We define \((a)\) translating to ordinary logic.

Now the network is a 1 on choose Formally define \(\text{choose} \equiv (a? \cdot c! \cdot 0 \cdot d! \cdot 0) \lor (b? \cdot c! \cdot 1 \cdot d! \cdot 1)\)

Now the network is

\[
\text{choose} \equiv (a? \cdot c! \cdot 0 \cdot d! \cdot 0) \lor (b? \cdot c! \cdot 1 \cdot d! \cdot 1)
\]

§ We define \(\text{choose}\) as follows:

\[
\text{choose} \equiv (a? \cdot c! \cdot 0 \cdot d! \cdot 0) \lor (b? \cdot c! \cdot 1 \cdot d! \cdot 1)
\]

The formal description of this network is

\[
\text{chan } a, b, c \cdot a! \cdot 0 \parallel \text{choose} \parallel (c? \cdot b! \cdot c)
\]

The following picture shows a network of communicating processes.

![Diagram of a network of communicating processes]

The formal description of this network is

\[
\text{chan } a, b, c \cdot a! \cdot 0 \parallel \text{choose} \parallel (c? \cdot b! \cdot c)
\]

Formally define \(\text{choose}\) as follows:

\[
\text{choose} \equiv (a? \cdot c! \cdot 0 \cdot d! \cdot 0) \lor (b? \cdot c! \cdot 1 \cdot d! \cdot 1)
\]

Now the network is

\[
\begin{align*}
\text{chan } a, b, c \cdot a! \cdot 0 & \parallel \text{choose} \parallel (c? \cdot b! \cdot c) \\
\equiv & \exists \mathcal{M}a, \mathcal{M}b, ra, ra', wa, wa', \mathcal{M}b, \mathcal{T}b, rb, rb', wb, wb', \mathcal{M}c, \mathcal{T}c, rc, rc', wc, wc'.
\end{align*}
\]

\[
\begin{align*}
\text{E}a, tc, tb & \cdot \\
& \quad ta = \mathcal{T}a_0 = 0 \land \mathcal{M}a_0 = 0 \land wa' = 1 & \\
& \land (tc = \mathcal{T}c_0 = \mathcal{T}d_{wd} = \mathcal{T}a_0 + 1 \land ra' = wc' = 1 \land \mathcal{M}c_0 = \mathcal{M}d_{wd} = 0 \land wd' = wd + 1) & \\
& \lor (tc = \mathcal{T}c_0 = \mathcal{T}d_{wd} = \mathcal{T}b_0 + 1 \land rb' = wc' = 1 \land \mathcal{M}c_0 = \mathcal{M}d_{wd} = 0 \land wd' = wd + 1) & \\
& \land (tb = \mathcal{T}b_0 = \mathcal{T}c_0 + 1 \land rc' = wb' = 1 \land \mathcal{M}b_0 = \mathcal{M}c_0) & \\
& \land t' = \text{MAX} \{ta; tc; tb\}.
\end{align*}
\]

\[
\begin{align*}
\text{E}a, tc, tb & \cdot \\
& \quad (tc = \mathcal{T}d_{wd} = t + 1 \land \mathcal{M}d_{wd} = 0 \land wd' = wd + 1) & \\
& \lor (tc = \mathcal{T}d_{wd} = t + 1 \land \mathcal{M}d_{wd} = 1 \land wd' = wd + 1) & \\
& \land tb = tc + 1 & \\
& \land t' = \text{MAX} \{t; tc; tb\}.
\end{align*}
\]

\[
\begin{align*}
\text{E}a, tc, tb & \cdot \\
& \quad tc = \mathcal{T}d_{wd} = t + 1 \land \mathcal{M}d_{wd} = 0 \land wd' = wd + 1 \land tb = tc + 1 \land t' = \text{MAX} \{t; tc; tb\} & \\
& \lor tc = \mathcal{T}d_{wd} = t + 1 \land \mathcal{M}d_{wd} = 1 \land wd' = wd + 1 \land tb = tc + 1 \land t' = \text{MAX} \{t; tc; tb\} & \\
& \text{now we can eliminate } tc \text{ and } tb \text{ in each disjunct separately} & \\
& \text{E}a, tc, tb & \cdot \\
& \quad tc = \mathcal{T}d_{wd} = t + 1 \land \mathcal{M}d_{wd} = 0 \land wd' = wd + 1 \land t' = t + 2 & \\
& \lor tc = \mathcal{T}d_{wd} = \infty \land \mathcal{M}d_{wd} = 1 \land wd' = wd + 1 \land t' = \infty & \\
& \text{E}a, tc, tb & \cdot \\
& \quad t = t + 1 \land d! \cdot 0 \land tc = t + 1 \land t' = \infty.
\end{align*}
\]

There is probably a better way to do this question by using laws of programs and not translating to ordinary logic.
(b) As in part (a), choose either reads from $a$ and outputs a 0 on $c$ and $d$, or reads from $b$ and outputs a 1 on $c$ and $d$. But this time the choice is not made freely; choose reads from the channel whose input is available first (if there's a tie, then take either one).

§ We define choose as follows:

$$
\text{choose } = (\sqrt{a} \lor \mathcal{T}_{a_r} \leq \mathcal{T}_{b_r}) \land (a = 0 \lor d) \land \\
(\sqrt{b} \lor \mathcal{T}_{b_r} \leq \mathcal{T}_{a_r}) \land (b = 1 \lor d)
$$

Now the network is

$$
\begin{align*}
\text{chan } a, b, c & \mid \text{ choose } \mid (c, b, c) \\
= & \exists \mathcal{M}_a, \mathcal{T}_a, ra, ra', wa, wa', \mathcal{M}_b, \mathcal{T}_b, rb, rb', wb, wb', \mathcal{M}_c, \mathcal{T}_c, rc, rc', wc, wc'. \\
ra := 0. & \text{ wa := 0. } \text{ rb := 0. } \text{ wb := 0. } \text{ rc := 0. } \text{ wc := 0.} \\
& \mathcal{M}_a_{wa} = 0 \land \mathcal{T}_a_{wa} = t \land (wa := wa + 1) \land \\
= & \begin{cases}
\mathcal{T}_a_{ra} \leq t & \lor \mathcal{T}_a_{ra} \leq \mathcal{T}_b_{rb}, \\
& \land (t := \max t (\mathcal{T}_a_{ra} + 1). \text{ ra := ra + 1.} \\
& \mathcal{M}_c_{wc} = 0 \land \mathcal{T}_c_{wc} = t \land (wc := wc + 1), \\
& \mathcal{M}_d_{wd} = 0 \land \mathcal{T}_d_{wd} = t \land (wd := wd + 1) ) ) \\
\lor & \begin{cases}
\mathcal{T}_b_{rb} \leq t & \lor \mathcal{T}_b_{rb} \leq \mathcal{T}_a_{ra}, \\
& \land (t := \max t (\mathcal{T}_b_{rb} + 1). \text{ rb := rb + 1.} \\
& \mathcal{M}_c_{wc} = 1 \land \mathcal{T}_c_{wc} = t \land (wc := wc + 1), \\
& \mathcal{M}_d_{wd} = 1 \land \mathcal{T}_d_{wd} = t \land (wd := wd + 1) ) ) \\
\end{cases}
\end{cases}
\end{align*}
$$

Except for time, the three processes in the independent composition change different variables, so it is easily replaced by a conjunction.

Also, make all substitutions indicated by assignments.

$$
\begin{align*}
= & \exists \mathcal{M}_a, \mathcal{T}_a, ra, ra', wa, wa', \mathcal{M}_b, \mathcal{T}_b, rb, rb', wb, wb', \mathcal{M}_c, \mathcal{T}_c, rc, rc', wc, wc'. \\
& \exists a, tc, tb: \\
ta = \mathcal{T}_a_{ta} = t \land \mathcal{M}_a_{ta} = 0 \land wa' = 1 \\
\land & \begin{cases}
\mathcal{T}_a_{ta} \leq t & \lor \mathcal{T}_a_{ta} \leq \mathcal{T}_b_{tb}, \\
& \land tc = \mathcal{T}_c_{tc} = \mathcal{T}_d_{wd} = t + 1 \land ra' = wc' = 1 \land \mathcal{M}_c_{tc} = \mathcal{M}_d_{wd} = 0 \land wd' = wd + 1, \\
\lor & \begin{cases}
\mathcal{T}_b_{tb} \leq t & \lor \mathcal{T}_b_{tb} \leq \mathcal{T}_a_{ta}, \\
& \land tc = \mathcal{T}_c_{tb} = \mathcal{T}_d_{wd} = \mathcal{T}_b_{tb} + 1 \land rb' = wc' = 1 \land \mathcal{M}_c_{tb} = \mathcal{M}_d_{wd} = 1 \land wd' = wd + 1, \\
\end{cases}
\end{cases}
\end{align*}
$$

use the One-Point laws to eliminate most quantifiers

$$
\begin{align*}
= & \exists tc, tb: \\
& \begin{cases}
(t \leq t & \lor t \leq tb) \land tc = \mathcal{T}_d_{wd} = t + 1 \land \mathcal{M}_d_{wd} = 0 \land wd' = wd + 1, \\
\lor & \begin{cases}
tb \leq t & \lor tb \leq t, \\
& \land tc = \mathcal{T}_d_{wd} = tb + 1 \land \mathcal{M}_d_{wd} = 1 \land wd' = wd + 1, 
\end{cases}
\end{cases}
\land \\
& tb = tc + 1 \\
\land \\
& t' = \text{MAX } [t; tc; tb] \text{ simplify the two minor disjunctions and move the conjuctions into the major disjunction}
\end{align*}
$$

$$
\begin{align*}
= & \exists tc, tb: \\
& \begin{cases}
tc = \mathcal{T}_d_{wd} = t + 1 \land \mathcal{M}_d_{wd} = 0 \land wd' = wd + 1 \land tb = tc + 1 \land t' = \text{MAX } [t; tc; tb] \\
\lor & \begin{cases}
tb \leq t & \lor tb \leq t, \\
& \land tc = \mathcal{T}_d_{wd} = tb + 1 \land \mathcal{M}_d_{wd} = 1 \land wd' = wd + 1 \land tb = tc + 1 \land t' = \text{MAX } [t; tc; tb]
\end{cases}
\end{cases}
\text{ now we can eliminate } tc \text{ and } tb \text{ in each disjunct separately}
\end{align*}
$$

$$
\begin{align*}
= & Td_{wd} = t + 1 \land \mathcal{M}_d_{wd} = 0 \land wd' = wd + 1 \land t' = t + 2 \\
\lor & \begin{cases}
\mathcal{T}_d_{wd} = \infty \land \mathcal{M}_d_{wd} = 1 \land wd' = wd + 1 \land t' = \infty
\end{cases}
\end{align*}
$$

If the computation starts before time $\infty$ the output is definitely 0. Again, there is probably a better way to do this question by using laws of programs and not translating to ordinary logic.