```
trigger1__realizes  THM

⊢  ∀T,A:Type, P:(A→T→𝔹), i:Id, k:Knd, a,x:Id.
|  A
|  ⇒
|  T
|  ⇒
|  ¬x = "trigger"
|  ⇒
|  ¬locl(a) = k
|  ⇒
|  λloc.⊕(trigger1(loc;T;A;P;i;k;a;x))
|  realizes es.(vartype(i;x) ⊆r A)
|              & ∀e@i.kind(e) = k ⇒ (valtype(e) ⊆r T)
|              & ∀e'@i.kind(e') = locl(a)
|                      ⇒
|                (∃e:E.
|                 (e <loc e') & kind(e) = k & P((x when e),val(e)))
|              & ∀e@i.kind(e) = k
|                     ⇒
|                  P((x when e),val(e)) ⇒ ∃e'@i.kind(e') = locl(a)
| by RealizerSetup
|    THEN
|    InstRealizerLemma `recognizer1` [T;A;P;k;i;"trigger";x]
|
1. T : Type
2. A : Type
3. P : A→T→𝔹
4. i : Id
5. k : Knd
6. a : Id
7. x : Id
8. A
9. T
10. ¬x = "trigger"
11. ¬locl(a) = k
12. (λloc.⊕(trigger1(loc;T;A;P;i;k;a;x))) ∈ Dsys
13. D' : Dsys
14. λloc.⊕(trigger1(loc;T;A;P;i;k;a;x)) ⊆ D'
15. w : World
16. p : FairFifo
17. PossibleWorld(D';w)
18. ∀e@i.kind(e) = k ⇒ (valtype(e) ⊆r T) & (vartype(i;x) ⊆r A)
|   & ∀e@i.first(e) ⇒ ("trigger" when e) = false₂
|   & ∀e'@i.("trigger" after e')
|           ⟺
|           (∃e:E.
|            (e <loc e') ∨ e = e'
|            & kind(e) = k
|            & P((x when e),val(e)))
⊢  (vartype(i;x) ⊆r A) & ∀e@i.kind(e) = k ⇒ (valtype(e) ⊆r T)
|  & ∀e'@i.kind(e') = locl(a)
|          ⇒
|    (∃e:E. (e <loc e') & kind(e) = k & P((x when e),val(e)))
|  & ∀e@i.kind(e) = k
```

```
|         ⇒
|    P((x when e),val(e)) ⇒ ∃e'@i.kind(e') = locl(a)
| by PreRule1 i a "trigger" 𝔹 Unit (λ2x,v. x)
|
1. T : Type
2. A : Type
3. P : A→T→𝔹
4. i : Id
5. k : Knd
6. a : Id
7. x : Id
8. A
9. T
10. ¬x = "trigger"
11. ¬locl(a) = k
12. (λloc.⊕(trigger1(loc;T;A;P;i;k;a;x))) ∈ Dsys
13. D' : Dsys
14. λloc.⊕(trigger1(loc;T;A;P;i;k;a;x)) ⊆ D'
15. w : World
16. p : FairFifo
17. PossibleWorld(D';w)
18. ∀e@i.kind(e) = k ⇒ (valtype(e) ⊆r T)
19. vartype(i;x) ⊆r A
20. ∀e@i.first(e) ⇒ ("trigger" when e) = false2
21. ∀e'@i.("trigger" after e')
|         ⟺
|         (∃e:E.
|          (e <loc e') ∨ e = e' & kind(e) = k & P((x when e),val(e)))
22. @i: ma-single-pre1("trigger";𝔹;a;Unit;x,v.x) ∈ Dsys
23. vartype(i;"trigger") ⊆r 𝔹
24. ∀e:E. loc(e) = i ⇒ kind(e) = locl(a) ⇒ (valtype(e) ⊆r Unit)
25. ∀e:E.
|   loc(e) = i
|   ⇒
|   (kind(e) = locl(a) ⇒ ("trigger" when e))
|   & (∃e':E.
|      (e <loc e') ∨ e = e'
|      & kind(e') = locl(a) ∨ (∀v:Unit. ¬("trigger" after e')))
⊢  (vartype(i;x) ⊆r A) & ∀e@i.kind(e) = k ⇒ (valtype(e) ⊆r T)
|  & ∀e'@i.kind(e') = locl(a)
|          ⇒
|    (∃e:E. (e <loc e') & kind(e) = k & P((x when e),val(e)))
|  & ∀e@i.kind(e) = k
|         ⇒
|    P((x when e),val(e)) ⇒ ∃e'@i.kind(e') = locl(a)
| by SplitAndConcl THEN Try Trivial THEN All (Unfold `alle-at`)
|

|\
| 1. T : Type
| 2. A : Type
| 3. P : A→T→𝔹
| 4. i : Id
| 5. k : Knd
| 6. a : Id
| 7. x : Id
```

```
| 8. A
| 9. T
| 10. ¬x = "trigger"
| 11. ¬locl(a) = k
| 12. (λloc.⊕(trigger1(loc;T;A;P;i;k;a;x))) ∈ Dsys
| 13. D' : Dsys
| 14. λloc.⊕(trigger1(loc;T;A;P;i;k;a;x)) ⊆ D'
| 15. w : World
| 16. p : FairFifo
| 17. PossibleWorld(D';w)
| 18. ∀e:E. loc(e) = i ⇒ kind(e) = k ⇒ (valtype(e) ⊆r T)
| 19. vartype(i;x) ⊆r A
| 20. ∀e:E. loc(e) = i ⇒ first(e) ⇒ ("trigger" when e) = false₂
| 21. ∀e':E.
| |   loc(e') = i
| |   ⇒
| |   (("trigger" after e')
| |    ⟺
| |    (∃e:E.
| |     (e <loc e') ∨ e = e' & kind(e) = k & P((x when e),val(e))))
| 22. @i: ma-single-pre1("trigger";𝔹;a;Unit;x,v.x) ∈ Dsys
| 23. vartype(i;"trigger") ⊆r 𝔹
| 24. ∀e:E. loc(e) = i ⇒ kind(e) = locl(a) ⇒ (valtype(e) ⊆r Unit)
| 25. ∀e:E.
| |   loc(e) = i
| |   ⇒
| |   (kind(e) = locl(a) ⇒ ("trigger" when e))
| |   & (∃e':E.
| |      (e <loc e') ∨ e = e'
| |      & kind(e') = locl(a) ∨ (∀v:Unit. ¬("trigger" after e')))
| ⊢  ∀e':E.
| |  loc(e') = i
| |  ⇒
| |  kind(e') = locl(a)
| |  ⇒
| |  (∃e:E. (e <loc e') & kind(e) = k & P((x when e),val(e)))
| | by RepeatFor 2 (ParallelOp -1) THEN Analyze -1 THEN ParallelOp -2
| |
| 1. T : Type
| 2. A : Type
| 3. P : A→T→𝔹
| 4. i : Id
| 5. k : Knd
| 6. a : Id
| 7. x : Id
| 8. A
| 9. T
| 10. ¬x = "trigger"
| 11. ¬locl(a) = k
| 12. (λloc.⊕(trigger1(loc;T;A;P;i;k;a;x))) ∈ Dsys
| 13. D' : Dsys
| 14. λloc.⊕(trigger1(loc;T;A;P;i;k;a;x)) ⊆ D'
| 15. w : World
| 16. p : FairFifo
| 17. PossibleWorld(D';w)
```

```
| 18. ∀e:E. loc(e) = i ⇒ kind(e) = k ⇒ (valtype(e) ⊆r T)
| 19. vartype(i;x) ⊆r A
| 20. ∀e:E. loc(e) = i ⇒ first(e) ⇒ ("trigger" when e) = false₂
| 21. ∀e':E.
| |   loc(e') = i
| |   ⇒
| |   (("trigger" after e')
| |    ⟺
| |    (∃e:E.
| |     (e <loc e') ∨ e = e' & kind(e) = k & P((x when e),val(e))))
| 22. @i: ma-single-pre1("trigger";𝔹;a;Unit;x,v.x) ∈ Dsys
| 23. vartype(i;"trigger") ⊆r 𝔹
| 24. ∀e:E. loc(e) = i ⇒ kind(e) = locl(a) ⇒ (valtype(e) ⊆r Unit)
| 25. ∀e:E.
| |   loc(e) = i
| |   ⇒
| |   (kind(e) = locl(a) ⇒ ("trigger" when e))
| |   & (∃e':E.
| |      (e <loc e') ∨ e = e'
| |      & kind(e') = locl(a) ∨ (∀v:Unit. ¬("trigger" after e')))
| 26. e' : E
| 27. loc(e') = i
| 28. ∃e'@0:E.
| |   (e' <loc e'@0) ∨ e' = e'@0
| |   & kind(e'@0) = locl(a) ∨ (∀v:Unit. ¬("trigger" after e'@0))
| 29. kind(e') = locl(a)
| 30. ("trigger" when e')
| ⊢  ∃e:E. (e <loc e') & kind(e) = k & P((x when e),val(e))
| | by Decide first(e')
| |
| |\
| | 1. T : Type
| | 2. A : Type
| | 3. P : A→T→𝔹
| | 4. i : Id
| | 5. k : Knd
| | 6. a : Id
| | 7. x : Id
| | 8. A
| | 9. T
| | 10. ¬x = "trigger"
| | 11. ¬locl(a) = k
| | 12. (λloc.⊕(trigger1(loc;T;A;P;i;k;a;x))) ∈ Dsys
| | 13. D' : Dsys
| | 14. λloc.⊕(trigger1(loc;T;A;P;i;k;a;x)) ⊆ D'
| | 15. w : World
| | 16. p : FairFifo
| | 17. PossibleWorld(D';w)
| | 18. ∀e:E. loc(e) = i ⇒ kind(e) = k ⇒ (valtype(e) ⊆r T)
| | 19. vartype(i;x) ⊆r A
| | 20. ∀e:E. loc(e) = i ⇒ first(e) ⇒ ("trigger" when e) = false₂
| | 21. ∀e':E.
| |     loc(e') = i
| |     ⇒
| |     (("trigger" after e')
```

```
| |      ⟺
| |      (∃e:E.
| |       (e <loc e') ∨ e = e' & kind(e) = k & P((x when e),val(e))))
| | 22. @i: ma-single-pre1("trigger";𝔹;a;Unit;x,v.x) ∈ Dsys
| | 23. vartype(i;"trigger") ⊆r 𝔹
| | 24. ∀e:E. loc(e) = i ⇒ kind(e) = locl(a) ⇒ (valtype(e) ⊆r Unit)
| | 25. ∀e:E.
| |     loc(e) = i
| |     ⇒
| |     (kind(e) = locl(a) ⇒ ("trigger" when e))
| |     & (∃e':E.
| |        (e <loc e') ∨ e = e'
| |        & kind(e') = locl(a) ∨ (∀v:Unit. ¬("trigger" after e')))
| | 26. e' : E
| | 27. loc(e') = i
| | 28. ∃e'@0:E.
| |     (e' <loc e'@0) ∨ e' = e'@0
| |     & kind(e'@0) = locl(a) ∨ (∀v:Unit. ¬("trigger" after e'@0))
| | 29. kind(e') = locl(a)
| | 30. ("trigger" when e')
| | 31. first(e')
| | ⊢  ∃e:E. (e <loc e') & kind(e) = k & P((x when e),val(e))
| |   by AllHyps
| |      (λh.
| |       ((RW (SweepDnC (HypC h)) -2) THEN (Reduce -2))
| |       THEN
| |       Complete Auto)
| |
| .\
| . 1. T : Type
| . 2. A : Type
| . 3. P : A→T→𝔹
| . 4. i : Id
| . 5. k : Knd
| . 6. a : Id
| . 7. x : Id
| . 8. A
| . 9. T
| . 10. ¬x = "trigger"
| . 11. ¬locl(a) = k
| . 12. (λloc.⊕(trigger1(loc;T;A;P;i;k;a;x))) ∈ Dsys
| . 13. D' : Dsys
| . 14. λloc.⊕(trigger1(loc;T;A;P;i;k;a;x)) ⊆ D'
| . 15. w : World
| . 16. p : FairFifo
| . 17. PossibleWorld(D';w)
| . 18. ∀e:E. loc(e) = i ⇒ kind(e) = k ⇒ (valtype(e) ⊆r T)
| . 19. vartype(i;x) ⊆r A
| . 20. ∀e:E. loc(e) = i ⇒ first(e) ⇒ ("trigger" when e) = false₂
| . 21. ∀e':E.
| . |   loc(e') = i
| . |   ⇒
| . |   (("trigger" after e')
| . |    ⟺
| . |    (∃e:E.
```

```
| . |     (e <loc e') ∨ e = e' & kind(e) = k & P((x when e),val(e))))
| . 22. @i: ma-single-pre1("trigger";𝔹;a;Unit;x,v.x) ∈ Dsys
| . 23. vartype(i;"trigger") ⊆r 𝔹
| . 24. ∀e:E. loc(e) = i ⇒ kind(e) = locl(a) ⇒ (valtype(e) ⊆r Unit)
| . 25. ∀e:E.
| . |   loc(e) = i
| . |   ⇒
| . |   (kind(e) = locl(a) ⇒ ("trigger" when e))
| . |   & (∃e':E.
| . |      (e <loc e') ∨ e = e'
| . |      & kind(e') = locl(a) ∨ (∀v:Unit. ¬("trigger" after e')))
| . 26. e' : E
| . 27. loc(e') = i
| . 28. ∃e'@0:E.
| . |   (e' <loc e'@0) ∨ e' = e'@0
| . |   & kind(e'@0) = locl(a) ∨ (∀v:Unit. ¬("trigger" after e'@0))
| . 29. kind(e') = locl(a)
| . 30. ("trigger" when e')
| . 31. ¬first(e')
| . ⊢  ∃e:E. (e <loc e') & kind(e) = k & P((x when e),val(e))
| . | by Assert ("trigger" after pred(e'))
| . |
| . |\
| . | .....assertion .....
| . | 1. T : Type
| . | 2. A : Type
| . | 3. P : A→T→𝔹
| . | 4. i : Id
| . | 5. k : Knd
| . | 6. a : Id
| . | 7. x : Id
| . | 8. A
| . | 9. T
| . | 10. ¬x = "trigger"
| . | 11. ¬locl(a) = k
| . | 12. (λloc.⊕(trigger1(loc;T;A;P;i;k;a;x))) ∈ Dsys
| . | 13. D' : Dsys
| . | 14. λloc.⊕(trigger1(loc;T;A;P;i;k;a;x)) ⊆ D'
| . | 15. w : World
| . | 16. p : FairFifo
| . | 17. PossibleWorld(D';w)
| . | 18. ∀e:E. loc(e) = i ⇒ kind(e) = k ⇒ (valtype(e) ⊆r T)
| . | 19. vartype(i;x) ⊆r A
| . | 20. ∀e:E. loc(e) = i ⇒ first(e) ⇒ ("trigger" when e) = false₂
| . | 21. ∀e':E.
| . |     loc(e') = i
| . |     ⇒
| . |     (("trigger" after e')
| . |      ⇐⇒
| . |      (∃e:E.
| . |       (e <loc e') ∨ e = e'
| . |       & kind(e) = k
| . |       & P((x when e),val(e))))
| . | 22. @i: ma-single-pre1("trigger";𝔹;a;Unit;x,v.x) ∈ Dsys
| . | 23. vartype(i;"trigger") ⊆r 𝔹
```

```
| . | 24. ∀e:E.
| . |     loc(e) = i ⇒ kind(e) = locl(a) ⇒ (valtype(e) ⊆r Unit)
| . | 25. ∀e:E.
| . |     loc(e) = i
| . |     ⇒
| . |     (kind(e) = locl(a) ⇒ ("trigger" when e))
| . |     & (∃e':E.
| . |        (e <loc e') ∨ e = e'
| . |        & kind(e') = locl(a) ∨ (∀v:Unit. ¬("trigger" after e')))
| . | 26. e' : E
| . | 27. loc(e') = i
| . | 28. ∃e'@0:E.
| . |     (e' <loc e'@0) ∨ e' = e'@0
| . |     & kind(e'@0) = locl(a) ∨ (∀v:Unit. ¬("trigger" after e'@0))
| . | 29. kind(e') = locl(a)
| . | 30. ("trigger" when e')
| . | 31. ¬first(e')
| . | ⊢  ("trigger" after pred(e'))
| . |   by Inst
| . |      Thm* ∀es:ES, x:Id, e:E.
| . |           ¬first(e)
| . |           ⇒
| . |           (x after pred(e)) = (x when e) ∈ vartype(loc(e);x)
| . |      {es_DASH_after_DASH_pred}
| . |      [ES(w);"trigger";e']
| . |      THEN
| . |      HypSubst -1 0
| . |      THEN
| . |      MaAuto
| . |      THEN
| . |      UseVartypeHyp
| . |
| . .\
| . . 1. T : Type
| . . 2. A : Type
| . . 3. P : A→T→𝔹
| . . 4. i : Id
| . . 5. k : Knd
| . . 6. a : Id
| . . 7. x : Id
| . . 8. A
| . . 9. T
| . . 10. ¬x = "trigger"
| . . 11. ¬locl(a) = k
| . . 12. (λloc.⊕(trigger1(loc;T;A;P;i;k;a;x))) ∈ Dsys
| . . 13. D' : Dsys
| . . 14. λloc.⊕(trigger1(loc;T;A;P;i;k;a;x)) ⊆ D'
| . . 15. w : World
| . . 16. p : FairFifo
| . . 17. PossibleWorld(D';w)
| . . 18. ∀e:E. loc(e) = i ⇒ kind(e) = k ⇒ (valtype(e) ⊆r T)
| . . 19. vartype(i;x) ⊆r A
| . . 20. ∀e:E. loc(e) = i ⇒ first(e) ⇒ ("trigger" when e) = false₂
| . . 21. ∀e':E.
| . . |   loc(e') = i
```

```
| . . |   ⇒
| . . |   (("trigger" after e')
| . . |    ⟺
| . . |    (∃e:E.
| . . |     (e <loc e') ∨ e = e'
| . . |     & kind(e) = k
| . . |     & P((x when e),val(e))))
| . . 22. @i: ma-single-pre1("trigger";𝔹;a;Unit;x,v.x) ∈ Dsys
| . . 23. vartype(i;"trigger") ⊆r 𝔹
| . . 24. ∀e:E.
| . . |   loc(e) = i ⇒ kind(e) = locl(a) ⇒ (valtype(e) ⊆r Unit)
| . . 25. ∀e:E.
| . . |   loc(e) = i
| . . |   ⇒
| . . |   (kind(e) = locl(a) ⇒ ("trigger" when e))
| . . |   & (∃e':E.
| . . |      (e <loc e') ∨ e = e'
| . . |      & kind(e') = locl(a) ∨ (∀v:Unit. ¬("trigger" after e')))
| . . 26. e' : E
| . . 27. loc(e') = i
| . . 28. ∃e'@0:E.
| . . |   (e' <loc e'@0) ∨ e' = e'@0
| . . |   & kind(e'@0) = locl(a) ∨ (∀v:Unit. ¬("trigger" after e'@0))
| . . 29. kind(e') = locl(a)
| . . 30. ("trigger" when e')
| . . 31. ¬first(e')
| . . 32. ("trigger" after pred(e'))
| . . ⊢  ∃e:E. (e <loc e') & kind(e) = k & P((x when e),val(e))
| . . | by AllHyps
| . . |    (λh.
| . . |     RW (SweepDnC (HypC h)) -1 THENA MaAuto THEN ParallelOp -1
| . . |     THEN
| . . |     ParallelOp -1
| . . |     THEN
| . . |     MaAuto)
| . . |
| . . 1. T : Type
| . . 2. A : Type
| . . 3. P : A→T→𝔹
| . . 4. i : Id
| . . 5. k : Knd
| . . 6. a : Id
| . . 7. x : Id
| . . 8. A
| . . 9. T
| . . 10. ¬x = "trigger"
| . . 11. ¬locl(a) = k
| . . 12. (λloc.⨁(trigger1(loc;T;A;P;i;k;a;x))) ∈ Dsys
| . . 13. D' : Dsys
| . . 14. λloc.⨁(trigger1(loc;T;A;P;i;k;a;x)) ⊆ D'
| . . 15. w : World
| . . 16. p : FairFifo
| . . 17. PossibleWorld(D';w)
| . . 18. ∀e:E. loc(e) = i ⇒ kind(e) = k ⇒ (valtype(e) ⊆r T)
| . . 19. vartype(i;x) ⊆r A
```

```
| . . 20. ∀e:E. loc(e) = i ⇒ first(e) ⇒ ("trigger" when e) = false₂
| . . 21. ∀e':E.
| . .     loc(e') = i
| . .     ⇒
| . .     (("trigger" after e')
| . .      ⟺
| . .      (∃e:E.
| . .       (e <loc e') ∨ e = e'
| . .       & kind(e) = k
| . .       & P((x when e),val(e))))
| . . 22. @i: ma-single-pre1("trigger";𝔹;a;Unit;x,v.x) ∈ Dsys
| . . 23. vartype(i;"trigger") ⊆r 𝔹
| . . 24. ∀e:E.
| . .     loc(e) = i ⇒ kind(e) = locl(a) ⇒ (valtype(e) ⊆r Unit)
| . . 25. ∀e:E.
| . .     loc(e) = i
| . .     ⇒
| . .     (kind(e) = locl(a) ⇒ ("trigger" when e))
| . .     & (∃e':E.
| . .        (e <loc e') ∨ e = e'
| . .        & kind(e') = locl(a) ∨ (∀v:Unit. ¬("trigger" after e')))
| . . 26. e' : E
| . . 27. loc(e') = i
| . . 28. ∃e'@0:E.
| . .     (e' <loc e'@0) ∨ e' = e'@0
| . .     & kind(e'@0) = locl(a) ∨ (∀v:Unit. ¬("trigger" after e'@0))
| . . 29. kind(e') = locl(a)
| . . 30. ("trigger" when e')
| . . 31. ¬first(e')
| . . 32. e : E
| . . 33. (e <loc pred(e')) ∨ e = pred(e')
| . . ⊢  (e <loc e')
| . .   by RW0
| . .      Thm* ∀the_es:ES, e,e':E.
| . .           (e <loc e')
| . .           ⟺
| . .           ¬first(e') & e = pred(e') ∈ E ∨ (e <loc pred(e'))
| . .      {es_DASH_locl_DASH_iff}
| . .      0
| . .      THEN
| . .      Analyze -1
| . .      THENL
| . .      [OrRight;OrLeft]
|
.\
. 1. T : Type
. 2. A : Type
. 3. P : A→T→𝔹
. 4. i : Id
. 5. k : Knd
. 6. a : Id
. 7. x : Id
. 8. A
. 9. T
. 10. ¬x = "trigger"
```

```
. 11. ¬locl(a) = k
. 12. (λloc.⊕(trigger1(loc;T;A;P;i;k;a;x))) ∈ Dsys
. 13. D' : Dsys
. 14. λloc.⊕(trigger1(loc;T;A;P;i;k;a;x)) ⊆ D'
. 15. w : World
. 16. p : FairFifo
. 17. PossibleWorld(D';w)
. 18. ∀e:E. loc(e) = i ⇒ kind(e) = k ⇒ (valtype(e) ⊆r T)
. 19. vartype(i;x) ⊆r A
. 20. ∀e:E. loc(e) = i ⇒ first(e) ⇒ ("trigger" when e) = false₂
. 21. ∀e':E.
. |   loc(e') = i
. |   ⇒
. |   (("trigger" after e')
. |    ⟺
. |    (∃e:E.
. |     (e <loc e') ∨ e = e' & kind(e) = k & P((x when e),val(e))))
. 22. @i: ma-single-pre1("trigger";𝔹;a;Unit;x,v.x) ∈ Dsys
. 23. vartype(i;"trigger") ⊆r 𝔹
. 24. ∀e:E. loc(e) = i ⇒ kind(e) = locl(a) ⇒ (valtype(e) ⊆r Unit)
. 25. ∀e:E.
. |   loc(e) = i
. |   ⇒
. |   (kind(e) = locl(a) ⇒ ("trigger" when e))
. |   & (∃e':E.
. |      (e <loc e') ∨ e = e'
. |      & kind(e') = locl(a) ∨ (∀v:Unit. ¬("trigger" after e')))
. ⊢  ∀e:E.
. |  loc(e) = i
. |  ⇒
. |  kind(e) = k ⇒ P((x when e),val(e)) ⇒ ∃e'@i.kind(e') = locl(a)
. | by MaAuto THEN Unfold `existse-at` 0
. |
. 1. T : Type
. 2. A : Type
. 3. P : A→T→𝔹
. 4. i : Id
. 5. k : Knd
. 6. a : Id
. 7. x : Id
. 8. A
. 9. T
. 10. ¬x = "trigger"
. 11. ¬locl(a) = k
. 12. (λloc.⊕(trigger1(loc;T;A;P;i;k;a;x))) ∈ Dsys
. 13. D' : Dsys
. 14. λloc.⊕(trigger1(loc;T;A;P;i;k;a;x)) ⊆ D'
. 15. w : World
. 16. p : FairFifo
. 17. PossibleWorld(D';w)
. 18. ∀e:E. loc(e) = i ⇒ kind(e) = k ⇒ (valtype(e) ⊆r T)
. 19. vartype(i;x) ⊆r A
. 20. ∀e:E. loc(e) = i ⇒ first(e) ⇒ ("trigger" when e) = false₂
. 21. ∀e':E.
. |   loc(e') = i
```

```
. |   ⇒
. |   (("trigger" after e')
. |    ⟺
. |    (∃e:E.
. |     (e <loc e') ∨ e = e' & kind(e) = k & P((x when e),val(e))))
. 22. @i: ma-single-pre1("trigger";𝔹;a;Unit;x,v.x) ∈ Dsys
. 23. vartype(i;"trigger") ⊆r 𝔹
. 24. ∀e:E. loc(e) = i ⇒ kind(e) = locl(a) ⇒ (valtype(e) ⊆r Unit)
. 25. ∀e:E.
. |   loc(e) = i
. |   ⇒
. |   (kind(e) = locl(a) ⇒ ("trigger" when e))
. |   & (∃e':E.
. |      (e <loc e') ∨ e = e'
. |      & kind(e') = locl(a) ∨ (∀v:Unit. ¬("trigger" after e')))
. 26. e : E
. 27. loc(e) = i
. 28. kind(e) = k
. 29. P((x when e),val(e))
. ⊢  ∃e':E. loc(e') = i & kind(e') = locl(a)
. | by InstHyp [e] -5 THEN Analyze -1 THEN ExRepD THEN Analyze -1
. |
. |\
. | 1. T : Type
. | 2. A : Type
. | 3. P : A→T→𝔹
. | 4. i : Id
. | 5. k : Knd
. | 6. a : Id
. | 7. x : Id
. | 8. A
. | 9. T
. | 10. ¬x = "trigger"
. | 11. ¬locl(a) = k
. | 12. (λloc.⊕(trigger1(loc;T;A;P;i;k;a;x))) ∈ Dsys
. | 13. D' : Dsys
. | 14. λloc.⊕(trigger1(loc;T;A;P;i;k;a;x)) ⊆ D'
. | 15. w : World
. | 16. p : FairFifo
. | 17. PossibleWorld(D';w)
. | 18. ∀e:E. loc(e) = i ⇒ kind(e) = k ⇒ (valtype(e) ⊆r T)
. | 19. vartype(i;x) ⊆r A
. | 20. ∀e:E. loc(e) = i ⇒ first(e) ⇒ ("trigger" when e) = false₂
. | 21. ∀e':E.
. |     loc(e') = i
. |     ⇒
. |     (("trigger" after e')
. |      ⟺
. |      (∃e:E.
. |       (e <loc e') ∨ e = e' & kind(e) = k & P((x when e),val(e))))
. | 22. @i: ma-single-pre1("trigger";𝔹;a;Unit;x,v.x) ∈ Dsys
. | 23. vartype(i;"trigger") ⊆r 𝔹
. | 24. ∀e:E. loc(e) = i ⇒ kind(e) = locl(a) ⇒ (valtype(e) ⊆r Unit)
. | 25. ∀e:E.
. |     loc(e) = i
```

```
. |     ⇒
. |     (kind(e) = locl(a) ⇒ ("trigger" when e))
. |     & (∃e':E.
. |        (e <loc e') ∨ e = e'
. |        & kind(e') = locl(a) ∨ (∀v:Unit. ¬("trigger" after e')))
. | 26. e : E
. | 27. loc(e) = i
. | 28. kind(e) = k
. | 29. P((x when e),val(e))
. | 30. kind(e) = locl(a) ⇒ ("trigger" when e)
. | 31. e' : E
. | 32. (e <loc e') ∨ e = e'
. | 33. kind(e') = locl(a)
. | ⊢  ∃e':E. loc(e') = i & kind(e') = locl(a)
. |   by InstConcl [e'] THEN SplitOrHyps THEN MaAuto
. |
. .\
. . 1. T : Type
. . 2. A : Type
. . 3. P : A→T→𝔹
. . 4. i : Id
. . 5. k : Knd
. . 6. a : Id
. . 7. x : Id
. . 8. A
. . 9. T
. . 10. ¬x = "trigger"
. . 11. ¬locl(a) = k
. . 12. (λloc.⊕(trigger1(loc;T;A;P;i;k;a;x))) ∈ Dsys
. . 13. D' : Dsys
. . 14. λloc.⊕(trigger1(loc;T;A;P;i;k;a;x)) ⊆ D'
. . 15. w : World
. . 16. p : FairFifo
. . 17. PossibleWorld(D';w)
. . 18. ∀e:E. loc(e) = i ⇒ kind(e) = k ⇒ (valtype(e) ⊆r T)
. . 19. vartype(i;x) ⊆r A
. . 20. ∀e:E. loc(e) = i ⇒ first(e) ⇒ ("trigger" when e) = false₂
. . 21. ∀e':E.
. . |   loc(e') = i
. . |   ⇒
. . |   (("trigger" after e')
. . |    ⟺
. . |    (∃e:E.
. . |     (e <loc e') ∨ e = e' & kind(e) = k & P((x when e),val(e))))
. . 22. @i: ma-single-pre1("trigger";𝔹;a;Unit;x,v.x) ∈ Dsys
. . 23. vartype(i;"trigger") ⊆r 𝔹
. . 24. ∀e:E. loc(e) = i ⇒ kind(e) = locl(a) ⇒ (valtype(e) ⊆r Unit)
. . 25. ∀e:E.
. . |   loc(e) = i
. . |   ⇒
. . |   (kind(e) = locl(a) ⇒ ("trigger" when e))
. . |   & (∃e':E.
. . |      (e <loc e') ∨ e = e'
. . |      & kind(e') = locl(a) ∨ (∀v:Unit. ¬("trigger" after e')))
. . 26. e : E
```

```
. . 27. loc(e) = i
. . 28. kind(e) = k
. . 29. P((x when e),val(e))
. . 30. kind(e) = locl(a) ⇒ ("trigger" when e)
. . 31. e' : E
. . 32. (e <loc e') ∨ e = e'
. . 33. ∀v:Unit. ¬("trigger" after e')
. . ⊢  ∃e':E. loc(e') = i & kind(e') = locl(a)  by InstHyp [·] -1
. . |
. . 1. T : Type
. . 2. A : Type
. . 3. P : A→T→𝔹
. . 4. i : Id
. . 5. k : Knd
. . 6. a : Id
. . 7. x : Id
. . 8. A
. . 9. T
. . 10. ¬x = "trigger"
. . 11. ¬locl(a) = k
. . 12. (λloc.⊕(trigger1(loc;T;A;P;i;k;a;x))) ∈ Dsys
. . 13. D' : Dsys
. . 14. λloc.⊕(trigger1(loc;T;A;P;i;k;a;x)) ⊆ D'
. . 15. w : World
. . 16. p : FairFifo
. . 17. PossibleWorld(D';w)
. . 18. ∀e:E. loc(e) = i ⇒ kind(e) = k ⇒ (valtype(e) ⊆r T)
. . 19. vartype(i;x) ⊆r A
. . 20. ∀e:E. loc(e) = i ⇒ first(e) ⇒ ("trigger" when e) = false₂
. . 21. ∀e':E.
. . |   loc(e') = i
. . |   ⇒
. . |   (("trigger" after e')
. . |    ⟺
. . |    (∃e:E.
. . |     (e <loc e') ∨ e = e' & kind(e) = k & P((x when e),val(e))))
. . 22. @i: ma-single-pre1("trigger";𝔹;a;Unit;x,v.x) ∈ Dsys
. . 23. vartype(i;"trigger") ⊆r 𝔹
. . 24. ∀e:E. loc(e) = i ⇒ kind(e) = locl(a) ⇒ (valtype(e) ⊆r Unit)
. . 25. ∀e:E.
. . |   loc(e) = i
. . |   ⇒
. . |   (kind(e) = locl(a) ⇒ ("trigger" when e))
. . |   & (∃e':E.
. . |      (e <loc e') ∨ e = e'
. . |      & kind(e') = locl(a) ∨ (∀v:Unit. ¬("trigger" after e')))
. . 26. e : E
. . 27. loc(e) = i
. . 28. kind(e) = k
. . 29. P((x when e),val(e))
. . 30. kind(e) = locl(a) ⇒ ("trigger" when e)
. . 31. e' : E
. . 32. (e <loc e') ∨ e = e'
. . 33. ∀v:Unit. ¬("trigger" after e')
. . 34. ¬("trigger" after e')
```

```
. . ⊢  ∃e':E. loc(e') = i & kind(e') = locl(a)  by Analyze -1
. . |
. . 1. T : Type
. . 2. A : Type
. . 3. P : A→T→𝔹
. . 4. i : Id
. . 5. k : Knd
. . 6. a : Id
. . 7. x : Id
. . 8. A
. . 9. T
. . 10. ¬x = "trigger"
. . 11. ¬locl(a) = k
. . 12. (λloc.⊕(trigger1(loc;T;A;P;i;k;a;x))) ∈ Dsys
. . 13. D' : Dsys
. . 14. λloc.⊕(trigger1(loc;T;A;P;i;k;a;x)) ⊆ D'
. . 15. w : World
. . 16. p : FairFifo
. . 17. PossibleWorld(D';w)
. . 18. ∀e:E. loc(e) = i ⇒ kind(e) = k ⇒ (valtype(e) ⊆r T)
. . 19. vartype(i;x) ⊆r A
. . 20. ∀e:E. loc(e) = i ⇒ first(e) ⇒ ("trigger" when e) = false₂
. . 21. ∀e':E.
. . |   loc(e') = i
. . |   ⇒
. . |   (("trigger" after e')
. . |    ⟺
. . |    (∃e:E.
. . |     (e <loc e') ∨ e = e' & kind(e) = k & P((x when e),val(e))))
. . 22. @i: ma-single-pre1("trigger";𝔹;a;Unit;x,v.x) ∈ Dsys
. . 23. vartype(i;"trigger") ⊆r 𝔹
. . 24. ∀e:E. loc(e) = i ⇒ kind(e) = locl(a) ⇒ (valtype(e) ⊆r Unit)
. . 25. ∀e:E.
. . |   loc(e) = i
. . |   ⇒
. . |   (kind(e) = locl(a) ⇒ ("trigger" when e))
. . |   & (∃e':E.
. . |      (e <loc e') ∨ e = e'
. . |      & kind(e') = locl(a) ∨ (∀v:Unit. ¬("trigger" after e')))
. . 26. e : E
. . 27. loc(e) = i
. . 28. kind(e) = k
. . 29. P((x when e),val(e))
. . 30. kind(e) = locl(a) ⇒ ("trigger" when e)
. . 31. e' : E
. . 32. (e <loc e') ∨ e = e'
. . 33. ∀v:Unit. ¬("trigger" after e')
. . ⊢  ("trigger" after e')  by RW (HypC 21) 0 THEN MaAuto
. . |
. . 1. T : Type
. . 2. A : Type
. . 3. P : A→T→𝔹
. . 4. i : Id
. . 5. k : Knd
. . 6. a : Id
```

```
. . 7. x : Id
. . 8. A
. . 9. T
. . 10. ¬x = "trigger"
. . 11. ¬locl(a) = k
. . 12. (λloc.⊕(trigger1(loc;T;A;P;i;k;a;x))) ∈ Dsys
. . 13. D' : Dsys
. . 14. λloc.⊕(trigger1(loc;T;A;P;i;k;a;x)) ⊆ D'
. . 15. w : World
. . 16. p : FairFifo
. . 17. PossibleWorld(D';w)
. . 18. ∀e:E. loc(e) = i ⇒ kind(e) = k ⇒ (valtype(e) ⊆r T)
. . 19. vartype(i;x) ⊆r A
. . 20. ∀e:E. loc(e) = i ⇒ first(e) ⇒ ("trigger" when e) = false₂
. . 21. ∀e':E.
. .     loc(e') = i
. .     ⇒
. .     (("trigger" after e')
. .      ⟺
. .      (∃e:E.
. .       (e <loc e') ∨ e = e' & kind(e) = k & P((x when e),val(e))))
. . 22. @i: ma-single-pre1("trigger";𝔹;a;Unit;x,v.x) ∈ Dsys
. . 23. vartype(i;"trigger") ⊆r 𝔹
. . 24. ∀e:E. loc(e) = i ⇒ kind(e) = locl(a) ⇒ (valtype(e) ⊆r Unit)
. . 25. ∀e:E.
. .     loc(e) = i
. .     ⇒
. .     (kind(e) = locl(a) ⇒ ("trigger" when e))
. .     & (∃e':E.
. .        (e <loc e') ∨ e = e'
. .        & kind(e') = locl(a) ∨ (∀v:Unit. ¬("trigger" after e')))
. . 26. e : E

. . 27. loc(e) = i
. . 28. kind(e) = k
. . 29. P((x when e),val(e))
. . 30. kind(e) = locl(a) ⇒ ("trigger" when e)
. . 31. e' : E
. . 32. (e <loc e') ∨ e = e'
. . 33. ∀v:Unit. ¬("trigger" after e')
. . ⊢ ∃e:E. (e <loc e') ∨ e = e' & kind(e) = k & P((x when e),val(e))
. .   by Unfold `and` 0 THEN Fold `cand` 0 THEN InstConcl [e]
. .      THEN
. .      MaAuto
END
```