```
function IS_BOOLEAN_FORMULA($spec$)
    if $spec\_Type \in BasicTypes$ then
        return $True$
    end if
    if $spec\_Type = \neg$ then
        return IS_BOOLEAN_FORMULA($spec.LHS$)
    end if
    if $spec\_Type \in BooleanOP$ then
        return IS_BOOLEAN_FORMULA($spec.LHS$)$\wedge$ IS_BOOLEAN_FORMULA($spec.RHS$)
    end if
    return $False$
end function

function CHECK_GF_FORMULA($spec$)
    if $spec\_Type \neq \Box$ then
        return $False$
    end if
    if $spec\_Type \neq \Diamond$ then
        return $False$
    end if
    if IS_BOOLEAN_FORMULA($spec.LHS$)= True then
        return $spec.LHS$
    else
        return $None$
    end if
end function

function PARSE_REACT($spec$)
    if $spec\_Type \neq Context$ then
        return $None$
    end if
    $spec \leftarrow spec\_RHS$
    if $spec\_Type \neq \rightarrow$ then
        return $None$
    end if
    $f \leftarrow$ CHECK_GF_FORMULA($spec\_LHS$)
    if $f = None$ then
        return $None$
    end if
    $g \leftarrow$ CHECK_GF_FORMULA($spec\_RHS$)
    if $g = None$ then
        return $None$
    end if
    return $(f, g)$
end function
```

```
function CHECK_REACT_SPEC($spec$)
    if Parse_React($spec$) = $None$ then
        return $None$
    else
        $f, g \leftarrow$ Parse_React($spec$)
        $ng \leftarrow \neg(g)$
        $bddspec\_f \leftarrow$ SPEC_TO_BDD($bddfsm, f$)
        $bddspec\_ng \leftarrow$ SPEC_TO_BDD($bddfsm, ng$)
        $reach \leftarrow$ REACH($bddfsm, init$)
        $new \leftarrow$ POST($bddfsm, reach$)
        while $new \neq$ INTERSECTION($reach, new$) do
            $reach \leftarrow$ UNION(DIFF($new, reach$), $reach$)
            $new \leftarrow$ POST($bddfsm, reach$)
        end while
        $cycle \leftarrow$INTERSECTION(INTERSECTION($reach, f$), $ng$)
        $found\_cycle \leftarrow \emptyset$
        $sub\_reach \leftarrow reach$
        while INTERSECTION($sub\_reach, cycle$) $\wedge \neg(found\_cycle)$  do
            $sub\_reach \leftarrow$INTERSECTION(PRE($reach, f$), $ng$)
            $new \leftarrow sub\_reach$
            while count_states(bddfsm, new) $> 0$ do
                $sub\_reach \leftarrow$ UNION(DIFF($new, sub\_reach$), $sub\_reach$)
                if entailed($sub\_reach, cycle$) then
                    $found\_cycle \leftarrow True$
                    $break$
                end if
                $new \leftarrow$ INTERSECTION(DIFF(PRE($bddfsm, new$), $sub\_reach$), $bddspec\_ng$)
            end while
            $cycle \leftarrow$ INTERSECTION($sub\_reach, cycle$)
        end while
        if $\neg(found\_cycle)$ = True then
            return $True$
        end if
    end if
end function
```