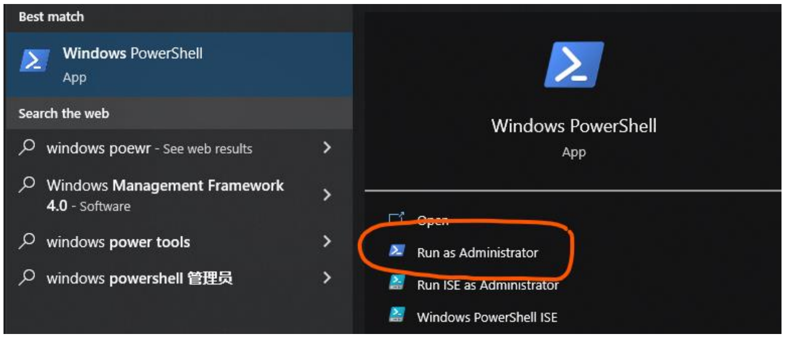
SWASH Parallel Mode on Windows OS via WSL2

This document will walk through the installation of SWASH v7.01 on a Windows OS (Windows 10, 64-bit). The workflow consists of 4 steps: (1) installing a windows subsystem for Linux (WSL2), (2) compiling MPICH2, (3) compiling SWASH, and (4) testing SWASH.

Step 1: Download Windows Subsystem for Linux (WSL2)

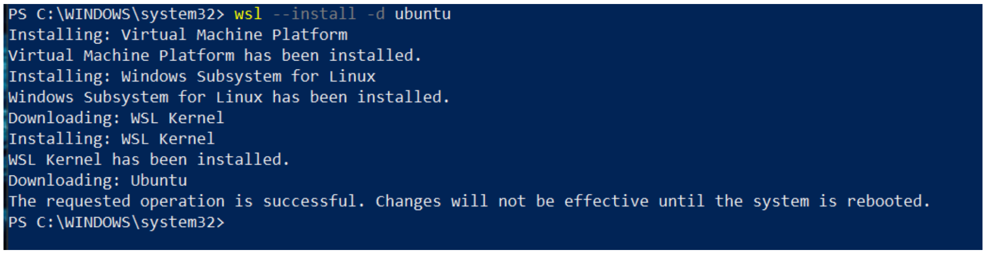
WSL2 is a tool that allows windows users to run a Linux distribution (in our case, Ubuntu) in a container-like environment. Calls from within this system are executed at native speed.

1.1 Navigate to Windows Power Shell and Run as Administrator



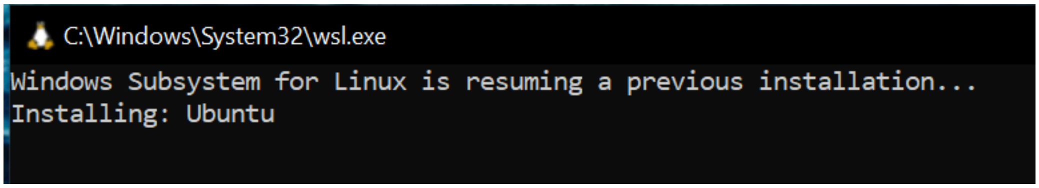
1.2 In the command prompt type:

wsl --install -d ubuntu

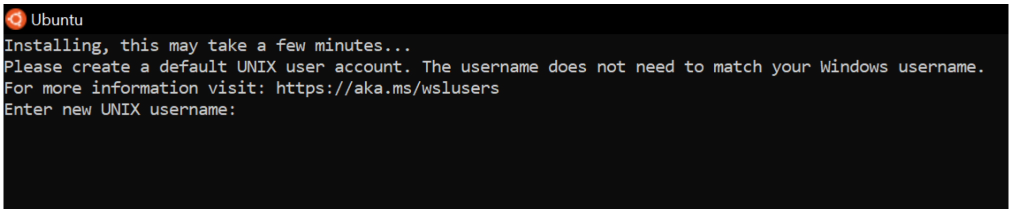


```
PS C:\WINDOWS\system32> wsl --install -d ubuntu
Installing: Virtual Machine Platform
Virtual Machine Platform has been installed.
Installing: Windows Subsystem for Linux
Windows Subsystem for Linux has been installed.
Downloading: WSL Kernel
Installing: WSL Kernel
WSL Kernel has been installed.
Downloading: Ubuntu
The requested operation is successful. Changes will not be effective until the system is rebooted.
PS C:\WINDOWS\system32>
```

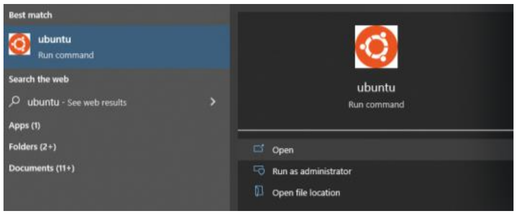
1.3 Restart computer. It will then continue running the install, prompt you to create a username and password, then you should have an Ubuntu terminal



```
Windows Subsystem for Linux is resuming a previous installation...
Installing: Ubuntu
```



```
Ubuntu
Installing, this may take a few minutes...
Please create a default UNIX user account. The username does not need to match your Windows username.
For more information visit: https://aka.ms/wslusers
Enter new UNIX username:
```



1.4 Make sure you have all the latest updates by running:

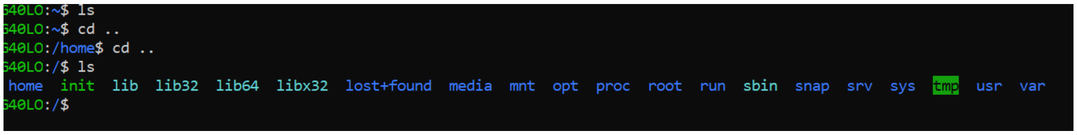
sudo apt update

sudo apt upgrade

1.5 I recommend then installing Windows Terminal Preview so you can have multiple command line tabs open simultaneously. It conveniently allows you to choose between PowerShell, Command Prompt, Ubuntu, etc.

Step 2: Install MPICH2 from source

2.1 Open an Ubuntu terminal, navigate to the where all the build directories are found

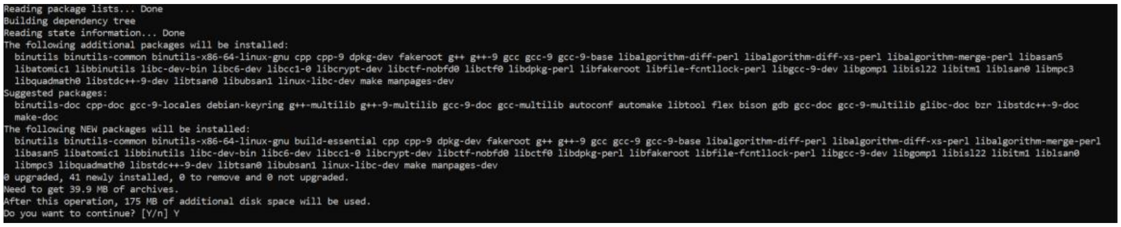


```
540LO:~$ ls
540LO:~$ cd ..
540LO:/home$ cd ..
540LO:/$ ls
 home  init  lib  lib32  lib64  libx32  lost+found  media  mnt  opt  proc  root  run  sbin  snap  srv  sys  tmp  usr  var
540LO:/$
```

2.2 Install C++ and Fortran 90 compilers, type ‘Y’ when prompted to confirm installation:

sudo apt-get install build-essential

sudo apt-get install gfortran



```
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu cpp cpp-9 dpkg-dev fakeroot g++ g++-9 gcc gcc-9 gcc-9-base libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl libasan5
  libatomic1 libbinutils libc-dev-bin libc6-dev libcc1-0 libcrypt-dev libctf-nobfd0 libctf0 libdpkg-perl libfakeroot libfile-fcntllock-perl libgcc-9-dev libgomp1 libisl22 libitm1 liblsan0 libmpc3
  libquadmath0 libstdc++-9-dev libtsan0 libubsan1 linux-libc-dev make manpages-dev
Suggested packages:
  binutils-doc cpp-doc gcc-9-locales debian-keyring g++-multilib g++-9-multilib gcc-9-doc gcc-multilib autoconf automake libtool flex bison gdb gcc-doc gcc-9-multilib glibc-doc bzr libstdc++-9-doc
  make-doc
The following NEW packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu build-essential cpp cpp-9 dpkg-dev fakeroot g++ g++-9 gcc gcc-9 gcc-9-base libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl
  libasan5 libatomic1 libbinutils libc-dev-bin libc6-dev libcc1-0 libcrypt-dev libctf-nobfd0 libctf0 libdpkg-perl libfakeroot libfile-fcntllock-perl libgcc-9-dev libgomp1 libisl22 libitm1 liblsan0
  libmpc3 libquadmath0 libstdc++-9-dev libtsan0 libubsan1 linux-libc-dev make manpages-dev
0 upgraded, 41 newly installed, 0 to remove and 0 not upgraded.
Need to get 39.9 MB of archives.
After this operation, 175 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
```

2.3 Create a new directory to store MPICH2 and SWASH builds (I call it mirror):

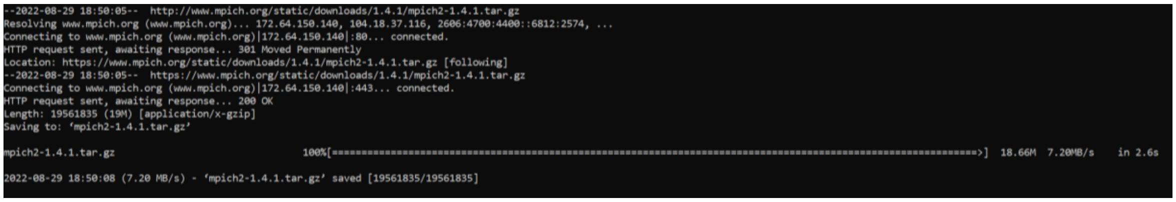
sudo mkdir mirror

2.4 Change into the new directory:

cd /mirror

2.5 Download mpich2 source code:

sudo wget http://www.mpich.org/static/downloads/1.4.1/mpich2-1.4.1.tar.gz



```
--2022-08-29 18:50:05--  http://www.mpich.org/static/downloads/1.4.1/mpich2-1.4.1.tar.gz
Resolving www.mpich.org (www.mpich.org)... 172.64.150.140, 104.18.37.116, 2606:4700:4400::6812:2574, ...
Connecting to www.mpich.org (www.mpich.org)|172.64.150.140|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://www.mpich.org/static/downloads/1.4.1/mpich2-1.4.1.tar.gz [following]
--2022-08-29 18:50:05--  https://www.mpich.org/static/downloads/1.4.1/mpich2-1.4.1.tar.gz
Connecting to www.mpich.org (www.mpich.org)|172.64.150.140|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 19561835 (19M) [application/x-gzip]
Saving to: ‘mpich2-1.4.1.tar.gz’

mpich2-1.4.1.tar.gz                       100%[============================================================================================>]  18.66M  7.20MB/s    in 2.6s

2022-08-29 18:50:08 (7.20 MB/s) - ‘mpich2-1.4.1.tar.gz’ saved [19561835/19561835]
```

2.6 Extract the source code:

sudo tar xzvf mpich2-1.4.1.tar.gz

2.7 Change into the extracted directory, configure, make, and install:

cd mpich2-1.4.1/

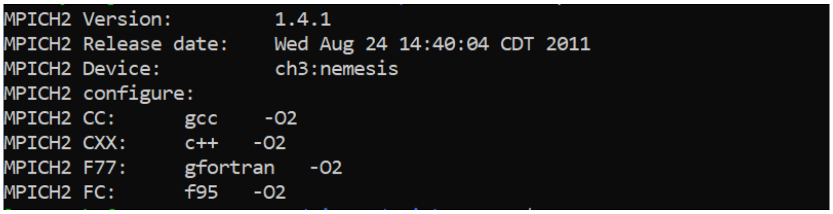
sudo ./configure

sudo make

sudo make install

2.8 Check that the installation was successful with:

mpich2version



```
MPICH2 Version:         1.4.1
MPICH2 Release date:    Wed Aug 24 14:40:04 CDT 2011
MPICH2 Device:          ch3:nemesis
MPICH2 configure:
MPICH2 CC:      gcc    -O2
MPICH2 CXX:     c++    -O2
MPICH2 F77:     gfortran   -O2
MPICH2 FC:      f95    -O2
```

2.9 Restart machine for good measure

Step 3: Install SWASH from source

3.1 Download SWASH 7.01 source code from this link:

https://swash.sourceforge.io/download/zip/swash-7.01.tar.gz

And you can read through the documentation here:

https://swash.sourceforge.io/online_doc/swashimp/node2.html

Keep in mind we are deviating from their listed recommendations for a Windows build by opting for a WSL instead of using Intel tools.

3.2 Copy swash-7.01.tar.gz into your build directory in Ubuntu terminal (mine is called mirror)

FYI - You can navigate to your local C: drive via:

cd /mnt/c

Here is how I copied from downloads into mirror:

Navigate to tar.gz location

sudo cp swash-7.01.tar.gz /mirror

3.3 Extract it:

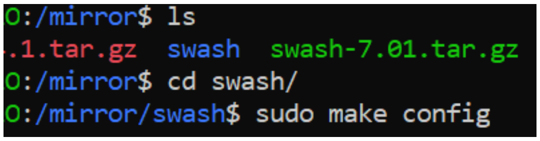
sudo tar xzvf swash-7.01.tar.gz

3.4 Enter into the swash directory

cd swash

3.5 Configure the build:

sudo make config



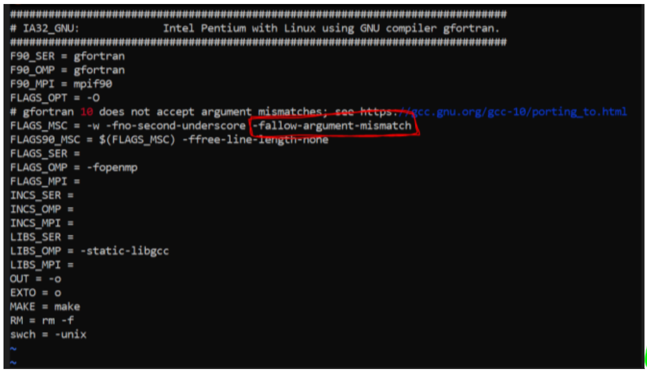
```
O:/mirror$ ls
.1.tar.gz  swash  swash-7.01.tar.gz
O:/mirror$ cd swash/
O:/mirror/swash$ sudo make config
```

3.6 Modify the macros file:

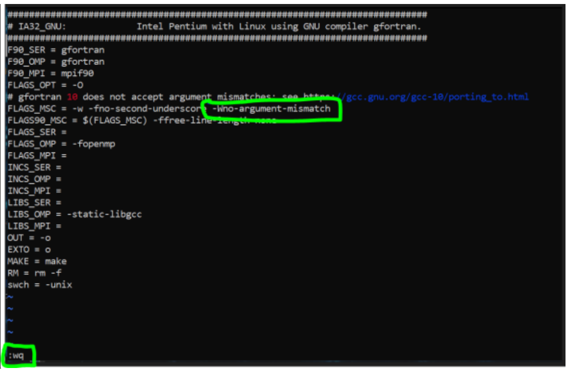
vi macros.inc

Change *-fallow statement* to *-Wno-argument-mismatch* (hit *i* to enter insert mode to edit)

Hit *Esc* then type *:wq* to write (save) the changes and quit



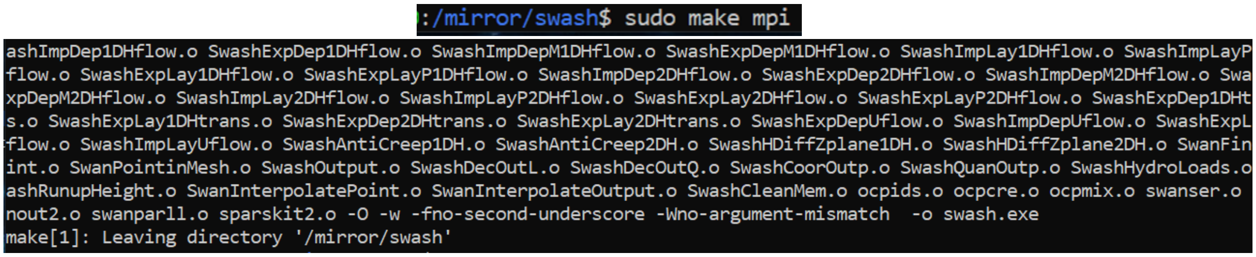
```
################################################################################
# IA32_GNU:             Intel Pentium with Linux using GNU compiler gfortran.
################################################################################
F90_SER = gfortran
F90_OMP = gfortran
F90_MPI = mpif90
FLAGS_OPT = -O
# gfortran 10 does not accept argument mismatches; see https://gcc.gnu.org/gcc-10/porting_to.html
FLAGS_MSC = -w -fno-second-underscore -fallow-argument-mismatch
FLAGS90_MSC = $(FLAGS_MSC) -ffree-line-length-none
FLAGS_SER =
FLAGS_OMP = -fopenmp
FLAGS_MPI =
INCS_SER =
INCS_OMP =
INCS_MPI =
LIBS_SER =
LIBS_OMP = -static-libgcc
LIBS_MPI =
OUT = -o
EXTO = o
MAKE = make
RM = rm -f
swch = -unix
~
~
```



```
################################################################################
# IA32_GNU:             Intel Pentium with Linux using GNU compiler gfortran.
################################################################################
F90_SER = gfortran
F90_OMP = gfortran
F90_MPI = mpif90
FLAGS_OPT = -O
# gfortran 10 does not accept argument mismatches; see https://gcc.gnu.org/gcc-10/porting_to.html
FLAGS_MSC = -w -fno-second-underscore -Wno-argument-mismatch
FLAGS90_MSC = $(FLAGS_MSC) -ffree-line-length-none
FLAGS_SER =
FLAGS_OMP = -fopenmp
FLAGS_MPI =
INCS_SER =
INCS_OMP =
INCS_MPI =
LIBS_SER =
LIBS_OMP = -static-libgcc
LIBS_MPI =
OUT = -o
EXTO = o
MAKE = make
RM = rm -f
swch = -unix
~
~
~
~
:wq
```

3.7 Build swash in parallel!

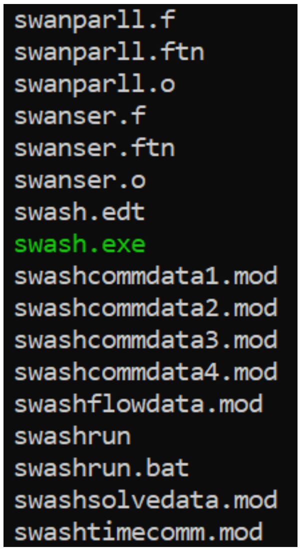
sudo make mpi



```
:/mirror/swash$ sudo make mpi
```

```
ashImpDep1DHflow.o SwashExpDep1DHflow.o SwashImpDepM1DHflow.o SwashExpDepM1DHflow.o SwashImpLay1DHflow.o SwashImpLayP
flow.o SwashExpLay1DHflow.o SwashExpLayP1DHflow.o SwashImpDep2DHflow.o SwashExpDep2DHflow.o SwashImpDepM2DHflow.o Swa
xpDepM2DHflow.o SwashImpLay2DHflow.o SwashImpLayP2DHflow.o SwashExpLay2DHflow.o SwashExpLayP2DHflow.o SwashExpDep1DHt
s.o SwashExpLay1DHtrans.o SwashExpDep2DHtrans.o SwashExpLay2DHtrans.o SwashExpDepUflow.o SwashImpDepUflow.o SwashExpL
flow.o SwashImpLayUflow.o SwashAntiCreep1DH.o SwashAntiCreep2DH.o SwashHDiffZplane1DH.o SwashHDiffZplane2DH.o SwanFin
int.o SwanPointinMesh.o SwashOutput.o SwashDecOutL.o SwashDecOutQ.o SwashCoorOutp.o SwashQuanOutp.o SwashHydroLoads.o
ashRunupHeight.o SwanInterpolatePoint.o SwanInterpolateOutput.o SwashCleanMem.o ocpids.o ocpcre.o ocpmix.o swanser.o
nout2.o swanparll.o sparskit2.o -O -w -fno-second-underscore -Wno-argument-mismatch  -o swash.exe
make[1]: Leaving directory '/mirror/swash'
```

You should now see the swash executable



```
swanparll.f
swanparll.ftn
swanparll.o
swanser.f
swanser.ftn
swanser.o
swash.edt
swash.exe
swashcommdata1.mod
swashcommdata2.mod
swashcommdata3.mod
swashcommdata4.mod
swashflowdata.mod
swashrun
swashrun.bat
swashsolvedata.mod
swashtimecomm.mod
```

Step 4: Test SWASH

4.1 Navigate to a directory with a swash test setup, and explicitly call the newly-built parallel executable, choose -n number of processors you want to allocate to the run:

mpiexec -n 18 /mirror/swash/swash.exe INPUT