Notations
We use $\overline{::}_2$ to push a continuation on the continuation stack of the topmost fiber, without modifyin the effect handler:

$$\kappa \mathbin{\overline{::}_2} (\langle \kappa s, \eta\rangle \mathbin{\overline{::}} \mu s) = \langle \kappa \mathbin{\overline{::}} \kappa s,\ \eta\rangle \mathbin{\overline{::}} \mu s$$

Computations

$$\begin{aligned}
[\![\mathrm{V}\ \mathrm{W}]\!] &= \overline{\lambda}\kappa s.\ [\![\mathrm{V}]\!] \mathbin{\underline{@}} [\![\mathrm{W}]\!] \mathbin{\underline{@}} \downarrow\kappa s \\
[\![\mathbf{absurd}\ \mathrm{V}]\!] &= \overline{\lambda}\kappa s.\ \mathbf{absurd}\ [\![\mathrm{V}]\!] \\
[\![\mathbf{return}\ \mathrm{V}]\!] &= \overline{\lambda}\langle \kappa \mathbin{\overline{::}} \kappa s,\ \eta\rangle \mathbin{\overline{::}} \mu s.\ \kappa \mathbin{\underline{@}} [\![\mathrm{V}]\!] \mathbin{\underline{@}} \downarrow(\langle \kappa s, \eta\rangle \mathbin{\overline{::}} \mu s) \\
[\![\mathbf{let}\ x \leftarrow \mathrm{M}\ \mathbf{in}\ \mathrm{N}]\!] &= \overline{\lambda}\langle \kappa \mathbin{\overline{::}} \kappa s,\ \eta\rangle \mathbin{\overline{::}} \mu s. \\
&\qquad [\![\mathrm{M}]\!] \mathbin{\overline{@}} \Big(\langle (\underline{\lambda} x\ \mathrm{ks}.\ [\![\mathrm{N}]\!] \mathbin{\overline{@}} (\kappa \mathbin{\overline{::}_2} \uparrow\mathrm{ks})) \mathbin{\overline{::}} \kappa s,\ \eta\rangle \mathbin{\overline{::}} \mu s\Big) \\
[\![\mathbf{perform}\ \mathrm{V}]\!] &= \overline{\lambda}\langle \kappa \mathbin{\overline{::}} \kappa s,\ \eta\rangle \mathbin{\overline{::}} \mu s.\ \eta \mathbin{\underline{@}} [\![\mathrm{V}]\!] \mathbin{\underline{@}} [\langle \downarrow\kappa s, \eta\rangle] \mathbin{\underline{@}} \downarrow\mu s \\
[\![\mathbf{handle}\ \mathrm{M}\ \mathbf{with}\ \{\mathrm{H}_v, \mathrm{H}_x, \mathrm{H}_f\}]\!] &= \overline{\lambda}\mu s.\ [\![\mathrm{M}]\!] \mathbin{\overline{@}} (\langle [\![\mathrm{H}_v]\!] \mathbin{\overline{::}} [\![\mathrm{H}_x]\!] \mathbin{\overline{::}} [],\ [\![\mathrm{H}_f]\!]\rangle \mathbin{\overline{::}} \mu s)
\end{aligned}$$

where

$$\begin{aligned}
\mathrm{H}_v &: & [\![\mathbf{return}\ x \mapsto \mathrm{N}]\!] &= \underline{\lambda} x\ \mathrm{ks}.\ \underline{\mathbf{let}}\ (\mathrm{kf} \mathbin{\underline{::}} \mathrm{ks}') = \mathrm{ks}\ \underline{\mathbf{in}} \\
&& &\qquad [\![\mathrm{N}]\!] \mathbin{\overline{@}} \uparrow\mathrm{ks}' \\
\mathrm{H}_f &: & [\![p\ r \mapsto \mathrm{N}]\!] &= \underline{\lambda} p\ s\ \mathrm{ks}.\ \underline{\mathbf{let}}\ r = \mathbf{fun}\ s\ \underline{\mathbf{in}} \\
&& &\qquad [\![\mathrm{N}_\ell]\!] \mathbin{\overline{@}} \uparrow\mathrm{ks} \\
\mathrm{H}_x &: & [\![e \mapsto \mathrm{N}]\!] &= \underline{\lambda} e\ \mathrm{ks}.\ [\![\mathrm{N}]\!] \mathbin{\overline{@}} \uparrow\mathrm{ks}
\end{aligned}$$

$$\begin{aligned}
[\![\mathbf{raise}\ \mathrm{V}]\!] &= \overline{\lambda}\langle \kappa \mathbin{\overline{::}} \chi \mathbin{\overline{::}} \kappa s\rangle \mathbin{\overline{::}} \mu s.\ \chi \mathbin{\underline{@}} [\![\mathrm{V}]\!] \mathbin{\underline{@}} (\langle \downarrow\kappa s, \eta\rangle \mathbin{\underline{::}} \mu s) \\
[\![\mathbf{try}\ \mathrm{M}\ \mathbf{with}\ \mathrm{H}]\!] &= \overline{\lambda}\langle \kappa s, \eta\rangle \mathbin{\overline{::}} \mu s.\ [\![\mathrm{M}]\!] \mathbin{\overline{@}} (\langle \mathrm{K}_{\mathrm{ret}} \mathbin{\overline{::}} [\![\mathrm{H}]\!] \mathbin{\overline{::}} \kappa s,\ \eta\rangle \mathbin{\overline{::}} \mu s)
\end{aligned}$$

where

$$\begin{aligned}
\mathrm{K}_{\mathrm{ret}} &= \underline{\lambda} x\ \mathrm{ks}.\ \underline{\mathbf{let}}\ \langle \mathrm{kx} \mathbin{\underline{::}} k \mathbin{\underline{::}} \mathrm{ks}',\ \eta\rangle \mathbin{\underline{::}} \mathrm{ms} = \mathrm{ks}\ \underline{\mathbf{in}} \\
&\qquad k \mathbin{\underline{@}} x \mathbin{\underline{@}} (\langle \mathrm{ks}', \eta\rangle \mathbin{\underline{::}} \mathrm{ms}) \\
[\![e \mapsto \mathrm{N}]\!] &= \underline{\lambda} e\ \mathrm{ks}.\ [\![\mathrm{N}]\!] \mathbin{\overline{@}} \uparrow\mathrm{ks}
\end{aligned}$$

Note: In the setting of js_of_ocaml, $\mathrm{K}_{\mathrm{ret}}$ is not strictly necessary, as the source language has explicit `POPTRAP` statements: we could either push a dummy continuation and have `POPTRAP` pop it (but we need to push something for the stack to look as expected to the rest of the translation), or push $\mathrm{K}_{\mathrm{ret}}$ and have `POPTRAP` simply return through it. Currently, I'm doing both, but it is superfluous.

Top level program

$$\top[\![\mathrm{M}]\!] = [\![\mathrm{M}]\!] \,\overline{@}\, \big(\langle(\underline{\lambda} x \text{ ks. } x) \mathbin{\overline{::}} (\underline{\lambda} z \text{ ks. } \mathbf{absurd}\ z) \mathbin{\overline{::}} \uparrow[]\rangle,\ (\underline{\lambda} z \text{ ks. } \mathbf{absurd}\ z) \mathbin{\overline{::}} \uparrow[]\big)$$