MultiExplorer - User Manual

Erick Rocha Amorim, Gregório Koslinski Neto, Igor Oliveira Mello, and Liana D. Duenha
{erick.amorim, gregorio.neto,igor.mello,liana.duenha}@ufms.br

August 21, 2023

1 Installing

To use **MultiExplorer** will you need the source code, a pre-compiled version of the **Sniper** simulator, **Docker** and **XLauncher**.

1. **MultiExplorer**'s source code.
   https://github.com/lscad-facom-ufms/multiexplorer
2. **Sniper**'s pre-compiled version.
   https://drive.google.com/file/d/1aXNxy6OZ7NjP1XUgnhOGuTFAUePtwZkW/view
3. **XLauncher**.
   https://sourceforge.net/projects/vcxsrv/
4. **Docker**.
   https://docs.docker.com/desktop/install/windows-install/

1.1 Source Code

MultiExplorer is open-source, the code is freely available at https://github.com/lscad-facom-ufms/multiexplorer. Becoming a contributor is also possible.

1.2 Sniper

After downloading **Sniper**'s pre-compiled version, unzip the whole folder under the source code's root.

1.3 XLauncher

This tool will allow Docker Desktop for Windows to display MultiExplorer's graphical user interface. Download, install and launch it - you will have to go through a prompt wizard. In step 1, there are multiple options for display setting: "Multiple windows" and an unlimited "Display number" (-1) is recommended.

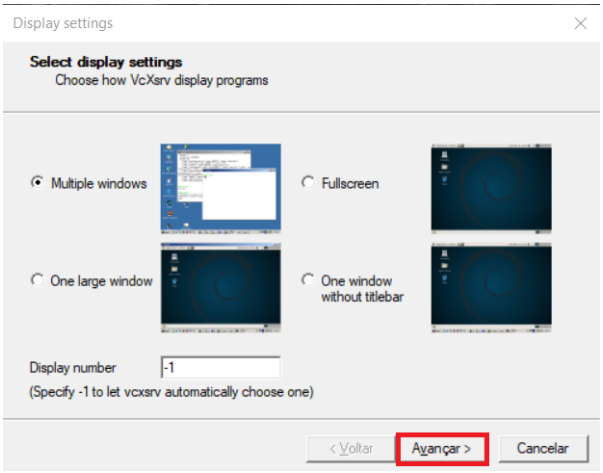

Figure 1: Select display settings. "Multiple windows" is recommended. Keep "Display number" value's at -1 to not limit the number of windows that can be launched.

In step 2 pick "Start no client" and click in "Advance".

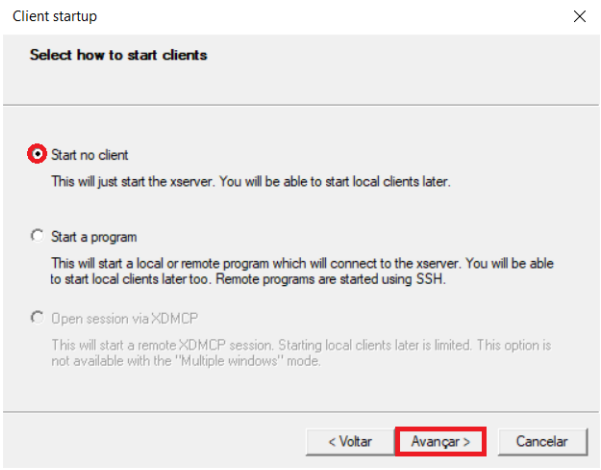

Figure 2: Select how to start clients. "Start no client" is recommended.

In step 3, you must check "Disable Access control". If you don't, the GUI won't work.

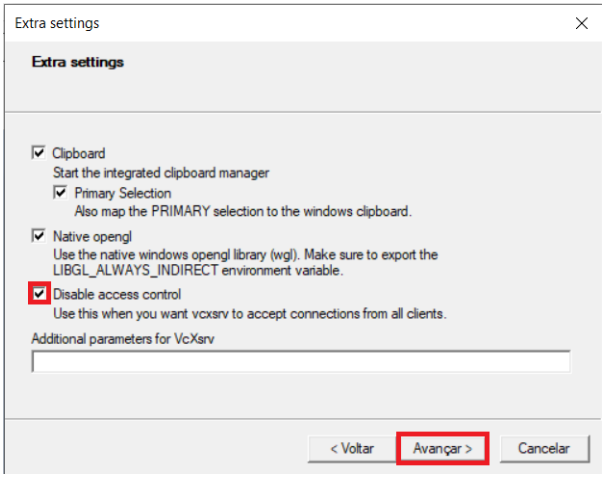

Figure 3: Extra settings. Disabling access control is required to use MultiExplorer's GUI from Docker.

In step 4 you can save your settings (optional) and finish this process.

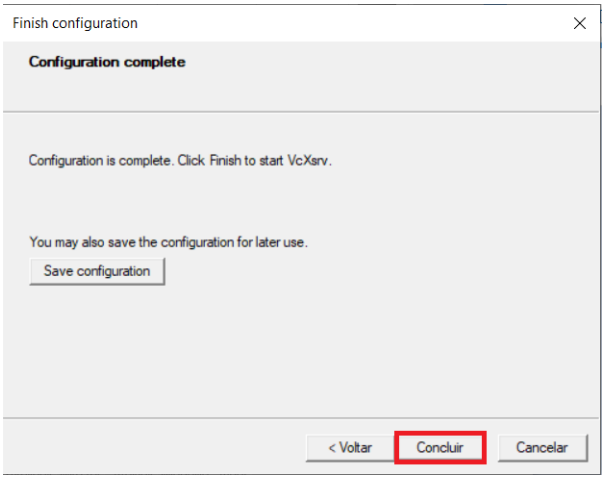

Figure 4:

XLauncher must be running whenever you want to use MultiExplorer's GUI.

1.4 Docker

This tool allow the establishment, upkeep and sharing of a homogeneous environment for the development, testing and/or execution of an application. If you don't use it, you must manually set MultiExplorer's requirements in a Linux based OS.

Once **Docker** is running, acquire your machine's local network address (IPV4):

$$ipconfig \tag{1}$$

If there's no ".env" file in the source code's root folder, create it. This environemnt file must contain the following line:

$$DISPLAY = "IPV4" : 0.0 \tag{2}$$

Replace "IPV4" with your machine's local network address, acquired in the previous step. Afterwards, go to the source code folder:

$$cd\ path\_to\_multiexplorer \tag{3}$$

Then, procceed with the following line command:

$$docker - compose\ up\ -d \tag{4}$$

This will build the preset **Docker** environment. Take note of the container's name. The next step is to run:

$$docker\ exec\ -it\ "container - name"\ bash \tag{5}$$

Replace "container-name" with the actual container name generated by docker when the preset environment was built. The above line will start a command-line interface inside the preset **Docker** environment. Then, procceed with:

$$make\ install \tag{6}$$

This line will use MultiExplorer's construction script (GNU Make [1]) to prepare the software requirements. The following line will finish the settings:

$$make\ config \tag{7}$$

After this, the GUI should be ready to use. To launch it, **XLaucher** must be running, then you use the command:

$$python\ ME.py \tag{8}$$

This way, the GUI should open, and you can start to use MultiExplorer.

2 Using MultiExplorer

In this section we will approach how to make pratical use of MultiExplorer.

[1] https://www.gnu.org/software/make/

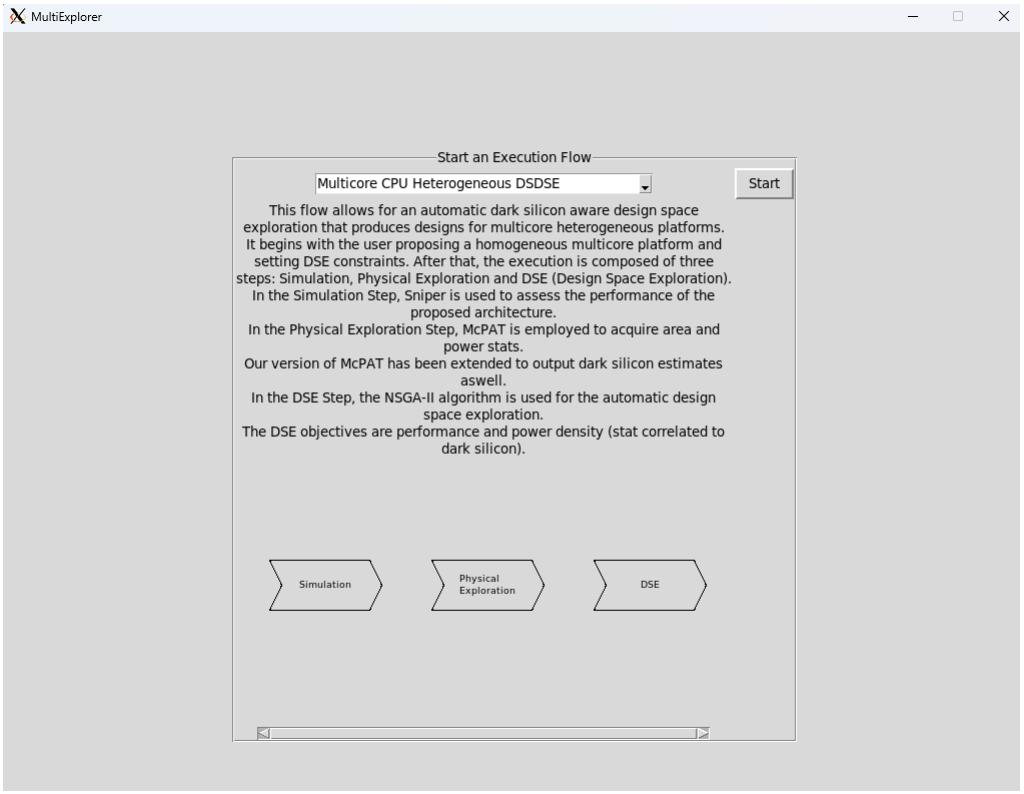

Figure 5: MultiExplorer's initial window. Select an Execution Flow and press "Start".

In figure 5, you will see the first step is to select an "Execution Flow". For learning purposes, select "Multicore CPU Heterogeneous DSDSE".

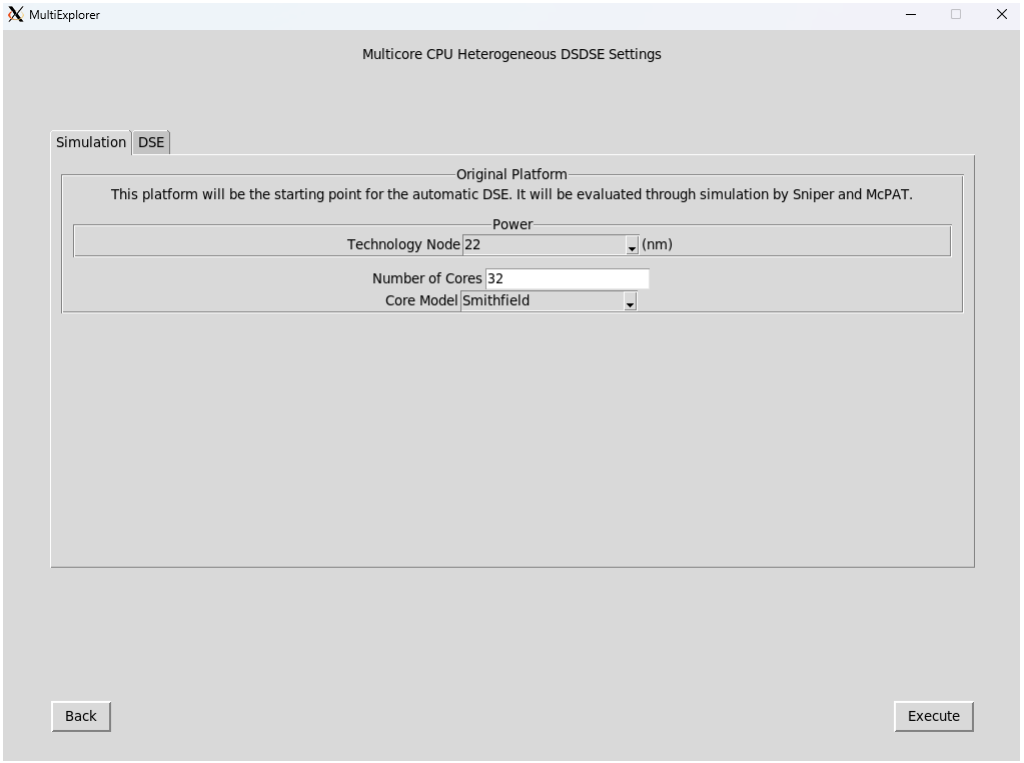

Figure 6: Platform description consist of technology node (lithograph), the number of cores, and the selection of a core model.

In figure 6, we can see the first user inputs. These inputs refer to the description of a homogeneous multicore platform. The user describes the platform in a high level of abstraction, merely selecting lithography, number of cores and core model. Each core model features a preset architecture specification that can be found in the folder "input-examples", in the form of JSON files.

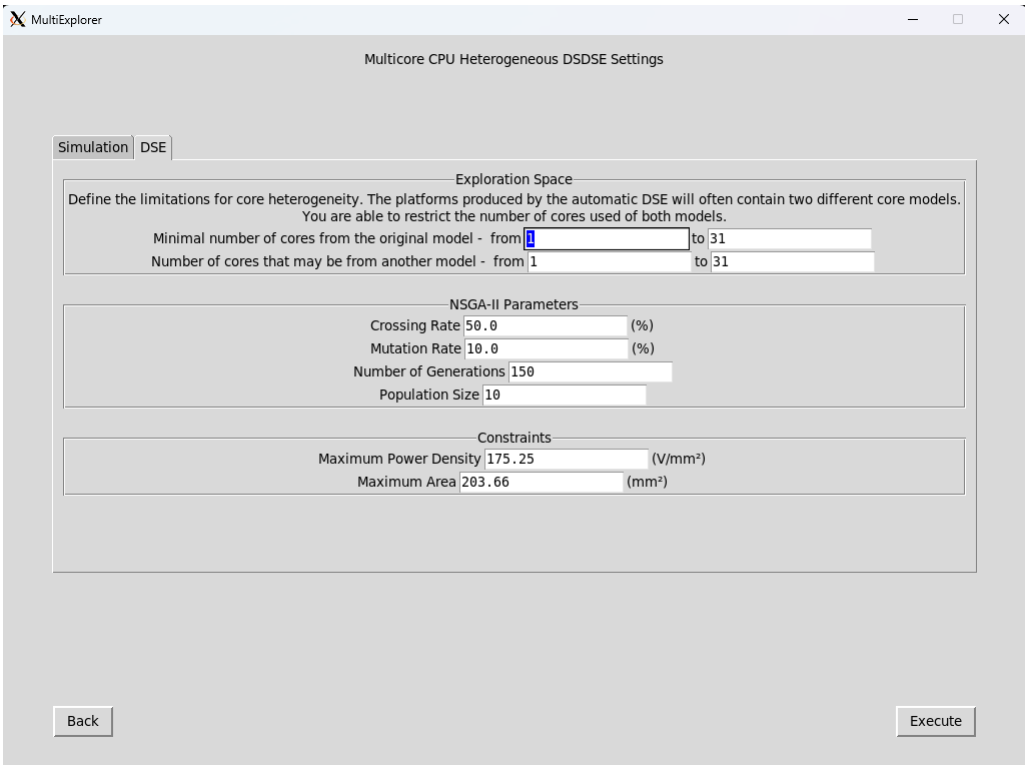

Figure 7: Design Space Exploration settings. Quantity of cores, NSGA-II calibration parameters, and DSE constraints are found here.

Figure 7 show a second tab of user inputs. In this tabe the user can calibrate the DSE algorithm, set goals and constraints.

The two inputs at the top are intervals (min-max) indicating how many cores the algorithm can allocate in a single multicore platform. Since the algorithm generates heterogeneous multicore platforms, you must set quantities for both the original core ("Core model", selected in the other tabe of user inputs), and for the alternative core model.

This tab also allows for some calibration parameters for NSGA-II to be changed. Usually, keeping the original parameters is recommended, unless you know what you are doing. Higher values for "Number of Generations" and "Population Size" can increase execution time exponentially.

Constraints for the generated multicore platforms are found at the bottom.

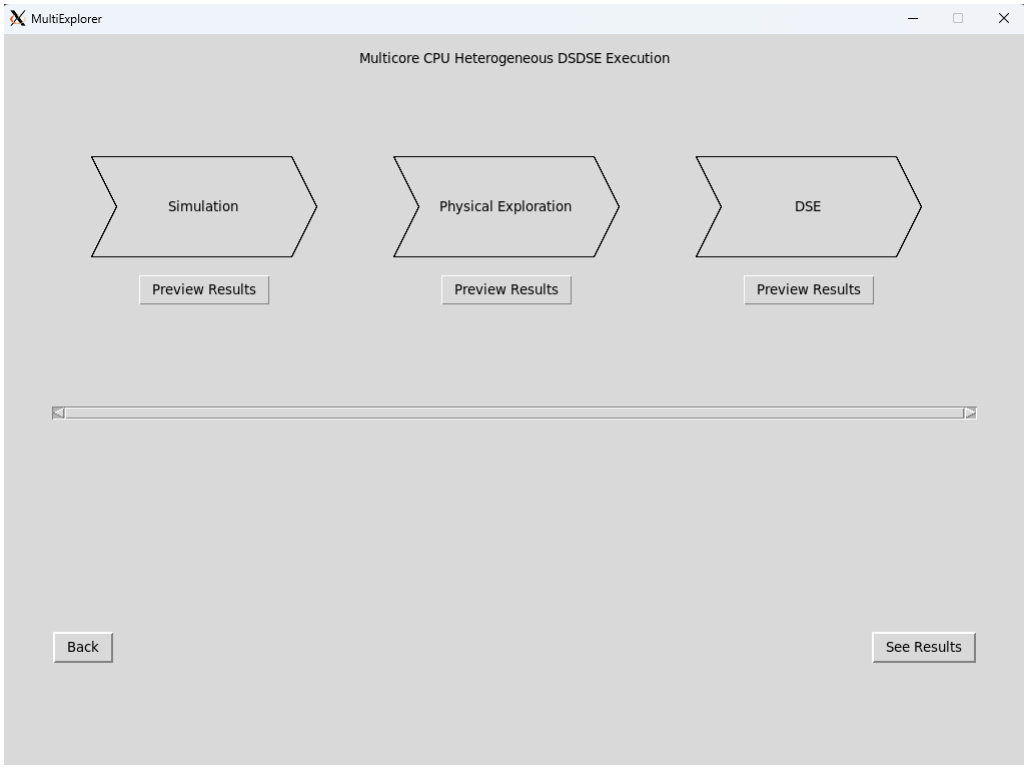

Figure 8: MultiExplorer's Flow Execution. The progress of the simulations and design space exploration can be tracked in this window. Partial results might be available for each step finished.

After the user finishes the settings and press the "Execute" button, the tool will go to the execution window (figure 8). Here the progress can be tracked, and after all steps are done, the user can use the button "See Results" to navigate to the results' window.

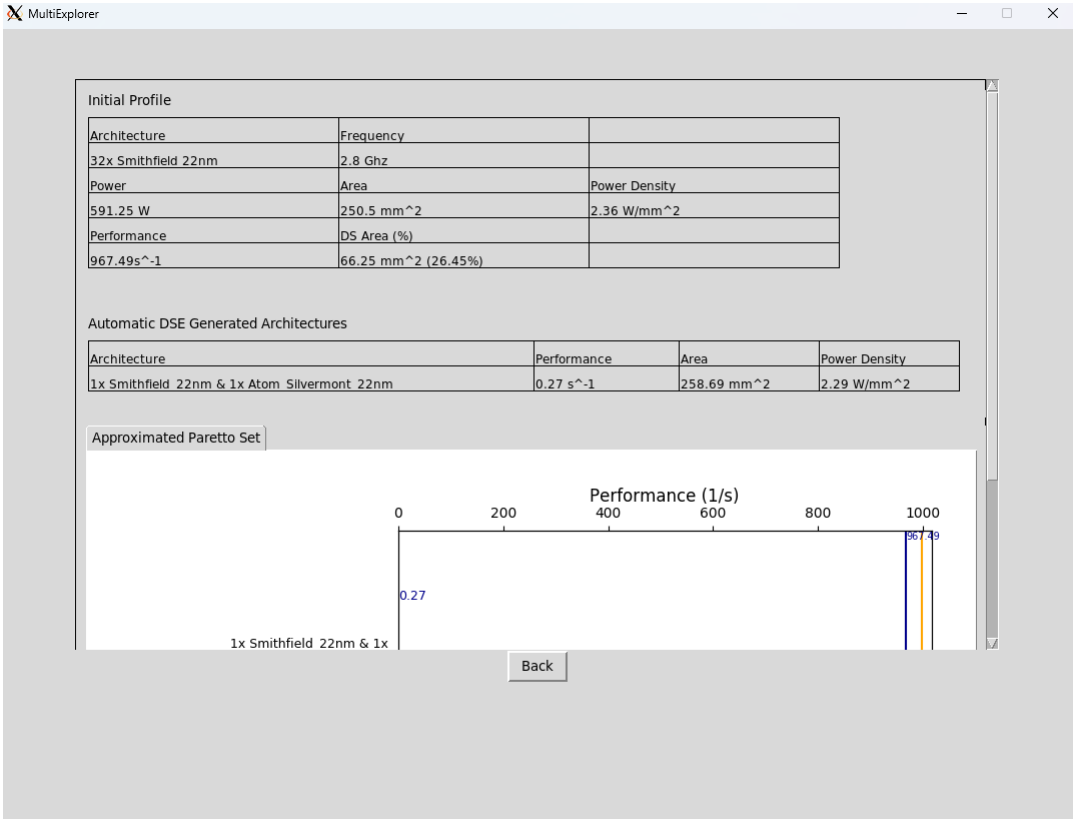

Figure 9: Results' Window. A profile of the user-specified homogeneous multicore platform can be found in the table at the top. Data on the heterogeneous multicore platforms generated by the algorithm can be found at the second table, and in the plot at the bottom.

The plot in the result's window (figure 9), presents performance and power density for each heterogeneous multicore platform in the approximated Pareto set found by the algorithm. These settings and results are also available in the form of files in the folders under "*/rundir/Multicore_CPU_Heterogeneous_DSDSE*".