EEE8068
Real Time Computer Systems
FSM miplementations in C

Dr. Bystrov

School of Electrical Electronic and Computer Engineering

University of Newcastle upon Tyne

Aims and Objectives

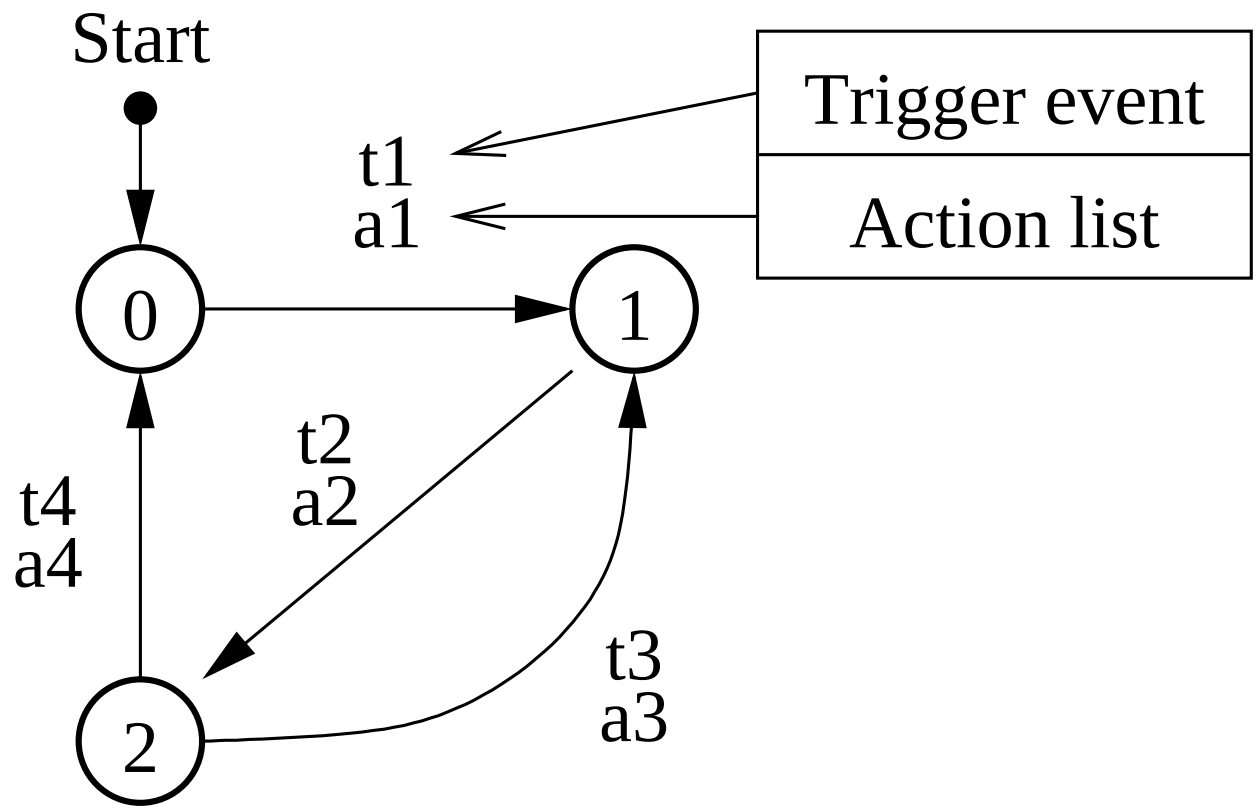
Aim: Implementation of SW systems that have been designed with FSM models

Objectives:

1. Switch-Case implementation style
2. Goto-Label implementation style
3. Finite State Table implementation style

The examples below are simpler than in the book by Williams. Only simple examples are included in the exam; the book is optional, but read it anyway.

The FSM benchmark



- Is it of Moore or Mealy type?
- Remember the formal definition and examples?

Switch-Case Implementation

```
main() {
  enum state {s0,s1,s2}=s0;
  while (1) {
    switch(state) {
      case s0: if (t1)
                    {a1(); state=s1;}
                  break;
      case s1: if (t2)
                    {a2(); state=s2;}
                  break;
      case s2: if (t3)
                    {a3(); state=s1;}
                  else if (t4)
                    {a4(); state=s0;}
    }
}
```

Goto-Label Implementation

```
main() {
  L0: if (t1) {a1(); goto L1;} else goto L0;
  L1: if (t2) {a2(); goto L2;} else goto L1;
  L2: if (t3) {a(3); goto L1;}
      else if (t4) {a4(); goto L0}
      else goto L2;
}
```

- This is faster than Switch-Case!
- Still, the size of the code is proportional to the size of the model.
- Very reliable, because the labels are stored in ROM.

FST Implementation – Data Structures

```
#define states 3
#define triggers 4
struct state
  {
    int next_state;
    action_type action;
  }
struct state state_table[triggers][states]=
  {
    {{1,1},{0,0},{0,0},}, //t1
    {{0,0},{2,2},{0,0},}, //t2
    {{0,0},{0,0},{1,3},}, //t3
    {{0,0},{0,0},{0,4},}, //t4
  };
```

- The FSM is “encoded” into the data structure. Size?

FST Implementation – Code

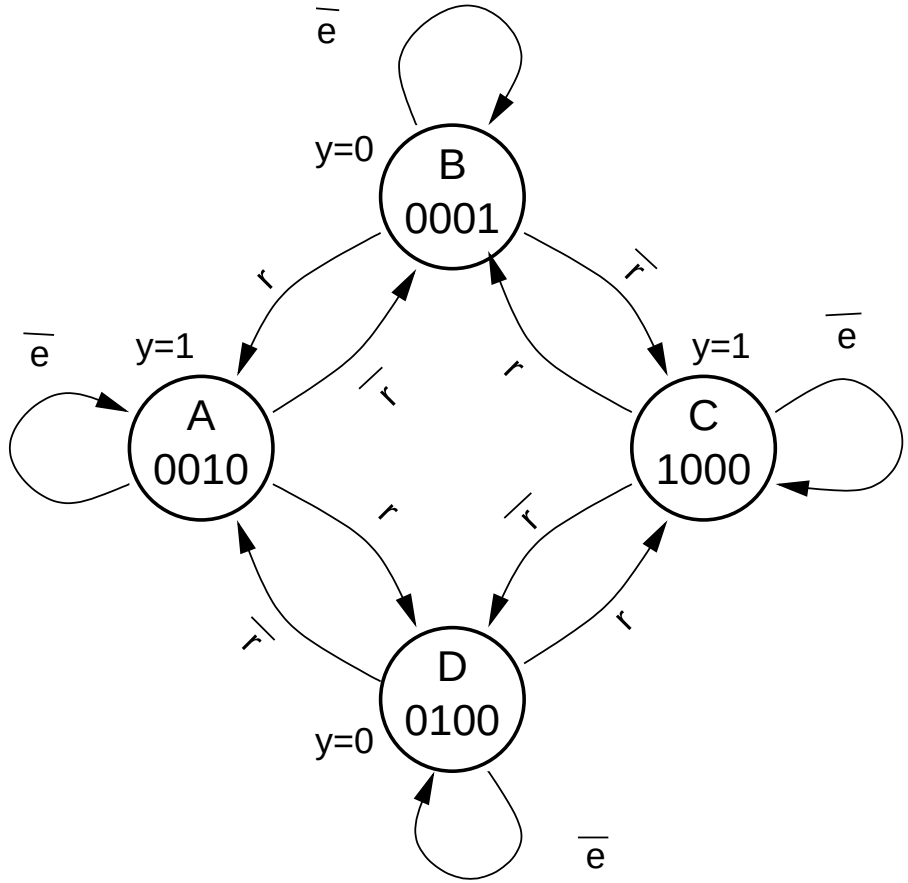
```
void main()
{
  int state=0;
  int t;
  while
    {
      input_trigger(t);
      execute_action(
         state_table[state][t].action);
      current_state=
         state_table[state][t].state;
    }
}
```

- Fast. The code size is independent from the spec size.

Conclusions

- Three implementation styles of FSM are covered
  - Switch-Case
  - Goto-Label
  - Finite State Tables
- They can be used with any language, not just C
- Simple actions can be replaced with action lists
- Actions can be associated with either transitions or states
- Several implementation styles are not covered
  - Direct Sequential Coding – a convoluted version of Goto-Label
  - Object-Oriented approach – specific to C++, the algorithm idea is similar to the above three.

Exercise 1 (exam material!)



- The inputs are *e* and *r*; the output is *y.*
- Implement this FSM in all three styles discussed above.

Exercise 2

- Develop an FSM model of a lift and implement it in the above three styles.
- Four floors
- One button on each floor
- Four buttons on the controller inside the lift
- There are no doors for simplicity