An Implementation of an Elevator System in the IOA Language and Toolset

24/3/2003
Northeastern University
Technical Report: NU-CCIS-0304

Authors:
Paul C. Attie
David H. Lorenz
Onur Aytar

Abstract

In this report, an elevator model consisting of static interfaces and replaceable input and decision units has been created and tested using the IOA Language and Toolset. The results were quite significant, since the components operated as desired, when the pre-conditions are conserved.

Component based programming allows programmers to create independent pieces of software which can work together, or support each other. Either with the purpose of reuse, or with the purpose of independent development, building software from components is quite efficient in terms of time and costs.

Implementing interfaces, components support a set of input and outputs while having a state of their own. IOA Language and Toolset aims to give the component designers create models of their components specifying the signature and the states of the design and check the design with a high level language implementation and preconditions on state transitions.

The feasibility of a component based system can be examined through a common lift model. Widely used and simple on the surface, the lift model, a.k.a the elevator model can be used to evaluate the behavior of a component based software.

Table of Contents

1. Introduction

<u>Statement of purpose:</u>
This report aims to analyze the correctness of a component based system by examining its execution.

<u>Scope of the report:</u>
This report covers the implementation and simulation of component based elevator model in IOA Language and Toolset. Only one model is created and further complement substitutions are not included. This report neither include the proofs of the pre-conditions for methods.

<u>Methods used:</u>
First the model is created with IOA Language according to the system specifications. Then the model is simulated using IOA simulator and the simulation results are checked for correctness.

2. Description of the System

One of the traditional and most widely studied examples in software engineering is the ``lift example'': a building contains F floors and E elevators. Requests for service come from ``panel'' buttons inside an elevator, indicating a request from a passenger within the elevator, and from ``floor'' buttons on a particular floor, indicating a request from a passenger waiting at a particular floor. The latter requests have a direction (up or down) associated with them, and can be satisfied by any of the elevators.

In a system with several elevators operating concurrently, and with requests from the various floors arriving concurrently, the coordination of the elevators and the efficient resolution of the floor requests is quite intricate. Furthermore, a straightforward solution suffers from significant state-explosion, since the global state-space is the product fo the state-spaces of all the elevators, (including all the panel requests), and all the floor requests.

The problem is to design software for the various components of an elevator system. The requirements have both static aspects and dynamic aspects. The static aspects are the interfaces between the input and the decision units.

The dynamic aspects are given by the following required temporal behavioral properties:

1. Safety: an elevator does not attempt to change direction without stopping first,
2. Liveness: every request for service is eventually satisfied, and
3. Efficiency: for a request from a floor button, only one elevator responds.

3. The Elevator System Model Creation

The elevator system model has been prepared in IOA language, aiming to verify the pre-conditions on state transitions. The model directly uses the system specifications except for the composition of controller components into one controller automaton.

A trait Building has been prepared to create the building and placing floors on top of each other, giving each floor a floor number.

The model consists of the following automata:

- Panel,
- Button,
- Motor,
- Controller

and in addition it includes type descriptions for state values.

3.1. Type descriptions

```
type PcDirection = enumeration of u, d, s
type PcStatus = enumeration of m, s, ii
type PcRequests = enumeration of i, p, null
type stateVal = enumeration of idle, up, broken, down
```

Types are defined to preserve the consistency between the specifications and the model.

3.2. Building trait

```
Building(I, J) : trait
  introduces
    first :        -> I
    ups, downs : I -> I
    val :        I -> J
  asserts with i, j : I
    sort I generated by first, downs;
    \E i (downs(i) = first);
    downs(i) = ups(j) <=> i = j;
    downs(ups(i)) = i;
    val(i) = val(j) <=> i = j
  implies with i: I
    downs(ups(i)) = i
```

The building trait is used for creating the aspect of a building. It defines an upstairs, a downstairs and a floor number for the automata it is used in. In the elevator model, the building trait is used in the panel and the button automata, as they represent floors.

3.3. Automaton Panel

```
automaton panel(F: type, f: F)
  assumes Building(F, Int)

  signature
    output reqp(const val(f))
    input  satp(const val(f))
  states
    lit: Bool := false,
    fno: Int := val(f)
  transitions
    output reqp(fl)
       pre ~lit
       eff lit := true
    input  satp(fl)
       eff lit := false
```

Automaton panel uses the Building trait and has as many floor buttons as the Building has. The buttons may be lit or not lit. Whenever a request is made the corresponding button becomes lit and stays lit until the request is satisfied.

3.4. Automaton Button

```
automaton button(FP: type, f: FP)
  assumes Building(FP, Int)

  signature
    output reqb(const val(f), direction: PcDirection)
    input  satb(const val(f), direction: PcDirection)
  states
    dLit: Bool := false,
    uLit: Bool := false,
    fno : Int := val(f)
```

The button automaton is similar to the panel, except that it can serve in up and down directions. It has two buttons for each floor, corresponding to the directions

The directions send requests and are satisfied independently. This allows a complete vision of the workflow of the elevator for the waiting passengers.

Transitions of the automaton follow as:

```
  transitions
    output reqb(fl, direction)
      pre ~(direction = d /\ dLit) /\ ~(direction = u /\ uLit)
      eff if direction = u then
              uLit := true
          else
              dLit := true
          fi
    input  satb(fl, direction)
      eff if direction = u then
              uLit := false
          else
              dLit := false
          fi
```

3.5. Automaton Motor

```
automaton motor
  signature
    input  up,
           down,
           stop
  states
    state: stateVal := idle
  transitions
    input up
      eff if state = down then
              state := broken
          else
              state := up
          fi
    input down
      eff if state = up then
              state := broken
          else
              state := down
          fi
    input stop
      eff if state ~= broken then
              state := idle
          fi
```

The motor acts upon the messages it receives from the controller. It can be either idle, rolling up, rolling down or broken.

3.6. Automaton Controller: The brain behind the system

The controller is a compsition of the interfaces to other components and their corresponding units. Therefore the automaton will be explained in parts.

```
automaton controller

  signature

% External signature

    input    reqp(f: Int),
             reqb(f: Int, dir: PcDirection)
    output   satp(f: Int),
             satb(f: Int, direction: PcDirection),
             up,
             down,
             stop

% Intenal signature

    internal

% CPP actions
             CPP_fsii,
             CPP_fumi,
             CPP_fusi,
             CPP_fump,
             CPP_fusp,
             CPP_fdsp,
             CPP_fdmp,
             CPP_fdsi,
             CPP_fdmi,

% CPC actions
             CPC_fsii,
             CPC_fumi,
             CPC_fusi,
             CPC_fump,
             CPC_fusp,
             CPC_fdsp,
             CPC_fdmp,
             CPC_fdsi,
             CPC_fdmi,
% CBP actions
             CBP_fsii,
             CBP_fumi,
             CBP_fusi,
             CBP_fump,
             CBP_fusp,
             CBP_fdsp,
             CBP_fdmp,
             CBP_fdsi,
             CBP_fdmi,

% CBC actions
             CBC_fsii,
             CBC_fumi,
             CBC_fusi,
             CBC_fump,
             CBC_fusp,
             CBC_fdsp,
             CBC_fdmp,
             CBC_fdsi,
             CBC_fdmi,

% CMM actions
             CMM_fmu,
             CMM_fsu,
             CMM_fsd,
             CMM_fmd,

% CMC actions
             CMC_fmu,
             CMC_fsu,
             CMC_fsd,
             CMC_fmd
```

The internal actions represent the valid states of each component. Each state have pre-conditions under which it can be selected. Since the controller components are completely asyncronous, the internal actions should be performed promptly for the system to work properly.

```
  states

%CPP state variables
    CPP_f: Int := 1,
    CPP_d: PcDirection := s,
    CPP_st: PcStatus := ii,
    CPP_r: PcRequests := null,

%CPC state variables
    CPC_f: Int := 1,
    CPC_d: PcDirection := s,
    CPC_st: PcStatus := ii,
    CPC_r: PcRequests := null,

%CBP state variables
    CBP_f: Int := 1,
    CBP_d: PcDirection := s,
    CBP_st: PcStatus := ii,
    CBP_r: PcRequests := null,

%CBC state variables
    CBC_f: Int := 1,
    CBC_d: PcDirection := s,
    CBC_st: PcStatus := ii,
    CBC_r: PcRequests := null,

%CMM state variables
    CMM_f: Int := 1,
    CMM_d: PcDirection := s,
    CMM_st: PcStatus := ii,

%CMC state variables
    CMC_f: Int := 1,
    CMC_d: PcDirection := s,
    CMC_st: PcStatus := ii,

% Other state variables
    CP_a : Bool := false,
    CB_a : Bool := false,
    CP_b : Bool := false,
    CB_b : Bool := false,
    CP_ps: Set[Int] := {},
    CB_abs: Set[Int] := {},
    CB_dbs: Set[Int] := {}
```

```
transitions
  input  reqp(f: Int)
    eff CP_ps := insert(f, CP_ps);
        if f > CPP_f then
           CP_a := true
        elseif f < CPP_f then
           CP_b := true
        fi
  input  reqb(f: Int, dir: PcDirection)
    eff if dir = u then
           CB_abs := insert(f, CB_abs);
           if f > CBP_f then
              CB_a := true
           elseif f < CBP_f then
              CB_b := true
           fi
        else
           CB_dbs := insert(f, CB_dbs);
           if f > CBP_f then
              CB_a := true
           elseif f < CBP_f then
              CB_b := true
           fi
        fi
  output satp(f)
    pre CPP_f \in CP_ps /\ CPP_st = m
    eff CP_ps := delete(f, CP_ps);
        if size(CP_ps) = 0 then
           CP_a := false;
           CP_b := false
        fi
  output satb(f: Int, dir: PcDirection)
    pre (CBP_f \in CB_abs \/ CBP_f \in CB_dbs) /\ CBP_st = m
    eff if dir = u then
           CB_abs := delete(f, CB_abs)
        else
           CB_dbs := delete(f, CB_dbs)
        fi
  output up
    pre CMM_st = s /\ CMM_d = u
    eff CMM_st := m
  output down
    pre CMM_st = s /\ CMM_d = d
    eff CMM_st := m
  output stop
    pre CMM_st = m
```

```
%CPP state transitions

   internal CPP_fsii
      pre ~CP_a /\ ~CP_b
      eff CPP_d := s;
          CPP_st := ii;
          CPP_r := null
   internal CPP_fumi
      pre (CPP_d = u /\ CPP_r = i) /\ (CPP_st = m \/ CPP_st = s)
      eff CPP_d := u;
          CPP_st := m;
          CPP_r := i;
          CPP_f := CPP_f + 1
   internal CPP_fusi
      pre (CPP_d = s /\ CPP_st = ii) \/ ((CPP_d = u /\ CPP_st = m /\ CPP_r = p)
/\ (CPP_f \in CP_ps /\ ~CP_a /\ CP_b))
      eff CPP_d := u;
          CPP_st := s;
          CPP_r := i
   internal CPP_fump
      pre (CPP_d = u /\ CPP_st = m /\ CPP_r = i) \/ (CPP_d = u /\ CPP_st = s /\
CPP_r = p) \/ (CPP_d = u /\ CPP_st = m /\ CPP_r = p /\ ~(CPP_f \in CP_ps) /\
CP_a)
      eff CPP_d := u;
          CPP_st := m;
          CPP_r := p;
          if ~(CPP_f \in CP_ps) /\ CP_a then
             CPP_f := CPP_f + 1
          fi
   internal CPP_fusp
      pre ((CPP_d = u /\ CPP_st = m /\ CPP_r = p) /\  (CPP_f \in CP_ps /\
CP_a)) \/ ((CPP_d = s /\ CPP_st = ii) /\ CP_a) \/ (CPP_d = d /\ CPP_st = m /\
CPP_r = i /\ CP_a)
      eff CPP_d := u;
          CPP_st := s;
          CPP_r := p;
   internal CPP_fdsp
      pre ((CPP_d = u /\ CPP_st = m /\ CPP_r = i ) /\ (CPP_f \in CP_ps /\
CP_b)) \/ (CPP_d = s /\ CPP_st = ii /\ CP_b) \/ (CPP_d = u /\ CPP_st = m /\
CPP_r = i /\ CP_b)
      eff CPP_d := d;
          CPP_st := s;
          CPP_r := p;
```

```
    internal CPP_fdmp
      pre (CPP_d = d /\ CPP_st = m /\ CPP_r = i) \/ (CPP_d = d /\ CPP_st = s /\
CPP_r = p) \/ (CPP_d = d /\ CPP_st = m /\ CPP_r = p /\ ~(CPP_f \in CP_ps) /\
CP_b)
      eff CPP_d := d;
          CPP_st := m;
          CPP_r := p;
          if ~(CPP_f \in CP_ps) /\ CP_b then
            CPP_f := CPP_f - 1
          fi
    internal CPP_fdsi
      pre (CPP_d = s /\ CPP_st = ii) \/ ((CPP_d = d /\ CPP_st = m /\ CPP_r = p)
/\ (CPP_f \in CP_ps /\ CP_a /\ ~CP_b))
      eff CPP_d := d;
          CPP_st := s;
          CPP_r := i
    internal CPP_fdmi
      pre (CPP_d = d /\ CPP_r = i) /\ (CPP_st = m \/ CPP_st = s)
      eff CPP_d := d;
          CPP_st := m;
          CPP_r := i;
          CPP_f := CPP_f - 1
```

CPP state transitions is a good example for explaining the system.

All valid states are defined as transitions in the automaton description. Each state has pre-conditions, ensuring that they can not be selected arbitrarily. In the implementations the state transitions will be invoked by other components of the controller, whereas in this model we determine the workflow and validity of the transitions.

The other components follow as:

```
% CPC STATE TRANSITIONS %

    internal CPC_fsii
      pre (CPC_st = m /\ CPC_r = p /\ CMM_f = CPC_f) \/ (CPC_d = d /\
CPC_st = m /\ CPC_r = p /\ (CMM_f = CPC_f /\ CMM_d = d)) \/ (CPC_d = u /\
CPC_st = m /\ CPC_r = i) \/ (CPC_d = d /\ CPC_st = m /\ CPC_r = i)
      eff CPC_d := s;
          CPC_st := ii;
          CPC_r := null
```

```
  internal CPC_fumi
    pre (CPC_d = u /\ CPC_r = i /\ CPC_st = m /\ CMM_f = CPC_f /\ CMM_d
= u) \/ (CPC_d = u /\ CPC_r = i /\ CPC_st = s /\ CMM_f = (CPC_f + 1) /\
CMM_d = u)
    eff CPC_d := u;
        CPC_st := m;
        CPC_r := i;
        CPC_f := CPC_f + 1
  internal CPC_fusi
    pre (CPC_d = u /\ CPC_st = m /\ CPC_r = p /\ CMC_f = CPC_f /\ CPC_d
= u) \/ (CPC_d = s /\ CPC_st = ii /\ CBP_d = u /\ CBP_r = p)
    eff CPC_d := u;
        CPC_st := s;
        CPC_r := i
  internal CPC_fump
    pre (CPC_d = u /\ CPC_st = m /\ CPC_r = i) \/ (CPC_d = u /\ CPC_st = s /\
CPC_r = p) \/ (CPC_d = u /\ CPC_st = m /\ CPC_r = p /\ CMM_d = u)
    eff CPC_d := u;
        CPC_st := m;
        CPC_r := p;
        if CMM_f = CPC_f /\ CMM_d = u then
          CPC_f := CPC_f + 1
        fi
  internal CPC_fusp
    pre (CPC_d = u /\ CPC_st = m /\ CPC_r = p /\ CMC_f = CPC_f /\ CMC_d
= u) \/ ((CPC_d = s /\ CPC_st = ii) /\ ((CBC_f = CPC_f /\ CBC_d = s /\ CBC_st
= ii) \/ (CBC_f = CPC_f /\ CBC_d = u /\ CBC_r = p))) \/ (CPC_d = d /\ CPC_st =
m /\ CPC_r = i /\ CMC_d = u /\ CMC_st = s)
    eff CPC_d := u;
        CPC_st := s;
        CPC_r := p;
        if CMM_f = (CPC_f + 1) /\ CMM_d = u then
          CPC_f := CPC_f + 1;
          CPC_st := m
        fi
  internal CPC_fdsp
    pre (((CPC_d = u /\ CPC_st = m /\ CPC_r = i) \/ (CPC_d = u /\ CPC_st =
m /\ CPC_r = p /\ CMC_f = CPC_f /\ CMC_d = d /\ CMC_st = s)) \/ ((CPC_d =
s /\ CPC_st = ii) /\ ((CBC_f = CPC_f /\ CBC_d = s /\ CBC_st = ii) \/ (CBC_f =
CPC_f /\ CBC_d = d /\ CBC_r = p))))
    eff CPC_d := d;
        CPC_st := s;
        CPC_r := p;
        if CMM_f = (CPC_f - 1) /\ CMM_d = d then
          CPC_f := CPC_f - 1;
          CPC_st := m
        fi
```

```
   internal CPC_fdmp
      pre (CPC_d = d /\ CPC_st = m /\ CPC_r = i) \/ (CPC_d = d /\ CPC_st = s /\
CPC_r = p) \/ (CPC_d = d /\ CPC_st = m /\ CPC_r = p /\ CMM_d = d)
      eff CPC_d := d;
          CPC_st := m;
          CPC_r := p;
          if CMM_f = CPC_f /\ CMM_d = d then
             CPC_f := CPC_f - 1
          fi
   internal CPC_fdsi
      pre (CPC_d = d /\ CPC_st = m /\ CPC_r = p /\ CMC_f = CPC_f /\ CMC_d
= d)  \/ (CPC_d = s /\ CPC_st = ii /\ CBP_d = d /\ CBP_r = p)
      eff CPC_d := d;
          CPC_st := s;
          CPC_r := i
   internal CPC_fdmi
      pre (CPC_d = d /\ CPC_r = i /\ CPC_st = m /\ CMM_f = CPC_f /\ CMM_d
= d) /\ (CPC_d = d /\ CPC_r = i /\ CPC_st = s /\ CMM_f = (CPC_f - 1) /\
CMM_d = d)
      eff CPC_d := d;
          CPC_st := m;
          CPC_r := i;
          CPC_f := CPC_f - 1

% CBP STATE TRANSITIONS %

   internal CBP_fsii
      pre size(CB_abs) = 0 /\ size(CB_dbs) = 0
      eff CBP_d := s;
          CBP_st := ii;
          CBP_r := null
   internal CBP_fumi
      pre (CBP_d = u /\ CBP_r = i) /\ (CBP_st = m \/ CBP_st = s)
      eff CBP_d := u;
          CBP_st := m;
          CBP_r := i;
          CBP_f := CBP_f + 1
   internal CBP_fusi
      pre (CBP_d = s /\ CBP_st = ii) \/ ((CBP_d = u /\ CBP_st = m /\ CBP_r = p)
/\ (CBP_f \in CB_abs /\ ~CB_a /\ CB_b))
      eff CBP_d := u;
          CBP_st := s;
          CBP_r := i
   internal CBP_fump
      pre (CBP_d = u /\ CBP_st = m /\ CBP_r = i) \/ (CBP_d = u /\ CBP_st = s /\
CBP_r = p) \/ (CBP_d = u /\ CBP_st = m /\ CBP_r = p)
      eff CBP_d := u;
          CBP_st := m;
          CBP_r := p;
          if ~(CBP_f \in CB_abs) /\ CB_a then
             CBP_f := CBP_f + 1
          fi
   internal CBP_fusp
      pre (CBP_d = u /\ CBP_st = m /\ CBP_r = p /\ CB_a) \/ (CBP_d = s /\
CBP_st = ii /\ CB_a) \/ (CBP_d = d /\ CBP_st = m /\ CBP_r = i /\ CB_a)
      eff CBP_d := u;
          CBP_st := s;
          CBP_r := p
   internal CBP_fdmi
      pre (CBP_d = d /\ CBP_r = i) /\ (CBP_st = m \/ CBP_st = s)
      eff CBP_d := d;
          CBP_st := m;
          CBP_r := i;
          CBP_f := CBP_f - 1
   internal CBP_fdsi
      pre (CBP_d = s /\ CBP_st = ii) \/ ((CBP_d = d /\ CBP_st = m /\ CBP_r = p)
/\ (CBP_f \in CB_dbs /\ CB_a /\ ~CB_b))
      eff CBP_d := d;
          CBP_st := s;
          CBP_r := i
   internal CBP_fdmp
      pre (CBP_d = d /\ CBP_st = m /\ CBP_r = i) \/ (CBP_d = d /\ CBP_st = s /\
CBP_r = p) \/  (CBP_d = d /\ CBP_st = m /\ CBP_r = p)
      eff CBP_d := d;
          CBP_st := m;
          CBP_r := p;
          if ~(CBP_f \in CB_dbs) /\ CB_b then
             CBP_f := CBP_f - 1
          fi
   internal CBP_fdsp
      pre (CBP_d = d /\ CBP_st = m /\ CBP_r = p /\ CB_b) \/ ((CBP_d = s /\
CBP_st = ii) /\ CB_b) \/ (CBP_d = u /\ CBP_st = m /\ CBP_r = i /\ CB_b)
      eff CBP_d := d;
          CBP_st := s;
          CBP_r := p
```

```
% CBC STATE TRANSITIONS %

   internal CBC_fsii
     pre (CBC_st = m /\ CBC_r = p /\ CMM_f = CBC_f) \/ ((CBC_d = d /\
CBC_st = m /\ CBC_r = p /\ (CMM_f = CBC_f /\ CMM_d = d))) \/ (CBC_d = u /\
CBC_st = m /\ CBC_r = i) \/ (CBC_d = d /\ CBC_st = m /\ CBC_r = i)
     eff CBC_d := s;
        CBC_st := ii;
        CBC_r := null
   internal CBC_fumi
     pre (CBC_d = u /\ CBC_st = m /\ CBC_r = i /\ CMM_f = CBC_f /\ CMM_d
= u) \/ (CBC_d = u /\ CBC_st = s /\ CBC_r = i /\ CMM_f = (CBC_f + 1) /\
CMM_d = u)
     eff CBC_d := u;
        CBC_st := m;
        CBC_r := i;
        CBC_f := CBC_f + 1
   internal CBC_fusi
     pre (CBC_d = u /\ CBC_st = m /\ CBC_r = p /\ CMC_f = CBC_f /\ CBC_d
= u) \/ (CBC_d = s /\ CBC_st = ii /\ CPC_f = CBC_f /\ CPC_d = u /\ CPC_r = p)
     eff CBC_d := u;
        CBC_st := s;
        CBC_r := i
   internal CBC_fump
     pre (CBC_d = u /\ CBC_st = m /\ CBC_r = i) \/ (CBC_d = u /\ CBC_st = s /\
CBC_r = p /\ CMC_f = (CBC_f + 1) /\ CMC_d = u) \/ (CBC_d = u /\ CBC_st = m
/\ CBC_r = p /\ CMC_f = CBC_f /\ CMC_d = u)
     eff CBC_d := u;
        CBC_st := m;
        CBC_r := p;
        if CMC_d = u then
          CBC_f := CBC_f + 1
        fi
   internal CBC_fusp
     pre (CBC_d = u /\ CBC_st = m /\ CBC_r = p /\ CMC_f = CBC_f /\ CMC_d
= u) \/ ((CBC_d = s /\ CBC_st = ii) /\ ((CPC_f = CBC_f /\ CPC_d = s /\ CPC_st
= ii) \/ (CPC_f = CBC_f /\ CPC_d = u /\ CPC_r = p))) \/ (CBC_d = d /\ CBC_st =
m /\ CBC_r = i /\ CMC_d = u /\ CMC_st = s)
     eff CBC_d := u;
        CBC_st := s;
        CBC_r := p
```

```
    internal CBC_fdmi
       pre (CBC_d = d /\ CBC_st = m /\ CBC_r = i /\ CMM_f = CBC_f /\ CMM_d
= d) \/ (CBC_d = d /\ CBC_st = s /\ CBC_r = i /\ CMM_f = (CBC_f - 1) /\
CMM_d = d)
       eff CBC_d := d;
           CBC_st := m;
           CBC_r := i;
           CBC_f := CBC_f - 1
    internal CBC_fdsi
       pre (CBC_d = d /\ CBC_st = m /\ CBC_r = p /\ CMC_f = CBC_f /\ CBC_d
= d) \/ (CBC_d = s /\ CBC_st = ii /\ CPC_f = CBC_f /\ CPC_d = d /\ CPC_r = p)
       eff CBC_d := d;
           CBC_st := s;
           CBC_r := i
    internal CBC_fdmp
       pre (CBC_d = d /\ CBC_st = m /\ CBC_r = i) \/ (CBC_d = d /\ CBC_st = s /\
CBC_r = p /\ CMC_f = (CBC_f - 1) /\ CMC_d = d) \/ (CBC_d = d /\ CBC_st = m
/\ CBC_r = p /\ CMC_f = CBC_f /\ CMC_d = d)
       eff CBC_d := d;
           CBC_st := m;
           CBC_r := p;
           if CMC_f = (CBC_f - 1) /\ CMC_d = d then
              CBC_f := CBC_f - 1
           fi
    internal CBC_fdsp
       pre (CBC_d = d /\ CBC_st = m /\ CBC_r = p /\ CMC_f = CBC_f /\ CMC_d
= d) \/ ((CBC_d = s /\ CBC_st = ii) /\ ((CPC_f = CBC_f /\ CPC_d = s /\ CPC_st
= ii) \/ (CPC_f = CBC_f /\ CPC_d = d /\ CPC_r = p)))  \/ (CBC_d = u /\ CBC_st
= m /\ CBC_r = i /\ CMC_d = d /\ CMC_st = s)
       eff CBC_d := d;
           CBC_st := s;
           CBC_r := p

% CMM STATE TRANSITIONS %

    internal CMM_fmu
       pre (CMM_st = m \/ CMM_st = s) /\ CMM_d = u
       eff CMM_f := CMM_f + 1;
           CMM_st := m;
           CMM_d := u
    internal CMM_fsu
       pre (CMM_st = s /\ CMM_d = d) \/ (CMM_st = m /\ CMM_d = u) \/ CMM_st
= ii
       eff CMM_st := s;
           CMM_d := u
    internal CMM_fsd
        pre (CMM_st = s /\ CMM_d = u) \/ (CMM_st = m /\ CMM_d = d) \/ CMM_st
= ii
        eff CMM_st := s;
            CMM_d := d
    internal CMM_fmd
        pre (CMM_st = m \/ CMM_st = s) /\ CMM_d = d
        eff CMM_f := CMM_f - 1;
            CMM_st := m;
            CMM_d := d

% CMC STATE TRANSITIONS %

    internal CMC_fmu
        pre ((CMC_st = m /\ CMC_d = u) /\ (CPP_f = (CMC_f + 1) /\ CBP_f =
(CMC_f + 1))) \/ ((CMC_st = s /\ CMC_d = u) /\ ((CPP_f = CMC_f /\ CPP_d = u
/\ CPP_r = p) \/ (CBP_f = CMC_f /\ CBP_d = u /\ CBP_r = p) \/ (CPP_f =
(CMC_f + 1) /\ CPP_d = u /\ CPP_r = p) \/ (CBP_f = (CMC_f + 1) /\ CBP_d = u
/\ CBP_r = p)))
        eff CMC_f := CMC_f + 1;
            CMC_st := m;
            CMC_d := u
    internal CMC_fsu
        pre ((CMC_st = m /\ CMC_d = u) /\ (((CPP_f = CMC_f /\ CPP_d = u /\
CPP_st = s /\ (CPP_r = p \/ CPP_r = i)) \/ (CPP_f = CMC_f /\ CPP_d = s /\
CPP_st = ii)) \/ ((CBP_f = CMC_f /\ CBP_d = u /\ CBP_st = s /\ (CBP_r = p \/
CBP_r = i)) \/ (CBP_f = CMC_f /\ CBP_d = s /\ CBP_st = ii)))) \/ ((CMC_st = s /\
CMC_d = d) /\ ((CPP_f = CMC_f /\ CPP_d = u /\ CPP_r = p) \/ ((CPP_f =
CMC_f \/ CPP_f = CMC_f + 1) /\ CPP_d = d /\ CPP_r = i) \/ ((CPP_f = CMC_f
\/ CPP_f = CMC_f + 1) /\ CPP_d = s /\ CPP_st = ii)) /\ ((CBP_f = CMC_f /\
CBP_d = u /\ CBP_r = p) \/ ((CBP_f = CMC_f \/ CBP_f = CMC_f + 1) /\ CBP_d
= d /\ CBP_r = i) \/ ((CBP_f = CMC_f \/ CBP_f = CMC_f + 1) /\ CBP_d = s /\
CBP_st = ii)) /\ ~((CPP_f = CMC_f /\ CPP_d = s /\ CPP_st = ii) /\ (CBP_f =
CMC_f /\ CBP_d = s /\ CBP_st = ii))) \/ CMC_st = ii
        eff CMC_st := s;
            CMC_d := u
    internal CMC_fsd
        pre ((CMC_st = m /\ CMC_d = d) /\ (((CPP_f = CMC_f /\ CPP_d = d /\
CPP_st = s /\ (CPP_r = p \/ CPP_r = i)) \/ (CPP_f = CMC_f /\ CPP_d = s /\
CPP_st = ii)) \/ ((CBP_f = CMC_f /\ CBP_d = d /\ CBP_st = s /\ (CBP_r = p \/
CBP_r = i)) \/ (CBP_f = CMC_f /\ CBP_d = s /\ CBP_st = ii)))) \/ ((CMC_st = s /\
CMC_d = u) /\ ((CPP_f = CMC_f /\ CPP_d = d /\ CPP_r = p) \/ ((CPP_f =
CMC_f \/ CPP_f = CMC_f - 1) /\ CPP_d = u /\ CPP_r = i) \/ ((CPP_f = CMC_f \/
CPP_f = CMC_f - 1) /\ CPP_d = s /\ CPP_st = ii)) /\ ((CBP_f = CMC_f /\ CBP_d
= d /\ CBP_r = p) \/ ((CBP_f = CMC_f \/ CBP_f = CMC_f - 1) /\ CBP_d = u /\
CBP_r = i) \/ ((CBP_f = CMC_f \/ CBP_f = CMC_f - 1) /\ CBP_d = s /\ CBP_st
= ii)) /\ ~((CPP_f = CMC_f /\ CPP_d = s /\ CPP_st = ii) /\ (CBP_f = CMC_f /\
CBP_d = s /\ CBP_st = ii))) \/ CMC_st = ii
```

```
      eff CMC_st := s;
          CMC_d := d
    internal CMC_fmd
      pre ((CMC_st = m /\ CMC_d = d) /\ (CPP_f = (CMC_f - 1) /\ CBP_f =
(CMC_f - 1))) \/ ((CMC_st = s /\ CMC_d = d) /\ ((CPP_f = CMC_f /\ CPP_d = d
/\ CPP_r = p) \/ (CBP_f = CMC_f /\ CBP_d = d /\ CBP_r = p) \/ (CPP_f =
(CMC_f - 1) /\ CPP_d = d /\ CPP_r = p) \/ (CBP_f = (CMC_f - 1) /\ CBP_d = d /\
CBP_r = p)))
      eff CMC_f := CMC_f - 1;
          CMC_st := m;
          CMC_d := d
```

As it can be seen in the automaton definition, all composed components have their transitions as internal actions. The invocation of these methods will be performed it the verification stage.

4. System Verification

In order to verify the system, a test case is prepared, which covers a sample execution trace. Although the test suite does not cover all possible states, the covered states can be reflected to the opposite direction, working with the same principles.

The test suite fires actions in the order they should be allowed in the implementation and checks if the pre-conditions hold or fail for the invoked state.

```
  schedule do
    fire input    reqp(4);
    fire internal CPP_fusp;
    fire internal CMC_fsu;
    fire internal CMM_fsu;
    fire internal CPC_fusp;
    fire internal CBC_fusi;
    fire internal CBP_fusi;
    fire output   up;
    fire internal CPP_fump;
    fire internal CPC_fump;
    fire internal CMM_fmu;
    fire internal CMC_fmu;
    fire internal CBC_fumi;
    fire internal CBP_fumi;
    fire internal CPP_fump;
    fire internal CBC_fumi;
    fire internal CBP_fumi;
    fire internal CPC_fump;
    fire internal CMC_fmu;
```

```
fire internal CMM_fmu;
fire internal CPP_fump;
fire internal CBC_fumi;
fire internal CBP_fumi;
fire internal CPC_fump;
fire internal CMC_fmu;
fire internal CMM_fmu;
fire output   satp(4);
fire internal CPP_fsii;
fire internal CBC_fsii;
fire internal CBP_fsii;
fire output   stop;
fire input    reqp(2);
fire internal CPP_fdsp;
fire internal CMC_fsu;
fire internal CMM_fsu;
fire internal CMC_fsd;
fire internal CMM_fsd;
fire internal CPC_fdsp;
fire internal CBC_fdsi;
fire internal CBP_fdsi;
fire output   down;
fire internal CPP_fdmp;
fire input    reqb(6, d);
fire internal CPC_fdmp;
fire internal CMC_fmd;
fire internal CMM_fmd;
fire internal CBC_fdmi;
fire internal CBP_fdmi;
fire internal CPP_fdmp;
fire input    reqb(5, u);
fire internal CPC_fdmp;
fire internal CBC_fdmi;
fire internal CBP_fdmi;
fire internal CMC_fmd;
fire internal CMM_fmd;
fire output   satp(2);
fire output   stop;
fire internal CMM_fsd;
fire internal CBP_fusp;
fire internal CPP_fsii;
fire internal CMC_fsd;
fire internal CMM_fsu;
fire internal CMC_fsu;
fire internal CBC_fusp;
```

```
  fire internal CPC_fsii;
  fire output   up;
  fire internal CMC_fmu;
  fire internal CMM_fmu;
  fire internal CBP_fump;
  fire internal CBC_fump;
  fire internal CPP_fusi;
  fire internal CPC_fusi;
  fire internal CPP_fumi;
  fire internal CPC_fumi;
  fire internal CBP_fump;
  fire internal CBC_fump;
  fire internal CPP_fumi;
  fire internal CPC_fumi;
  fire internal CMM_fmu;
  fire internal CMC_fmu;
  fire internal CBP_fump;
  fire internal CBC_fump;
  fire internal CPP_fumi;
  fire internal CPC_fumi;
  fire internal CMM_fmu;
  fire internal CMC_fmu;
  fire output   stop;
  fire output   satb(CBC_f, CBC_d);
  fire internal CBP_fusp;
  fire internal CBC_fusp;
  fire internal CMM_fsu;
  fire internal CMC_fsu;
  fire output   up;
  fire internal CPP_fumi;
  fire internal CPC_fumi;
  fire internal CMM_fmu;
  fire internal CMC_fmu;
  fire internal CBP_fump;
  fire internal CBC_fump;
  fire output   stop;
  fire internal CBC_fsii;
  fire output   satb(CBC_f, CBC_d);
  fire internal CBP_fsii;
  fire internal CMM_fsu;
  fire internal CMC_fsu
od
```

5. Test results

During the test, the elevator moves from the first floor to the forth floor by a request from the panel, then a new passenger gets in at the fourth floor and requests to go down to the second floor. On the way down a down request is made from the sixth floor. The elavator transports the passenger to the second floor and hurries to the sixth floor, however, before it ca reach its destination, an up request from the fifth floor arrives. At the end the elevator satifies all requests and the test case ends where the elevator is waiting for further requests.

The test case includes states where the elevator has to handle requests both from the panel and the floor buttons simultaneously, including requests for different directions. The test case is selected by sampling over the valid states for the elevator. Even though there are a large number of valid states, the test case includes a fair amount of the states and requests can also be considered in the opposite direction, doubling the covered states, since the directions are mirrored from the viewpoint of the operation.

The state transitions created by the IOA simulator are included in the appendix and they are quite self explanatory showing how the elevator works.

6. Conclusion

Throughout the report, we tried to demonstrate:

- How to represent components as automata
- How to create components using interfaces
- How components can work asynchronously in a composition
- How to test the preconditions using the IOA simulator

In regard of the simulator results, it is safe to conclude that independently developed components can be composed into larger systems and produce the desired behavior.

Although it is safe to say that components can be substituted as long as they share the same interface, meaning the same input and output actions and the same pre- and post-conditions, a further research can be useful for proofing the automata substitution. Some of the decision units can be implemented with different internal actions, keeping the external signature and pre-conditions of the output actions. Then the traces of the compositions may be compared and used as a proof sketch for component substitution.

Bibliography

IOA Language and Toolset, http://theory.lcs.mit.edu/tds/ioa

Stephen J. Garland, Nancy A. Lynch, Mandana Vaziri, *IOA: A Language for Specifying, Programming and Validating Distributed Systems*, IOA documentation, Oct. 2001

Stephen J. Garland, Nancy A. Lynch, *Using I/O Automata for Developing Distributed Systems*, IOA documentation

[Att99a] P. C. Attie, *Synthesis of large concurrent programs via pairwise composition*, CONCUR'99: 10th International Conference on Concurrency Theory, number 1664 in LNCS. Springer-Verlag, Aug. 1999.

[AE98] P. C. Attie and E. A. Emerson, *Synthesis of concurrent systems with many similar processes*, ACM Trans. Program. Lang. Syst., 20(1):51--115, Jan. 1998.

[CES86] E. M. Clarke, E. A. Emerson, and P. Sistla*, Automatic verification of finite-state concurrent systems using temporal logic specifications*, ACM Trans. Program. Lang. Syst., 8(2):244--263, Apr. 1986, Extended abstract in Proceedings of the 10th Annual ACM Symposium on Principles of Programming Languages.

[AH01] Luca de Alfaro and Thomas A. Henzinger, *Interface automata*, Proceedings of the Ninth Annual Symposium on Foundations of Software Engineering (FSE), pages 109--120. ACM, 2001.

[Dij76] E. W. Dijkstra, *A Discipline of Programming*, Prentice-Hall Inc., Englewood Cliffs, N.J., 1976.

[Em90] E. A. Emerson, *Temporal and modal logic*, J. Van Leeuwen, editor, Handbook of Theoretical Computer Science, volume, Formal Models and Semantics, The MIT Press/Elsevier, Cambridge, Mass., 1990.

[EC82] E. A. Emerson and E. M. Clarke, *Using branching time temporal logic to synthesize synchronization skeletons*, Sci. Comput. Program., 2:241 -- 266, 1982.

[VW01] W. Vanderperren and B. Wydaeghe, *Towards a new component composition process*, Proceedings of ECBS 2001, Apr. 2001.

Appendix: Complete Simulator Results

```
[[[[ Begin initialization [[[[
%%%% State variables:
     CPP_f --> 1
     CPP_d --> s
     CPP_st --> ii
     CPP_r --> null
     CPC_f --> 1
     CPC_d --> s
     CPC_st --> ii
     CPC_r --> null
     CBP_f --> 1
     CBP_d --> s
     CBP_st --> ii
     CBP_r --> null
     CBC_f --> 1
     CBC_d --> s
     CBC_st --> ii
     CBC_r --> null
     CMM_f --> 1
     CMM_d --> s
     CMM_st --> ii
     CMC_f --> 1
     CMC_d --> s
     CMC_st --> ii
     CP_a --> false
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> ()
]]]] End initialization ]]]]
[[[[ Begin step 1 [[[[
    transition: input reqp(4) in automaton controller
%%%% State variables:
     CPP_f --> 1
     CPP_d --> s
     CPP_st --> ii
     CPP_r --> null
     CPC_f --> 1
     CPC_d --> s
     CPC_st --> ii
     CPC_r --> null
     CBP_f --> 1
     CBP_d --> s
     CBP_st --> ii
    CBP_r --> null
    CBC_f --> 1
    CBC_d --> s
    CBC_st --> ii
    CBC_r --> null
    CMM_f --> 1
    CMM_d --> s
    CMM_st --> ii
    CMC_f --> 1
    CMC_d --> s
    CMC_st --> ii
    CP_a --> true
    CB_a --> false
    CP_b --> false
    CB_b --> false
    CP_ps --> (4)
    CB_abs --> ()
    CB_dbs --> ()
]]]] End step 1 ]]]]
[[[[ Begin step 2 [[[[
   transition: internal CPP_fusp in automaton controller
%%%% State variables:
    CPP_f --> 1
    CPP_d --> u
    CPP_st --> s
    CPP_r --> p
    CPC_f --> 1
    CPC_d --> s
    CPC_st --> ii
    CPC_r --> null
    CBP_f --> 1
    CBP_d --> s
    CBP_st --> ii
    CBP_r --> null
    CBC_f --> 1
    CBC_d --> s
    CBC_st --> ii
    CBC_r --> null
    CMM_f --> 1
    CMM_d --> s
    CMM_st --> ii
    CMC_f --> 1
    CMC_d --> s
    CMC_st --> ii
    CP_a --> true
    CB_a --> false
    CP_b --> false
    CB_b --> false
    CP_ps --> (4)
    CB_abs --> ()
    CB_dbs --> ()
]]]] End step 2 ]]]]
[[[[ Begin step 3 [[[[
   transition: internal CMC_fsu in automaton controller
%%%% State variables:
    CPP_f --> 1
    CPP_d --> u
    CPP_st --> s
    CPP_r --> p
    CPC_f --> 1
    CPC_d --> s
    CPC_st --> ii
    CPC_r --> null
    CBP_f --> 1
    CBP_d --> s
    CBP_st --> ii
    CBP_r --> null
    CBC_f --> 1
    CBC_d --> s
    CBC_st --> ii
    CBC_r --> null
    CMM_f --> 1
    CMM_d --> s
    CMM_st --> ii
    CMC_f --> 1
    CMC_d --> u
    CMC_st --> s
    CP_a --> true
    CB_a --> false
    CP_b --> false
    CB_b --> false
    CP_ps --> (4)
    CB_abs --> ()
    CB_dbs --> ()
]]]] End step 3 ]]]]
[[[[ Begin step 4 [[[[
   transition: internal CMM_fsu in automaton controller
%%%% State variables:
    CPP_f --> 1
    CPP_d --> u
    CPP_st --> s
    CPP_r --> p
    CPC_f --> 1
    CPC_d --> s
    CPC_st --> ii
    CPC_r --> null
    CBP_f --> 1
    CBP_d --> s
    CBP_st --> ii
    CBP_r --> null
    CBC_f --> 1
    CBC_d --> s
    CBC_st --> ii
    CBC_r --> null
    CMM_f --> 1
    CMM_d --> u
    CMM_st --> s
    CMC_f --> 1
    CMC_d --> u
    CMC_st --> s
    CP_a --> true
    CB_a --> false
    CP_b --> false
    CB_b --> false
    CP_ps --> (4)
    CB_abs --> ()
    CB_dbs --> ()
]]]] End step 4 ]]]]
[[[[ Begin step 5 [[[[
   transition: internal CPC_fusp in automaton controller
%%%% State variables:
    CPP_f --> 1
    CPP_d --> u
    CPP_st --> s
    CPP_r --> p
    CPC_f --> 1
    CPC_d --> u
    CPC_st --> s
    CPC_r --> p
    CBP_f --> 1
    CBP_d --> s
    CBP_st --> ii
    CBP_r --> null
    CBC_f --> 1
    CBC_d --> s
    CBC_st --> ii
    CBC_r --> null
    CMM_f --> 1
```

```
     CMM_d --> u
     CMM_st --> s
     CMC_f --> 1
     CMC_d --> u
     CMC_st --> s
     CP_a --> true
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> (4)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 5 ]]]]
[[[[ Begin step 6 [[[[
    transition: internal CBC_fusi in automaton controller
%%%% State variables:
     CPP_f --> 1
     CPP_d --> u
     CPP_st --> s
     CPP_r --> p
     CPC_f --> 1
     CPC_d --> u
     CPC_st --> s
     CPC_r --> p
     CBP_f --> 1
     CBP_d --> s
     CBP_st --> ii
     CBP_r --> null
     CBC_f --> 1
     CBC_d --> u
     CBC_st --> s
     CBC_r --> i
     CMM_f --> 1
     CMM_d --> u
     CMM_st --> s
     CMC_f --> 1
     CMC_d --> u
     CMC_st --> s
     CP_a --> true
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> (4)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 6 ]]]]
```

```
[[[[ Begin step 7 [[[[
   transition: internal CBP_fusi in automaton controller
%%%% State variables:
    CPP_f --> 1
    CPP_d --> u
    CPP_st --> s
    CPP_r --> p
    CPC_f --> 1
    CPC_d --> u
    CPC_st --> s
    CPC_r --> p
    CBP_f --> 1
    CBP_d --> u
    CBP_st --> s
    CBP_r --> i
    CBC_f --> 1
    CBC_d --> u
    CBC_st --> s
    CBC_r --> i
    CMM_f --> 1
    CMM_d --> u
    CMM_st --> s
    CMC_f --> 1
    CMC_d --> u
    CMC_st --> s
    CP_a --> true
    CB_a --> false
    CP_b --> false
    CB_b --> false
    CP_ps --> (4)
    CB_abs --> ()
    CB_dbs --> ()
]]]] End step 7 ]]]]
[[[[ Begin step 8 [[[[
   transition: output up in automaton controller
%%%% State variables:
    CPP_f --> 1
    CPP_d --> u
    CPP_st --> s
    CPP_r --> p
    CPC_f --> 1
    CPC_d --> u
    CPC_st --> s
    CPC_r --> p
    CBP_f --> 1
    CBP_d --> u
    CBP_st --> s
    CBP_r --> i
    CBC_f --> 1
    CBC_d --> u
    CBC_st --> s
    CBC_r --> i
    CMM_f --> 1
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 1
    CMC_d --> u
    CMC_st --> s
    CP_a --> true
    CB_a --> false
    CP_b --> false
    CB_b --> false
    CP_ps --> (4)
    CB_abs --> ()
    CB_dbs --> ()
]]]] End step 8 ]]]]
[[[[ Begin step 9 [[[[
   transition: internal CPP_fump in automaton controller
%%%% State variables:
    CPP_f --> 2
    CPP_d --> u
    CPP_st --> m
    CPP_r --> p
    CPC_f --> 1
    CPC_d --> u
    CPC_st --> s
    CPC_r --> p
    CBP_f --> 1
    CBP_d --> u
    CBP_st --> s
    CBP_r --> i
    CBC_f --> 1
    CBC_d --> u
    CBC_st --> s
    CBC_r --> i
    CMM_f --> 1
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 1
    CMC_d --> u
    CMC_st --> s
    CP_a --> true
```

```
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> (4)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 9 ]]]]
[[[[ Begin step 10 [[[[
    transition: internal CPC_fump in automaton controller
%%%% State variables:
     CPP_f --> 2
     CPP_d --> u
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 2
     CPC_d --> u
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 1
     CBP_d --> u
     CBP_st --> s
     CBP_r --> i
     CBC_f --> 1
     CBC_d --> u
     CBC_st --> s
     CBC_r --> i
     CMM_f --> 1
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 1
     CMC_d --> u
     CMC_st --> s
     CP_a --> true
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> (4)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 10 ]]]]
[[[[ Begin step 11 [[[[
    transition: internal CMM_fmu in automaton controller
%%%% State variables:
     CPP_f --> 2
     CPP_d --> u
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 2
     CPC_d --> u
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 1
     CBP_d --> u
     CBP_st --> s
     CBP_r --> i
     CBC_f --> 1
     CBC_d --> u
     CBC_st --> s
     CBC_r --> i
     CMM_f --> 2
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 1
     CMC_d --> u
     CMC_st --> s
     CP_a --> true
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> (4)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 11 ]]]]
[[[[ Begin step 12 [[[[
    transition: internal CMC_fmu in automaton controller
%%%% State variables:
     CPP_f --> 2
     CPP_d --> u
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 2
     CPC_d --> u
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 1
     CBP_d --> u
     CBP_st --> s
     CBP_r --> i
     CBC_f --> 1
     CBC_d --> u
     CBC_st --> s
     CBC_r --> i
    CMM_f --> 2
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 2
    CMC_d --> u
    CMC_st --> m
    CP_a --> true
    CB_a --> false
    CP_b --> false
    CB_b --> false
    CP_ps --> (4)
    CB_abs --> ()
    CB_dbs --> ()
]]]] End step 12 ]]]]
[[[[ Begin step 13 [[[[
   transition: internal CBC_fumi in automaton controller
%%%% State variables:
    CPP_f --> 2
    CPP_d --> u
    CPP_st --> m
    CPP_r --> p
    CPC_f --> 2
    CPC_d --> u
    CPC_st --> m
    CPC_r --> p
    CBP_f --> 1
    CBP_d --> u
    CBP_st --> s
    CBP_r --> i
    CBC_f --> 2
    CBC_d --> u
    CBC_st --> m
    CBC_r --> i
    CMM_f --> 2
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 2
    CMC_d --> u
    CMC_st --> m
    CP_a --> true
    CB_a --> false
    CP_b --> false
    CB_b --> false
    CP_ps --> (4)
    CB_abs --> ()
    CB_dbs --> ()
```

```
]]]] End step 13 ]]]]
[[[[ Begin step 14 [[[[
   transition: internal CBP_fumi in automaton controller
%%%% State variables:
    CPP_f --> 2
    CPP_d --> u
    CPP_st --> m
    CPP_r --> p
    CPC_f --> 2
    CPC_d --> u
    CPC_st --> m
    CPC_r --> p
    CBP_f --> 2
    CBP_d --> u
    CBP_st --> m
    CBP_r --> i
    CBC_f --> 2
    CBC_d --> u
    CBC_st --> m
    CBC_r --> i
    CMM_f --> 2
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 2
    CMC_d --> u
    CMC_st --> m
    CP_a --> true
    CB_a --> false
    CP_b --> false
    CB_b --> false
    CP_ps --> (4)
    CB_abs --> ()
    CB_dbs --> ()
]]]] End step 14 ]]]]
[[[[ Begin step 15 [[[[
   transition: internal CPP_fump in automaton controller
%%%% State variables:
    CPP_f --> 3
    CPP_d --> u
    CPP_st --> m
    CPP_r --> p
    CPC_f --> 2
    CPC_d --> u
    CPC_st --> m
    CPC_r --> p
    CBP_f --> 2
```

```
     CBP_d --> u
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 2
     CBC_d --> u
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 2
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 2
     CMC_d --> u
     CMC_st --> m
     CP_a --> true
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> (4)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 15 ]]]]
[[[[ Begin step 16 [[[[
    transition: internal CBC_fumi in automaton controller
%%%% State variables:
     CPP_f --> 3
     CPP_d --> u
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 2
     CPC_d --> u
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 2
     CBP_d --> u
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 3
     CBC_d --> u
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 2
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 2
     CMC_d --> u
     CMC_st --> m
     CP_a --> true
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> (4)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 16 ]]]]
[[[[ Begin step 17 [[[[
    transition: internal CBP_fumi in automaton controller
%%%% State variables:
     CPP_f --> 3
     CPP_d --> u
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 2
     CPC_d --> u
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 3
     CBP_d --> u
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 3
     CBC_d --> u
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 2
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 2
     CMC_d --> u
     CMC_st --> m
     CP_a --> true
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> (4)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 17 ]]]]
[[[[ Begin step 18 [[[[
    transition: internal CPC_fump in automaton controller
%%%% State variables:
     CPP_f --> 3
     CPP_d --> u
    CPP_st --> m
    CPP_r --> p
    CPC_f --> 3
    CPC_d --> u
    CPC_st --> m
    CPC_r --> p
    CBP_f --> 3
    CBP_d --> u
    CBP_st --> m
    CBP_r --> i
    CBC_f --> 3
    CBC_d --> u
    CBC_st --> m
    CBC_r --> i
    CMM_f --> 2
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 2
    CMC_d --> u
    CMC_st --> m
    CP_a --> true
    CB_a --> false
    CP_b --> false
    CB_b --> false
    CP_ps --> (4)
    CB_abs --> ()
    CB_dbs --> ()
]]]] End step 18 ]]]]
[[[[ Begin step 19 [[[[
   transition: internal CMC_fmu in automaton controller
%%%% State variables:
    CPP_f --> 3
    CPP_d --> u
    CPP_st --> m
    CPP_r --> p
    CPC_f --> 3
    CPC_d --> u
    CPC_st --> m
    CPC_r --> p
    CBP_f --> 3
    CBP_d --> u
    CBP_st --> m
    CBP_r --> i
    CBC_f --> 3
    CBC_d --> u
    CBC_st --> m
```

```
     CBC_r --> i
     CMM_f --> 2
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 3
     CMC_d --> u
     CMC_st --> m
     CP_a --> true
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> (4)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 19 ]]]]
[[[[ Begin step 20 [[[[
    transition: internal CMM_fmu in automaton controller
%%%% State variables:
     CPP_f --> 3
     CPP_d --> u
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 3
     CPC_d --> u
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 3
     CBP_d --> u
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 3
     CBC_d --> u
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 3
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 3
     CMC_d --> u
     CMC_st --> m
     CP_a --> true
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> (4)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 20 ]]]]
[[[[ Begin step 21 [[[[
    transition: internal CPP_fump in automaton controller
%%%% State variables:
     CPP_f --> 4
     CPP_d --> u
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 3
     CPC_d --> u
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 3
     CBP_d --> u
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 3
     CBC_d --> u
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 3
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 3
     CMC_d --> u
     CMC_st --> m
     CP_a --> true
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> (4)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 21 ]]]]
[[[[ Begin step 22 [[[[
    transition: internal CBC_fumi in automaton controller
%%%% State variables:
     CPP_f --> 4
     CPP_d --> u
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 3
     CPC_d --> u
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 3
     CBP_d --> u
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 4
     CBC_d --> u
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 3
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 3
     CMC_d --> u
     CMC_st --> m
     CP_a --> true
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> (4)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 22 ]]]]
[[[[ Begin step 23 [[[[
    transition: internal CBP_fumi in automaton controller
%%%% State variables:
     CPP_f --> 4
     CPP_d --> u
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 3
     CPC_d --> u
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> u
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 4
     CBC_d --> u
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 3
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 3
     CMC_d --> u
     CMC_st --> m
     CP_a --> true
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> (4)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 23 ]]]]
[[[[ Begin step 24 [[[[
    transition: internal CPC_fump in automaton controller
%%%% State variables:
     CPP_f --> 4
     CPP_d --> u
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 4
     CPC_d --> u
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> u
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 4
     CBC_d --> u
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 3
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 3
     CMC_d --> u
     CMC_st --> m
     CP_a --> true
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> (4)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 24 ]]]]
[[[[ Begin step 25 [[[[
    transition: internal CMC_fmu in automaton controller
%%%% State variables:
     CPP_f --> 4
     CPP_d --> u
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 4
     CPC_d --> u
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> u
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 4
     CBC_d --> u
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 3
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 4
     CMC_d --> u
     CMC_st --> m
     CP_a --> true
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> (4)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 25 ]]]]
[[[[ Begin step 26 [[[[
    transition: internal CMM_fmu in automaton controller
%%%% State variables:
     CPP_f --> 4
     CPP_d --> u
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 4
     CPC_d --> u
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> u
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 4
     CBC_d --> u
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 4
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 4
     CMC_d --> u
     CMC_st --> m
     CP_a --> true
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> (4)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 26 ]]]]
[[[[ Begin step 27 [[[[
    transition: output satp(4) in automaton controller
%%%% State variables:
     CPP_f --> 4
     CPP_d --> u
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 4
     CPC_d --> u
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> u
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 4
     CBC_d --> u
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 4
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 4
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 27 ]]]]
[[[[ Begin step 28 [[[[
    transition: internal CPP_fsii in automaton controller
%%%% State variables:
     CPP_f --> 4
     CPP_d --> s
     CPP_st --> ii
     CPP_r --> null
     CPC_f --> 4
     CPC_d --> u
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> u
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 4
     CBC_d --> u
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 4
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 4
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 28 ]]]]
[[[[ Begin step 29 [[[[
    transition: internal CBC_fsii in automaton controller
%%%% State variables:
     CPP_f --> 4
     CPP_d --> s
     CPP_st --> ii
     CPP_r --> null
     CPC_f --> 4
     CPC_d --> u
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> u
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 4
     CBC_d --> s
     CBC_st --> ii
     CBC_r --> null
     CMM_f --> 4
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 4
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 29 ]]]]
[[[[ Begin step 30 [[[[
    transition: internal CBP_fsii in automaton controller
%%%% State variables:
     CPP_f --> 4
     CPP_d --> s
     CPP_st --> ii
     CPP_r --> null
     CPC_f --> 4
     CPC_d --> u
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> s
     CBP_st --> ii
     CBP_r --> null
     CBC_f --> 4
     CBC_d --> s
     CBC_st --> ii
     CBC_r --> null
     CMM_f --> 4
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 4
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 30 ]]]]
[[[[ Begin step 31 [[[[
    transition: output stop in automaton controller
%%%% State variables:
     CPP_f --> 4
     CPP_d --> s
     CPP_st --> ii
     CPP_r --> null
     CPC_f --> 4
     CPC_d --> u
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> s
     CBP_st --> ii
     CBP_r --> null
     CBC_f --> 4
     CBC_d --> s
     CBC_st --> ii
     CBC_r --> null
     CMM_f --> 4
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 4
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> false
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 31 ]]]]
[[[[ Begin step 32 [[[[
    transition: input reqp(2) in automaton controller
%%%% State variables:
    CPP_f --> 4
    CPP_d --> s
    CPP_st --> ii
    CPP_r --> null
    CPC_f --> 4
    CPC_d --> u
    CPC_st --> m
    CPC_r --> p
    CBP_f --> 4
    CBP_d --> s
    CBP_st --> ii
    CBP_r --> null
    CBC_f --> 4
    CBC_d --> s
    CBC_st --> ii
    CBC_r --> null
    CMM_f --> 4
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 4
    CMC_d --> u
    CMC_st --> m
    CP_a --> false
    CB_a --> false
    CP_b --> true
    CB_b --> false
    CP_ps --> (2)
    CB_abs --> ()
    CB_dbs --> ()
]]]] End step 32 ]]]]
[[[[ Begin step 33 [[[[
   transition: internal CPP_fdsp in automaton controller
%%%% State variables:
    CPP_f --> 4
    CPP_d --> d
    CPP_st --> s
    CPP_r --> p
    CPC_f --> 4
    CPC_d --> u
    CPC_st --> m
    CPC_r --> p
    CBP_f --> 4
    CBP_d --> s
    CBP_st --> ii
    CBP_r --> null
    CBC_f --> 4
    CBC_d --> s
    CBC_st --> ii
    CBC_r --> null
    CMM_f --> 4
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 4
    CMC_d --> u
    CMC_st --> m
    CP_a --> false
    CB_a --> false
    CP_b --> true
    CB_b --> false
    CP_ps --> (2)
    CB_abs --> ()
    CB_dbs --> ()
]]]] End step 33 ]]]]
[[[[ Begin step 34 [[[[
   transition: internal CMC_fsu in automaton controller
%%%% State variables:
    CPP_f --> 4
    CPP_d --> d
    CPP_st --> s
    CPP_r --> p
    CPC_f --> 4
    CPC_d --> u
    CPC_st --> m
    CPC_r --> p
    CBP_f --> 4
    CBP_d --> s
    CBP_st --> ii
    CBP_r --> null
    CBC_f --> 4
    CBC_d --> s
    CBC_st --> ii
    CBC_r --> null
    CMM_f --> 4
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 4
    CMC_d --> u
    CMC_st --> s
    CP_a --> false
    CB_a --> false
    CP_b --> true
    CB_b --> false
    CP_ps --> (2)
    CB_abs --> ()
    CB_dbs --> ()
]]]] End step 34 ]]]]
[[[[ Begin step 35 [[[[
   transition: internal CMM_fsu in automaton controller
%%%% State variables:
    CPP_f --> 4
    CPP_d --> d
    CPP_st --> s
    CPP_r --> p
    CPC_f --> 4
    CPC_d --> u
    CPC_st --> m
    CPC_r --> p
    CBP_f --> 4
    CBP_d --> s
    CBP_st --> ii
    CBP_r --> null
    CBC_f --> 4
    CBC_d --> s
    CBC_st --> ii
    CBC_r --> null
    CMM_f --> 4
    CMM_d --> u
    CMM_st --> s
    CMC_f --> 4
    CMC_d --> u
    CMC_st --> s
    CP_a --> false
    CB_a --> false
    CP_b --> true
    CB_b --> false
    CP_ps --> (2)
    CB_abs --> ()
    CB_dbs --> ()
]]]] End step 35 ]]]]
[[[[ Begin step 36 [[[[
   transition: internal CMC_fsd in automaton controller
%%%% State variables:
    CPP_f --> 4
    CPP_d --> d
    CPP_st --> s
    CPP_r --> p
    CPC_f --> 4
    CPC_d --> u
```

```
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> s
     CBP_st --> ii
     CBP_r --> null
     CBC_f --> 4
     CBC_d --> s
     CBC_st --> ii
     CBC_r --> null
     CMM_f --> 4
     CMM_d --> u
     CMM_st --> s
     CMC_f --> 4
     CMC_d --> d
     CMC_st --> s
     CP_a --> false
     CB_a --> false
     CP_b --> true
     CB_b --> false
     CP_ps --> (2)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 36 ]]]]
[[[[ Begin step 37 [[[[
    transition: internal CMM_fsd in automaton controller
%%%% State variables:
     CPP_f --> 4
     CPP_d --> d
     CPP_st --> s
     CPP_r --> p
     CPC_f --> 4
     CPC_d --> u
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> s
     CBP_st --> ii
     CBP_r --> null
     CBC_f --> 4
     CBC_d --> s
     CBC_st --> ii
     CBC_r --> null
     CMM_f --> 4
     CMM_d --> d
     CMM_st --> s
     CMC_f --> 4
     CMC_d --> d
     CMC_st --> s
     CP_a --> false
     CB_a --> false
     CP_b --> true
     CB_b --> false
     CP_ps --> (2)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 37 ]]]]
[[[[ Begin step 38 [[[[
    transition: internal CPC_fdsp in automaton controller
%%%% State variables:
     CPP_f --> 4
     CPP_d --> d
     CPP_st --> s
     CPP_r --> p
     CPC_f --> 4
     CPC_d --> d
     CPC_st --> s
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> s
     CBP_st --> ii
     CBP_r --> null
     CBC_f --> 4
     CBC_d --> s
     CBC_st --> ii
     CBC_r --> null
     CMM_f --> 4
     CMM_d --> d
     CMM_st --> s
     CMC_f --> 4
     CMC_d --> d
     CMC_st --> s
     CP_a --> false
     CB_a --> false
     CP_b --> true
     CB_b --> false
     CP_ps --> (2)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 38 ]]]]
[[[[ Begin step 39 [[[[
    transition: internal CBC_fdsi in automaton controller
```

```
%%%% State variables:
    CPP_f --> 4
    CPP_d --> d
    CPP_st --> s
    CPP_r --> p
    CPC_f --> 4
    CPC_d --> d
    CPC_st --> s
    CPC_r --> p
    CBP_f --> 4
    CBP_d --> s
    CBP_st --> ii
    CBP_r --> null
    CBC_f --> 4
    CBC_d --> d
    CBC_st --> s
    CBC_r --> i
    CMM_f --> 4
    CMM_d --> d
    CMM_st --> s
    CMC_f --> 4
    CMC_d --> d
    CMC_st --> s
    CP_a --> false
    CB_a --> false
    CP_b --> true
    CB_b --> false
    CP_ps --> (2)
    CB_abs --> ()
    CB_dbs --> ()
]]]] End step 39 ]]]]
[[[[ Begin step 40 [[[[
   transition: internal CBP_fdsi in automaton controller
%%%% State variables:
    CPP_f --> 4
    CPP_d --> d
    CPP_st --> s
    CPP_r --> p
    CPC_f --> 4
    CPC_d --> d
    CPC_st --> s
    CPC_r --> p
    CBP_f --> 4
    CBP_d --> d
    CBP_st --> s
    CBP_r --> i
```

```
     CBC_f --> 4
     CBC_d --> d
     CBC_st --> s
     CBC_r --> i
     CMM_f --> 4
     CMM_d --> d
     CMM_st --> s
     CMC_f --> 4
     CMC_d --> d
     CMC_st --> s
     CP_a --> false
     CB_a --> false
     CP_b --> true
     CB_b --> false
     CP_ps --> (2)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 40 ]]]]
[[[[ Begin step 41 [[[[
    transition: output down in automaton controller
%%%% State variables:
     CPP_f --> 4
     CPP_d --> d
     CPP_st --> s
     CPP_r --> p
     CPC_f --> 4
     CPC_d --> d
     CPC_st --> s
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> d
     CBP_st --> s
     CBP_r --> i
     CBC_f --> 4
     CBC_d --> d
     CBC_st --> s
     CBC_r --> i
     CMM_f --> 4
     CMM_d --> d
     CMM_st --> m
     CMC_f --> 4
     CMC_d --> d
     CMC_st --> s
     CP_a --> false
     CB_a --> false
     CP_b --> true
     CB_b --> false
     CP_ps --> (2)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 41 ]]]]
[[[[ Begin step 42 [[[[
    transition: internal CPP_fdmp in automaton controller
%%%% State variables:
     CPP_f --> 3
     CPP_d --> d
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 4
     CPC_d --> d
     CPC_st --> s
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> d
     CBP_st --> s
     CBP_r --> i
     CBC_f --> 4
     CBC_d --> d
     CBC_st --> s
     CBC_r --> i
     CMM_f --> 4
     CMM_d --> d
     CMM_st --> m
     CMC_f --> 4
     CMC_d --> d
     CMC_st --> s
     CP_a --> false
     CB_a --> false
     CP_b --> true
     CB_b --> false
     CP_ps --> (2)
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 42 ]]]]
[[[[ Begin step 43 [[[[
    transition: input reqb(6, d) in automaton controller
%%%% State variables:
     CPP_f --> 3
     CPP_d --> d
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 4
     CPC_d --> d
     CPC_st --> s
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> d
     CBP_st --> s
     CBP_r --> i
     CBC_f --> 4
     CBC_d --> d
     CBC_st --> s
     CBC_r --> i
     CMM_f --> 4
     CMM_d --> d
     CMM_st --> m
     CMC_f --> 4
     CMC_d --> d
     CMC_st --> s
     CP_a --> false
     CB_a --> true
     CP_b --> true
     CB_b --> false
     CP_ps --> (2)
     CB_abs --> ()
     CB_dbs --> (6)
]]]] End step 43 ]]]]
[[[[ Begin step 44 [[[[
    transition: internal CPC_fdmp in automaton controller
%%%% State variables:
     CPP_f --> 3
     CPP_d --> d
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 3
     CPC_d --> d
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> d
     CBP_st --> s
     CBP_r --> i
     CBC_f --> 4
     CBC_d --> d
     CBC_st --> s
     CBC_r --> i
     CMM_f --> 4
     CMM_d --> d
     CMM_st --> m
     CMC_f --> 4
     CMC_d --> d
     CMC_st --> s
     CP_a --> false
     CB_a --> true
     CP_b --> true
     CB_b --> false
     CP_ps --> (2)
     CB_abs --> ()
     CB_dbs --> (6)
]]]] End step 44 ]]]]
[[[[ Begin step 45 [[[[
    transition: internal CMC_fmd in automaton controller
%%%% State variables:
     CPP_f --> 3
     CPP_d --> d
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 3
     CPC_d --> d
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> d
     CBP_st --> s
     CBP_r --> i
     CBC_f --> 4
     CBC_d --> d
     CBC_st --> s
     CBC_r --> i
     CMM_f --> 4
     CMM_d --> d
     CMM_st --> m
     CMC_f --> 3
     CMC_d --> d
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> true
     CB_b --> false
     CP_ps --> (2)
     CB_abs --> ()
     CB_dbs --> (6)
]]]] End step 45 ]]]]
[[[[ Begin step 46 [[[[
    transition: internal CMM_fmd in automaton controller
%%%% State variables:
     CPP_f --> 3
     CPP_d --> d
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 3
     CPC_d --> d
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> d
     CBP_st --> s
     CBP_r --> i
     CBC_f --> 4
     CBC_d --> d
     CBC_st --> s
     CBC_r --> i
     CMM_f --> 3
     CMM_d --> d
     CMM_st --> m
     CMC_f --> 3
     CMC_d --> d
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> true
     CB_b --> false
     CP_ps --> (2)
     CB_abs --> ()
     CB_dbs --> (6)
]]]] End step 46 ]]]]
[[[[ Begin step 47 [[[[
    transition: internal CBC_fdmi in automaton controller
%%%% State variables:
     CPP_f --> 3
     CPP_d --> d
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 3
     CPC_d --> d
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 4
     CBP_d --> d
     CBP_st --> s
    CBP_r --> i
    CBC_f --> 3
    CBC_d --> d
    CBC_st --> m
    CBC_r --> i
    CMM_f --> 3
    CMM_d --> d
    CMM_st --> m
    CMC_f --> 3
    CMC_d --> d
    CMC_st --> m
    CP_a --> false
    CB_a --> true
    CP_b --> true
    CB_b --> false
    CP_ps --> (2)
    CB_abs --> ()
    CB_dbs --> (6)
]]]] End step 47 ]]]]
[[[[ Begin step 48 [[[[
   transition: internal CBP_fdmi in automaton controller
%%%% State variables:
    CPP_f --> 3
    CPP_d --> d
    CPP_st --> m
    CPP_r --> p
    CPC_f --> 3
    CPC_d --> d
    CPC_st --> m
    CPC_r --> p
    CBP_f --> 3
    CBP_d --> d
    CBP_st --> m
    CBP_r --> i
    CBC_f --> 3
    CBC_d --> d
    CBC_st --> m
    CBC_r --> i
    CMM_f --> 3
    CMM_d --> d
    CMM_st --> m
    CMC_f --> 3
    CMC_d --> d
    CMC_st --> m
    CP_a --> false
    CB_a --> true
```

```
     CP_b --> true
     CB_b --> false
     CP_ps --> (2)
     CB_abs --> ()
     CB_dbs --> (6)
]]]] End step 48 ]]]]
[[[[ Begin step 49 [[[[
    transition: internal CPP_fdmp in automaton controller
%%%% State variables:
     CPP_f --> 2
     CPP_d --> d
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 3
     CPC_d --> d
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 3
     CBP_d --> d
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 3
     CBC_d --> d
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 3
     CMM_d --> d
     CMM_st --> m
     CMC_f --> 3
     CMC_d --> d
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> true
     CB_b --> false
     CP_ps --> (2)
     CB_abs --> ()
     CB_dbs --> (6)
]]]] End step 49 ]]]]
[[[[ Begin step 50 [[[[
    transition: input reqb(5, u) in automaton controller
%%%% State variables:
     CPP_f --> 2
     CPP_d --> d
     CPP_st --> m
     CPP_r --> p
    CPC_f --> 3
    CPC_d --> d
    CPC_st --> m
    CPC_r --> p
    CBP_f --> 3
    CBP_d --> d
    CBP_st --> m
    CBP_r --> i
    CBC_f --> 3
    CBC_d --> d
    CBC_st --> m
    CBC_r --> i
    CMM_f --> 3
    CMM_d --> d
    CMM_st --> m
    CMC_f --> 3
    CMC_d --> d
    CMC_st --> m
    CP_a --> false
    CB_a --> true
    CP_b --> true
    CB_b --> false
    CP_ps --> (2)
    CB_abs --> (5)
    CB_dbs --> (6)
]]]] End step 50 ]]]]
[[[[ Begin step 51 [[[[
   transition: internal CPC_fdmp in automaton controller
%%%% State variables:
    CPP_f --> 2
    CPP_d --> d
    CPP_st --> m
    CPP_r --> p
    CPC_f --> 2
    CPC_d --> d
    CPC_st --> m
    CPC_r --> p
    CBP_f --> 3
    CBP_d --> d
    CBP_st --> m
    CBP_r --> i
    CBC_f --> 3
    CBC_d --> d
    CBC_st --> m
    CBC_r --> i
    CMM_f --> 3
```

```
     CMM_d --> d
     CMM_st --> m
     CMC_f --> 3
     CMC_d --> d
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> true
     CB_b --> false
     CP_ps --> (2)
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 51 ]]]]
[[[[ Begin step 52 [[[[
    transition: internal CBC_fdmi in automaton controller
%%%% State variables:
     CPP_f --> 2
     CPP_d --> d
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 2
     CPC_d --> d
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 3
     CBP_d --> d
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 2
     CBC_d --> d
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 3
     CMM_d --> d
     CMM_st --> m
     CMC_f --> 3
     CMC_d --> d
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> true
     CB_b --> false
     CP_ps --> (2)
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 52 ]]]]
```

```
[[[[ Begin step 53 [[[[
   transition: internal CBP_fdmi in automaton controller
%%%% State variables:
    CPP_f --> 2
    CPP_d --> d
    CPP_st --> m
    CPP_r --> p
    CPC_f --> 2
    CPC_d --> d
    CPC_st --> m
    CPC_r --> p
    CBP_f --> 2
    CBP_d --> d
    CBP_st --> m
    CBP_r --> i
    CBC_f --> 2
    CBC_d --> d
    CBC_st --> m
    CBC_r --> i
    CMM_f --> 3
    CMM_d --> d
    CMM_st --> m
    CMC_f --> 3
    CMC_d --> d
    CMC_st --> m
    CP_a --> false
    CB_a --> true
    CP_b --> true
    CB_b --> false
    CP_ps --> (2)
    CB_abs --> (5)
    CB_dbs --> (6)
]]]] End step 53 ]]]]
[[[[ Begin step 54 [[[[
   transition: internal CMC_fmd in automaton controller
%%%% State variables:
    CPP_f --> 2
    CPP_d --> d
    CPP_st --> m
    CPP_r --> p
    CPC_f --> 2
    CPC_d --> d
    CPC_st --> m
    CPC_r --> p
    CBP_f --> 2
    CBP_d --> d
```

```
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 2
     CBC_d --> d
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 3
     CMM_d --> d
     CMM_st --> m
     CMC_f --> 2
     CMC_d --> d
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> true
     CB_b --> false
     CP_ps --> (2)
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 54 ]]]]
[[[[ Begin step 55 [[[[
    transition: internal CMM_fmd in automaton controller
%%%% State variables:
     CPP_f --> 2
     CPP_d --> d
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 2
     CPC_d --> d
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 2
     CBP_d --> d
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 2
     CBC_d --> d
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 2
     CMM_d --> d
     CMM_st --> m
     CMC_f --> 2
     CMC_d --> d
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> true
     CB_b --> false
     CP_ps --> (2)
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 55 ]]]]
[[[[ Begin step 56 [[[[
    transition: output satp(2) in automaton controller
%%%% State variables:
     CPP_f --> 2
     CPP_d --> d
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 2
     CPC_d --> d
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 2
     CBP_d --> d
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 2
     CBC_d --> d
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 2
     CMM_d --> d
     CMM_st --> m
     CMC_f --> 2
     CMC_d --> d
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 56 ]]]]
[[[[ Begin step 57 [[[[
    transition: output stop in automaton controller
%%%% State variables:
     CPP_f --> 2
     CPP_d --> d
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 2
     CPC_d --> d
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 2
     CBP_d --> d
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 2
     CBC_d --> d
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 2
     CMM_d --> d
     CMM_st --> m
     CMC_f --> 2
     CMC_d --> d
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 57 ]]]]
[[[[ Begin step 58 [[[[
    transition: internal CMM_fsd in automaton controller
%%%% State variables:
     CPP_f --> 2
     CPP_d --> d
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 2
     CPC_d --> d
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 2
     CBP_d --> d
     CBP_st --> m
     CBP_r --> i
     CBC_f --> 2
     CBC_d --> d
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 2
     CMM_d --> d
     CMM_st --> s
     CMC_f --> 2
     CMC_d --> d
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 58 ]]]]
[[[[ Begin step 59 [[[[
    transition: internal CBP_fusp in automaton controller
%%%% State variables:
     CPP_f --> 2
     CPP_d --> d
     CPP_st --> m
     CPP_r --> p
     CPC_f --> 2
     CPC_d --> d
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 2
     CBP_d --> u
     CBP_st --> s
     CBP_r --> p
     CBC_f --> 2
     CBC_d --> d
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 2
     CMM_d --> d
     CMM_st --> s
     CMC_f --> 2
     CMC_d --> d
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
```

```
]]]] End step 59 ]]]]
[[[[ Begin step 60 [[[[
   transition: internal CPP_fsii in automaton controller
%%%% State variables:
    CPP_f --> 2
    CPP_d --> s
    CPP_st --> ii
    CPP_r --> null
    CPC_f --> 2
    CPC_d --> d
    CPC_st --> m
    CPC_r --> p
    CBP_f --> 2
    CBP_d --> u
    CBP_st --> s
    CBP_r --> p
    CBC_f --> 2
    CBC_d --> d
    CBC_st --> m
    CBC_r --> i
    CMM_f --> 2
    CMM_d --> d
    CMM_st --> s
    CMC_f --> 2
    CMC_d --> d
    CMC_st --> m
    CP_a --> false
    CB_a --> true
    CP_b --> false
    CB_b --> false
    CP_ps --> ()
    CB_abs --> (5)
    CB_dbs --> (6)
]]]] End step 60 ]]]]
[[[[ Begin step 61 [[[[
   transition: internal CMC_fsd in automaton controller
%%%% State variables:
    CPP_f --> 2
    CPP_d --> s
    CPP_st --> ii
    CPP_r --> null
    CPC_f --> 2
    CPC_d --> d
    CPC_st --> m
    CPC_r --> p
    CBP_f --> 2
```

```
     CBP_d --> u
     CBP_st --> s
     CBP_r --> p
     CBC_f --> 2
     CBC_d --> d
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 2
     CMM_d --> d
     CMM_st --> s
     CMC_f --> 2
     CMC_d --> d
     CMC_st --> s
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 61 ]]]]
[[[[ Begin step 62 [[[[
    transition: internal CMM_fsu in automaton controller
%%%% State variables:
     CPP_f --> 2
     CPP_d --> s
     CPP_st --> ii
     CPP_r --> null
     CPC_f --> 2
     CPC_d --> d
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 2
     CBP_d --> u
     CBP_st --> s
     CBP_r --> p
     CBC_f --> 2
     CBC_d --> d
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 2
     CMM_d --> u
     CMM_st --> s
     CMC_f --> 2
     CMC_d --> d
     CMC_st --> s
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 62 ]]]]
[[[[ Begin step 63 [[[[
    transition: internal CMC_fsu in automaton controller
%%%% State variables:
     CPP_f --> 2
     CPP_d --> s
     CPP_st --> ii
     CPP_r --> null
     CPC_f --> 2
     CPC_d --> d
     CPC_st --> m
     CPC_r --> p
     CBP_f --> 2
     CBP_d --> u
     CBP_st --> s
     CBP_r --> p
     CBC_f --> 2
     CBC_d --> d
     CBC_st --> m
     CBC_r --> i
     CMM_f --> 2
     CMM_d --> u
     CMM_st --> s
     CMC_f --> 2
     CMC_d --> u
     CMC_st --> s
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 63 ]]]]
[[[[ Begin step 64 [[[[
    transition: internal CBC_fusp in automaton controller
%%%% State variables:
     CPP_f --> 2
     CPP_d --> s
    CPP_st --> ii
    CPP_r --> null
    CPC_f --> 2
    CPC_d --> d
    CPC_st --> m
    CPC_r --> p
    CBP_f --> 2
    CBP_d --> u
    CBP_st --> s
    CBP_r --> p
    CBC_f --> 2
    CBC_d --> u
    CBC_st --> s
    CBC_r --> p
    CMM_f --> 2
    CMM_d --> u
    CMM_st --> s
    CMC_f --> 2
    CMC_d --> u
    CMC_st --> s
    CP_a --> false
    CB_a --> true
    CP_b --> false
    CB_b --> false
    CP_ps --> ()
    CB_abs --> (5)
    CB_dbs --> (6)
]]]] End step 64 ]]]]
[[[[ Begin step 65 [[[[
   transition: internal CPC_fsii in automaton controller
%%%% State variables:
    CPP_f --> 2
    CPP_d --> s
    CPP_st --> ii
    CPP_r --> null
    CPC_f --> 2
    CPC_d --> s
    CPC_st --> ii
    CPC_r --> null
    CBP_f --> 2
    CBP_d --> u
    CBP_st --> s
    CBP_r --> p
    CBC_f --> 2
    CBC_d --> u
    CBC_st --> s
```

```
     CBC_r --> p
     CMM_f --> 2
     CMM_d --> u
     CMM_st --> s
     CMC_f --> 2
     CMC_d --> u
     CMC_st --> s
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 65 ]]]]
[[[[ Begin step 66 [[[[
    transition: output up in automaton controller
%%%% State variables:
     CPP_f --> 2
     CPP_d --> s
     CPP_st --> ii
     CPP_r --> null
     CPC_f --> 2
     CPC_d --> s
     CPC_st --> ii
     CPC_r --> null
     CBP_f --> 2
     CBP_d --> u
     CBP_st --> s
     CBP_r --> p
     CBC_f --> 2
     CBC_d --> u
     CBC_st --> s
     CBC_r --> p
     CMM_f --> 2
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 2
     CMC_d --> u
     CMC_st --> s
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
    CB_dbs --> (6)
]]]] End step 66 ]]]]
[[[[ Begin step 67 [[[[
   transition: internal CMC_fmu in automaton controller
%%%% State variables:
    CPP_f --> 2
    CPP_d --> s
    CPP_st --> ii
    CPP_r --> null
    CPC_f --> 2
    CPC_d --> s
    CPC_st --> ii
    CPC_r --> null
    CBP_f --> 2
    CBP_d --> u
    CBP_st --> s
    CBP_r --> p
    CBC_f --> 2
    CBC_d --> u
    CBC_st --> s
    CBC_r --> p
    CMM_f --> 2
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 3
    CMC_d --> u
    CMC_st --> m
    CP_a --> false
    CB_a --> true
    CP_b --> false
    CB_b --> false
    CP_ps --> ()
    CB_abs --> (5)
    CB_dbs --> (6)
]]]] End step 67 ]]]]
[[[[ Begin step 68 [[[[
   transition: internal CMM_fmu in automaton controller
%%%% State variables:
    CPP_f --> 2
    CPP_d --> s
    CPP_st --> ii
    CPP_r --> null
    CPC_f --> 2
    CPC_d --> s
    CPC_st --> ii
    CPC_r --> null
```

```
     CBP_f --> 2
     CBP_d --> u
     CBP_st --> s
     CBP_r --> p
     CBC_f --> 2
     CBC_d --> u
     CBC_st --> s
     CBC_r --> p
     CMM_f --> 3
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 3
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 68 ]]]]
[[[[ Begin step 69 [[[[
    transition: internal CBP_fump in automaton controller
%%%% State variables:
     CPP_f --> 2
     CPP_d --> s
     CPP_st --> ii
     CPP_r --> null
     CPC_f --> 2
     CPC_d --> s
     CPC_st --> ii
     CPC_r --> null
     CBP_f --> 3
     CBP_d --> u
     CBP_st --> m
     CBP_r --> p
     CBC_f --> 2
     CBC_d --> u
     CBC_st --> s
     CBC_r --> p
     CMM_f --> 3
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 3
     CMC_d --> u
    CMC_st --> m
    CP_a --> false
    CB_a --> true
    CP_b --> false
    CB_b --> false
    CP_ps --> ()
    CB_abs --> (5)
    CB_dbs --> (6)
]]]] End step 69 ]]]]
[[[[ Begin step 70 [[[[
   transition: internal CBC_fump in automaton controller
%%%% State variables:
    CPP_f --> 2
    CPP_d --> s
    CPP_st --> ii
    CPP_r --> null
    CPC_f --> 2
    CPC_d --> s
    CPC_st --> ii
    CPC_r --> null
    CBP_f --> 3
    CBP_d --> u
    CBP_st --> m
    CBP_r --> p
    CBC_f --> 3
    CBC_d --> u
    CBC_st --> m
    CBC_r --> p
    CMM_f --> 3
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 3
    CMC_d --> u
    CMC_st --> m
    CP_a --> false
    CB_a --> true
    CP_b --> false
    CB_b --> false
    CP_ps --> ()
    CB_abs --> (5)
    CB_dbs --> (6)
]]]] End step 70 ]]]]
[[[[ Begin step 71 [[[[
   transition: internal CPP_fusi in automaton controller
%%%% State variables:
    CPP_f --> 2
    CPP_d --> u
    CPP_st --> s
    CPP_r --> i
    CPC_f --> 2
    CPC_d --> s
    CPC_st --> ii
    CPC_r --> null
    CBP_f --> 3
    CBP_d --> u
    CBP_st --> m
    CBP_r --> p
    CBC_f --> 3
    CBC_d --> u
    CBC_st --> m
    CBC_r --> p
    CMM_f --> 3
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 3
    CMC_d --> u
    CMC_st --> m
    CP_a --> false
    CB_a --> true
    CP_b --> false
    CB_b --> false
    CP_ps --> ()
    CB_abs --> (5)
    CB_dbs --> (6)
]]]] End step 71 ]]]]
[[[[ Begin step 72 [[[[
   transition: internal CPC_fusi in automaton controller
%%%% State variables:
    CPP_f --> 2
    CPP_d --> u
    CPP_st --> s
    CPP_r --> i
    CPC_f --> 2
    CPC_d --> u
    CPC_st --> s
    CPC_r --> i
    CBP_f --> 3
    CBP_d --> u
    CBP_st --> m
    CBP_r --> p
    CBC_f --> 3
    CBC_d --> u
```

```
     CBC_st --> m
     CBC_r --> p
     CMM_f --> 3
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 3
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 72 ]]]]
[[[[ Begin step 73 [[[[
    transition: internal CPP_fumi in automaton controller
%%%% State variables:
     CPP_f --> 3
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
     CPC_f --> 2
     CPC_d --> u
     CPC_st --> s
     CPC_r --> i
     CBP_f --> 3
     CBP_d --> u
     CBP_st --> m
     CBP_r --> p
     CBC_f --> 3
     CBC_d --> u
     CBC_st --> m
     CBC_r --> p
     CMM_f --> 3
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 3
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 73 ]]]]
[[[[ Begin step 74 [[[[
    transition: internal CPC_fumi in automaton controller
%%%% State variables:
     CPP_f --> 3
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
     CPC_f --> 3
     CPC_d --> u
     CPC_st --> m
     CPC_r --> i
     CBP_f --> 3
     CBP_d --> u
     CBP_st --> m
     CBP_r --> p
     CBC_f --> 3
     CBC_d --> u
     CBC_st --> m
     CBC_r --> p
     CMM_f --> 3
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 3
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 74 ]]]]
[[[[ Begin step 75 [[[[
    transition: internal CBP_fump in automaton controller
%%%% State variables:
     CPP_f --> 3
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
     CPC_f --> 3
     CPC_d --> u
     CPC_st --> m
    CPC_r --> i
    CBP_f --> 4
    CBP_d --> u
    CBP_st --> m
    CBP_r --> p
    CBC_f --> 3
    CBC_d --> u
    CBC_st --> m
    CBC_r --> p
    CMM_f --> 3
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 3
    CMC_d --> u
    CMC_st --> m
    CP_a --> false
    CB_a --> true
    CP_b --> false
    CB_b --> false
    CP_ps --> ()
    CB_abs --> (5)
    CB_dbs --> (6)
]]]] End step 75 ]]]]
[[[[ Begin step 76 [[[[
   transition: internal CBC_fump in automaton controller
%%%% State variables:
    CPP_f --> 3
    CPP_d --> u
    CPP_st --> m
    CPP_r --> i
    CPC_f --> 3
    CPC_d --> u
    CPC_st --> m
    CPC_r --> i
    CBP_f --> 4
    CBP_d --> u
    CBP_st --> m
    CBP_r --> p
    CBC_f --> 4
    CBC_d --> u
    CBC_st --> m
    CBC_r --> p
    CMM_f --> 3
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 3
```

```
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 76 ]]]]
[[[[ Begin step 77 [[[[
    transition: internal CPP_fumi in automaton controller
%%%% State variables:
     CPP_f --> 4
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
     CPC_f --> 3
     CPC_d --> u
     CPC_st --> m
     CPC_r --> i
     CBP_f --> 4
     CBP_d --> u
     CBP_st --> m
     CBP_r --> p
     CBC_f --> 4
     CBC_d --> u
     CBC_st --> m
     CBC_r --> p
     CMM_f --> 3
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 3
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 77 ]]]]
[[[[ Begin step 78 [[[[
    transition: internal CPC_fumi in automaton controller
%%%% State variables:
    CPP_f --> 4
    CPP_d --> u
    CPP_st --> m
    CPP_r --> i
    CPC_f --> 4
    CPC_d --> u
    CPC_st --> m
    CPC_r --> i
    CBP_f --> 4
    CBP_d --> u
    CBP_st --> m
    CBP_r --> p
    CBC_f --> 4
    CBC_d --> u
    CBC_st --> m
    CBC_r --> p
    CMM_f --> 3
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 3
    CMC_d --> u
    CMC_st --> m
    CP_a --> false
    CB_a --> true
    CP_b --> false
    CB_b --> false
    CP_ps --> ()
    CB_abs --> (5)
    CB_dbs --> (6)
]]]] End step 78 ]]]]
[[[[ Begin step 79 [[[[
   transition: internal CMM_fmu in automaton controller
%%%% State variables:
    CPP_f --> 4
    CPP_d --> u
    CPP_st --> m
    CPP_r --> i
    CPC_f --> 4
    CPC_d --> u
    CPC_st --> m
    CPC_r --> i
    CBP_f --> 4
    CBP_d --> u
    CBP_st --> m
    CBP_r --> p
    CBC_f --> 4
```

```
     CBC_d --> u
     CBC_st --> m
     CBC_r --> p
     CMM_f --> 4
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 3
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 79 ]]]]
[[[[ Begin step 80 [[[[
    transition: internal CMC_fmu in automaton controller
%%%% State variables:
     CPP_f --> 4
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
     CPC_f --> 4
     CPC_d --> u
     CPC_st --> m
     CPC_r --> i
     CBP_f --> 4
     CBP_d --> u
     CBP_st --> m
     CBP_r --> p
     CBC_f --> 4
     CBC_d --> u
     CBC_st --> m
     CBC_r --> p
     CMM_f --> 4
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 4
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
    CP_ps --> ()
    CB_abs --> (5)
    CB_dbs --> (6)
]]]] End step 80 ]]]]
[[[[ Begin step 81 [[[[
   transition: internal CBP_fump in automaton controller
%%%% State variables:
    CPP_f --> 4
    CPP_d --> u
    CPP_st --> m
    CPP_r --> i
    CPC_f --> 4
    CPC_d --> u
    CPC_st --> m
    CPC_r --> i
    CBP_f --> 5
    CBP_d --> u
    CBP_st --> m
    CBP_r --> p
    CBC_f --> 4
    CBC_d --> u
    CBC_st --> m
    CBC_r --> p
    CMM_f --> 4
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 4
    CMC_d --> u
    CMC_st --> m
    CP_a --> false
    CB_a --> true
    CP_b --> false
    CB_b --> false
    CP_ps --> ()
    CB_abs --> (5)
    CB_dbs --> (6)
]]]] End step 81 ]]]]
[[[[ Begin step 82 [[[[
   transition: internal CBC_fump in automaton controller
%%%% State variables:
    CPP_f --> 4
    CPP_d --> u
    CPP_st --> m
    CPP_r --> i
    CPC_f --> 4
    CPC_d --> u
    CPC_st --> m
    CPC_r --> i
    CBP_f --> 5
    CBP_d --> u
    CBP_st --> m
    CBP_r --> p
    CBC_f --> 5
    CBC_d --> u
    CBC_st --> m
    CBC_r --> p
    CMM_f --> 4
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 4
    CMC_d --> u
    CMC_st --> m
    CP_a --> false
    CB_a --> true
    CP_b --> false
    CB_b --> false
    CP_ps --> ()
    CB_abs --> (5)
    CB_dbs --> (6)
]]]] End step 82 ]]]]
[[[[ Begin step 83 [[[[
   transition: internal CPP_fumi in automaton controller
%%%% State variables:
    CPP_f --> 5
    CPP_d --> u
    CPP_st --> m
    CPP_r --> i
    CPC_f --> 4
    CPC_d --> u
    CPC_st --> m
    CPC_r --> i
    CBP_f --> 5
    CBP_d --> u
    CBP_st --> m
    CBP_r --> p
    CBC_f --> 5
    CBC_d --> u
    CBC_st --> m
    CBC_r --> p
    CMM_f --> 4
    CMM_d --> u
    CMM_st --> m
```

```
     CMC_f --> 4
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 83 ]]]]
[[[[ Begin step 84 [[[[
    transition: internal CPC_fumi in automaton controller
%%%% State variables:
     CPP_f --> 5
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
     CPC_f --> 5
     CPC_d --> u
     CPC_st --> m
     CPC_r --> i
     CBP_f --> 5
     CBP_d --> u
     CBP_st --> m
     CBP_r --> p
     CBC_f --> 5
     CBC_d --> u
     CBC_st --> m
     CBC_r --> p
     CMM_f --> 4
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 4
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 84 ]]]]
[[[[ Begin step 85 [[[[
    transition: internal CMM_fmu in automaton controller
```

```
%%%% State variables:
    CPP_f --> 5
    CPP_d --> u
    CPP_st --> m
    CPP_r --> i
    CPC_f --> 5
    CPC_d --> u
    CPC_st --> m
    CPC_r --> i
    CBP_f --> 5
    CBP_d --> u
    CBP_st --> m
    CBP_r --> p
    CBC_f --> 5
    CBC_d --> u
    CBC_st --> m
    CBC_r --> p
    CMM_f --> 5
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 4
    CMC_d --> u
    CMC_st --> m
    CP_a --> false
    CB_a --> true
    CP_b --> false
    CB_b --> false
    CP_ps --> ()
    CB_abs --> (5)
    CB_dbs --> (6)
]]]] End step 85 ]]]]
[[[[ Begin step 86 [[[[
   transition: internal CMC_fmu in automaton controller
%%%% State variables:
    CPP_f --> 5
    CPP_d --> u
    CPP_st --> m
    CPP_r --> i
    CPC_f --> 5
    CPC_d --> u
    CPC_st --> m
    CPC_r --> i
    CBP_f --> 5
    CBP_d --> u
    CBP_st --> m
    CBP_r --> p
```

```
     CBC_f --> 5
     CBC_d --> u
     CBC_st --> m
     CBC_r --> p
     CMM_f --> 5
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 5
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 86 ]]]]
[[[[ Begin step 87 [[[[
    transition: output stop in automaton controller
%%%% State variables:
     CPP_f --> 5
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
     CPC_f --> 5
     CPC_d --> u
     CPC_st --> m
     CPC_r --> i
     CBP_f --> 5
     CBP_d --> u
     CBP_st --> m
     CBP_r --> p
     CBC_f --> 5
     CBC_d --> u
     CBC_st --> m
     CBC_r --> p
     CMM_f --> 5
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 5
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> (5)
     CB_dbs --> (6)
]]]] End step 87 ]]]]
[[[[ Begin step 88 [[[[
    transition: output satb(5, u) in automaton controller
%%%% State variables:
     CPP_f --> 5
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
     CPC_f --> 5
     CPC_d --> u
     CPC_st --> m
     CPC_r --> i
     CBP_f --> 5
     CBP_d --> u
     CBP_st --> m
     CBP_r --> p
     CBC_f --> 5
     CBC_d --> u
     CBC_st --> m
     CBC_r --> p
     CMM_f --> 5
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 5
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> (6)
]]]] End step 88 ]]]]
[[[[ Begin step 89 [[[[
    transition: internal CBP_fusp in automaton controller
%%%% State variables:
     CPP_f --> 5
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
     CPC_f --> 5
    CPC_d --> u
    CPC_st --> m
    CPC_r --> i
    CBP_f --> 5
    CBP_d --> u
    CBP_st --> s
    CBP_r --> p
    CBC_f --> 5
    CBC_d --> u
    CBC_st --> m
    CBC_r --> p
    CMM_f --> 5
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 5
    CMC_d --> u
    CMC_st --> m
    CP_a --> false
    CB_a --> true
    CP_b --> false
    CB_b --> false
    CP_ps --> ()
    CB_abs --> ()
    CB_dbs --> (6)
]]]] End step 89 ]]]]
[[[[ Begin step 90 [[[[
   transition: internal CBC_fusp in automaton controller
%%%% State variables:
    CPP_f --> 5
    CPP_d --> u
    CPP_st --> m
    CPP_r --> i
    CPC_f --> 5
    CPC_d --> u
    CPC_st --> m
    CPC_r --> i
    CBP_f --> 5
    CBP_d --> u
    CBP_st --> s
    CBP_r --> p
    CBC_f --> 5
    CBC_d --> u
    CBC_st --> s
    CBC_r --> p
    CMM_f --> 5
    CMM_d --> u
```

```
     CMM_st --> m
     CMC_f --> 5
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> (6)
]]]] End step 90 ]]]]
[[[[ Begin step 91 [[[[
    transition: internal CMM_fsu in automaton controller
%%%% State variables:
     CPP_f --> 5
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
     CPC_f --> 5
     CPC_d --> u
     CPC_st --> m
     CPC_r --> i
     CBP_f --> 5
     CBP_d --> u
     CBP_st --> s
     CBP_r --> p
     CBC_f --> 5
     CBC_d --> u
     CBC_st --> s
     CBC_r --> p
     CMM_f --> 5
     CMM_d --> u
     CMM_st --> s
     CMC_f --> 5
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> (6)
]]]] End step 91 ]]]]
[[[[ Begin step 92 [[[[
    transition: internal CMC_fsu in automaton controller
%%%% State variables:
     CPP_f --> 5
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
     CPC_f --> 5
     CPC_d --> u
     CPC_st --> m
     CPC_r --> i
     CBP_f --> 5
     CBP_d --> u
     CBP_st --> s
     CBP_r --> p
     CBC_f --> 5
     CBC_d --> u
     CBC_st --> s
     CBC_r --> p
     CMM_f --> 5
     CMM_d --> u
     CMM_st --> s
     CMC_f --> 5
     CMC_d --> u
     CMC_st --> s
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> (6)
]]]] End step 92 ]]]]
[[[[ Begin step 93 [[[[
    transition: output up in automaton controller
%%%% State variables:
     CPP_f --> 5
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
     CPC_f --> 5
     CPC_d --> u
     CPC_st --> m
     CPC_r --> i
     CBP_f --> 5
     CBP_d --> u
     CBP_st --> s
     CBP_r --> p
     CBC_f --> 5
     CBC_d --> u
     CBC_st --> s
     CBC_r --> p
     CMM_f --> 5
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 5
     CMC_d --> u
     CMC_st --> s
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> (6)
]]]] End step 93 ]]]]
[[[[ Begin step 94 [[[[
    transition: internal CPP_fumi in automaton controller
%%%% State variables:
     CPP_f --> 6
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
     CPC_f --> 5
     CPC_d --> u
     CPC_st --> m
     CPC_r --> i
     CBP_f --> 5
     CBP_d --> u
     CBP_st --> s
     CBP_r --> p
     CBC_f --> 5
     CBC_d --> u
     CBC_st --> s
     CBC_r --> p
     CMM_f --> 5
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 5
     CMC_d --> u
     CMC_st --> s
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> (6)
]]]] End step 94 ]]]]
[[[[ Begin step 95 [[[[
    transition: internal CPC_fumi in automaton controller
%%%% State variables:
     CPP_f --> 6
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
     CPC_f --> 6
     CPC_d --> u
     CPC_st --> m
     CPC_r --> i
     CBP_f --> 5
     CBP_d --> u
     CBP_st --> s
     CBP_r --> p
     CBC_f --> 5
     CBC_d --> u
     CBC_st --> s
     CBC_r --> p
     CMM_f --> 5
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 5
     CMC_d --> u
     CMC_st --> s
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> (6)
]]]] End step 95 ]]]]
[[[[ Begin step 96 [[[[
    transition: internal CMM_fmu in automaton controller
%%%% State variables:
     CPP_f --> 6
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
    CPC_f --> 6
    CPC_d --> u
    CPC_st --> m
    CPC_r --> i
    CBP_f --> 5
    CBP_d --> u
    CBP_st --> s
    CBP_r --> p
    CBC_f --> 5
    CBC_d --> u
    CBC_st --> s
    CBC_r --> p
    CMM_f --> 6
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 5
    CMC_d --> u
    CMC_st --> s
    CP_a --> false
    CB_a --> true
    CP_b --> false
    CB_b --> false
    CP_ps --> ()
    CB_abs --> ()
    CB_dbs --> (6)
]]]] End step 96 ]]]]
[[[[ Begin step 97 [[[[
   transition: internal CMC_fmu in automaton controller
%%%% State variables:
    CPP_f --> 6
    CPP_d --> u
    CPP_st --> m
    CPP_r --> i
    CPC_f --> 6
    CPC_d --> u
    CPC_st --> m
    CPC_r --> i
    CBP_f --> 5
    CBP_d --> u
    CBP_st --> s
    CBP_r --> p
    CBC_f --> 5
    CBC_d --> u
    CBC_st --> s
    CBC_r --> p
    CMM_f --> 6
```

```
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 6
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> (6)
]]]] End step 97 ]]]]
[[[[ Begin step 98 [[[[
    transition: internal CBP_fump in automaton controller
%%%% State variables:
     CPP_f --> 6
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
     CPC_f --> 6
     CPC_d --> u
     CPC_st --> m
     CPC_r --> i
     CBP_f --> 6
     CBP_d --> u
     CBP_st --> m
     CBP_r --> p
     CBC_f --> 5
     CBC_d --> u
     CBC_st --> s
     CBC_r --> p
     CMM_f --> 6
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 6
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> (6)
]]]] End step 98 ]]]]
```

```
[[[[ Begin step 99 [[[[
   transition: internal CBC_fump in automaton controller
%%%% State variables:
    CPP_f --> 6
    CPP_d --> u
    CPP_st --> m
    CPP_r --> i
    CPC_f --> 6
    CPC_d --> u
    CPC_st --> m
    CPC_r --> i
    CBP_f --> 6
    CBP_d --> u
    CBP_st --> m
    CBP_r --> p
    CBC_f --> 6
    CBC_d --> u
    CBC_st --> m
    CBC_r --> p
    CMM_f --> 6
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 6
    CMC_d --> u
    CMC_st --> m
    CP_a --> false
    CB_a --> true
    CP_b --> false
    CB_b --> false
    CP_ps --> ()
    CB_abs --> ()
    CB_dbs --> (6)
]]]] End step 99 ]]]]
[[[[ Begin step 100 [[[[
   transition: output stop in automaton controller
%%%% State variables:
    CPP_f --> 6
    CPP_d --> u
    CPP_st --> m
    CPP_r --> i
    CPC_f --> 6
    CPC_d --> u
    CPC_st --> m
    CPC_r --> i
    CBP_f --> 6
    CBP_d --> u
```

```
     CBP_st --> m
     CBP_r --> p
     CBC_f --> 6
     CBC_d --> u
     CBC_st --> m
     CBC_r --> p
     CMM_f --> 6
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 6
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> (6)
]]]] End step 100 ]]]]
[[[[ Begin step 101 [[[[
    transition: internal CBC_fsii in automaton controller
%%%% State variables:
     CPP_f --> 6
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
     CPC_f --> 6
     CPC_d --> u
     CPC_st --> m
     CPC_r --> i
     CBP_f --> 6
     CBP_d --> u
     CBP_st --> m
     CBP_r --> p
     CBC_f --> 6
     CBC_d --> s
     CBC_st --> ii
     CBC_r --> null
     CMM_f --> 6
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 6
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> (6)
]]]] End step 101 ]]]]
[[[[ Begin step 102 [[[[
    transition: output satb(6, s) in automaton controller
%%%% State variables:
     CPP_f --> 6
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
     CPC_f --> 6
     CPC_d --> u
     CPC_st --> m
     CPC_r --> i
     CBP_f --> 6
     CBP_d --> u
     CBP_st --> m
     CBP_r --> p
     CBC_f --> 6
     CBC_d --> s
     CBC_st --> ii
     CBC_r --> null
     CMM_f --> 6
     CMM_d --> u
     CMM_st --> m
     CMC_f --> 6
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 102 ]]]]
[[[[ Begin step 103 [[[[
    transition: internal CBP_fsii in automaton controller
%%%% State variables:
     CPP_f --> 6
     CPP_d --> u
     CPP_st --> m
    CPP_r --> i
    CPC_f --> 6
    CPC_d --> u
    CPC_st --> m
    CPC_r --> i
    CBP_f --> 6
    CBP_d --> s
    CBP_st --> ii
    CBP_r --> null
    CBC_f --> 6
    CBC_d --> s
    CBC_st --> ii
    CBC_r --> null
    CMM_f --> 6
    CMM_d --> u
    CMM_st --> m
    CMC_f --> 6
    CMC_d --> u
    CMC_st --> m
    CP_a --> false
    CB_a --> true
    CP_b --> false
    CB_b --> false
    CP_ps --> ()
    CB_abs --> ()
    CB_dbs --> ()
]]]] End step 103 ]]]]
[[[[ Begin step 104 [[[[
   transition: internal CMM_fsu in automaton controller
%%%% State variables:
    CPP_f --> 6
    CPP_d --> u
    CPP_st --> m
    CPP_r --> i
    CPC_f --> 6
    CPC_d --> u
    CPC_st --> m
    CPC_r --> i
    CBP_f --> 6
    CBP_d --> s
    CBP_st --> ii
    CBP_r --> null
    CBC_f --> 6
    CBC_d --> s
    CBC_st --> ii
    CBC_r --> null
```

```
     CMM_f --> 6
     CMM_d --> u
     CMM_st --> s
     CMC_f --> 6
     CMC_d --> u
     CMC_st --> m
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> ()
]]]] End step 104 ]]]]
[[[[ Begin step 105 [[[[
    transition: internal CMC_fsu in automaton controller
%%%% State variables:
     CPP_f --> 6
     CPP_d --> u
     CPP_st --> m
     CPP_r --> i
     CPC_f --> 6
     CPC_d --> u
     CPC_st --> m
     CPC_r --> i
     CBP_f --> 6
     CBP_d --> s
     CBP_st --> ii
     CBP_r --> null
     CBC_f --> 6
     CBC_d --> s
     CBC_st --> ii
     CBC_r --> null
     CMM_f --> 6
     CMM_d --> u
     CMM_st --> s
     CMC_f --> 6
     CMC_d --> u
     CMC_st --> s
     CP_a --> false
     CB_a --> true
     CP_b --> false
     CB_b --> false
     CP_ps --> ()
     CB_abs --> ()
     CB_dbs --> ()
```

```
]]]] End step 105 ]]]]
No errors
```