Lemma 1. *Child-state is at x due to parent receiving x-resp and child is at y due to child receiving y-resp $\Rightarrow$ x = y.*

Proof. Proof by contradiction, assuming x $\neq$ y.

Child has sent x-resp before receiving y-resp ($\because$ y-resp is the last message causing child transition).

Parent has sent y-resp before receiving x-resp ($\because$ x-resp is the last message causing child-state transition).

$\Rightarrow$ The corresponding y-req should have been sent by the child before sending x-resp ($\because$ FIFO ordering between child to parent req, resp).

$\Rightarrow$ x-resp is not voluntary.

$\Rightarrow$ x-req is sent by parent before sending y-resp ($\because$ x-resp is sent by child, and hence x-req is received by child before receving y-resp and FIFO order ordering between parent to child messages).

But y-resp will not be sent by parent if there is a pending request.

$\Rightarrow\Leftarrow$ $\square$

Lemma 2. *Child-state is at x due to parent sending x-resp and child is at y due to child sending y-resp, corresponding x-resp was sent by parent before receiving y-resp and y-resp was sent by child before receiving x-resp $\Rightarrow$ x = y.*

Proof. Proof by contradiction, assuming x $\neq$ y.

x-resp has not reached child ($\because$ y-resp is the last message sent by child causing child transition, and child is still in y).

y-resp has not reached parent ($\because$ x-resp is the last message sent by parent causing child-state transition, and child-state is still in x).

Corresponding x-req was received by parent before receiving y-resp.

$\Rightarrow$ x-req was sent by child before sending y-resp ($\because$ FIFO ordering between child to parent req, resp).

$\Rightarrow$ y-resp is not voluntary.

$\Rightarrow$ Corresponding y-req was sent by parent before sending x-resp ($\because$ FIFO ordering between parent to child messages).

$\Rightarrow$ x-resp was sent by parent before receiving y-resp for y-req sent by parent which is not possible ($\because$ parent does not respond when there is a pending request).

$\Rightarrow\Leftarrow$

$\square$

Lemma 3. *Child-state is at x due to parent sending x-resp and child is at y due to child receiving y-resp $\Rightarrow$ y $\leq$ x.*

Proof. Proof by contradiction, assuming x < y.

Parent sends x-resp only after y-resp ($\because$ x-resp is the last message causing child-state transition).

Child can send corresponding x-req for the x-resp from parent only after child receives y-resp ($\because$ only one pending request is allowed). But y-resp is the last message causing child transition, so child is still at y.

$\Rightarrow$ x-req can not be sent by child since x < y.

$\Rightarrow\Leftarrow$

$\square$

Lemma 4. *Child-state is at x due to parent receiving x-resp and child is at y due to child sending y-resp* $\Rightarrow y \leq x$.

Proof. Proof by contradiction, assuming x < y.

Child sends y-resp only after x-resp ($\because$ y-resp is the last message causing child transition).

Child must be in z > y to send y-resp.

$\Rightarrow$ x < z ($\because$ x < y).

$\Rightarrow$ Child must have sent z-req after sending x-resp and transitioning to x, and eventually got back z-resp.

$\Rightarrow$ Parent must have received z-req and sent z-resp, while child-state transitions to z. But this is not possible as x-resp was the last message causing child state transition.

$\Rightarrow\Leftarrow$

$\square$

Lemma 5. *Child-state is at x due to parent sending x-resp and child is at y due to child sending y-resp* $\Rightarrow y \leq x$.

Proof. Proof by contradiction, assuming x < y.

1. Parent sends x-resp before receiving y-resp and x-resp reaches child after child sends y-resp - Not possible by Lemma 2.

2. Parent sends x-resp before receiving y-resp and x-resp reaches child before child sends y-resp.

   Child must be in z > y to send y-resp.

   $\Rightarrow$ x < z ($\because$ x < y and y < z)

   $\Rightarrow$ After child received x-resp, somehow it must reach higher state z. This is possible only if parent sends z-resp, thereby child-state transitioning to z. But x-resp was the last message causing child-state transition.

   $\Rightarrow\Leftarrow$

3. Parent receives y-resp before parent sends x-resp.

   (a) Child sends x-req corresponding to x-resp after sending y-resp.
   Child is at y > x as y-resp is the last message causing transition for child. Since x < y, child wont send x-req.
   $\Rightarrow\Leftarrow$

   (b) Child sends x-req corresponding to x-resp before sending y-resp.
   Child is at z < x when x-req is sent and z' > y when y-resp is sent.
   $\Rightarrow$ z < z' ($\because$ x < y and z' > y).
   $\Rightarrow$ z'-req must be sent by child before x-resp is received (for x-req).
   But this is not possible as only one pending request is allowed.
   $\Rightarrow\Leftarrow$

$\square$

Lemma 6. *Child-state is at x due to parent sending x-resp and child is at y due to child receiving y-resp and parent just received z-req $\Rightarrow$ $y = x$.*

Proof. Proof by contradiction, assuming x $\neq$ y.

Parent sends x-resp only after y-resp ($\because$ x-resp is the last message causing child-state transition).

Child sends corresponding x-req only after receiving y-resp ($\because$ only one pending request is allowed).

$\Rightarrow$ Child can not send z-req before receiving x-resp ($\because$ only one pending request is allowed and x-resp has not reached child yet).

$\Rightarrow\Leftarrow$

$\square$

Lemma 7. *Child-state is at x due to parent receiving x-resp and child is at y due to child sending y-resp and parent just received z-req $\Rightarrow$ $y = x$.*

Proof. Proof by contradiction, assuming x $\neq$ y.

Child sends y-resp only after sending x-resp ($\because$ y-resp is the last message causing child transition).

Parent receives z-req before receiving y-resp ($\because$ x-resp is the last message causing child-state transition).

$\Rightarrow$ Child sent z-req before sending y-resp ($\because$ FIFO ordering between child to parent req, resp).

$\Rightarrow$ y-resp is not voluntary ($\because$ child has pending z-req).

$\Rightarrow$ Parent does not receive z-req till it receives y-resp for the corresponding y-req that parent sent.

$\Rightarrow\Leftarrow$

$\square$

Lemma 8. *Child-state is at x due to parent sending x-resp and child is at y due to child sending y-resp and parent just received z-req* $\Rightarrow$ $y = x$.

Proof. Proof by contradiction, assuming x $\neq$ y.

1. Parent sends x-resp before receiving y-resp and x-resp reaches child after child sends y-resp - Not possible because of Lemma 2.

2. Parent sends x-resp before receiving y-resp and x-resp reaches child before child sends y-resp.

   y-resp can not be sent voluntarily by child as z-req is pending.

   $\Rightarrow$ There is a pending y-req from parent.

   $\Rightarrow$ Parent wont receive z-req ($\because$ only one pending request is allowed)

   $\Rightarrow\Leftarrow$

3. Parent receives y-resp before parent sends x-resp.

   $\Rightarrow$ There is a pending x-req from child. This is not possible since there are two pending requests from child as child has not received x-resp at this moment.

   $\Rightarrow\Leftarrow$

$\square$

Lemma 9. *Child-state is at x due to parent sending x-resp and child was at y before sending z-resp due to child receiving y-resp and parent just received z-resp* $\Rightarrow$ $y = x$.

Proof. Proof by contradiction, assuming x $\neq$ y.

Parent sends x-resp only after y-resp ($\because$ x-resp is the last message causing child-state transition).

Child sends corresponding x-req only after receiving y-resp ($\because$ only one pending request is allowed).

$\Rightarrow$ z-resp is not voluntary.

$\Rightarrow$ Corresponding z-req must be received by child before receiving x-resp ($\because$ x-resp has not yet been received by child when it sent z-resp).

$\Rightarrow$ x-resp is sent by parent between sending z-req ($\because$ FIFO ordering between parent to child messages) and receiving z-resp, which is not possible.

$\Rightarrow\Leftarrow$

$\square$

Lemma 10. *Child-state is at x due to parent receiving x-resp and child is at y before sending z-resp due to child sending y-resp and parent just received z-resp* $\Rightarrow$ $y = x$.

Proof. Parent must receive y-resp before receiving z-resp ($\because$ FIFO ordering between child to parent resps). This means that x-resp is not the last message inducing child-state transition.

$\Rightarrow\Leftarrow$

$\square$

Lemma 11. *Child-state is at x due to parent sending x-resp and child is at y before sending z-resp due to child sending y-resp and parent just received z-resp* $\Rightarrow$ $y = x$.

Proof. Proof by contradiction, assuming x $\neq$ y.

1. Parent sends x-resp before receiving y-resp.

   z-resp reaches parent.

   $\Rightarrow$ y-resp reaches parent ($\because$ FIFO ordering between child to parent resps), after x-resp is sent which is not possible.

   $\Rightarrow\Leftarrow$

2. Parent receives y-resp before parent sends x-resp.

   There is a corresponding x-req that parent receives when sending x-resp. Parent receives x-req before z-resp is received.

   $\Rightarrow$ x-req was sent by child before z-resp was sent ($\because$ FIFO ordering between child to parent req, resp).

   $\Rightarrow$ z-resp is not voluntary.

   $\Rightarrow$ Corresponding z-req was received by child before x-resp was received ($\because$ x-resp was not yet received).

   $\Rightarrow$ x-resp was sent by parent between sending z-req ($\because$ FIFO ordering between parent to child messages) and z-resp. This is not possible.

   $\Rightarrow\Leftarrow$

$\square$

Lemma 12. *At a moment, child-state is x and child is y* $\Rightarrow$ $y \leq x$.

Lemma 13. *When parent receives z-req, child-state is x and child is y* $\Rightarrow$ $y = x$.

Lemma 14. *When parent receives z-resp, child-state is x and child is y* $\Rightarrow$ $y = x$.