StarCCM+

On this page you find variants of job scripts which can be used to run Siemens StarCCM+. If you are not yet familiar with SLURM, it is advised to use one of these scripts.

These scripts are updated from time to time. Therefore, review them once in a while.

If any job got killed before it could close itself, use the manual clean-up script.

Variant 1

job-starccm.sh

```
#!/bin/bash
# Version 07.2019
##################### Job Settings
##################################################################
# Specific Commands for the work load manager SLURM are lines beginning
with "#SBATCH"
#SBATCH -J tst                # Setting the display name for the
submission
#SBATCH -N 4                  # Number of nodes to reserve, -N 2-5  for
variable number of requested node count
#SBATCH --ntasks-per-node 16  # typically 16, range: 1..16 (max 16
cores per node)
#SBATCH -t 001:00:00          # set walltime in hours, format:
hhh:mm:ss, days-hh, days-hhh:mm:ss
#SBATCH -p short              # Desired Partition, alternatively
comment this line out and submit the script with 'sbatch -p big
jobscript.sh'
#SBATCH --mem 120000          # A Default Memory limit in MB.


##################### Simulation Settings
################################################################
## Work directory. No "/" at the end.
WORKDIR="/scratch/tmp/$USER/my_sim_dir"

## Simulation File (location in work directory)
SIMULATIONFILE="star.sim"

## Macro file (location in work directory)
MACROFILE="macro.java"
```

```
## Personal POD key
PERSONAL_PODKEY="XXXXXXXXXXXXXXXXXXXXX"

## Decide which version by commenting out the desired version.
#module load starCCM/11.06.011
#module load starCCM/12.02.011
module load starCCM/13.02.013

## Application. Can be kept constant if modules are used.
APPLICATION="starccm+"

## for macro usage
OPTIONS="$WORKDIR/$SIMULATIONFILE -batch $WORKDIR/$MACROFILE -licpath
1999@flex.cd-adapco.com -power -podkey $PERSONAL_PODKEY -collab -time -
rsh /usr/bin/ssh"
## just run
#OPTIONS="$WORKDIR/$SIMULATIONFILE -batch run -licpath 1999@flex.cd-
adapco.com -power -podkey $PERSONAL_PODKEY -collab -time -rsh
/usr/bin/ssh"


##################### Printing some Debug Information
################################################
# simplify debugging:
echo "SLURM_JOB_NODELIST=$SLURM_JOB_NODELIST"
echo "SLURM_NNODES=$SLURM_NNODES
SLURM_TASKS_PER_NODE=$SLURM_TASKS_PER_NODE"
env | grep -e MPI -e SLURM
echo "host=$(hostname) pwd=$(pwd) ulimit=$(ulimit -v) \$1=$1 \$2=$2"
exec 2>&1 # send errors into stdout stream

# list and echo loaded Modules
echo "Loaded Modules: $LOADEDMODULES"

## Change into Work Directory
cd $WORKDIR; echo pwd=$(pwd)
#
export OMP_WAIT_POLICY="PASSIVE"
export
OMP_NUM_THREADS=$((16/((SLURM_NPROCS+SLURM_NNODES-1)/SLURM_NNODES)))
[ $OMP_NUM_THREADS == 16 ] && export GOMP_CPU_AFFINITY="0-15:1" # task-
specifique
export OMP_PROC_BIND=TRUE
echo OMP_NUM_THREADS=$OMP_NUM_THREADS

[ "$SLURM_NNODES" ] && [ $SLURM_NNODES -lt 4 ] && srun bash -c "echo
task \$SLURM_PROCID of \$SLURM_NPROCS runs on \$SLURMD_NODENAME"


##################### Preparing the Simulation
#########################################################
```

```
## creating machinefile & temp in work directory
MACHINEFILE="machinefile.$SLURM_JOBID.txt"
TEMPMACHINEFILE="machinefile.temp.$SLURM_JOBID.txt"
scontrol show hostnames $SLURM_JOB_NODELIST > $WORKDIR/$TEMPMACHINEFILE
touch $WORKDIR/$MACHINEFILE

## Check the reserved nodes for running processes, when user processes
are left on reserved nodes then remove them from machinefile.
echo "Checking whether servers are clean (of user processes)"
for s in $(cat $WORKDIR/$TEMPMACHINEFILE)
do
    serverload=$(timeout 10s ssh $s ps aux | \
                 tail -n +2 | \
                 sed '/sshd/d;/ps
aux/d;/slurm_script/d;/cut/d;/sort/d;/sed/d;/tail/d;/uniq/d' | \
                 cut -d' ' -f1 | \
                 sort| \
                 sed
'/root/d;/munge/d;/dbus/d;/ldap/d;/nslcd/d;/postfix/d;/ntp/d;/nscd/d' |
\
                 uniq -c )
        if [ -n "$serverload" ] ; then
                echo "=== ERROR === On server ${s}, user processes are
running: \"${serverload}\""
        else
           echo "${s} seems clear."
               echo "${s}" >> $WORKDIR/$MACHINEFILE
        fi
done
rm $WORKDIR/$TEMPMACHINEFILE

## Calculating remaining NPROCS
CLEAN_NNODES=$(cat $WORKDIR/$MACHINEFILE | wc -l)
CLEAN_NPROCS=$(( $CLEAN_NNODES * $SLURM_NTASKS_PER_NODE ))

echo "Finally Running on $CLEAN_NPROCS processes on $CLEAN_NNODES
servers."


#################### Running the simulation
############################################################
## Let StarCCM+ wait for licenses on startup
export STARWAIT=1

## Start time stamp
date +%Y-%m-%d_%H:%M:%S_%s_%Z # date as YYYY-MM-DD_HH:MM:SS_Ww_ZZZ

## Command to run application (StarCCM+)
echo "Now, running the simulation ...."
$APPLICATION $OPTIONS -np $CLEAN_NPROCS -machinefile
$WORKDIR/$MACHINEFILE >
```

```
$WORKDIR/$SIMULATIONFILE.$SLURM_JOBID.output.log 2>&1


## Final time stamp
date +%Y-%m-%d_%H:%M:%S_%s_%Z



##################### Brute Force Clean-Up
############################################################
## This part kills ALL starccm processes (of your account in this job)
after a run.
## Though, it will only run, if starccm is not killed by SLRUM
before...
##
## Therefore, MAKE SURE starccm stops itself before the job is killed
(see LSS wiki)
##
## Probably, this part has become obsolete.

echo "Start Brute-Force clean up"
[ "$UID" != "0" ] && srun --ntasks-per-node=1 bash -c \
"find /dev/shm /tmp -xdev -mindepth 1 -maxdepth 1 -user $USER \
  -exec rm -rf --one-file-system {} \;"

[ "$UID" != "0" ] && srun --ntasks-per-node=1 bash -c \
"pkill -u $UID -9 star " 2>/dev/null

[ "$UID" != "0" ] && srun --ntasks-per-node=1 bash -c \
"pkill -u $UID -9 mpid " 2>/dev/null

echo "done."
```

Manual Clean-Up script

If your similation didn't end before the job has been killed by Slurm run the following script with your machinefile.
Run it by calling

```
./cleanup_afterKilledJob.sh mylastmaschinefile
```

cleanup_afterKilledJob.sh

```
#!/bin/bash
echo "Manual clean up starccm run"
for s in $(cat $1)
do
```

```
        echo "cleaning $s"
        ssh $s pkill -9 starccm+
        ssh $s pkill -9 star-ccm+
        ssh $s pkill -9 mpid
        ssh $s rm -v /dev/shm/*
done
```

Another example for a default Star-CCM+ simulation job simulation template:

Variant 2

simulationjob_20170926.sh

```
#!/bin/bash
# An example batch script to launch a Star-CCM+ simulation on the
Neumann cluster
# From command line on Neumann, log into a node, e.g. type "ssh c002"
# then submit this job to the queue system by typing "sbatch
simulationjob_20170926.sh"


##############################################################################
############################
# Queue system requests

#SBATCH --job-name myjobname      # job name displayed by squeue
#SBATCH --partition sw01_short        # queue in which this is going
#SBATCH --nodes 2         # number of nodes
#SBATCH --time 001:00:00      # time budget [HHH:MM:SS]
#SBATCH --mem 80G         # RAM memory allocated to each node

#SBATCH --dependency singleton        # singleton dependency: do not
start this job before any other job with the same job name has finished
#SBATCH --exclude c[005,006,007,008,009,010,011,012,013,014,015] #
exclude these nodes temporarily, for they have a different version of
MPI
#SBATCH --ntasks-per-node 16      # always leave this to 16 when using
Star-CCM+


##############################################################################
############################
# Basic setup

# simulation-specific
WORKINGDIRECTORY="/scratch/tmp/$USER/somethingsomething"       #  the
directory where the sim file is, without trailing slash
```

```
BASESIMFILE="somefilename"                # the name of the simfile,
without the ".sim" extension

# custom parameters, change once
PERSONAL_PODKEY="XXXXXXXXXXXX"

MACRO="" #"$WORKINGDIRECTORY/mymacrofile.java"      # if any macro file
is required, then uncomment as needed

# standard stuff, should normally not require editing
EXECUTABLE="starccm+"
SIMFILE="$BASESIMFILE.sim"
PATHTOSIMFILE="$WORKINGDIRECTORY/$SIMFILE"
MACHINEFILE="$WORKINGDIRECTORY/machinefile$SLURM_JOBID"
LOGFILE="$WORKINGDIRECTORY/simulationlog$SLURM_JOBID.log"
BREADCRUMBFILE="/home/$USER/breadcrumbs$SLURM_JOBID.log"


##############################################################################
############################
# Leave bread crumbs in home folder, in case something goes wrong (e.g.
scratch not available)

date +%Y-%m-%d_%H:%M:%S_%s_%Z >> $BREADCRUMBFILE # date as YYYY-MM-
DD_HH:MM:SS_Ww_ZZZ
echo "SLURM_JOB_NODELIST=$SLURM_JOB_NODELIST" >> $BREADCRUMBFILE
echo "SLURM_NNODES=$SLURM_NNODES
SLURM_TASKS_PER_NODE=$SLURM_TASKS_PER_NODE" >> $BREADCRUMBFILE
env | grep -e MPI -e SLURM >> $BREADCRUMBFILE
echo "host=$(hostname) pwd=$(pwd) ulimit=$(ulimit -v) \$1=$1 \$2=$2" >>
$BREADCRUMBFILE
srun -l /bin/hostname | sort -n | awk '{print $2}' >> $BREADCRUMBFILE


##############################################################################
############################
# Clean up files from previous sim. This is mostly useful if you want
to resume unsteady simulations

cd $WORKINGDIRECTORY
mkdir -pv old # create folder 'old' if it does not exist
mv -vf machinefile* old/
mv -vf simulationlog* old/
mv -vf *.sim~ old/


##############################################################################
############################
# Standard output for debugging + load modules

date +%Y-%m-%d_%H:%M:%S_%s_%Z >> $LOGFILE # date as YYYY-MM-
```

```
DD_HH:MM:SS_Ww_ZZZ
echo "SLURM_JOB_NODELIST=$SLURM_JOB_NODELIST" >> $LOGFILE
echo "SLURM_NNODES=$SLURM_NNODES
SLURM_TASKS_PER_NODE=$SLURM_TASKS_PER_NODE" >> $LOGFILE
env | grep -e MPI -e SLURM >> $LOGFILE
echo "host=$(hostname) pwd=$(pwd) ulimit=$(ulimit -v) \$1=$1 \$2=$2" >>
$LOGFILE
exec 2>&1 # send errors into stdout stream

# load modulefiles which set paths to mpirun and libs (see website for
more infos)
echo "Loaded modules so far: $LOADEDMODULES" >> $LOGFILE
#module load starCCM/11.04.012
module load starCCM/12.02.011
echo "Loaded modules are now: $LOADEDMODULES" >> $LOGFILE

## jobscript should not be started in /scratch (conflicting link@master
vs. mount@nodes), see website for more info
#cd /scratch/tmp/${USER}01;echo new_pwd=$(pwd) # change local path for
faster or massive I/O

export OMP_WAIT_POLICY="PASSIVE"
export
OMP_NUM_THREADS=$((16/((SLURM_NPROCS+SLURM_NNODES-1)/SLURM_NNODES)))
[ $OMP_NUM_THREADS == 16 ] && export GOMP_CPU_AFFINITY="0-15:1" # task-
specifique
export OMP_PROC_BIND=TRUE
echo OMP_NUM_THREADS=$OMP_NUM_THREADS >> $LOGFILE


##############################################################################
############################
# Prepare simulation

cd $WORKINGDIRECTORY

# Find out what resources are available
[ "$SLURM_NNODES" ] && [ $SLURM_NNODES -lt 4 ] && srun bash -c "echo
task \$SLURM_PROCID of \$SLURM_NPROCS runs on \$SLURMD_NODENAME"

echo "task $SLURM_PROCID of $SLURM_NPROCS runs on $SLURMD_NODENAME" >>
$LOGFILE

echo "SLURM_NPROCS = $SLURM_NPROCS" >> $LOGFILE

echo "OMP_NUM_THREADS=$OMP_NUM_THREADS" >> $LOGFILE

# List available machines in machinefile
srun -l /bin/hostname | sort -n | awk '{print $2}' > $MACHINEFILE
```

```
##############################################################################
############################
# Make backup of sim file. This is mostly useful if you want to resume
unsteady simulations, and is commented-out by default

# Quick and precarious bakcup of previous start file
#mv -vf $SIMFILE old/thepreviousssimfile.sim

# List all files, select autosave files among them, take one with
biggest file name, make copy of it named $SIMFILE
#ls | grep $BASESIMFILE@ | tail -1 | xargs -I file cp -vf file
$PATHTOSIMFILE

# Wait 2 minutes: workaround bug where Star-CCM+ starts before the copy
above has really completed
#echo "sleeping 120 seconds…" >> $LOGFILE
#sleep 120s
#echo "finished sleeping." >> $LOGFILE


##############################################################################
############################
# Run the actual simulation

# Run Star-CCM+ & output to logfile
$EXECUTABLE $PATHTOSIMFILE -v -machinefile $MACHINEFILE -rsh
/usr/bin/ssh -licpath 1999@flex.cd-adapco.com -power -podkey
$PERSONAL_PODKEY -np $SLURM_NPROCS -batch $MACRO -collab -clientcore >>
$LOGFILE



##############################################################################
############################
# Brute-force cleanup

wait

echo "Start Brute-force clean up" >> $LOGFILE
for s in $(cat $MACHINEFILE)
do
    ssh $s pkill -9 starccm+
    ssh $s pkill -9 star-ccm+
    ssh $s pkill -9 mpid
        ssh $s rm -v /dev/shm/*
done
```