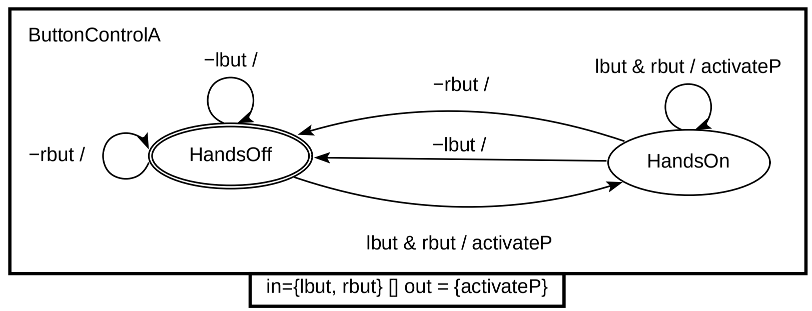
Monotonicity concerns with μ-Charts

- μ-Charts — a structured language for the specifications of reactive systems
- Presentation based on ongoing investigation of the derivation of Z-based semantics and refinement for μ-Charts
  - Introduce charts using the two-handed press example
    - Chart semantics — trace and Z
    - Semantics of composition
  - Refinement
  - Monotonicity results for chart composition

The Two-handed Press Problem

- Problem: to specify the interaction with a metal-working press that ensures the safety of the operator
- Requires: two buttons sufficiently separated so that the operator can only activate the press when her hands are both on the control buttons and therefore not in the press itself

Two-handed Press—Button Control



- Inputs *lbut* and *rbut* are generated by the physical control buttons
- This chart generates one output *activateP*
- The trigger $-lbut$ is true when the signal *lbut* is *not* in the input

Trace Semantics

$$[\![ButtonControlA]\!]^{\omega}_{\tau\text{-}chaos} =_{def} \{(i, o) \mid o \text{ is the output trace resulting from processing the input trace } i\}$$

Example Traces:

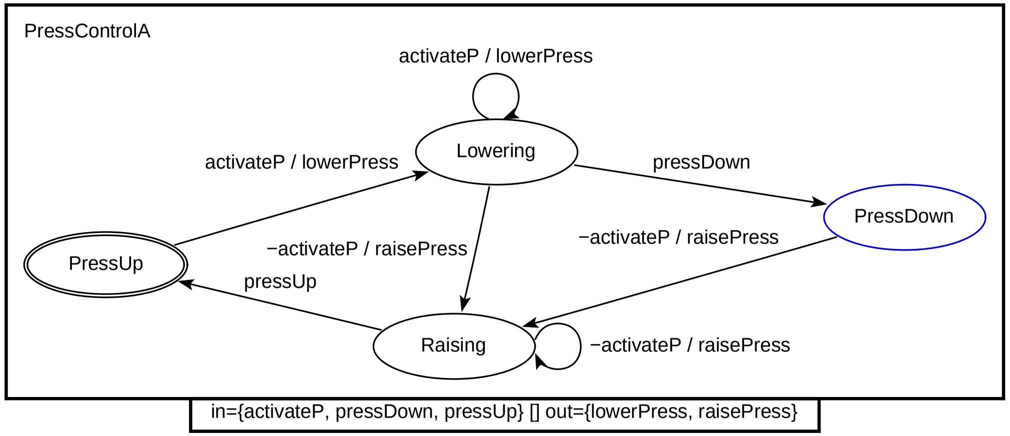
$$i = \langle \varnothing,\ \varnothing,\ \varnothing,\ \varnothing,\ \ldots$$
$$o = \langle \varnothing,\ \varnothing,\ \varnothing,\ \varnothing,\ \ldots$$

$$i = \langle \varnothing,\ \varnothing, \{lbut\}, \{lbut, rbut\}, \{lbut, rbut\}, \{rbut\}, \ldots$$
$$o = \langle \varnothing,\ \varnothing,\ \ \varnothing,\ \ \{activateP\}, \{activateP\},\ \ \varnothing, \ldots$$

Z Semantics

- Describe each transition as a schema
  - Each ‘transition schema’ has observations that describe:
    - the *before state* and *after state* of the chart
    - the input from the environment
    - the output from the chart
  - The predicate of the schema describes the desired relationship between input and output
- Each of these ‘transition schemas’ is combined using schema disjunction to denote the overall transition behaviour of the chart
- Z generated automatically by a tool—ZooM

Two-handed Press—Press Control



- Input *activateP* is generated by the button control system
- Inputs *pressUp* and *pressDown* are generated by the physical press
- This chart generates outputs *lowerPress* and *raisePress* to control the press
- This chart does not specify a reaction to the input *activateP* in state *Raising*

Chaotic Trace Semantics

Example Traces:

$$\begin{aligned} i &= \langle \{activateP\}, \ \{activateP\}, \ \{activateP, pressDown\}, \ \ldots \\ o &= \langle \{lowerPress\}, \{lowerPress\}, \ \varnothing, \ \ldots \end{aligned}$$

$$\begin{aligned} i &= \langle \{activateP\}, \ \{activateP\}, \ \{activateP, pressDown\}, \ \ldots \\ o &= \langle \{lowerPress\}, \{lowerPress\}, \ \{lowerPress\}, \ \ldots \end{aligned}$$

Let the special set of traces

$$Chaos = \{out \frown o \mid out \subseteq \{lowerPress, raisePress, \bot\} \wedge o \in Chaos\}$$

Now, for any input i that begins *without* signal $activateP$ the chart can exhibit all of the output traces from the set $Chaos$

For example:

$$\begin{aligned} i &= \langle \ \varnothing, \ \{activateP\}, \ \{activateP\}, \ \ldots \\ o &= \langle \{pressDown\}, \{pressDown, \bot\}, \ \varnothing, \ \ldots \end{aligned}$$

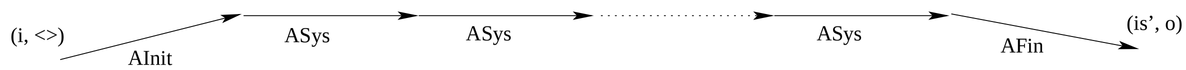
Relating the Trace semantics to the Z

To relate the Z semantics to the trace semantics we (carefully) embed the schema that describes the charts behaviour into a relational ADT framework.

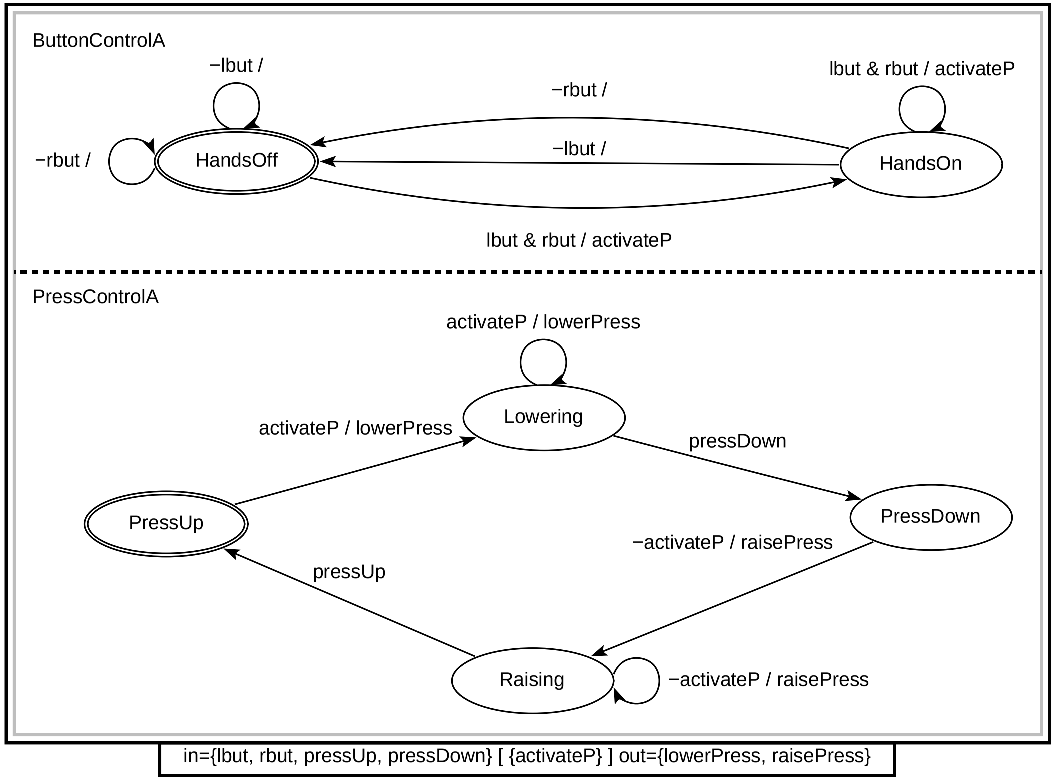
For arbitrary chart A think of the schema $ASys$ as the description of the one (and only) operation of this ADT, *i.e.* the 'lifted totalised' relational meaning of the schema gives the (possibly nondeterministic) operation in the relational ADT. The programs are made up by applying this operation again and again.

It is assumed that these programs consume inputs from a sequence provided at initialisation and produce an output sequence as the program is run.

Because these programs will typically be non-deterministic, we can view the global meaning of the ADT as a relation between input and output sequences. And therefore giving the trace semantic of a chart.



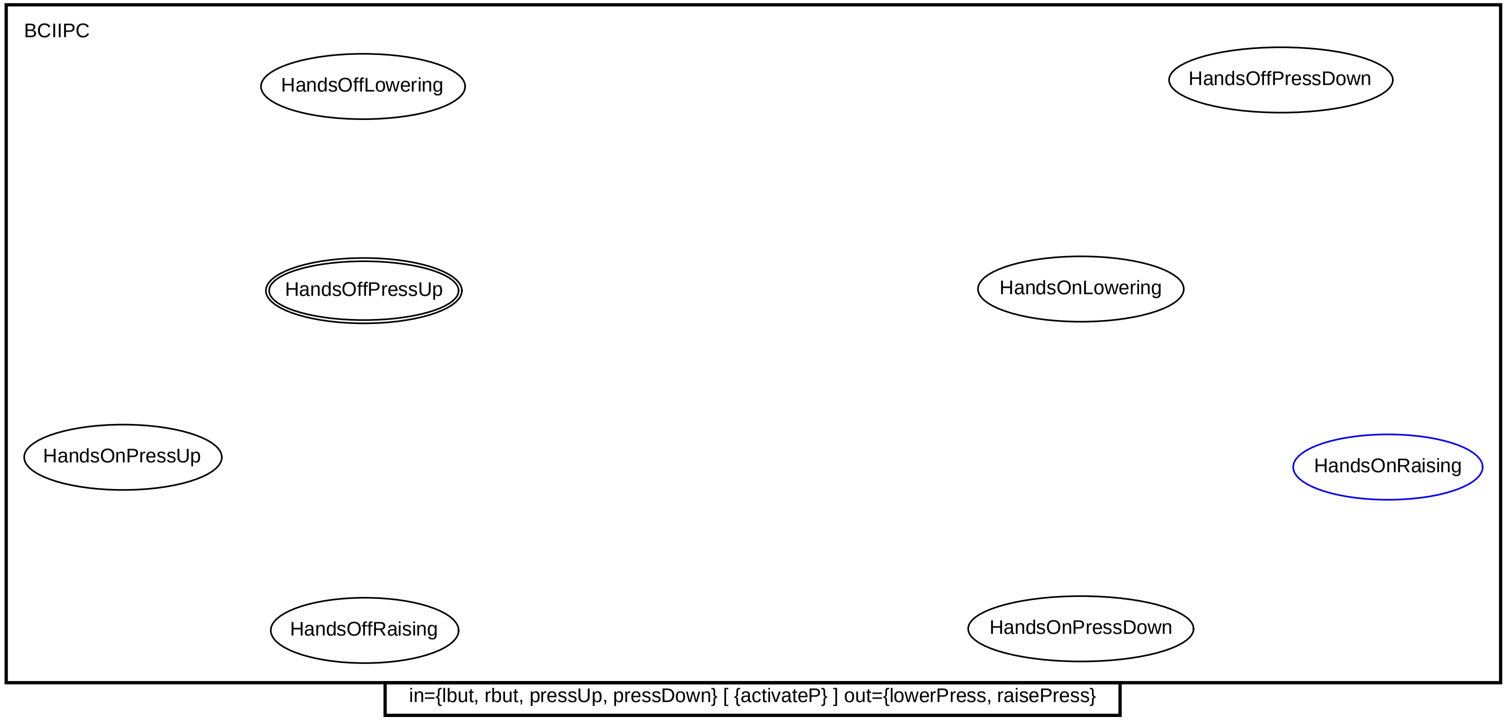
Semantics of Chart Composition—by example



- Transitions happen in lock-step, *i.e.* no other synchronisation
- Charts communicate using instantaneous feedback of signals

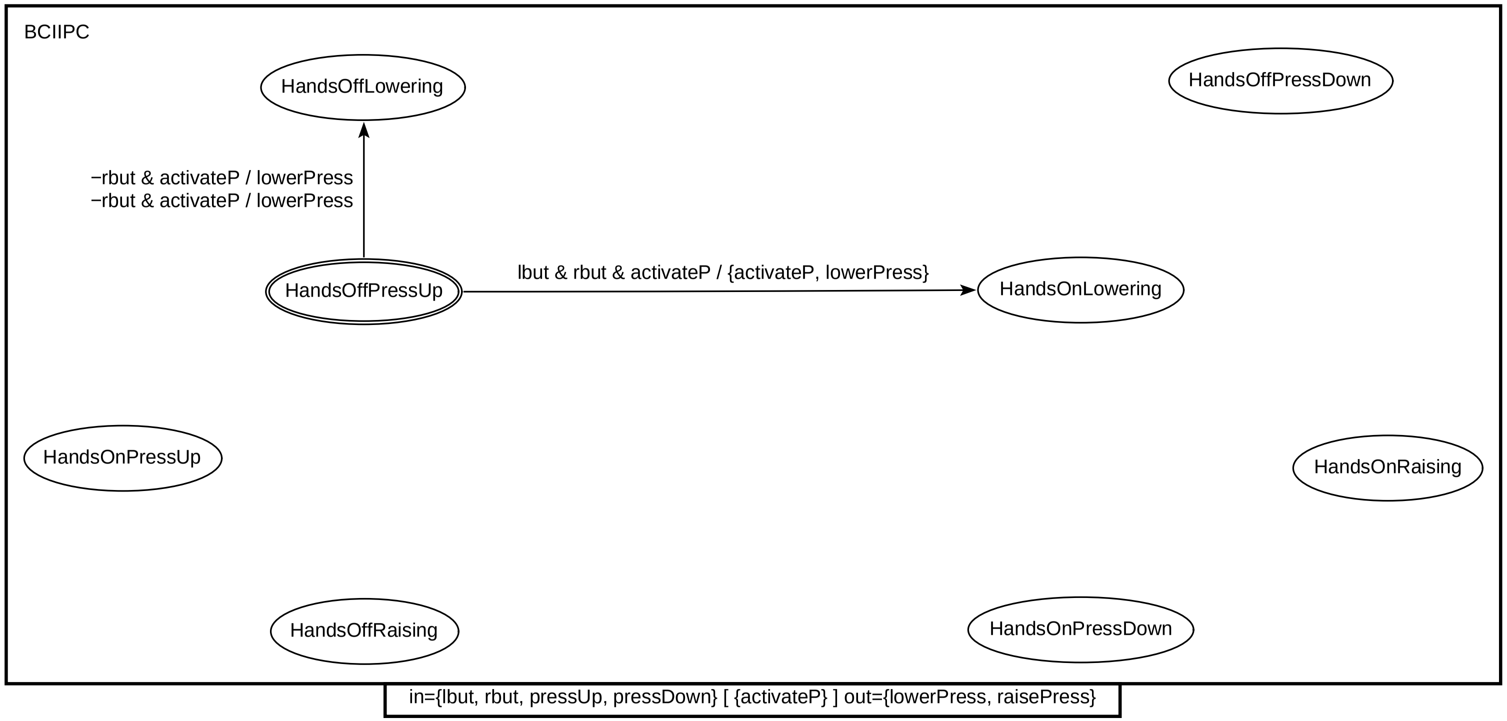
Calculating the composite meaning

- The meaning of the composition can be described by calculating an equivalent sequential chart
- First we take the cross product of the states:



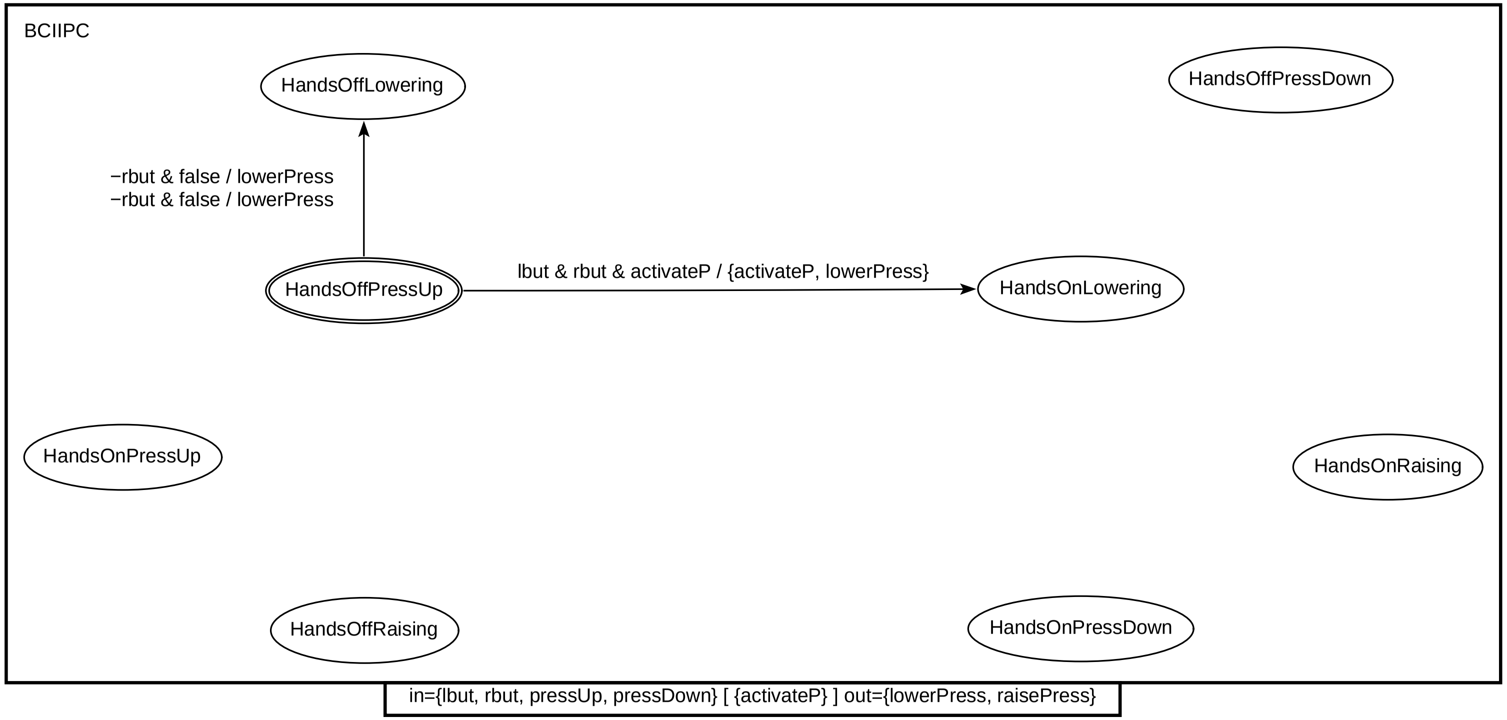
Calculating the composite meaning

- Then we take the cross product of transitions and simplify
- Three such transitions leave state $HandsOffPressUp$:



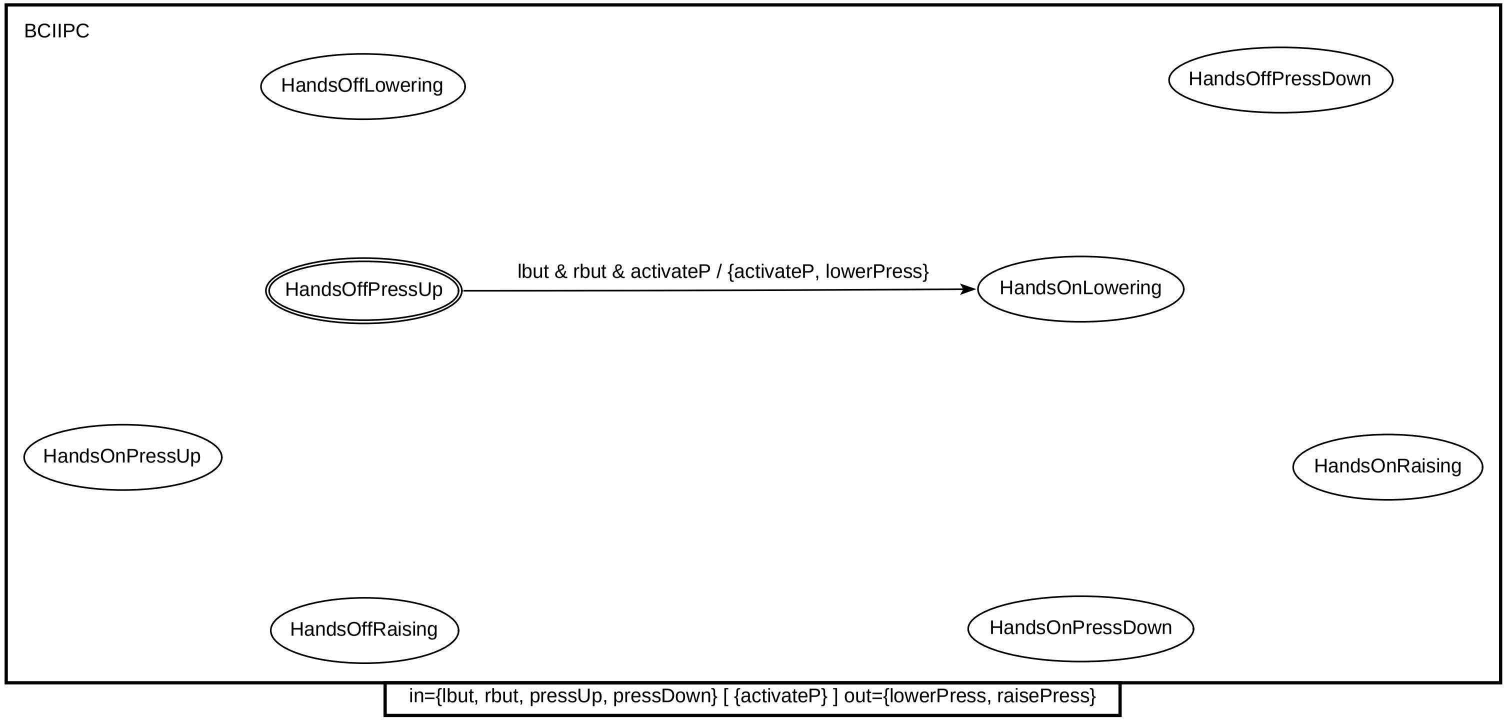
Calculating the composite meaning

- The signal *activateP* cannot be generated from the environment and therefore will never be seen as input to this chart
- Hence, any trigger that relies on input *activateP* cannot be true:



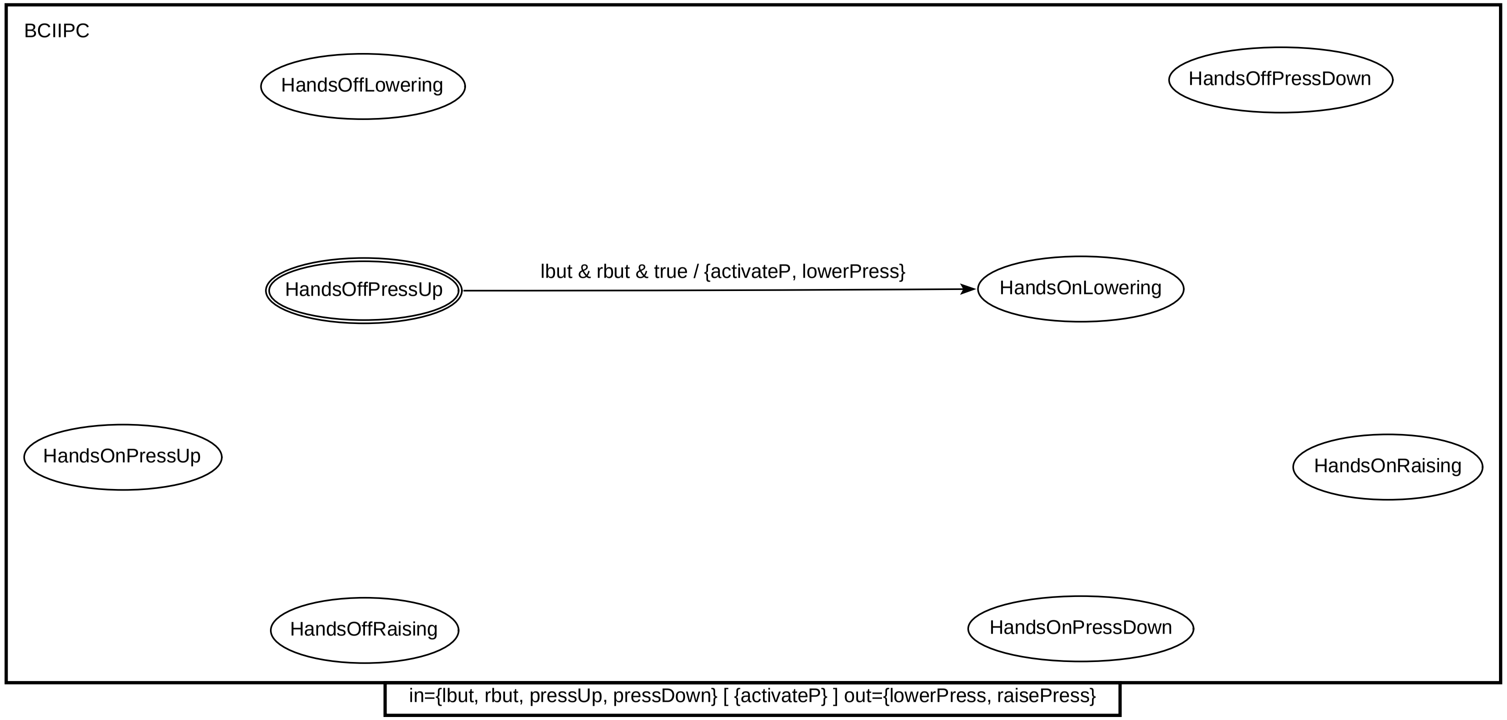
Calculating the composite meaning

- Transitions with false triggers can never happen and therefore can be removed:



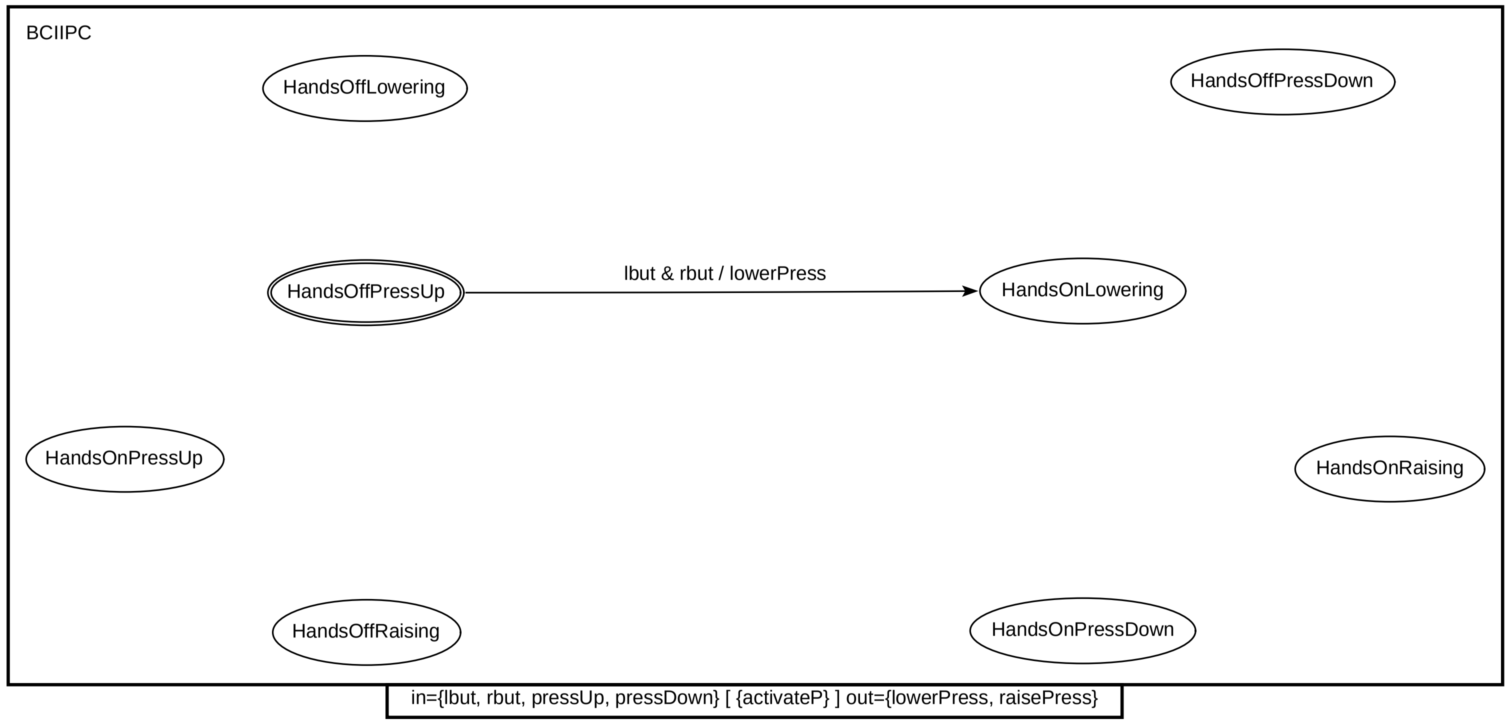
Calculating the composite meaning

- The output signal *activateP* is instantaneously available as input
- Hence, any transition that relies on input *activateP* and outputs *activateP* can be simplified:



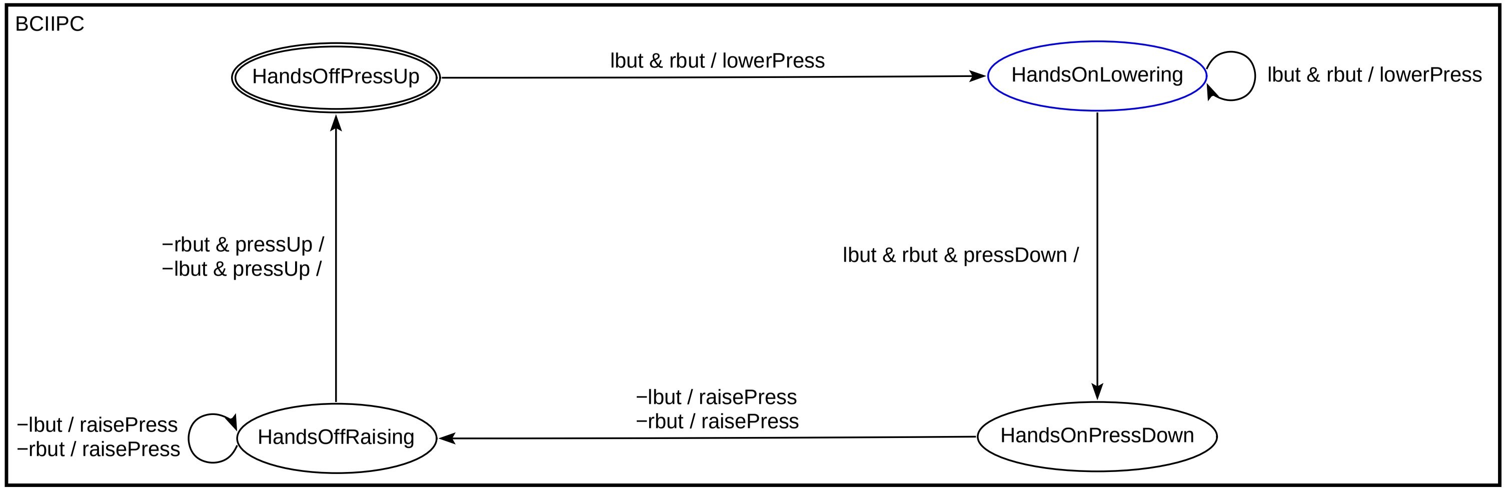
Calculating the composite meaning

- The output *activateP* resulted from an internal communication and is hidden or ignored by the environment.
- Hence, *activateP* can be removed from the output of any transition:



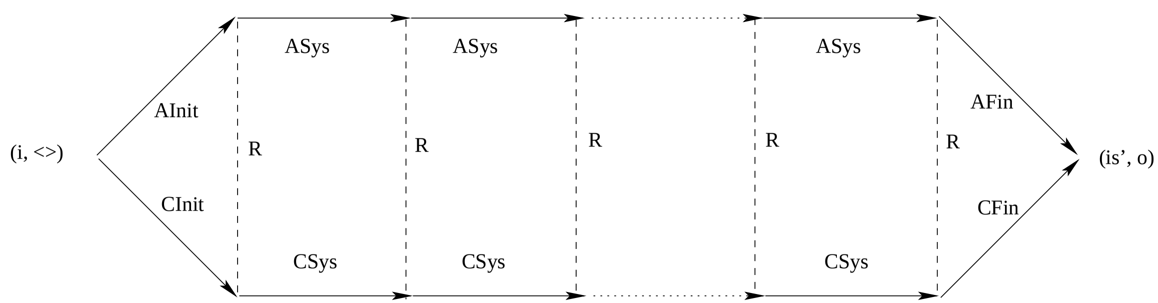
Calculating the composite meaning

- After completing this process for the entire cross product the resulting chart gives the meaning of the composition



Refinement for charts

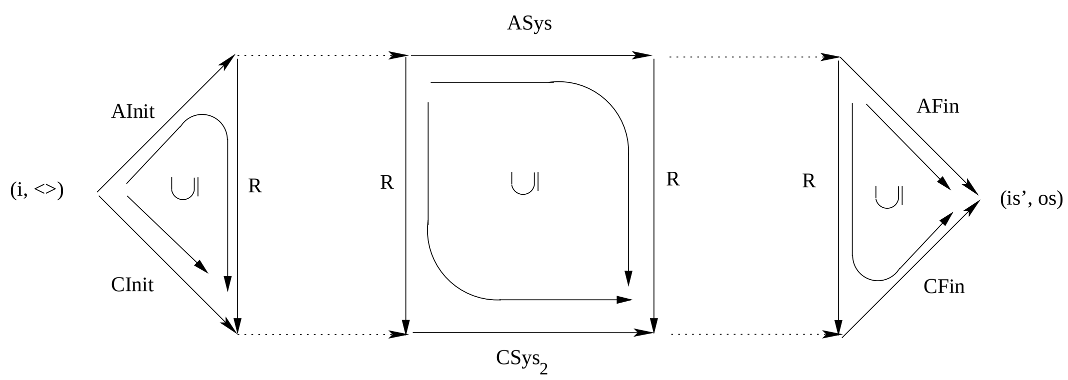
- Based on Data Refinement for Z
  - Woodcock and Davies
  - Derrick and Boiten
  - Moshe Deutsch
- Uses the relational ADT framework to derive refinement for charts using the Z semantics



Forward simulation refinement

For arbitrary charts A and C, and bindings u_c, v_a, v_c, y_a, y_c, and z_c, we have,

$$\frac{\begin{array}{ll} & \vdash out_A \subseteq out_C \\ u_c \in Init_C & \vdash t_1 \in Init_A \\ u_c \in Init_C & \vdash t_1 * u_c \mathrel{\dot{\in}} R \\ Pre\ ASys\ v_a,\ v_a \star v_c \mathrel{\dot{\in}} R & \vdash Pre\ CSys\ v_c \\ Pre\ ASys\ y_a,\ y_a \star y_c \mathrel{\dot{\in}} R,\ y_c \star z_c^! \in CSys & \vdash y_a \star t_2^! \in ASys \\ Pre\ ASys\ y_a,\ y_a \star y_c \mathrel{\dot{\in}} R,\ y_c \star z_c^! \in CSys & \vdash t_2 \star z_c \mathrel{\dot{\in}} R \end{array}}{C \sqsupseteq_{\tau f} A} \quad (\sqsupseteq_{\tau f}^{+})$$



Backwards simulation refinement

For arbitrary charts A and C, and arbitrary bindings u_a, u_c, v_a, v_c, y_a, y_c, x_c and $z_c \in T^+$, we have,

$$\frac{\begin{array}{ll} & \vdash z_c \star t_1 \mathrel{\dot{\in}} R \\ & \vdash in_A \subseteq in_C \\ u_c \in Init_C,\ u_c \star u_a \mathrel{\dot{\in}} R & \vdash u_a \in Init_A \\ v_c \star v_a \mathrel{\dot{\in}} R \Rightarrow Pre\ ASys\ v_a & \vdash Pre\ CSys\ v_c \\ y_c \star v_a \mathrel{\dot{\in}} R \Rightarrow Pre\ ASys\ v_a,\ y_c \star x_c^! \in CSys,\ x_c \star y_a \mathrel{\dot{\in}} R & \vdash y_c \star t_2 \mathrel{\dot{\in}} R \\ y_c \star v_a \mathrel{\dot{\in}} R \Rightarrow Pre\ ASys\ v_a,\ y_c \star x_c^! \in CSys,\ x_c \star y_a \mathrel{\dot{\in}} R & \vdash t_2 \star y_a^! \in ASys \end{array}}{C \sqsupseteq_{\tau b} A} \quad (\sqsupseteq_{\tau b}^+)$$

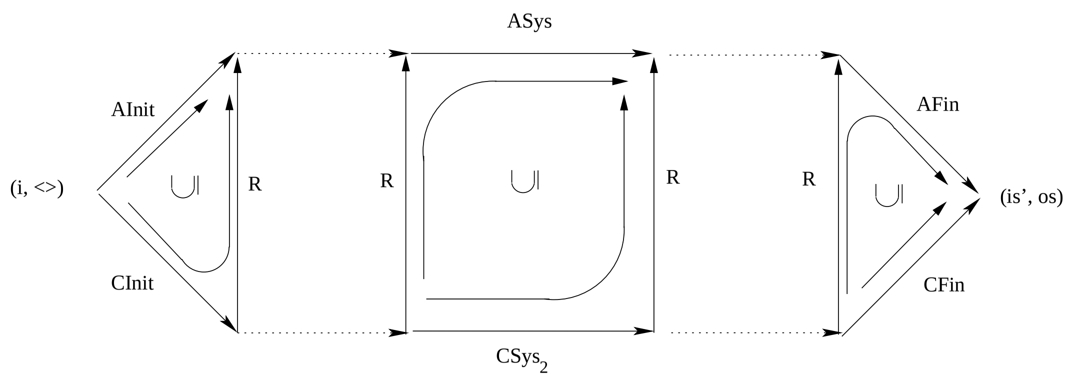


Chart refinement in terms of traces

In the simple case where $in_C = in_A$ and $out_C = out_A$ we have:

$$C \sqsupseteq_{\tau f} A \;\Rightarrow\; [\![C]\!]^{\omega}_{\tau\text{-}chaos} \subseteq [\![A]\!]^{\omega}_{\tau\text{-}chaos}$$

$$C \sqsupseteq_{\tau b} A \;\Rightarrow\; [\![C]\!]^{\omega}_{\tau\text{-}chaos} \subseteq [\![A]\!]^{\omega}_{\tau\text{-}chaos}$$

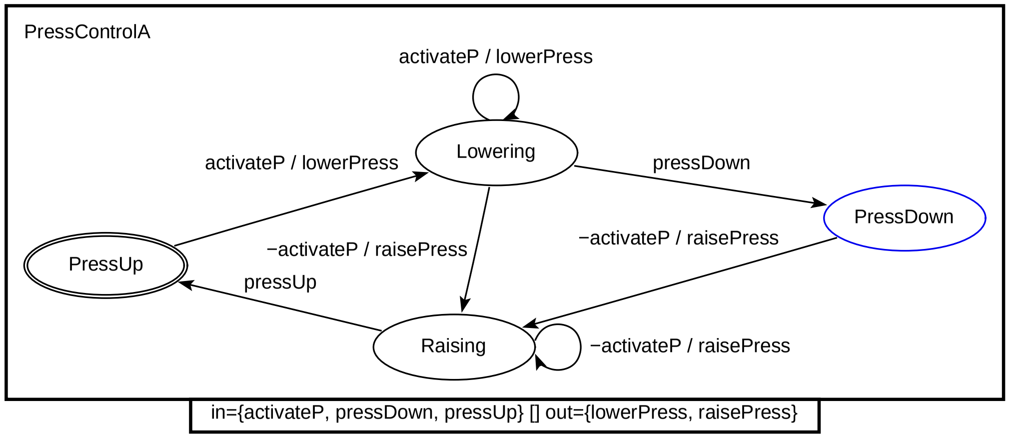
In the general we have:

$$C \sqsupseteq_{\tau f} A \;\Rightarrow\; \forall\, i, o \bullet (i_{\rhd(in_C)}, o_{\rhd(out_C)}) \in [\![C]\!]^{\omega}_{\tau\text{-}chaos} \Rightarrow (i_{\rhd(in_A)}, o_{\rhd(out_A)}) \in [\![A]\!]^{\omega}_{\tau\text{-}chaos}$$

$$C \sqsupseteq_{\tau b} A \;\Rightarrow\; \forall\, i, o \bullet (i_{\rhd(in_C)}, o_{\rhd(out_C)}) \in [\![C]\!]^{\omega}_{\tau\text{-}chaos} \Rightarrow (i_{\rhd(in_A)}, o_{\rhd(out_A)}) \in [\![A]\!]^{\omega}_{\tau\text{-}chaos}$$

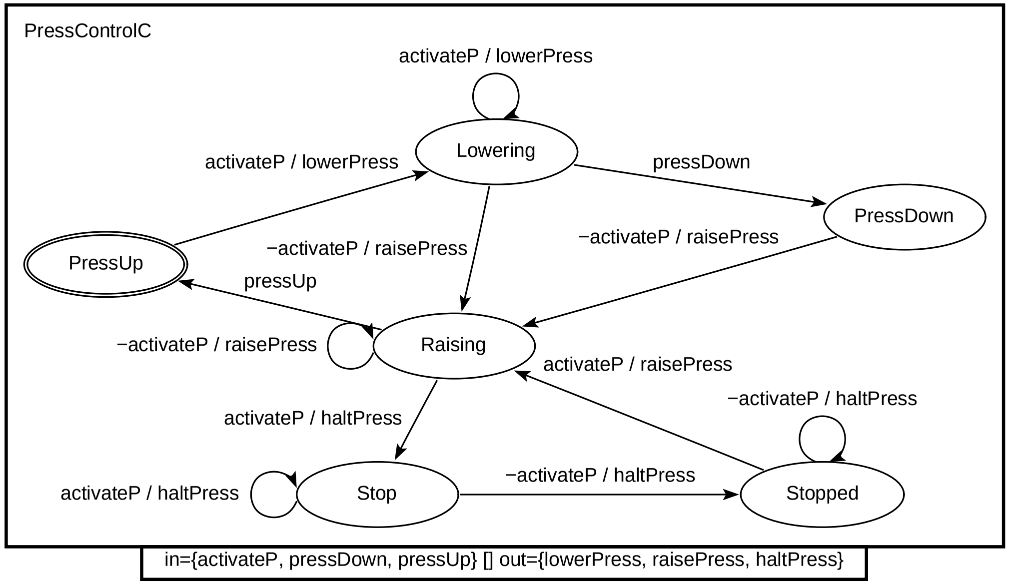
Chart refinement example

Recall the chart *PressControlA*



During design of the press control it is identified that it is useful if the press can be halted while it is returning to its uppermost position

The chart *PressControlC* describes a refinement of the original chart that allows for this new behaviour.



That is, we can show that: $PressControlC \sqsupseteq_{\tau f} PressControlA$

Monotonicity properties of chart refinement

- Monotonicity is concerned with structure development
- Given that μ-Charts contains structuring operators such as composition. Ideally we would like to be able show that refining one part of a composition results in a refinement of the composition itself.
- Therefore allowing structured formal development
- To show this desired monotonicity holds we would need to prove the following:

For arbitrary charts A_1, C_2, and B,

$$\frac{C_2 \quad \sqsupseteq^{R}_{\tau f} \quad A_1}{C_2 \mid \Psi \mid B \quad \sqsupseteq^{S}_{\tau f} \quad A_1 \mid \Psi \mid B}$$

where $S =_{def} \ Corr^{A_1}_{C_2} \wedge Corr^{B}_{B} \wedge IO^{A}_{C}, \quad R =_{def} \ Corr^{A_1}_{C_2} \wedge IO^{A_1}_{C_2}.$

From attempting to prove this property the following side-conditions were identified:

For arbitrary charts A_1, C_2, bindings y_a, z_a, y_c, and signal sets i_a, i_c and Ψ,

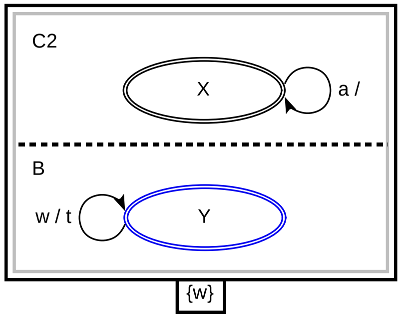
$$\frac{y_a \star z_a^{!} \mathrel{\dot\in} A_1 Sys \quad y_a \star y_c \mathrel{\dot\in} R}{\exists\, z_c \bullet y_c \star z_c \mathrel{\dot\in} C_2 Sys \wedge z_a.o_{A_1}! \cap \Psi = z_c.o_{C_2}! \cap \Psi}\ SC_1$$

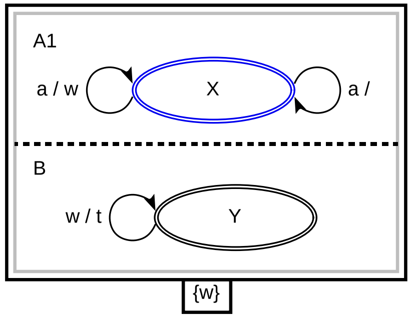
$$\frac{}{out_{A_1} \cap (\Psi \cup out_B) = out_{C_2} \cap (\Psi \cup out_B)}\ SC_2$$

$$\frac{Pre\ \ A_1 Sys\ (y_a \star y_{ia_\Psi}) \quad y_a \star y_{ia_1} \star y_c \star y_{ic_2} \mathrel{\dot\in} R \quad Pre\ \ C_2 Sys\ (y_c \star y_{ic_2})}{Pre\ \ A_1 Sys\ (y_a \star y_{ia_1})}\ SC_3$$

where $y_{ic_2} = \langle\!|\ i_{C_2}? \Rrightarrow (i_c \cup fb_C) \cap in_{C_2}\ |\!\rangle$, $y_{ia_1} = \langle\!|\ i_{A_1}? \Rrightarrow (i_a \cup fb_C) \cap in_{A_1}\ |\!\rangle$, and for arbitrary $fb_\Psi \subseteq \Psi$, $y_{ia_\Psi} = \langle\!|\ i_{A_1}? \Rrightarrow (i_a \cup fb_\Psi) \cap in_{A_1}\ |\!\rangle$.

- Side-condition SC_1 requires that the concrete chart C_2 has the same output with respect to feedback as the abstract chart A_1
- Take for example the following charts A_1, C_2 and B





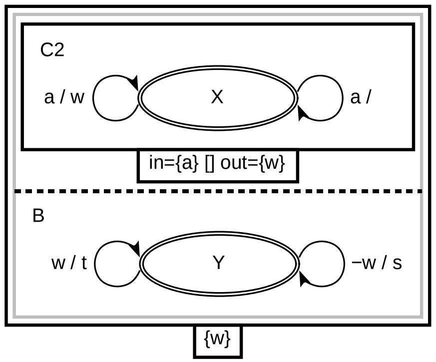
Here we have:

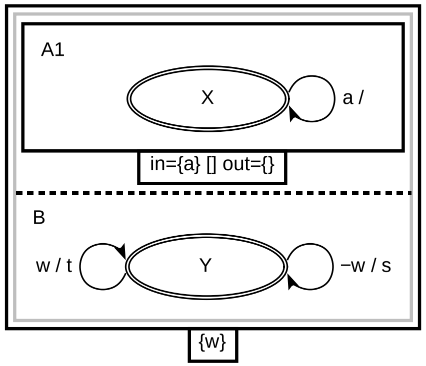
$$C_2 \sqsupseteq_{\tau f} A_1$$

But...

$$(C_2 \mid \{w\} \mid B) \not\sqsupseteq_{\tau f} (A_1 \mid \{w\} \mid B)$$

- SC_2 can be broken down into two conditions. The first requires that any feed back produced by C_2 is also produced by A_1
- Failure to meet this condition means that C_1 in composition can cause additional behaviour that A_1 cannot.





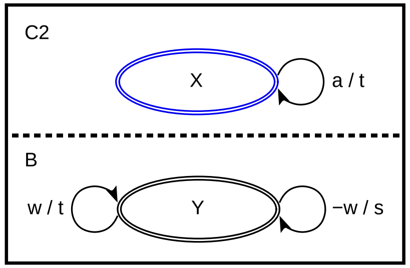
Again we have:

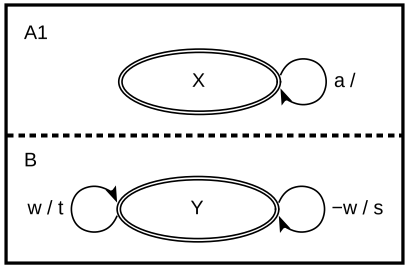
$$C_2 \sqsupseteq_{\tau f} A_1$$

But...

$$(C_2 \mid \{w\} \mid B) \not\sqsupseteq_{\tau f} (A_1 \mid \{w\} \mid B)$$

- The second condition imposed by SC_2 prevents refinements of part of a composition that increase the amount of control that a chart exhibits using signals “belonging” to the other part of the composition





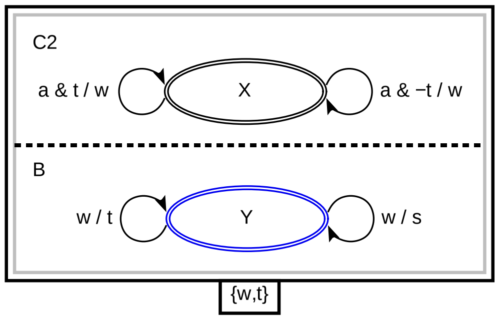
Again we have:

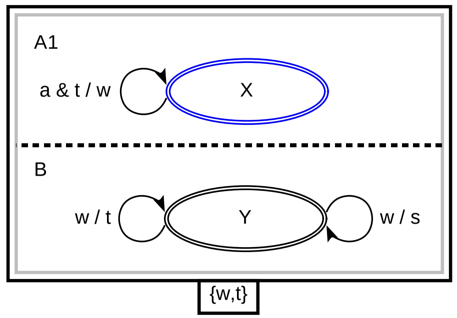
$$C_2 \sqsupseteq_{\tau f} A_1$$

But...

$$(C_2 \mid \{w\} \mid B) \not\sqsupseteq_{\tau f} (A_1 \mid \{w\} \mid B)$$

- SC_3 is the most subtle condition
- Placing a chart in composition can cause undefined behaviour to become defined





By itself chart A_1 is undefined for the singleton input a, yet when placed in composition with chart B the composition is exactly defined for input a, the additional input t is fed back from chart B.

A refinement of A_1 is free to define any reaction to input $\{a\}$. This newly defined behaviour may allow a new reaction to a when the refined chart is placed in the original composition.

Conclusions

We have demonstrated:

- how we can use the denotational semantics and logic for Z to induce a semantics for charts
- what data refinement on the Z semantics means in terms of traces of a charts behaviour
- the side conditions necessary to ensure this refinement is monotonic
- how we can use the logic and semantics provided by Z to investigate (with a high level of confidence) the properties of the language

Future:

- "Infectious chaos" versus unobservable chaos