R-base-ma[11,40]

R-base-ma(R)
$\equiv_{\text{def}}$ case R of
Rnone =>
Rplus(*left*,*right*)=>rec_1,rec_2.
Rinit(loc,T,x,V)=> case V
of inl(v) => x : T initially $x = \lambda t.v$
| inr(v) => x : T initially $x = v$
Rframe(loc,T,x,L)=> only members of L affect x :T
Rsframe(*lnk*,*tag*,L)=> only L sends on (*lnk* with *tag*)
Reffect(loc,*ds*,*knd*,T,x,F)=> case F
of inl(f) => with declarations
ds:*ds*
da:*knd* : T
effect of *knd*(v) is $x := \lambda s,v,t.\ f(s,v)$ s v
| inr(f) => with declarations
ds:*ds*
da:*knd* : T
effect of *knd*(v) is $x := f$ s v
Rsends(*ds*,*knd*,T,l,*dt*,g)=> with declarations
ds:*ds*
da:*knd* : $T \oplus$ lnk-decl(l;*dt*)
knd(v) sends g s v on link l
Rpre(loc,*ds*,a,p,P)=> (precondition a:Outcome(p) is
P:State(*ds*) -> Bool)
Rkframe(loc,k,L)=> k affects only members of L
Rksframe(loc,k,L)=> k sends only on links in L
Rrframe(loc,x,L)=> only members of L read x

clarification:

R-base-ma(R)
$\equiv_{\text{def}}$ case R of
Rnone =>
Rplus(*left*,*right*)=>rec_1,rec_2.
Rinit(loc,T,x,V)=> case V
of inl(v) => x : T initially $x = \lambda t.v$
| inr(v) => x : T initially $x = v$
Rframe(loc,T,x,L)=> only members of L affect x :T
Rsframe(*lnk*,*tag*,L)=> only L sends on (*lnk* with *tag*)
Reffect(loc,*ds*,*knd*,T,x,F)=> case F
of inl(f) => with declarations
ds:*ds*

da:knd : T
effect of knd(v) is $x := \lambda s,v,t.\ f(s,v)$ s v
| inr(f) => with declarations
ds:ds
da:knd : T
effect of knd(v) is $x := f$ s v
Rsends(ds,knd,T,l,dt,g)=> with declarations
ds:ds
da:fpf-join(KindDeq;knd : T;lnk-decl(l;dt))
knd(v) sends g s v on link l
Rpre(loc,ds,a,p,P)=> (precondition a:Outcome(p) is
P:State(ds) -> Bool)
Rkframe(loc,k,L)=> k affects only members of L
Rksframe(loc,k,L)=> k sends only on links in L
Rrframe(loc,x,L)=> only members of L read x

---

http://www.nuprl.org/FDLcontent/p0_963683_/p41_446081_{R-base-ma}.html