MODULE *MoneyTransfer*

EXTENDS *Integers*

VARIABLES $accountA$, $accountB$, $money$, $pc$

CONSTANTS $A$, $B$, $M$

$$PositiveInteger(n) \triangleq n \in Nat \land n \neq 0$$

The "money" is transferred from account A to account $B$. The "$pc$" variable is used to control the program flow. Remember, there is no order of execution between "Subtract" and "Add"

$$\begin{aligned} TypeInvariant \triangleq\ & \land pc \in \{\text{"SUB"}, \text{"ADD"}, \text{"DONE"}\} \\ & \land accountA \in Nat \\ & \land accountB \in Nat \\ & \land PositiveInteger(money) \quad \text{no zero transfer} \end{aligned}$$

$$\begin{aligned} Init \triangleq\ & \land accountA = A \\ & \land accountB = B \\ & \land money = M \\ & \land pc = \text{"SUB"} \end{aligned}$$

The "money" is subtracted from account A if it has enough funds. After that the flow is passed to "Add" by changing "$pc$" values to "ADD". If the balance is not enough, there is no next state, preventing from adding "money" to account $B$.

$$\begin{aligned} Subtract \triangleq\ & \land pc = \text{"SUB"} \\ & \land \text{IF } accountA \geq money \\ & \qquad \text{THEN } accountA' = accountA - money \land pc' = \text{"ADD"} \\ & \qquad \text{ELSE UNCHANGED } \langle accountA \rangle \land pc' = \text{"DONE"} \\ & \land \text{UNCHANGED } \langle accountB, money \rangle \end{aligned}$$

The "money" is added to account $B$ and after that the flow is changing to "DONE", meaning that there is no next state.

$$\begin{aligned} Add \triangleq\ & \land pc = \text{"ADD"} \\ & \land accountB' = accountB + money \\ & \land pc' = \text{"DONE"} \\ & \land \text{UNCHANGED } \langle accountA, money \rangle \end{aligned}$$

$$Next \triangleq Subtract \lor Add$$

$$vars \triangleq \langle accountA, accountB, money, pc \rangle$$

$$Spec \triangleq Init \land \Box[Next]_{\langle vars \rangle}$$