**theory** *TupleOrd* = *Main*:

**instance** $*$ :: (*ord*, *ord*) *ord* **..**

**defs**
*leq-pair-def*: $p <= q == fst\ p < fst\ q \vee fst\ p = fst\ q \wedge snd\ p <= snd\ q$
*le-pair-def*: $p < q == (p::'a::ord*'b::ord) <= q \wedge p \neq q$

**lemma** *leq-pairI1*: $a < a' \Longrightarrow (a, b) <= (a', b')$
**by** (*simp add*: *leq-pair-def*)

**lemma** *leq-pairI2*: $b <= b' \Longrightarrow (a, b) <= (a, b')$
**by** (*simp add*: *leq-pair-def*)

**lemma** *leq-pairE*: $(a, b) <= (a', b') \Longrightarrow (a < a' \Longrightarrow P) \Longrightarrow (a = a' \Longrightarrow b <= b' \Longrightarrow P) \Longrightarrow P$
**apply** (*unfold leq-pair-def*)
**apply** (*erule disjE*)
**apply** *simp*+
**done**

**instance** $*$ :: (*order*, *order*) *order*
**apply** *intro-classes*
**apply** (*simp-all add*: *split-paired-all*)
**apply** (*rule leq-pairI2*)
**apply** (*rule order-refl*)
**apply** (*erule leq-pairE*)+
**apply** (*rule leq-pairI1*)
**apply** (*erule order-less-trans*)
**apply** *assumption*
**apply** (*rule leq-pairI1*)
**apply** (*erule order-less-le-trans*)
**apply** (*erule order-eq-refl*)
**apply** (*erule leq-pairE*)
**apply** *simp*
**apply** (*erule leq-pairI1*)
**apply** *simp*
**apply** (*rule leq-pairI2*)
**apply** (*erule order-trans*)
**apply** *assumption*
**apply** (*erule leq-pairE*)+
**apply** (*drule order-less-not-sym*)
**apply** *simp*
**apply** *simp*
**apply** (*erule leq-pairE*)
**apply** *simp*
**apply** (*drule order-antisym*)
**apply** *assumption*
**apply** *simp*

**apply** (*simp add*: *le-pair-def*)
**done**

**instance** $*$ ::(*linorder*, *linorder*) *linorder*
**apply** *intro-classes*
**apply** (*simp add*:*split-paired-all*)
**apply** (*case-tac* (*a*,*b*) <= (*aa*,*ba*))
**apply** *simp*
**apply** *simp*
**apply** (*simp only*: *leq-pair-def fst-conv snd-conv*)
**apply** *simp*
**apply** (*erule conjE*)
**apply** (*simp only*: *linorder-not-less*)
**apply** (*case-tac aa* < *a*)
**apply** (*rule disjI1*)
**apply** *assumption*
**apply** (*simp only*: *linorder-not-less*)
**apply** (*drule order-antisym*)
**apply** *assumption*
**apply** *simp*
**apply** (*simp only*: *linorder-not-le*)
**apply** (*rule order-less-imp-le*)
**apply** *assumption*
**done**

**end**