Использование ICC

Установил icc в Fedora 7 (только компилятор). Пришлось дополнительно yum install compat-libstdc++-33.i386

Документация: /opt/intel/cc/10.0.023/doc/main_cls/whnjs.htm

Использовал векторизацию с ключём -xP. Всовокупности с ippsSin время 5.808/кадр.

Изменения в Makefile и циклах:

```
diff -ruN src-org/Makefile src-icc/Makefile
--- src-org/Makefile    2007-09-17 17:43:08.000000000 +0400
+++ src-icc/Makefile    2007-11-03 22:28:21.000000000 +0300
@@ -53,22 +53,23 @@

 OBJ = .o

-CC         = gcc
+CC         = icc
 CPLAT      =
 CPROC      =
 CINC       = -I$(SRC_DIR)
 CDEFS      =
 COBJ       = -c -o$(OBJ_DIR)/$@
-CDEFOPT    = -O2
+CDEFOPT    = -O3 -xP -fp-model fast -fp-speculation fast -fno-math-errno -g
 COPT       =
 CFLAGS     =
 CFLAGS_ALL = $(CFLAGS) $(CINC) $(CDEFS) $(CDEFOPT) $(CPROC) $(CPLAT)

 LD         = g++
-LDPLAT     =
-LDFLAGS    =
+LDPLAT     =
+LDFLAGS    = -L/opt/intel/cc/10.0.023/lib
 LDOUTOPT   = -o "$(OUT_DIR)/$(BENCHMARK)"
-LIBS       = -lm -lc
+LIBS       = -lm -lc -lirc -limf -lsvml -lippcore -lippvm -lipgo
+# -lompstub -lomp_db -lguide
 LIBS_ALL   = $(LIBS)

 endif
diff -ruN src-org/sunset.cpp src-icc/sunset.cpp
--- src-org/sunset.cpp  2007-09-16 12:04:44.000000000 +0400
+++ src-icc/sunset.cpp  2007-11-05 11:38:25.000000000 +0300
@@ -45,6 +45,7 @@
 #include <omp.h>
 #endif
 #include "sunset.h"
```

```
+#include <ippvm.h>

 #define MIN(x,y)    (((x) < (y)) ? (x) : (y))
 #define MAX(x,y)    (((x) < (y)) ? (y) : (x))
@@ -730,6 +731,9 @@
 !!!!!!!!!!!!!!!! Water surface modelling !!!!!!!!!!!!!!!!!!!!!!!
 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
 */
+
+                pFlTmp = flArgSin[currentthread].aptr;
+
                 for(t = 0; t < NKMAX; t++)
                 {
                     OT  = flOmega[t] * flTime;
@@ -739,17 +743,16 @@
                     for(l = 0; l < iAngleHarmNum; l++)
                     {
                         iSinIndex1 = t * iAngleHarmNum + l;
-                        flArgSin[currentthread].aptr[iSinIndex1] = OT -
+                        pFlTmp[iSinIndex1] = OT -
                             KX1 * flAzimuthCosFi[l] - KY1 *
flAzimuthSinFi[l] +
-                            flRandomPhase[t*iAngleHarmNum + l];
+                            flRandomPhase[iSinIndex1];
                     } /* end for l */
                 } /* end for t */

-                pFlTmp = flArgSin[currentthread].aptr;
-
-                #pragma ivdep
-                for(t=0; t<iWaveMeshSize; t++)
-                    pFlTmp[t] = (float)sinf(pFlTmp[t]);
+                ippsSin_32f_A11(pFlTmp,pFlTmp,iWaveMeshSize);
+                //#pragma ivdep
+                //for(t=0; t<iWaveMeshSize; t++)
+                //    pFlTmp[t] = (float)sinf(pFlTmp[t]);

                 /* initialize the values of derivation */
                 flDerivX = 0.0f;
```