```
switch_theorem  THM

⊢  ∀E:EventStruct, P:((|E| List)→Prop), A:Type, evt:(A→|E|)
|  , tg:(A→Label), tr:A List.
|  switchable(E)(P) ⇒
|  No-dup-send(E)(map(evt;tr)) ⇒
|  switch_inv(<A,evt,tg>(E))(tr) ⇒
|  (∀m:Label. P(map(evt;<tr>_m))) ⇒ P(map(evt;tr))
| by Auto THEN Reduce 0 THEN Auto
THEN AssertBY
|    switchable(<A,evt,tg>(E))(P o evt)

|        (BackThruLemma ‘switchable_induced_tagged‘ THEN Auto)

|    THEN InstLemma ‘switch_inv_theorem2‘ [<A,evt,tg>(E);P o evt]

|    THENA (Auto THEN Try (Unfold ‘trace_property‘ 0 THEN Auto))

|    THEN AllHyps (\h. NthHypSq h THEN EqCD THEN Try Trivial

|                     THEN Unfold ‘induced_tagged_event_str‘ 0

|                     THEN Reduce 0 THEN Trivial)
|
1. E : EventStruct
2. P : (|E| List)→Prop
3. A : Type
4. evt : A→|E|
5. tg : A→Label
6. tr : A List
7. switchable(E)(P)
8. No-dup-send(E)(map(evt;tr))
9. switch_inv(<A,evt,tg>(E))(tr)
10. ∀m:Label. P(map(evt;<tr>_m))
11. switchable(<A,evt,tg>(E))(P o evt)
12. (switch_inv(<A,evt,tg>(E))
|   ∧ No-dup-send(<A,evt,tg>(E))) fuses P o evt
⊢  P(map(evt;tr))
| by ...
THEN BackThruHyp (-1) THEN Auto

|    THENA (Unfold ‘str_trace‘ 0 THEN Auto)

|    THEN Unfold ‘prop_and‘ 0 THEN Reduce 0 THEN Auto
|
1. E : EventStruct
2. P : (|E| List)→Prop
3. A : Type
4. evt : A→|E|
5. tg : A→Label
6. tr : A List
7. switchable(E)(P)
8. No-dup-send(E)(map(evt;tr))
9. switch_inv(<A,evt,tg>(E))(tr)
10. ∀m:Label. P(map(evt;<tr>_m))
```

```
11. switchable(<A,evt,tg>(E))(P o evt)
12. ∀tr:Trace(<A,evt,tg>(E)).
    (∀m:Label. P(map(evt;<tr>_m))) ⇒
    (switch_inv(<A,evt,tg>(E)) ∧ No-dup-send(<A,evt,tg>(E)))(tr) ⇒
    P(map(evt;tr))
⊢  No-dup-send(<A,evt,tg>(E))(tr)
  by BackThruLemma' `no_dup_send_induced` THEN Auto
END
```