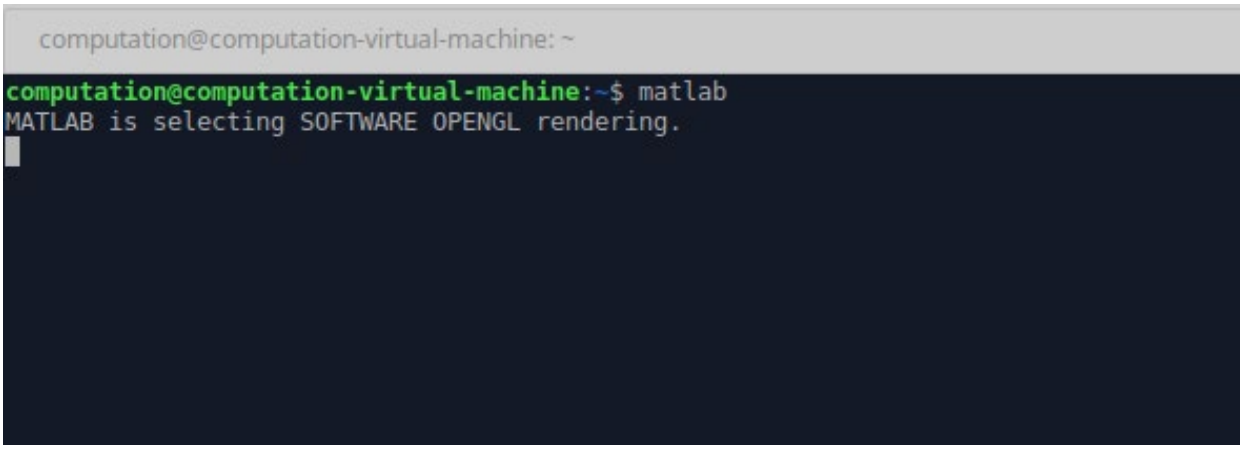
IMACS User Guide for Processor-in-the-Loop (PiL) Simulation

Section I: PiL implementation for sequential configuration

- open MATLAB: To open MATLAB in the IMACS virtual machine, run the following command in the new terminal.

  Note: If MATLAB is already opened, then no need to reopen it.

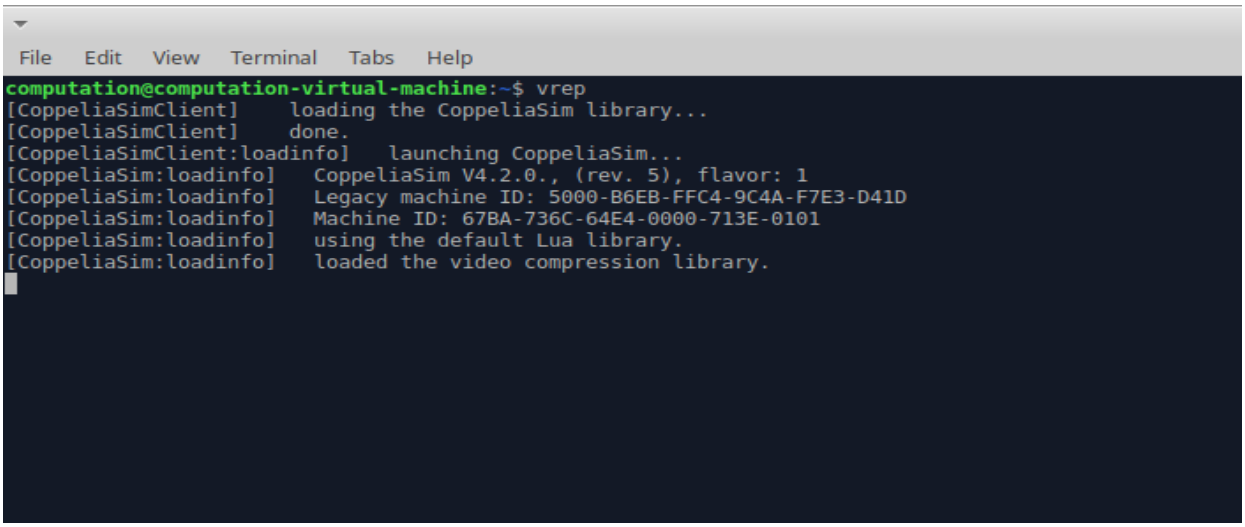
  ```
  $ matlab
  ```



- open CoppeliaSim: to open the CoppeliaSim, run the following command in a new terminal.

  Note: If CoppeliaSim is already opened, then no need to reopen it.

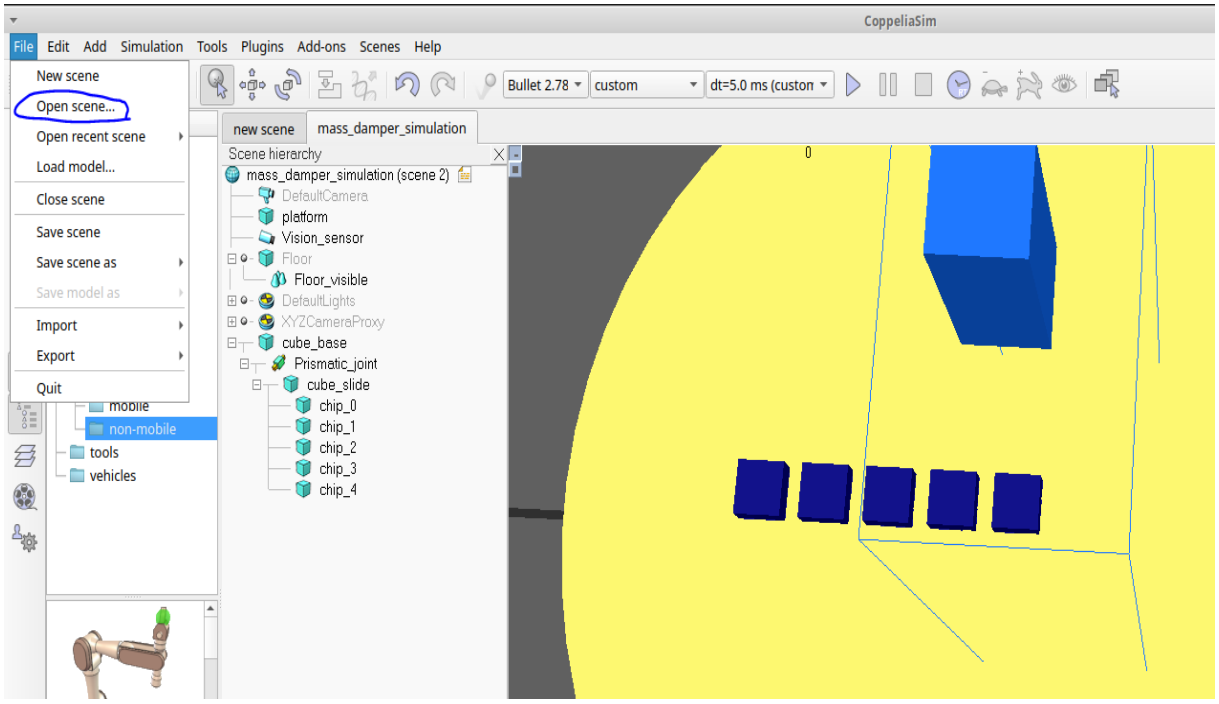
  ```
  $ vrep
  ```

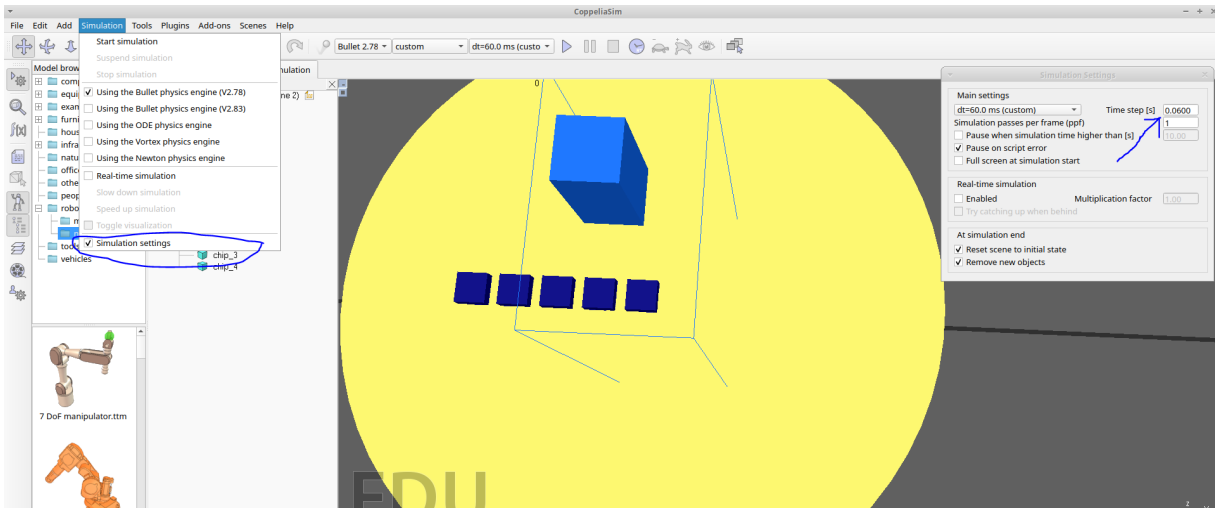


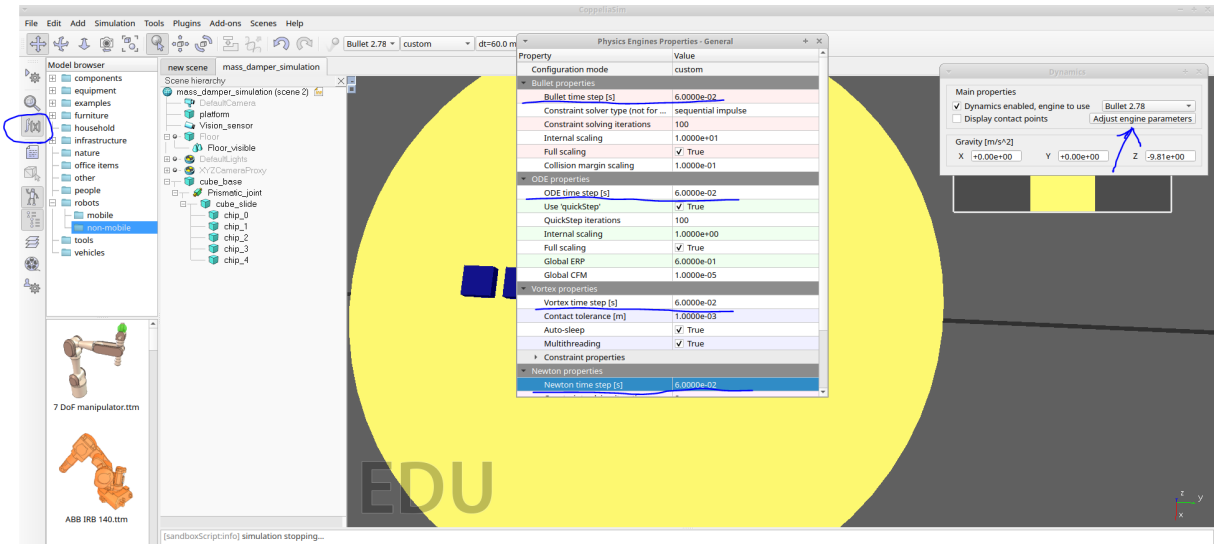
- open CoppeliaSim scene for the mass-damper system.
  - go to File→Open scene→ select the following directory /home/computation/5lij0_ECS_Course_2023/PIL_Implementation/PIL_sequential_impl

open **mass_damper_simulation.ttt**, check the following screenshot for reference.

We have implemented the PIL of sequential configuration considering the 60ms sample period. The same sample period should be used in CoppeliaSim. In the case of different sample periods, students can make the changes in the CoppeliaSim, as explained below. For example, the following screenshot shows how to change the time step in the CoppeliaSim scene.

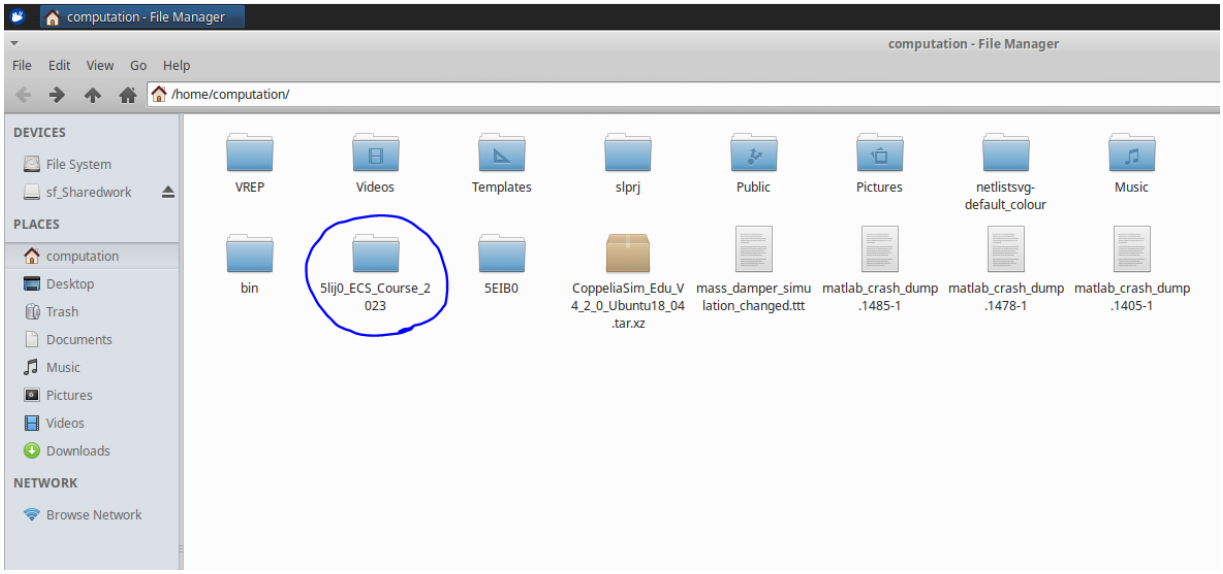


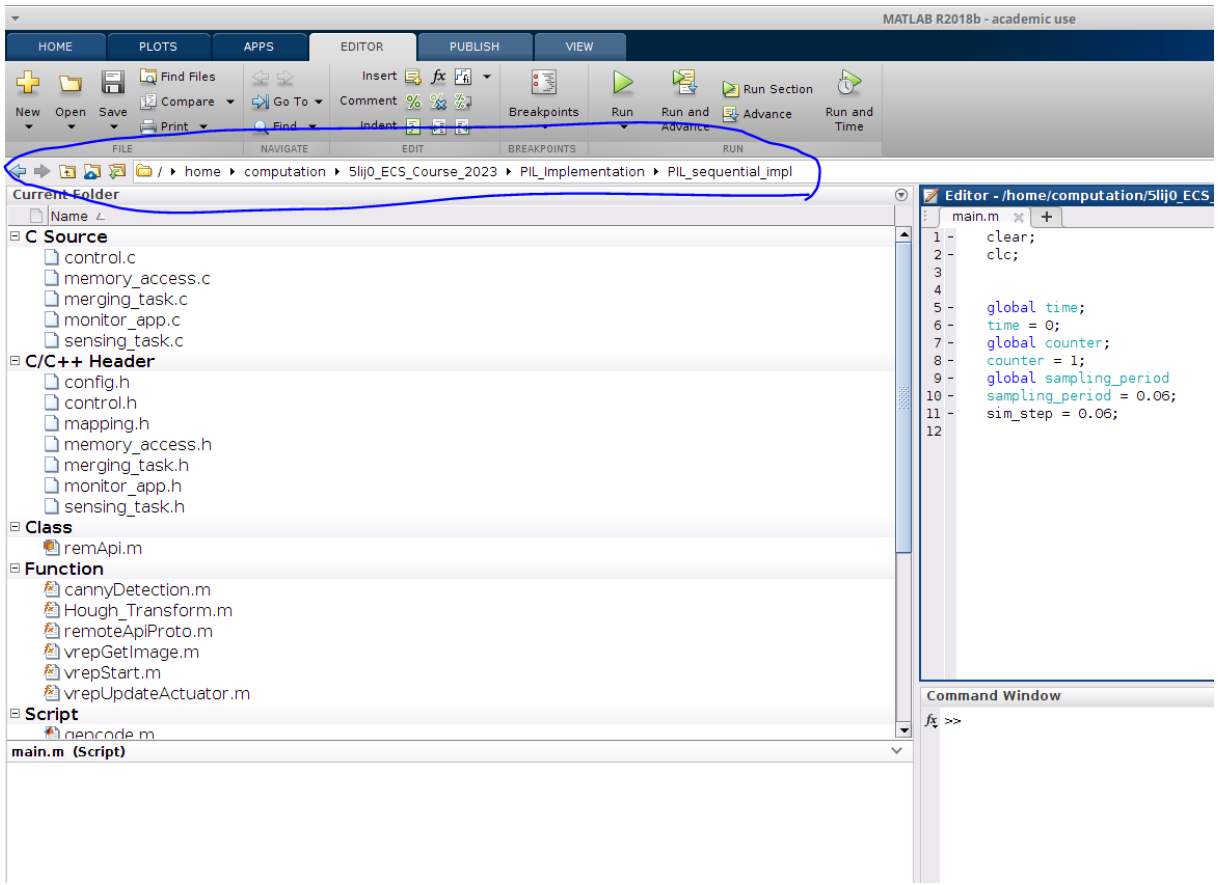




- open the folder for PiL sequential implementation.
  - go to the following folder path and open in MATLAB

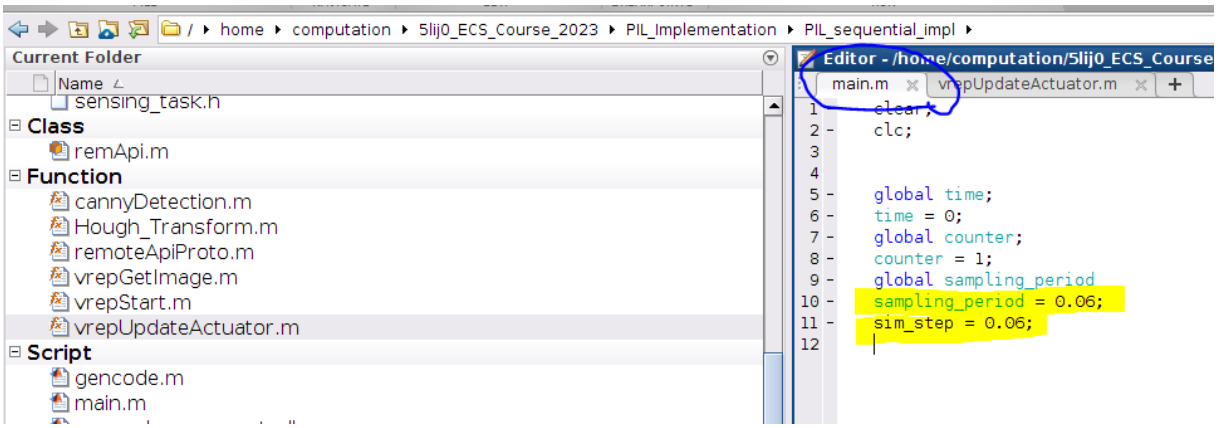
    computation→5lij0_ECS_Course_2023→PIL_Implementation→PIL_sequential_impl



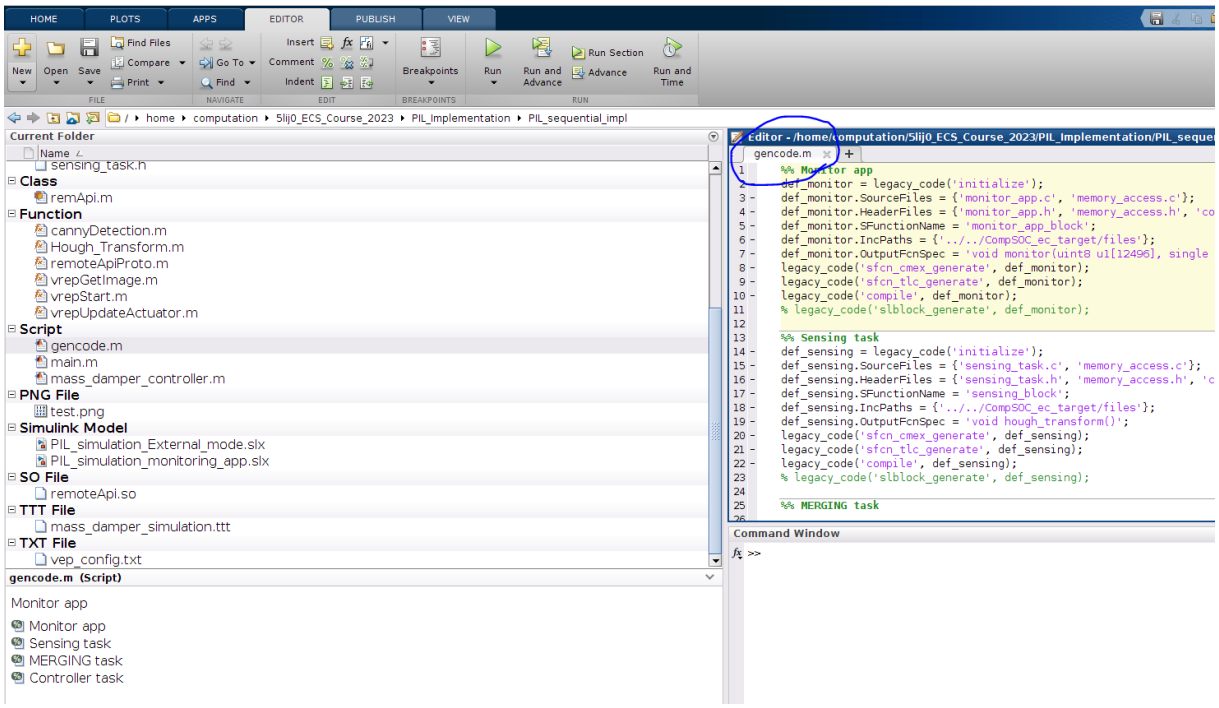


- open the main.m file and make changes for the highlighted parameters, as shown in the following screenshot.

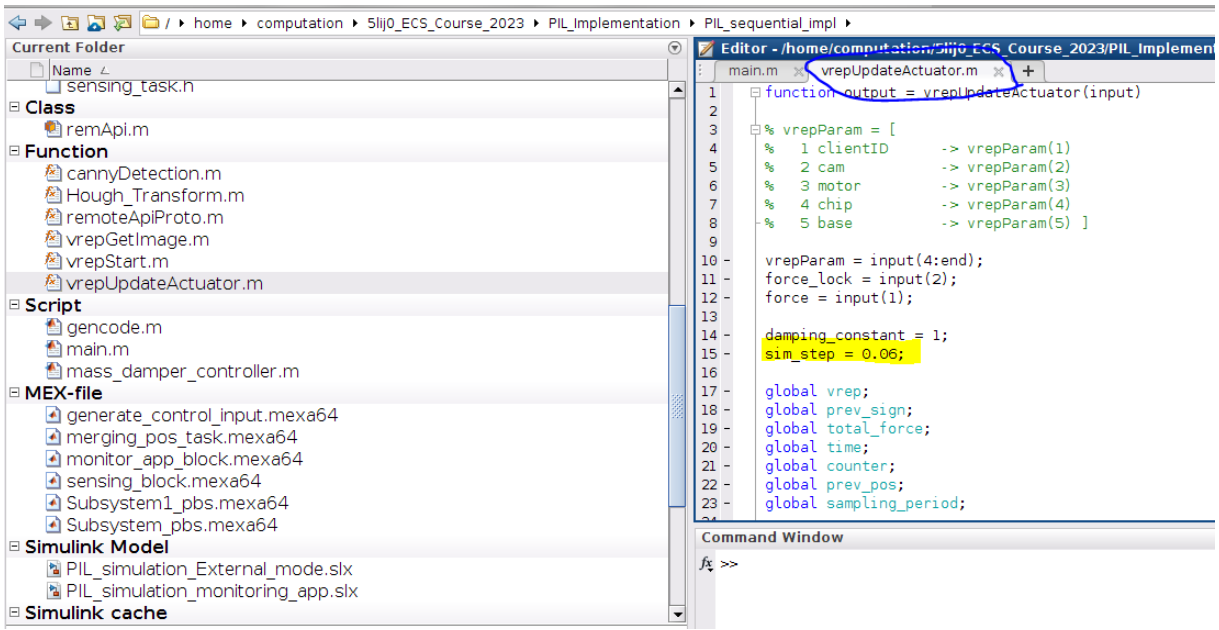
  Note: In this screenshot, we have explained for the 60ms sample period, students can use different sample periods according to the implementation design and make the changes for the following parameters. Once the changes are done, run the main.m script. This file includes the initialization, which is required for the PiL simulation.



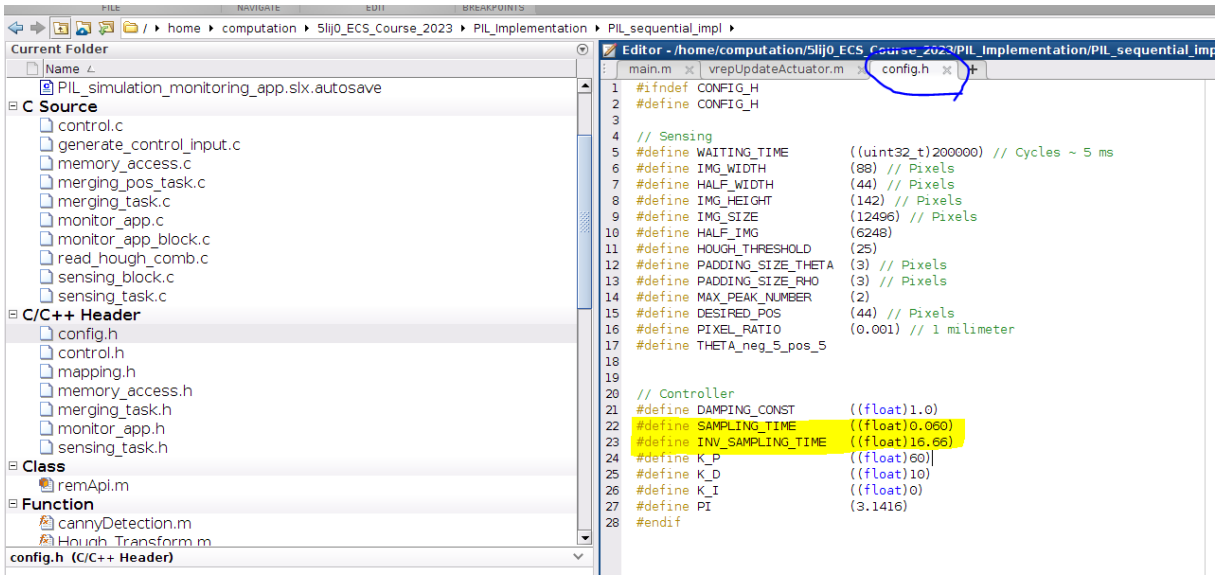
- open the gencode.m file and run the script. This file uses legacy code tool functionality, integrating the existing C or C++ functions into MATLAB Simulink. Once this gencode.m runs, it creates the S-function block inside the Simulink model.



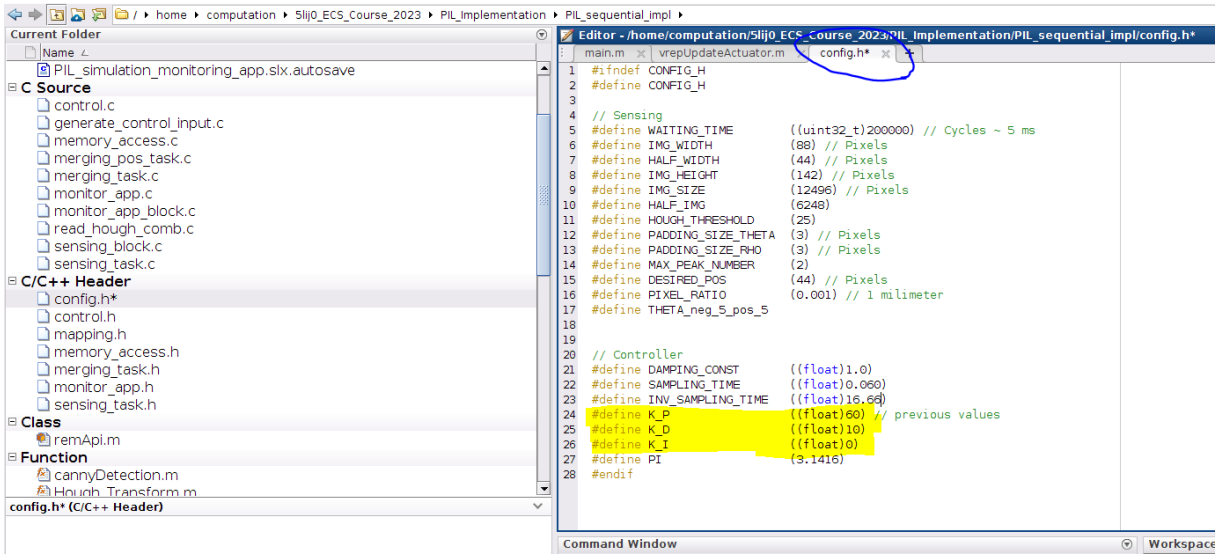
- open the **vrepUpdateActuator.m** file. Make the necessary changes in the file, as shown below in the screenshot. This file is required for the actuation.



- open the **config.h** file. Make the necessary changes in the file, as shown below in the screenshot. This file contains all the required parameters used in the PiL simulation.



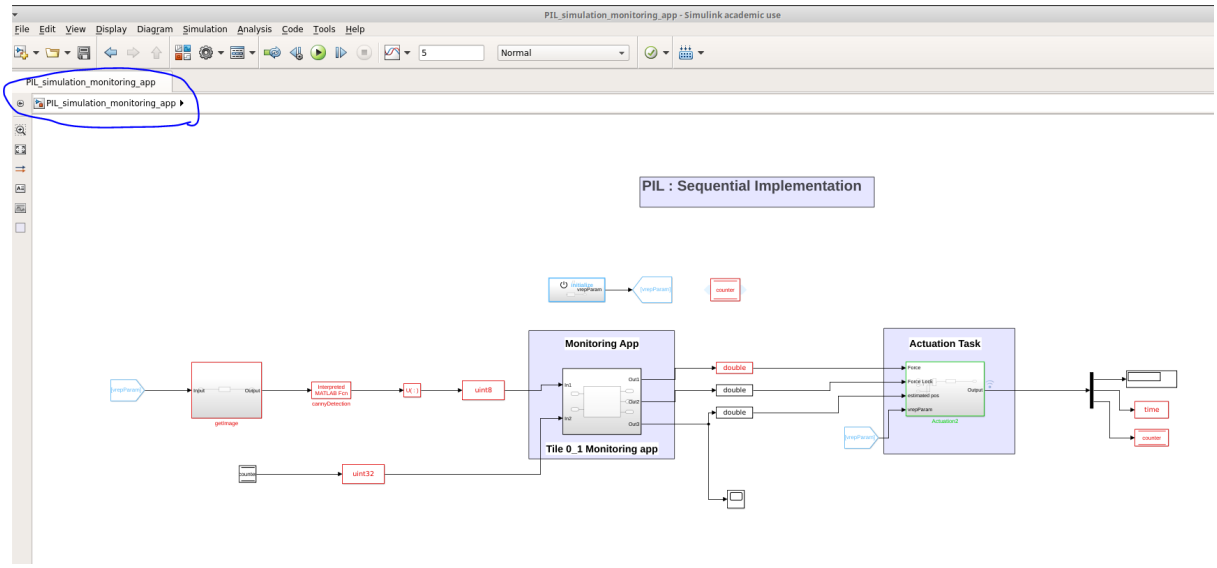
- for tuning the PID controller gains, students can change the gains in the **config.h** file, check the below screenshot.



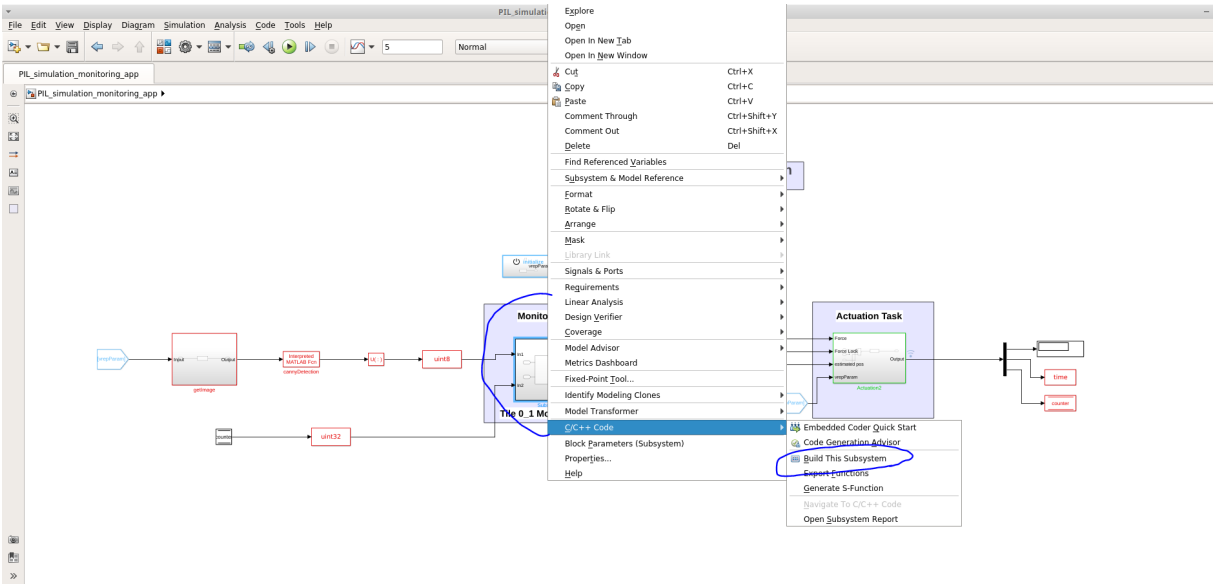
- once the steps mentioned above are done, open the PiL simulation file named **PIL_simulation_monitoring_app.slx**

  Students can find this file in the following directory:

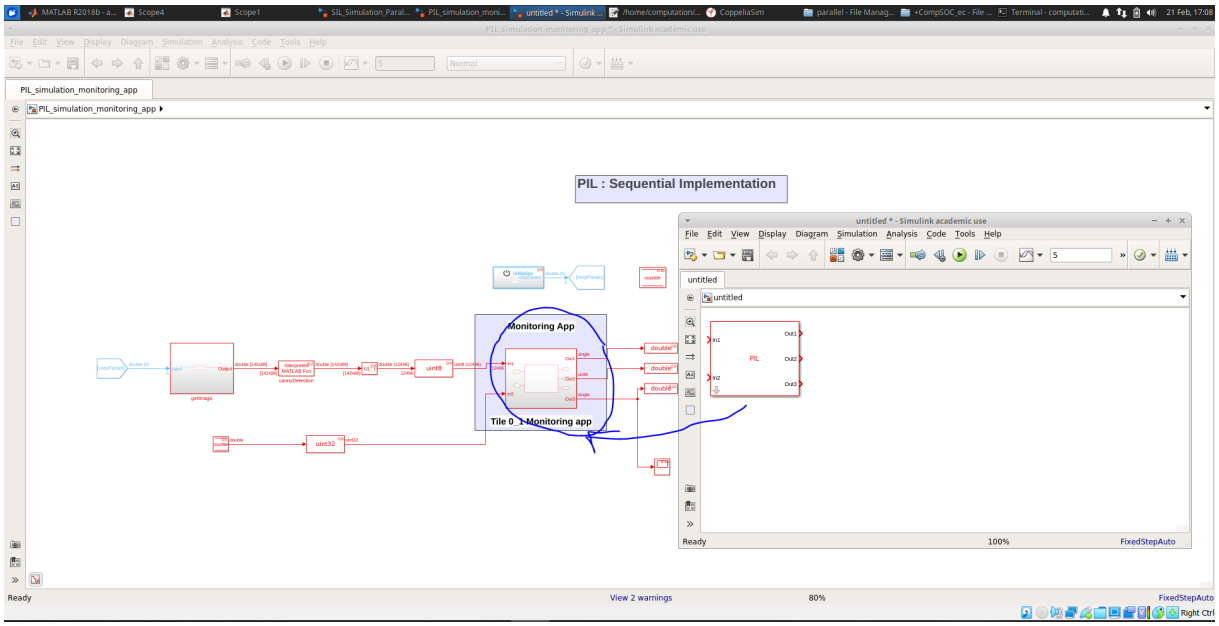
  computation→5lij0_ECS_Course_2023→PIL_Implementation→PIL_sequential_impl



- right-click on the block, select the C/C++ code option, and click on build this subsystem. Check the following screenshot for reference.



- once, the build process is completed successfully; it will generate the PIL block; check the following screenshot:



- after completing the above steps, check the vep_config.txt file. In this example, we have considered the 60ms sample period. Therefore, in the vep_config.txt, Time-division multiplexing (TDM) scheduling is given according to the 60ms sample period.
- we have provided the vep_config.txt file in the following folder

  /home/computation/5lij0_ECS_Course_2023/PIL_Implementation/PIL_sequential_impl
- students need to copy the contents from the above vep_config file into the main **vep-config.txt** file, which is inside the following folder path :

  or the student can enter the TDM scheduling calculation directly in the vep-config.txt file, which is stored in the following location.

  /home/computation/CompSOC_ec_target/CompSOC_ec/+CompSOC_ec/vep-config.txt

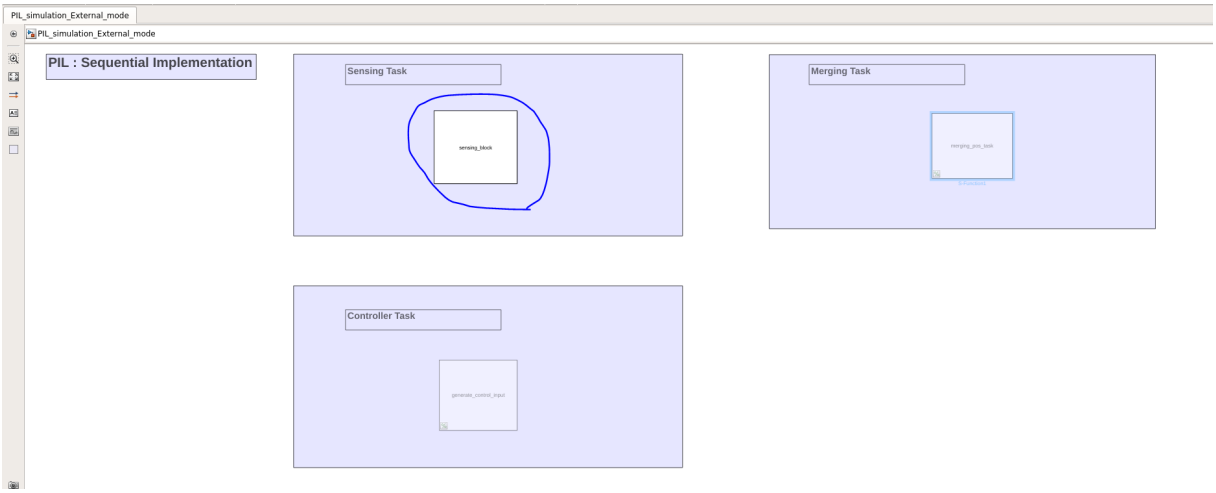
- the contents in the vep_config file should be as follows:

```
##
## you can comment out & modify the "on tile ... " lines
##
## - partitions can only have a memory size of 32K or 64K
## - the size of all partitions on a tile must be at most 96K
##   (i.e. 128K including the system application)
## - default memory is 32K, default stack is 4K
##
## use /opt/riscv/bin/riscv32-unknown-elf-size [-A] *.elf
## to analyse a partition's memory usage
##
## memory and stack allocation
##
on tile 0 partition 1 has 32K memory and 4K stack
on tile 0 partition 2 has 32K memory and 4K stack
on tile 0 partition 3 has 32K memory and 4K stack
on tile 1 partition 1 has 64K memory and 4K stack
on tile 1 partition 2 has 32K memory and 4K stack
#on tile 1 partition 3 has 32K memory and 4K stack
#on tile 2 partition 1 has 64K memory and 4K stack
#on tile 2 partition 2 has 32K memory and 4K stack
#on tile 2 partition 3 has 32K memory and 4K stack
##
## scheduling
##
## - max 3 slots per tile
## - it is allowed to not schedule anything on a tile
## - a partition can have more than one slot
## - the system partition always get a slot of 5000 cycles
##   at the end: you don't have to add this
##
#on tile 0 next slot is for partition 1 with 200000 cycles
on tile 0 next slot is for partition 1 with 1600000 cycles
on tile 0 next slot is for partition 2 with 10000 cycles
on tile 0 next slot is for partition 3 with 777000 cycles
on tile 1 next slot is for partition 1 with 1600000 cycles
on tile 1 next slot is for partition 2 with 789000 cycles
#on tile 1 next slot is for partition 3 with 372000 cycles
#on tile 2 next slot is for partition 1 with 700000 cycles
#on tile 2 next slot is for partition 2 with 289000 cycles
# on tile 2 next slot is for partition 2 with 10000 cycles
# on tile 2 next slot is for partition 3 with 100000 cycles
```

- open the PIL_simulation_External_mode.slx. The folder path to this Simulink model is as follows:

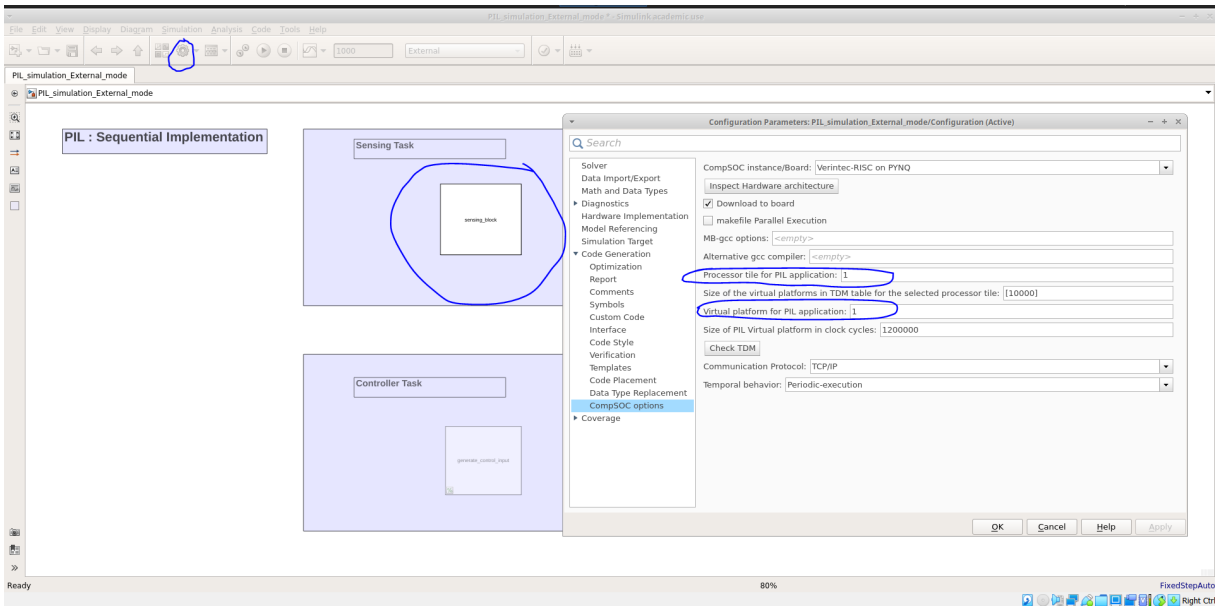
  /home/computation/5lij0_ECS_Course_2023/PIL_Implementation/PIL_sequential_impl
- PIL_simulation_External_mode.slx file has three different tasks, i.e., sensing, merging, and controller, which will run in the external mode on CompSOC based on the TDM scheduling. To run each task independently in the external mode, follow the below steps and check the screenshot for reference:

- to run the sensing task in the external mode, comment the other Simulink block and only uncomment the block which needs to be run. In the below screenshot, we consider the sensing task block first, and other blocks kept commenting.



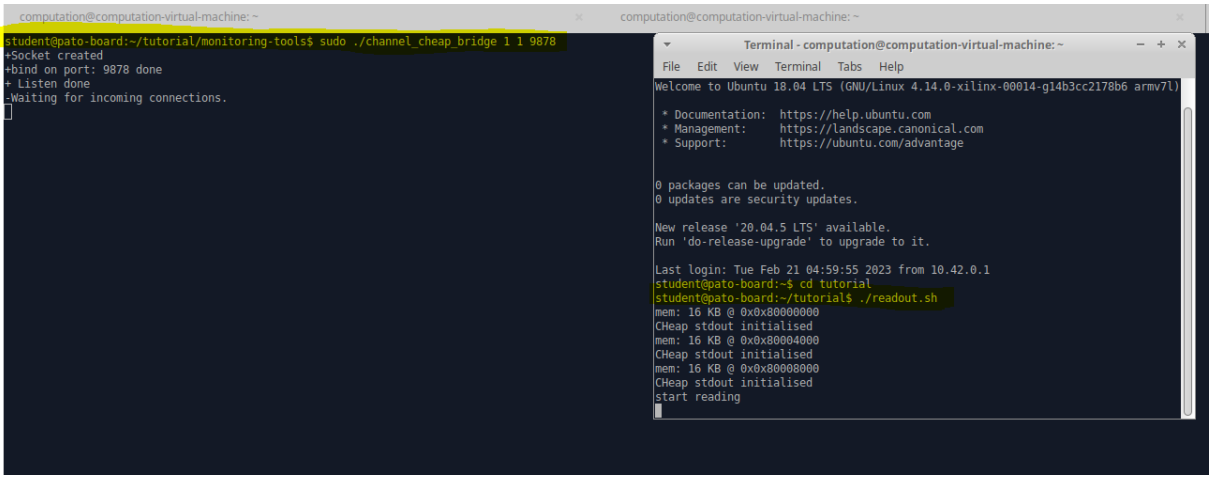
- open the configuration setting and check the required setting for the particular task. In this example, we are running a sensing task on processor 1 and partition 1. Therefore, consider the following changes :

  - Processor tile for PIL application: 1
  - Virtual platform for PIL application: 1

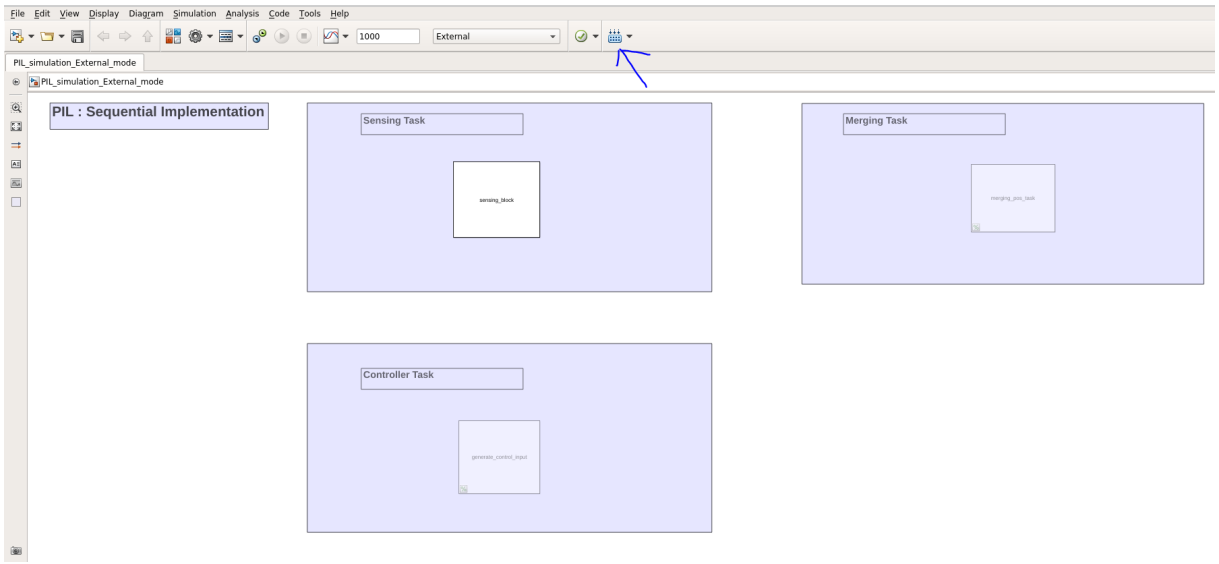


- open the two new terminal windows, connect the PYNQ board, and use the following command:
  - ensure that vep-config.txt is updated in the main CompSOC folder as per the required TDM scheduling. This step is essential before running the task in external mode on CompSOC.
  - vep-config file path: /home/computation/CompSOC_ec_target/CompSOC_ec/+CompSOC_ec/vep-config.txt
  - commands for running the sensing task on processor 1 and partition 1

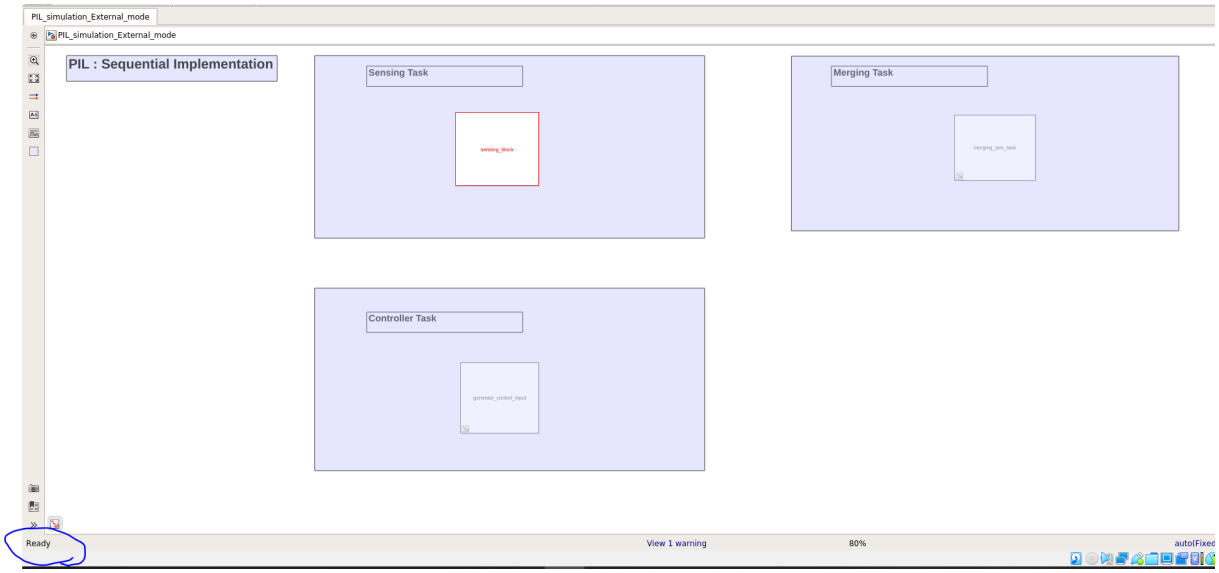
    - Terminal 1 :
      - cd tutorial
      - cd monitoring-tools/
      - sudo ./channel_cheap_bridge 1 1 9878
    - Terminal 2 :
      - cd tutorial
      - ./readout.sh



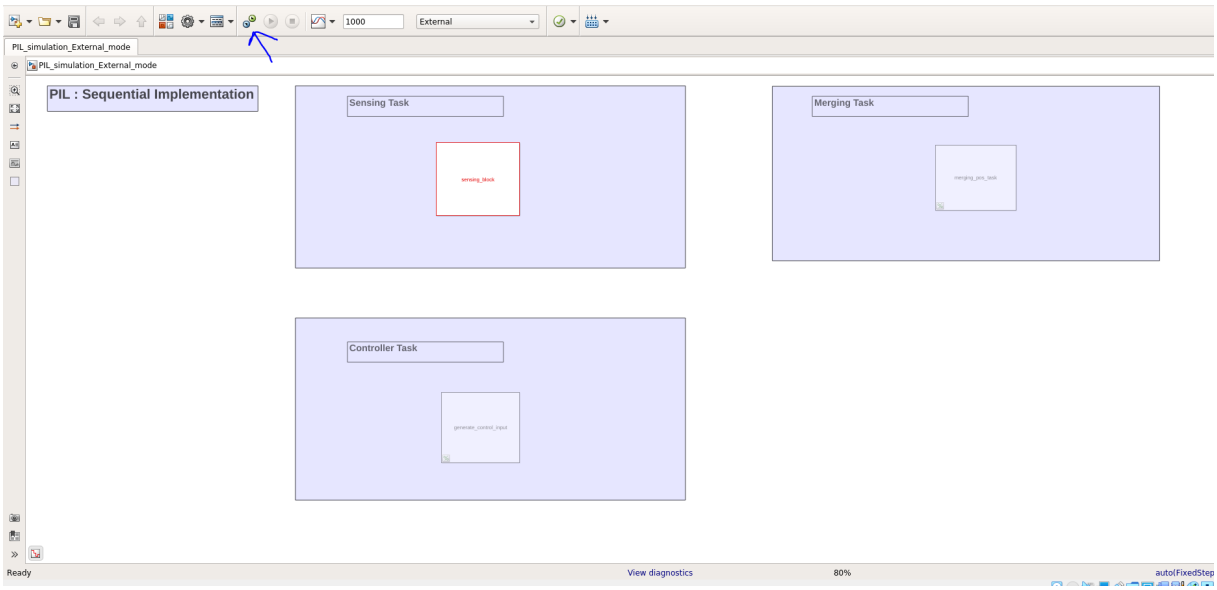
- click on the build button, as shown by the arrow in the below screenshot.



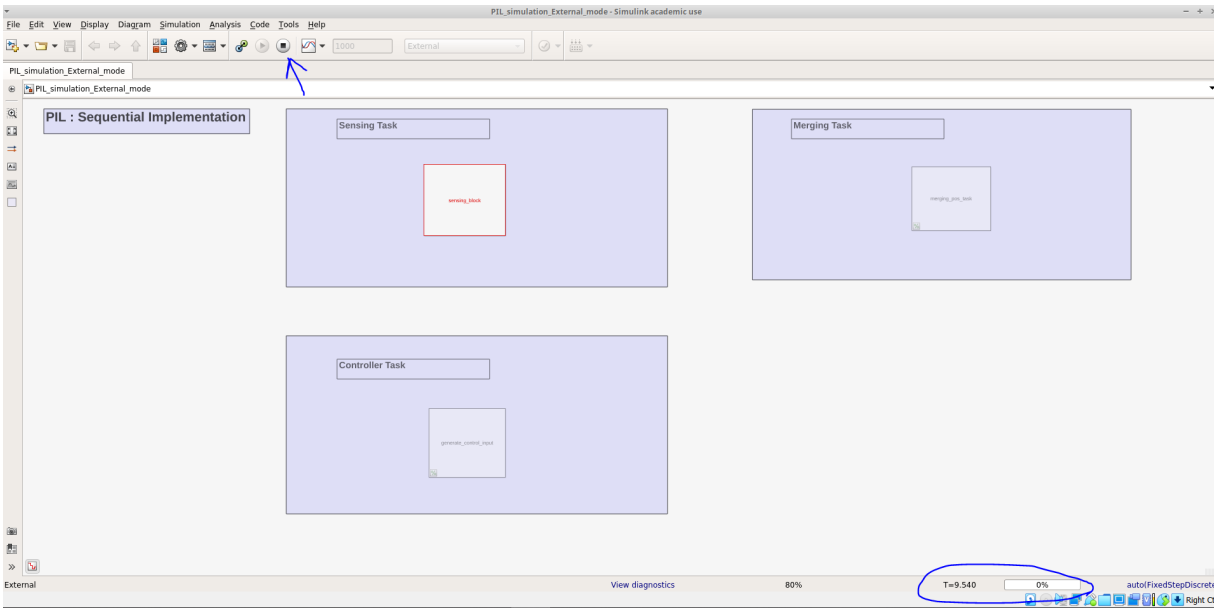
- after the build finishes, the Simulink model shows ready status; check the circle in the following screenshot.



- if the build finishes successfully, ready status is available in the model. After that, run the task in external mode by clicking the connect to target button. Check the following screenshot for reference.

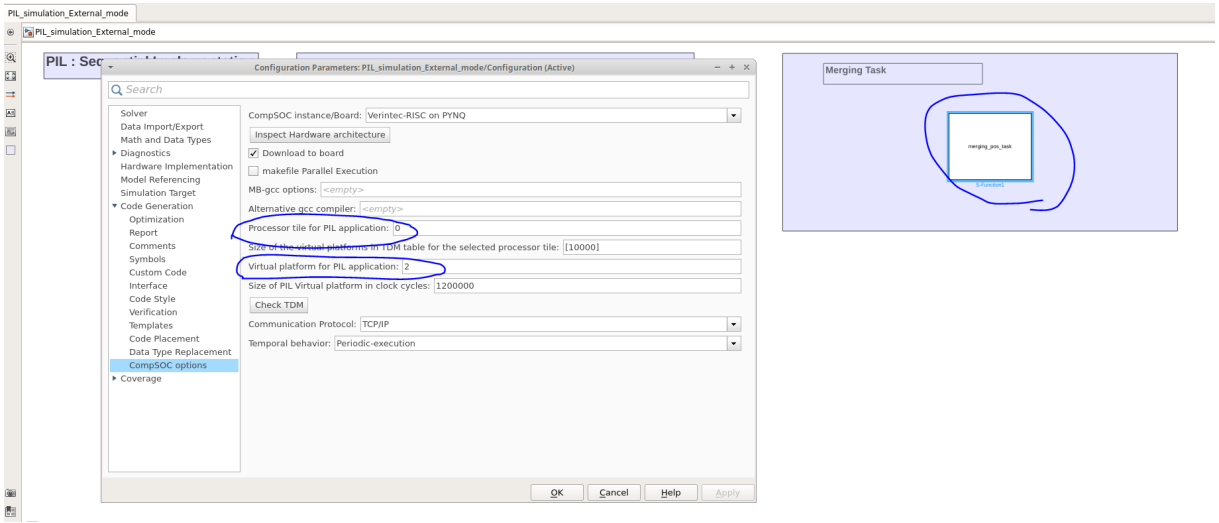


- after the above step, the sensing task connects to the target and starts running on the CompSOC; let it starts and runs for some time; check the highlighted circle in the below screenshot, then stop the running of the task by clicking on the stop button, check the arrow in the screenshot. No need to run the task for a long time.



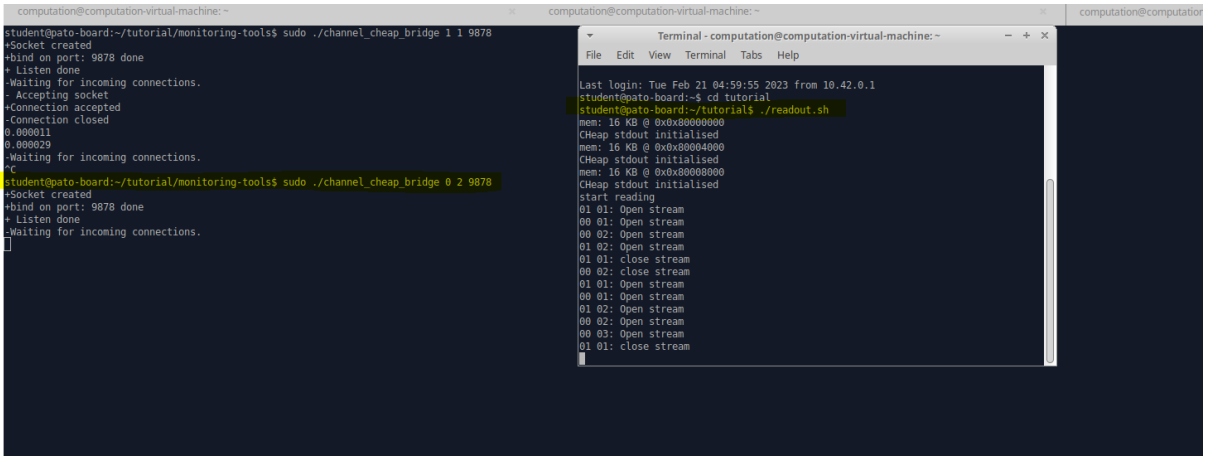
- move to the next task and comment all blocks except the one you want to run in external mode. Check the following screenshot; we have considered merging task after finishing the sensing task. Only the merging task is uncommented. Other blocks kept commenting.
- check the configuration setting. We are running the merging task on processor 0 and partition 2. So, the following setting should be considered,

  - Processor tile for PIL application : 0
  - Virtual platform for PIL application: 2

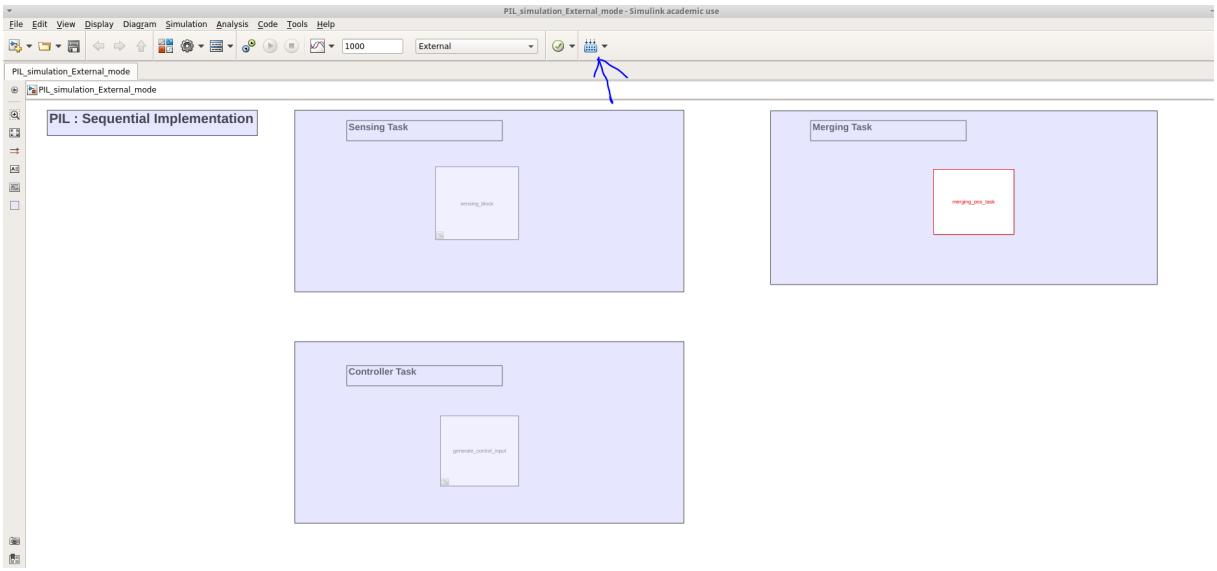


- in the same two terminal windows opened for the previous task, you can use the following highlighted commands from the screenshot.

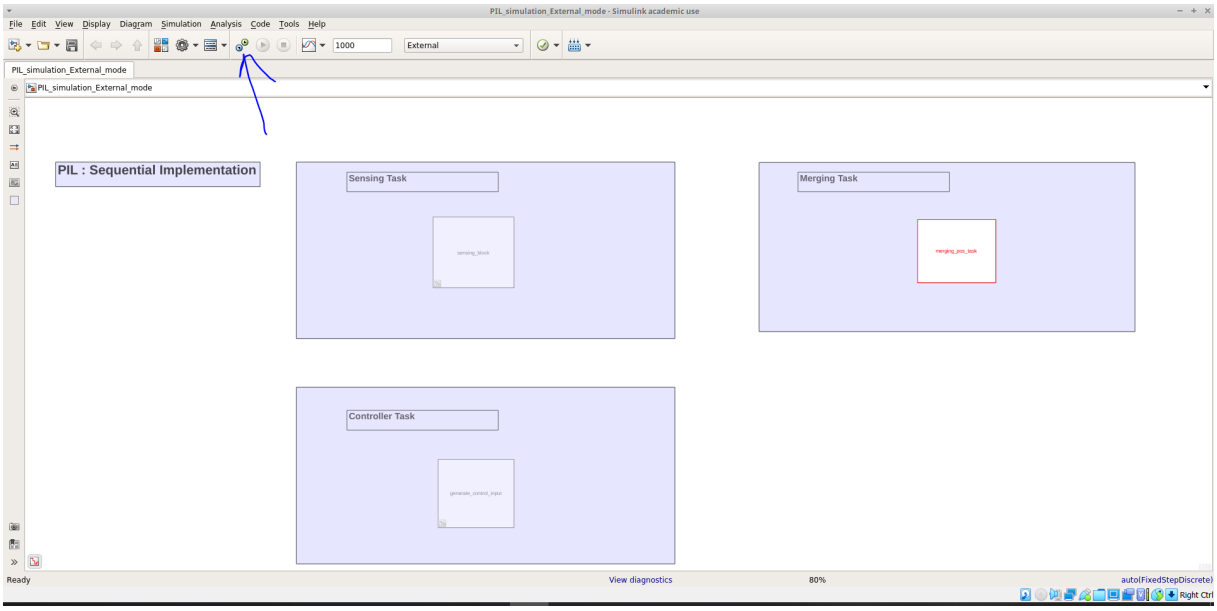
  - Terminal 1 :
    - cd tutorial
    - cd monitoring-tools/
    - sudo ./channel_cheap_bridge 0 2 9878
  - Terminal 2 :
    - cd tutorial
    - ./readout.sh



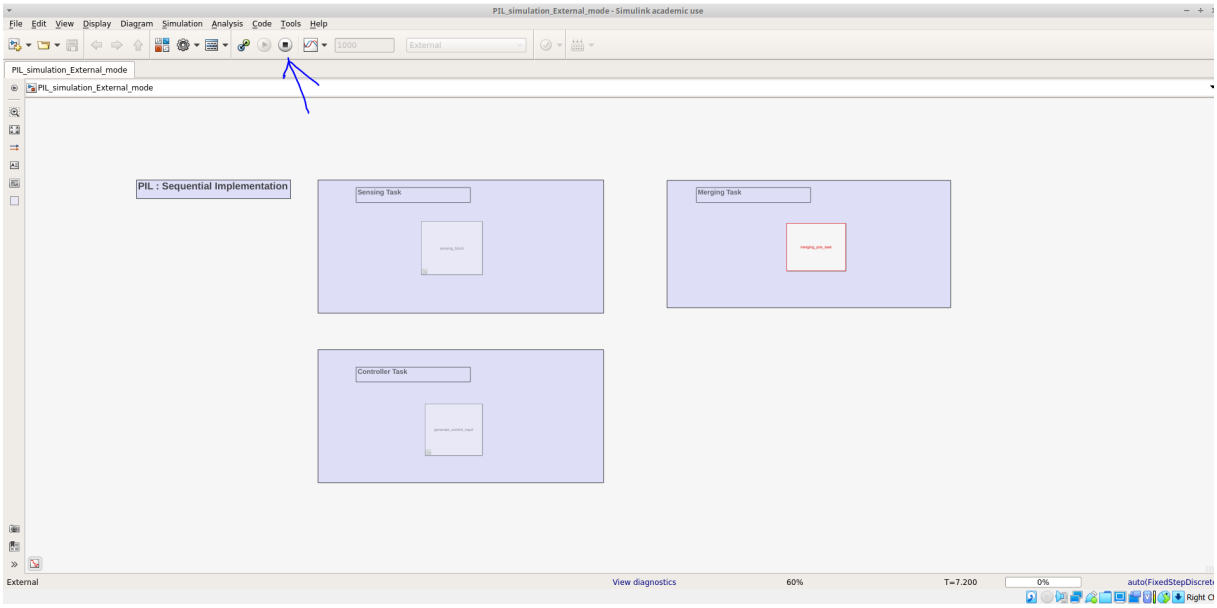
- click on the build button, as shown by the arrow in the below screenshot.



- if the build finishes successfully, ready status is available in the model. After that, run the task in external mode by clicking on the connect to target button. Check the following screenshot for reference.

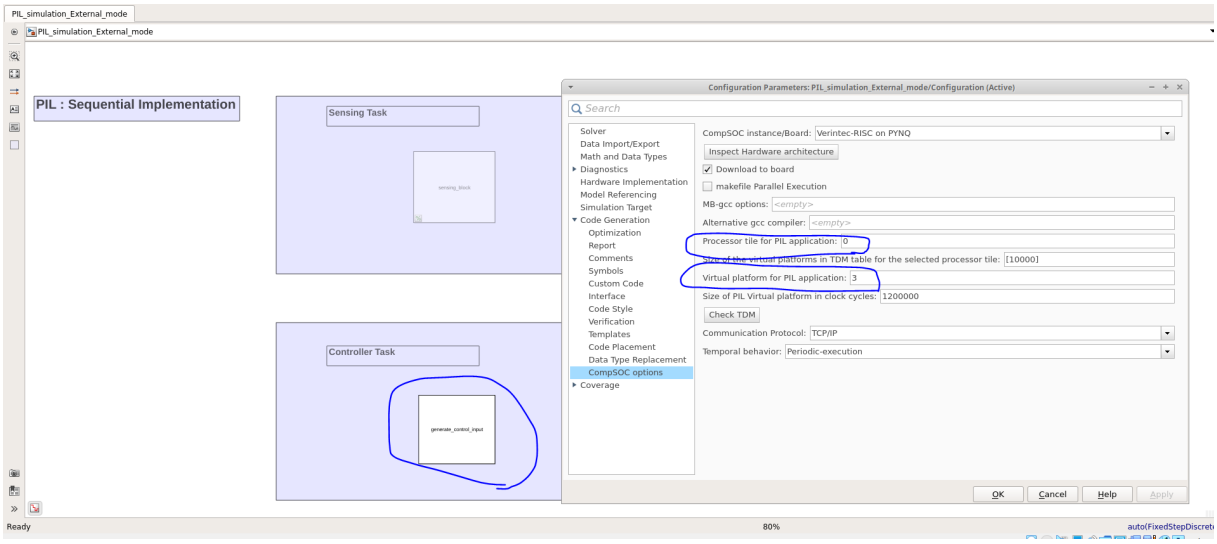


- after the above step, the merging task connects to the target and starts running on the CompSOC; let it starts and runs for some time, then stop the running of the task by clicking on the stop button. Check the arrow in the screenshot. No need to run the task for a long time.



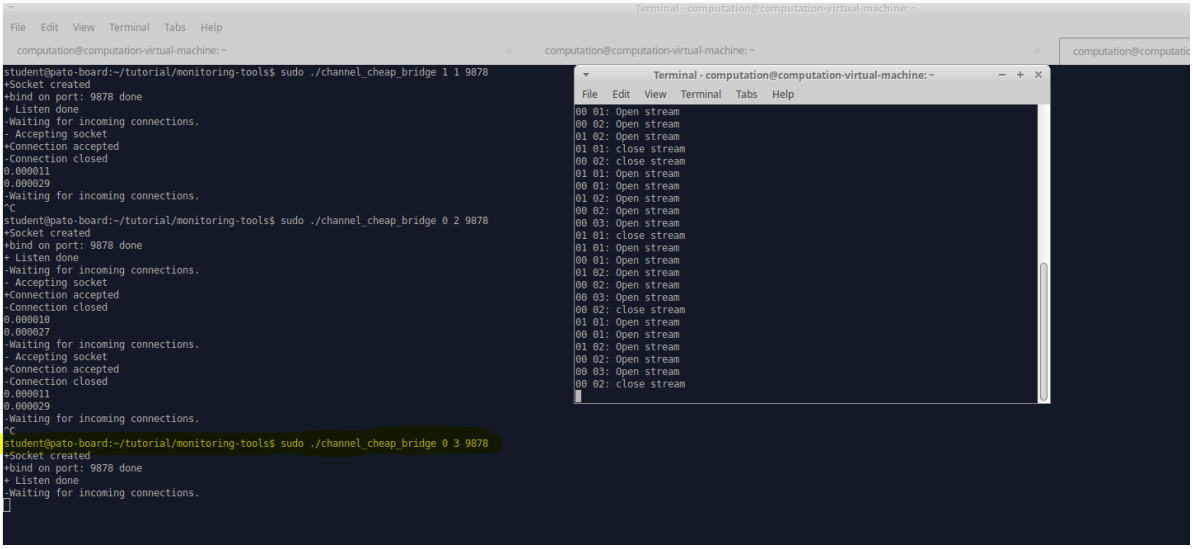
- move to the next task and comment all blocks except the one you want to run in external mode. Check the following screenshot; we have considered the controller task after finishing the merging task. Only the controller task is uncommented. Other blocks kept commenting.
- check the configuration setting. We are running the controller task on processor 0 and partition 3. So, the following setting should be considered,

  - Processor tile for PIL application : 0
  - Virtual platform for PIL application: 3

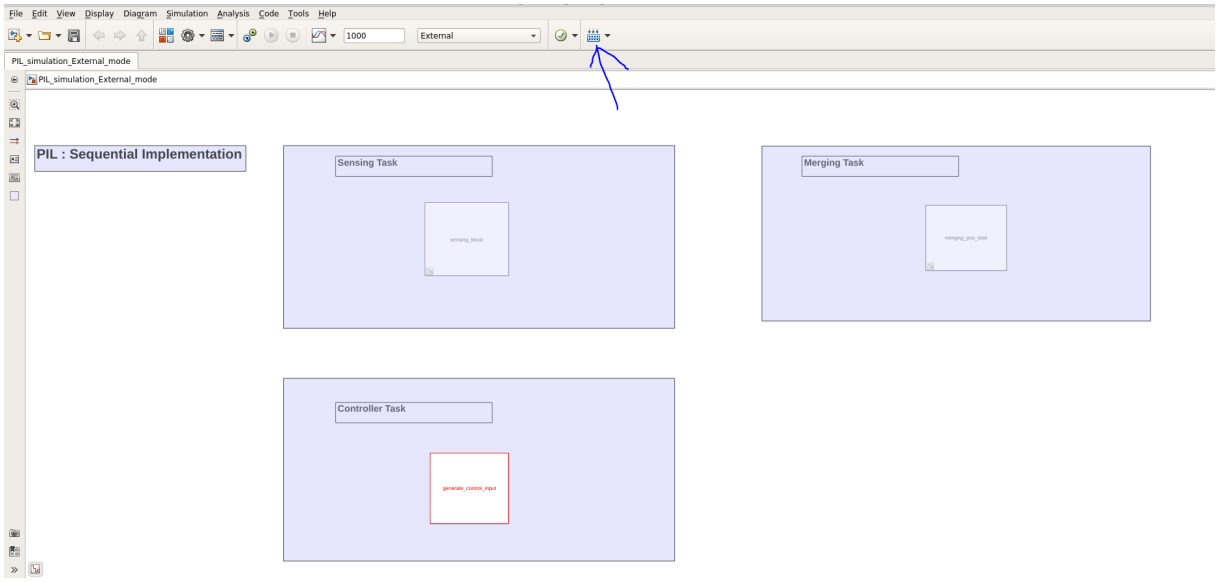


- in the same two terminal windows opened for the previous task, you can use the following highlighted commands from the screenshot.

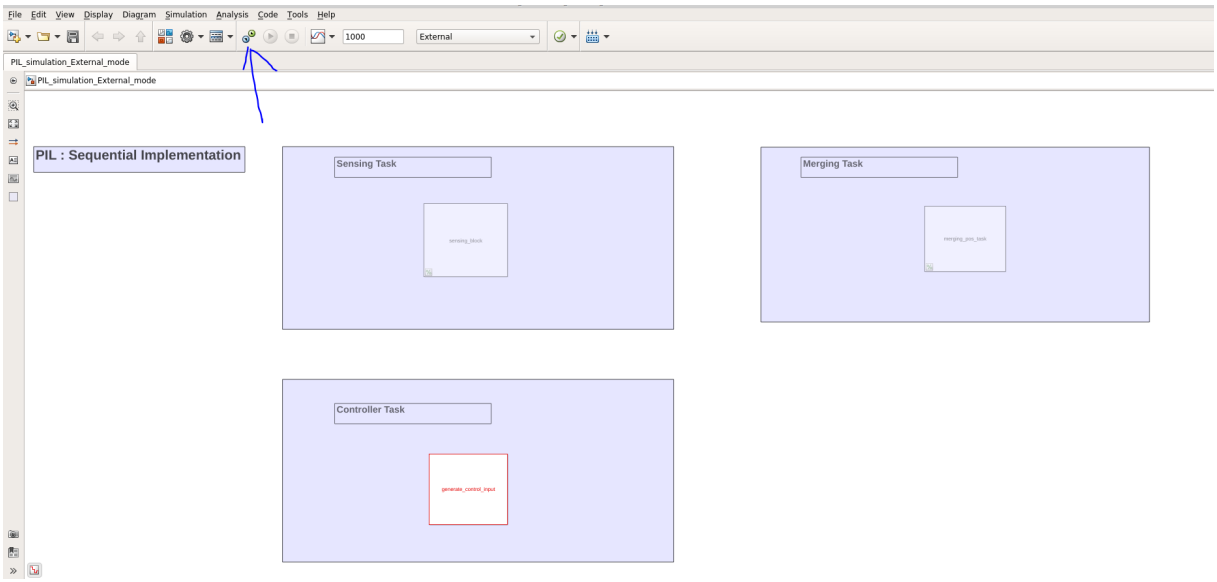
- Terminal 1 :
  - cd tutorial
  - cd monitoring-tools/
  - sudo ./channel_cheap_bridge 0 3 9878
- Terminal 2 :
  - cd tutorial
  - ./readout.sh



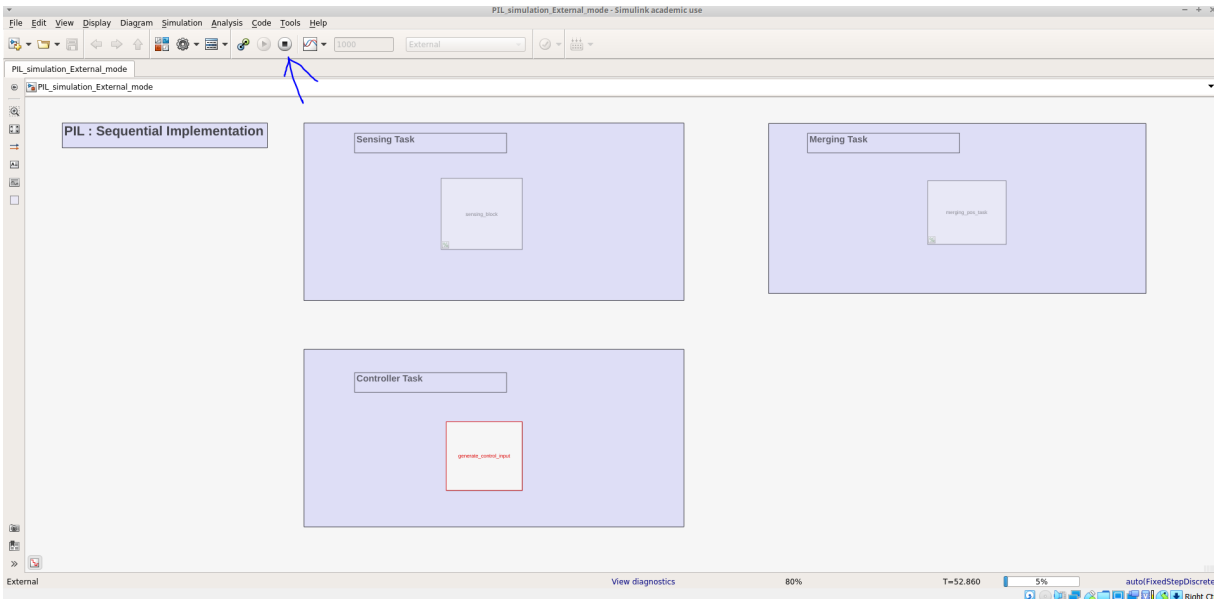
➢ click on the build button, as shown by the arrow in the below screenshot.



➢ if the build finishes successfully, ready status is available in the model. After that, run the task in external mode by clicking the connect to target button. Check the following screenshot for reference.

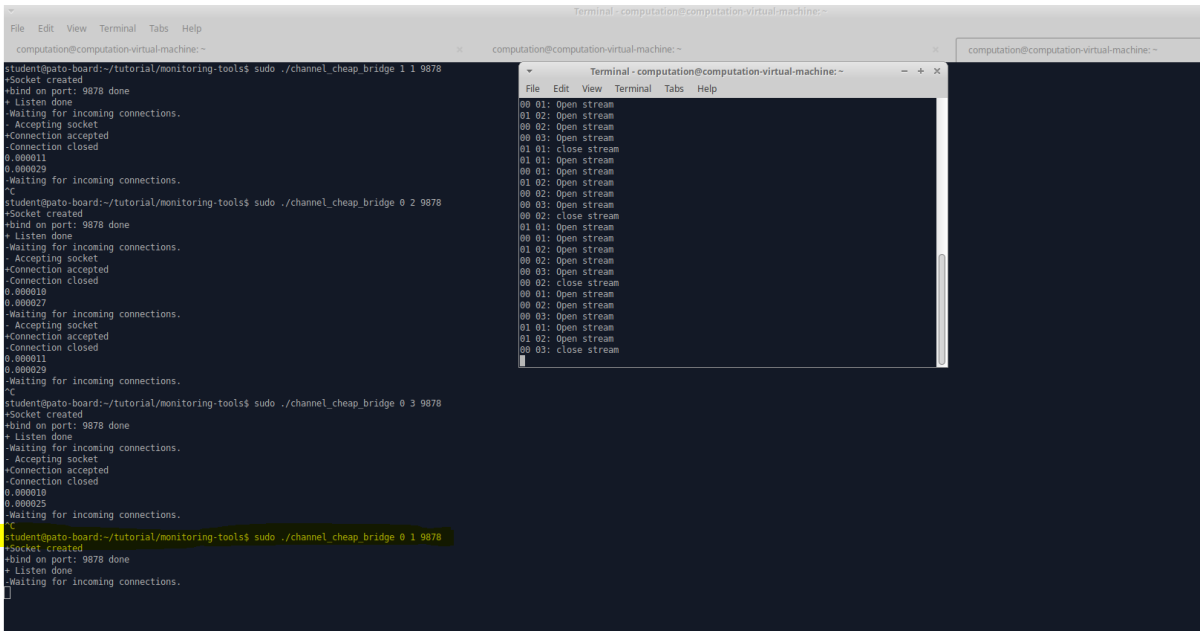


- after the above step, the controller task connects to the target and starts running on the CompSOC; let it starts and runs for some time, then stop the running of the task by clicking on the stop button; check the arrow in the screenshot. No need to run the task for a long time.

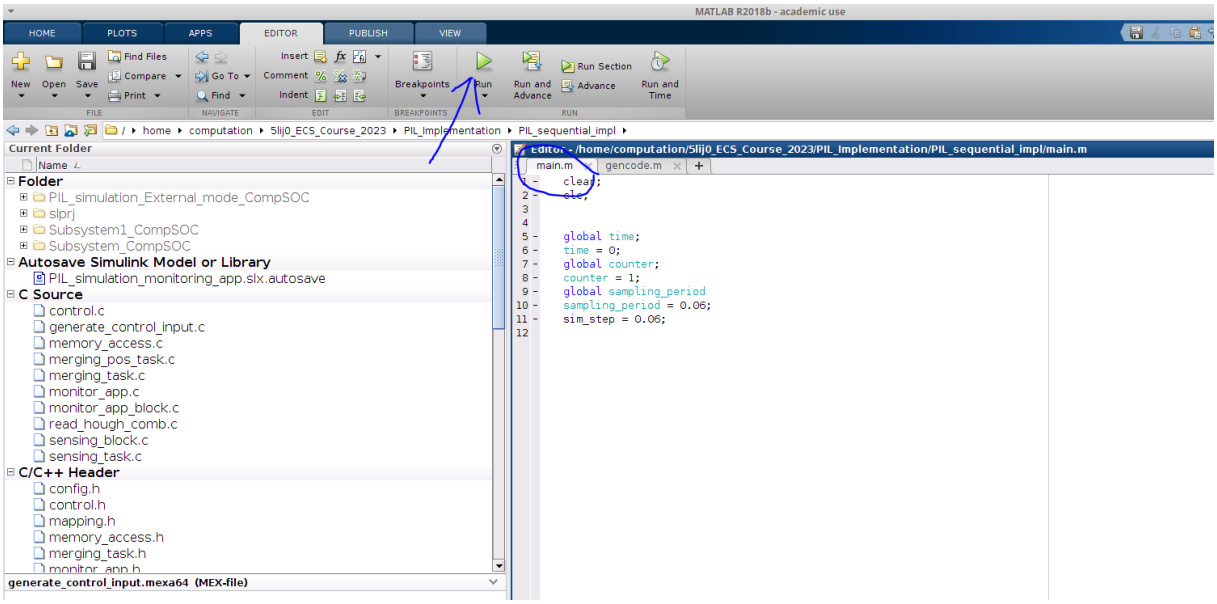


- after completing all the steps, run the following highlighted command in the same terminal window opened for the previous steps. This step is required for running the final PIL simulation, which will be run from **PIL_simulation_monitoring_app.slx** Simulink model.

  - Terminal 1 :
    - cd tutorial
    - cd monitoring-tools/
    - sudo ./channel_cheap_bridge 0 1 9878
  - Terminal 2 :
    - cd tutorial
    - ./readout.sh

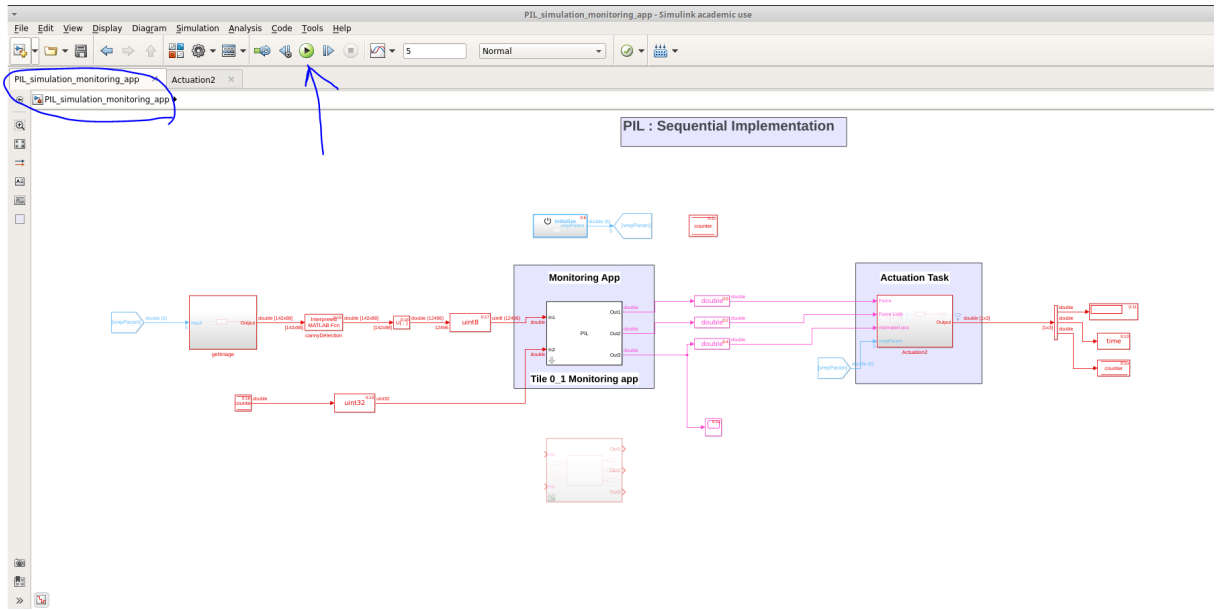


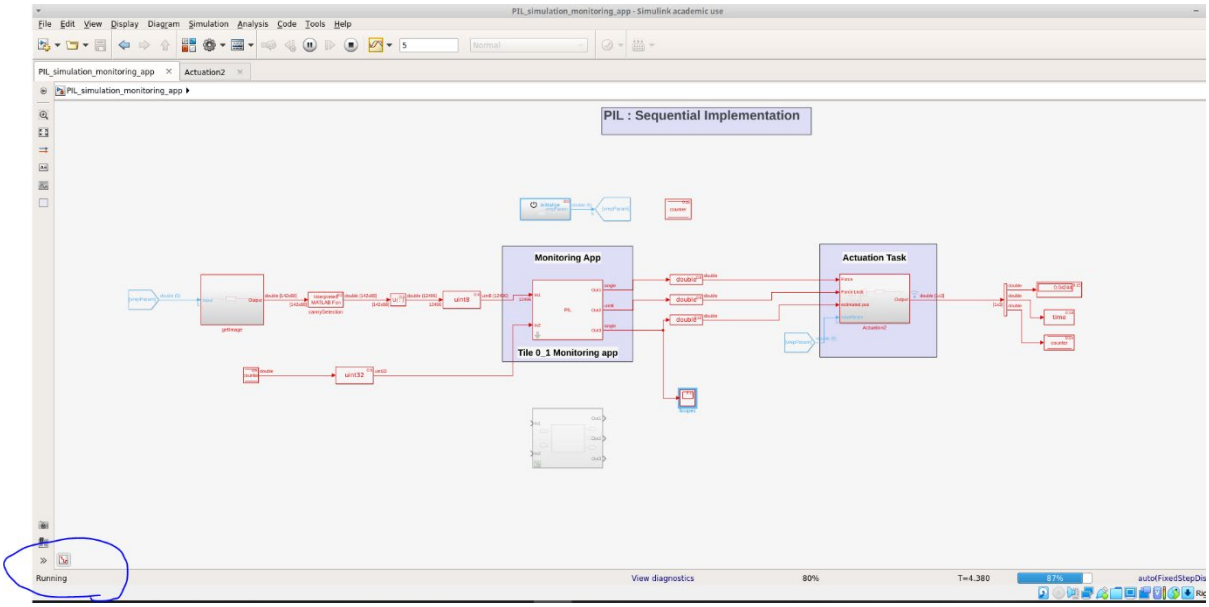
- next step is to run the main.m file and check the CoppeliaSim software; the mass-damper scene should be opened and ensure it is not running automatically; if it is running, stop it.



- open the **PIL_simulation_monitoring_app.slx** Simulink model, and click on the Run button. This will start running the simulation. It will take some time to simulate as it initializes all the required settings for running the simulation. So wait until you see the running status. Check the following screenshots for reference.

- check the current position returned from the monitoring app using the scope linked to the monitoring app PIL block. Also, visualize the output in CoppeliaSim.


PIL_simulation_monitoring_app - Simulink academic use
File Edit View Display Diagram Simulation Analysis Code Tools Help
PIL_simulation_monitoring_app
Actuation2
PIL : Sequential Implementation
Monitoring App
Tile 0_1 Monitoring app
Actuation Task
uint8
uint32
time
counter

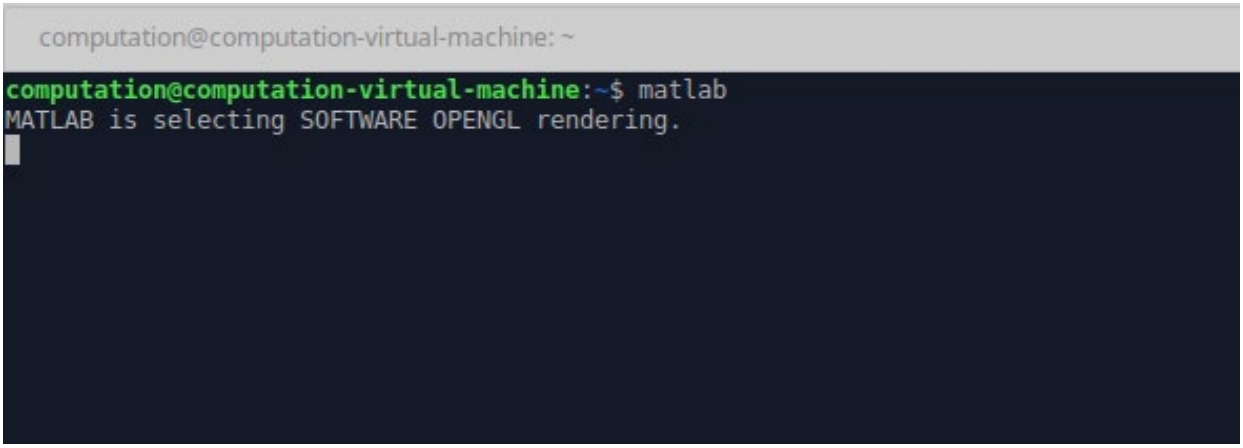

PIL_simulation_monitoring_app - Simulink academic use
File Edit View Display Diagram Simulation Analysis Code Tools Help
PIL_simulation_monitoring_app
Actuation2
PIL : Sequential Implementation
Monitoring App
Tile 0_1 Monitoring app
Actuation Task
uint8
uint32
time
counter
Running
View diagnostics
80%
T=4.380
87%

Section II: PiL implementation for parallel configuration

- open MATLAB: To open MATLAB in the IMACS virtual machine, run the following command in the new terminal.

  Note: If MATLAB is already opened, then no need to reopen it.

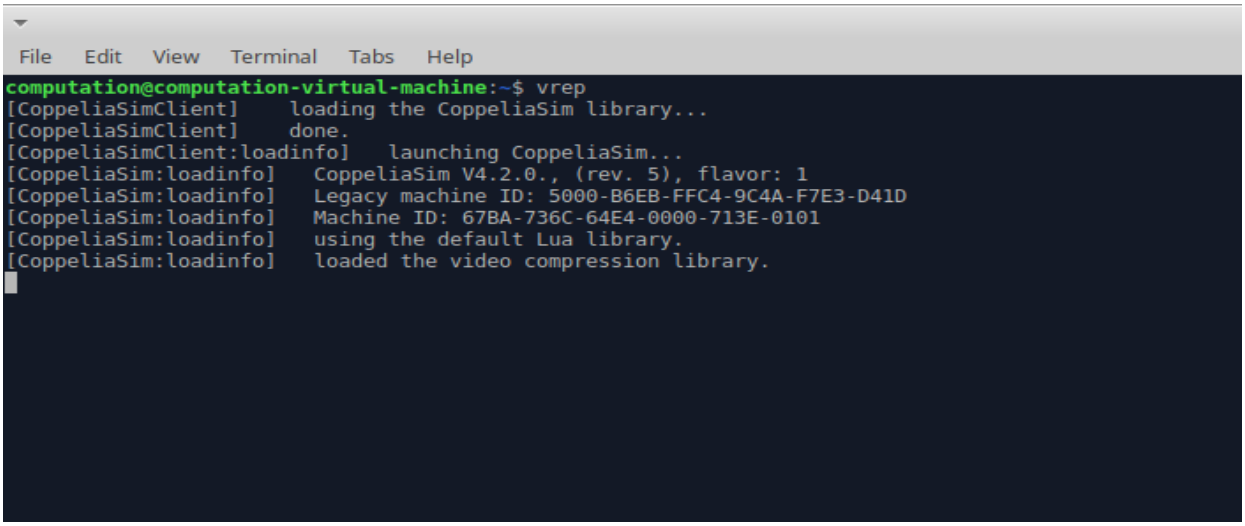
  ```
  $ matlab
  ```



- open CoppeliaSim: to open the CoppeliaSim, run the following command in a new terminal.

  Note: If CoppeliaSim is already opened, then no need to reopen it.

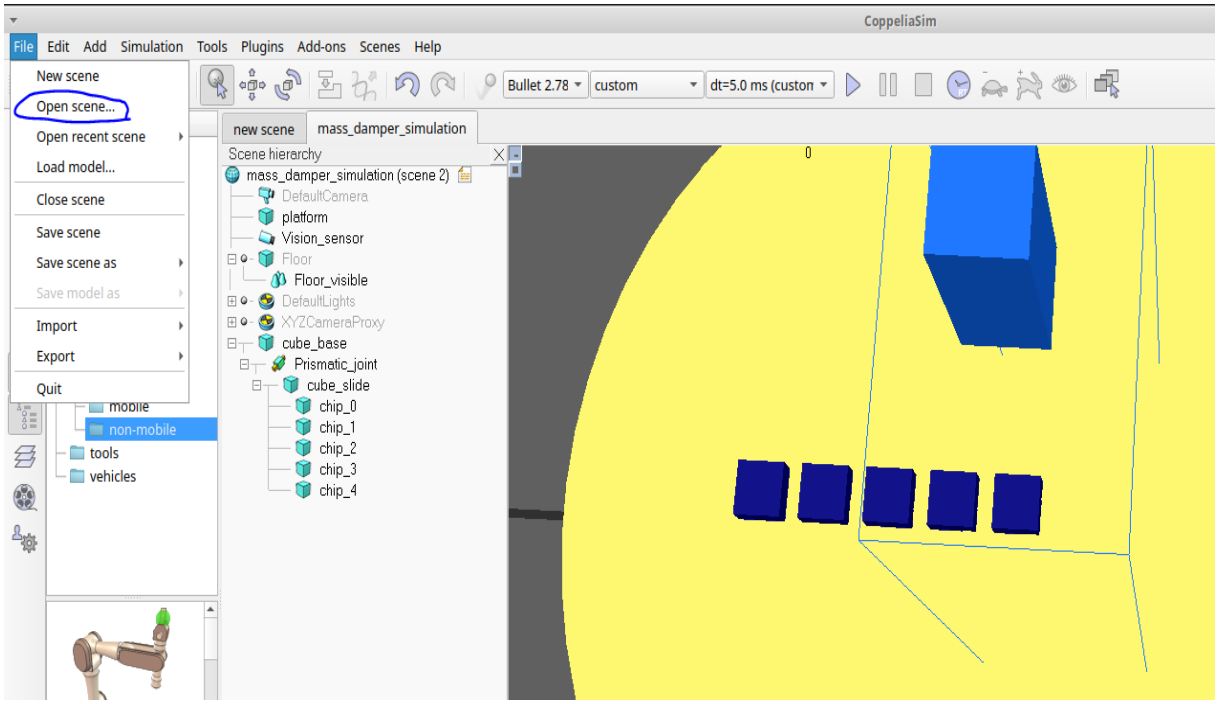
  ```
  $ vrep
  ```

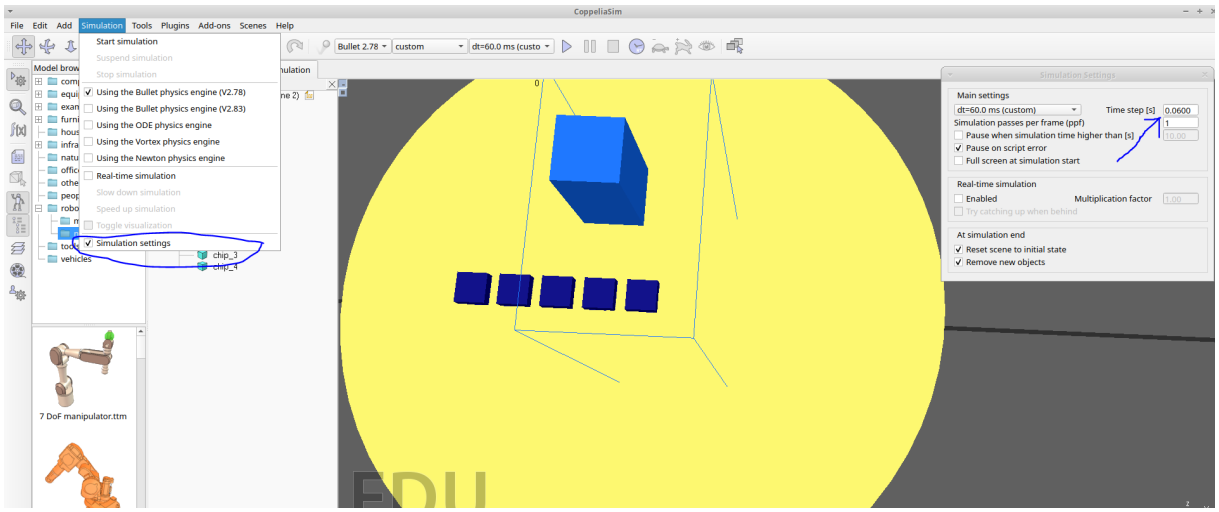


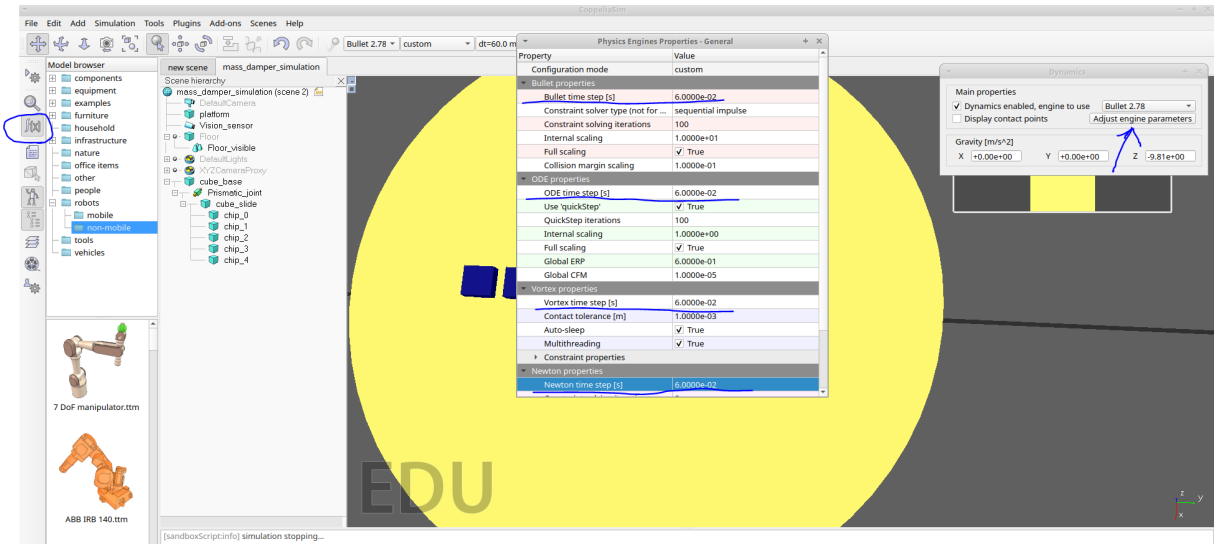
- open CoppeliaSim scene for the mass-damper system.
  - go to File→Open scene→ select the following directory /home/computation/5lij0_ECS_Course_2023/PIL_Implementation/PIL_parallel_impl

open **mass_damper_simulation.ttt**, check the following screenshot for reference.

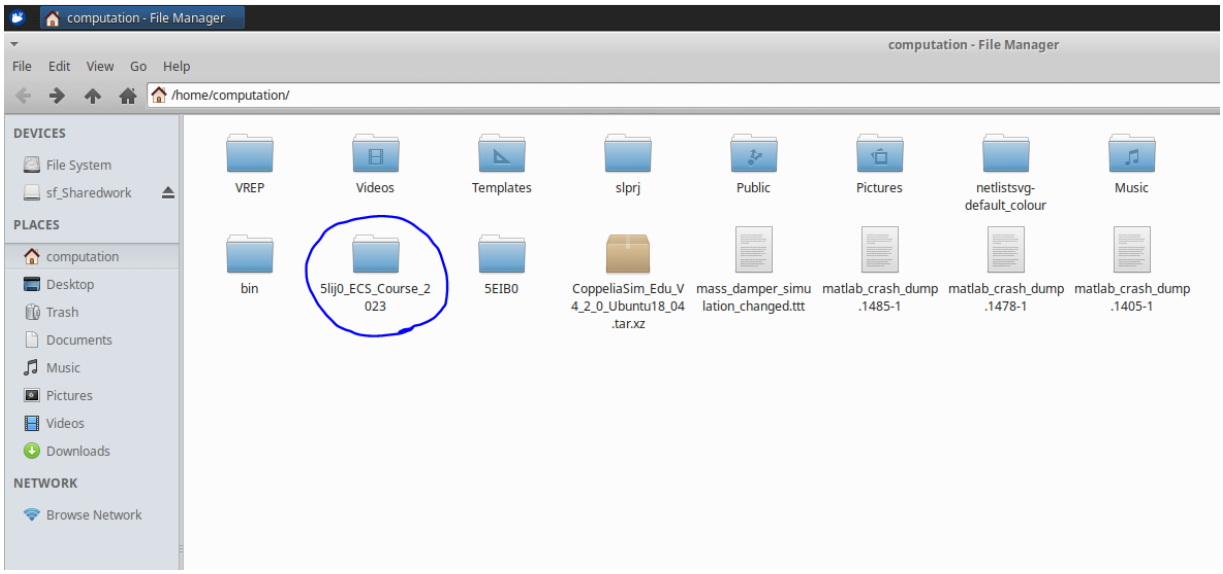
We have implemented the PIL of sequential configuration considering the 60ms sample period. The same sample period should be used in CoppeliaSim. In case of different sample periods, students can make the changes in the CoppeliaSim as explained below. For example, the following screenshot shows how to change the time step in the CoppeliaSim scene.

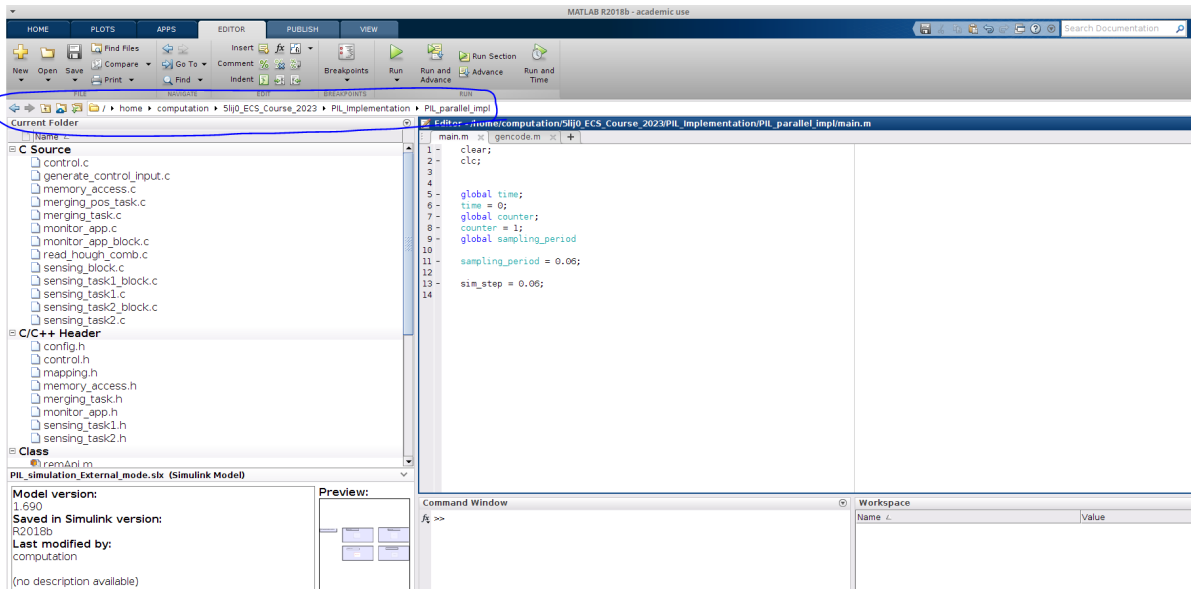






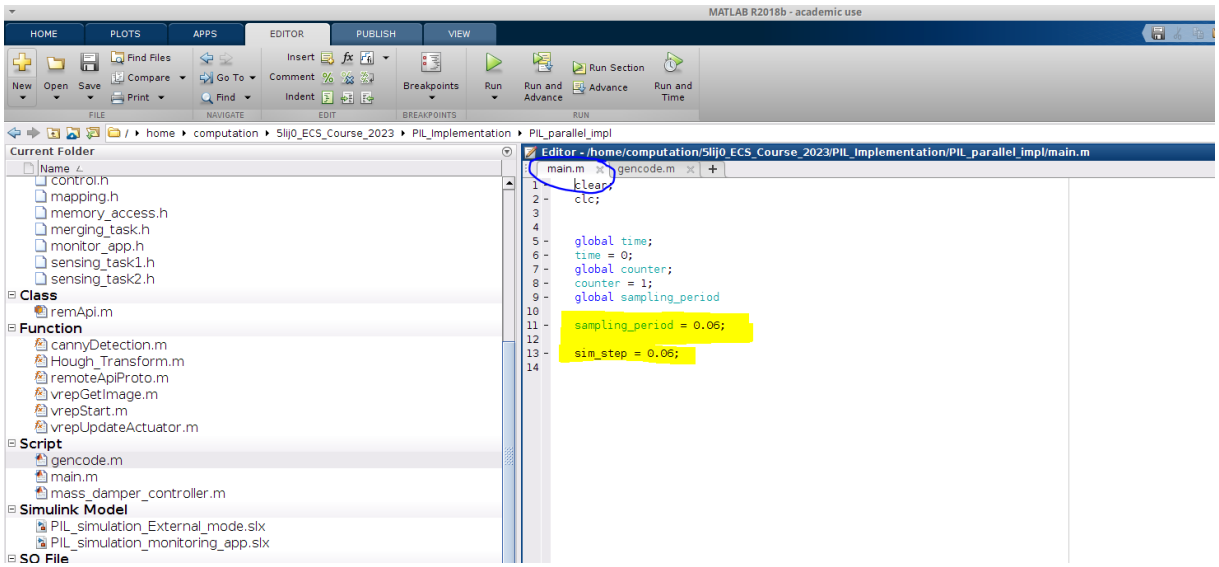
- open the folder for PiL parallel implementation.
  - go to the following folder path and open in MATLAB computation→5lij0_ECS_Course_2023→PIL_Implementation→PIL_parallel_impl



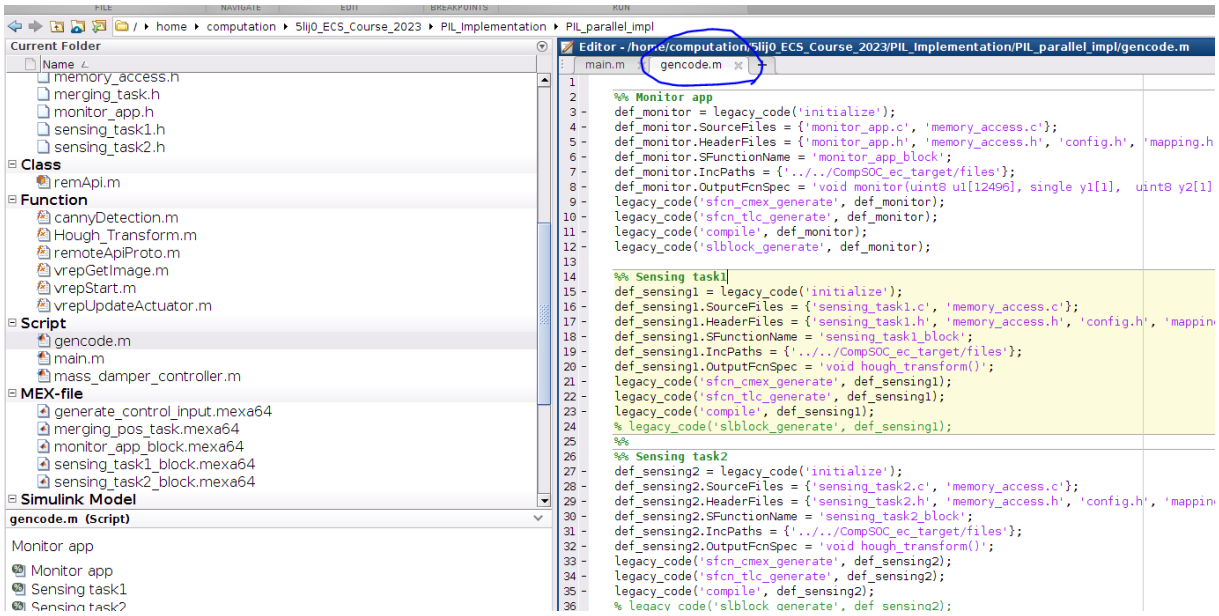


➢ open the main.m file and make changes for the highlighted parameters, as shown in the following screenshot.

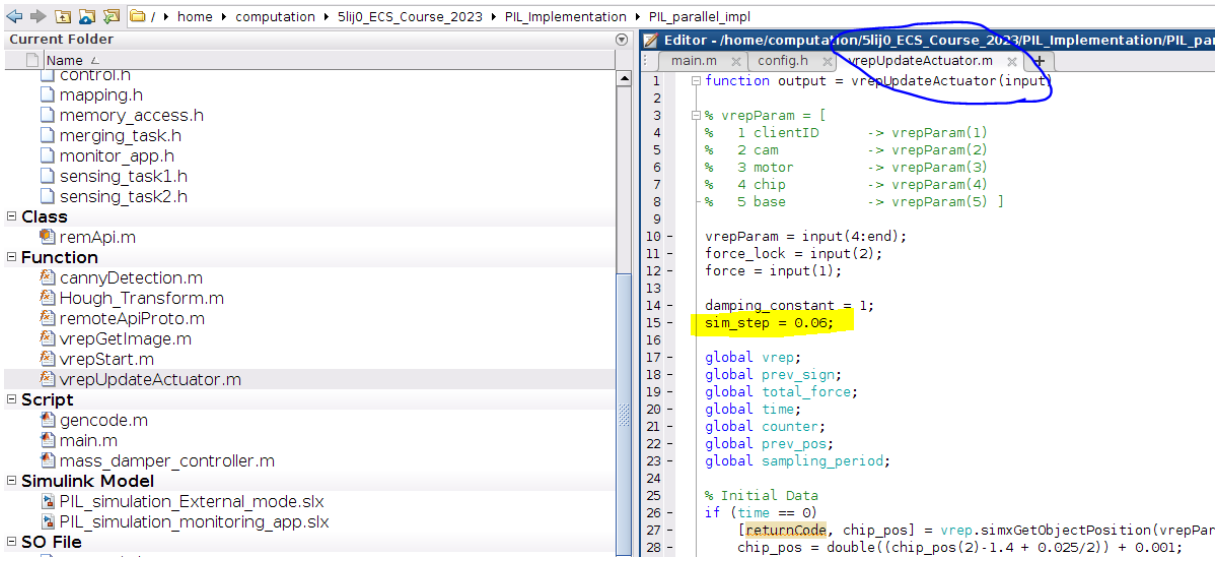
Note: In this screenshot, we have explained for the 60ms sample period, students can use different sample periods according to the implementation design and make the changes for the following parameters. Once the changes are done, run the main.m script. This file includes the initialization, which is required for the PiL simulation.



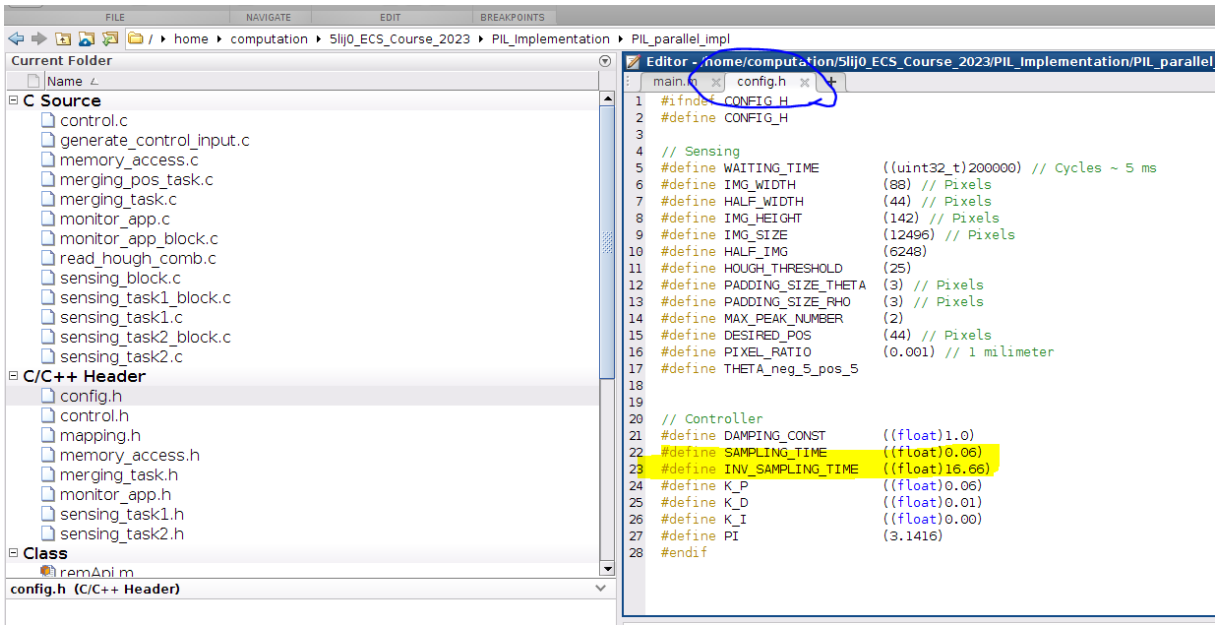
➢ open the gencode.m file and run the script. This file uses legacy code tool functionality, integrating the existing C or C++ functions into MATLAB Simulink. Once this gencode.m runs, it creates the S-function block inside the Simulink model.



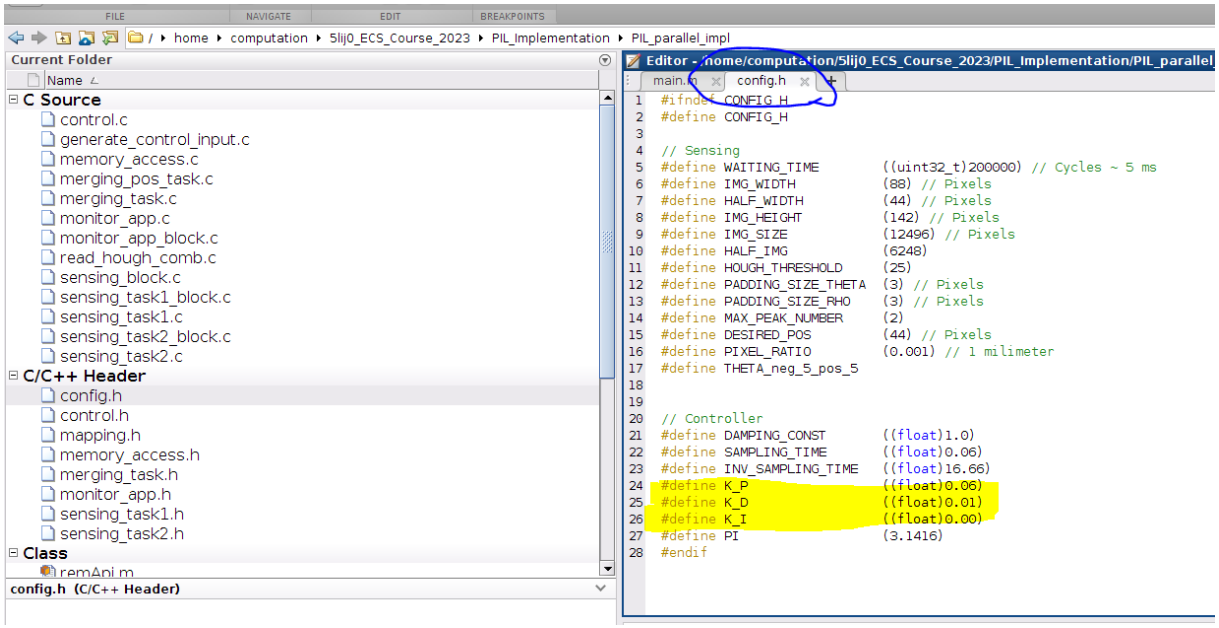
- open the **vrepUpdateActuator.m** file. Make the necessary changes in the file, as shown below in the screenshot. This file is required for the actuation.



- open the **config.h** file. Make the necessary changes in the file, as shown below in the screenshot. This file contains all the required parameters used in the PiL simulation.



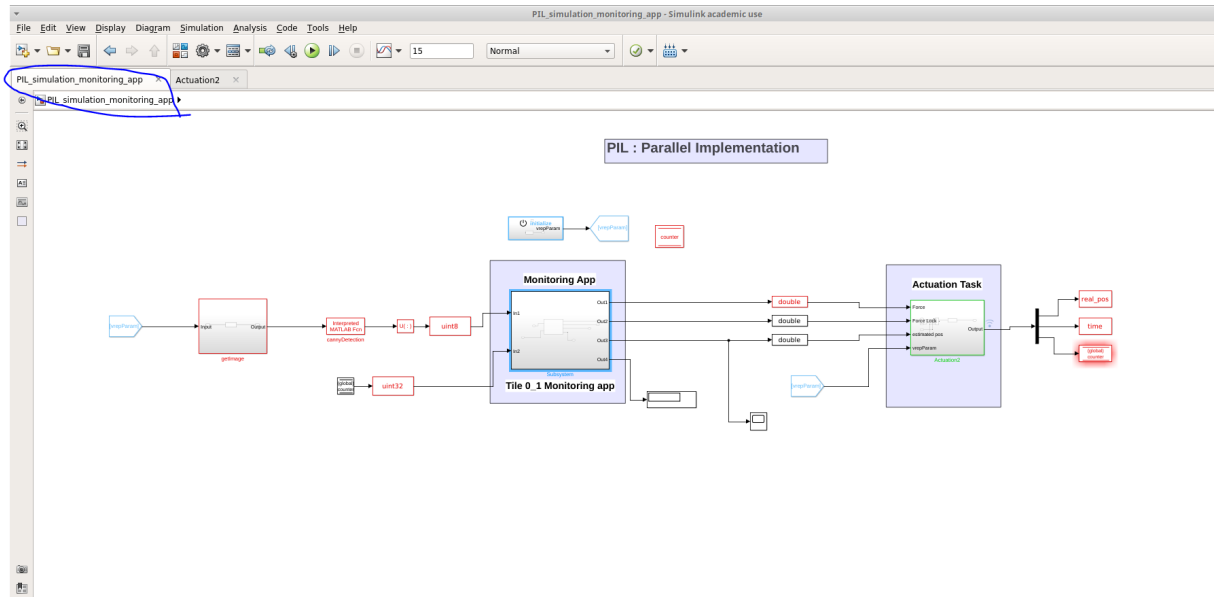
- for tuning the PID controller gains, students can change the gains in the **config.h** file, check the below screenshot.



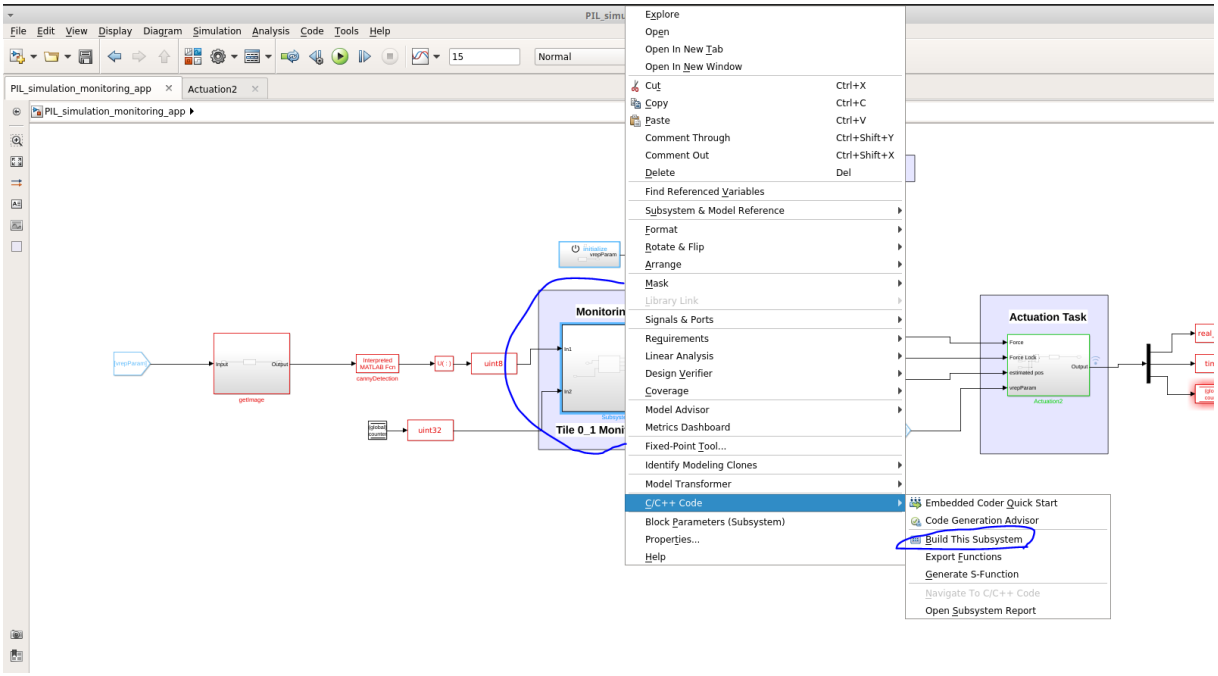
- once the steps mentioned above are done, open the PiL simulation file named **PIL_simulation_monitoring_app.slx**

  Students can find this file in the following directory:

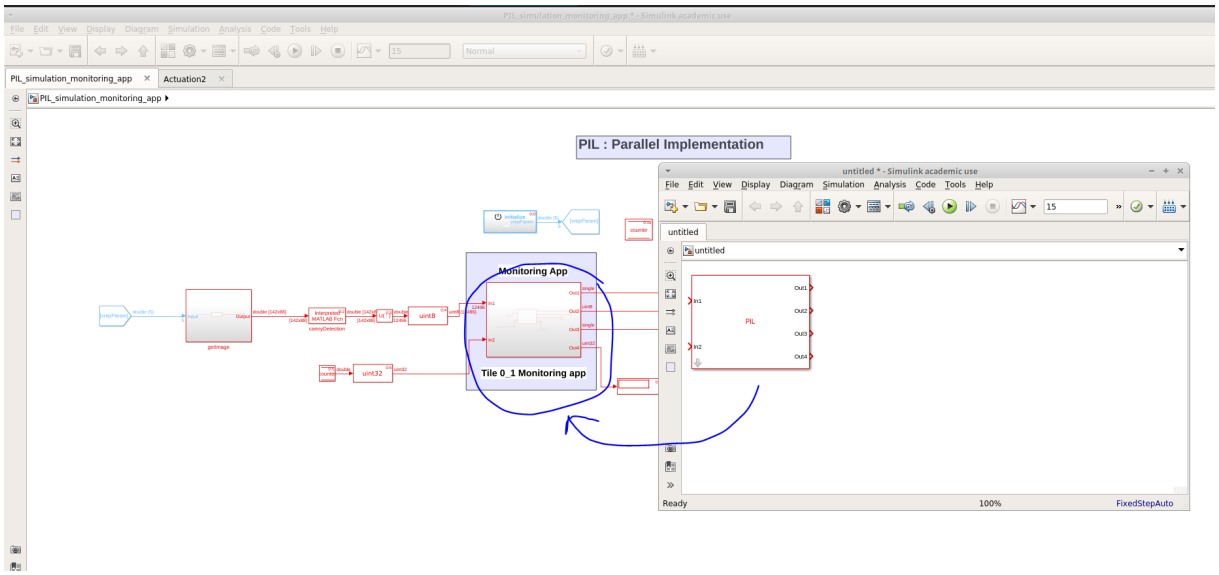
  computation→5lij0_ECS_Course_2023→PIL_Implementation→PIL_parallel_impl



➢ right-click on the block, select the C/C++ code option, and click on build this subsystem. Check the following screenshot for reference.



➢ once, the build process is completed successfully; it will generate the PIL block; check the following screenshot:



- after completing the above steps, check the vep_config.txt file. In this example, we have considered the 60ms sample period. Therefore, in the vep_config.txt, scheduling is given according to the 60ms sample period.
- we have provided the vep_config.txt file in the following folder

  /home/computation/5lij0_ECS_Course_2023/PIL_Implementation/PIL_parallel_impl
- students need to copy the contents from the above vep_config file into the main **vep-config.txt** file, which is inside the following folder path :

  or the student can enter the TDM scheduling calculation directly in the vep-config.txt file, which is stored in the following location.

  /home/computation/CompSOC_ec_target/CompSOC_ec/+CompSOC_ec/vep-config.txt

- the contents in the vep_config file should be as follows:

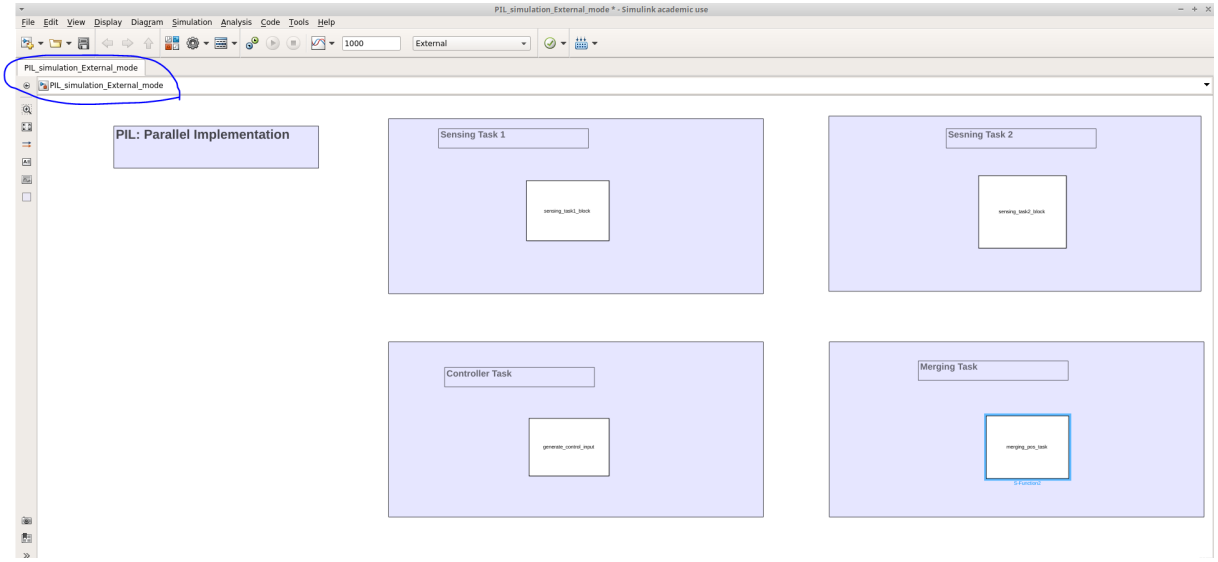
File Edit Search View Document Help

```
##
## you can comment out & modify the "on tile ... " lines
##
## - partitions can only have a memory size of 32K or 64K
## - the size of all partitions on a tile must be at most 96K
##   (i.e. 128K including the system application)
## - default memory is 32K, default stack is 4K
##
## use /opt/riscv/bin/riscv32-unknown-elf-size [-A] *.elf
## to analyse a partition's memory usage
##
## memory and stack allocation
##
on tile 0 partition 1 has 32K memory and 4K stack
on tile 0 partition 2 has 32K memory and 4K stack
on tile 0 partition 3 has 32K memory and 4K stack
on tile 1 partition 1 has 64K memory and 4K stack
on tile 1 partition 2 has 32K memory and 4K stack
#on tile 1 partition 3 has 32K memory and 4K stack
on tile 2 partition 1 has 64K memory and 4K stack
on tile 2 partition 2 has 32K memory and 4K stack
#on tile 2 partition 3 has 32K memory and 4K stack
##
## scheduling
##
## - max 3 slots per tile
## - it is allowed to not schedule anything on a tile
## - a partition can have more than one slot
## - the system partition always get a slot of 5000 cycles
##   at the end: you don't have to add this
##
#on tile 0 next slot is for partition 1 with 200000 cycles
on tile 0 next slot is for partition 1 with 800000 cycles
on tile 0 next slot is for partition 2 with 10000 cycles
on tile 0 next slot is for partition 3 with 1577000 cycles
#on tile 0 next slot is for partition 1 with 71000 cycles
on tile 1 next slot is for partition 1 with 800000 cycles
on tile 1 next slot is for partition 2 with 1589000 cycles
on tile 2 next slot is for partition 1 with 800000 cycles
on tile 2 next slot is for partition 2 with 1589000 cycles
# on tile 2 next slot is for partition 2 with 10000 cycles
# on tile 2 next slot is for partition 3 with 100000 cycles
```

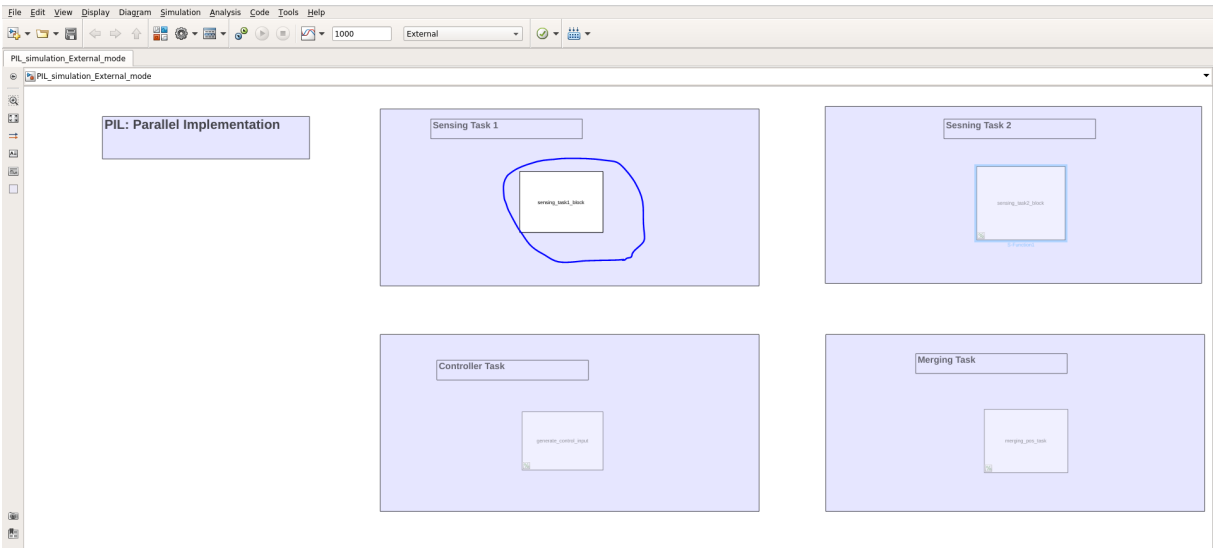
- open the PIL_simulation_External_mode.slx. The folder path to this Simulink model is as follows:

  /home/computation/5lij0_ECS_Course_2023/PIL_Implementation/PIL_parallel_impl

- PIL_simulation_External_mode.slx file has four different tasks, i.e., sensing task 1, sensing task 2, merging, and controller, which will run in the external mode on CompSOC based on the TDM scheduling. To run each task independently in the external mode, follow the below steps and check the screenshot for reference:

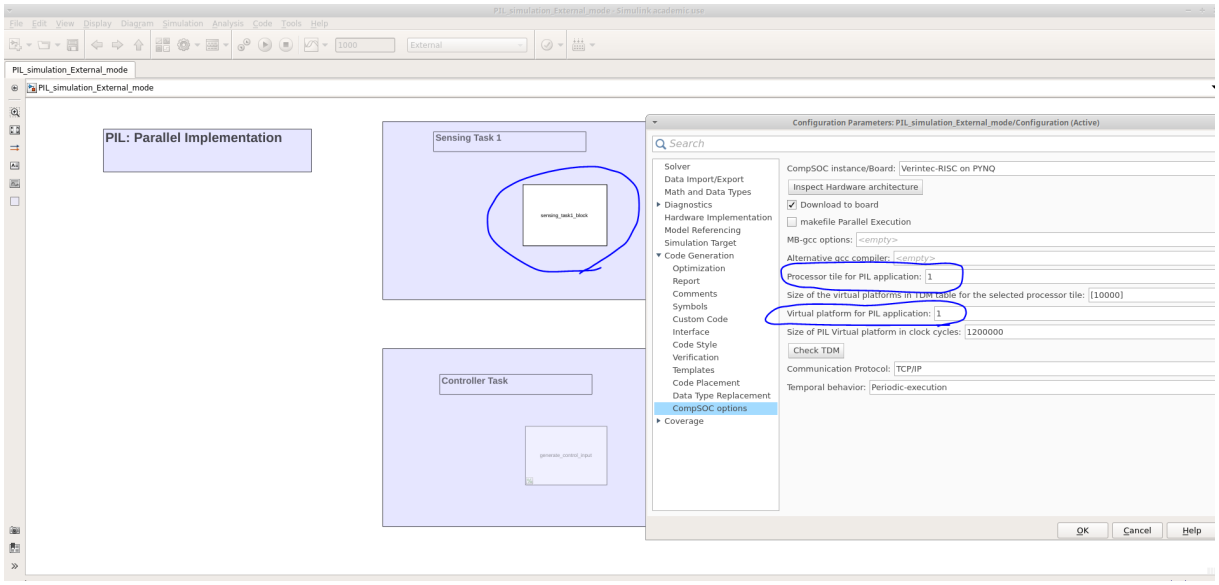


- to run sensing task 1 in the external mode, comment the other Simulink block and only uncomment the block which needs to be run. In the below screenshot, we consider the sensing task 1 block first, and other blocks kept commenting.



- open the configuration setting and check the required setting for the particular task. In this example, we run a sensing task 1 on processor 1 and partition 1. Therefore, consider the following changes :

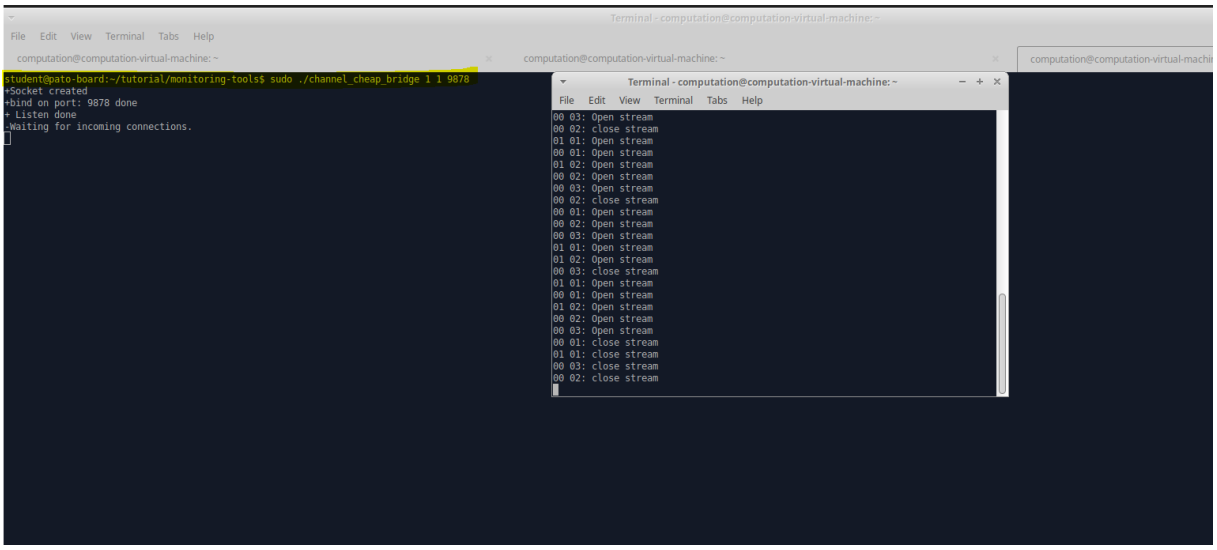
- Processor tile for PIL application: 1
- Virtual platform for PIL application: 1



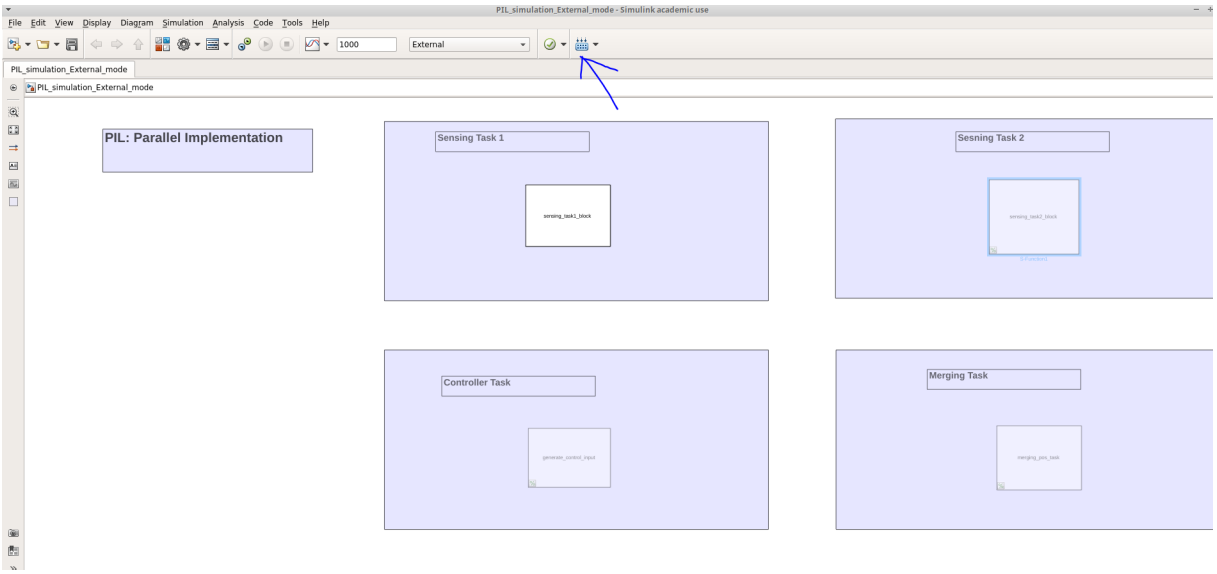
➢ open the two new terminal windows, connect the PYNQ board, and use the following command:

- ensure that vep-config.txt is updated in the main CompSOC folder as per the required TDM scheduling. This step is essential before running the task in external mode on CompSOC.
- vep-config file path: /home/computation/CompSOC_ec_target/CompSOC_ec/+CompSOC_ec/vep-config.txt
- commands for running the sensing task on processor 1 and partition 1

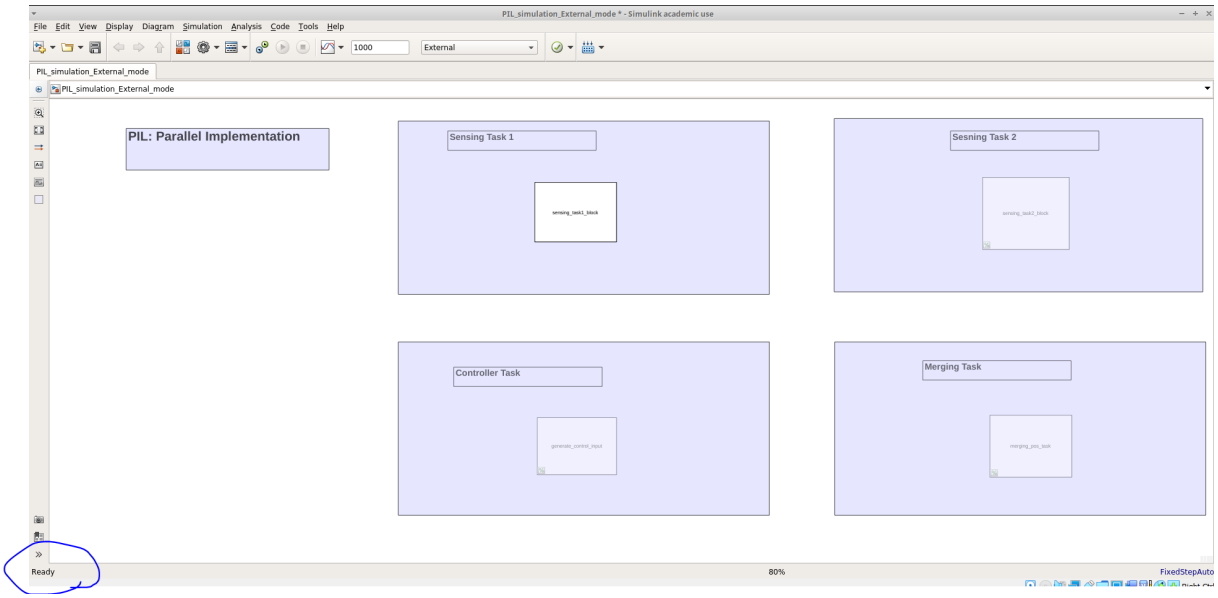
  - Terminal 1 :
    - cd tutorial
    - cd monitoring-tools/
    - sudo ./channel_cheap_bridge 1 1 9878
  - Terminal 2 :
    - cd tutorial
    - ./readout.sh



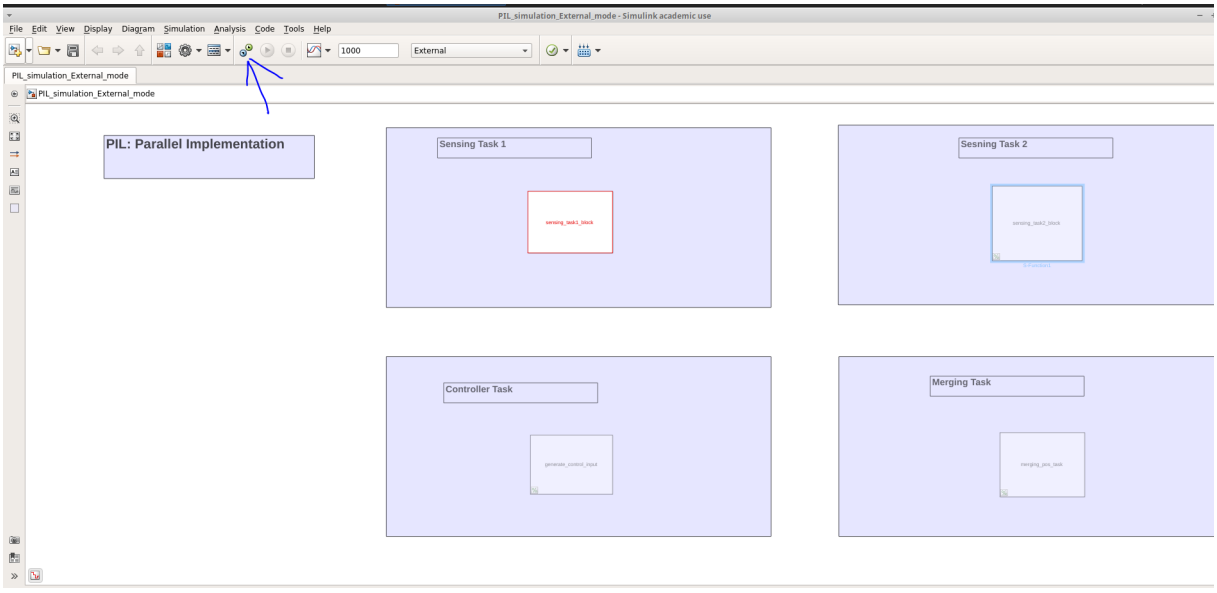
- click on the build button, as shown by the arrow in the below screenshot.



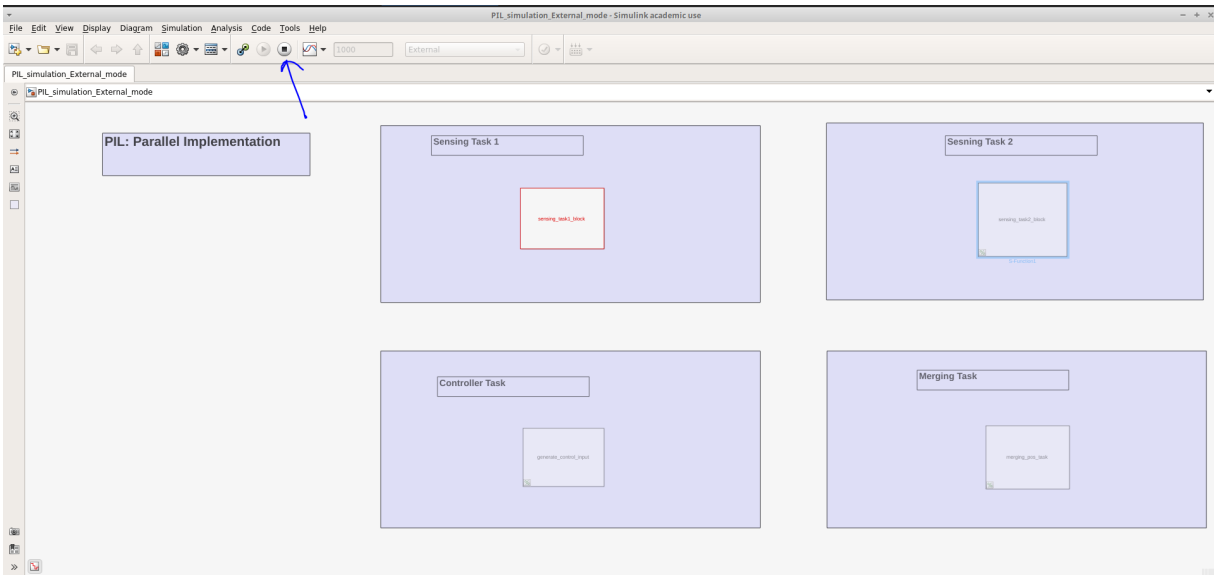
- after the build finishes, the Simulink model shows ready status; check the circle in the following screenshot.



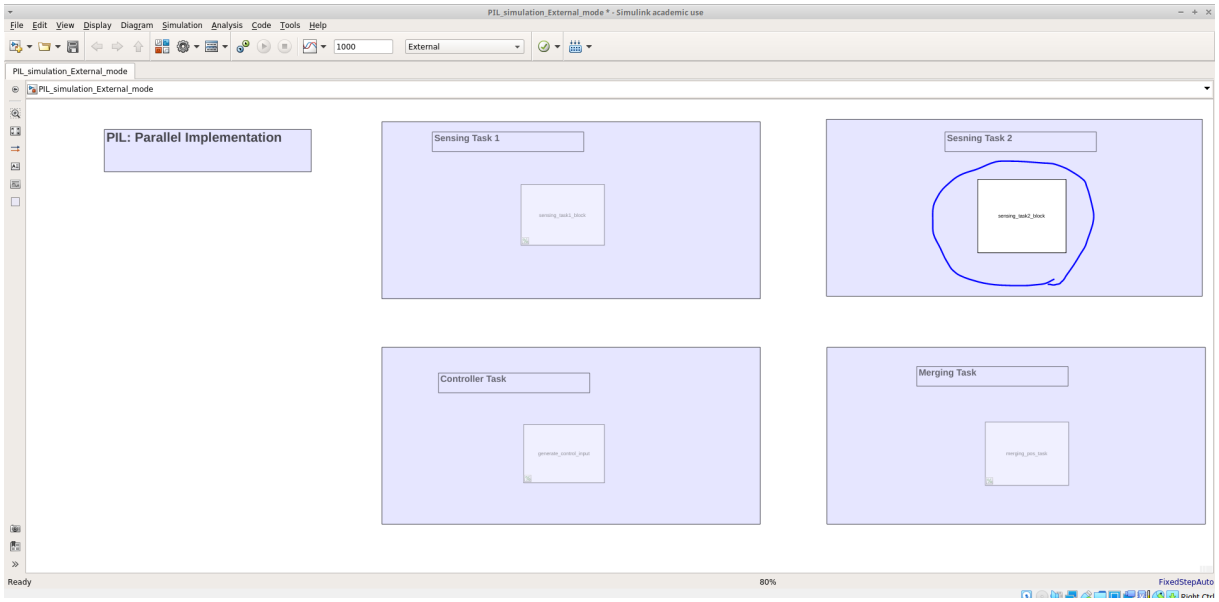
- if the build finishes successfully, ready status is available in the model. After that, run the task in external mode by clicking the connect to target button. Check the following screenshot for reference.



- after the above step, the sensing task connects to the target and starts running on the CompSOC; let it starts and runs for some time; check the highlighted circle in the below screenshot, then stop the running of the task by clicking on the stop button, check the arrow in the screenshot. No need to run the task for a long time.

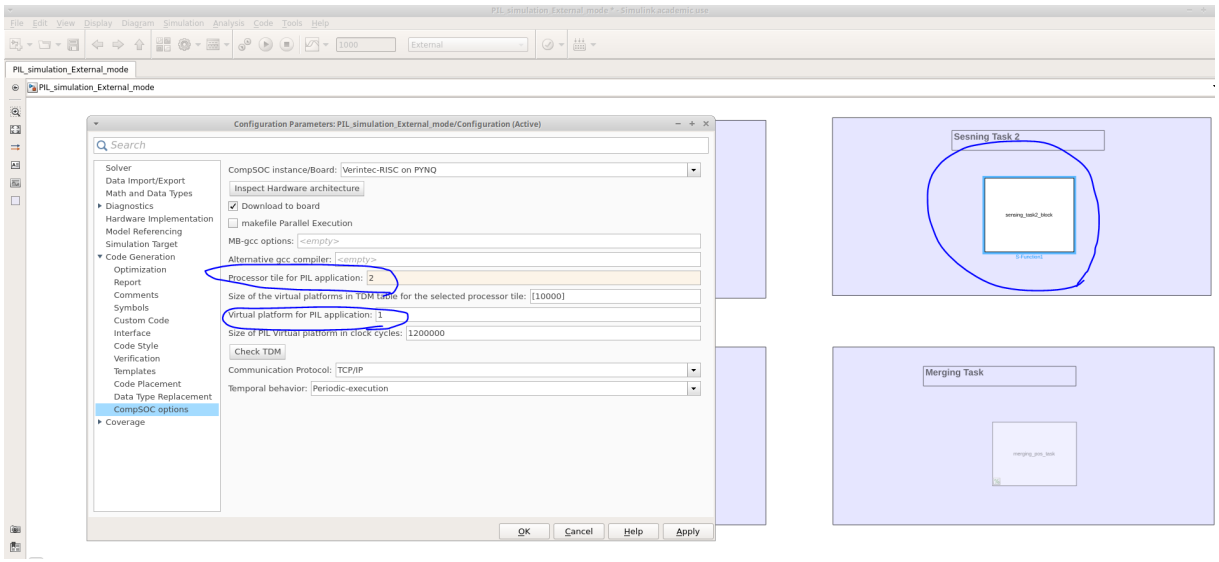


➢ move to the next task and comment all blocks except the one you want to run in external mode. Check the following screenshot; we have considered sensing task 2 after finishing sensing task 1. Only sensing task 2 is uncommented. Other blocks kept commenting.



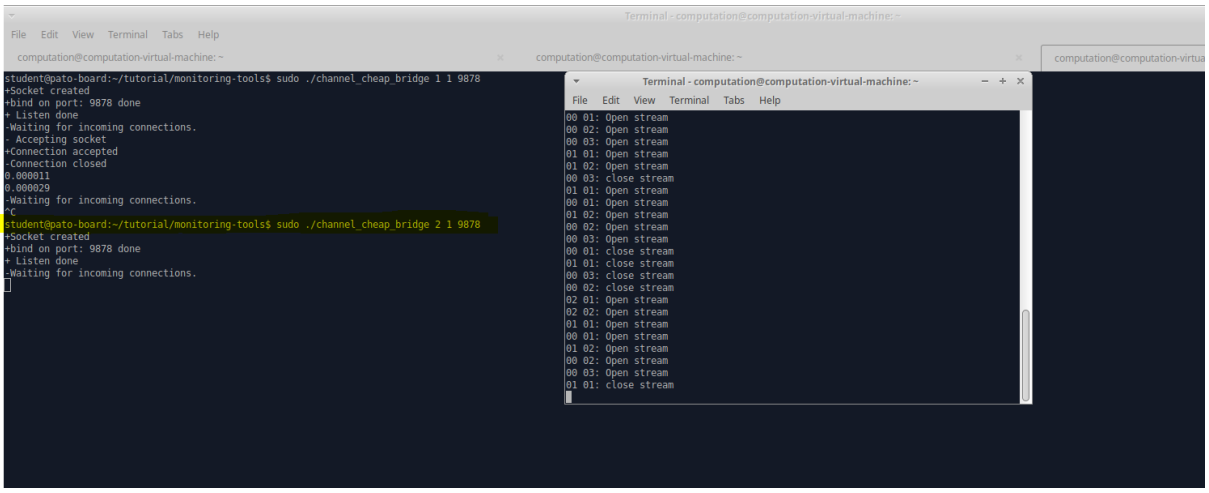
➢ open the configuration setting and check the required setting for the particular task. In this example, we run sensing task 2 on processor 2 and partition 1. Therefore, consider the following changes :

- Processor tile for PIL application: 2
- Virtual platform for PIL application: 1

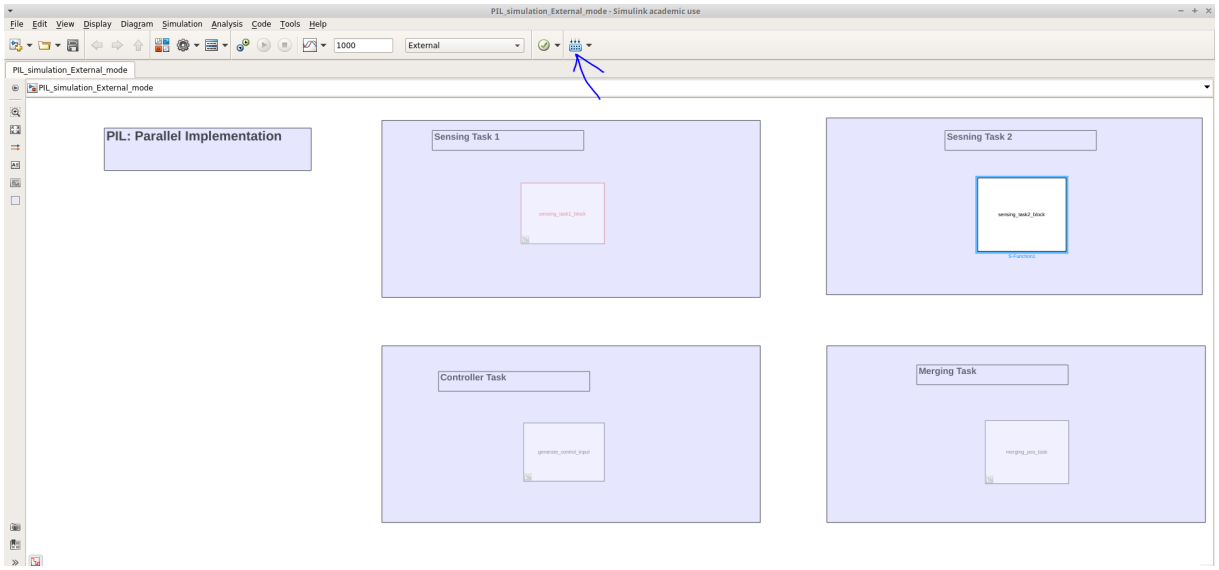


- open the two new terminal windows, connect the PYNQ board, and use the following command:
  - ensure that vep-config.txt is updated in the main CompSOC folder as per the required TDM scheduling. This step is essential before running the task in external mode on CompSOC.
  - vep-config file path: /home/computation/CompSOC_ec_target/CompSOC_ec/+CompSOC_ec/vep-config.txt
  - commands for running the sensing task on processor 2 and partition 1

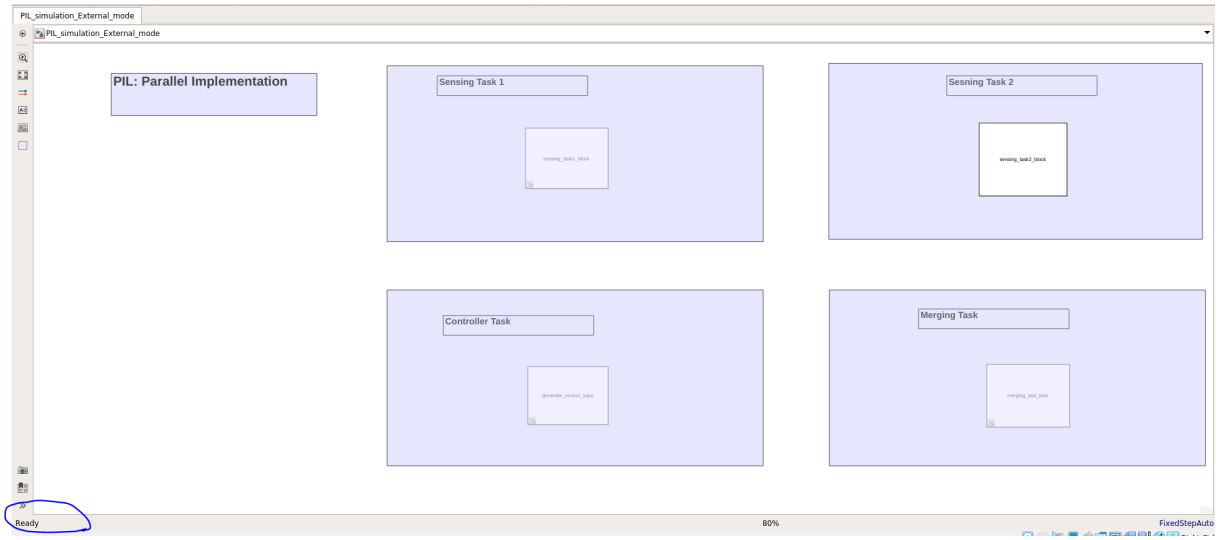
    - Terminal 1 :
      - cd tutorial
      - cd monitoring-tools/
      - sudo ./channel_cheap_bridge 2 1 9878
    - Terminal 2 :
      - cd tutorial
      - ./readout.sh



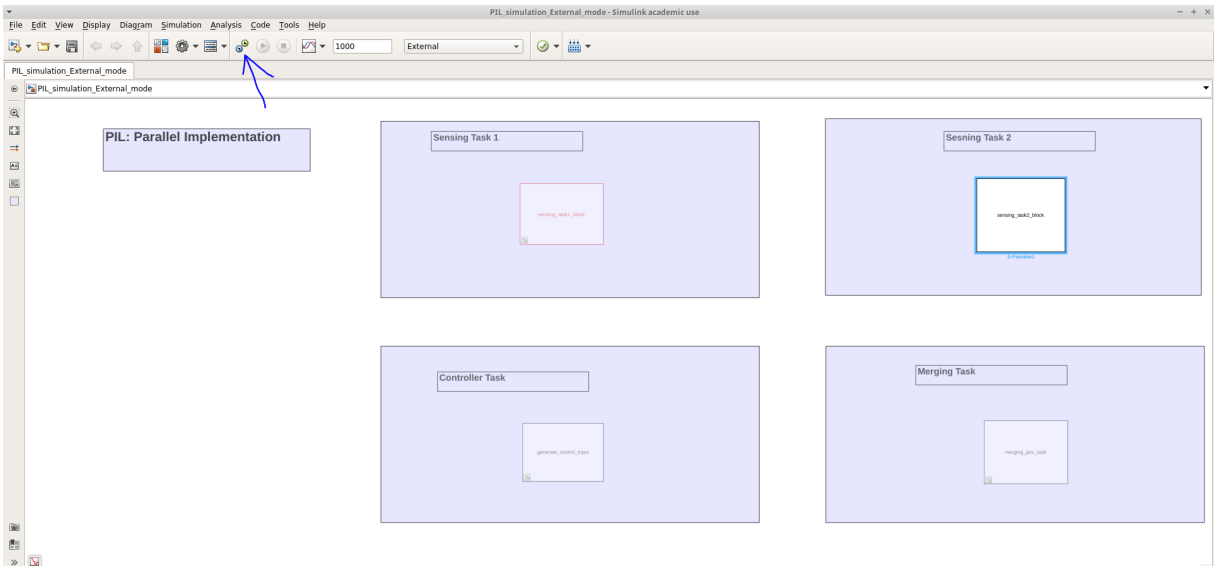
- click on the build button, as shown by the arrow in the below screenshot.



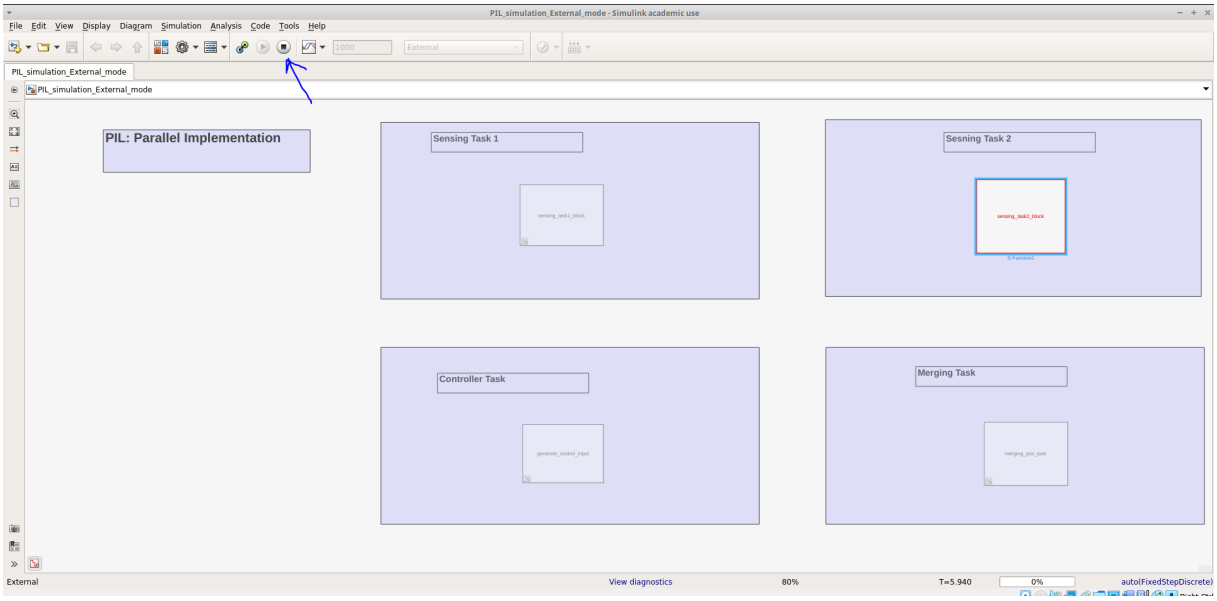
- after the build finishes, the Simulink model shows ready status; check the circle in the following screenshot.



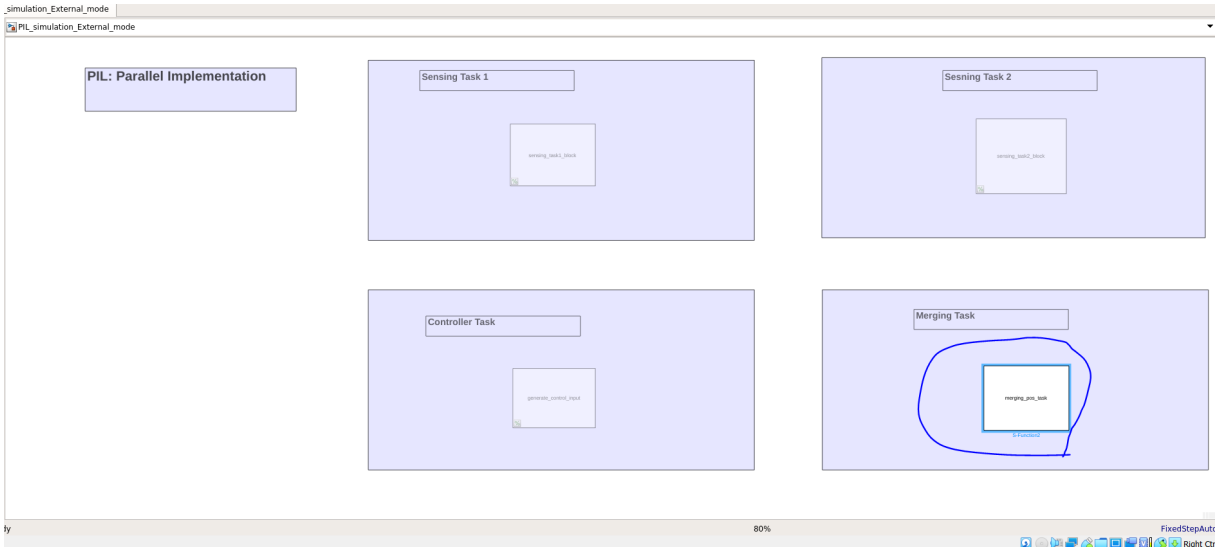
- if the build finishes successfully, ready status is available in the model. After that, run the task in external mode by clicking the connect to target button. Check the following screenshot for reference.



- after the above step, the sensing task connects to the target and starts running on the CompSOC; let it starts and runs for some time; check the highlighted circle in the below screenshot, then stop the running of the task by clicking on the stop button, check the arrow in the screenshot. No need to run the task for a long time.

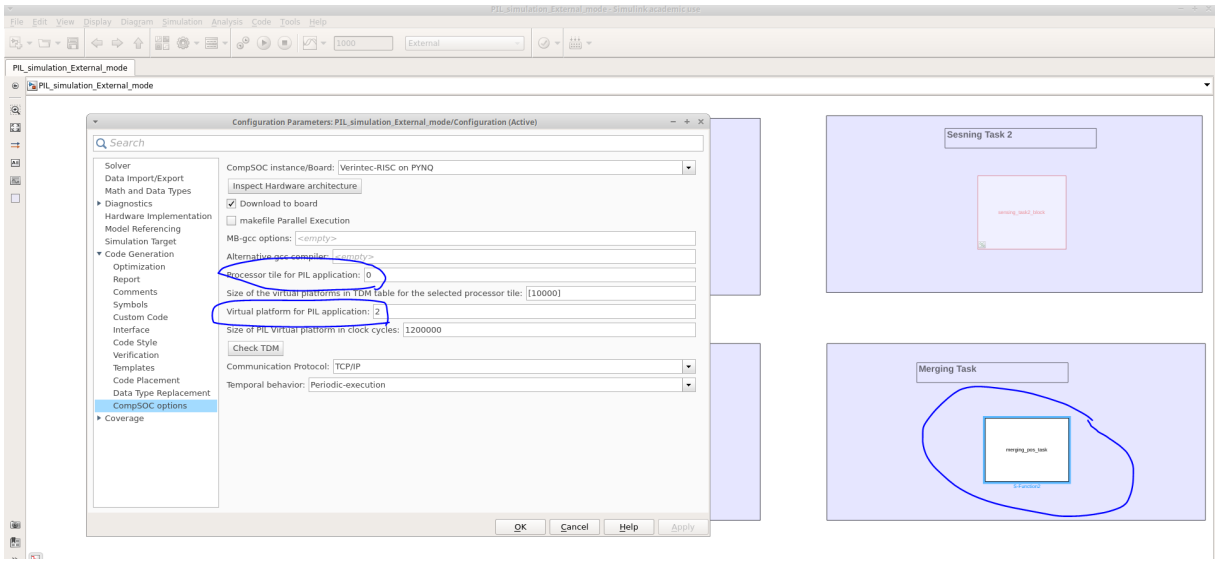


- move to the next task and comment all blocks except the one you want to run in external mode. Check the following screenshot; we have considered merging task after finishing the sensing task. Only the merging task is uncommented. Other blocks kept commenting.



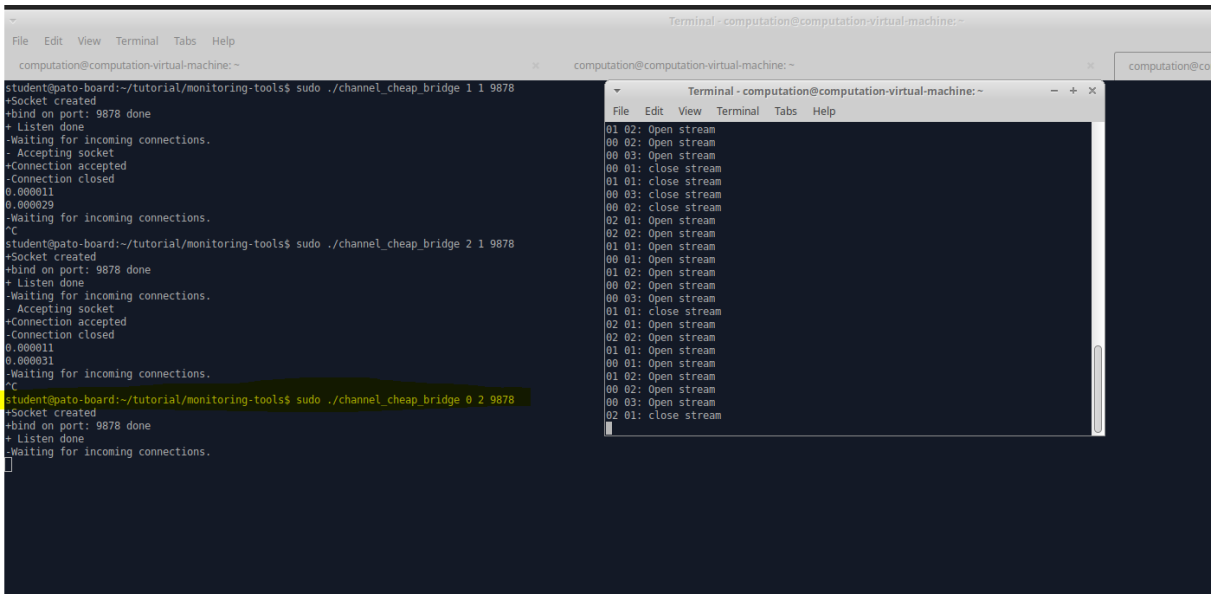
- check the configuration setting. We are running the merging task on processor 0 and partition 2. So, the following setting should be considered,

  - Processor tile for PIL application : 0
  - Virtual platform for PIL application: 2

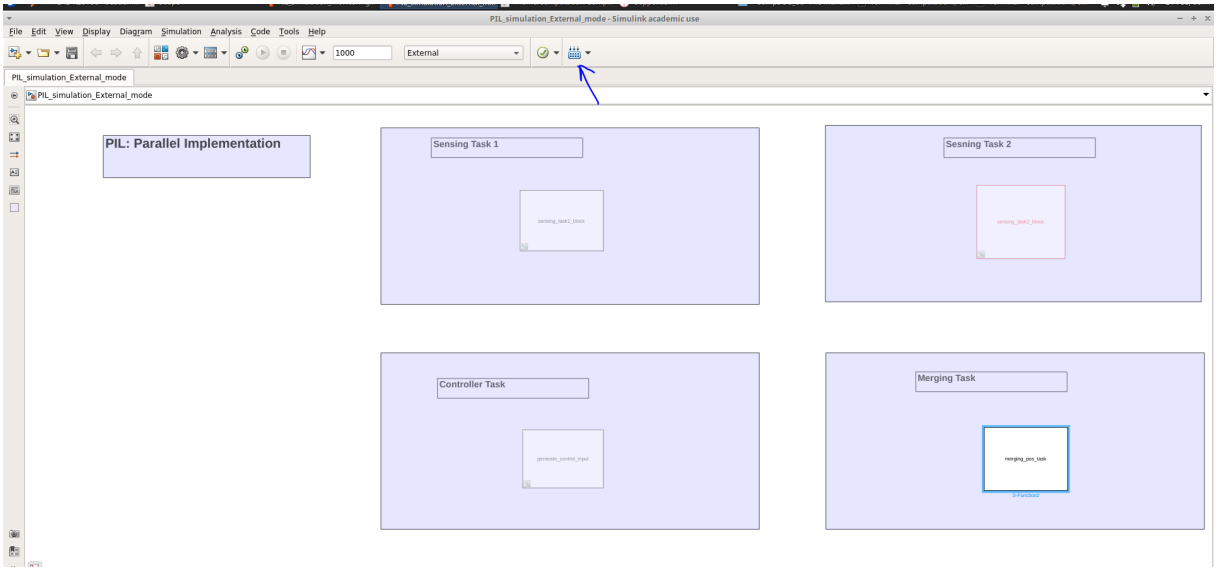


- in the same two terminal windows opened for the previous task, you can use the following highlighted commands from the screenshot.

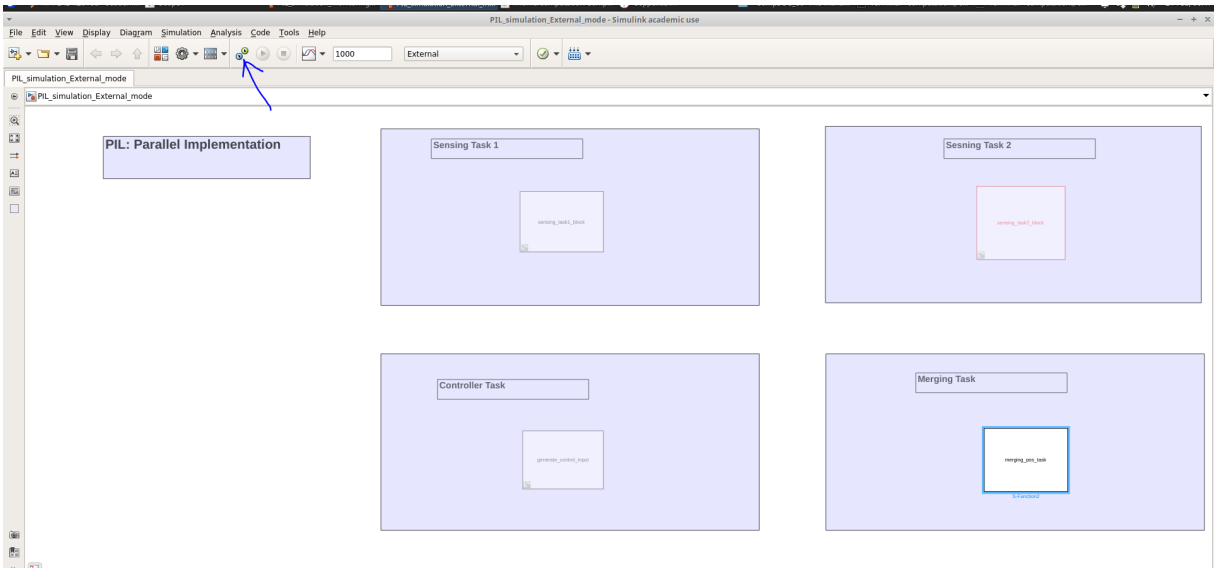
  - Terminal 1 :
    - cd tutorial
    - cd monitoring-tools/
    - sudo ./channel_cheap_bridge 0 2 9878
  - Terminal 2 :
    - cd tutorial
    - ./readout.sh



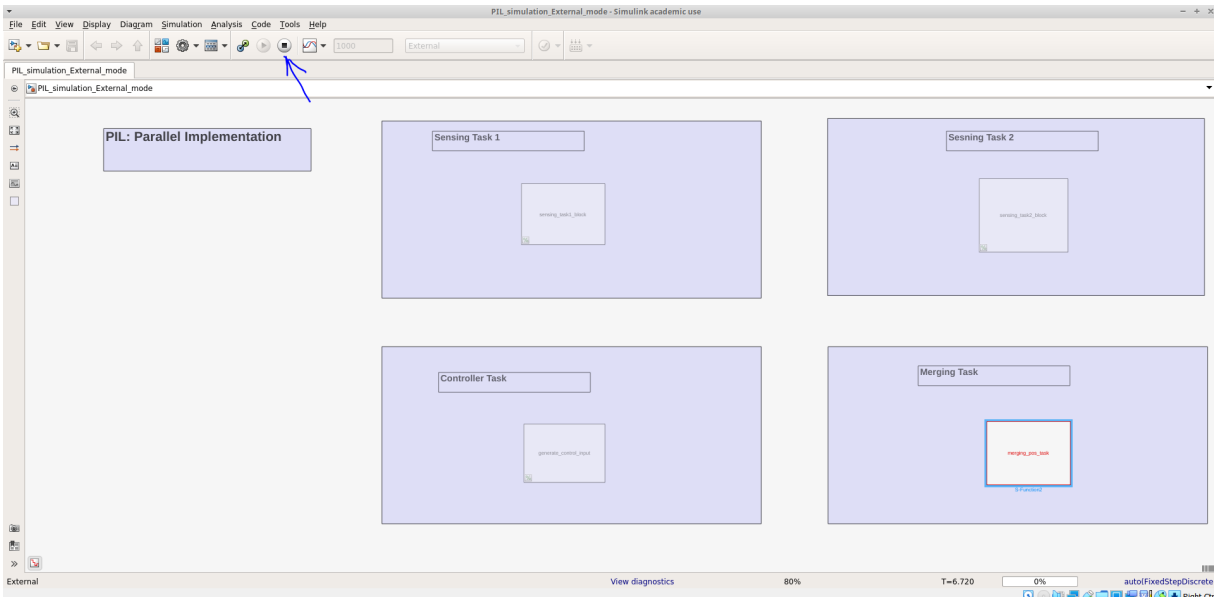
- click on the build button, as shown by the arrow in the below screenshot.



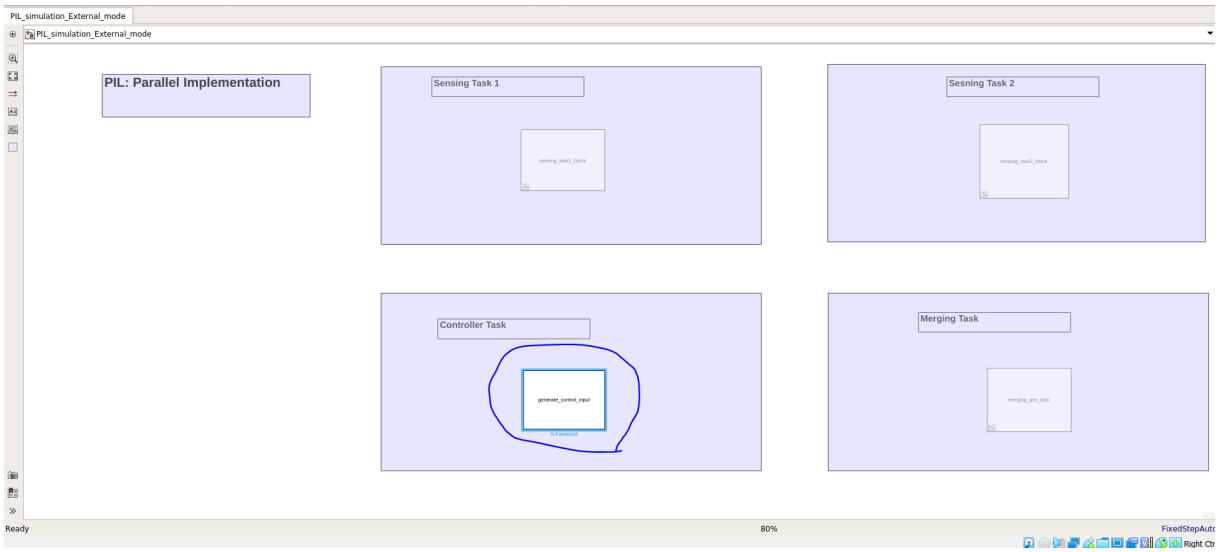
- if the build finishes successfully, ready status is available in the model. After that, run the task in external mode by clicking on the connect to target button. Check the following screenshot for reference.



- after the above step, the merging task connects to the target and starts running on the CompSOC; let it starts and runs for some time, then stop the running of the task by clicking on the stop button. Check the arrow in the screenshot. No need to run the task for a long time.

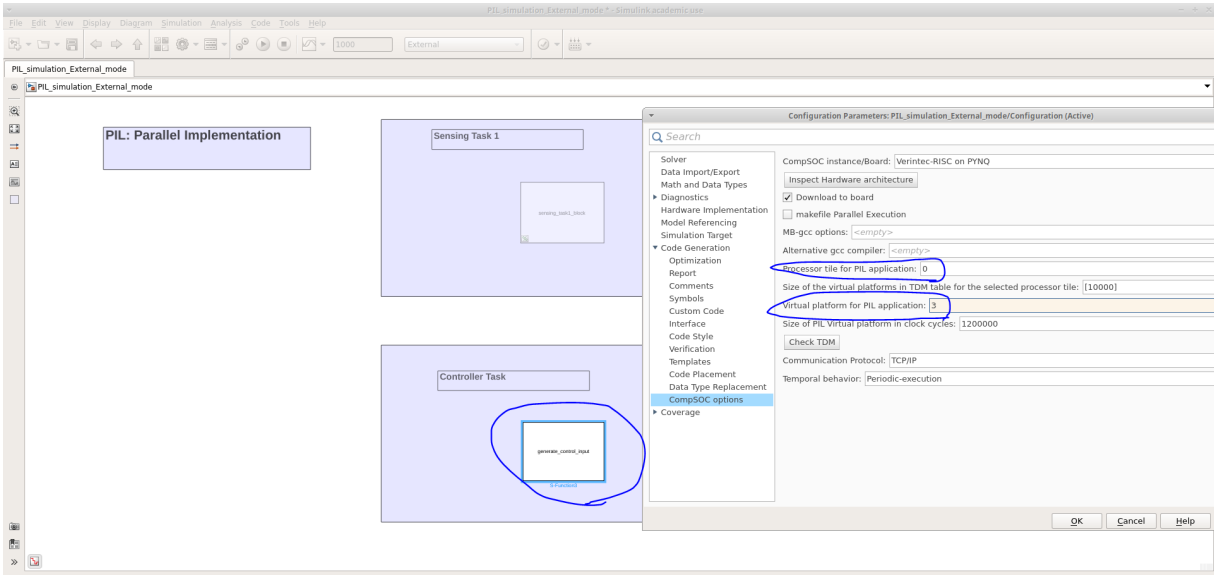


- move to the next task and comment all blocks except the one you want to run in external mode. Check the following screenshot; we have considered the controller task after finishing the merging task. Only the controller task is uncommented. Other blocks kept commenting.



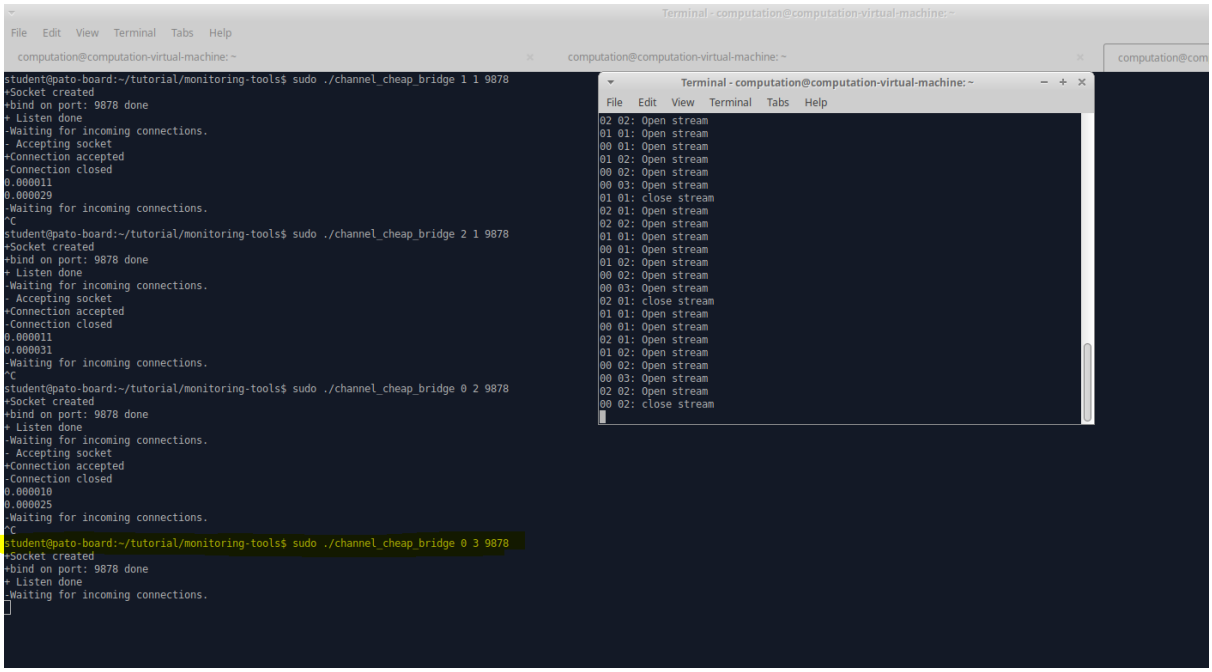
- check the configuration setting. We are running the controller task on processor 0 and partition 3. So, the following setting should be considered,

  - Processor tile for PIL application : 0
  - Virtual platform for PIL application: 3

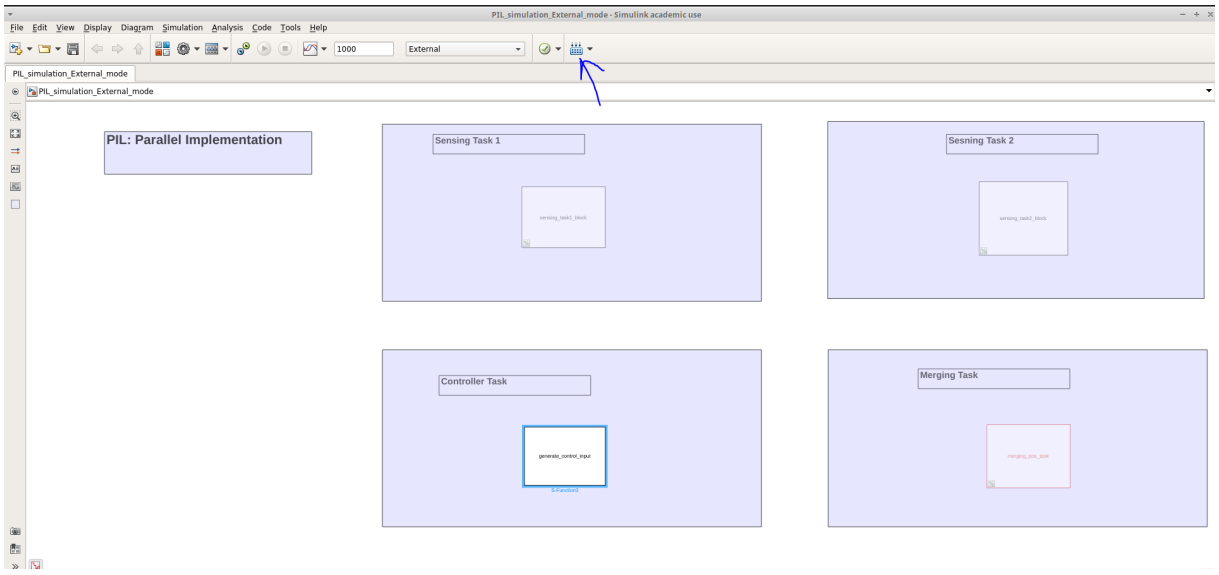


- in the same two terminal windows opened for the previous task, you can use the following highlighted commands from the screenshot.

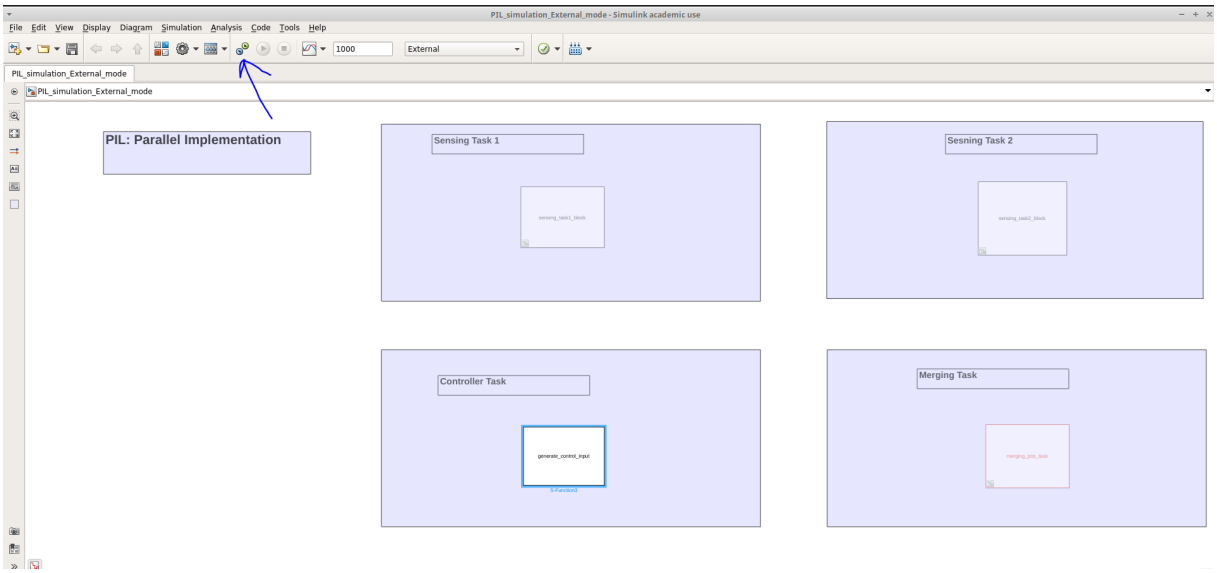
  - Terminal 1 :
    - cd tutorial
    - cd monitoring-tools/
    - sudo ./channel_cheap_bridge 0 3 9878
  - Terminal 2 :
    - cd tutorial
    - ./readout.sh



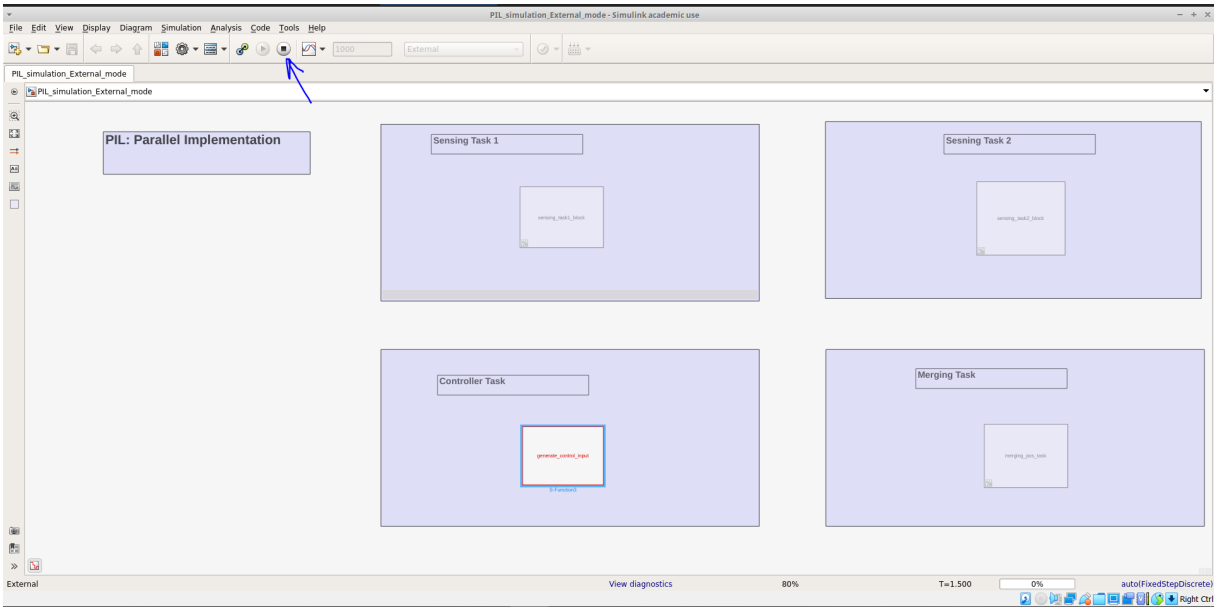
- click on the build button, as shown by the arrow in the below screenshot.



- if the build finishes successfully, ready status is available in the model. After that, run the task in external mode by clicking the connect to target button. Check the following screenshot for reference.

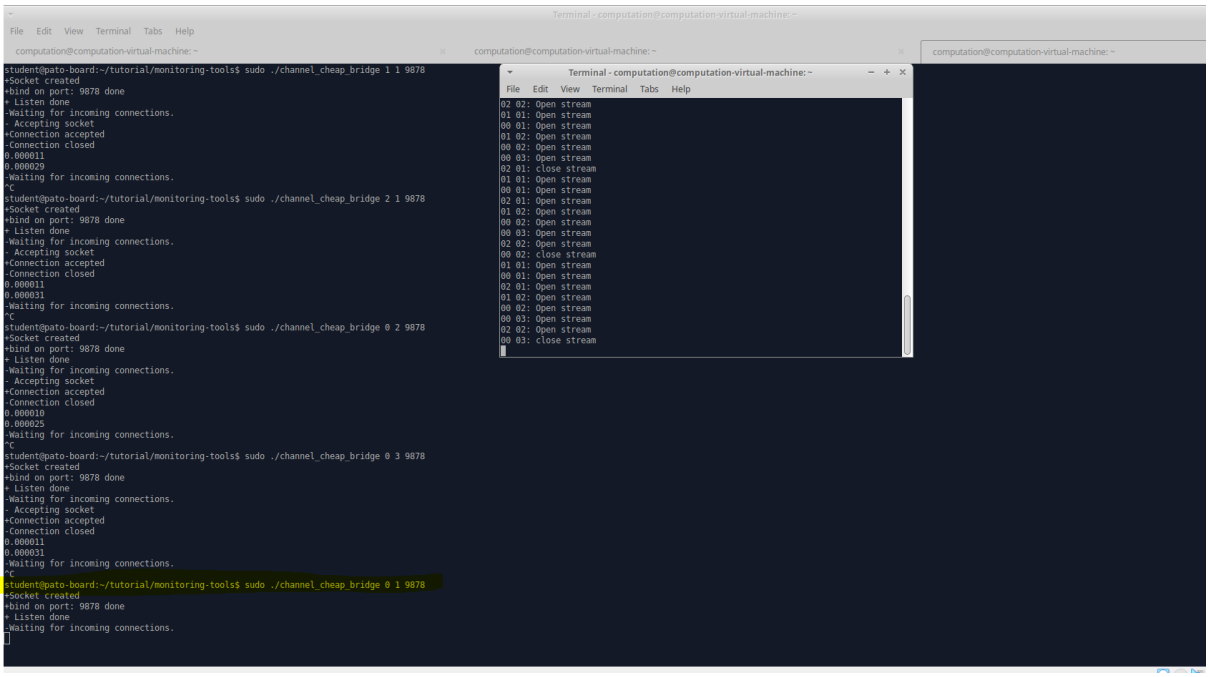


- after the above step, the controller task connects to the target and starts running on the CompSOC; let it starts and runs for some time, then stop the running of the task by clicking on the stop button; check the arrow in the screenshot. No need to run the task for a long time.

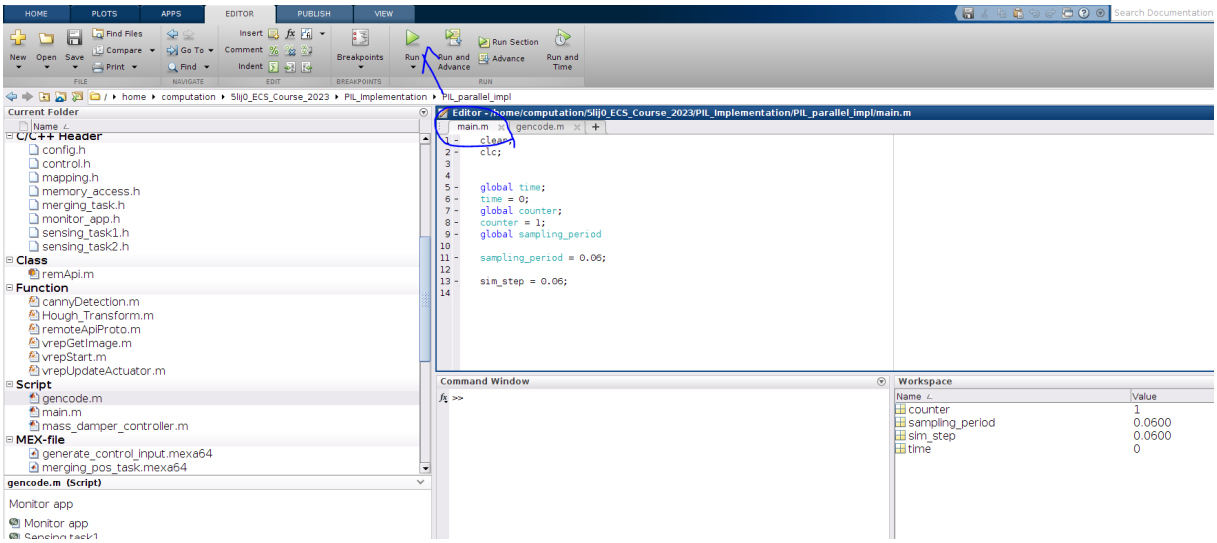


- after completing all the steps, run the following highlighted command in the same terminal window opened for the previous steps. This step is required for running the final PIL simulation, which will be run from **PIL_simulation_monitoring_app.slx** Simulink model.

  - Terminal 1 :
    - cd tutorial
    - cd monitoring-tools/
    - sudo ./channel_cheap_bridge 0 1 9878
  - Terminal 2 :
    - cd tutorial
    - ./readout.sh

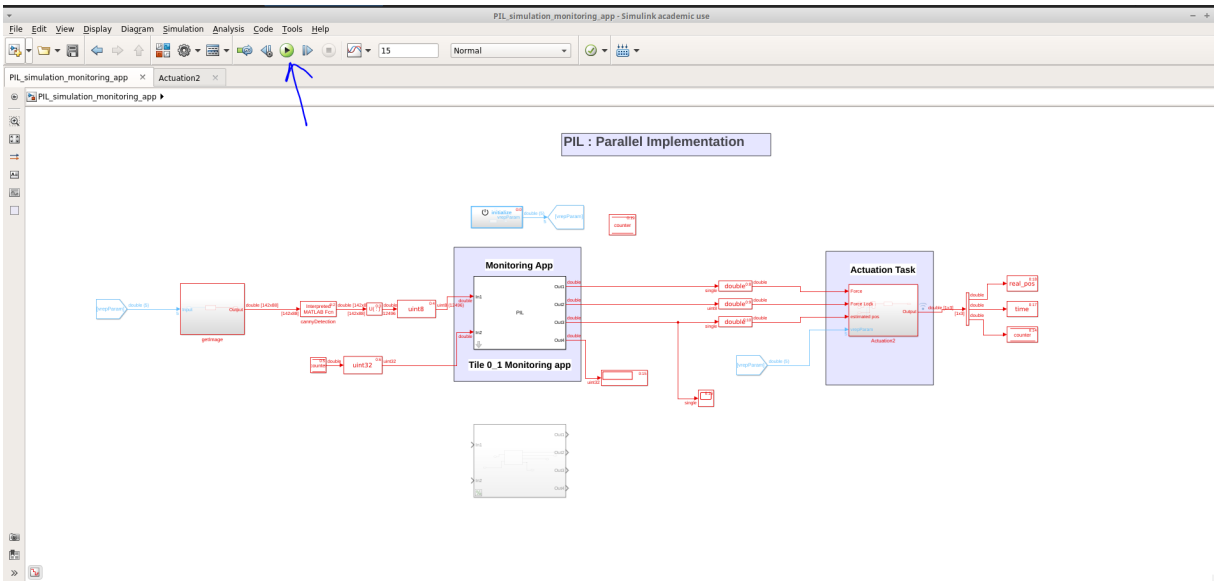


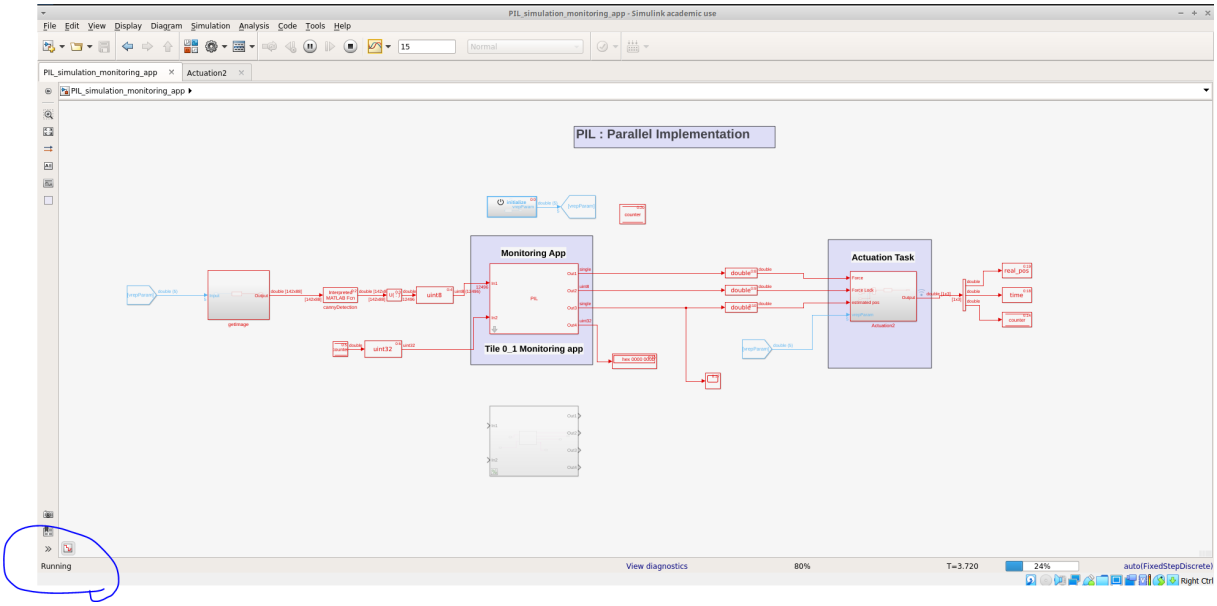
- next step is to run the main.m file and check the CoppeliaSim software; the mass-damper scene should be opened and ensure it is not running automatically; if it is running, stop it.



- open the **PIL_simulation_monitoring_app.slx** Simulink model, and click on the Run button. This will start running the simulation. It will take some time to simulate as it initializes all the required settings for running the simulation. So wait until you see the running status. Check the following screenshots for reference.

- check the current position returned from the monitoring app using the scope linked to the monitoring app PIL block. Also, visualize the output in CoppeliaSim.


PIL_simulation_monitoring_app - Simulink academic use
File Edit View Display Diagram Simulation Analysis Code Tools Help
PIL_simulation_monitoring_app
Actuation2
PIL : Parallel Implementation
Monitoring App
Tile 0_1 Monitoring app
Actuation Task

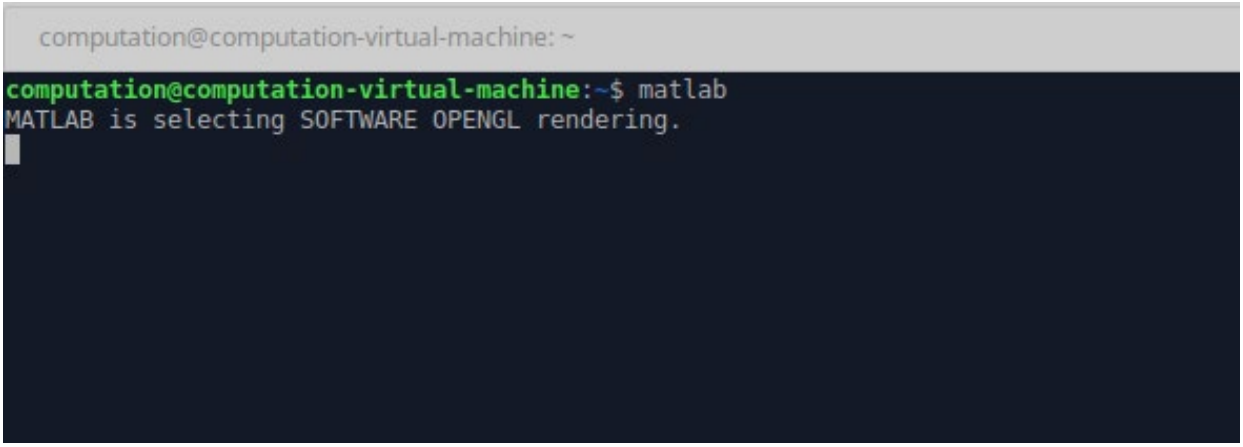

PIL_simulation_monitoring_app - Simulink academic use
File Edit View Display Diagram Simulation Analysis Code Tools Help
PIL_simulation_monitoring_app
Actuation2
PIL : Parallel Implementation
Monitoring App
Tile 0_1 Monitoring app
Actuation Task
Running
View diagnostics
80%
T=3.720
24%
auto(FixedStepDiscrete)

Section III: PiL implementation for pipeline configuration

- open MATLAB: To open MATLAB in the IMACS virtual machine, run the following command in the new terminal.

  Note: If MATLAB is already opened, then no need to reopen it.

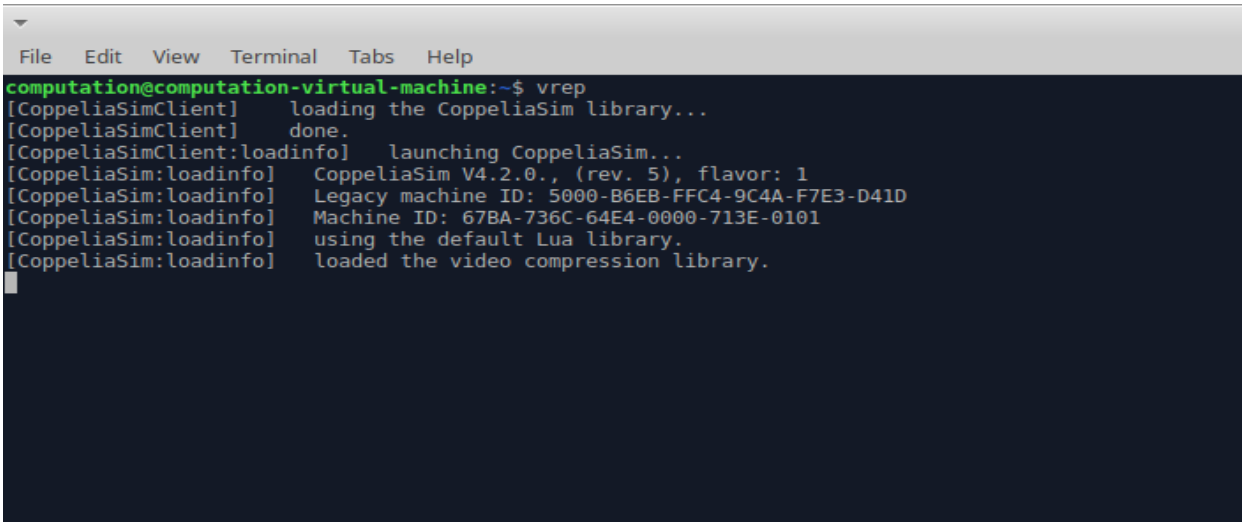
  ```
  $ matlab
  ```



- open CoppeliaSim: to open the CoppeliaSim, run the following command in a new terminal.

  Note: If CoppeliaSim is already opened, then no need to reopen it.

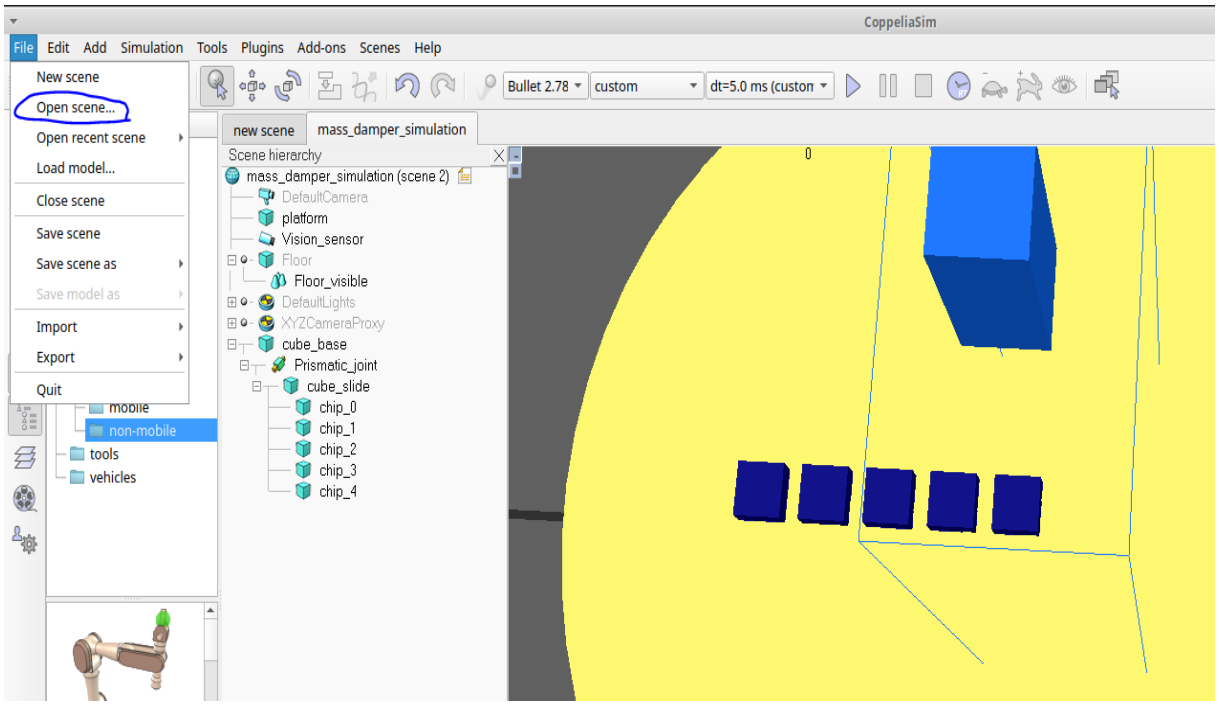
  ```
  $ vrep
  ```

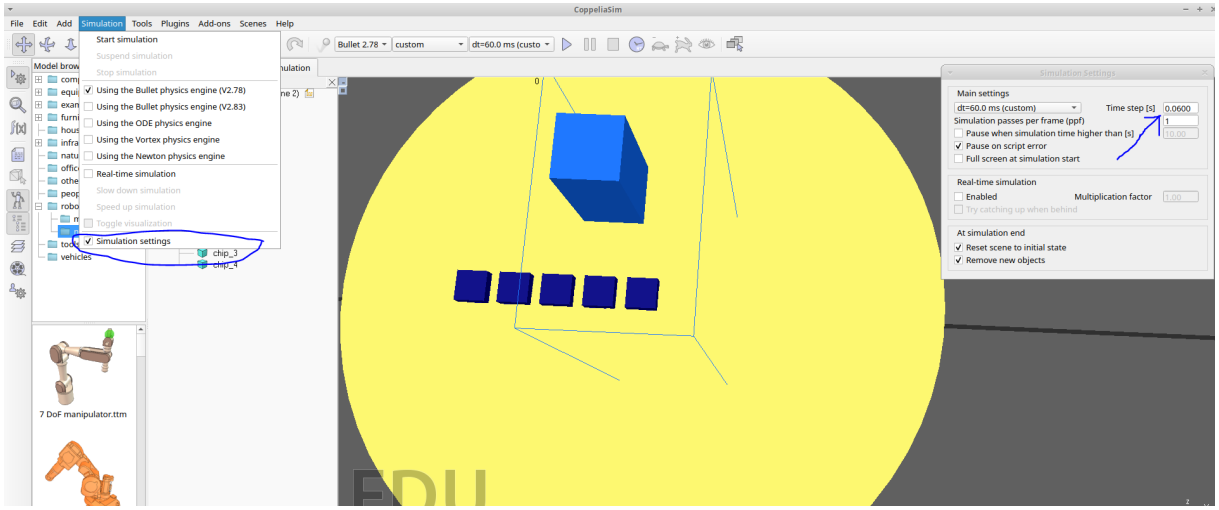


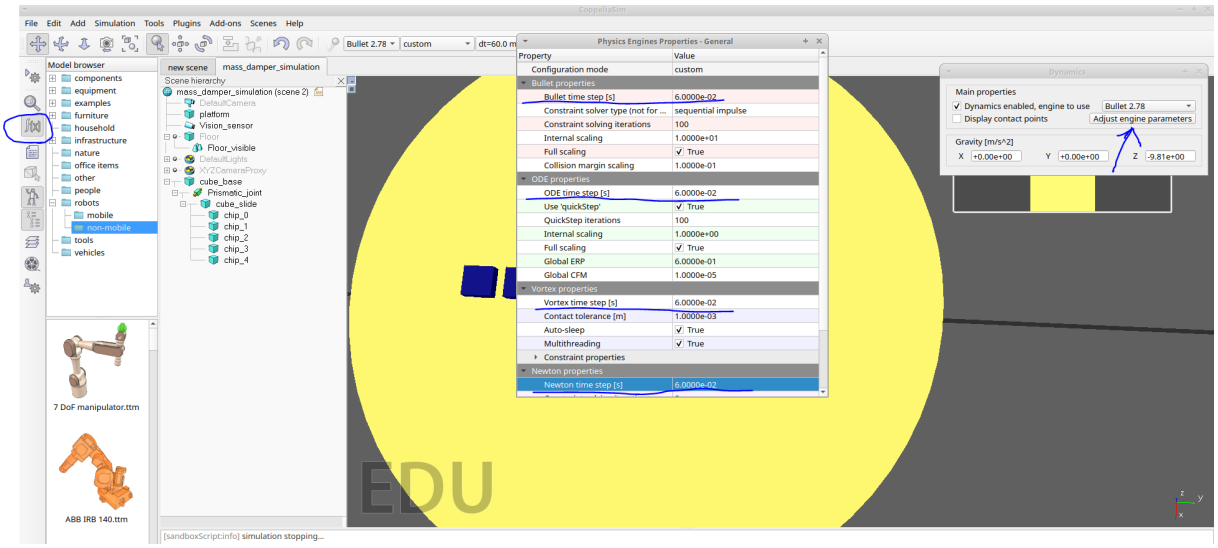
- open CoppeliaSim scene for the mass-damper system.
  - go to File→Open scene→ select the following directory /home/computation/5lij0_ECS_Course_2023/PIL_Implementation/PIL_pipeline_impl

open **mass_damper_simulation.ttt**, check the following screenshot for reference.

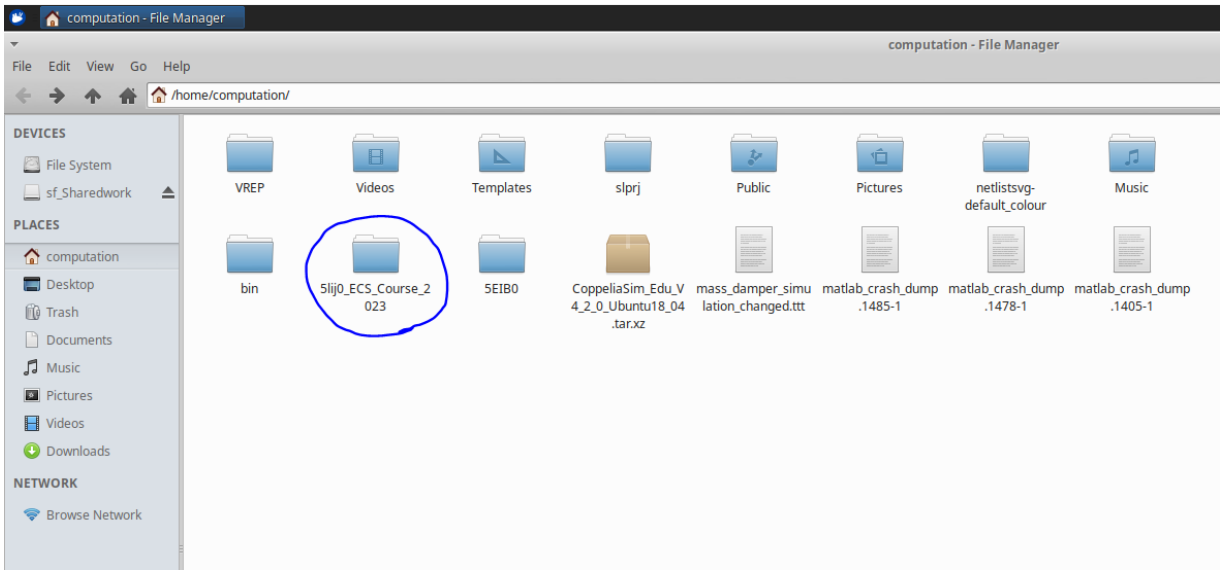
We have implemented the PIL of sequential configuration considering the 60ms sample period. The same sample period should be used in CoppeliaSim. In the case of different sample periods, students can make the changes in the CoppeliaSim, as explained below. For example, the following screenshot shows how to change the time step in the CoppeliaSim scene.

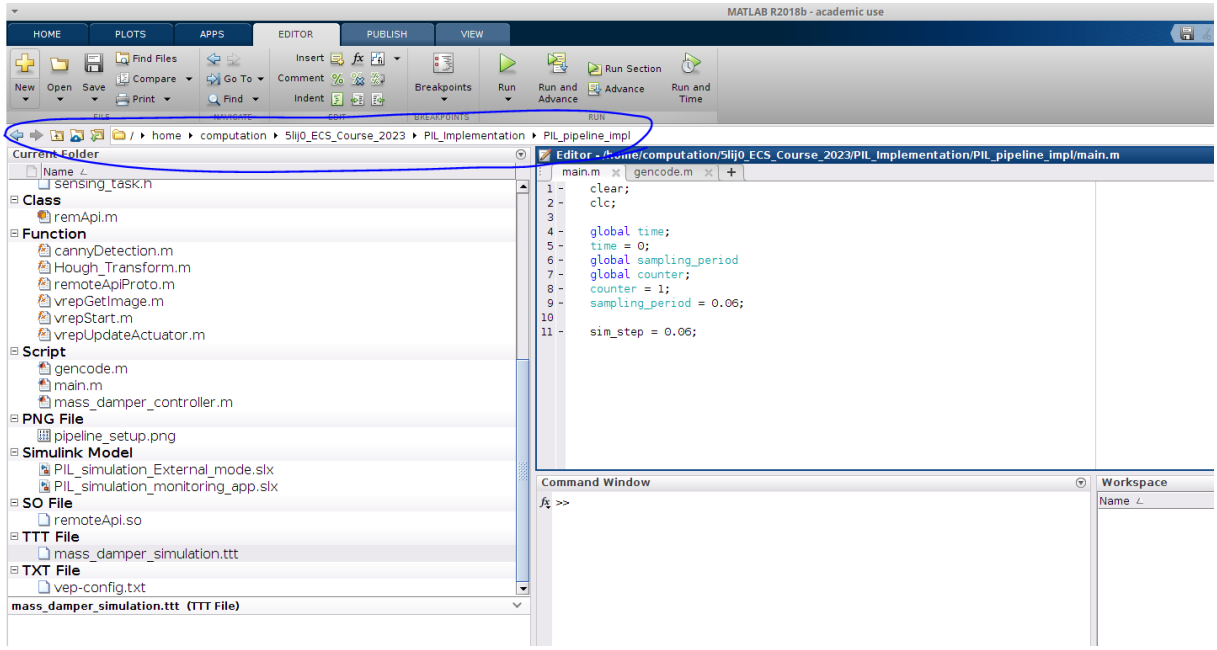






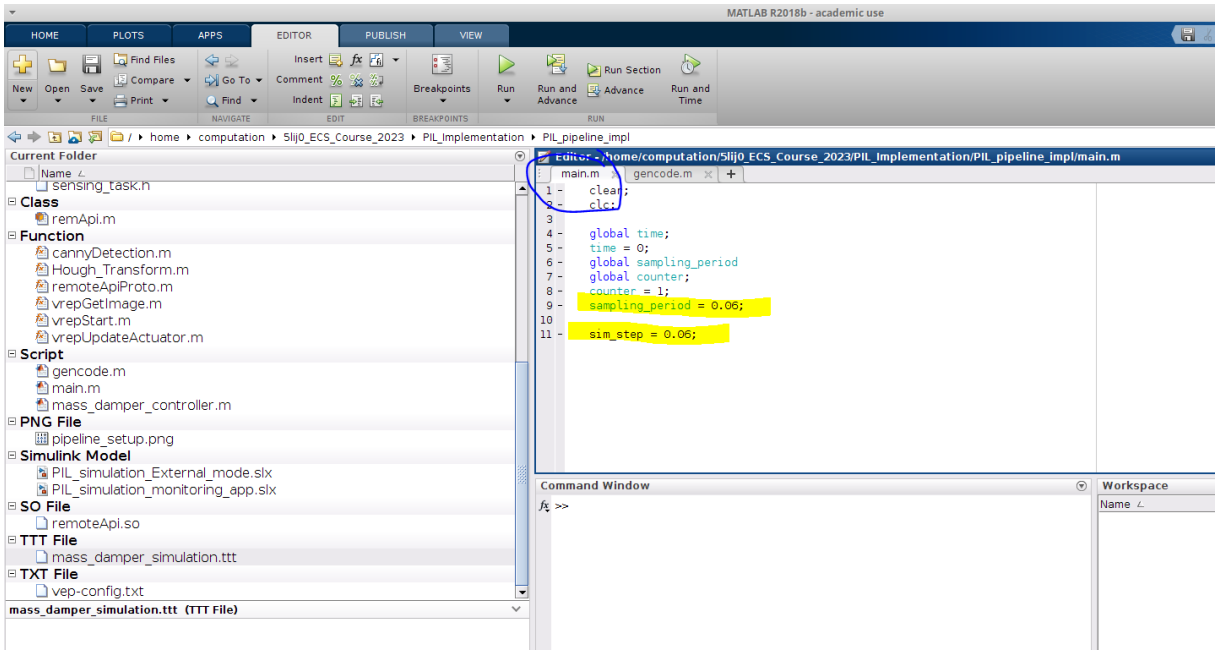
- open the folder for PiL pipeline implementation.
  - go to the following folder path and open in MATLAB
    computation→5lij0_ECS_Course_2023→PIL_Implementation→PIL_pipeline_impl



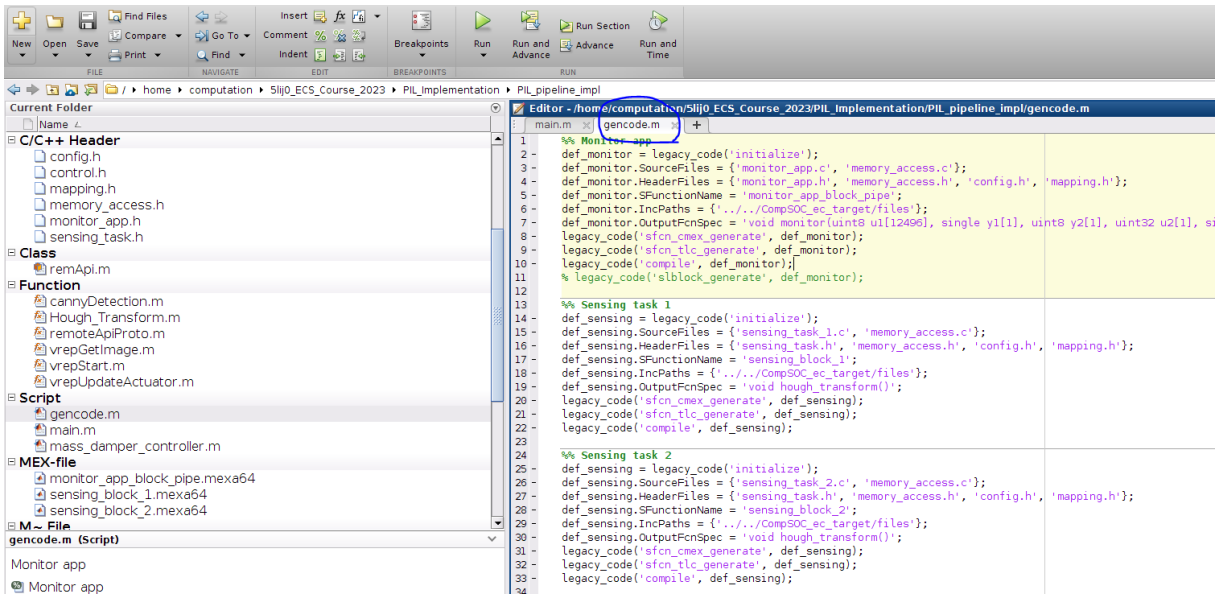


- open the main.m file and make changes for the highlighted parameters, as shown in the following screenshot.

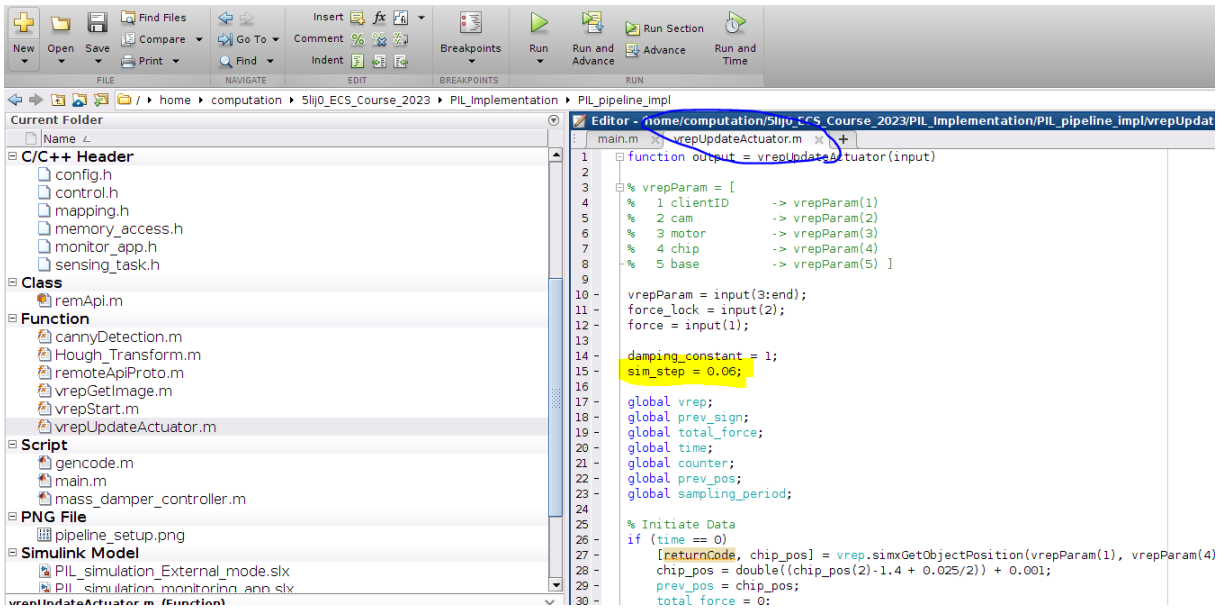
  Note: In this screenshot, we have explained for the 60ms sample period, students can use different sample periods according to the implementation design and make the changes for the following parameters. Once the changes are done, run the main.m script. This file includes the initialization, which is required for the PiL simulation.



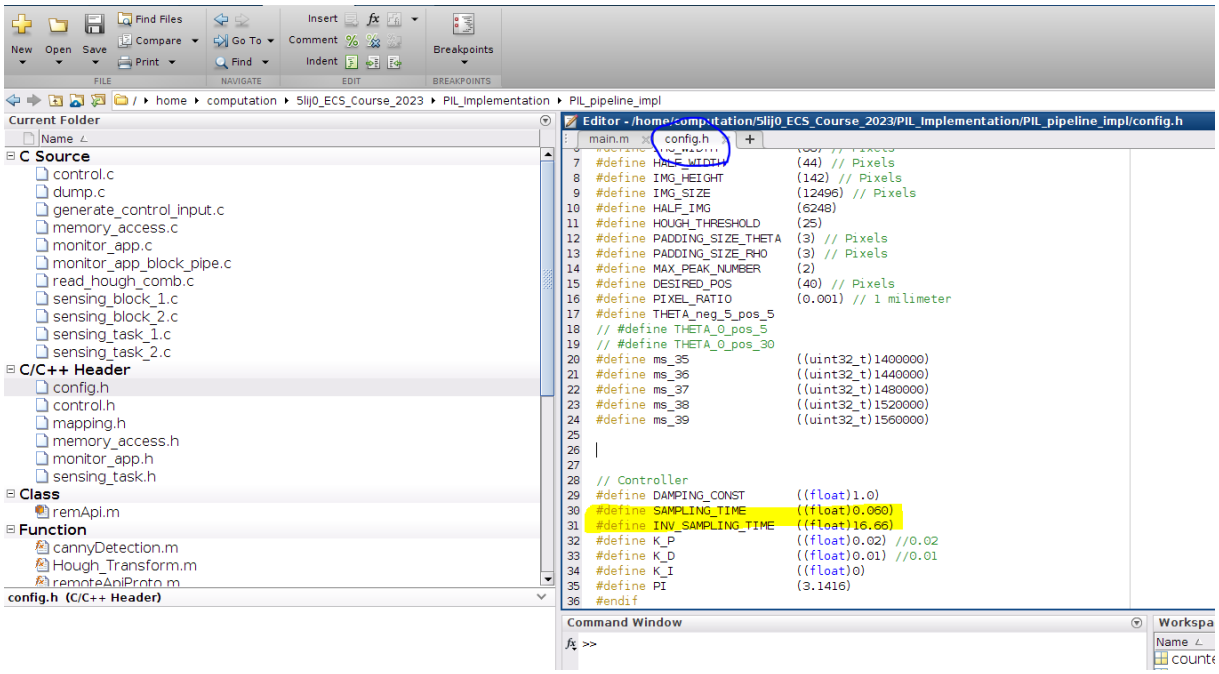
- open the gencode.m file and run the script. This file uses legacy code tool functionality, integrating the existing C or C++ functions into MATLAB Simulink. Once this gencode.m runs, it creates the S-function block inside the Simulink model.



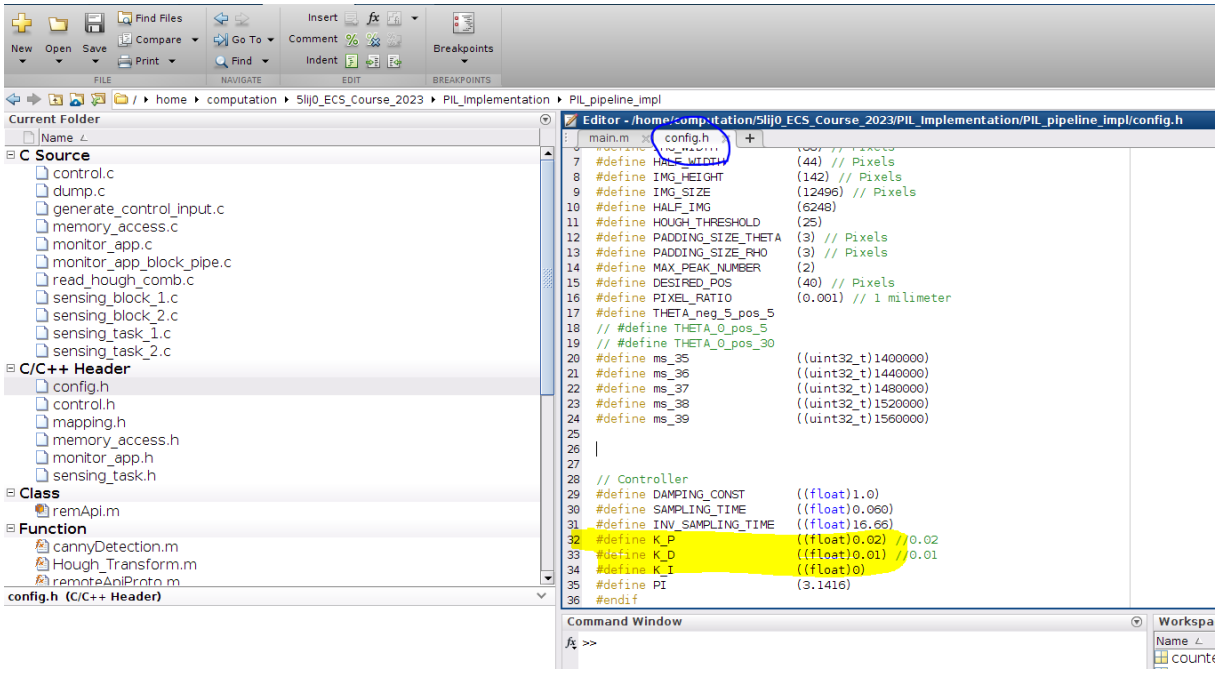
- open the **vrepUpdateActuator.m** file. Make the necessary changes in the file, as shown below in the screenshot. This file is required for the actuation.



- open the **config.h** file. Make the necessary changes in the file, as shown below in the screenshot. This file contains all the required parameters used in the PiL simulation.



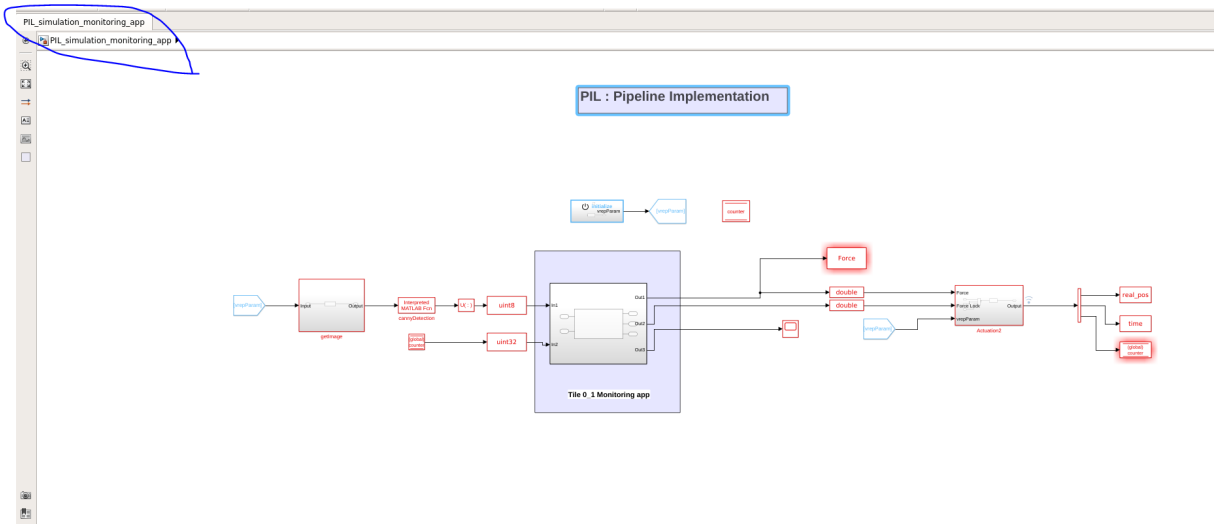
- for tuning the PID controller gains, students can change the gains in the **config.h** file, check the below screenshot.



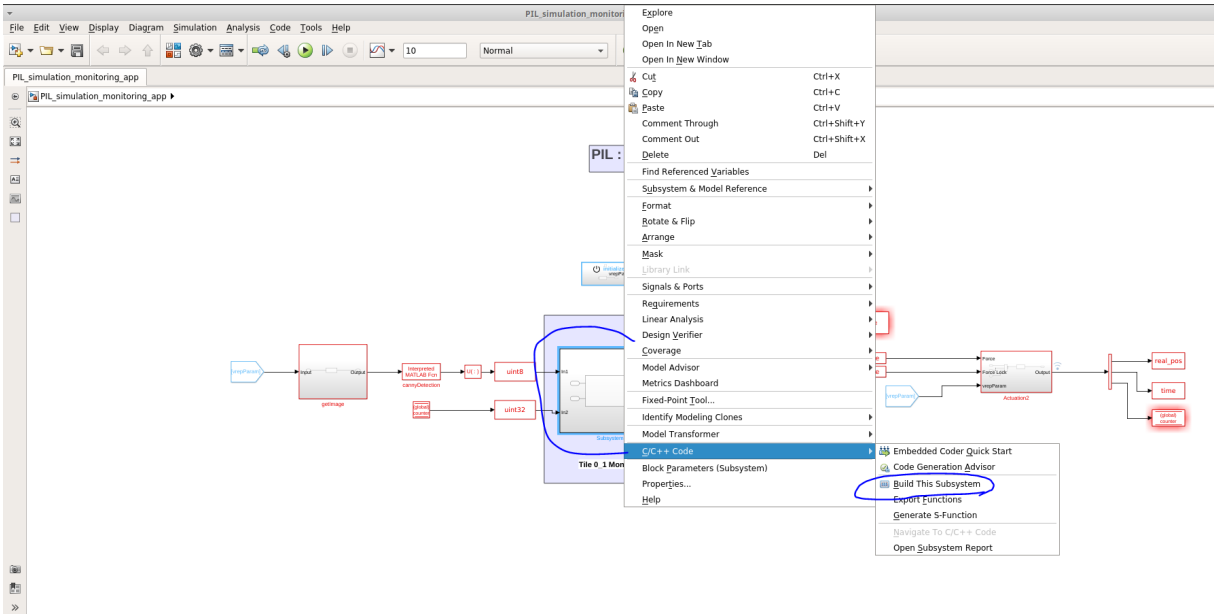
- once the steps mentioned above are done, open the PiL simulation file named **PIL_simulation_monitoring_app.slx**

  Students can find this file in the following directory:

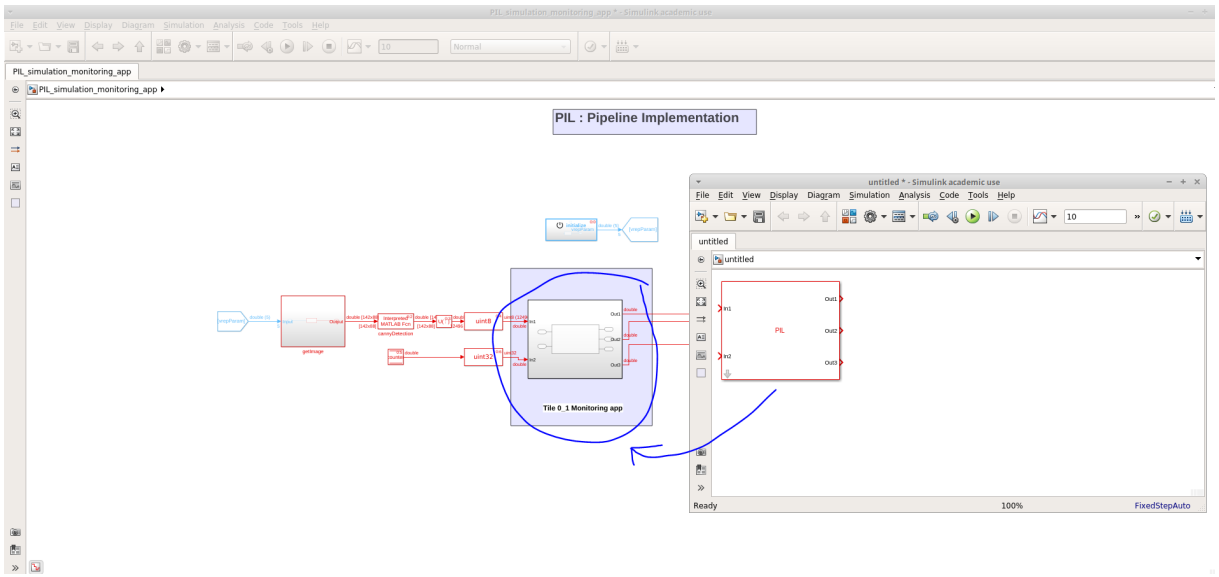
  computation→5lij0_ECS_Course_2023→PIL_Implementation→PIL_pipeline_impl



- right-click on the block, select the C/C++ code option, and click on build this subsystem. Check the following screenshot for reference.



- once, the build process is completed successfully; it will generate the PIL block; check the following screenshot:



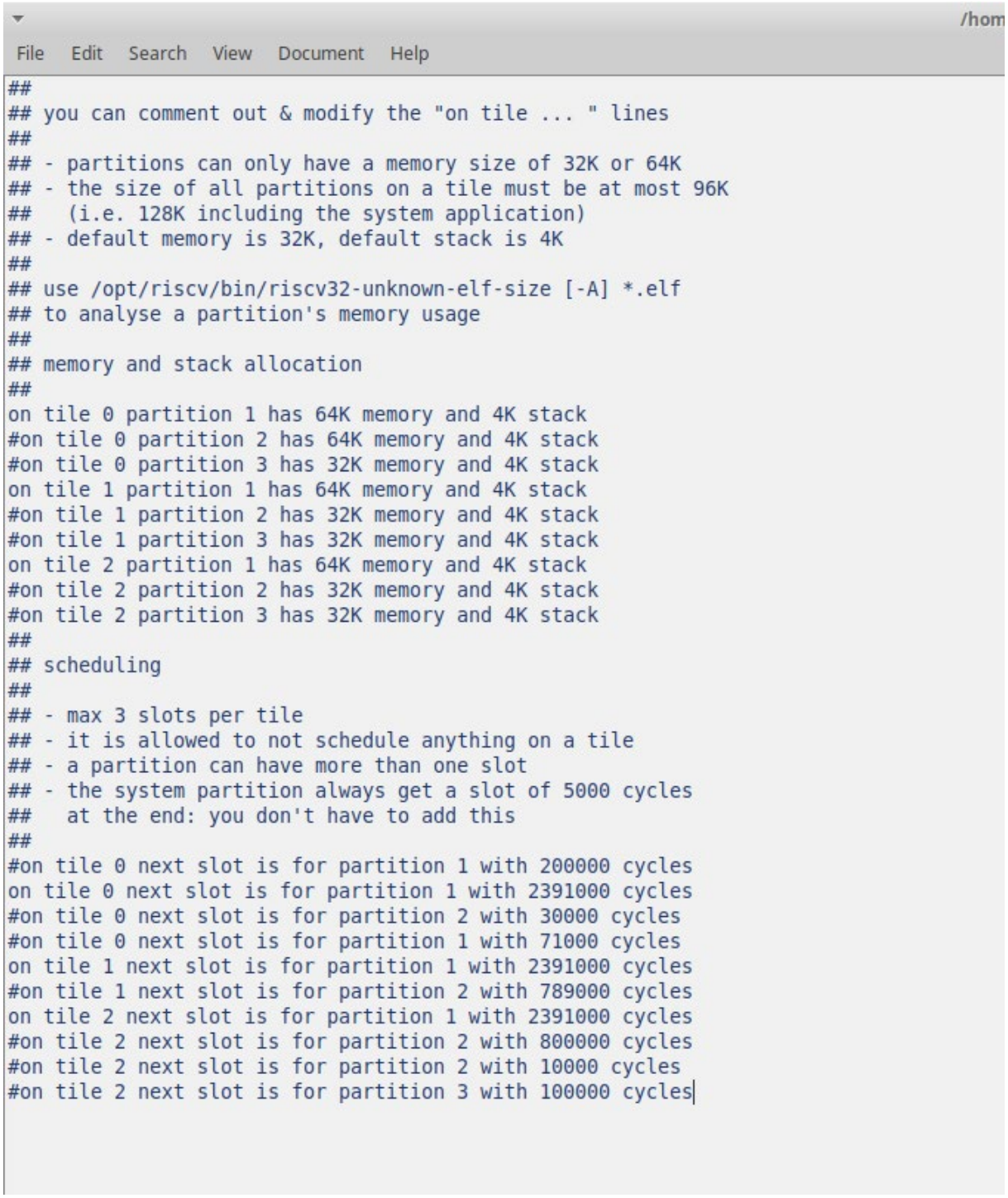
- after completing the above steps, check the vep_config.txt file. In this example, we have considered the 60ms sample period. Therefore, in the vep_config.txt, scheduling is given according to the 60ms sample period.
- we have provided the vep_config.txt file in the following folder

  /home/computation/5lij0_ECS_Course_2023/PIL_Implementation/PIL_pipeline_impl
- students need to copy the contents from the above vep_config file into the main **vep-config.txt** file, which is inside the following folder path :

  or the student can enter the TDM scheduling calculation directly in the vep-config.txt file, which is stored in the following location.

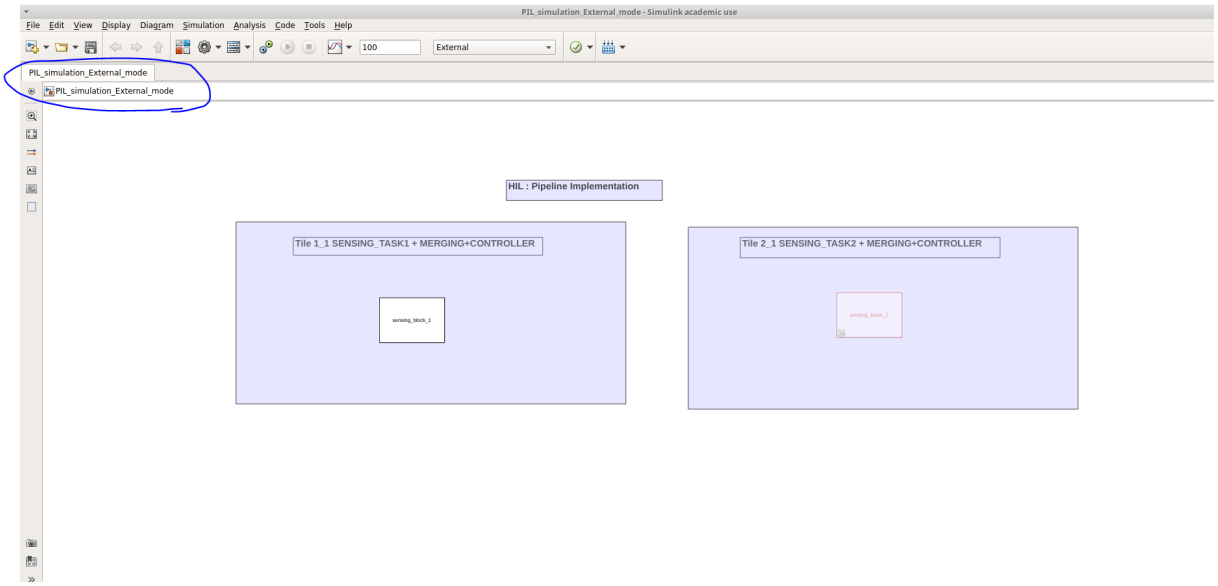
  /home/computation/CompSOC_ec_target/CompSOC_ec/+CompSOC_ec/vep-config.txt

- the contents in the vep_config file should be as follows:

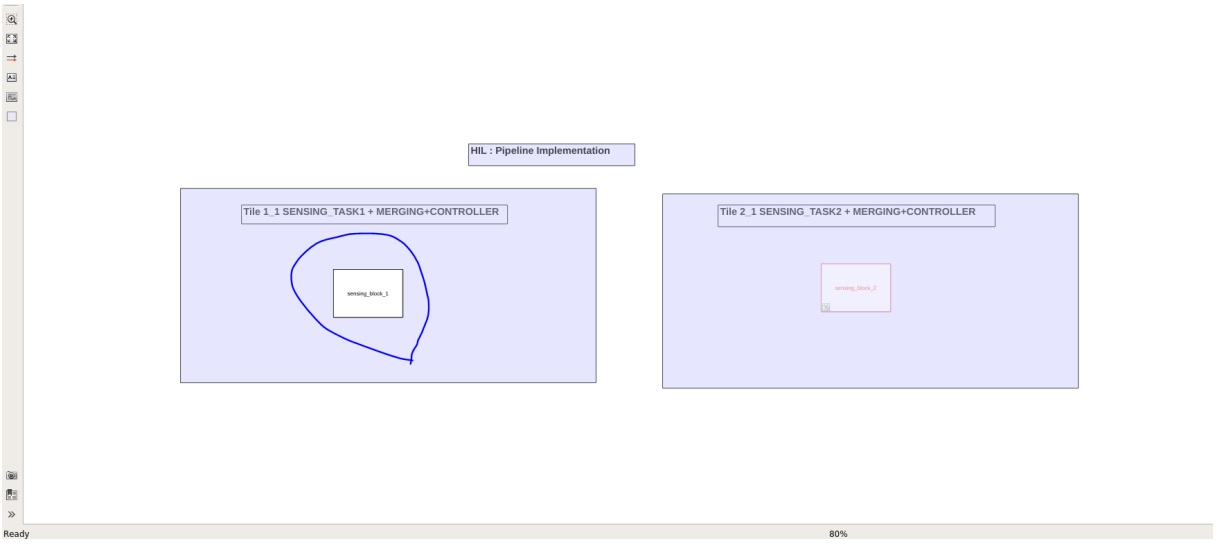


- open the PIL_simulation_External_mode.slx. The folder path to this Simulink model is as follows:

  /home/computation/5lij0_ECS_Course_2023/PIL_Implementation/PIL_pipeline_impl
- PIL_simulation_External_mode.slx file has two blocks, i.e., one block for (sensing task 1 + merging + controller tasks) and other blocks for (sensing task 2 + merging + controller tasks), which will run in the external mode on CompSOC based on the TDM scheduling. To run the task independently in the external mode, follow the below steps and check the screenshot for reference:

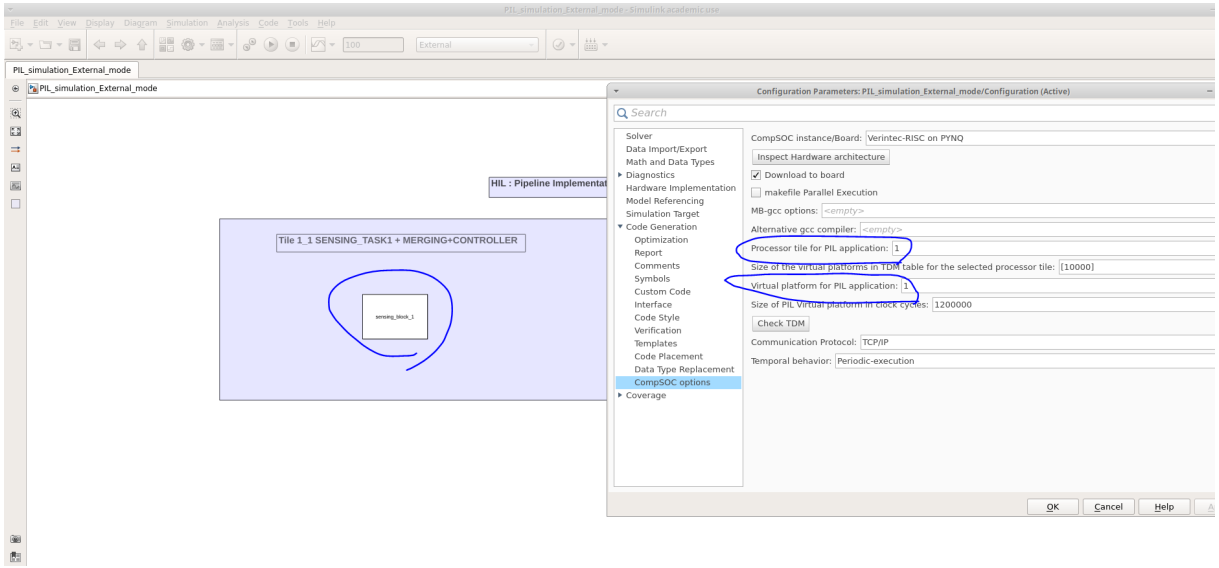


- to run the (sensing task 1 + merging + controller tasks) in the external mode, comment the other Simulink block and only uncomment the block which needs to be run. In the below screenshot, we consider the (sensing task 1 + merging + controller tasks) block first, and other blocks kept commenting.



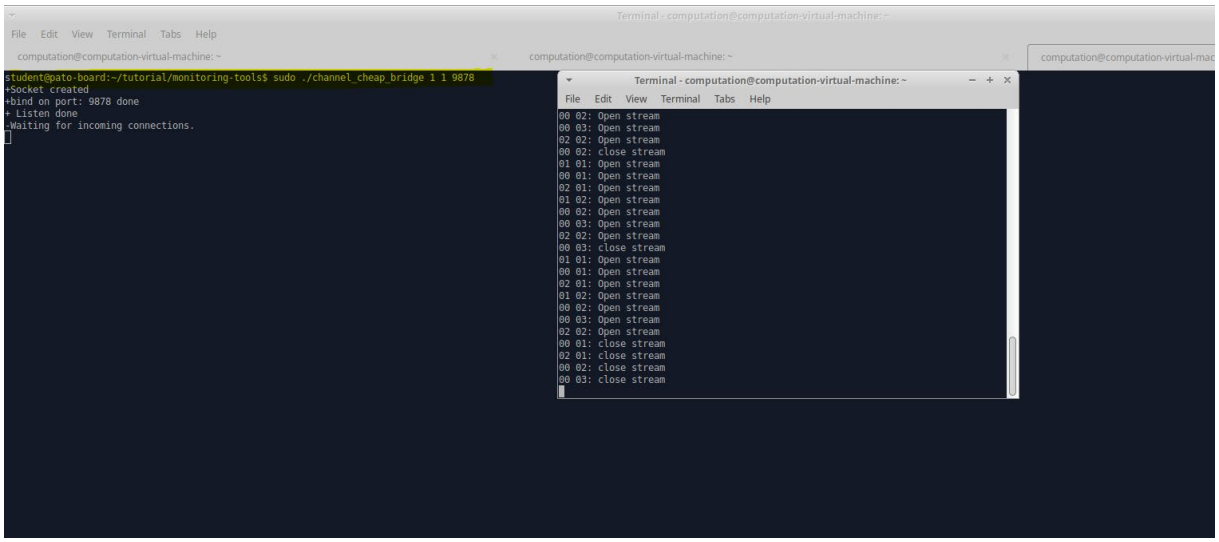
- open the configuration setting and check the required setting for the particular task. In this example, we run (sensing task 1 + merging + controller tasks) on processor 1 and partition 1. Therefore, consider the following changes :

  - Processor tile for PIL application: 1
  - Virtual platform for PIL application: 1

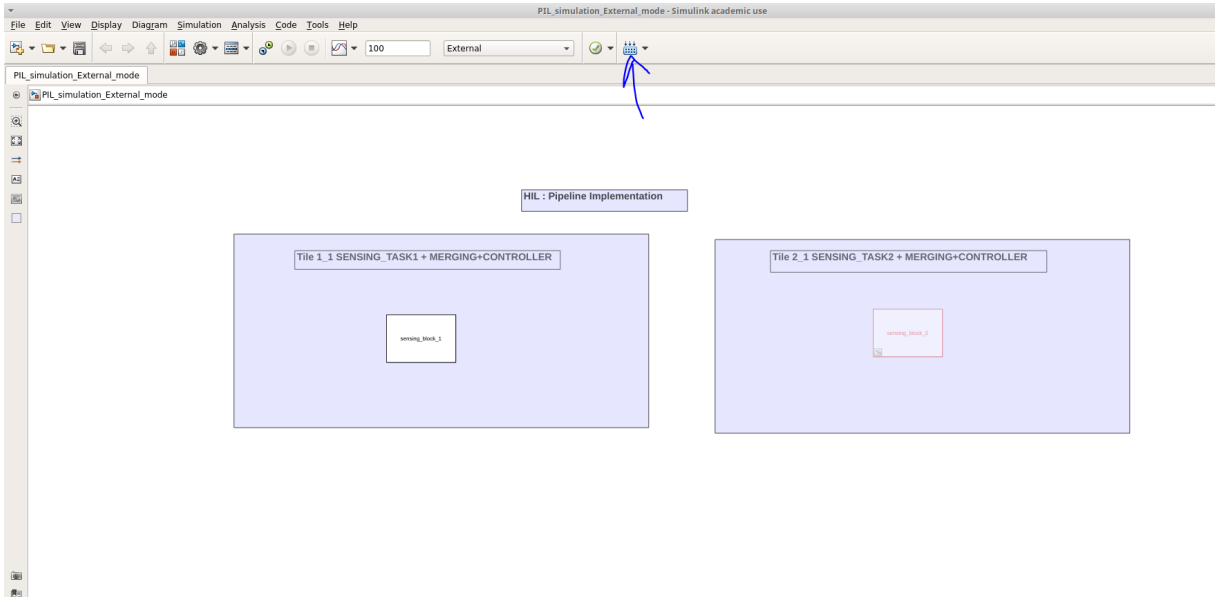


- open the two new terminal windows, connect the PYNQ board, and use the following command:
  - ensure that vep-config.txt is updated in the main CompSOC folder as per the required TDM scheduling. This step is essential before running the task in external mode on CompSOC.
  - vep-config file path: /home/computation/CompSOC_ec_target/CompSOC_ec/+CompSOC_ec/vep-config.txt
  - commands for running the sensing task on processor 1 and partition 1

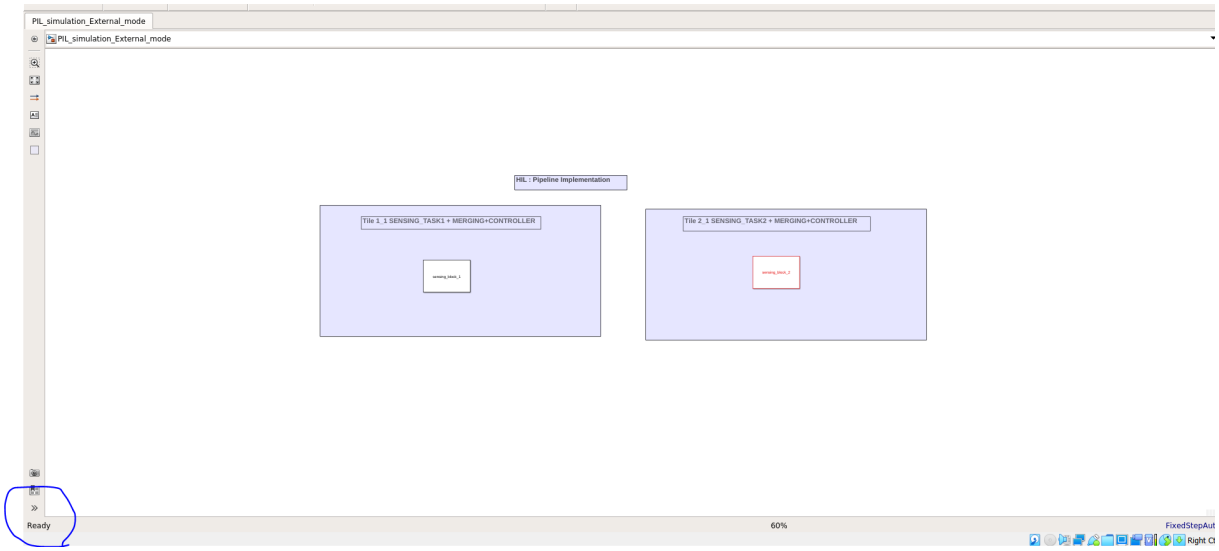
    - Terminal 1 :
      - cd tutorial
      - cd monitoring-tools/
      - sudo ./channel_cheap_bridge 1 1 9878
    - Terminal 2 :
      - cd tutorial
      - ./readout.sh



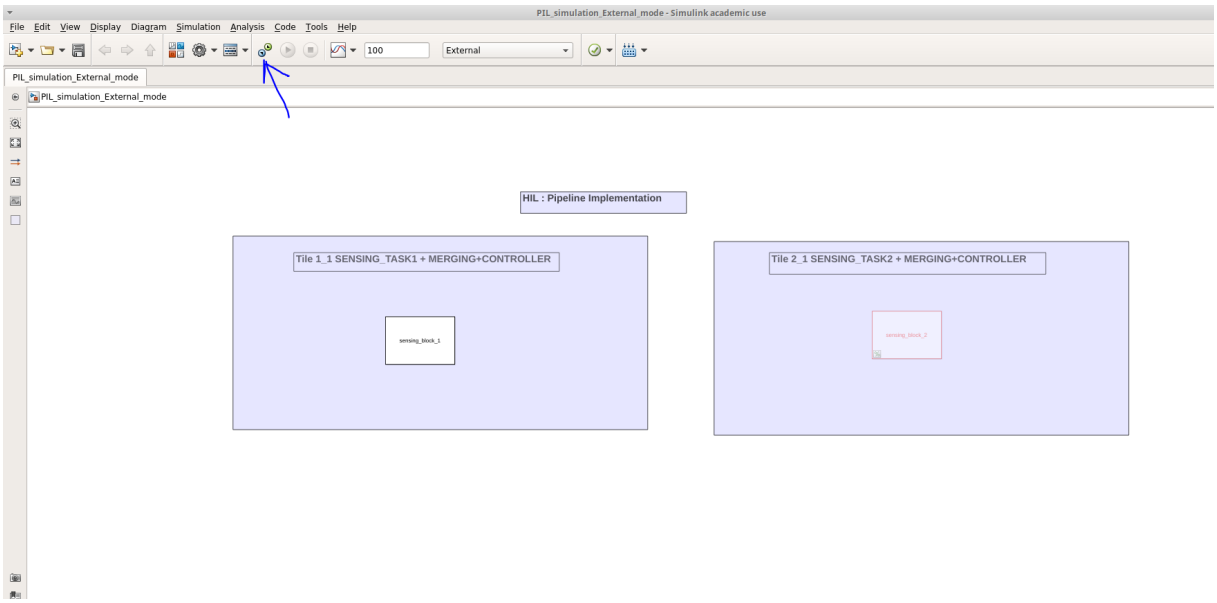
- click on the build button, as shown by the arrow in the below screenshot.



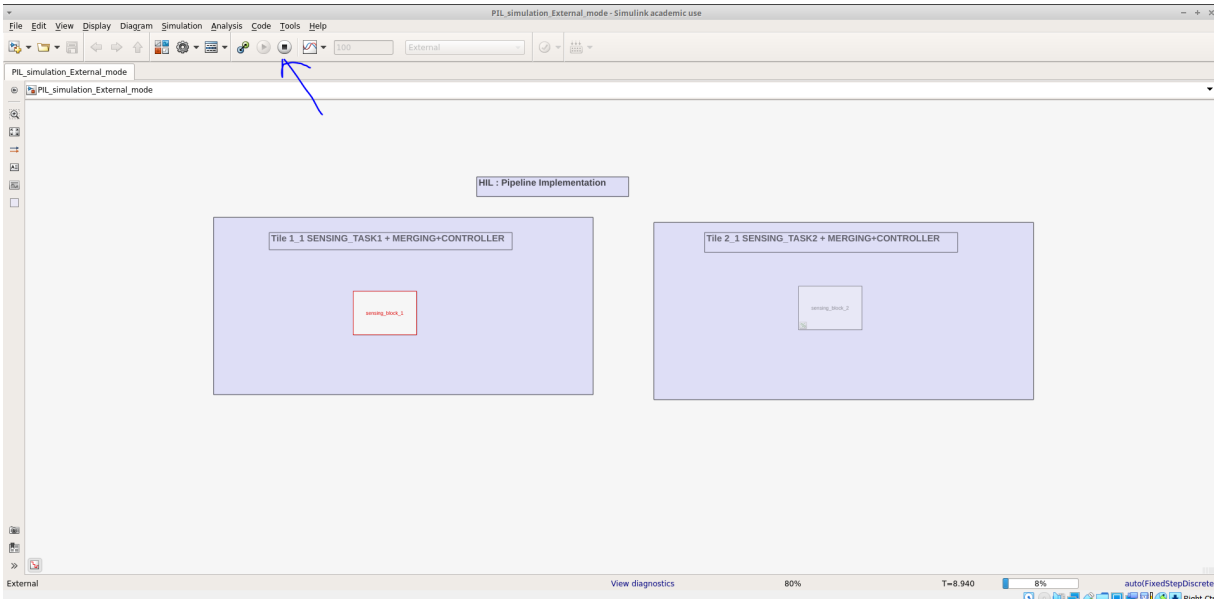
- after the build finishes, the Simulink model shows ready status; check the circle in the following screenshot.



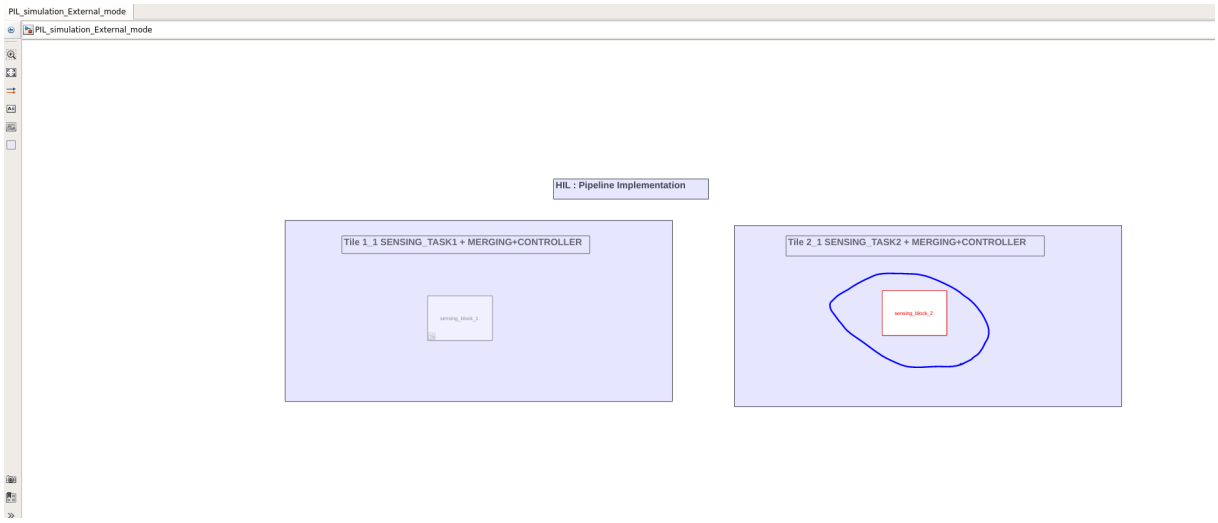
- if the build finishes successfully, ready status is available in the model. After that, run the task in external mode by clicking the connect to target button. Check the following screenshot for reference.



- after the above step, the (sensing task 1+merging+controller) connects to the target and starts running on the CompSOC; let it starts and runs for some time; check the highlighted circle in the below screenshot, then stop the running of the task by clicking on the stop button, check the arrow in the screenshot. No need to run the task for a long time.

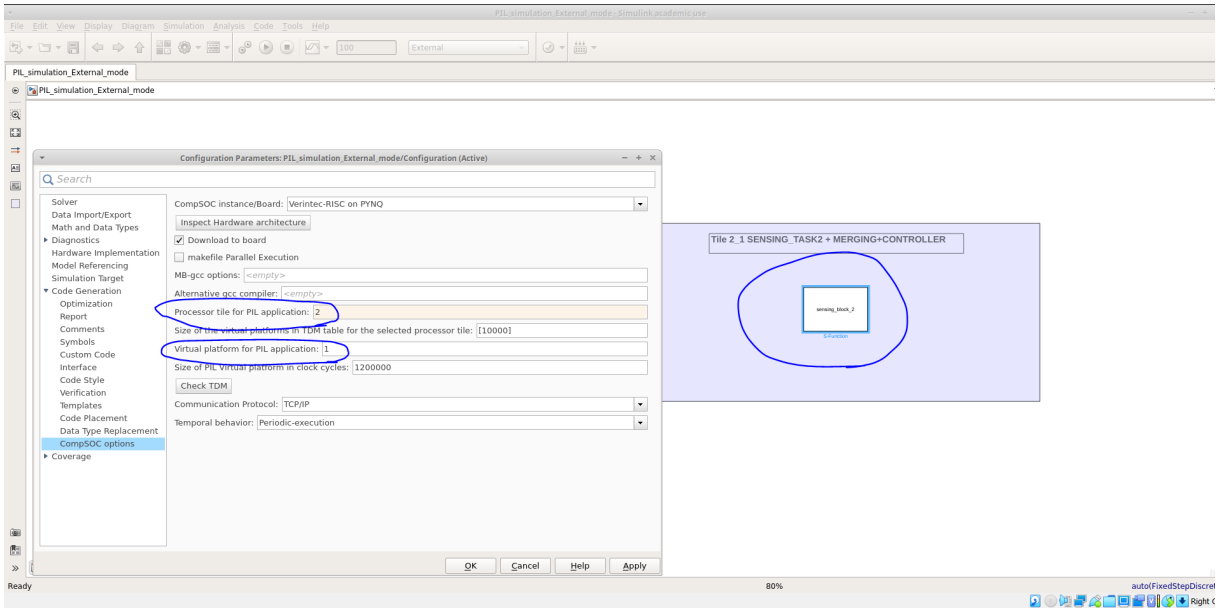


- move to the next task and comment all blocks except the one you want to run in external mode. Check the following screenshot; we have considered the (sensing task 2 + merging + controller tasks). Only the (sensing task 2 + merging + controller tasks) block is uncommented. Other block kept commenting.



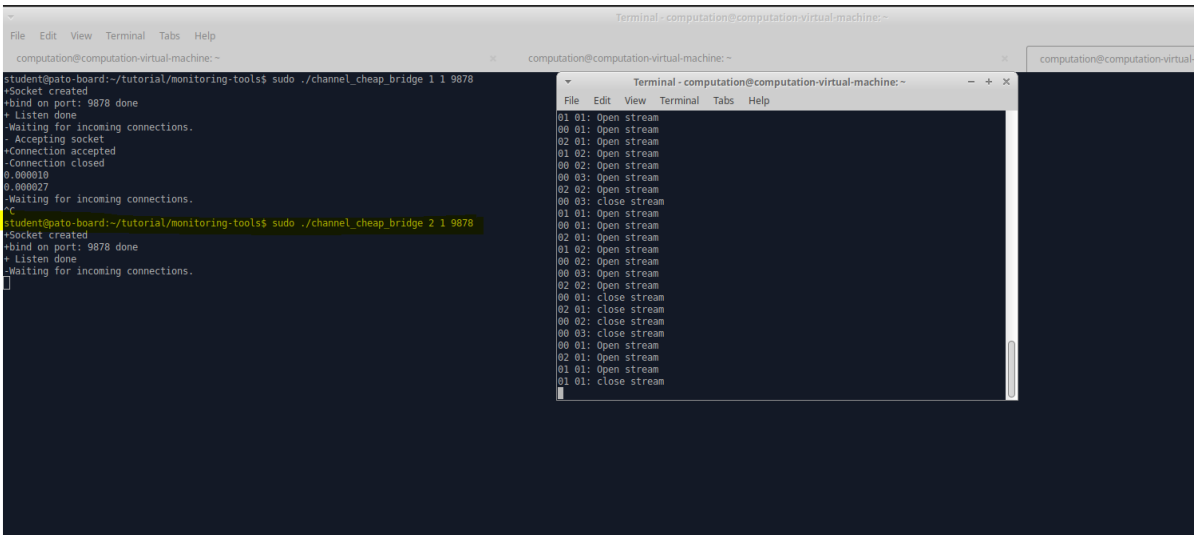
- open the configuration setting and check the required setting for the particular task. In this example, we run (sensing task 2 + merging + controller tasks) on processor 2 and partition 1. Therefore, consider the following changes :

  - Processor tile for PIL application: 2
  - Virtual platform for PIL application: 1

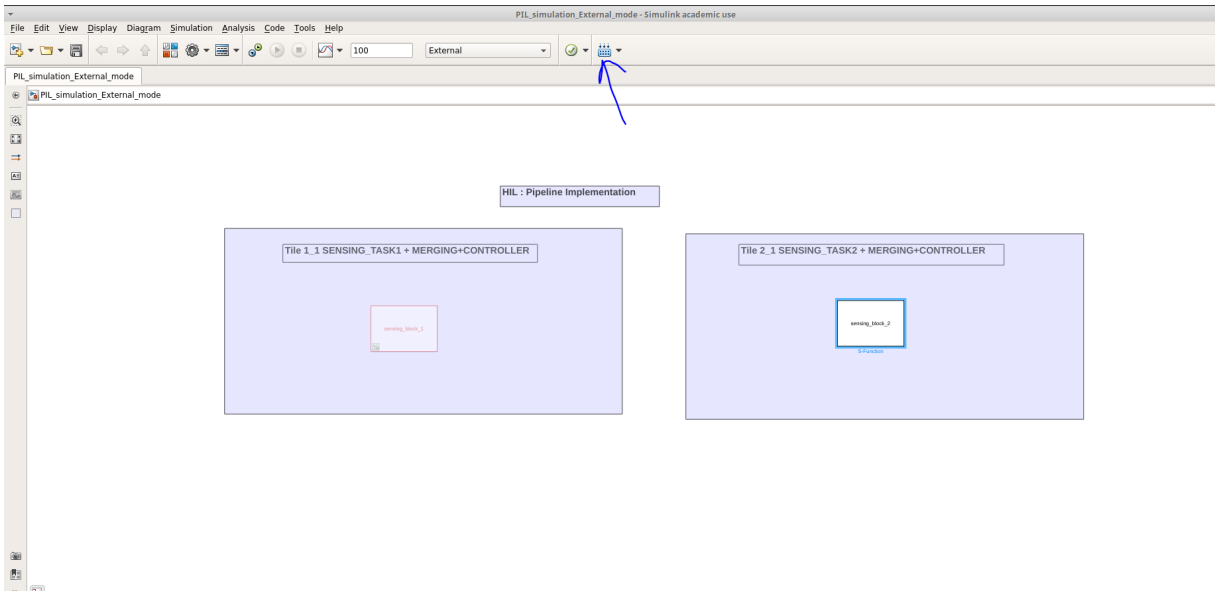


- open the two new terminal windows, connect the PYNQ board, and use the following command:
    - ensure that vep-config.txt is updated in the main CompSOC folder as per the required TDM scheduling. This step is essential before running the task in external mode on CompSOC.
    - vep-config file path: /home/computation/CompSOC_ec_target/CompSOC_ec/+CompSOC_ec/vep-config.txt
    - commands for running the sensing task on processor 2 and partition 1

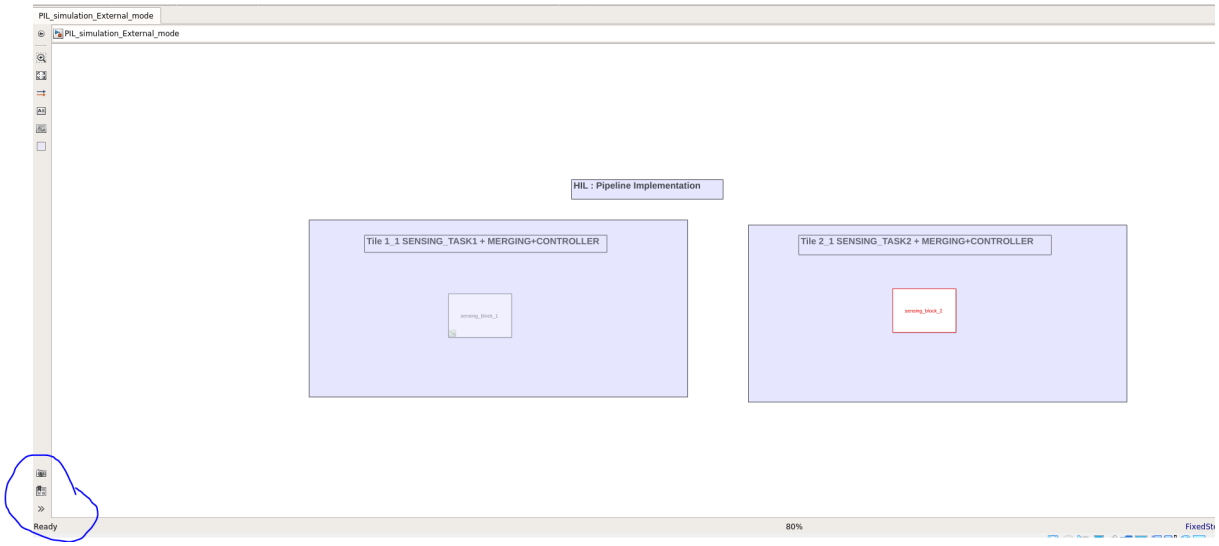
        - Terminal 1 :
            - cd tutorial
            - cd monitoring-tools/
            - sudo ./channel_cheap_bridge 2 1 9878
        - Terminal 2 :
            - cd tutorial
            - ./readout.sh



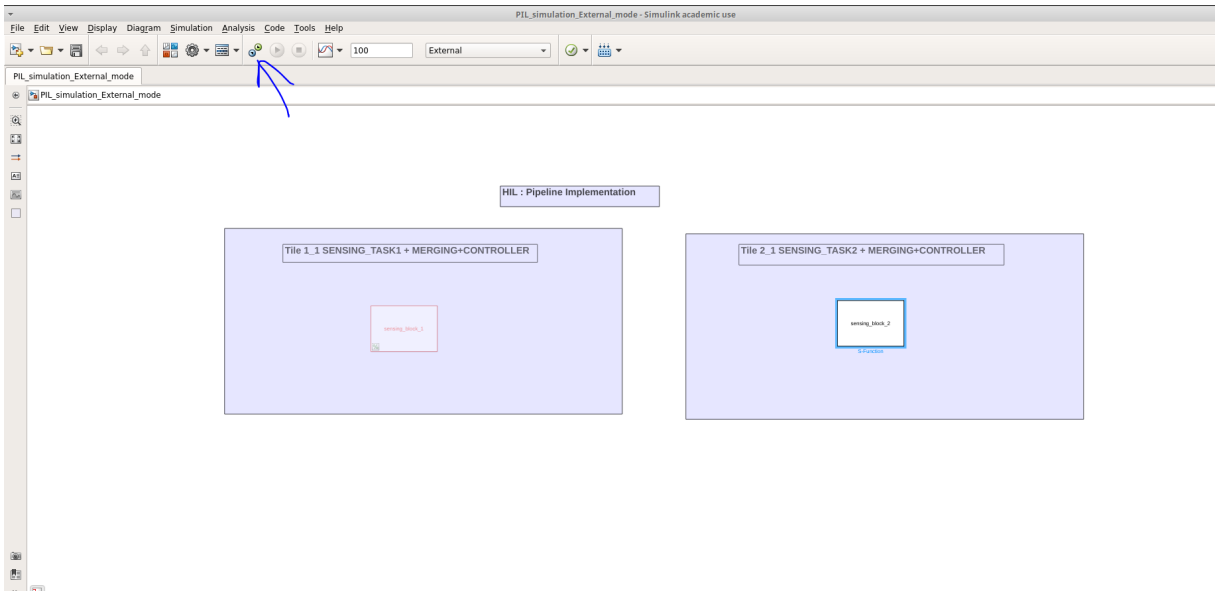
- click on the build button, as shown by the arrow in the below screenshot.



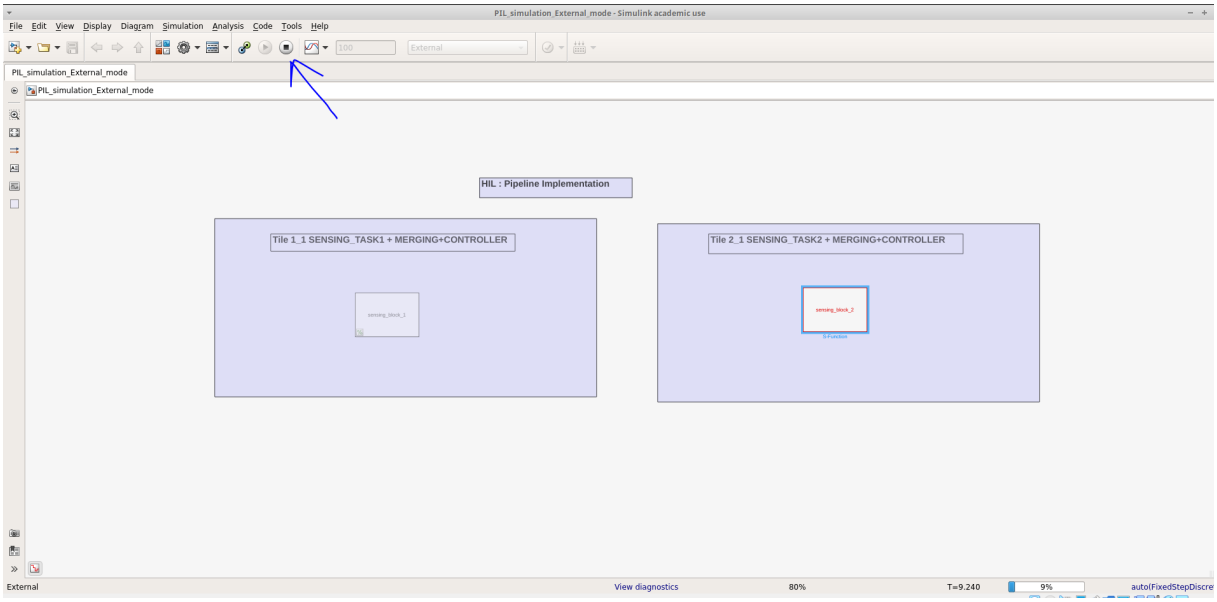
- after the build finishes, the Simulink model shows ready status; check the circle in the following screenshot.



- if the build finishes successfully, ready status is available in the model. After that, run the task in external mode by clicking the connect to target button. Check the following screenshot for reference.

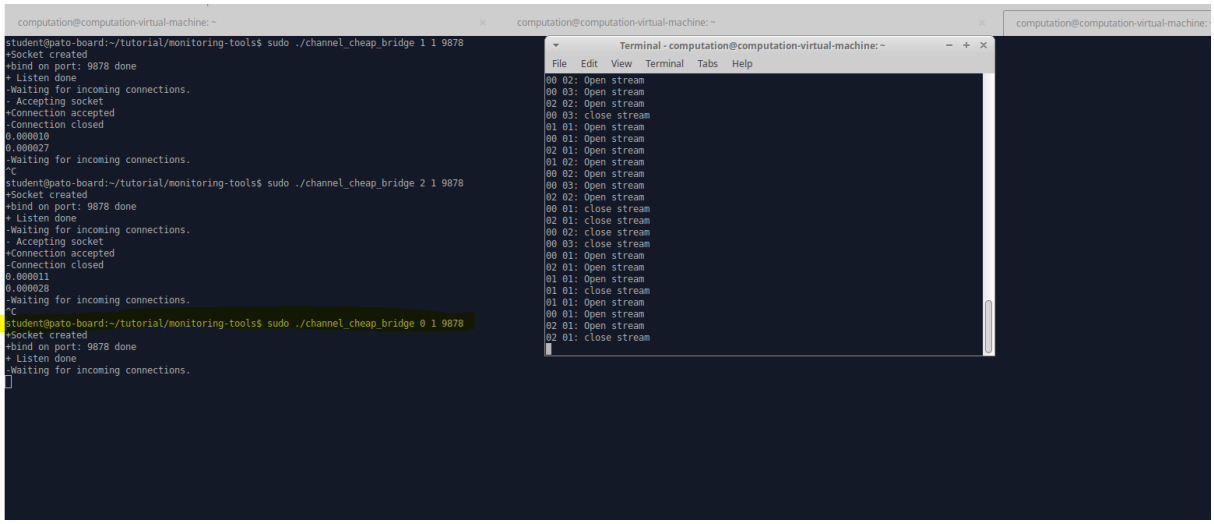


- after the above step, the (sensing task 2 + merging + controller) connects to the target and starts running on the CompSOC; let it starts and runs for some time; check the highlighted circle in the below screenshot, then stop the running of the task by clicking on the stop button, check the arrow in the screenshot. No need to run the task for a long time.

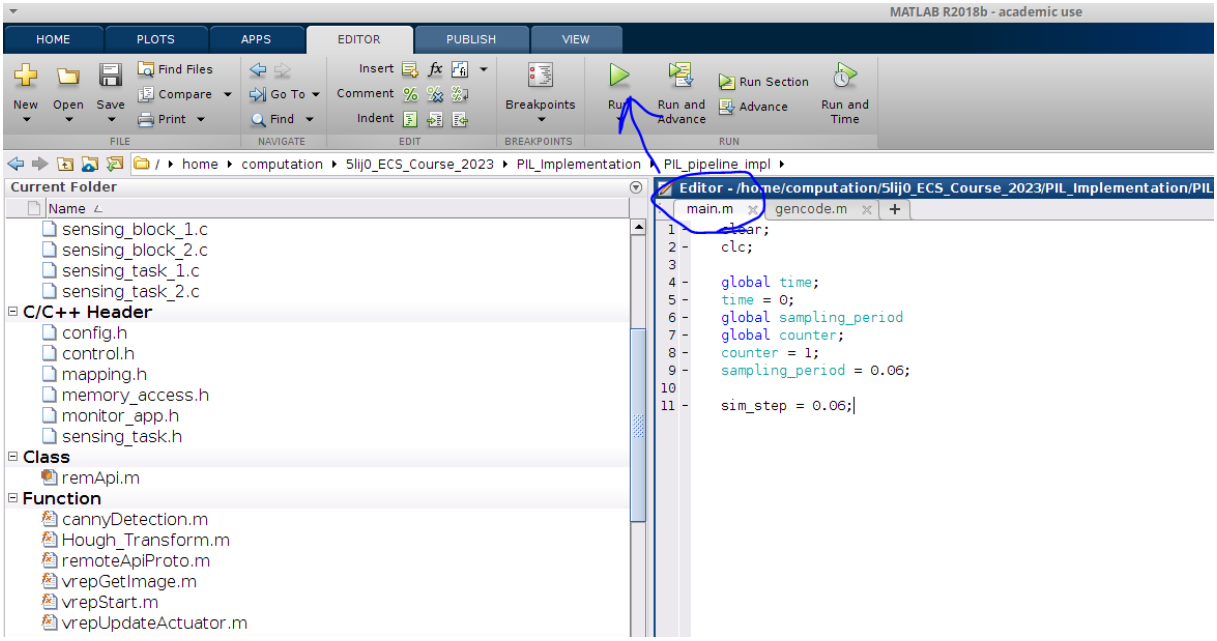


- after completing all the steps, run the following highlighted command in the same terminal window opened for the previous steps. This step is required for running the final PIL simulation, which will be run from **PIL_simulation_monitoring_app.slx** Simulink model.

    - Terminal 1 :
        - cd tutorial
        - cd monitoring-tools/
        - sudo ./channel_cheap_bridge 0 1 9878
    - Terminal 2 :
        - cd tutorial
        - ./readout.sh



- next step is to run the main.m file and check the CoppeliaSim software; the mass-damper scene should be opened and ensure it is not running automatically; if it is running, stop it.



- open the **PIL_simulation_monitoring_app.slx** Simulink model, and click on the Run button. This will start running the simulation. It will take some time to simulate as it initializes all the

required settings for running the simulation. So wait until you see the running status. Check the following screenshots for reference.

- check the current position returned from the monitoring app using the scope linked to the monitoring app PIL block. Also, visualize the output in CoppeliaSim.