

---

**Rule 0.23** *(Cache Storage Variables in Loops)*


---

<pre> [...]  <b>contract</b> <math>A</math> {    [...]    <b>function</b> <math>f(pds)</math> {      [...]      <b>for</b>(<math>init</math>; <math>cond</math>; <math>upd</math>) {        <math>stmts[storageVar]</math>      }      <math>stmts'</math>    }    [...]  }</pre>	=	<pre> [...]  <b>contract</b> <math>A'</math> {    [...]    <b>function</b> <math>f(pds)</math> {      [...]      <math>T</math> <math>cache = storageVar</math>;      <b>for</b>(<math>init</math>; <math>cond</math>; <math>upd</math>) {        <math>stmts[cache]</math>      }      <math>storageVar = cache</math>;      <math>stmts'</math>    }    [...]  }</pre>
---	---	--

**where**

$storageVar$  is a storage variable accessed repeatedly within the loop;  
 $cache$  is a local memory variable of type  $T$  that caches  $storageVar$ ;  
 $T$  is the type of the storage variable;  
 $stmts[storageVar]$  represents loop body statements that access  $storageVar$ ;  
 $stmts[cache]$  represents the same statements with  $storageVar$  replaced by  $cache$ ;  
 $init$ ,  $cond$ , and  $upd$  are the loop initialization, condition, and update expressions;  
 $pds$  are the parameter declarations of function  $f$ ;  
 $stmts'$  represents statements following the loop.

**provided**

The storage variable  $storageVar$  is accessed multiple times within the loop;  
 No external calls or state-modifying operations within the loop affect  $storageVar$ ;  
 The cached value is written back to storage after the loop completes;  
 All modifications to  $storageVar$  within the loop can be safely performed on  $cache$ ;  
 The loop does not modify  $storageVar$  through aliasing or indirect references.

**Invariant:**

Let  $s_i$  and  $s'_i$  be the initial state of  $A$  and  $A'$ , respectively.  
 Let  $s_f$  and  $s'_f$  be the state reached by  $A$  and  $A'$ , respectively, after  $A.f()$  and  $A'.f()$  are executed from  $s_i$  and  $s'_i$ , respectively.  
 Then, the coupling invariant is

$$\forall s_i, s'_i . (s_i = s'_i) \rightarrow (s_f = s'_f)$$


---