

Anleitung für zwei Fortran-Openmp-Beispiele auf der NWZSuperdome

(Timo Heinrich, t_hein03@uni-muenster.de)

Inhaltsverzeichnis:

0. Einleitung

1. Teil: Helloworldprogramm

1.1 Quellcode: Helloworld.f90

1.2 Einloggen auf der Superdome

1.3 Kompilieren des Fortranprogramms mit Openmp

1.4 Erstellen eines PBS-Skripts zur Ausführung des Jobs

1.5 Starten des Jobs auf der Superdome

2. Teil: parallelisierte Schleife

2.1 Quellcode: Schleife.f90

2.2 PBS-Datei zum Starten des Jobs

2.3 Starten des Jobs

0. Einleitung:

In dieser Beschreibung wird an zwei einfachen Beispielen der Quellcode eines Openmp-Programms, programmiert in Fortran, erläutert und explizit mit einigen Screenshots die Ausführung auf der NWZSuperdome (dem Shared-Memory-Rechner der IVV4) demonstriert. Alle Informationen findet man auch auf der Seite <http://www.uni-muenster.de/IVVNWZ/computing/superdome/index.html>

Ziel dieses Dokuments ist alle notwendigen Schritte für die Ausführung eines mit Openmp parallelisierten Programms zu erlernen.

1. Teil: Helloworldprogramm:

1.1 Quellcode: Helloworld.f90

Zunächst muss der Quellcode des auszuführenden Programms erstellt werden. Hierzu kopiert man einfach den folgenden Programmcode in einen Editor (z.B. Proton, Emacs,...) und speichert die Datei in einem Ordner auf dem Laufwerk I: . In diesem Artikel wird der Speicherpfad: "I:\Superdomedemonstration" verwendet.

Quellcode des Helloworld-Programms:

```
PROGRAM HelloworldOpenmp
IMPLICIT NONE

INTEGER*4 NUMTHREADS           !Anzahl der Threads
INTEGER*4 THREADID             !Nummer des Threads

CALL HELLOWORLD(NUMTHREADS,THREADID)      !siehe Subroutine

END PROGRAM

SUBROUTINE HELLOWORLD(NUMTHREADS,THREADID)
IMPLICIT NONE

INTEGER*4 NUMTHREADS, THREADID, OMP_GET_NUM_THREADS, OMP_GET_THREAD_NUM

!$OMP PARALLEL PRIVATE(NUMTHREADS, THREADID)           !Initialisierung der parallelen Umgebung

    THREADID=OMP_GET_THREAD_NUM()
    !Auslesen der Nummer des Threads mit Hilfe der function OMP_GET_THREAD_NUM()

    NUMTHREADS=OMP_GET_NUM_THREADS()
    !Auslesen der Anzahl der Threads mit Hilfe der function OMP_GET_THREAD_NUM()

    WRITE(*,*) 'Das ist Nummer', THREADID, 'von', NUMTHREADS

!$OMP END PARALLEL                                     !Beenden der parallelen Umgebung

END SUBROUTINE
```

Zur Erklärung des Quellcodes:

Das Programm öffnet in Zeile 18 in der Subroutine eine parallele Umgebung und schließt diese in Zeile 28. Innerhalb dieses Bereichs liest jeder Thread seine Nummer und die Anzahl aller Threads aus. Die Kommentare im Quellcode beschreiben das Programm noch detaillierter

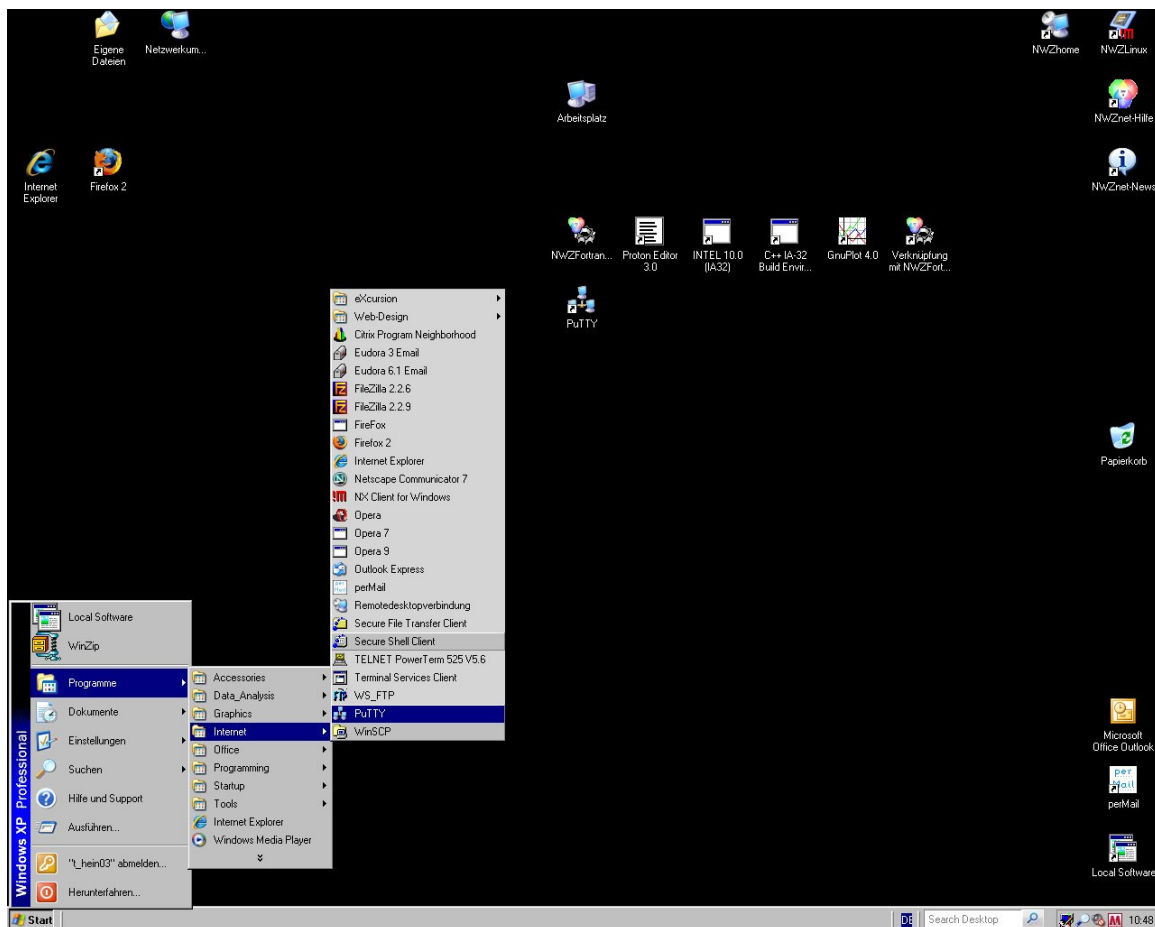
Als Ausgabe ist zu erwarten:

```
Das ist Nummer 0 von 4
Das ist Nummer 1 von 4
Das ist Nummer 2 von 4
Das ist Nummer 3 von 4
```

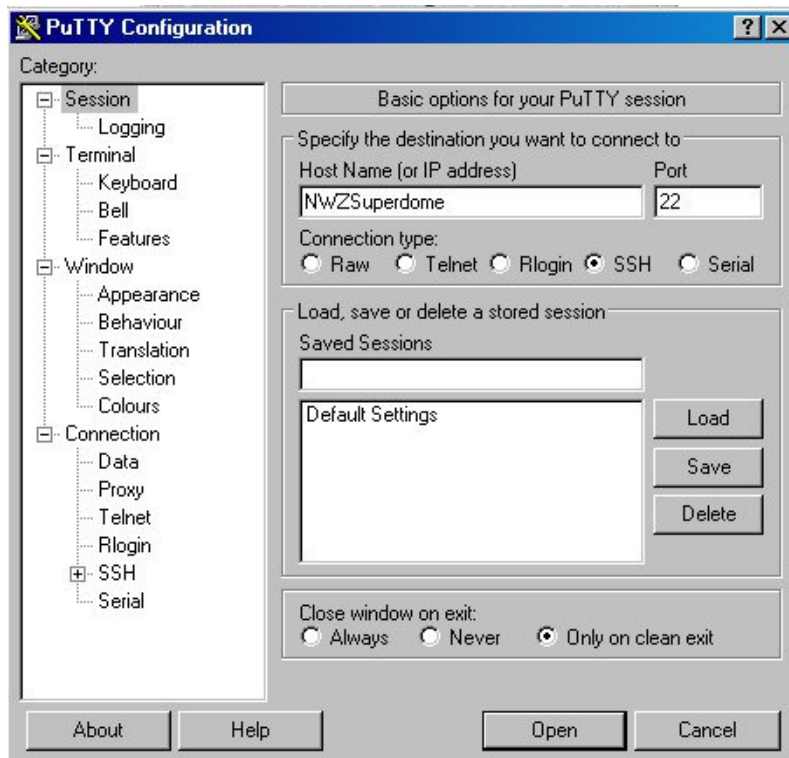
Die Nummerierung beginnt per Definition bei 0, wobei Thread Nummer 0 der Masterthread ist. Die Reihenfolge kann unter Umständen auch variieren.

1.2 Einloggen auf der NWZSuperdome

Nachdem jetzt das Programm unter I:\Superdomedemonstration\Helloworld.f90 gespeichert ist soll das Programm auf der Superdome ausgeführt werden. Deshalb muss man sich als nächstes auf der NWZSuperdome einloggen. Vorteilhaft ist, dass das I:-Laufwerk automatisch angebunden wird, so dass das gespeicherte Programm auf der Superdome direkt verfügbar ist. Auf der Superdome muss man sich mit dem Client Putty per SSH einloggen. Diesen findet man unter: **Start > Programme > Internet > Putty** :



Bei dem sich öffnenden Fenster muss man bei Hostname **NWZSuperdome** eintragen. Außerdem muss darauf geachtet werden, dass als Connection type SSH gewählt wird und der Port auf 22 eingestellt ist:



Hat man diese Einstellungen vorgenommen muss man auf *Open* klicken. Es öffnet sich die Konsole, bei der man sich noch mit seiner NWZ-Kennung und dem zugehörigen Passwort einloggen muss:



Als nächstes wechselt man in das Verzeichnis, in dem sich die Datei `Helloworld.f90` befindet. In meinem Fall geschieht dies mit `cd Superdomedemonstration`. Hierbei ist darauf zu achten, dass Linux "case sensitive" ist, das heißt Groß- und Kleinschreibung wird unterschieden.

Mit dem Befehl `ls` werden alle Dateien in diesem Ordner angezeigt, wie man sieht, befindet sich dort nur die bisher erstellte Datei `Helloworld.f90`.



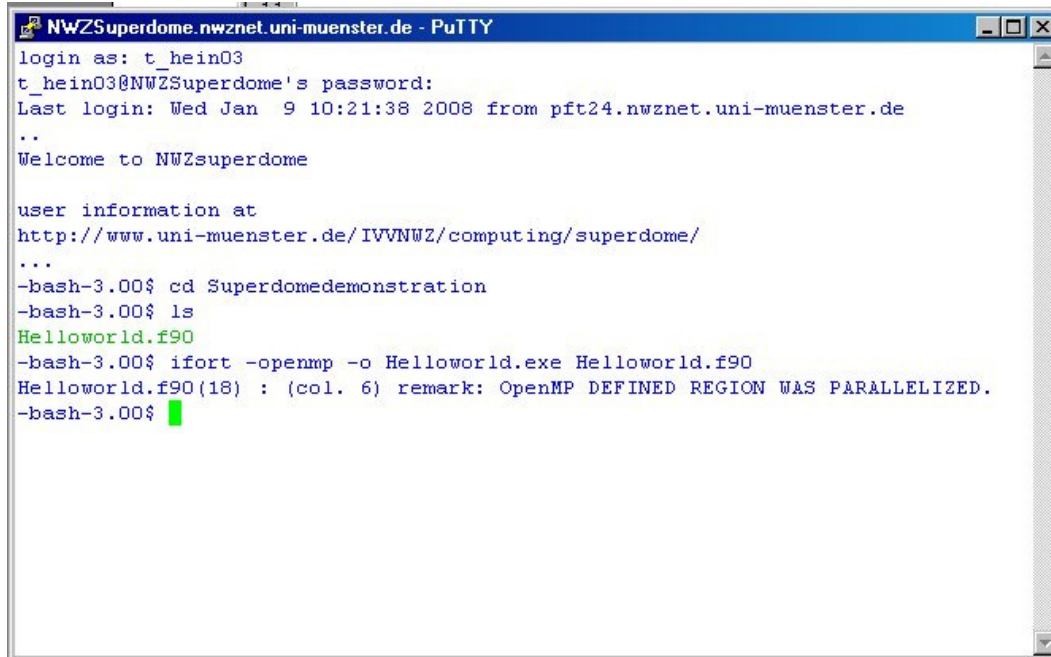
```
NWZSuperdome.nwznet.uni-muenster.de - PuTTY
login as: t_hein03
t_hein03@NWZSuperdome's password:
Last login: Wed Jan  9 10:21:38 2008 from pft24.nwznet.uni-muenster.de
..
Welcome to NWZsuperdome

user information at
http://www.uni-muenster.de/IVVNWZ/computing/superdome/
...
-bash-3.00$ cd Superdomedemonstration
-bash-3.00$ ls
Helloworld.f90
-bash-3.00$ █
```

1.3 Kompilieren des Fortranprogramms mit Openmp:

Das Programm wird kompiliert durch das Kommando:

```
ifort -openmp -o Helloworld.exe Helloworld.f90
```



```
NWZSuperdome.nwznet.uni-muenster.de - PuTTY
login as: t_hein03
t_hein03@NWZSuperdome's password:
Last login: Wed Jan  9 10:21:38 2008 from pft24.nwznet.uni-muenster.de
..
Welcome to NWZsuperdome

user information at
http://www.uni-muenster.de/IVVNWZ/computing/superdome/
...
-bash-3.00$ cd Superdomedemonstration
-bash-3.00$ ls
Helloworld.f90
-bash-3.00$ ifort -openmp -o Helloworld.exe Helloworld.f90
Helloworld.f90(18) : (col. 6) remark: OpenMP DEFINED REGION WAS PARALLELIZED.
-bash-3.00$
```

Wenn das Kompilieren erfolgreich war wird angezeigt welche Region parallelisiert wurde. Die Auszuführende, hier Helloworld.exe, befindet sich dann im gleichen Ordner wie das Programm.

1.4 Erstellen eines PBS-Skripts zur Ausführung des Jobs:

Bis jetzt befinden sich das Fortranprogramm und die zugehörige Auszuführende in unserem Ordner. Um das Programm starten zu können, muss man ein PBS-Skript erstellen.

```
#PBS -q test
#PBS -N Helloworld
#PBS -e error.txt
#PBS -o output.txt
#PBS -m abe
#PBS -M abc@uni-muenster.de
#PBS -l nodes=1:ppn=4
source /opt/intel/fc/9.1.036/bin/ifortvars.sh
PARNODES=`wc -l $PBS_NODEFILE |gawk '{print $1}'`
export OMP_NUM_THREADS=$PARNODES
~/Superdomedemonstration/Helloworld.exe
```


Der Einfachheit halber kopiert man sich die oben stehenden Zeilen in einen Editor (Proton, Emacs,...) und speichert die Datei z.B. unter Helloworld.PBS im selben Ordner in dem sich ihr Fortranprogramm befindet, in diesem Fall also "I:\Superdomedemonstration".

Zur Erklärung des Skripts:

- q test - hier wird festgelegt auf welcher Partition gerechnet werden soll, in diesem Fall auf der NWZSupertest-Partition
 - N Helloworld - gibt den Namen des Jobs an, der auf der Superdome gestartet wird. Dieser Name dient nur der Übersicht und hat sonst keine weitere Bedeutung
 - e error.txt - Die Fehlerausgabe wird in die Datei error.txt umgeleitet
 - o output.txt - Die Standard(Bildschirm-)ausgabe wird in die Datei output.txt umgeleitet
 - m abe - Bei Start, Ende oder Abbruch des Jobs wird eine e-mail ...
 - M abc@uni-muenster.de - ... an die hier stehende Adresse versandt
 - l nodes=1:ppn=4 - hier wird die Anzahl der verwendeten Prozessoren festgelegt nodes muss immer auf 1 gesetzt werden, ppn ist die Anzahl der Prozessoren, die je nach Partition frei gewählt werden kann
- source /opt/intel/icc/9.1.036/bin/ifortvars.sh – muss durchgeführt werden, damit dynamische Bibliotheken eingebunden werden
- PARNODES=`wc -l \$PBS_NODEFILE |gawk '{print \$1}'`
export OMP_NUM_THREADS=\$PARNODES - liest die Anzahl der Prozessoren aus und setzt die Anzahl der Threads auf diesen Wert
- ~/Superdomedemonstration/Helloworld.exe - startet das Programm

1.5 Starten des Jobs auf der Superdome:

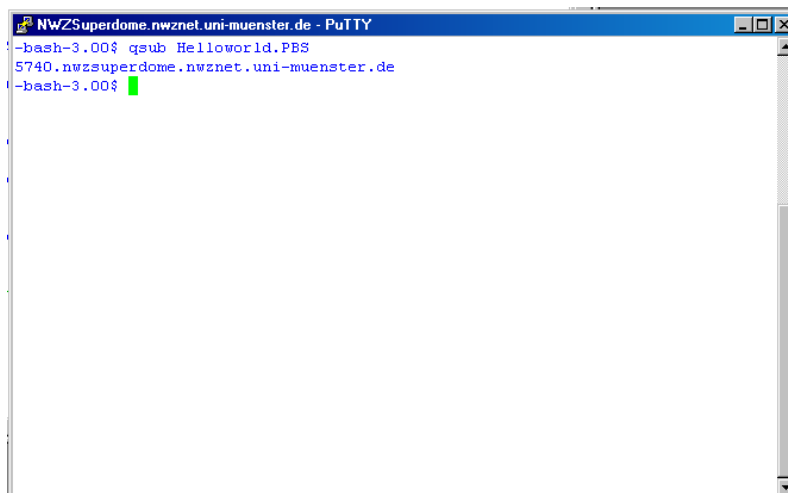
Zunächst vergewissern wir uns, dass sich die Auszuführende (HelloWorld.exe) und das PBS-Skript im gleichen Ordner befinden:



```
NWZSuperdome.nwznet.uni-muenster.de - PuTTY
login as: t_hein03
t_hein03@NWZSuperdome's password:
Last login: Wed Jan  9 10:21:38 2008 from pft24.nwznet.uni-muenster.de
..
Welcome to NWZsuperdome

user information at
http://www.uni-muenster.de/IVVNWZ/computing/superdome/
...
-bash-3.00$ cd Superdomedemonstration
-bash-3.00$ ls
HelloWorld.f90
-bash-3.00$ ifort -openmp -o HelloWorld.exe HelloWorld.f90
HelloWorld.f90(18) : (col. 6) remark: OpenMP DEFINED REGION WAS PARALLELIZED.
-bash-3.00$ ls
HelloWorld.exe HelloWorld.f90 HelloWorld.PBS
-bash-3.00$
```

Um den Job zu starten, was implizit in unserem PBS-Skript geschieht, muss das PBS-Skript ausgeführt werden. Dies geschieht mit `qsub` und dem Namen der PBS-Datei, also: `qsub HelloWorld.PBS`
Danach wird dem Job eine Kennung zugewiesen.



```
NWZSuperdome.nwznet.uni-muenster.de - PuTTY
-bash-3.00$ qsub HelloWorld.PBS
5740.nwzsuperdome.nwznet.uni-muenster.de
-bash-3.00$
```


Der Status des Jobs kann mit dem Befehl qstat eingesehen werden.

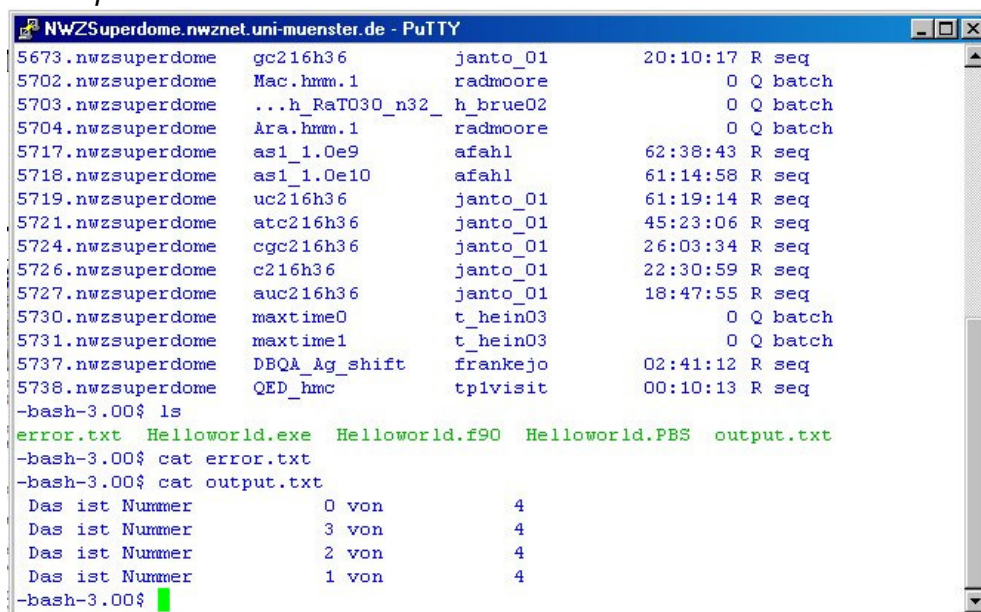
```
NWZSuperdome.nwznet.uni-muenster.de - PuTTY
5740.nwzsuperdome.nwznet.uni-muenster.de
-bash-3.00$ qstat
Job id          Name          User          Time Use S Queue
-----
5592.nwzsuperdome Mac.hmm.2     radmoore      56:36:56 R batch
5643.nwzsuperdome ac216h36      janto_01     20:10:18 R seq
5671.nwzsuperdome tc216h36      janto_01     20:10:17 R seq
5672.nwzsuperdome cc216h36      janto_01     20:10:17 R seq
5673.nwzsuperdome gc216h36      janto_01     20:10:17 R seq
5702.nwzsuperdome Mac.hmm.1     radmoore      0 Q batch
5703.nwzsuperdome ...h_RaT030_n32_h_brue02 0 Q batch
5704.nwzsuperdome Ara.hmm.1     radmoore      0 Q batch
5717.nwzsuperdome as1_1.0e9     afahl         62:38:43 R seq
5718.nwzsuperdome as1_1.0e10    afahl         61:14:58 R seq
5719.nwzsuperdome uc216h36      janto_01     61:19:14 R seq
5721.nwzsuperdome atc216h36     janto_01     45:23:06 R seq
5724.nwzsuperdome cgc216h36     janto_01     26:03:34 R seq
5726.nwzsuperdome c216h36       janto_01     22:30:59 R seq
5727.nwzsuperdome auc216h36     janto_01     18:47:55 R seq
5730.nwzsuperdome maxtime0      t_hein03     0 Q batch
5731.nwzsuperdome maxtime1      t_hein03     0 Q batch
5737.nwzsuperdome DBQA_Ag_shift frankejo      02:41:12 R seq
5738.nwzsuperdome QED_hmc       tp1visit     00:10:13 R seq
-bash-3.00$
```

Der Job taucht hier allerdings nicht auf. Dies liegt daran, dass die Rechenzeit des Programms sehr klein ist, so dass das der Job bereits beendet wurde. Je nach Länge des Programms ergeben sich natürlich größere Rechenzeiten. Außerdem können Wartezeiten für den Job auftreten, wenn die Superdome gerade stark benutzt wird.

Nachdem der Job berechnet wurde findet man die error.txt und output.txt im selben Ordner wieder:

```
NWZSuperdome.nwznet.uni-muenster.de - PuTTY
Job id          Name          User          Time Use S Queue
-----
5592.nwzsuperdome Mac.hmm.2     radmoore      56:36:56 R batch
5643.nwzsuperdome ac216h36      janto_01     20:10:18 R seq
5671.nwzsuperdome tc216h36      janto_01     20:10:17 R seq
5672.nwzsuperdome cc216h36      janto_01     20:10:17 R seq
5673.nwzsuperdome gc216h36      janto_01     20:10:17 R seq
5702.nwzsuperdome Mac.hmm.1     radmoore      0 Q batch
5703.nwzsuperdome ...h_RaT030_n32_h_brue02 0 Q batch
5704.nwzsuperdome Ara.hmm.1     radmoore      0 Q batch
5717.nwzsuperdome as1_1.0e9     afahl         62:38:43 R seq
5718.nwzsuperdome as1_1.0e10    afahl         61:14:58 R seq
5719.nwzsuperdome uc216h36      janto_01     61:19:14 R seq
5721.nwzsuperdome atc216h36     janto_01     45:23:06 R seq
5724.nwzsuperdome cgc216h36     janto_01     26:03:34 R seq
5726.nwzsuperdome c216h36       janto_01     22:30:59 R seq
5727.nwzsuperdome auc216h36     janto_01     18:47:55 R seq
5730.nwzsuperdome maxtime0      t_hein03     0 Q batch
5731.nwzsuperdome maxtime1      t_hein03     0 Q batch
5737.nwzsuperdome DBQA_Ag_shift frankejo      02:41:12 R seq
5738.nwzsuperdome QED_hmc       tp1visit     00:10:13 R seq
-bash-3.00$ ls
error.txt Helloworld.exe Helloworld.f90 Helloworld.PBS output.txt
-bash-3.00$
```

Die Ausgabe bzw. die Fehlerausgabe kann man sich zum Beispiel mit dem Befehl `cat output.txt` bzw. `cat error.txt` ansehen:



```
NWZSuperdome.nwznet.uni-muenster.de - PuTTY
5673.nwzsuperdome gc216h36 janto_01 20:10:17 R seq
5702.nwzsuperdome Mac_hmm.1 radmoore 0 Q batch
5703.nwzsuperdome ...h_RaT030_n32_h_brue02 0 Q batch
5704.nwzsuperdome Ara_hmm.1 radmoore 0 Q batch
5717.nwzsuperdome as1_1.0e9 afahl 62:38:43 R seq
5718.nwzsuperdome as1_1.0e10 afahl 61:14:58 R seq
5719.nwzsuperdome uc216h36 janto_01 61:19:14 R seq
5721.nwzsuperdome atc216h36 janto_01 45:23:06 R seq
5724.nwzsuperdome cgc216h36 janto_01 26:03:34 R seq
5726.nwzsuperdome c216h36 janto_01 22:30:59 R seq
5727.nwzsuperdome auc216h36 janto_01 18:47:55 R seq
5730.nwzsuperdome maxtime0 t_hein03 0 Q batch
5731.nwzsuperdome maxtime1 t_hein03 0 Q batch
5737.nwzsuperdome DBQA_Ag_shift frankejo 02:41:12 R seq
5738.nwzsuperdome QED_hmc tplvisit 00:10:13 R seq
-bash-3.00$ ls
error.txt Helloworld.exe Helloworld.f90 Helloworld.PBS output.txt
-bash-3.00$ cat error.txt
-bash-3.00$ cat output.txt
Das ist Nummer 0 von 4
Das ist Nummer 3 von 4
Das ist Nummer 2 von 4
Das ist Nummer 1 von 4
-bash-3.00$
```

Die error.txt-Datei ist leer, da kein Fehler aufgetreten ist.

Das Ergebnis in der output.txt Datei stimmt mit der erwarteten Ausgabe überein (bis auf die Reihenfolge der Threads). Damit ist das erste Beispiel beendet.

2. Teil: parallelisierte Schleife

Das Vorgehen ist genau dasselbe wie im ersten Teil, deshalb wird dieser Abschnitt etwas knapper sein:

2.1 Quellcode: Schleife.f90

(Quelle: <http://www.openmp.org/drupal/mp-documents/spec25.pdf>)

```
PROGRAM parallelierteSchleife
IMPLICIT NONE

INTEGER*4 I,N
REAL B(100),A(100)

N=100

DO I=1,N
A(I)=I**2                                !beliebige Definition für den array A
END DO

CALL SCHLEIFE(N,A,B)                    !siehe Subroutine

WRITE(*,* ) B                           !Ergebnis der Schleife
END PROGRAM

SUBROUTINE SCHLEIFE(N,A,B)
IMPLICIT NONE

INTEGER I,N
INTEGER OMP_GET_THREAD_NUM
REAL B(N),A(N)

B(1)=0.0d0

!$OMP PARALLEL                          !Initialisierung der parallelen Umgebung
THREADID=OMP_GET_THREAD_NUM()
!$OMP DO                                 !Initialisierung des (parallelen) Schleifenkonstrukts
DO I=2,N
                                        !Ausführen einer Schleife
        B(I)=(A(I)+A(I-1))/2.0d0
WRITE (*,*) 'Threadnummer:', THREADID, 'Schleifendurchlaufnummer:', I, 'Ergebnis:', B(I)
END DO
!$OMP END DO                             !Beenden des Schleifenkonstrukts
!$OMP END PARALLEL                       !Beenden der parallelen Umgebung

END SUBROUTINE
```

In dem Programm wird zunächst ein Vektor A willkürlich definiert und hieraus in einer Schleife ein neuer Vektor B berechnet. Um diese Schleife wurde eine parallele Umgebung gesetzt, so dass die Schleife nun parallel berechnet wird. Ausgegeben wird die Threadnummer, die Anzahl der Schleifendurchläufe und das Ergebnis B(I). Somit lässt sich nachvollziehen welcher Thread welche Komponente berechnet hat.

2.2 PBS-Datei zum Starten des Jobs:

Das PBS-Skript ändert sich nur geringfügig, nämlich die Bezeichnung des Jobs und der Pfad der exe-Datei:

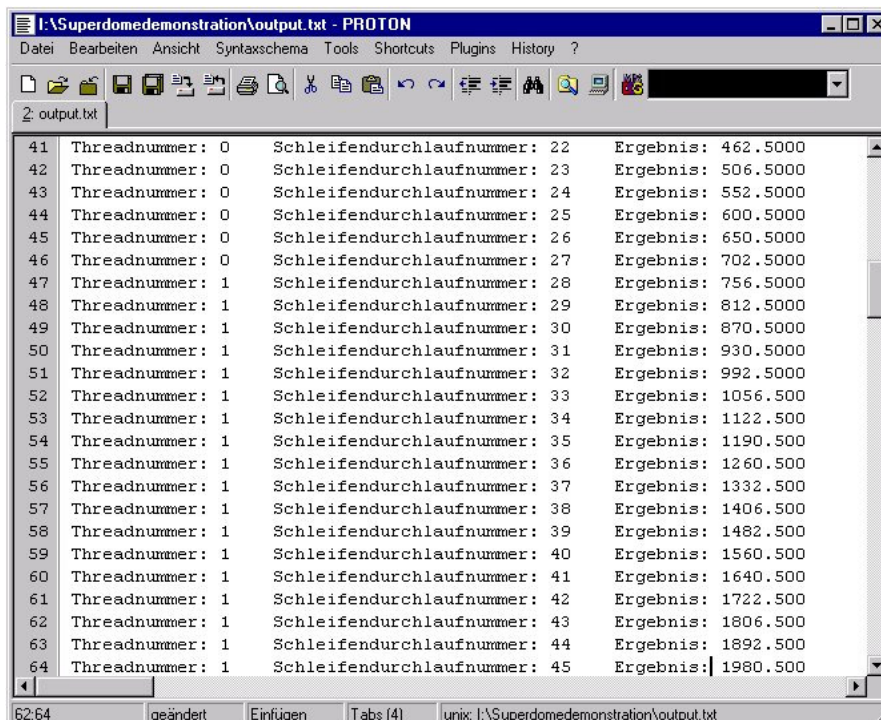
```
#PBS -q test
#PBS -N loop
#PBS -e error2.txt
#PBS -o output2.txt
#PBS -m abe
#PBS -M abc@uni-muenster.de
#PBS -l nodes=1:ppn=4
source /opt/intel/icc/9.1.036/bin/ifortvars.sh
PARNODES=`wc -l $PBS_NODEFILE |gawk '{print $1}'`
export OMP_NUM_THREADS=$PARNODES
~/Superdomedemonstration/Schleife.exe
```

Diese Zeilen kopieren und in eine Datei z.B. mit dem Namen Schleife.PBS speichern.

2.3 Starten des Jobs:

Nachdem man sich wieder auf der Superdome eingeloggt hat, startet man den Job mit `qsub Schleife.PBS`

Hier ein kleiner Auszug des Ergebnisses:



```
I:\Superdomedemonstration\output.txt - PROTON
Datei Bearbeiten Ansicht Syntaxschema Tools Shortcuts Plugins History ?
2: output.txt
41 Threadnummer: 0 Schleifendurchlaufnummer: 22 Ergebnis: 462.5000
42 Threadnummer: 0 Schleifendurchlaufnummer: 23 Ergebnis: 506.5000
43 Threadnummer: 0 Schleifendurchlaufnummer: 24 Ergebnis: 552.5000
44 Threadnummer: 0 Schleifendurchlaufnummer: 25 Ergebnis: 600.5000
45 Threadnummer: 0 Schleifendurchlaufnummer: 26 Ergebnis: 650.5000
46 Threadnummer: 0 Schleifendurchlaufnummer: 27 Ergebnis: 702.5000
47 Threadnummer: 1 Schleifendurchlaufnummer: 28 Ergebnis: 756.5000
48 Threadnummer: 1 Schleifendurchlaufnummer: 29 Ergebnis: 812.5000
49 Threadnummer: 1 Schleifendurchlaufnummer: 30 Ergebnis: 870.5000
50 Threadnummer: 1 Schleifendurchlaufnummer: 31 Ergebnis: 930.5000
51 Threadnummer: 1 Schleifendurchlaufnummer: 32 Ergebnis: 992.5000
52 Threadnummer: 1 Schleifendurchlaufnummer: 33 Ergebnis: 1056.500
53 Threadnummer: 1 Schleifendurchlaufnummer: 34 Ergebnis: 1122.500
54 Threadnummer: 1 Schleifendurchlaufnummer: 35 Ergebnis: 1190.500
55 Threadnummer: 1 Schleifendurchlaufnummer: 36 Ergebnis: 1260.500
56 Threadnummer: 1 Schleifendurchlaufnummer: 37 Ergebnis: 1332.500
57 Threadnummer: 1 Schleifendurchlaufnummer: 38 Ergebnis: 1406.500
58 Threadnummer: 1 Schleifendurchlaufnummer: 39 Ergebnis: 1482.500
59 Threadnummer: 1 Schleifendurchlaufnummer: 40 Ergebnis: 1560.500
60 Threadnummer: 1 Schleifendurchlaufnummer: 41 Ergebnis: 1640.500
61 Threadnummer: 1 Schleifendurchlaufnummer: 42 Ergebnis: 1722.500
62 Threadnummer: 1 Schleifendurchlaufnummer: 43 Ergebnis: 1806.500
63 Threadnummer: 1 Schleifendurchlaufnummer: 44 Ergebnis: 1892.500
64 Threadnummer: 1 Schleifendurchlaufnummer: 45 Ergebnis: 1980.500
62:64 geändert Einfügen Tabs (4) unix: I:\Superdomedemonstration\output.txt
```