Lemma 4

$$\vdash P \iff \vdash \mathcal{P}(P)$$

Proof

"$\Longrightarrow$" We need to show that

$$P \vdash c \Rightarrow \mathcal{P}(P) \vdash c$$

This requires showing that the 6 requirements for well-formed SCHOOL+F classes from FLD-WF-CLASS imply the four requirements for well-formed SCHOOL class from WF_CLASS. The first requirement is identical. The second requirement, i.e. $\mathcal{M}(\mathcal{P}(P), \mathcal{P}(P)\downarrow_4, m)$ follows from the similar requirement in P, from the fact that $P\downarrow_4(c) = \mathcal{P}(P)\downarrow_4(c)$, if the method signature is already defined in $\mathcal{M}(P, c, m)$. Otherwise, the method signature comes from a field encoding, in which case we use the sixth requirement from well-formed classes in FLD-WF-CLASS. Similarly for the requirements for asynchronous and synchronous chords. For the well-formedness of chord bodies we need to show that the encoding preserves type, and this was already shown in Theorem 3.

"$\Longleftarrow$" We now need to show that

$$\mathcal{P}(P) \vdash c \Rightarrow P \vdash c$$

And for this we need to show that the properties of the encoding implies on the four requirements for well-formed in WF Class, ensure the six requirements for FLD.WF-Class. 4.2

The preservation of the first requirement is trivial.
The preservation of the second requirement can be easily shown, because $\Phi(P)$ has more method signatures than P.
The preservation of the third requirement is also easy, because again $\Phi(P)$ has strictly more chords than P, and we use theorem 4 to argue preservation of the types.
Similarly for the fourth requirement.

With regards to the fifth requirement, ie that all field types are class types, however, there is no guarantee in the well formedness of $\Phi(P)$. Therefore it is possible to have a P whose fields have void or async type, and $\Phi(P)$ would be wellformed. This is a small proviso that we have not expressed in the lemma.

Finally, with regards to the last requirement, that any field inherited from a superclass has same type in the subclass, this requirement follows from the fact that for any field f l in in C in P there would be a method t get_f(l) in C in $\Phi(P)$, and then application of rules about method overriding.