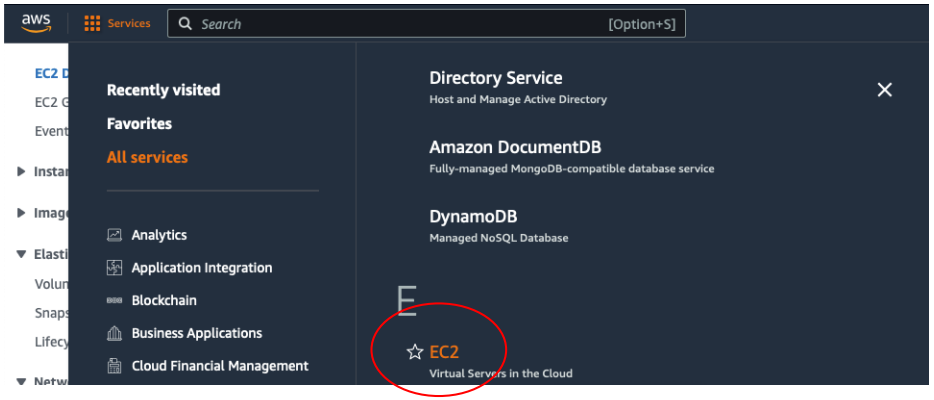
CPU Compile on AWS

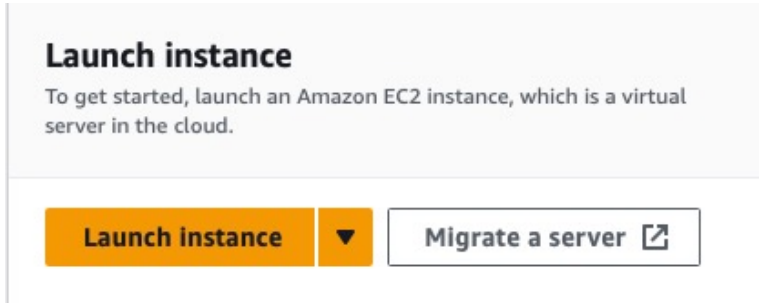
Programming FPGAs for Economics:
An Introduction to Electrical Engineering Economics

Bhagath Cheela, Alessandro Peri, André DeHon, Jesús Fernández-Villaverde

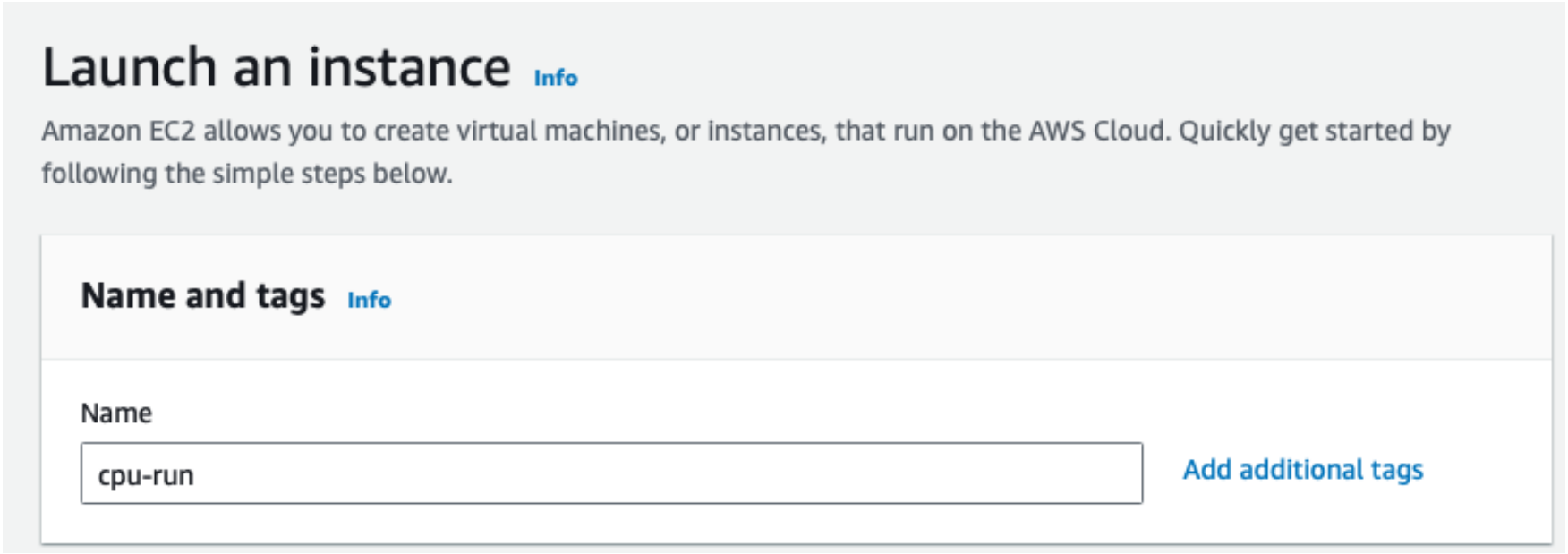
Steps



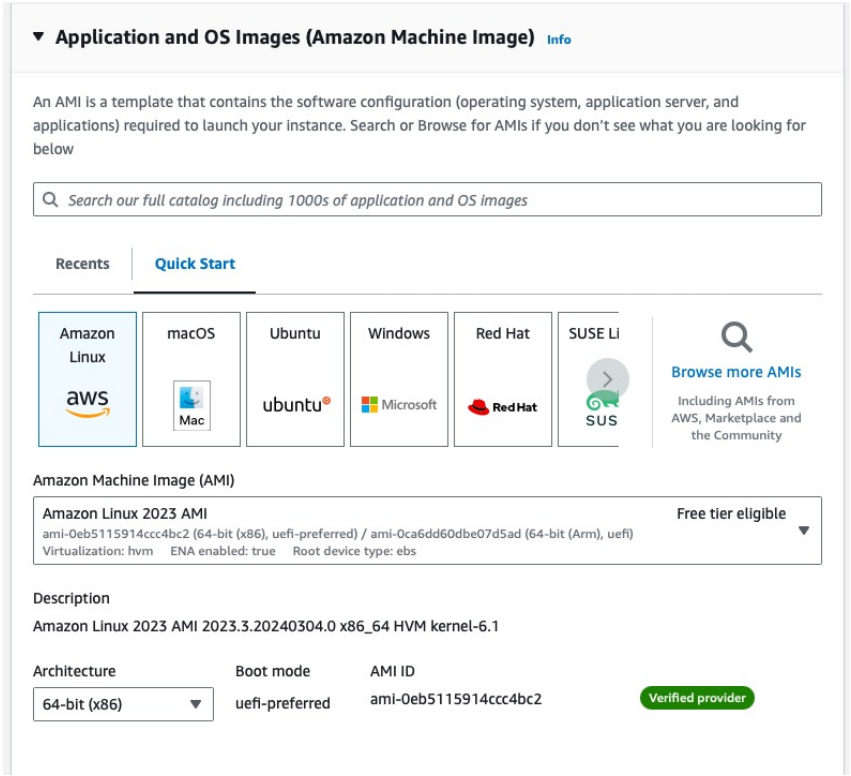
1. **Log into your AWS account:**
2. Navigate to the Home Console
3. Select **EC2**
4. Launch Instance



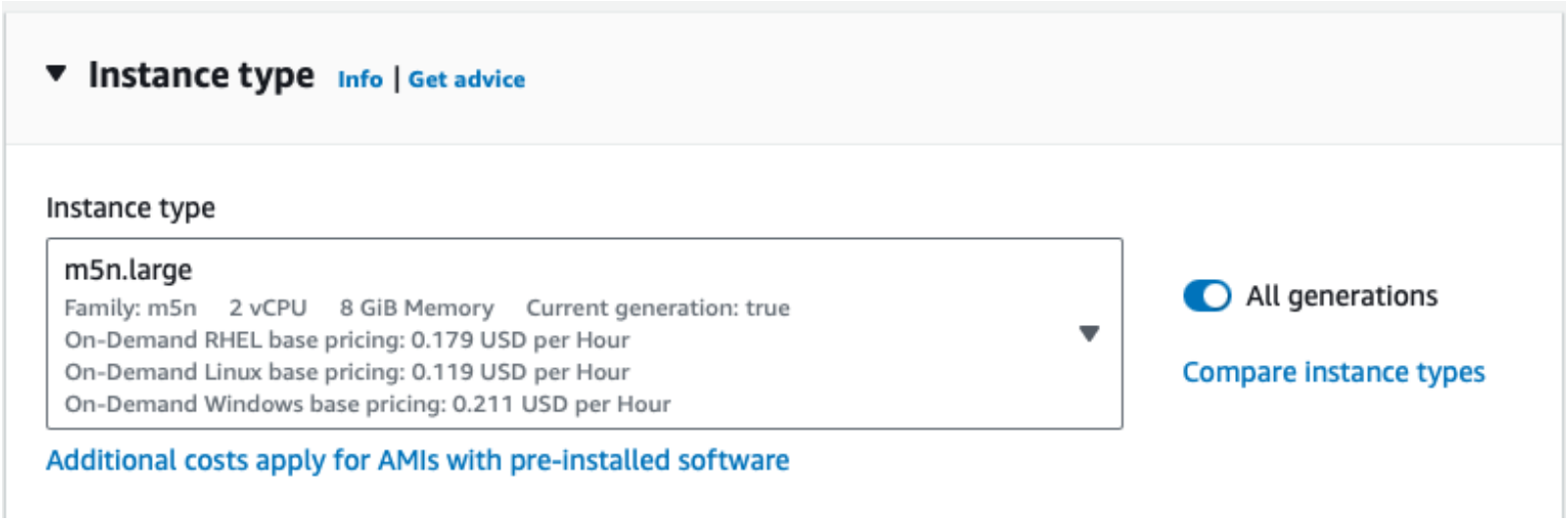
Steps: Name and tags



Select Amazon Linux Instance

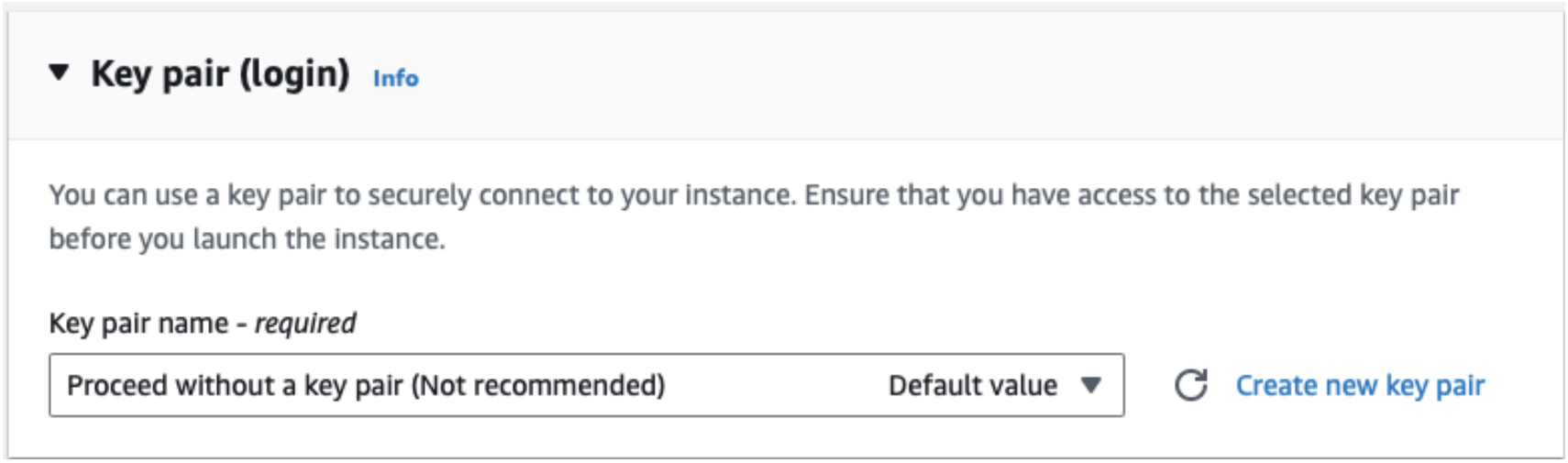


Select CPU Instance



Create all binaries from an m5n.large instance

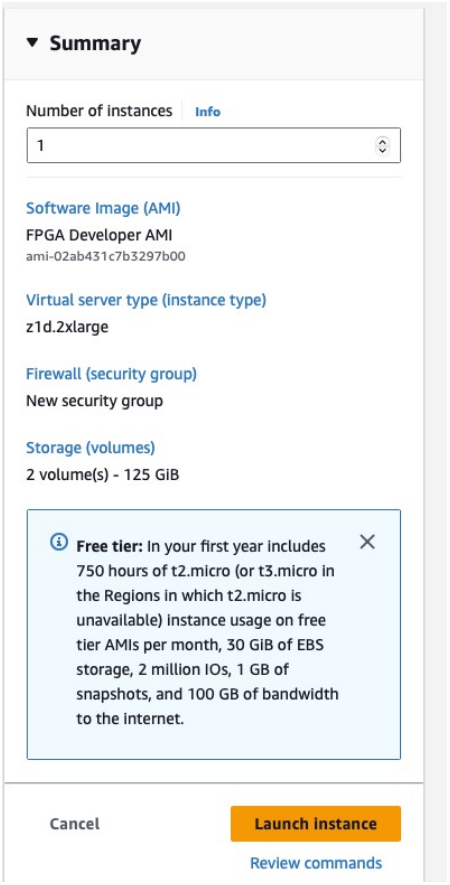
Key pair



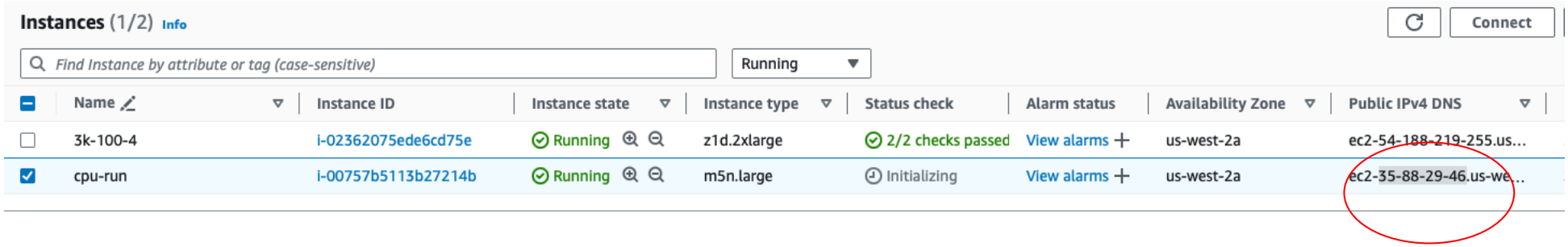
For information on how to create a new key pair go here

Launch m5n.large Instance

Repeat the same for all other CPU instances

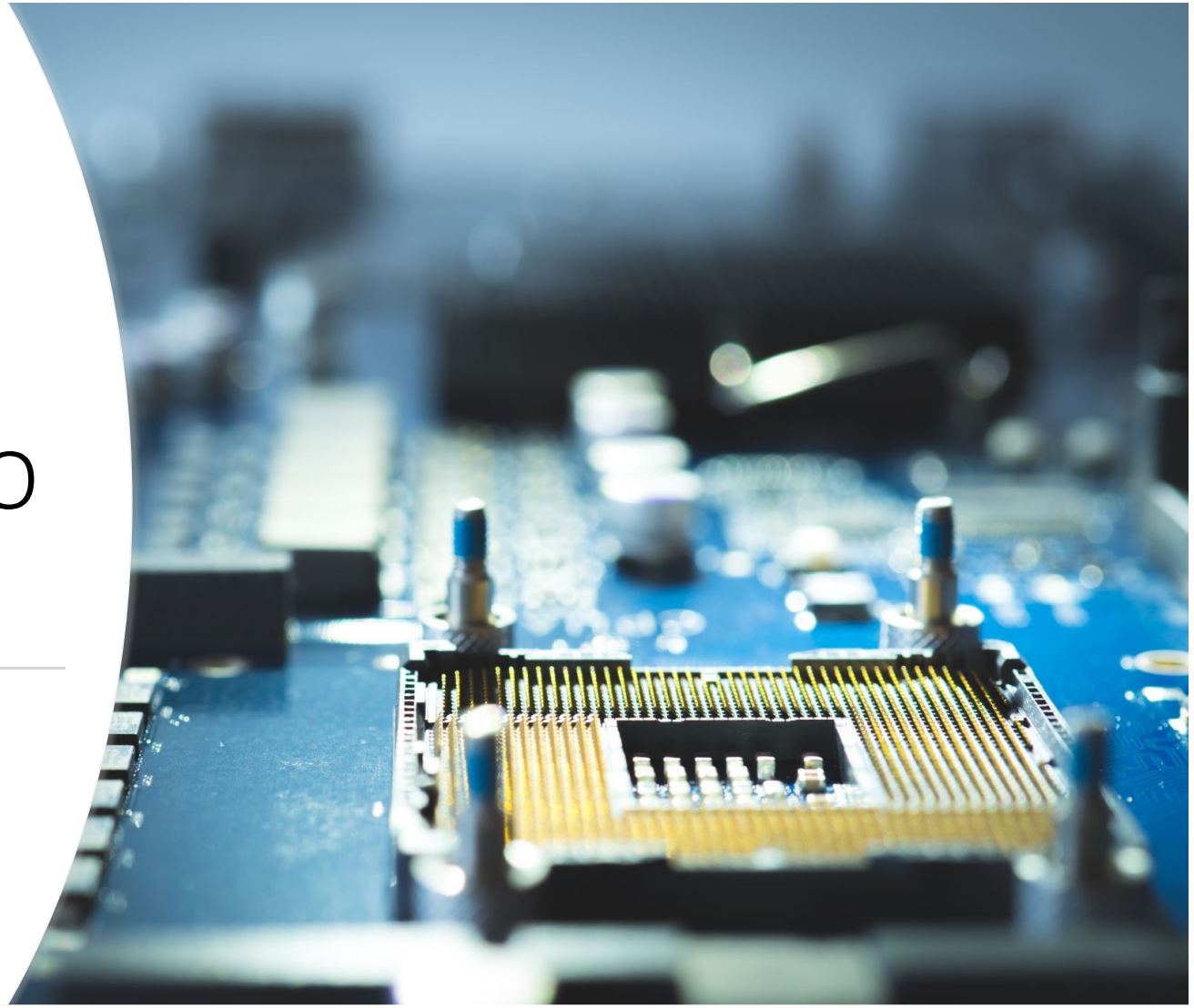


EC2 Instances



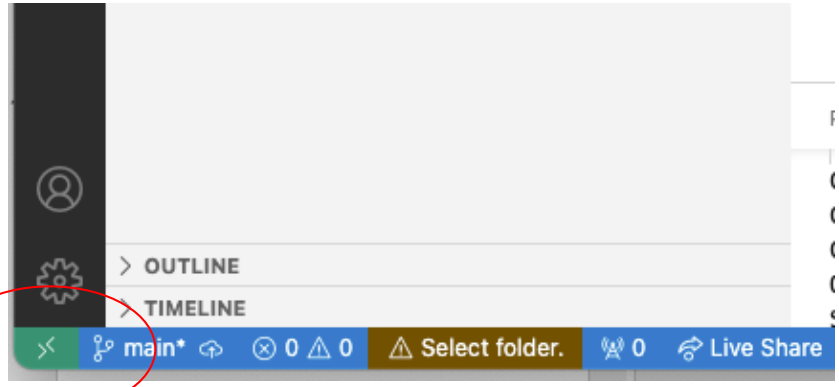
- In the top-left menu, select 'Instances’
- Copy the public IPv4 address in Visual Studio code

VISUAL STUDIO CODE



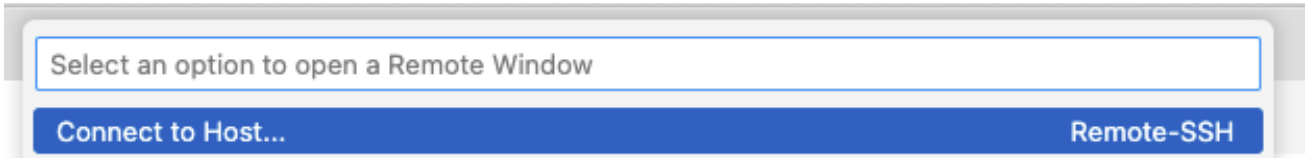
Open a Remote Window

- On the bottom-left corner of Visual Studio Code click on the green button ‘Open a Remote Window’
- Click on Connect to Host
- Click on Configure SSH Hosts.
  - Give a name to the Host (e.g. aws-ec-cpu)
  - Copy the public IP address (red circle) and set the path to your key
  - Set the user to **ec2-user**
- Connect to aws-ec2.




```
# ------------------------------ CONNECT TO AMAZON AWS
Host aws-ec-cpu
HostName ec2-34-219-191-114.us-west-2.compute.amazonaws.com
User ec2-user
IdentityFile ~/[YOUR KEY PATH].pem
# ----------------------------
```

CLICK the GREEN BUTTON HERE



Compile the Binaries

Install utilities and Copy Repo

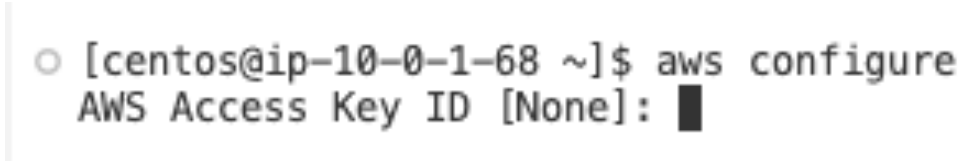
- Initiate a terminal session on the AWS instance and run the subsequent script to install the utilities

```
sudo yum install git -y
sudo yum install make -y
sudo yum install tmux -y
sudo yum install wget -y
```

- Clone our GitHub repository into a directory of your preference (e.g., /home/ec2-user):

```
git clone https://github.com/AleP83/FPGA-Econ.git
```

AWS Configure



1. Go to your aws account and set (one time thing):
   - AWS Access Key ID
   -AWS Secret Access Key
2. Go to the terminal in visual studio and type aws configure

```
aws configure
 AWS Access Key ID [**************xxxx]: <Your AWS Access Key ID>
 AWS Secret Access Key [**************xxxx]: <Your AWS Secret Access Key>
 Default region name: [us-west-1]: us-east-1
 Default output format [None]: json
```

3. Set:
   - AWS Access Key ID:
   - AWS Secret Access Key
   - Default Region name: us-west-2
     Note: this depends in which region you launched your instance.
   - Default output format [json]: json

Install OpenMPI ans set the Environment

- Install `OpenMPI`. Then, run the following script from the terminal

```
sh code/common/util/OpenMPI_install.sh
```

- To compile binaries for parallel execution set the OpenMPI environment by executing the following commands in the terminal from the parent directory:

```
export PATH=$PATH:$HOME/openmpi/bin
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$HOME/openmpi/lib
```

Modify the Makefile

- Set the AWS S3 Bucket Name. Specify the S3 bucket name by replacing S3-NAME-GOES-HERE

```
S3_EXE_BUCKET_NAME := S3-NAME-GOES-HERE
```

Remark: The S3 bucket name must be globally unique within AWS. If an error occurs during bucket creation, it may be due to the name being already in use by another user.

- Select the AWS region of the S3 bucket (default is us-west-2):

```
AWS_REGION := us-west-2
```

Modify the following files

Modify the files and Compile

- Open `/code/common/app.cpp` and set the number of models `N_MODEL` you want to compute (1,200 in our benchmark specification)

```
#define N_MODEL 1200 // total number of models
```

- Open `/code/common/definitions.h` and set the grid size

```
#define NKGRID 100 // grid points on individual capital grid
#define NKM_GRID 4 // grid points on aggregate capital grid
```

- Open `/code/common/dev options.h` and select the interpolation-range search algorithm:

```
#define _LINEAR_SEARCH 0
#define _BINARY_SEARCH 0
#define _CUSTOM_BINARY_SEARCH 1
```

- In the terminal, compile the application for CPU execution (e.g. for benchmark model)

```
make cpu_to_s3 CPU_EXE=1200_100k_4km
make openmpi_to_s3 OPENMPI_EXE=mpi_1200_100k_4km
```

Create all Binaries for Sequential Execution

Grid Sizes: 100-4, N. Economies: 1200

/common/app.cpp

```
#define N_MODEL 1200 // total number of models
```

/common/definitions.h

```
#define NKGRID 100 // grid points on individual capital grid
#define NKM_GRID 4 // grid points on aggregate capital grid
```

/common/dev_options.h

```
#define _LINEAR_SEARCH 0
#define _BINARY_SEARCH 0
#define _CUSTOM_BINARY_SEARCH 1
```

Terminal launch:

```
make clean
make cpu_to_s3 CPU_EXE=1200_100k_4km
```

Grid Sizes: 200-4, N. Economies: 1200

/common/app.cpp

```
#define N_MODEL 1200 // total number of models
```

/common/definitions.h

```
#define NKGRID 200 // grid points on individual capital grid
#define NKM_GRID 4 // grid points on aggregate capital grid
```

/common/dev_options.h

```
#define _LINEAR_SEARCH 0
#define _BINARY_SEARCH 0
#define _CUSTOM_BINARY_SEARCH 1
```

Terminal launch:

```
make clean
make cpu_to_s3 CPU_EXE=1200_200k_4km
```

Grid Sizes: 300-4, N. Economies: 1200

/common/app.cpp

```
#define N_MODEL 1200 // total number of models
```

/common/definitions.h

```
#define NKGRID 300 // grid points on individual capital grid
#define NKM_GRID 4 // grid points on aggregate capital grid
```

/common/dev_options.h

```
#define _LINEAR_SEARCH 0
#define _BINARY_SEARCH 0
#define _CUSTOM_BINARY_SEARCH 1
```

Terminal launch:

```
make clean
make cpu_to_s3 CPU_EXE=1200_300k_4km
```

Grid Sizes: 100-8, N. Economies: 1200

/common/app.cpp

```
#define N_MODEL 1200 // total number of models
```

/common/definitions.h

```
#define NKGRID 100 // grid points on individual capital grid
#define NKM_GRID 8 // grid points on aggregate capital grid
```

/common/dev_options.h

```
#define _LINEAR_SEARCH 0
#define _BINARY_SEARCH 0
#define _CUSTOM_BINARY_SEARCH 1
```

Terminal launch:

```
make clean
make cpu_to_s3 CPU_EXE=1200_100k_8km
```

Grid Sizes: 200-8, N. Economies: 1200

/common/app.cpp

```
#define N_MODEL 1200 // total number of models
```

/common/definitions.h

```
#define NKGRID 200 // grid points on individual capital grid
#define NKM_GRID 8 // grid points on aggregate capital grid
```

/common/dev_options.h

```
#define _LINEAR_SEARCH 0
#define _BINARY_SEARCH 0
#define _CUSTOM_BINARY_SEARCH 1
```

Terminal launch:

```
make clean
make cpu_to_s3 CPU_EXE=1200_200k_8km
```

Grid Sizes: 300-8, N. Economies: 1200

/common/app.cpp

```
#define N_MODEL 1200 // total number of models
```

/common/definitions.h

```
#define NKGRID 300 // grid points on individual capital grid
#define NKM_GRID 8 // grid points on aggregate capital grid
```

/common/dev_options.h

```
#define _LINEAR_SEARCH 0
#define _BINARY_SEARCH 0
#define _CUSTOM_BINARY_SEARCH 1
```

Terminal launch:

```
make clean
make cpu_to_s3 CPU_EXE=1200_300k_8km
```

Linear, Grid Sizes: 100-4, N. Economies: 1200

/common/app.cpp

```
#define N_MODEL 1200 // total number of models
```

/common/definitions.h

```
#define NKGRID 100 // grid points on individual capital grid
#define NKM_GRID 4 // grid points on aggregate capital grid
```

/common/dev_options.h

```
#define _LINEAR_SEARCH 1
#define _BINARY_SEARCH 0
#define _CUSTOM_BINARY_SEARCH 0
```

Terminal launch:

```
make clean
make cpu_to_s3 CPU_EXE=1200_linear
```

Binary, Grid Sizes: 100-4, N. Economies: 1200

/common/app.cpp

```
#define N_MODEL 1200 // total number of models
```

/common/definitions.h

```
#define NKGRID 100 // grid points on individual capital grid
#define NKM_GRID 4 // grid points on aggregate capital grid
```

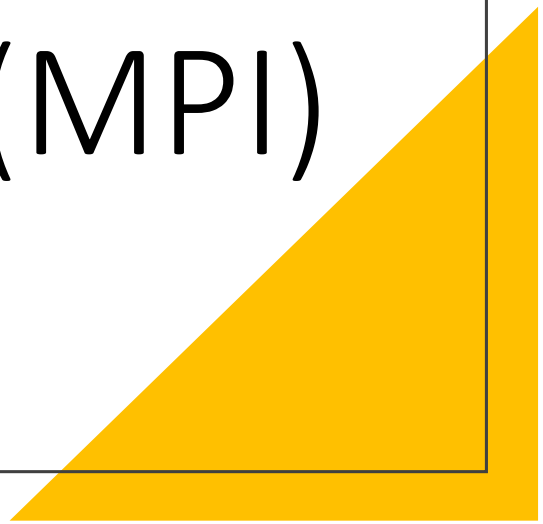
/common/dev_options.h

```
#define _LINEAR_SEARCH 0
#define _BINARY_SEARCH 1
#define _CUSTOM_BINARY_SEARCH 0
```

Terminal launch:

```
make clean
make openmpi OPENMPI_EXE =1200_binary
```

Create Binaries for Parallel Execution (MPI)



Grid Sizes: 100-4, N. Economies: 1200

/common/app.cpp

```
#define N_MODEL 1200 // total number of models
```

/common/definitions.h

```
#define NKGRID 100 // grid points on individual capital grid
#define NKM_GRID 4 // grid points on aggregate capital grid
```

/common/dev_options.h

```
#define _LINEAR_SEARCH 0
#define _BINARY_SEARCH 0
#define _CUSTOM_BINARY_SEARCH 1
```

Terminal launch:

```
export PATH=$PATH:$HOME/openmpi/bin
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$HOME/openmpi/lib
make clean
make openmpi_to_s3 OPENMPI_EXE=mpi_1200_100k_4km
```