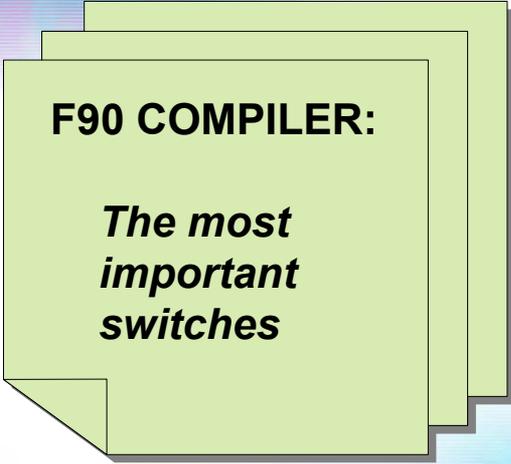


THEORY IV

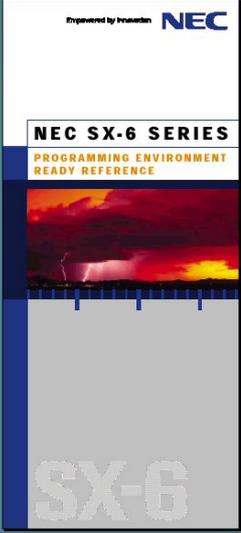
Empowered by Innovation **NEC**



F90 COMPILER:

*The most
important
switches*

Empowered by Innovation **NEC**



923982 Jan 15 11:26 Ready-Reference.pdf on CD

Empowered by Innovation **NEC**

Important switches

F90 and C++

- C aopt (only C++) aggressive vectorization and optimization (e.g. netcdf)
- C hopt Full use of vectorization and optimization
- C vopt Standard use of vectorization and optimization
- C sopt Full use of optimization in scalar code
- C vsafe Very safe use of vectorization and optimization
- C ssafe Safe use of optimization in scalar code
- C noopt (only C++) almost all optimizations and vectorization are suppressed
- C debug No vectorization and optimization at all

Important switches F90

- c *compile only*
- ew *Define basic storage unit as 8 bytes*
- eC *Enable runtime array bounds checking*
- Ep *Invoke Preprocessing*
- EP *-Ep + save intermediate files*
- f0 *standard fixed format*
- f3 *extended fixed format*
- f4 *standard free format*
- ftrace *flow trace profiling*
- g *Generate program debugging tables*
- p *Enable Profiling*

Empowered by Innovation

NEC

Word-Length

Please have always in mind

| | Real | Real*4 | Integer |
|-------------|------|--------|---------|
| Cray | 8 | 8 | 8 |
| SX-6 | 4 | 4 | 4 |
| SX-6 (-ew) | 8 | 8 | 8 |
| SX-6 (dbl4) | 8 | 8 | 4 |
| Workstation | 4 | 4 | 4 |
| WS (-r8) | 8 | 4 | 4 |

Empowered by Innovation

NEC

Out of bounds references?

- f90 -eC
 - ‘array bounds checking’ enabled
- f90 -eC -Cvsafe
 - because otherwise:

```
real a(n,m)
...
Do j = 1, m
  do i = 1, n
    a(i,j) = 0.0
  end do
end do
```

Collapse



```
real a(n,m)
...
do i = 1, n*m
  a(i,1) = 0.0
end do
...
```

Empowered by Innovation

NEC

Important switches F90

| | |
|-----------|------------------------------------------------------------------------------|
| -P static | <i>Enable only vector processing (Default)</i> |
| -P auto | <i>Enable automatic parallel processing (not OpenMp)</i> |
| -P openmp | <i>Enable OpenMP parallel processing</i> |
| -P multi | <i>Enable parallel processing (local data allocated in task data memory)</i> |
| -P stack | <i>Enable parallel processing (local data allocated on stack)</i> |

Empowered by Innovation

NEC

Important switches F90

| | | |
|-----------|------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| pi | | Specifies that the procedure inline function be used. |
| | auto | (default) Specifies automatic expansion. |
| | noauto | Specifies explicit inline expansion. |
| | line=<i>n</i> | Specifies the maximum number of lines (<i>n</i>) in a procedure for which automatic inline expansion is performed. The default value is 50. |
| | nest=<i>n</i> | Indicates that a procedure call is nested. The range of procedure subject to automatic inline expansion is limited by nesting level (<i>n</i>). The default level is 1. |
| | exp=<i>routine-name</i> [<i>routine-name...</i>] | Specifies that the specified routine name is expanded inline regardless of the specification of automatic expansion. |
| | noexp=<i>routine-name</i> [<i>routine-name...</i>] | Specifies that the specified routine name is not expanded inline regardless of the specification of automatic expansion. |
| | rexp=<i>routine-name</i> [<i>routine-name...</i>] | Specifies that the specified routine name, which is called nested expansion, is expanded inline regardless of the specification of automatic expansion. |
| | expin= { <i>directory-name</i> <i>file-name</i> } [,{ <i>directory-name</i> <i>file-name</i> }] | Specifies that the routine, which is expanded inline, is in the specified file or in Fortran source program files under the specified directory. |

- A Specifies to operate the automatic precision expansion function. (The options `-e w`, `-float2` are ignored)
- db1 Specifies the precision expansion as R(4)->R(8), R(8)->R(16), C(4) ->C(8), and C(8) -> C(16).
- dbl4 Specifies the precision expansion as R(4)->R(8), and C(4)->C(8).
- dbl8 Specifies the precision expansion as R(8)->R(16), and C(8)->C(16).
- idb1 Specifies the precision expansion as R(4) -> R(8), R(8) -> R(16), C(4) -> C(8) and C(8) -> C(16).
However, a variable or a constant with kind type parameter is not applied the precision expansion.
- idbl4 Specifies the precision expansion as R(4) -> R(8) and C(4) -> C(8).
However, a variable or a constant with kind type parameter is not applied the precision expansion.
- idbl8 Specifies the precision expansion as R(8) -> R(16) and C(8) -> C(16).
However, a variable or a constant with kind type parameter is not applied the precision expansion.
- NA Specifies not to operate the automatic precision expansion function. (default)

Listing options

- -Wf" ... "
 - Because it is passed to the compile phase
- -L fmtlist *formatted list*
- -L transform *transformations*
- -L map *reference listing*
- -L summary *detailed options used*
- Combine them!: -L fmtlist transform ...

Empowered by Innovation

NEC

Optimization options F90

Sometimes -C hopt leads to problems. Then go back to -C vopt or

- try to identify the routine and
- switch off specific optimizations for this routine:

With -C hopt (default):

- i inline intrinsic functions (default!)
- O nodiv Division may not be changed (rec. multiplication)
- O nomove Disable automatic movement of invariants outside of loop
- O nounroll Disable automatic unrolling
- ...
- ...
- ...

OpenMP - Linkage with Fortran

```
f90 -Popenmp -c a.f
```

```
c++ -Popenmp -c b.cc
```

```
c++ -Popenmp -f90lib a.o b.o
```

```
f90 -Popenmp -c -ew a.f
```

```
c++ -Popenmp -c b.cc
```

```
c++ -Popenmp -f90libew a.o b.o
```

Empowered by Innovation

NEC

OpenMP - Runtime

Use Following Runtime Environment Variables

```
#!/bin/ksh
```

```
export OMP_NUM_THREADS=8
```

```
export F_PROGINF=DETAIL
```

```
openmp.x
```

Empowered by Innovation

NEC

ftrace and parallel execution

- single tasking:

| PROG.UNIT | FREQUENCY | EXCLUSIVE TIME[sec](%) | AVER.TIME [msec] | MOPS | MFLOPS | V.OP RATIO | AVER. V.LEN | I-CACHE MISS | O-CACHE MISS | BANK CONF |
|-----------|-----------|-----------------------------|---------------------|--------|--------|---------------|----------------|-----------------|-----------------|--------------|
| vdiff | 30960 | 52.098(13.5) | 1.683 | 7997.0 | 3544.1 | 99.46 | 248.8 | 0.8383 | 0.2774 | 0.4611 |
| progexp | 720 | 46.437(12.0) | 64.496 | 7341.0 | 3389.2 | 97.90 | 121.0 | 2.1296 | 0.1366 | 0.0000 |
| ... | | | | | | | | | | |

- multi 4 processors:

| PROG.UNIT | FREQUENCY | EXCLUSIVE TIME[sec](%) | AVER.TIME [msec] | MOPS | MFLOPS | V.OP RATIO | AVER. V.LEN | I-CACHE MISS | O-CACHE MISS | BANK CONF |
|-----------|-----------|-----------------------------|---------------------|--------|--------|---------------|----------------|-----------------|-----------------|--------------|
| vdiff | 30960 | 54.823(12.9) | 1.771 | 7600.4 | 3367.9 | 99.46 | 248.8 | 0.9858 | 0.3081 | 0.4892 |
| -micro1 | 7952 | 14.175(3.3) | 1.783 | 7577.9 | 3358.2 | 99.46 | 249.7 | 0.2498 | 0.0765 | 0.1239 |
| -micro2 | 7806 | 13.841(3.3) | 1.773 | 7583.0 | 3360.1 | 99.46 | 248.5 | 0.2478 | 0.0775 | 0.1232 |
| -micro3 | 7786 | 13.771(3.2) | 1.769 | 7588.5 | 3362.3 | 99.46 | 248.1 | 0.2506 | 0.0788 | 0.1231 |
| -micro4 | 7416 | 13.036(3.1) | 1.758 | 7656.1 | 3392.6 | 99.46 | 248.8 | 0.2377 | 0.0753 | 0.1191 |
| progexp | 2880 | 50.762(11.9) | 17.626 | 6818.4 | 3146.6 | 97.79 | 121.1 | 2.6227 | 0.6504 | 0.0000 |
| -micro1 | 720 | 12.650(3.0) | 17.570 | 6877.1 | 3173.1 | 97.79 | 121.1 | 0.6162 | 0.0409 | 0.0000 |
| -micro2 | 720 | 12.753(3.0) | 17.713 | 6762.8 | 3121.0 | 97.79 | 121.1 | 0.6526 | 0.3125 | 0.0000 |
| -micro3 | 720 | 12.656(3.0) | 17.578 | 6816.6 | 3145.8 | 97.79 | 121.1 | 0.6735 | 0.1455 | 0.0000 |
| -micro4 | 720 | 12.702(3.0) | 17.642 | 6817.7 | 3146.6 | 97.79 | 121.1 | 0.6805 | 0.1514 | 0.0000 |
| ... | | | | | | | | | | |

Empowered by Innovation **NEC**

How to compile MPI-model

```
mpif90 mpicode.f90 -o mpicode
```

or (crosscompiler)

```
sxmpif90 mpicode.f90 -o mpicode
```

Don't link with `-lmpi` or include anything related to `mpi`, it is not necessary!

Empowered by Innovation **NEC**

How to run a MPI-model

```
mpirun -np 2 a.out (arg1) (arg2)
```

```
mpirun -np 2 host1 -np 3 host2 a.out(arg1)(arg2)
```

or

```
mpiexec -n ...
```

MPIPROGINF=DETAIL

Empowered by Innovation

NEC

How to run a MPI-model with configuration file

```
mpirun [options] { -f | -machinefile } <configuration-file-name>
```

Empowered by Innovation

NEC

How to run a MPI-model with configuration file

```
mpirun -f sx.conf
```

sx.conf:

```
-h sx5      -p 1 -e coupler.x  
-h sx5      -p 1 -e appl4.x  
-h sx5      -p 1 -e appl2.x  
-h sx5      -p 1 -e appl1.x  
-h sx5      -p 1 -e appl3.x
```

Empowered by Innovation

NEC

MPI Tracing

MPI Procedure Tracing Facility

Use `-mpitrace`
one can see the following information.

- which MPI procedure is called
- MPI process rank calling the MPI procedure
- entering and returning to/from MPI procedure

Empowered by Innovation

NEC

MPI: result of mpitrace

```
cs17 % doit 2
Starting MPI_Init...
Starting MPI_Init...
Starting MPI_Init...
Starting MPI_Init...
[0,0] Ending MPI_Init
[0,2] Ending MPI_Init
[0,1] Ending MPI_Init
[0,3] Ending MPI_Init
[0,1] Starting MPI_Comm_size...
[0,1] Ending MPI_Comm_size
[0,1] Starting MPI_Comm_rank...
[0,1] Ending MPI_Comm_rank
[0,0] Starting MPI_Comm_size...
[0,1] Starting MPI_Barrier...
[0,0] Ending MPI_Comm_size
[0,0] Starting MPI_Comm_rank...
[0,0] Ending MPI_Comm_rank
PIP: number of processors: 4
[0,0] Starting MPI_Barrier...
[0,3] Starting MPI_Comm_size...
[0,3] Ending MPI_Comm_size
[0,3] Starting MPI_Comm_rank...
[0,3] Ending MPI_Comm_rank
[0,2] Starting MPI_Comm_size...
[0,3] Starting MPI_Barrier...
[0,2] Ending MPI_Comm_size
[0,2] Starting MPI_Comm_rank...
[0,2] Ending MPI_Comm_rank
[0,2] Starting MPI_Barrier...
[0,1] Ending MPI_Barrier
```

NEC

MPI Communication Information

Compile with

-mpiprof

Use Environment Variable

MPICOMMINF=(NO/YES/ALL)

Empowered by Innovation

NEC

MPI Communication Information:

```

-----
Real MPI Idle Time (sec) :      0.006 [0,0]      0.009 [0,1]      0.007
User MPI Idle Time (sec) :      0.006 [0,0]      0.008 [0,1]      0.007
Total real MPI Time (sec) :      0.047 [0,0]      0.068 [0,1]      0.058
Send      count      :      9 [0,0]      9 [0,0]      9
Recv      count      :      9 [0,0]      9 [0,0]      9
Barrier   count      :      14 [0,0]      14 [0,0]      14
Bcast     count      :      1 [0,0]      1 [0,0]      1
Reduce    count      :      0 [0,0]      0 [0,0]      0
Allreduce count      :      0 [0,0]      0 [0,0]      0
Scan      count      :      0 [0,0]      0 [0,0]      0
Exscan    count      :      0 [0,0]      0 [0,0]      0
Redscat   count      :      0 [0,0]      0 [0,0]      0
Gather    count      :      0 [0,0]      0 [0,0]      0
Gatherv   count      :      0 [0,0]      0 [0,0]      0
Allgather count      :      0 [0,0]      0 [0,0]      0
Allgather count      :      0 [0,0]      0 [0,0]      0
Scatter   count      :      0 [0,0]      0 [0,0]      0
Scatterv  count      :      0 [0,0]      0 [0,0]      0
Alltoall  count      :      0 [0,0]      0 [0,0]      0
Alltoallv count      :      0 [0,0]      0 [0,0]      0
Alltoallw count      :      0 [0,0]      0 [0,0]      0
Number of bytes sent :      4088 [0,0]      4088 [0,0]      4088
Number of bytes rcv  :      4088 [0,0]      4088 [0,0]      4088
Put        count      :      0 [0,0]      0 [0,0]      0
Get        count      :      0 [0,0]      0 [0,0]      0
Accumulate count      :      0 [0,0]      0 [0,0]      0
Number of bytes put  :      0 [0,0]      0 [0,0]      0
Number of bytes got  :      0 [0,0]      0 [0,0]      0
Number of bytes accum :      0 [0,0]      0 [0,0]      0

```

[0,0] Ending MPI_Finalize

- S Generate Assembly listing

```

ident "FORTRAN90/SX Rev 290"
ident "'-S'"
ident "quiz1.f90"
# Data section ordering define
data
align 16
# ===== Begin MAIN__ =====
# === BSS SECTION ===
# == UNINIT LOCAL AREA ==
bss  _1.MAIN__.RTP.INTF_AREAPACKET, 0x0000001c0, 16
bss  _1.MAIN__.LOCALSTATIC.c, 0x000000190, 8
bss  _1.MAIN__.LOCALSTATIC.b, 0x000000190, 8
bss  _1.MAIN__.LOCALSTATIC.a, 0x000000190, 8
set  .2.1.2spill_p, 0x000000130
# === TEXT SECTION ===
# == LINKAGE AREA ==
text
align 128
.3.H.LK.MAIN_:
using .3.H.LK.MAIN_,$s4
.2.1.2H.LK.p.1:
llong 0xffffffffffffffff
long  .3.H.CD.MAIN_- .3.H.LK.MAIN_
long  25
llong 0x0
llong .3.H.TB.MAIN_
rlong 7,0

```

F90 Compiler Directives

Repeat the most important directives

F90 Compiler Directives Collapse

```
do i=1,n  
  do j=1,m  
    a(i,j)=b(i,j)+c(i,j)*x  
  end do  
end do
```

Tuned

```
do i=1,n*m  
  a(i,1)=b(i,1)+c(i,1)*x  
end do
```

F90 Compiler Directives

-Wf,-L fmtlist summary map transform

```
!CDIR COLLAPSE
W  _____ do i=1,n
  | _____ do j=1,m
  |           a(i,j)=b(i,j)+c(i,j)*x
  | _____ end do
W  _____ end do
```

Compiler directives F90: LOOPCHG

Example: $n \gg m$

```
_____ do i=1,n
v _____ do j=1,m
  |           a(i,j) = b(i,j)+m(j)
  | _____ end do
v _____ end do
```

Compiler directives F90: LOOPCHG

```
!CDIR LOOPCHG
X  _____ do i=1,n
  | _____ do j=1,m
  | _____ a(i,j) = b(i,j)+m(j)
  | _____ end do
X  _____ end do
```

But: Look for bank conflicts in PROGINF,
if you do it manually

Compiler directives F90: UNROLL

```
_____ do j=1,100
  | _____ ... any stuff which prevents vectorization
  | v _____ do i=1,n
  | | _____ a(i) = a(i) + b(i,k) * c(k,j)
  | | _____ end do
  | _____ end do
  _____ end do
```

Unrolling an outer loop that is not vectorized is generally effective,
especially if it reduces the number of references to a memory area

Compiler directives F90: UNROLL

Example: Unroll by 2

```
!CDIR UNROLL=2
  _____ do j=1,99,2
  |_____ do i=1,n
  |_____ a(i) = a(i) + b(i,k) * c(k,j) + b(i,k)*c(k,j+1)
  |_____ end do
  |_____ end do
  _____ end do
```

The load count and store count of $a(i)$ and the load count of $b(i,k)$ is halved by unrolling

Compiler directives F90: NODEP

```
+ _____ DO J=1,N
|_____ X(J-1)=X(JW)*Y(J)
|_____ JW=JW+1
S _____ END DO
```

Investigate the values of JW
e.g with print the values in a list and look for duplicate indices

Optimization options F90

Please refer to the NEC Manual

FORTRAN90/SX - Programmers Guide
Chapter 10 contains very useful informations

Empowered by Innovation

NEC

F90-Compiler Runtime Environment Variables

```
F_FILEINF=(YES/DETAIL/NO)  
F_PROGINF=(YES/DETAIL/NO)  
F_FTRACE=(YES/NO)  
F_EXPRCW=unitnumber (>2 GB unform. output)*  
F_FMTBUF=num (num bytes, 32 kbytes)  
F_SETBUF=n kbytes (128 kbytes)
```

*Record Control Word is 8 byte storage
Additional Env. Var. in your handbook

Empowered by Innovation

NEC

COMPILER:

Investigation for correctness

Investigation of Correctness

- f90 -Cdebug
- f90 -eC (array bounds checking)
- f90 -eC -Cvsafe
- f90 -Pstack (default is -Pstatic)
- f90 -Pstack -Wf"-init stack=nan heap=nan"
- sxc++ -init stack=nan
- sxc++ -g

Code doesn't run (well?): f90 -Cdebug

- often already helps
 - switches off all optimisations
 - if it works: *not necessarily a compiler bug!*
- No core dump?
 - Nothing is worse than a code that runs but produces wrong results!
 - small test case!
 - 'trusted' reference output

Empowered by Innovation

NEC

Out of bounds references?

- f90 -eC
 - 'array bounds checking' enabled
- f90 -eC -Cvsafe
 - because otherwise:

```
real a(n,m)
...
do j = 1, m
  do i = 1, n
    a(i,j) = 0.0
  end do
end do
```



```
real a(n,m)
...
do i = 1, n*m
  a(i,1) = 0.0
end do
...
```

Empowered by Innovation

NEC

Out of bounds references?

```
cs24 % a.out
* 240 Subscript error  array=a size=100 subscript=101 eln=4
PROG=ec_test E
LN=4(400000908)
cs24 %
```

Variables initialised? Second example

- Typo on common block

```
common / .. / retv, valeu
real a(n,m)
do j = 1, m
  do i = 1, n
    retv = retv + a(i,j)
  end do
end do
retv = max( retv, value )
value = retv
```

- Use implicit none
- FORTRAN include files!

Variables initialised? The ultimate test

- f90 -Pstack -Wf"-init stack=nan heap=nan"
 - everything is initialised with the IEEE NaN
 - especially every local variable and array on call to a subroutine!
- Some overhead!
 - Do not use this for production
- for common blocks and other static data:
 - add to above: -Wl"-f nan"

C++ COMPILER:

*The most
important
switches*

C++ COMPILER:

***Replaces also the
C-Compiler !***

Empowered by Innovation

NEC

C++ Compiler

C according to ANSI/ISO 9899:1990[1992], ISO/IEC 9899:1992 und ISO/IEC 9899:1990/Amd.1:1995 are supported,

C++ Compiler is compliant with ISO/IEC 14882-1998 and ISO/IEC 9899-1999

Empowered by Innovation

NEC

C++-Compiler

2.2.3 Inlining Options

```
-dir { inline | noinline }  
-pi { auto | noauto }  
  { inline | noinline }  
max_depth=n (default: max_depth=2)  
max_size=n (default: max_size=50)  
Table 2-4 Inlining Options
```

| Option | Suboption | Description | Default |
|--------|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|----------|
| -dir | [no]inline | Inlining directive options are [not] validated. | noinline |
| -pi | | Enables inlining functions. | |
| | [no]auto | Enable[Disable] automatic inlining. | noauto |
| | [no]inline | inline specifiers in a C++ source program are [not] effective. This option is effective in the C++ mode. | inline |
| | max_depth= <i>n</i> | Specifies the level of functions to be inlined automatically from the bottom of the calling. This suboption is effective when -pi,auto is specified. | 2 |
| | max_size= <i>n</i> | Specifies the size of intermediate representation of functions to be inlined automatically. This suboption is effective when -pi,auto is specified. | 50 |

C++-Compiler

[Chapter 2 Compiler Options](#)

- [2.1 Option Category](#)
- [2.2 Options](#)
 - [2.2.1 Overall Options](#)
 - [2.2.2 Vector/Scalar Optimization Options](#)
 - [2.2.3 Inlining Options](#)
 - [2.2.4 Parallelization Options](#)
 - [2.2.5 Code Generation Options](#)
 - [2.2.6 Language Options](#)
 - [2.2.7 Performance Measurement Options](#)
 - [2.2.8 Debug Options](#)
 - [2.2.9 Preprocessor Options](#)
 - [2.2.10 Message Options](#)
 - [2.2.11 Linker Options](#)
 - [2.2.12 Directory Options](#)
- [2.3 Predefined Macros](#)



C++-Compiler

2.2.6 Language Options

```
{ -Xp | -Xa | -Xc | -Xkr }  
-K { const_string_literals | noconst_string_literals }  
{ exceptions | noexceptions }  
{ new_for_init | old_for_init }  
{ restrict | norestrict }  
{ rtti | nortti }  
{ unsigned_char | signed_char }  
{ using_std | nousing_std }  
-T { auto | noauto }  
{ none | all | used | local }  
implicit_include
```

Table 2-7 Language Options

| Option | Description | Default |
|--------|---------------------------------------------------------------------------|---------|
| -Xp | C++ mode. Enables compilation and linking of C++. | default |
| -Xa | ANSI C mode. Enables compilation and linking of C rather than C++. | - |
| -Xc | Strict ANSI C mode. Enables compilation and linking of C rather than C++. | - |
| -Xkr | K&R C mode. Enables compilation and linking of C rather than C++. | - |

Empowered by Innovation

NEC

Important switches C++

```
-c          compile only  
-f90lib    Link with FORTRAN90/SX runtime libraries  
-math      Use mathematical intrinsic functions  
           vector Use vector versions  
           scalar Use scalar versions  
-over2g    Enable structures/arrays/classes/unions > 2GB  
-restrict=  Enable optimization and vectorization of loops  
           with pointers and pointer references  
           all
```

Empowered by Innovation

NEC

C++-Compiler

| Option | Suboption | Description | Default |
|--------|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|----------|
| -dir | [no]inline | Inlining directive options are [not] validated. | noinline |
| -pi | | Enables inlining functions. | |
| | [no]auto | Enable[Disable] automatic inlining. | noauto |
| | [no]inline | inline specifiers in a C++ source program are [not] effective. This option is effective in the C++ mode. | inline |
| | max_depth= <i>n</i> | Specifies the level of functions to be inlined automatically from the bottom of the calling. This suboption is effective when -pi,auto is specified. | 2 |
| | max_size= <i>n</i> | Specifies the size of intermediate representation of functions to be inlined automatically. This suboption is effective when -pi,auto is specified. | 50 |

Empowered by Innovation

NEC

C++-Compiler Vector Directives

almost similar to F90

```
#pragma cdir vector_directive
```

```
#pragma cdir nodep
```

Empowered by Innovation

NEC

C++ Compiler

Q: How are Fortran function names in the calling convention?

A: Simply write them lowercase and append an underscore.

Example:

```
PROGRAM TEST  
CALL CFUNC(5)  
END  
  
void cfunc_(int *a) {  
}
```

Empowered by Innovation

NEC

C++ Compiler

Q: How to link when C and Fortran are mixed?

A: Link with sxf90. It makes it right.

```
sxf90 fortran.o c++.o -o a.out
```

When using C++ as linker use:

```
sxc++ fortran.o c++.o -f90lib -o a.out
```

Empowered by Innovation

NEC

C++ Compiler, Porting Pitfalls

Q: I'm using malloc(), and my code dumps core.

A: Make sure stdlib.h is included where malloc() is used.

C++-Compiler

Chapter 2 Compiler Options

- 2.1 Option Category
- 2.2 Options
 - 2.2.1 Overall Options
 - 2.2.2 Vector/Scalar Optimization Options
 - 2.2.3 Inlining Options
 - 2.2.4 Parallelization Options
 - 2.2.5 Code Generation Options
 - 2.2.6 Language Options
 - 2.2.7 Performance Measurement Options
 - 2.2.8 Debug Options
 - 2.2.9 Preprocessor Options
 - 2.2.10 Message Options
 - 2.2.11 Linker Options
 - 2.2.12 Directory Options
- 2.3 Predefined Macros

C++-Compiler

Chapter 3 Compiler Directives

- 3.1 Overview
- 3.2 Format
- 3.3 Scope
- 3.4 Relation with Compiler Option
- 3.5 Option Summary
 - 3.5.1 Optimization Directive Options
 - 3.5.2 Vectorization Directive Options
 - 3.5.3 Inlining Directive Options
 - 3.5.4 Parallelization Directive Options
 - 3.5.5 Parallelization Control Options

Empowered by Innovation

NEC

C++-Compiler

Chapter 4 Optimization Features

- 4.1 Code Optimization
 - 4.1.1 Common Expression Elimination
 - 4.1.2 Code Motion
 - 4.1.3 Simple Assignment Elimination
 - 4.1.4 Deletion of Unnecessary Codes
 - 4.1.5 Converting Division to Equivalent Multiplication
 - 4.1.6 Other Optimizations
- 4.2 Notes on the Optimization Function
 - 4.2.1 Side Effects of Optimization
 - 4.2.2 Slow-Down Resulting from Optimization
 - 4.2.3 Coding that Affects Optimization
 - 4.2.3.1 Expressions
 - 4.2.3.2 Subscript Expressions
 - 4.2.3.3 Number of Variable Names
 - 4.2.3.4 External Functions as Invariant Expressions
 - 4.2.4 Options that Affect Optimization
 - 4.2.4.1 Compiler Options that Affect Optimization
 - 4.2.4.2 Recognition of Loops

EC

C++-Compiler

Chapter 5 Vectorization Functions

Chapter 6 Inlining

Chapter 7 Multitasking

Empowered by Innovation

NEC

CPP-Options

f90 -Ep Invoke CPP to perform preprocessing

f90 -EP Invoke CPP to perform preprocessing and save
intermediate file

Empowered by Innovation

NEC

CPP-Predefined Variables

-Uname

Unname the previously defined reserve symbol of the preprocessor. The current reserve symbols in SUPER-UX are "unix", "_FLOAT0", "_FLOAT1", "_FLOAT2", and "SX".

-Dname

-Dname=def

Define name with value def as if by a #define. If no =def is given, name is defined with the value 1. The -D option has lower precedence than the -U option. That is, if the same name is used in both a -U option and a -D option, the name will be undefined regardless of the order of the options. Characters from supplementary code sets can be used in def.

Empowered by Innovation

NEC

CPP-Predefined Variables, Example 1

```
program cpp_test
dimension SX(20),temp(20)
SX=10
do i=1,20
temp(i)=SX(i)
enddo
print*, temp(10)
end
```

```
cs24 % f90 -EP cpp_test.F
./i.cpp_test.F:
```

```
f90: error(106): i.cpp_test.F, line 3: Syntax error in assignment statement.
f90: error(110): i.cpp_test.F, line 5: Extra text follows the end of statement.
f90: warn(82): i.cpp_test.F, line 8: Name "sx" is not used.
f90: warn(81): i.cpp_test.F, line 8: Name "temp" is referenced but not defined.
f90: i.cpp_test.F, cpp_test: There are 2 errors and 2 warnings.
f90 fatal : /usr/lib/f90com command error : 1
cs24 %
```

CPP-Predefined Variables, Example 1

```
program cpp_test
  dimension SX(20),temp(20)
  SX=10
  do i=1,20
    temp(i)=SX(i)
  enddo
  print*, temp(10)
end
```

```
cs24 % f90 -EP -USX cpp_test.F
./i.cpp_test.F:
```

```
f90: vec(4): i.cpp_test.F, line 3: Vectorized array expression.
f90: vec(1): i.cpp_test.F, line 4: Vectorized loop.
f90: i.cpp_test.F, cpp_test: There are 2 diagnoses.
cs24 %
```

Empowered by Innovation **NEC**

CPP-Predefined Variables, Example 2

```
program cpp_test
  dimension sx(20),temp(20)
  sx=10
  do i=1,20
    temp(i)=sx(i)
  enddo
  #if defined(SX)
    print*, 'This is an SX'
  #else
    print*, 'This is no SX'
  #endif
end
```

sxf90 -EP cpp_test.F gives:

```
program cpp_test
  dimension sx(20),temp(20)
  sx=10
  do i=1,20
    temp(i)=sx(i)
  enddo

  print*, 'This is an SX'

end
```

C++-Compiler Runtime Environment Variables

`C_FILEINF= (YES/DETAIL/NO)`

`C_FTRACE= (NO/YES)`

`C_PROGINF=(NO/YES/DETAIL)`

Empowered by Innovation

NEC

Debugging:

- xdbx, X-based dbx
- don't forget to set DISPLAY
- refer to manual
- compile and link with `-C debug`
- very similar to other vendors' debugger

Empowered by Innovation

NEC

Debugging:

For programs coded in C

```
% cc -g myprog.c -o myprog
```

For programs coded in FORTRAN

```
% f90 -Cdebug myprog.f -o myprog
```

```
% dbx myprog  
dbx version X.XX of MM/DD/YY HH:MM.  
Type 'help' for help.  
reading symbolic information  
(dbx)
```

Empowered by Innovation

NEC

Debugging:

```
(dbx) stop at 20  
[1] stop at "myprog.":20  
(dbx) stop at 25  
[2] stop at "myprog.":25  
(dbx)
```

Setting stops

```
(dbx) run  
[1] stopped in MAIN_ at line 20 in file "myprog.f"*  
20 INT4=3  
21 (dbx)
```

```
(dbx) list 17, 20  
17 IF (INT1 .LT. 0) THEN  
18 INT4=0  
19 ELSE  
20 INT4=3  
(dbx) print int1
```

powered by Innovation

NEC

Debugging:

(dbx) cont

[2] stopped in MAIN_ at line 25 in file "myprog.f"

* 25 INT4=INT4*2

(dbx)

(dbx) quit

NQS II

```
#PBS -S /bin/ksh
#PBS -l cpunum_prc=8          # 8 cpus per node
#PBS -l cputim_job=10:00:00  # 10 h cputime per node
#PBS -l memsz_job=6gb       # 6 GB Memory per node
#PBS -T mpisx
#PBS -b 2                    # job runs on 2 nodes
#PBS -j o                    # join err and out to out
#PBS -M tschoenemeyer@hpce.nec.com # you should always specify your email
                                # address for error messages etc
#PBS -N mm5mpp              # job name
```

NQS

```
#!/bin/ksh
#
#@$-s /bin/ksh
#@$-lM 4200MB
#@$-lT 1:30:00
#@$-c 4
#@$-eo
#@$-r echmf4cpu
#@$-x
```

Empowered by Innovation

NEC

Linking MPI Programs with Vampirtrace/SX

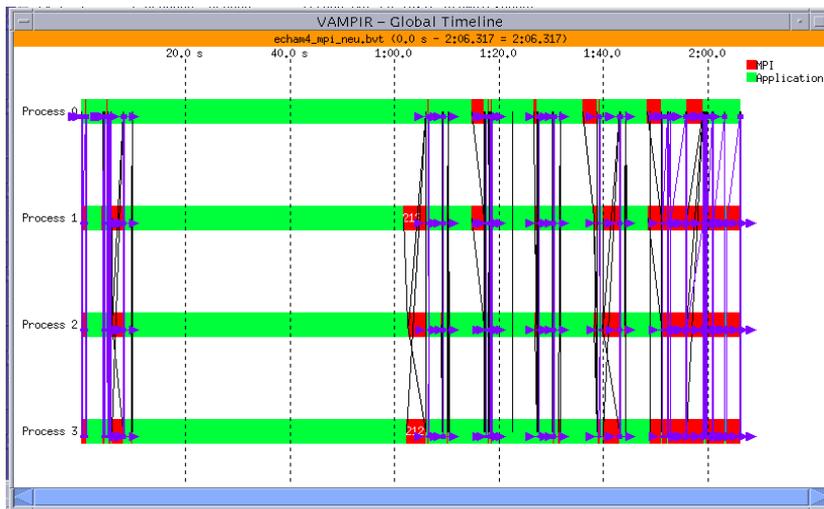
```
sam% sxmpif90 -vampir mpi_1.f
mpi_1.f:
sam%
```

```
cs1 % mpirun -np 2 a.out
number of processors: 2
MPI_comm_rank( 1 ): born to be alive
number of processors: 2
MPI_comm_rank( 0 ): born to be alive
Writing tracefile a.out.bvt
cs1 %
```

Empowered by Innovation

NEC

Global Timeline for all processes of ECHAM 4.6 (4 processor run): Time in seconds



End