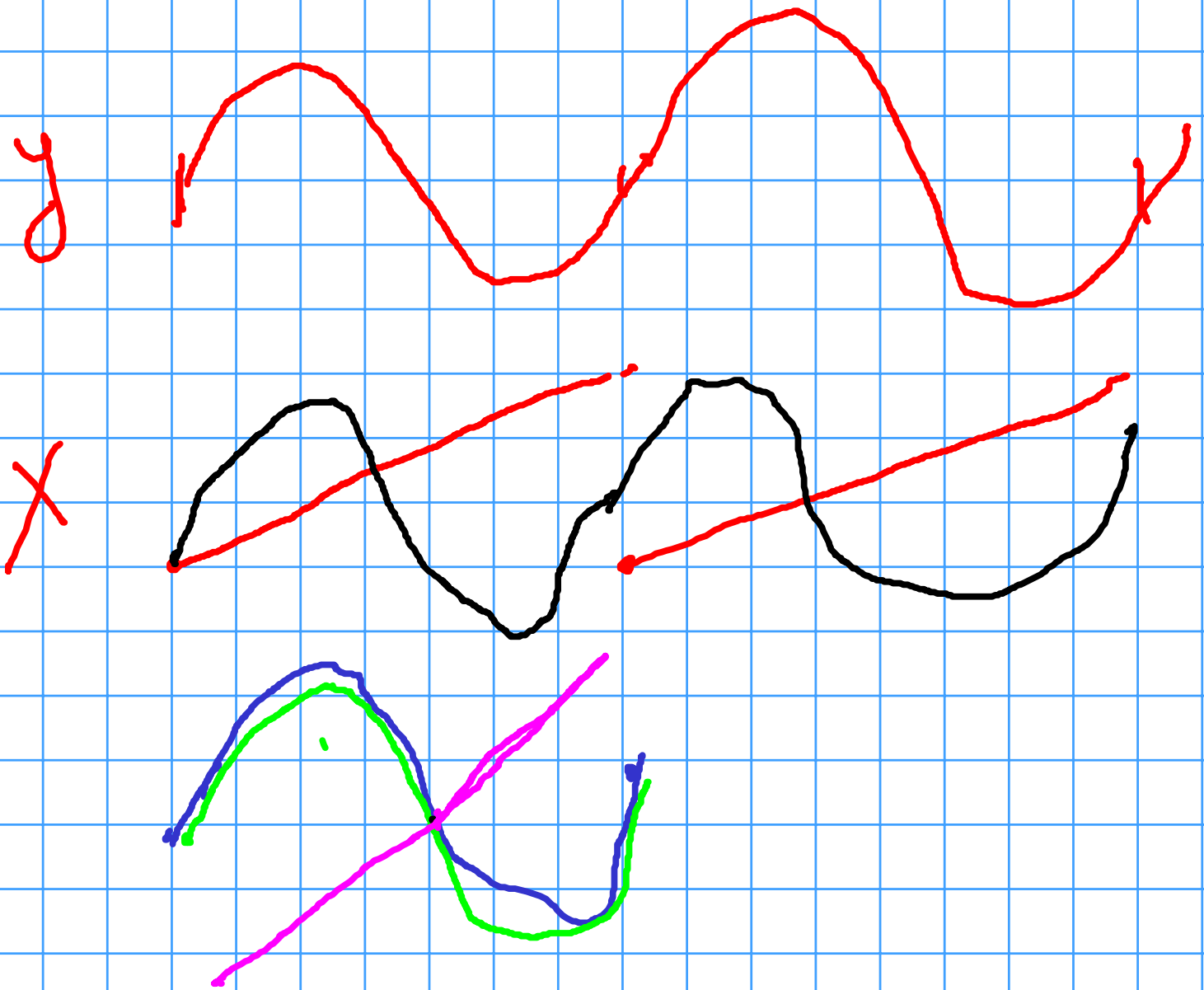
Examples of Multi-modal Systems

Digital Oscilloscope

- choice of y signal (input channel), amplification.
- choice of x signal : sine, ramp, frequency, trigger and phase.

Digital Watch

- Clock mode : outputs current time
- Set clock mode: increments/decrements time based on user key presses.
- Stopwatch mode: outputs time since last reset. Reset on user key press.
- Timer mode: counts down time from initial preset time on press of start button. Beeps when timer becomes 0.
- Set timer mode: sets timer based on user key press.


y
x

Variety of Controller Programmng Languages

- Data dominated Uni-modal with system of equations.
  Lustre, Signal, Simulink, Verilog/VHDL.
- Control dominated Finite state automata (FSM) based.
  State charts, Stateflow, Esterel,
- Mixed Tight integration of data-flow equations and FSM control.
  SCADE, Heptagon, Simulink+Stateflow.

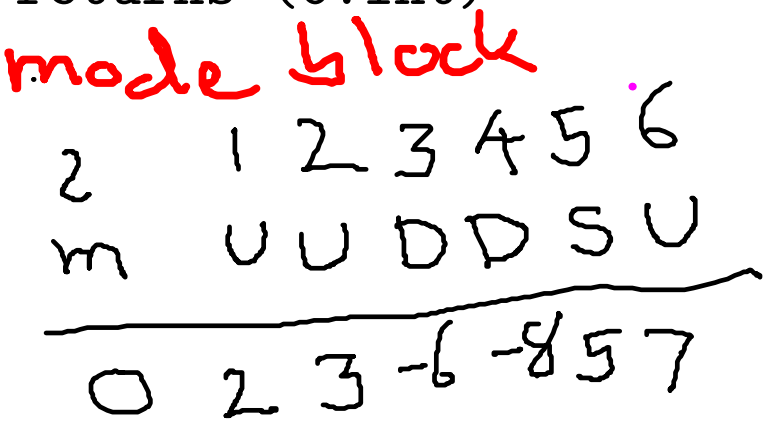
Mixed language

- States are modes.
- Each state has an associated set of equations.
- Transitions specify conditions for state (i.e. mode) change.

Multi-modal Control: the switch construct

```
type modes = Up | Down | Stop

node counter(m: mode; i:int) returns (o:int)
let
   switch m
    | Up   do o = i+1
    | Down do o = -2*i
    | Stop do o = i
   end
tel
```



- Output o has an equation in each mode.
- System is in one mode at each clock cycle.
  The corresponding equation defines the output returned.

Modes as Name spaces and Clock Domains

```
node display(updown:bool) returns (o:int)
let
  switch updown
  | true      var y:int; do y = 100 -> pre(y)+1;
                         o=y
  | false     var y:int; do y = 10 -> pre(y)-1;
                         o=y
  end
tel
```

updown	0	0	0	1	1	1	1	0	1	...
o	10	9	8	100	101	102	103	7	104	...

Sharing Memory between modes: last versus pre

```
node display(updown:bool) returns (o,q:int)
var
    last z: int = 20;
    y:int;
let
  q = z;
  switch updown
| true  do  y = 100 -> pre(y)+1;
                       z = (last z) + 1;
                       o=y
| false do  y = 10  -> pre(y)-1;
                       z = (last z) - 1;
                       o=y
end
tel
```

updown	1	1	1	0	0	0	1	1	1	1	1	0	0	0	0	...
rst	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	...
o	100	101	102	10	9	8	103	104	100	101	102	7	6	5	10	...
q	21	22	23	22	21	20	21	22	23	24	25	24	23	22	21	...

modes with last, pre and reset together

```
node display(updown:bool; rst:bool) returns (o,q:int)
var
    last z: int = 20;      y:int;
let q = z;
switch updown
| true   do  reset
                        y = 100 -> pre(y)+1;
                        z = (last z) + 1;
                        o=y
             every rst
| false  do  y = 10 -> pre(y)-1;
                        z = (last z) - 1;
                        o=y
end
tel
```

updown	1	1	1	0	0	0	1	1	1	1	1	0	0	0	0	...
rst	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	...
o	100	101	102	10	9	8	103	104	100	101	102	7	6	5	10	...
q	21	22	23	22	21	20	21	22	23	24	25	24	23	22	21	...