```
switch_main_theorem  THM

⊢  ∀E:EventStruct, P:TraceProperty(E), A:Type, evt:(A→|E|)
|  , tg:(A→Label), tr_u:Trace(E), tr_l:A List.
|  switchable(E)(P) ⇒
|  No-dup-send(E)(tr_u) ⇒
|  full_switch_inv(E;A;evt;tg;tr_u;tr_l) ⇒
|  (∀m:Label. P(map(evt;<tr_l>_m))) ⇒ P(tr_u)
| by Unfolds ``trace_property str_trace`` 0 THEN Auto

|    THEN Unfold `full_switch_inv` (-2) THEN ExRepD

|    THEN InstLemma `switch_theorem` [E;P;A;evt;tg;tr_m] THEN Auto
|
|\
| .....antecedent .....
| 1. E : EventStruct
| 2. P : (|E| List)→Prop
| 3. A : Type
| 4. evt : A→|E|
| 5. tg : A→Label
| 6. tr_u : |E| List
| 7. tr_l : A List
| 8. switchable(E)(P)
| 9. No-dup-send(E)(tr_u)
| 10. tr_m : A List
| 11. tr_l R(tg) tr_m
| 12. map(evt;tr_m) layerR(E) tr_u
| 13. switch_inv(<A,evt,tg>(E))(tr_m)
| 14. ∀m:Label. P(map(evt;<tr_l>_m))
| ⊢  No-dup-send(E)(map(evt;tr_m))
|   by InstLemma `no_duplicate_send_layer` [E] THENA Auto
|
|      THEN Unfolds ``preserved_by rel_inverse`` (-1) THEN Reduce (-1)
|
| THEN EasyHyp
|
|\
| 1. E : EventStruct
| 2. P : (|E| List)→Prop
| 3. A : Type
| 4. evt : A→|E|
| 5. tg : A→Label
| 6. tr_u : |E| List
| 7. tr_l : A List
| 8. switchable(E)(P)
| 9. No-dup-send(E)(tr_u)
| 10. tr_m : A List
| 11. tr_l R(tg) tr_m
| 12. map(evt;tr_m) layerR(E) tr_u
| 13. switch_inv(<A,evt,tg>(E))(tr_m)
| 14. ∀m:Label. P(map(evt;<tr_l>_m))
| 15. m : Label
| ⊢  P(map(evt;<tr_m>_m))  by Subst <tr_m>_m = <tr_l>_m 0 THEN Auto
| |
```

```
| |\
| | .....equality .....
| | 1. E : EventStruct
| | 2. P : (|E| List)→Prop
| | 3. A : Type
| | 4. evt : A→|E|
| | 5. tg : A→Label
| | 6. tr_u : |E| List
| | 7. tr_l : A List
| | 8. switchable(E)(P)
| | 9. No-dup-send(E)(tr_u)
| | 10. tr_m : A List
| | 11. tr_l R(tg) tr_m
| | 12. map(evt;tr_m) layerR(E) tr_u
| | 13. switch_inv(<A,evt,tg>(E))(tr_m)
| | 14. ∀m:Label. P(map(evt;<tr_l>_m))
| | 15. m : Label
| | ⊢  <tr_m>_m = <tr_l>_m
| |   by Symmetry THEN BackThruLemma ‘tag_sublist_preserved‘ THEN Auto
| |
| |
| .\
| . 1. E : EventStruct
| . 2. P : (|E| List)→Prop
| . 3. A : Type
| . 4. evt : A→|E|
| . 5. tg : A→Label
| . 6. tr_u : |E| List
| . 7. tr_l : A List
| . 8. switchable(E)(P)
| . 9. No-dup-send(E)(tr_u)
| . 10. tr_m : A List
| . 11. tr_l R(tg) tr_m
| . 12. map(evt;tr_m) layerR(E) tr_u
| . 13. switch_inv(<A,evt,tg>(E))(tr_m)
| . 14. ∀m:Label. P(map(evt;<tr_l>_m))
| . 15. m : Label
| . ⊢  P(map(evt;<tr_l>_m))  by EasyHyp
|
.\
. 1. E : EventStruct
. 2. P : (|E| List)→Prop
. 3. A : Type
. 4. evt : A→|E|
. 5. tg : A→Label
. 6. tr_u : |E| List
. 7. tr_l : A List
. 8. switchable(E)(P)
. 9. No-dup-send(E)(tr_u)
. 10. tr_m : A List
. 11. tr_l R(tg) tr_m
. 12. map(evt;tr_m) layerR(E) tr_u
. 13. switch_inv(<A,evt,tg>(E))(tr_m)
. 14. ∀m:Label. P(map(evt;<tr_l>_m))
. 15. P(map(evt;tr_m))
```

```
. ⊢  P(tr_u)  by Assert layerR(E) preserves P
. |

. |\
. | .....assertion .....
. | 1. E : EventStruct
. | 2. P : (|E| List)→Prop
. | 3. A : Type
. | 4. evt : A→|E|
. | 5. tg : A→Label
. | 6. tr_u : |E| List
. | 7. tr_l : A List
. | 8. switchable(E)(P)
. | 9. No-dup-send(E)(tr_u)
. | 10. tr_m : A List
. | 11. tr_l R(tg) tr_m
. | 12. map(evt;tr_m) layerR(E) tr_u
. | 13. switch_inv(<A,evt,tg>(E))(tr_m)
. | 14. ∀m:Label. P(map(evt;<tr_l>_m))
. | 15. P(map(evt;tr_m))
. | ⊢  layerR(E) preserves P
. |   by ...
. | THEN AllHyps (\h. Unfold ‘b_switchable‘ h THEN Reduce h)
. |
. | THEN Repeat (BackThruLemma ‘preserved_by_or‘ THEN Auto)
. |
. .\
. . 1. E : EventStruct
. . 2. P : (|E| List)→Prop
. . 3. A : Type
. . 4. evt : A→|E|
. . 5. tg : A→Label
. . 6. tr_u : |E| List
. . 7. tr_l : A List
. . 8. switchable(E)(P)
. . 9. No-dup-send(E)(tr_u)
. . 10. tr_m : A List
. . 11. tr_l R(tg) tr_m
. . 12. map(evt;tr_m) layerR(E) tr_u
. . 13. switch_inv(<A,evt,tg>(E))(tr_m)
. . 14. ∀m:Label. P(map(evt;<tr_l>_m))
. . 15. P(map(evt;tr_m))
. . 16. layerR(E) preserves P
. . ⊢  P(tr_u)
. .   by Unfold ‘preserved_by‘ (-1) THEN BackThruHyp’ (-1) THEN Auto
END
```