**3GPP TSG-SA4 Meeting #122 *S4-230355***

**Athens, Greece, February 20 - 24, 2023**  (***Rev 1* - *S4-230054***)

|  |
| --- |
| *CR-Form-v12.2* |
| **CHANGE REQUEST** |
|  |
|  | **26.173** | **CR** | **0035** | **rev** | **-** | **Current version:** | **17.0.0** |  |
|  |
| *For* [***HELP***](http://www.3gpp.org/3G_Specs/CRs.htm#_blank)*on using this form: comprehensive instructions can be found at* [*http://www.3gpp.org/Change-Requests*](http://www.3gpp.org/Change-Requests)*.* |
|  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Proposed change affects:*** | UICC apps |  | ME | **X** | Radio Access Network |  | Core Network | **X** |

|  |
| --- |
|  |
| ***Title:***  | Correction to Undefined Behaviour caused by out-of-bounds pointer arithmetic |
|  |  |
| ***Source to WG:*** | Google |
| ***Source to TSG:*** | SA4 |
|  |  |
| ***Work item code:*** | 34004 |  | ***Date:*** | 2023-02-10 |
|  |  |  |  |  |
| ***Category:*** | **F** |  | ***Release:*** | Rel-17 |
|  | *Use one of the following categories:****F*** *(correction)****A*** *(mirror corresponding to a change in an earlier release)****B*** *(addition of feature),* ***C*** *(functional modification of feature)****D*** *(editorial modification)*Detailed explanations of the above categories canbe found in 3GPP [TR 21.900](http://www.3gpp.org/ftp/Specs/html-info/21900.htm). | *Use one of the following releases:Rel-8 (Release 8)Rel-9 (Release 9)Rel-10 (Release 10)Rel-11 (Release 11)…Rel-16 (Release 16)Rel-17 (Release 17)Rel-18 (Release 18)Rel-19 (Release 19)* |
|  |  |
| ***Reason for change:*** | Maintaining consistency with the CR #S4-230053 for 26.204 (the floating-point implementation) to fix an out-of-bounds pointer arithmetic Undefined Behavior. |
|  |  |
| ***Summary of change:*** | The CR corrects the UB by obtaining the offset and passing it around alongside the original array to avoid out-of-bounds pointer arithmetic. |
|  |  |
| ***Consequences if not approved:*** | An Undefined Behaviour in the code, UBSAN failures. |
|  |  |
| ***Clauses affected:*** | c-code/c4t64fx.cc-code/pitch\_f4.c |
|  |  |
|  | **Y** | **N** |  |  |
| ***Other specs*** |  | **X** |  Other core specifications  | TS/TR ... CR ...  |
| ***affected:*** |  | **X** |  Test specifications | TS/TR ... CR ...  |
| ***(show related CRs)*** |  | **X** |  O&M Specifications | TS/TR ... CR ...  |
|  |  |
| ***Other comments:*** |  |
|  |  |
| ***This CR's revision history:*** |  |

**\*\*\* Start change 1 \*\*\***

--- a/c4t64fx.c

+++ b/c4t64fx.c

@@ -100,6 +100,7 @@ void ACELP\_4t64\_fx(

 Word16 i, j, k, st, ix, iy, pos, index, track, nb\_pulse, nbiter;

 Word16 psk, ps, alpk, alp, val, k\_cn, k\_dn, exp;

 Word16 \*p0, \*p1, \*p2, \*p3, \*psign;

+ Word16 p0\_offset;

 Word16 \*h, \*h\_inv, \*ptr\_h1, \*ptr\_h2, \*ptr\_hf, h\_shift;

 Word32 s, cor, L\_tmp, L\_index;

@@ -366,13 +367,14 @@ void ACELP\_4t64\_fx(

 pos = MSIZE - 1; move16();

 ptr\_hf = h + 1; move16();

+ p0\_offset = -NB\_POS; move16();

 for (k = 0; k < NB\_POS; k++)

 {

 p3 = &rrixiy[2][pos]; move16();

 p2 = &rrixiy[1][pos]; move16();

 p1 = &rrixiy[0][pos]; move16();

- p0 = &rrixiy[3][pos - NB\_POS]; move16();

+ p0 = &rrixiy[3][pos]; move16();

 cor = 0x00008000L; move32(); /\* for rounding \*/

 ptr\_h1 = h; move16();

@@ -395,7 +397,7 @@ void ACELP\_4t64\_fx(

 cor = L\_mac(cor, \*ptr\_h1, \*ptr\_h2);

 ptr\_h1++;

 ptr\_h2++;

- \*p0 = extract\_h(cor); move16();

+ \*(p0 + p0\_offset) = extract\_h(cor); move16();

 p3 -= (NB\_POS + 1);

 p2 -= (NB\_POS + 1);

--- a/pitch\_f4.c

+++ b/pitch\_f4.c

@@ -23,7 +23,8 @@ static void Norm\_Corr(

 Word16 L\_subfr, /\* (i) : Length of subframe \*/

 Word16 t\_min, /\* (i) : minimum value of pitch lag. \*/

 Word16 t\_max, /\* (i) : maximum value of pitch lag. \*/

- Word16 corr\_norm[] /\* (o) Q15 : normalized correlation \*/

+ Word16 corr\_norm[], /\* (o) Q15 : normalized correlation \*/

+ Word16 corr\_offset /\* (i) : normalized correlation offset. \*/

 );

 static Word16 Interpol\_4( /\* (o) : interpolated value \*/

 Word16 \* x, /\* (i) : input vector \*/

@@ -47,7 +48,7 @@ Word16 Pitch\_fr4( /\* (o) : pitch period.

 Word16 i;

 Word16 t\_min, t\_max;

 Word16 max, t0, fraction, step, temp;

- Word16 \*corr;

+ Word16 corr\_v\_offset;

 Word16 corr\_v[40]; /\* Total length = t0\_max-t0\_min+1+2\*L\_inter \*/

 /\* Find interval to compute normalized correlation \*/

@@ -55,16 +56,16 @@ Word16 Pitch\_fr4( /\* (o) : pitch period.

 t\_min = sub(t0\_min, L\_INTERPOL1);

 t\_max = add(t0\_max, L\_INTERPOL1);

- corr = &corr\_v[-t\_min];

+ corr\_v\_offset = -t\_min;

 move16();

 /\* Compute normalized correlation between target and filtered excitation \*/

- Norm\_Corr(exc, xn, h, L\_subfr, t\_min, t\_max, corr);

+ Norm\_Corr(exc, xn, h, L\_subfr, t\_min, t\_max, corr\_v, corr\_v\_offset);

 /\* Find integer pitch \*/

- max = corr[t0\_min];

+ max = corr\_v[corr\_v\_offset + t0\_min];

 move16();

 t0 = t0\_min;

 move16();

@@ -72,10 +73,10 @@ Word16 Pitch\_fr4( /\* (o) : pitch period.

 for (i = add(t0\_min, 1); i <= t0\_max; i++)

 {

 test();

- if (sub(corr[i], max) >= 0)

+ if (sub(corr\_v[corr\_v\_offset + i], max) >= 0)

 {

- max = corr[i]; move16();

- t0 = i; move16();

+ max = corr\_v[corr\_v\_offset + i]; move16();

+ t0 = i; move16();

 }

 }

@@ -111,11 +112,11 @@ Word16 Pitch\_fr4( /\* (o) : pitch period.

 fraction = 0;

 move16();

 }

- max = Interpol\_4(&corr[t0], fraction);

+ max = Interpol\_4(&corr\_v[corr\_v\_offset + t0], fraction);

 for (i = add(fraction, step); i <= 3; i = (Word16) (i + step))

 {

- temp = Interpol\_4(&corr[t0], i);

+ temp = Interpol\_4(&corr\_v[corr\_v\_offset + t0], i);

 test();

 if (sub(temp, max) > 0)

@@ -157,7 +158,8 @@ static void Norm\_Corr(

 Word16 L\_subfr, /\* (i) : Length of subframe \*/

 Word16 t\_min, /\* (i) : minimum value of pitch lag. \*/

 Word16 t\_max, /\* (i) : maximum value of pitch lag. \*/

- Word16 corr\_norm[]) /\* (o) Q15 : normalized correlation \*/

+ Word16 corr\_norm[], /\* (o) Q15 : normalized correlation \*/

+ Word16 corr\_offset) /\* (i) : normalized correlation offset. \*/

 {

 Word16 i, k, t;

 Word16 corr, exp\_corr, norm, exp\_norm, exp, scale;

@@ -215,7 +217,7 @@ static void Norm\_Corr(

 L\_tmp = L\_mult(corr, norm);

 L\_tmp = L\_shl(L\_tmp, add(add(exp\_corr, exp\_norm), scale));

- corr\_norm[t] = round(L\_tmp); move16();

+ corr\_norm[corr\_offset + t] = round(L\_tmp); move16();

 /\* modify the filtered excitation excf[] for the next iteration \*/

**\*\*\* End change 1 \*\*\***