vHILS INVESTIGATION REPORT

**Editing history**

|  |  |  |  |
| --- | --- | --- | --- |
| Sửa đổi số | Sự miêu tả | Người sáng tạo | ngày tháng |
| 1 | Nguyên | Duy Từ | 20 Tháng hai, 2020 |
| 2 | Tích hợp biện pháp đối phó với sự cố đồng bộ hóa | phúc giang | 11 Tháng ba, 2020 |
| 3 | Hỗ trợ U2A.  Thời gian hỗ trợ | phúc giang | 12 Tháng Sáu, 2020 |
| 4 | Thêm GUI vHILS và điều khiển quản lý giấy phép | phúc giang | 08 Tháng Bảy, 2020 |
| 5 | Cập nhật quy trình thực hiện quy trình sinh mã trong chương 3 | phúc giang | Ngày 15 tháng 9 năm 2020 |
| 6 | Thêm thông số kỹ thuật để tạo mã | phúc giang | 21 Tháng chín, 2020 |
| 7 | Hỗ trợ biên dịch mã nguồn bằng Renesas Compiler | phúc giang | 25 tháng 10 năm 2020 |
| số 8 | Hỗ trợ chuyển đổi mô hình tự động | phúc giang | 23 tháng 11 năm 2020 |
| 9 | - Yêu cầu hỗ trợ 05 Kiểm tra kết quả (để thực thi vHILS)  - Hỗ trợ thực thi vHILS mà không cần mở VLAB.  - Bổ sung thêm thông tin chi tiết về môi trường thiết lập cho vHILS (thiết lập VLAB) | phúc giang | 30 tháng mười hai 2020 |
| 10 | [Yêu cầu 05.1] kiểm tra kết quả - Graph Viewer  Chuẩn bị Trình xem biểu đồ cho dữ liệu thực thi | phúc giang | 25 Tháng Một, 2021 |
| 11 | [ Yêu cầu 01] Hỗ trợ khối ngoại vi bổ sung RLIN3n | Trí Nguyễn | 15 Tháng ba, 2021 |
| 12 | Hỗ trợ cài đặt tốc độ truyền cho RLIN3n | Trí Nguyễn | 23 Tháng Sáu, 2021 |
| 13 | Khắc phục sự cố thực thi nhỏ của Chế độ xem biểu đồ | phúc giang | Ngày 14 tháng 7 năm 2021 |

# lục mục

[lục mục](#_Toc56953549)  [1](#_Toc56953549)

[1. Tổng quan vHILS](#_Toc56953550)  [3](#_Toc56953550)

[2. vHILS Design (gói demo)](#_Toc56953551)  [3](#_Toc56953551)

[2.1. Tạo tệp](#_Toc56953552)  [4](#_Toc56953552)

[2.1.1. Engine thế hệ](#_Toc56953553)  [4](#_Toc56953553)

[2.1.2. Constructor create file](#_Toc56953554)  [5](#_Toc56953554)

[3. Khối chức năng S sinh mã nguồn](#_Toc56953555)  [7](#_Toc56953555)

[3.1.1. Tổng quan](#_Toc56953556)  [7](#_Toc56953556)

[3.1.2. QUẢNG CAO](#_Toc56953557)  [7](#_Toc56953557)

[3.1.3. CỔNG](#_Toc56953558)  [12](#_Toc56953558)

[3.2. Mã nguồn ngoại vi](#_Toc56953559)  [18](#_Toc56953559)

[3.3. Tính năng đo thời gian](#_Toc56953560)  [20](#_Toc56953560)

[3.4. vHILS GUI](#_Toc56953561)  [23](#_Toc56953561)

[3.5. giấy phép vHILS](#_Toc56953562)  [25](#_Toc56953562)

[4. Biên dịch mã nguồn đã tạo](#_Toc56953563)  [26](#_Toc56953563)

[4.1. Tổng quan về Makefile](#_Toc56953564)  [26](#_Toc56953564)

[4.2. Hỗ trợ đặc điểm kỹ thuật Renesas biên dịch.](#_Toc56953565)  [26](#_Toc56953565)

[4.2.1. Build and link lines](#_Toc56953566)  [26](#_Toc56953566)

[4.2.2. Cập nhật thời điểm khi hỗ trợ biên dịch Renesas](#_Toc56953567)  [27](#_Toc56953567)

[4.2.3. Kết quả sau khi thực hiện](#_Toc56953568)  [27](#_Toc56953568)

[5. Các bước chạy mô phỏng](#_Toc56953569)  [29](#_Toc56953569)

[6. Vấn đề:](#_Toc56953570)  [32](#_Toc56953570)

[6.1. Bước đầu không đồng bộ (Vẫn chưa giải quyết được)](#_Toc56953571)  [32](#_Toc56953571)

[6.2. Không đồng bộ hóa với mã thực thi thời gian dài (Đã giải quyết)](#_Toc56953572)  [32](#_Toc56953572)

[6.2.1. Về vấn đề](#_Toc56953573)  [33](#_Toc56953573)

[6.2.2. Biện pháp đối phó - Cài đặt để sử dụng OS TIMER và thời gian lấy mẫu hợp lý.](#_Toc56953574)  [34](#_Toc56953574)

[7. Các hạng mục công việc của giai đoạn sau (quý sau)](#_Toc56953575)  [38](#_Toc56953575)

# Tổng quan vHILS

- Mục tiêu nhúng được sử dụng để kiểm tra chức năng và vấn đề đo lường hiệu quả bằng cách chạy mã được xem xét trên MILS trên thiết bị. (Xác minh liên tiếp MILS-PILS)

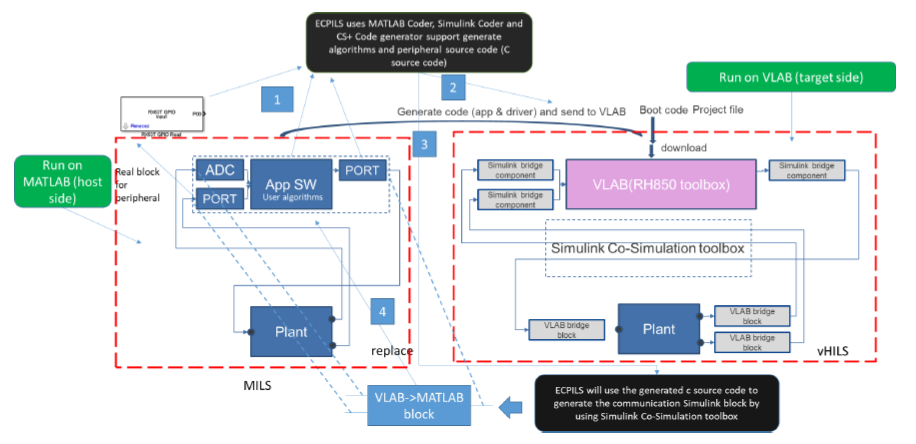
- Nhưng ở giai đoạn khảo sát thuật toán thì chưa có môi trường đo lường hiệu năng kể cả thiết bị ngoại vi

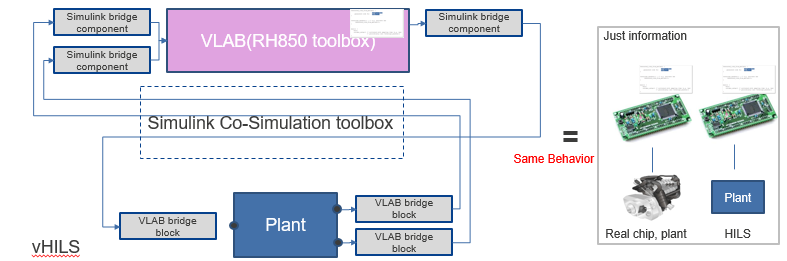
- Trong khi đó, nếu ước tính thời gian với thiết bị ngoại vi, cần kết nối mã điều khiển với mã được tạo từ ứng dụng và đó là nỗ lực cao đối với OEM không biết MCU.

- VHILS sử dụng hộp công cụ Co-simulation để giao tiếp giữa MATLAB Simulink và ASTC VLAB.

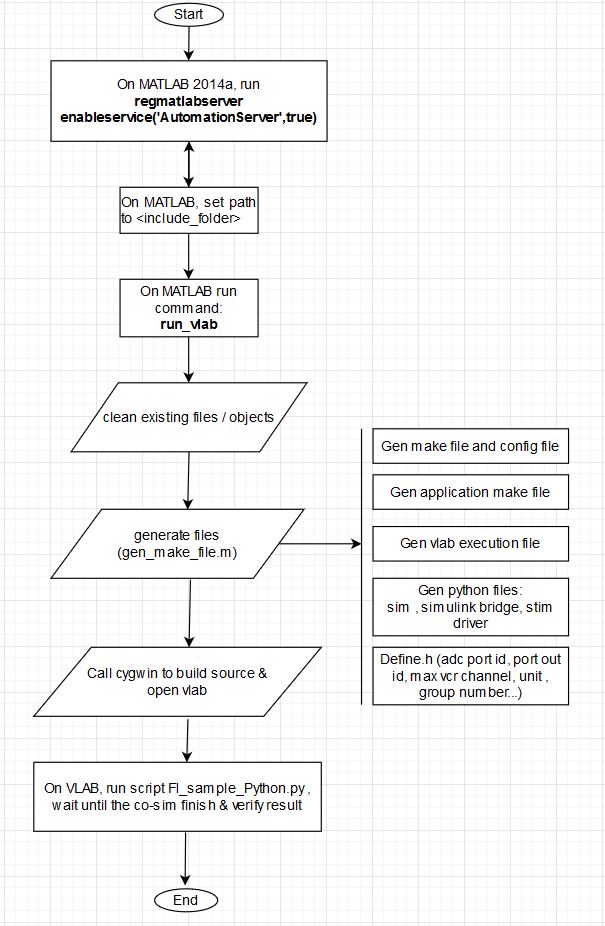
- Mục đích là để ước tính thời gian ở giai đoạn đầu bằng cách chuẩn bị mã bao gồm mã trình điều khiển ngoại vi xác minh chức năng và ước tính hiệu suất bằng cách thực hiện xác minh giáp lưng b/w MILS-vHILS.

Bằng cách hỗ trợ vHILS, Renesas có thể cung cấp giải pháp MBD mới cho khách hàng hơn là các nhà cung cấp khác. Khi công nghệ này được áp dụng cho vHILS, Renesas sẽ có thể hỗ trợ HILS trong tương lai. Hiện tại, có nhiều người dùng HILS, phương pháp này phải được khách hàng chấp nhận.





# vHILS Design (gói demo)



OSTM\_define.h: define the PCLK and the interval time

## Create file

### Động cơ thế hệ

Sử dụng phương pháp đọc/ghi MATLAB để tạo các tệp cần thiết. Trước khi tạo tệp mới, chức năng xóa tệp hiện đã được kích hoạt.

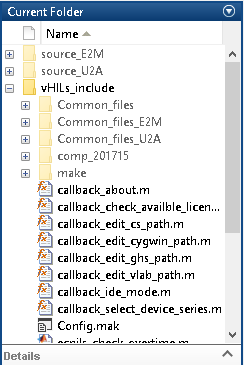
Ưu điểm của phương pháp này là chúng ta có thể chuyển đổi các biến trong MATLAB/Simulink sang các tệp được tạo (C, Python, Makefile, Config) một cách dễ dàng cũng như các tham số như <port\_id>, <sample\_time>, <quantum\_time>.

Dễ dàng sửa đổi nguồn tạo sau đó thực hiện lại.

Khi phát hành, chúng ta có thể dễ dàng ẩn mã nguồn được tạo bằng chức năng pcode của MATLAB.

### Constructor create file

#### vHILS Cấu hình mã nguồn



vHILS source code for:

* Create the vHILS GUI
* Execute generating and simulate the communication with vHILS.
* Draw the Graph Viewer for the execution time

Sample model for E2M

Sample model for U2A

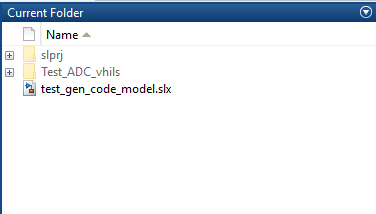
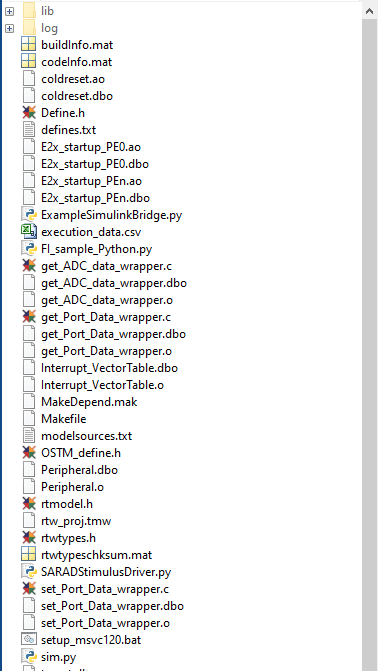
Generated common files

Common files for RH850/E2M and RH850/U2A

* Boot code
* Peripheral code
* Macro for define register address

#### Cấu trúc tệp cấu trúc được tạo

Tệp được tạo nằm trong thư mục <Khối tạo mã tiêu đề thư mục>\_vhils.



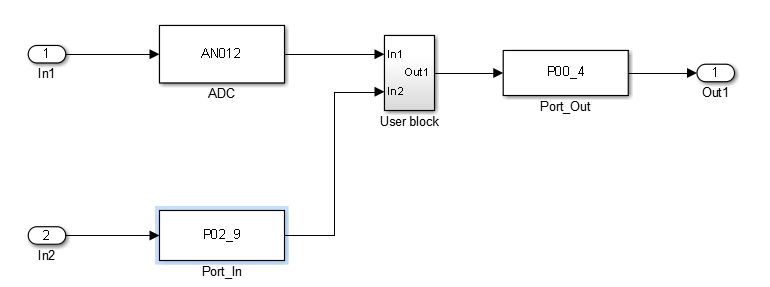
# Khối chức năng S để tạo mã nguồn

### tổng quan

Mô-đun ngoại vi được hỗ trợ:

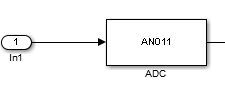
* ADC
* Cổng vào/Ra

Dưới đây là tổng quan về cách sử dụng các chức năng của khối.



### ADCH

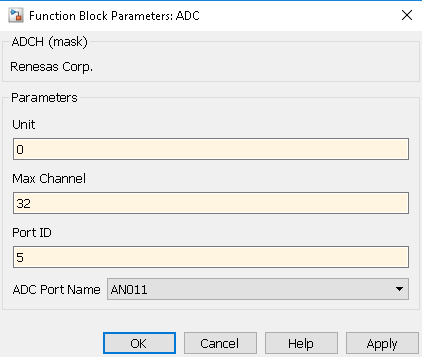
Để tạo mã nguồn cho cổng ADC, hãy sử dụng khối ADC trong vhils\_s\_function.slx:



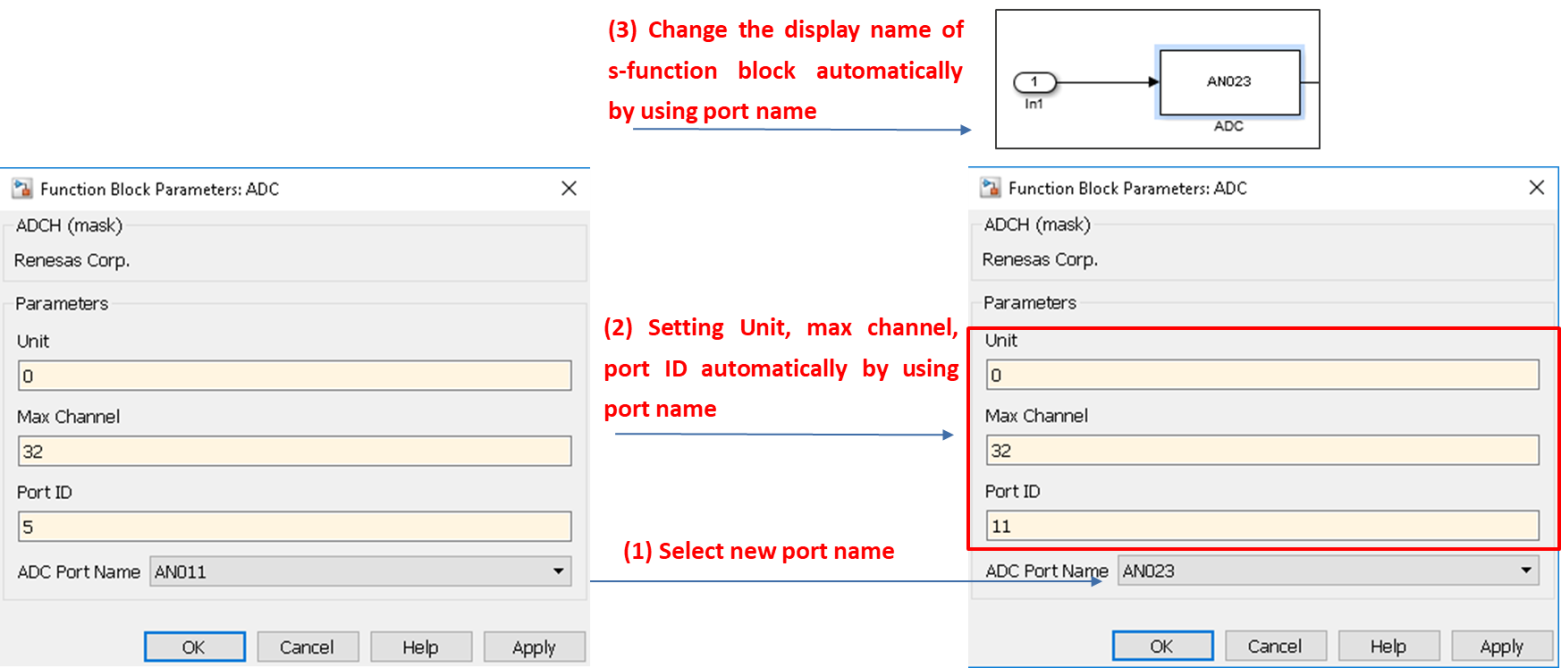
#### Feature:

* Lấy dữ liệu từ MATLAB từ cổng ADC trong model RH850.
* Trả lại cùng một giá trị từ MATLAB cho thuật toán người dùng.

Giao diện người dùng cho chức năng khối S.



#### Use the user interface:



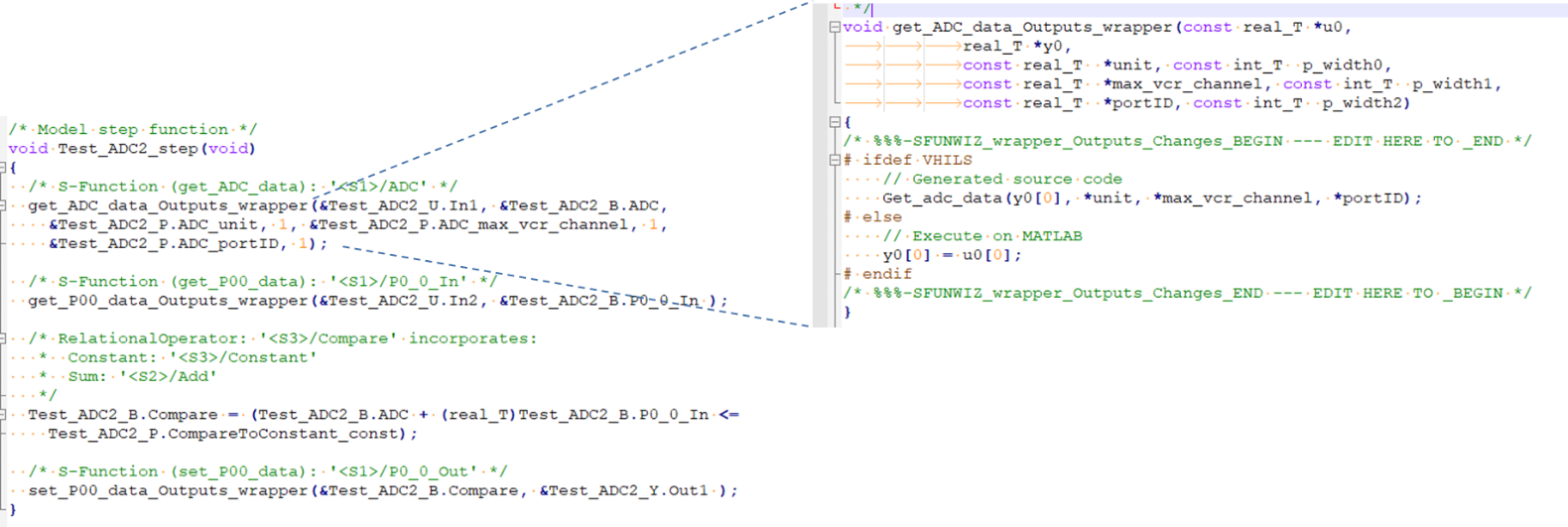
Giao diện người dùng được tạo bằng cách sử dụng Mặt nạ và chức năng gọi lại có tên callback\_my\_custom.m

Đặt tên cổng ADC, đơn vị, kênh tối đa và cổng ID sẽ tự động tạo.

Đơn vị, kênh tối đa và cổng ID được sử dụng làm tham số cho chức năng này.

Mục đích của nó chỉ là đích tên cổng ADC.

Khi xây dựng mô hình, các tham số này sẽ sinh ra đồng thời với dữ liệu đầu vào.

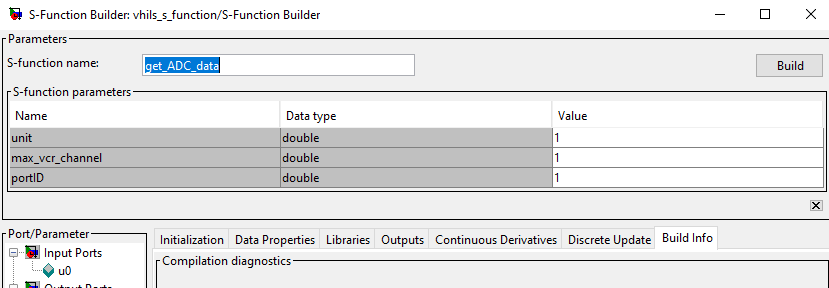


#### Thông tin chi tiết

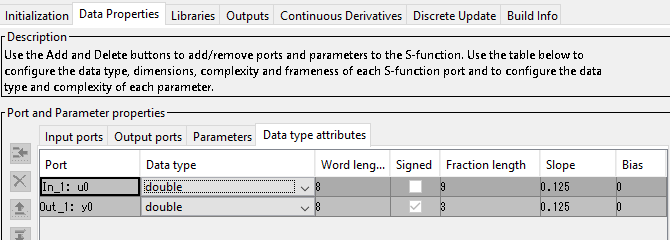
**Cách tạo khối này**

1. Tạo chức năng khối chức năng bằng cách sử dụng trình tạo chức năng.

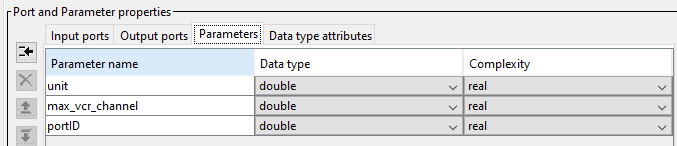
Create name function s



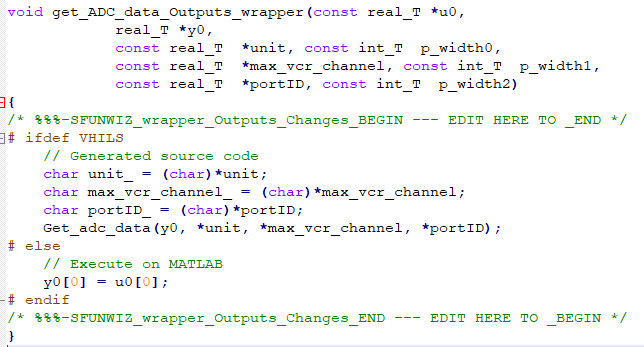
Set data type of outport and inport



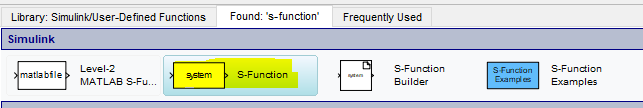
Đặt tham số cho ADC, trong trường hợp này là đơn vị, max\_vcr\_channel và cổng ID.

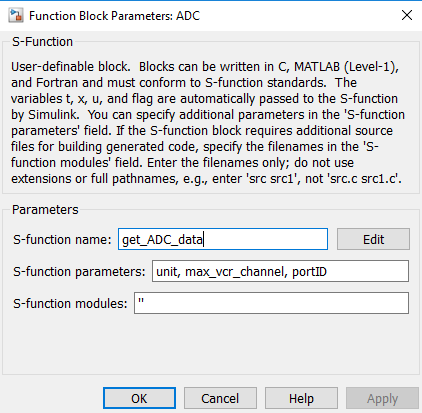


Mã nguồn cho chức năng sử dụng inport, outport, unit, max\_vcr\_channel, portID để chuyển sang chức năng ngoại vi.



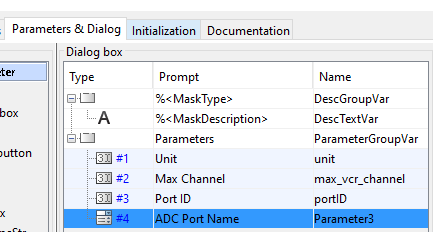
1. Liên kết s-function đã được tạo với khối s-function trong Thư viện Simulink.



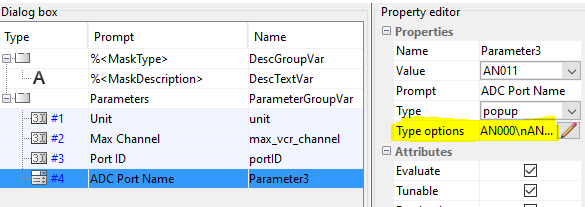


1. Thêm Mặt nạ vào khối chức năng bằng cách nhấp chuột phải -> Chọn Mặt nạ -> Tạo Mặt nạ.

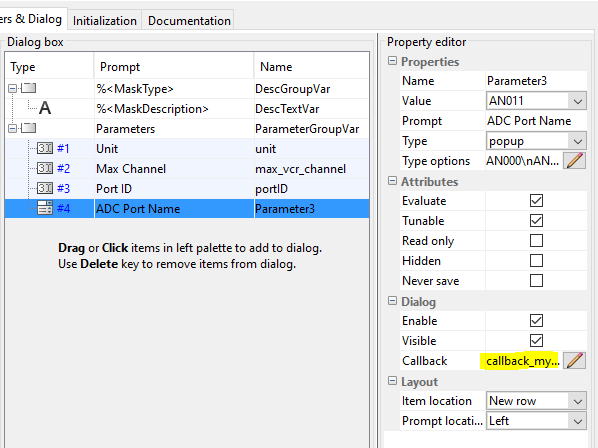
Vào tab Tham số & Hộp thoại để thêm tham số



Đối với tham số danh sách, hãy nhấp vào TODO và thêm danh sách tên cổng cho nó.

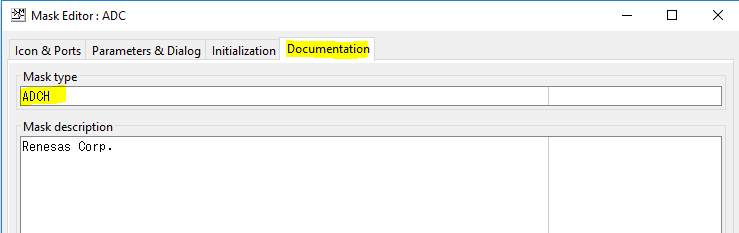


Tạo gọi lại cho cửa sổ bật lên



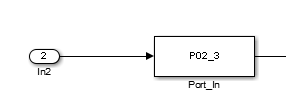
Tham khảo callback\_my\_custom.m

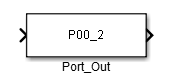
Move to description tab to add MaskType.



### CẢNH BÁO

Để tạo mã nguồn cho Cổng vào/ra, sử dụng khối Port\_In và Port\_Out trong vhils\_s\_function.slx:





#### Sử dụng chức năng block S

Đối chiếu với khối Port\_In:

* Lấy dữ liệu từ MATLAB và gửi đến Port trong VLAB.
* Gửi giá trị nhận được cho các thuật toán người dùng .

Đối chiếu với khối Port\_Out:

* Đặt giá trị từ thuật toán người dùng thành tên cổng đích.
* Gửi giá trị của cổng tới MATLAB (đầu ra của Đồng mô phỏng MATLAB).

#### User interface

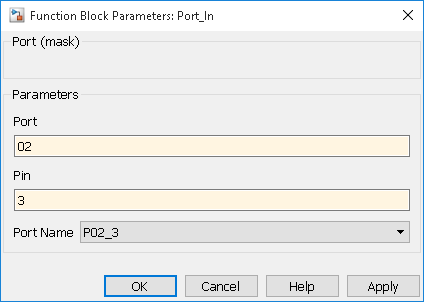
bên ngoài khoảng trống Set\_port\_value **( rút** ngắn **\*** u0 **,** chưa ký port ký **tự** , unchart mã pin **);**

bên ngoài khoảng trống Get\_port\_value **( rút** ngắn **\*** y0 **,** chưa ký port ký **tự** , unchart mã pin **);**

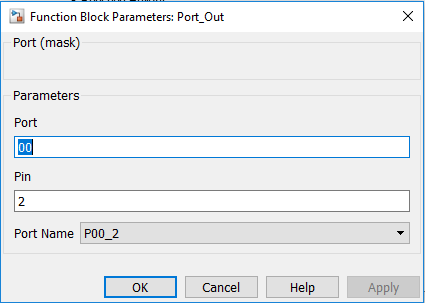
Tùy chỉnh chức năng C trong giao diện ngoại vi mà chúng ta cần tạo giao diện người dùng tương ứng.

Trong trường hợp này, giao diện người dùng phải có cổng và mã pin bắt đầu.

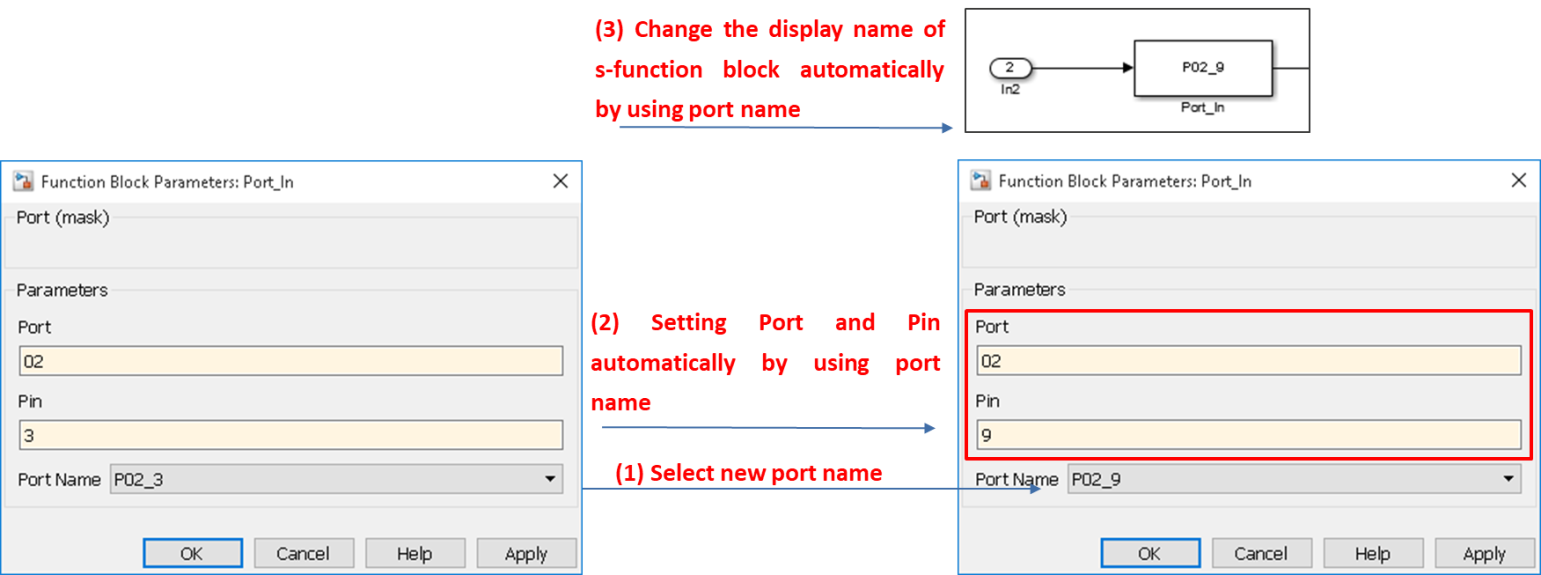
**Giao diện người dùng của Port\_In**



User interface of Port\_Out



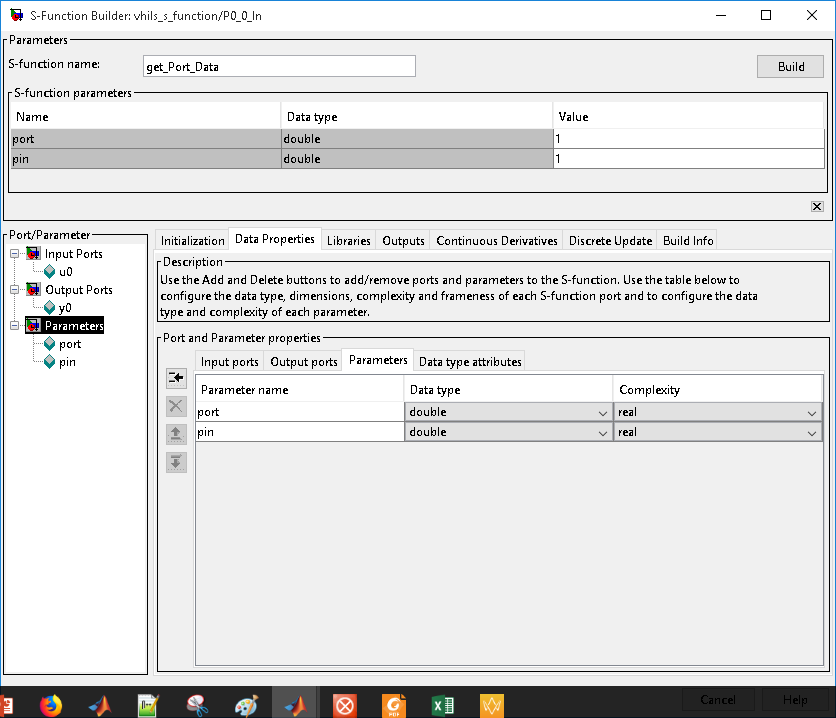
**Use the user interface**



#### Thông tin chi tiết

Cách tạo khối này:

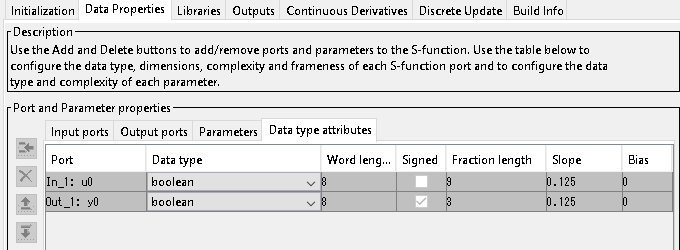
* 1. Sử dụng trình tạo hàm s để tạo khối hàm s.
* Create S chức năng tên



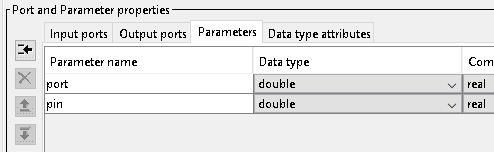
* Đặt kiểu dữ liệu cho cổng vào và cổng ra, trong trường hợp này boolean được sử dụng cho cổng.

**Nhận xét:**

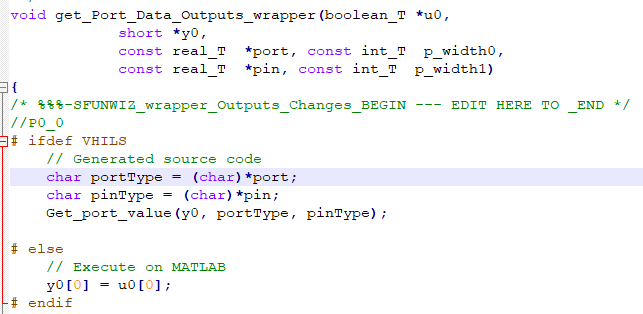
* Inport, outport dùng để gửi nhận dữ liệu giữa MATLAB và VLAB, thay đổi từng bước.
* Tham số được sử dụng để xác định mục tiêu cổng, giá trị chỉ thay đổi một lần khi được đặt trên giao diện người dùng.



* Tạo tham số cần thiết.



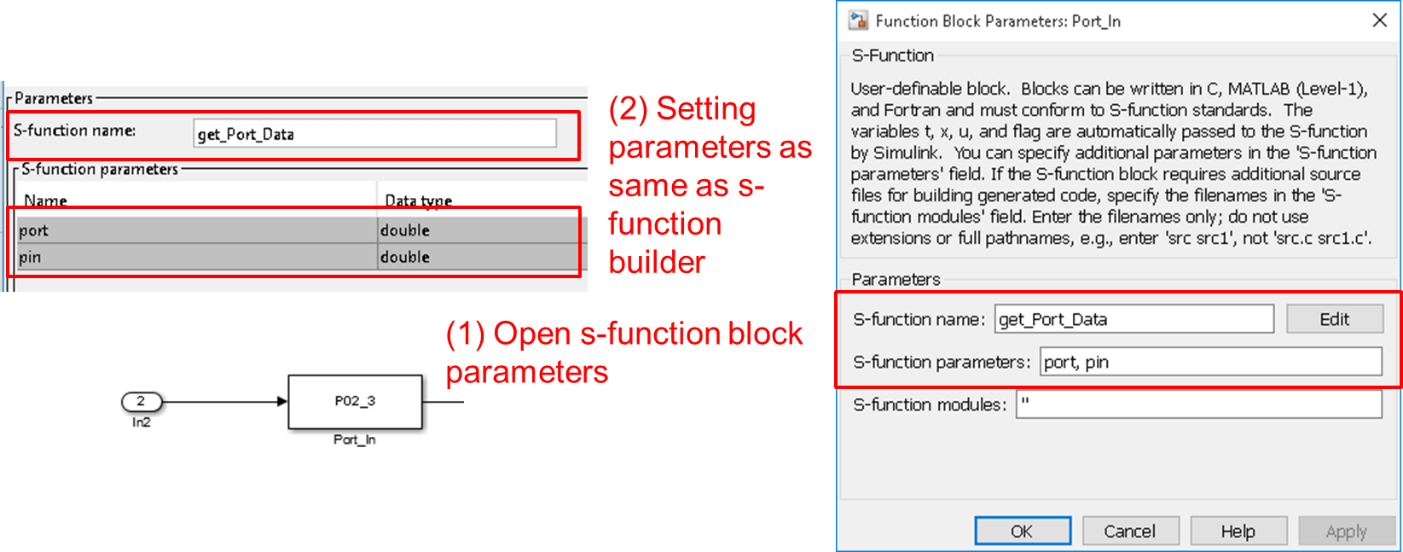
Mã nguồn đầu ra của chương trình tạo s-function



* 1. Sử dụng chức năng khối chức năng trong thư viện Simulink để tạo

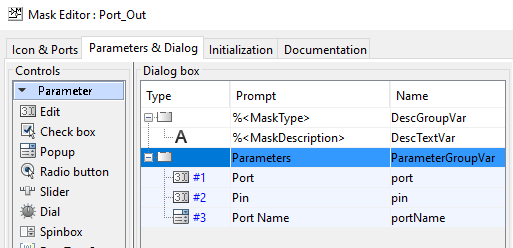
Vào thư viện Simulink copy vào model.

Chuột phải -> chọn cài đặt khối chức năng, thêm tên chức năng và cổng, ghim vào hộp thoại giống như bên dưới

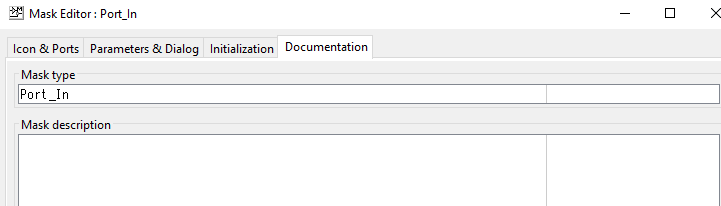


* 1. Add mask.

Thêm cổng và ghim thuộc tính mặt nạ, tên cổng được bật lên.



Add mask mask



Nhận xét:

Loại mặt nạ được sử dụng để tạo tệp python trong việc tạo khối Đồng mô phỏng MATLAB.

* 1. Tạo cuộc gọi lại cho tính năng nâng cao.
* Cập nhật tên khối
* Tự động cập nhật thông số cổng và chân.

### RLIN3n

#### Sử dụng chức năng block S

Đối chiếu với khối RLIN3n\_Transmission:

* Trong VLAB: lấy dữ liệu kép từ thuật toán Người dùng, truyền liên tục 8bit qua cổng RLIN3n, sau đó ghép kết quả sau khi nhận đủ 64bit. And output ra MATLAB (giá trị kép).
* Module hỗ trợ RLIN3n (n = 0 – 23) của RH850 U2A

#### User interface

Graphical user interface, text, application

Description automatically generated

#### Thông tin chi tiết

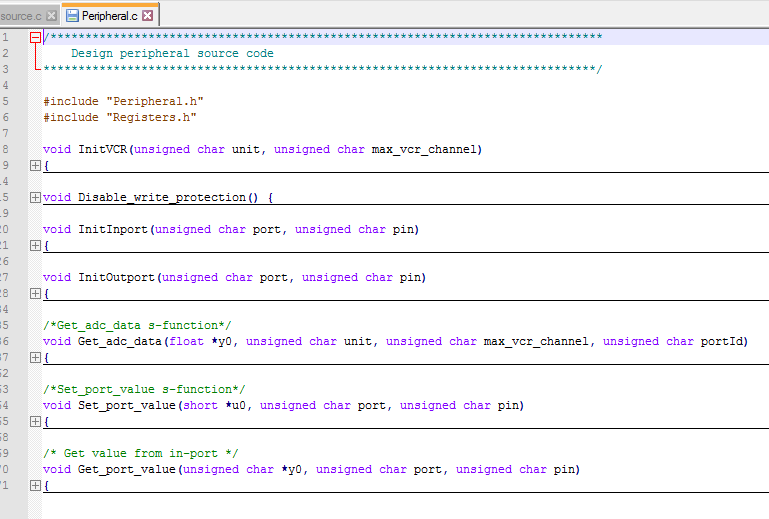
Bước tạo chức năng này giống như 3.1.3. PORT   
Mã nguồn của khối này:  
Graphical user interface, text, application, email

Description automatically generated

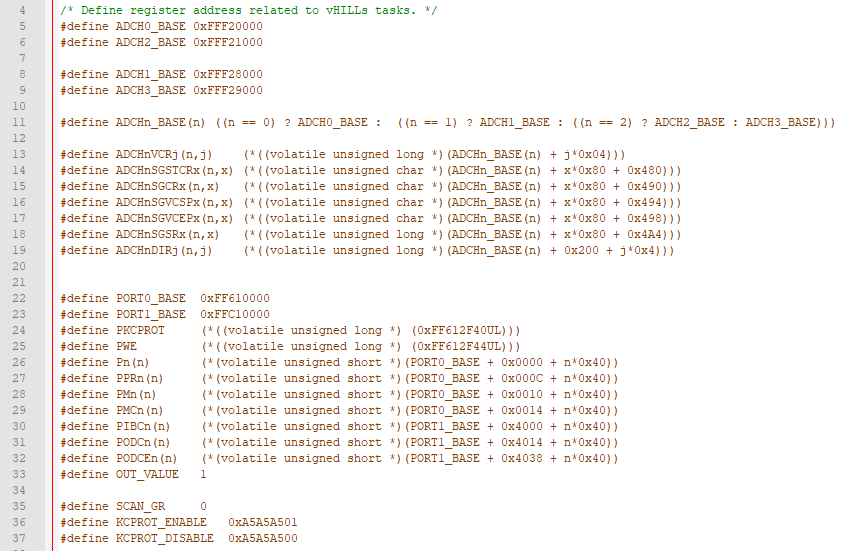
## ngoại vi mã nguồn

Đây là mã được bao gồm để nhận giá trị được chuyển đổi ADC từ thanh ghi, nhận giá trị từ cổng vào và ghi giá trị ra cổng.

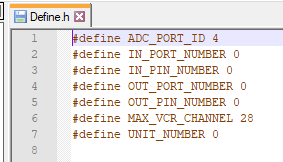
Ở phiên bản nháp hiện tại, mã đã được chuẩn bị sẵn, trong giai đoạn sau, chúng tôi sẽ sử dụng công cụ xây dựng hàm S của Simulink để tạo mã tự động.



Địa chỉ đăng ký đã được xác định trong Register.h

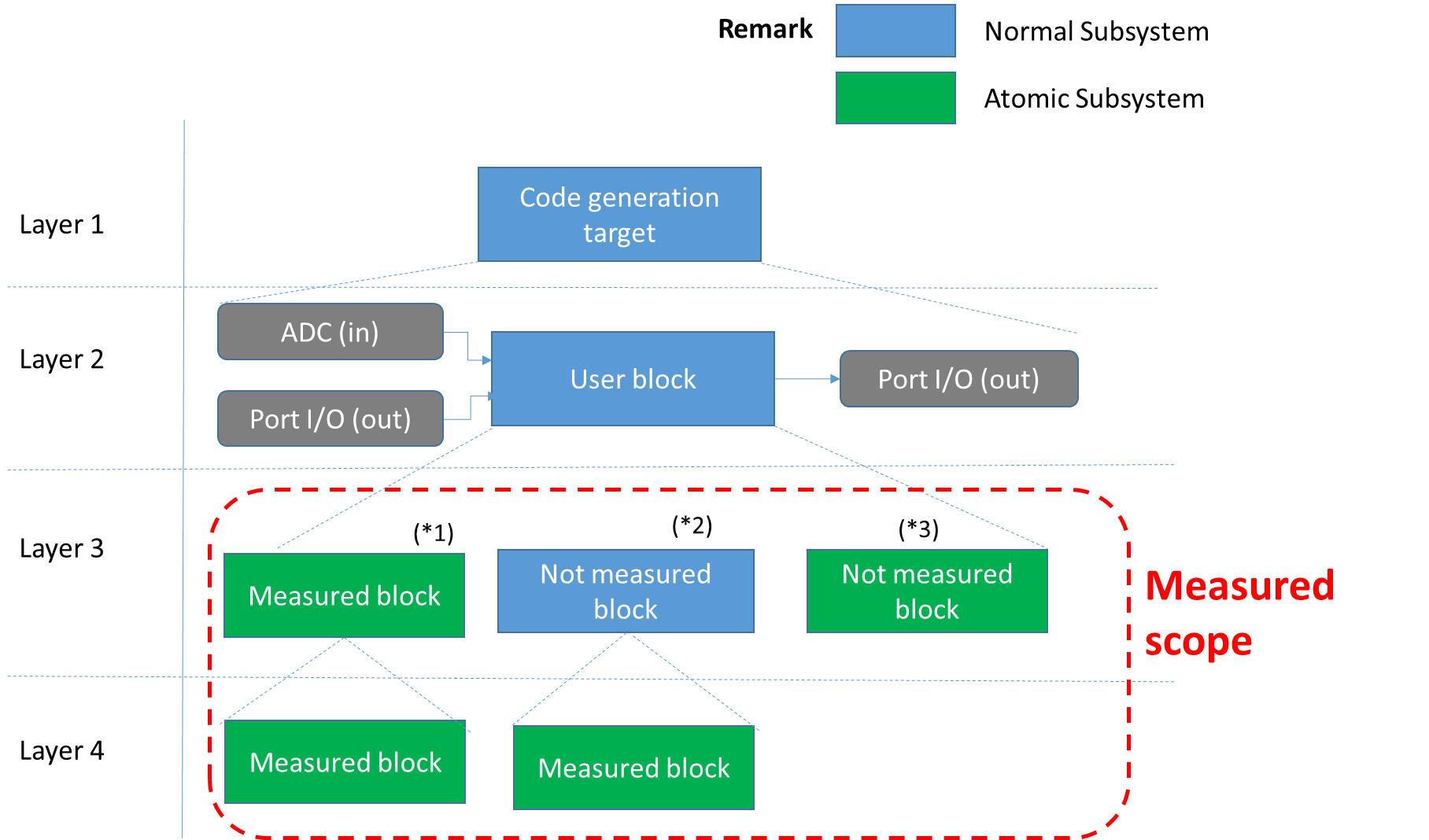


Các biến được sử dụng trong mã nguồn C (các giá trị được tạo dựa trên cài đặt của người dùng)



## Tính năng đo lường thời gian

### Cấu trúc của Mô hình Simulink để đo lường



**Nhận xét:**

(\*1): Khối được đo đáp ứng mọi yêu cầu về phép đo.

(\*2): Quyết định của người dùng: Không đo được khối này. Vì vậy, họ không cần phải chuyển đổi nó thành Hệ thống con nguyên tử. Tham khảo mục 3.3.4.

(\*3): Tuy là Atomic Subsystem nhưng user không định nghĩa trong file input\_subsystem.txt.

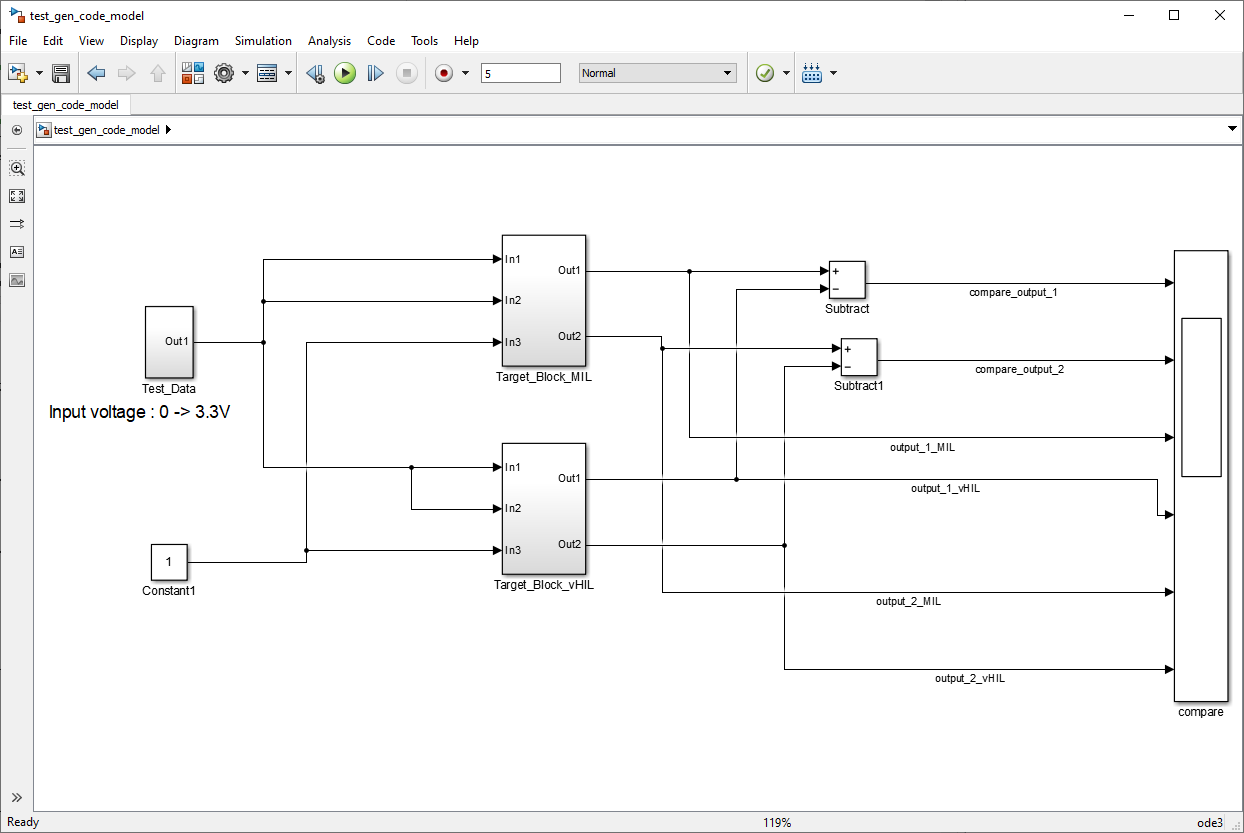
Dưới đây là yêu cầu đối với khối được đo:

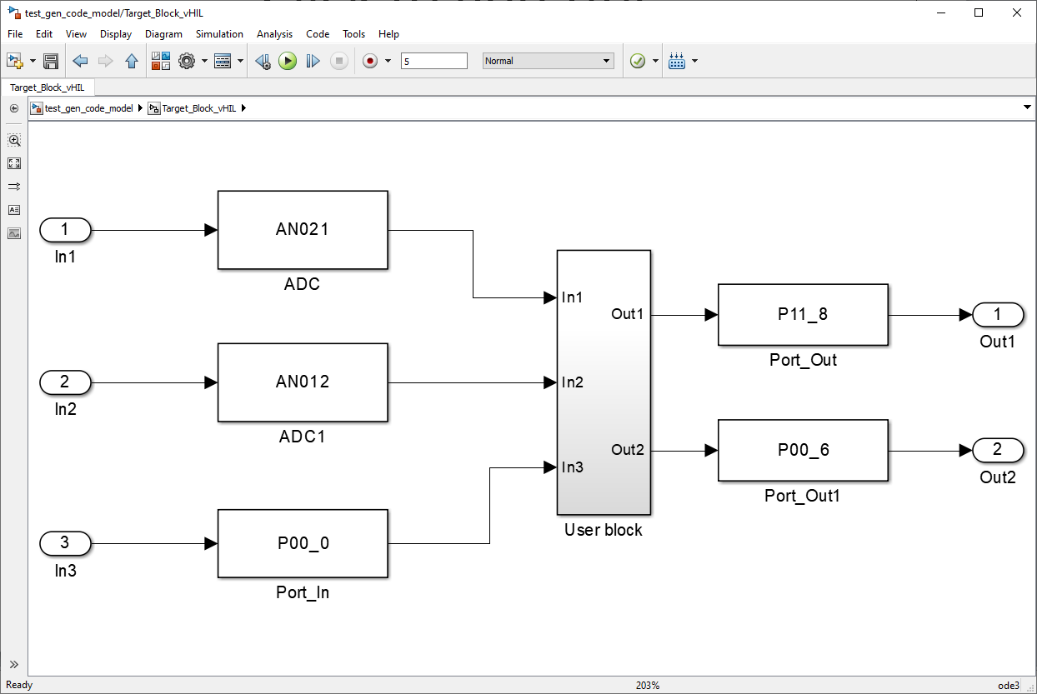
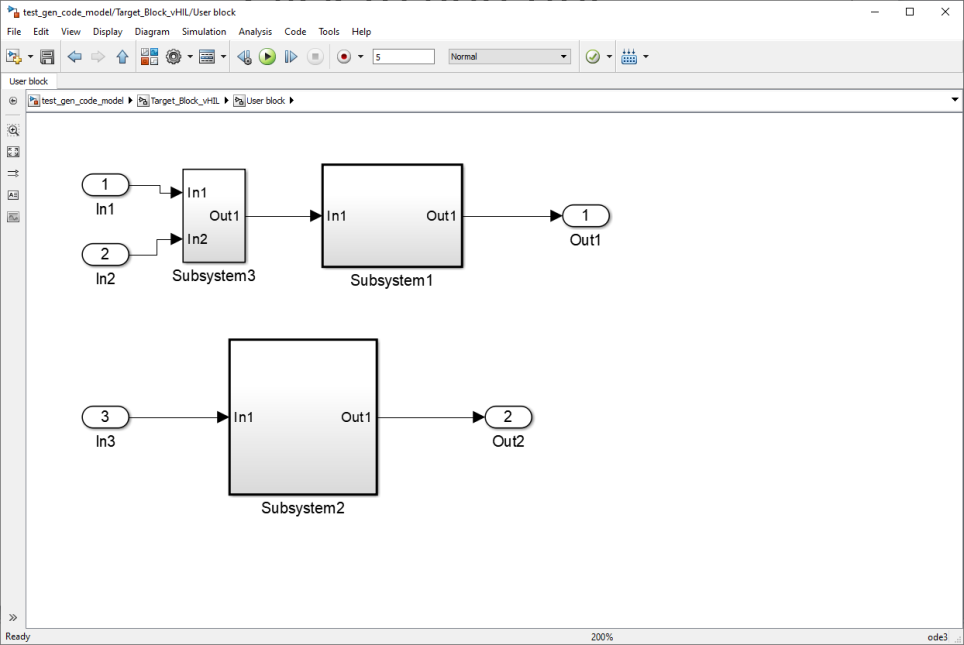
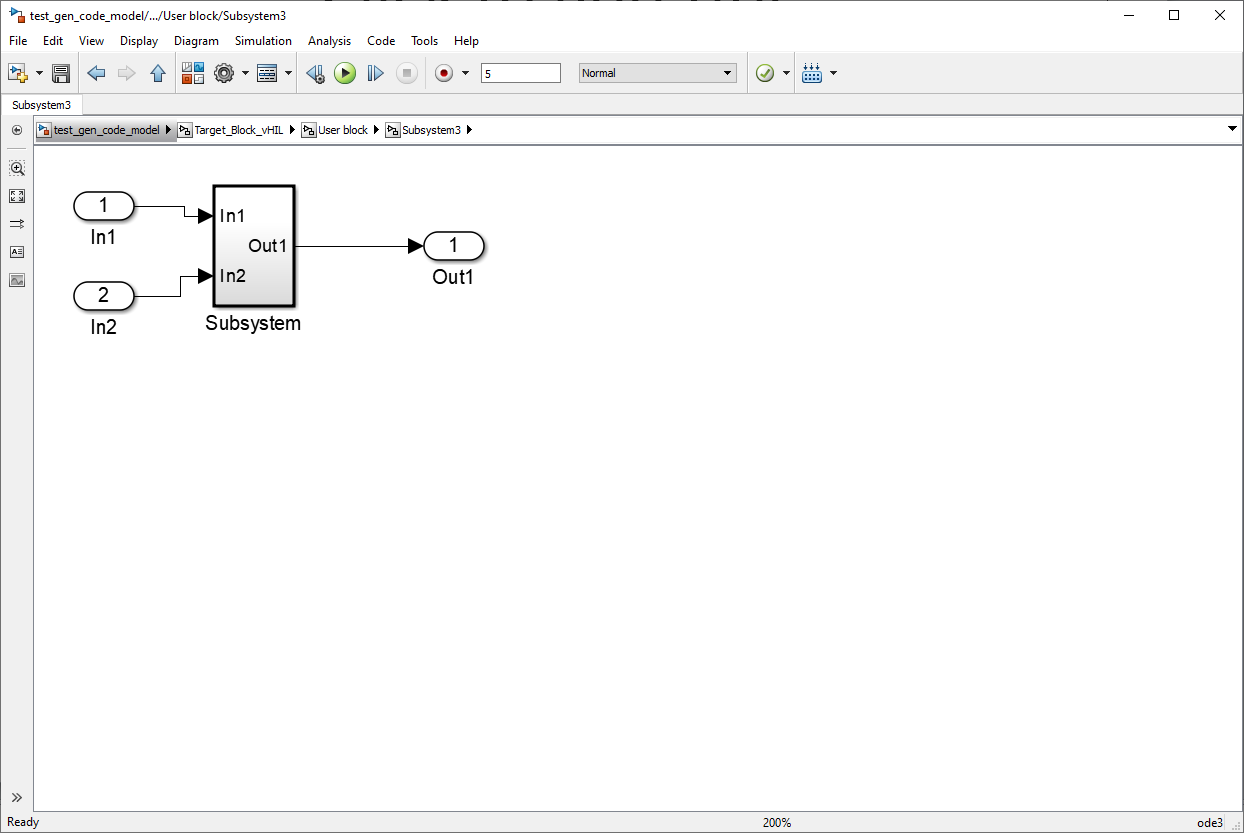
1. Chỉ đo các khối bên trong Khối người dùng.

2. Chỉ đo cho khối được xác định trong input\_subsystem.txt. Tham khảo phần 3.3.3.

3. Nó phải là Hệ thống con nguyên tử.

### Simulink phong phú để đo lường





**Lưu ý:** Phải đặt khối nguyên tử cho khối Người dùng.

### File đầu vào để đo lường

Trong mục mô hình thư mục, để xác định bất kỳ khối nào được đo lường, hãy chuẩn bị input\_subsystem.txt với định dạng bên dưới:

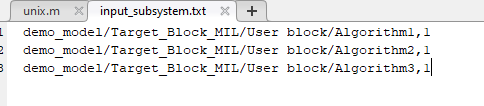
<Đường dẫn Atomic\_Subsystem\_name\_1>[Enter]

<Đường dẫn của Atomic\_Subsystem\_name\_2>[Enter]

…

<Đường dẫn của Atomic\_Subsystem\_name\_N>[Enter]

**Ví dụ:**



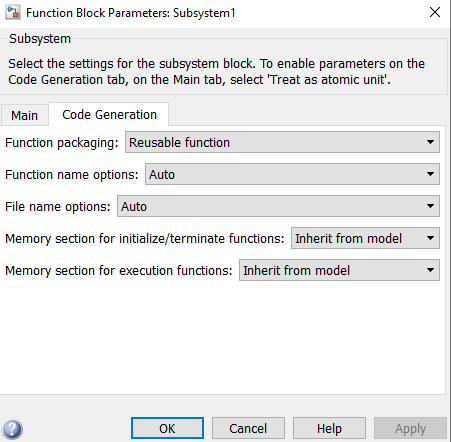
### Cách thay đổi hệ thống con bình thường thành hệ thống con nguyên tử

Để thay đổi Hệ thống con bình thường thành Hệ thống con nguyên tử, hãy làm theo các bước sau:

(1) Click chuột phải vào Hệ thống thông thường -> Chọn Thuộc tính

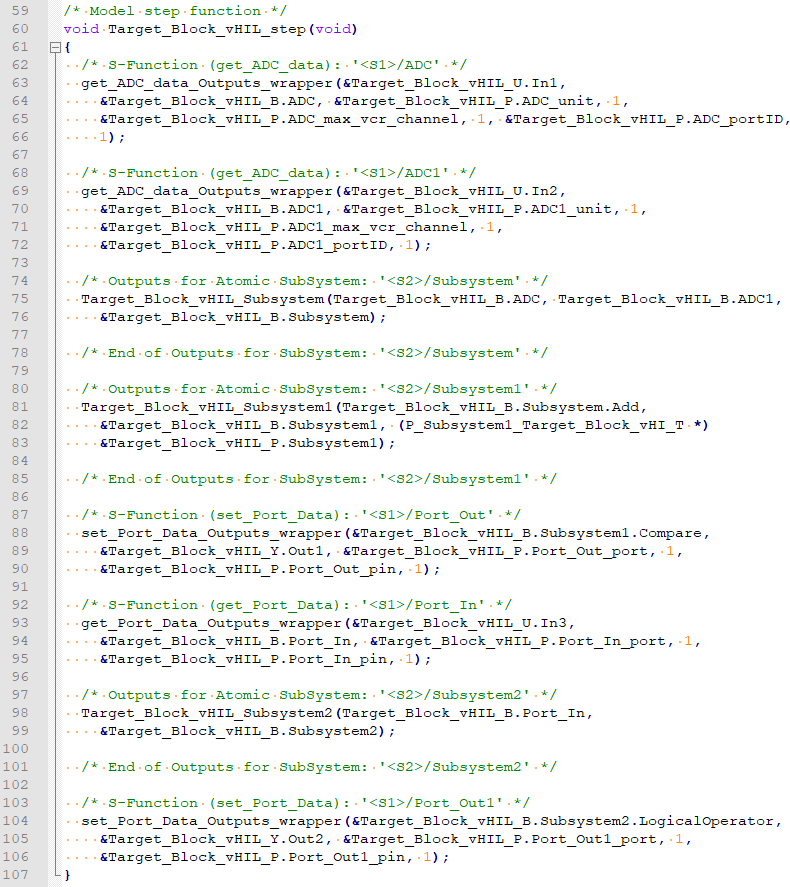
(2) Trong tab Chính, chọn cài đặt [coi as đơn vị nguyên tử].

(3) Trong tab Tạo mã, đặt [Đóng gói chức năng] thành “Chức năng tái sử dụng”.



### MATLAB đã tạo mã nguồn cho mô hình Simulink

MATLAB tạo mã nguồn cho Hệ thống con trong bước chức năng. Mỗi hệ thống con nguyên tử có một chức năng nhãn và chứa các thuật toán của nó.

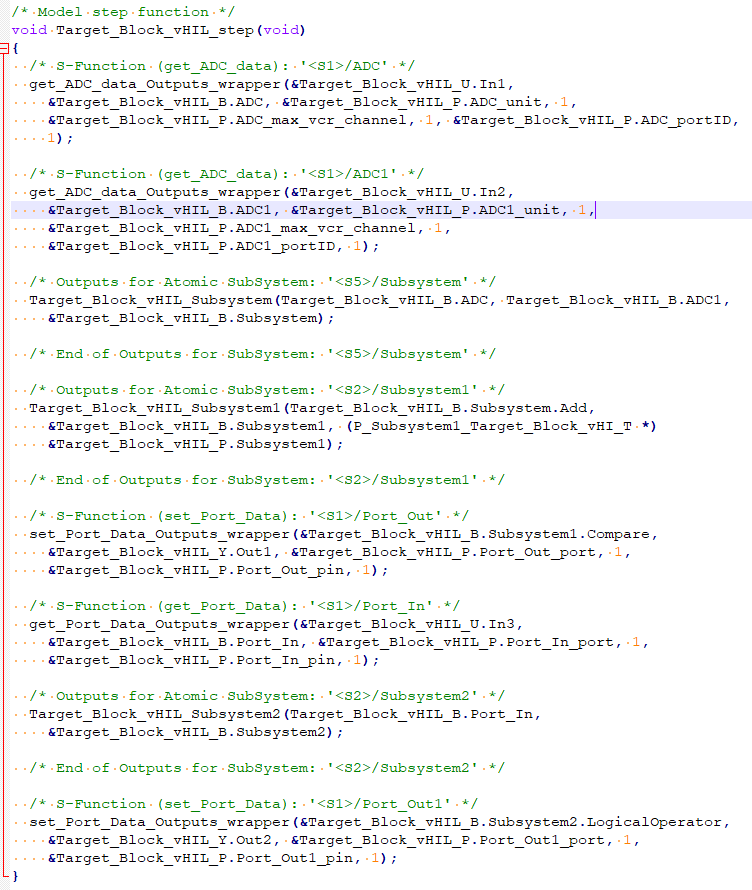


For “subsystem” block

For subsystem1 block

For subsystem2 block

Để lấy tên chức năng, chúng tôi sử dụng nhận xét liên quan đến Hệ thống con đích.



Get the function label

Using the string:

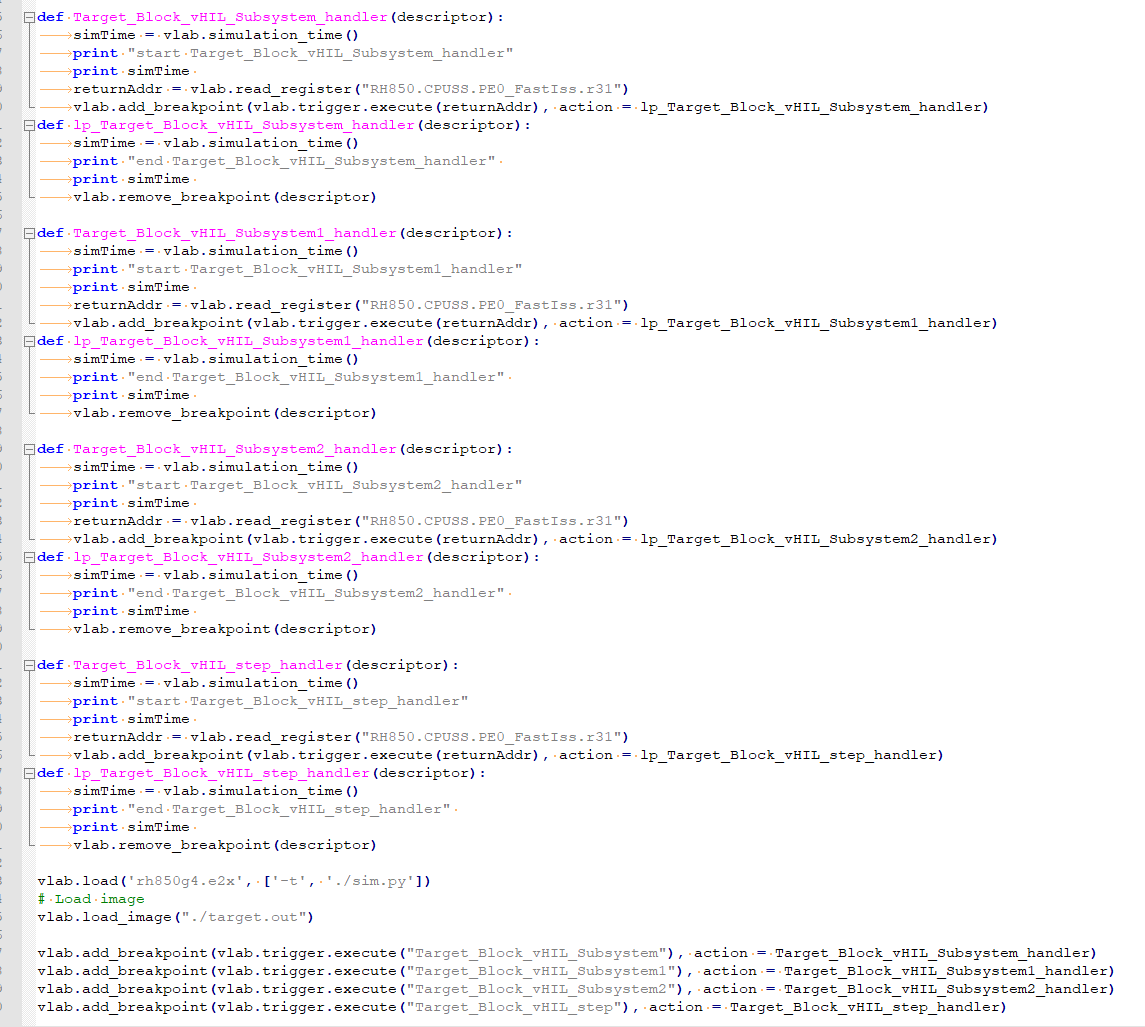
“Outputs for Atomic Subsystem”

And the name of the Subsystem in the input\_subsystem.txt to detect the location

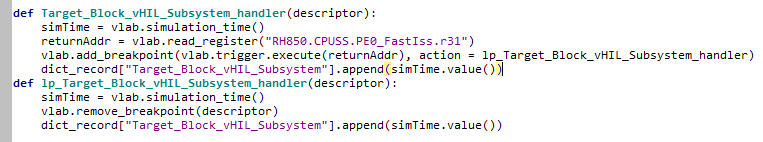
### Mã nguồn đo lường liên quan đến chức năng VLAB

#### Add Point Stop and Recording Time

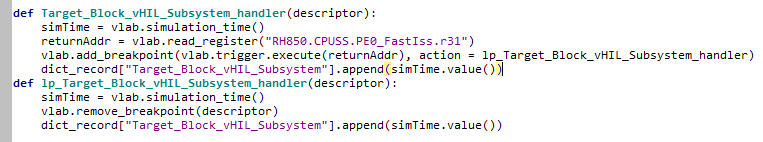
Thêm điểm dừng tại chức năng nhãn và thêm trình xử lý cho nó.



Trong chức năng xử lý nhãn chức năng, ghi lại thời gian thực hiện tại thời điểm bắt đầu. Sau đó, thêm điểm ngắt cho địa chỉ trả về (giá trị của thanh ghi LP) và trình xử lý của nó.



Trong quá trình xử lý địa chỉ trả về, hãy ghi lại thời gian thực hiện tại thời điểm kết thúc.



#### Constructor data of the current current Lưu trữ

Từ điển thực hiện:

“System con1”: [bắt đầu kết thúc bắt đầu kết thúc bắt đầu ….]

“System con2”: [bắt đầu kết thúc bắt đầu kết thúc bắt đầu ….]

“System con3”: [bắt đầu kết thúc bắt đầu kết thúc bắt đầu ….]

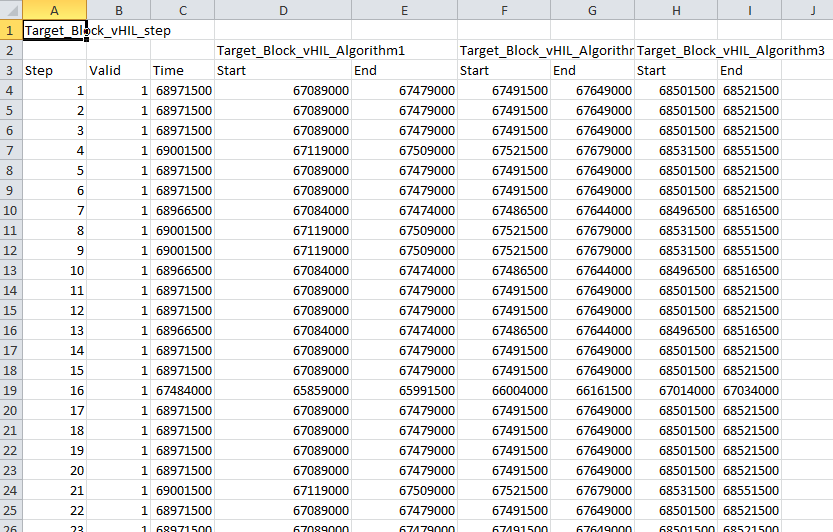
Bằng cách sử dụng phương pháp này, có thể lấy/đặt thời gian bắt đầu và kết thúc bằng cách sử dụng chức năng nhãn.

### View map for vHILS

#### Đầu vào:

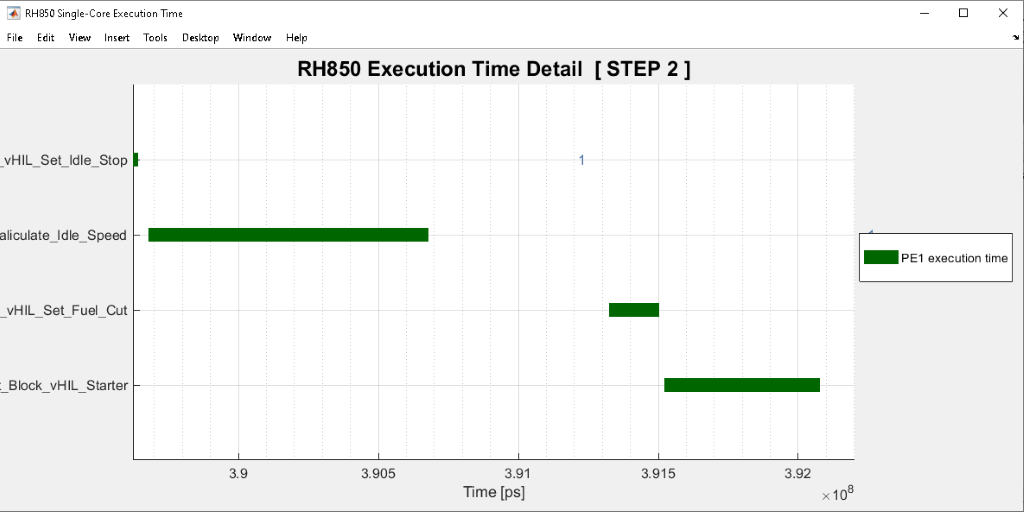
Đầu vào 1: Tệp data.csv thực thi trong 3.3.6.2

* Unit of it is pico seconds
* Nó chứa thời gian thực hiện của từng bước và bắt đầu kết thúc của mỗi hệ thống con
* Giá trị của start end là giá trị tương đối.



Đầu vào 2: tệp input\_subsystem.txt tham khảo 3.3.3

#### Heading shape:



**Ghi chú:** Hiện tại trong Graph Viewer chưa có thời gian thực thi của mã nguồn bên ngoài vi

#### Lý do cập nhật Phiên bản MATLAB lên MATLAB R2017b

Do thiếu chức năng trong MATLAB R2014a. Đã xảy ra lỗi khi sử dụng Graph Viewer. Bằng cách sử dụng MATLAB R2017b, chúng ta có thể sử dụng lại các tính năng tốt của Graph Viewer hiện tại của ECPILS. Ngoài ra, phiên bản MATLAB mới tiện lợi hơn cho người dùng.

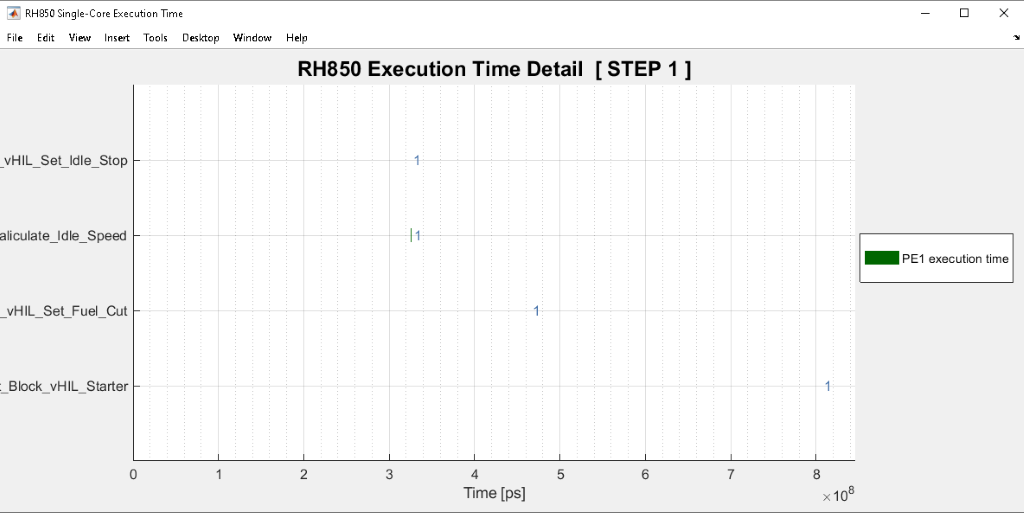
#### ~~Đã cập nhật luồng MATLAB R2017b để giải quyết sự cố liên quan đến sự cố MATLAB khi thực thi vHILS~~

* ~~Không thể biến quy trình MATLAB hiện tại thành máy chủ Tự động hóa bằng cách sử dụng lệnh enable\_service(“AutomationServer”, false). Nó sẽ mở một Máy chủ tự động hóa mới để ngăn chặn xung đột.~~
* ~~Trong vHILS\_testbench\_sample.py, hãy đợi cho đến khi mô hình thực thi đóng rồi đóng VLAB. Lý do: nếu đóng VLAB thì mô hình thực thi cũng đóng. Sau đó, người dùng không thể nhìn thấy kết quả.~~

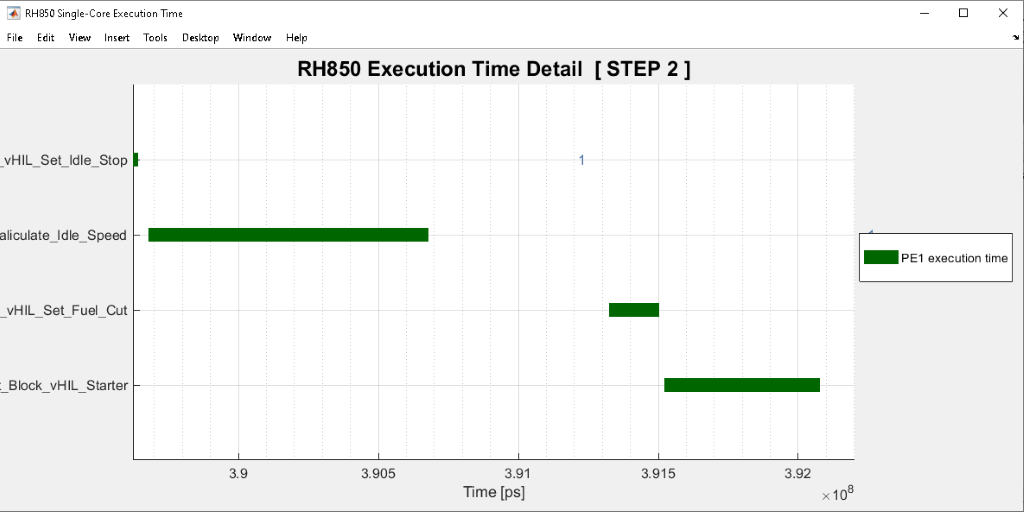
#### Đã cập nhật chế độ xem biểu đồ để giải quyết sự cố thực thi nhỏ

Để giải quyết vấn đề này, chúng tôi đã giải phóng thời gian thực thi của hệ thống con trong thuật toán người dùng.

**Nguyên:**



**Cố định hình ảnh:**



Để khai thác nó, chúng ta tiến hành 2 bước:

**Bước 1:** Khắc phục sự cố hệ thống phụ trong bản gốc không được thực thi liên tục. Dưới đây là ví dụ:

con1\_function system ()

Hệ thống con2\_function()

Con3\_function() system

**RLIN3\_Send\_function() -> Chờ lâu.**

Con4\_function() system

Sau đó, mặc dù tôi loại bỏ phần thừa giữa đầu (0ps -> thời gian bắt đầu của khối đầu tiên) và đuôi. Còn dư thừa (RLIN3 Send\_function ở giữa)

=> Tôi đã giải quyết bằng cách thực hiện tất cả các thuật toán người dùng trước, sau đó tất cả chức năng Gửi RLIN3 sẽ thực hiện. Như sau:

con1\_function system ()

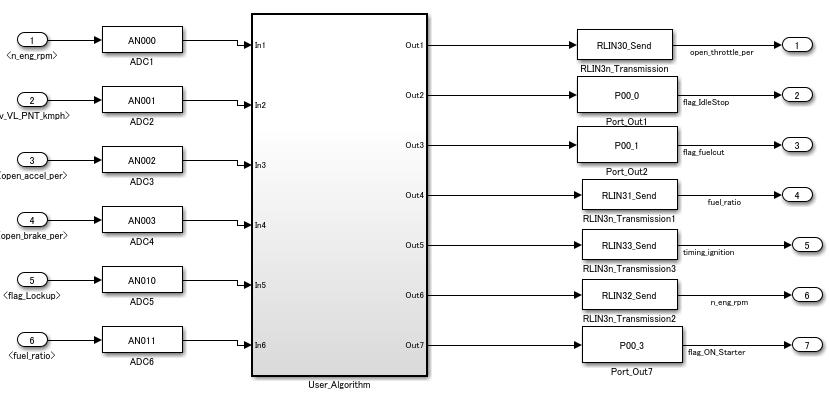
Hệ thống con2\_function()

Con3\_function() system

Con4\_function() system

**RLIN3\_Send\_function() -> Chờ lâu.**

Cách thực hiện: set khối User\_Algorithms (connected with the external block) thành khối nguyên tử.



**Bước 2:** Sau khi chọn chức năng hệ thống con chạy liên tục, chúng ta loại bỏ phần đầu và phần đuôi như bạn thấy trong cấu hình cố định.

### Giới hạn đối với phương pháp đo lường hiện tại

**1. Tên của các khối được đo phải là duy nhất. (Sẽ tìm giải pháp trong tương lai)**

Lý do: Phương pháp hiện tại để lấy hàm được tạo cho khối được đo đang sử dụng thông tin trong nhận xét. Bình luận của nó chỉ có tên Hệ thống con. Sau đó, không thể xác định khối đã được đo mục tiêu bằng đường dẫn đầy đủ của Hệ thống con.

## vHILS GUI

cái bàn‑ Mô tả món hàng

|  |  |
| --- | --- |
| Tên mục | Sự miêu tả |
| Cygwin Path | Can run file make |
| VLAB path | VLAB path |
| IDE mode | Tạo mới hoặc Mở dự án hiện có. Hiện tại, chỉ có việc tạo dự án được thực hiện. |
| Build tool | Chọn Trình biên dịch Renesas hoặc Trình biên dịch GHS |
| path CS+ | Sử dụng để tìm biên dịch Renesas.  Nếu trống:  Sử dụng biên dịch GHS biên dịch  If have con đường  Sử dụng Renesas biên dịch biên dịch. |
| path GHS | path GHS |
| Device Line | RH850/E2M hoặc RH850/U2A hoặc …. |
| PCLK OSTM |  |
| Kiểm tra giấy phép có sẵn | kiểm tra giấy phép |
| Trong khoảng | tool information, version… |

cái bàn‑ Chức năng gọi lại của mục

|  |  |
| --- | --- |
| Tên mục | Recall function |
| Cygwin Path | call back\_edit\_cygwin\_path.mm |
| VLAB path | call back\_edit\_vlab\_path.m |
| IDE mode | callback\_ide\_mode.m |
| Build tool | call back\_build\_tool.m |
| path CS+ | call back\_edit\_cs\_path.m |
| path GHS | call back\_edit\_ghs\_path.m |
| Device Line | callback\_select\_device\_series.m |
| PCLK OSTM |  |
| Kiểm tra giấy phép có sẵn | callback\_check\_availble\_license.m |
| Trong khoảng | callback\_about.m |

## giấy phép vHILS

Hồ sơ cụ thể để kiểm tra giấy phép

* callback\_lic.m
* get\_license\_definition.m
* vhils\_check\_avai\_features.m
* vhils\_check\_license.m
* tlc\_switch\_state.m

Table Definition ‑of License

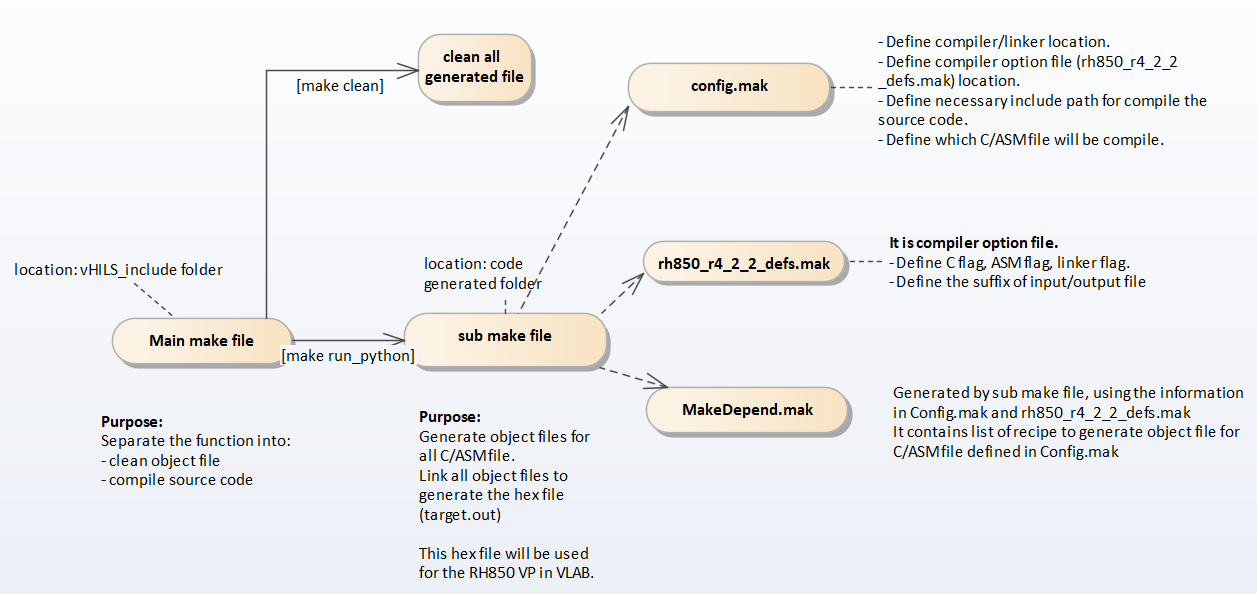
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tên giấy phép | thiết bị được hỗ trợ | | Tính năng đo lường thời gian | Công cụ xây dựng được hỗ trợ | |
| RH850/E2M | RH850/U2A | GHS biên dịch | Renesas biên dịch |
| Mục tiêu nhúng cho vHILS | √ | √ | √ | √ | √ |

**Giấy phép sẽ được kiểm tra:**

* Kiểm tra giấy phép khi mở mô hình và thay đổi thành vhils.tlc. Nếu không có giấy phép, hãy thay đổi tlc thành ert.tlc và tắt tất cả các mục nhập.
* Kiểm tra giấy phép cho GUI khi chọn Dòng thiết bị tham khảo Bảng 2-3.
* Kiểm tra giấy phép khi sử dụng lệnh MATLAB run\_vlab.m.
* Kiểm tra giấy phép cho khối S-Function. Lý do: người dùng có thể sử dụng chức năng khối S để tạo mã nguồn ngoại vi theo cách thủ công. Do đó, cần phải kiểm tra giấy phép cho khối S-Function.

# Biên dịch mã nguồn được tạo

## Tổng quan về Makefile

Sơ đồ bên dưới là hệ thống tổng quan Make file (áp dụng cho cả GHS và Renesas Compiler)

## Hỗ trợ đặc điểm kỹ thuật Renesas biên dịch.

### Build and link line command

Tham khảo tệp C:\Program Files (x86)\Renesas Electronics\CS+\CC\CC-RH\Help\Compiler-CCRH.chm. Đây là những dòng lệnh cần thiết để sử dụng Renesas biên dịch biên dịch.

To compile mã nguồn c và mã nguồn gắn kết thành tệp đối tượng

**Đối xứng với mã nguồn cài đặt nhanh:**

ccrh.exe <filename>.asm … <filename\_N>.asm –Xcpu=g4mh –g –c –Xmsg\_lang=English -I<include dir> -<macro1>,<macro2>,…<macro\_N>

**Đối với mã nguồn c:**

ccrh.exe <filename>.c … <filename\_N>.c –Xcpu=g4mh -g –g\_line –c –Xmsg\_lang=English -I<include dir> -<macro1>,<macro2>,…<macro\_N>

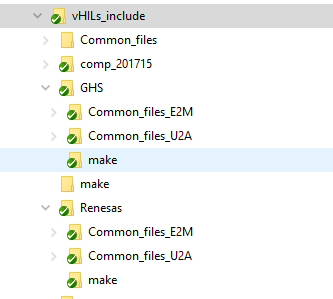
**Lệnh liên kết :**

Liên kết tất cả các đối tượng được tạo từ mã nguồn C và mã nguồn gắn kết nhanh để tạo tệp hex.

rlink.exe –subcommand=“.\<filename>.clnk”

### Cập nhật thời điểm khi hỗ trợ Renesas biên dịch

Đây là những điểm được cập nhật liên quan đến việc tạo tệp.

* Chuẩn bị mã khởi động mẫu cho Renesas biên dịch biên dịch (all E2M, U2A)
* Cập nhật dòng lệnh trên các tệp tùy chọn của trình biên dịch ( *Common file\compiler\rh850\_r4\_2\_2\_defs.mak* ).
* Cập nhật vị trí công cụ cụ thể trong tệp config.mak.
* Cập nhật dòng lệnh của liên kết trình trong tệp makefile phụ.
* Build lại thư mục tệp chung và thư mục tạo tệp mẫu.  
  ****

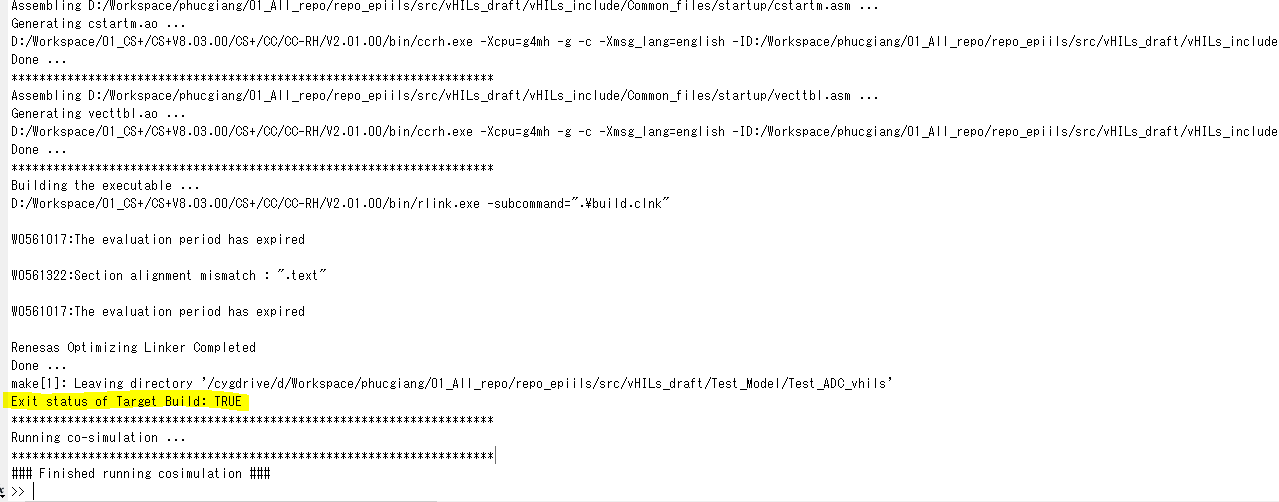
Thư mục Common\_files và thư mục tạo là thư mục được tạo cho Dòng thiết bị đích và Bảng công cụ đích.

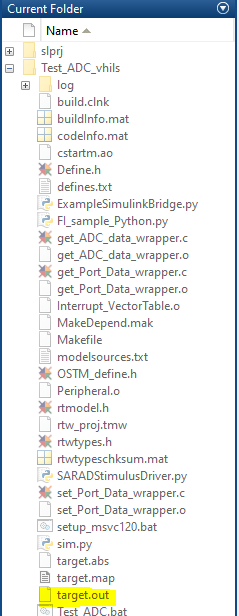
Thư mục tạo là thư mục được tạo cho Công cụ xây dựng đích.

* Liệt kê tất cả các tệp ASM có đuôi .asm. Sau đó, thêm chúng vào config.mak. (Khi sử dụng GHS, suffix là .850)
* Sửa lỗi không khớp về cú pháp giữa trình biên dịch GHS và Renesas (ví dụ: \_\_asm(<chỉ lệnh asm>), \_\_interrupt…) trong mã nguồn C.

### Kết quả sau khi thực hiện

Tệp đối tượng được tạo bằng trình biên dịch Renesas.





# Cài đặt môi trường bổ sung cho vHILS

Chuyển đến cài đặt [Windows môi trường]:

Set the left side variable

|  |  |
| --- | --- |
| Change | value |
| RH850\_USE\_ERAY\_CANOE | Điều đó |
| RLM\_LICENSE | Giấy phép của bạn.  Giấy phép RVC (Bảo mật):  5053@172.29.138.155 |
| VLAB\_HOME | Thư mục cài đặt VLAB của bạn |
| Con đường | Add <thư mục cài đặt MATLAB>  Add <thư mục cài đặt MATLAB>/bin/win64 |

# Các bước chạy mô phỏng

**1.** Mở MATLAB 2014a (MATLAB 2011 sẽ được xác minh sau).

**2.** Trên bảng điều khiển MATLAB, đăng ký máy chủ bằng lệnh **>>regmatlabserver**

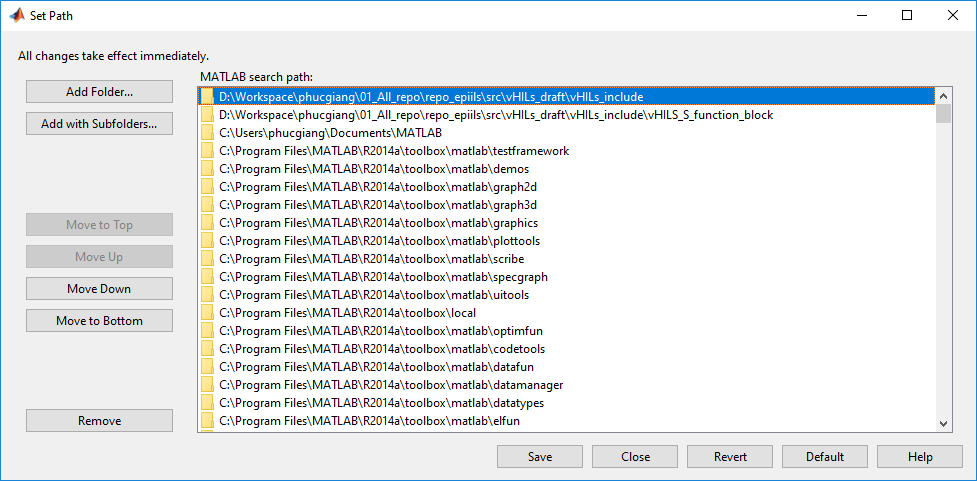
**3.** Đặt đường dẫn đến thư mục **<vHILS\_include>** (chỉ định nghĩa địa chỉ thanh ghi, mã ngoại vi, trình biên dịch Greenhill và các tệp liên kết…).

to **< vHILS\_S\_function\_block>** (chứa chức năng khối).

Tôi đã đính kèm gói với thư mục bao gồm bên trong.

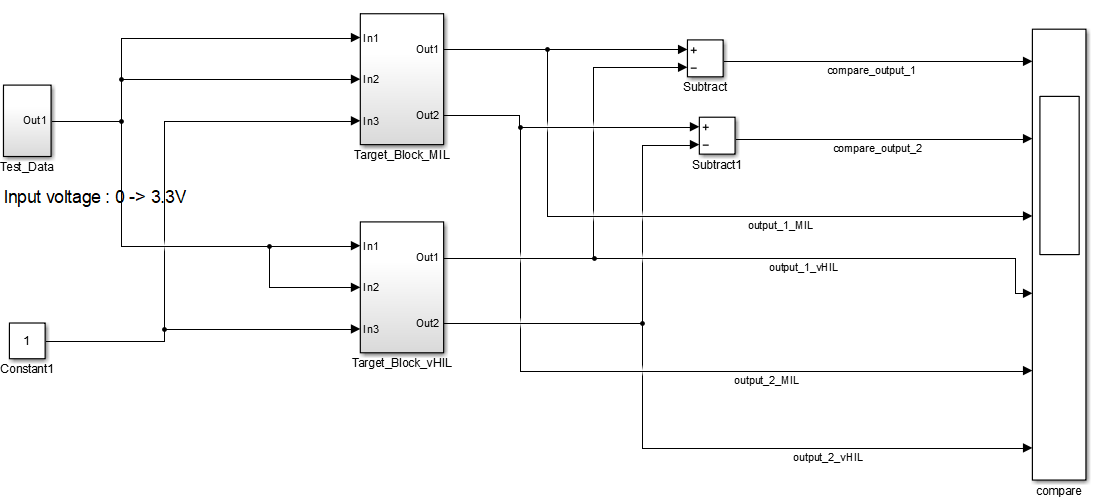
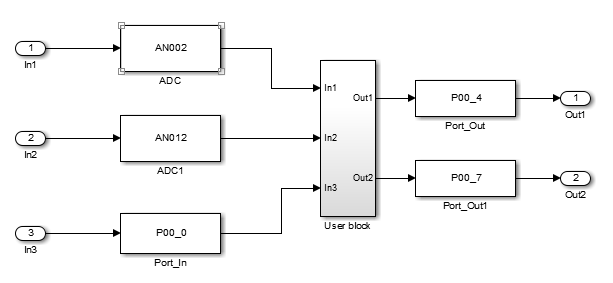
**Ghi chú:**

Để đặt đường dẫn, hãy truy cập thanh công cụ MATLAB -> Tab Trang chủ -> Danh mục môi trường -> Đặt đường dẫn

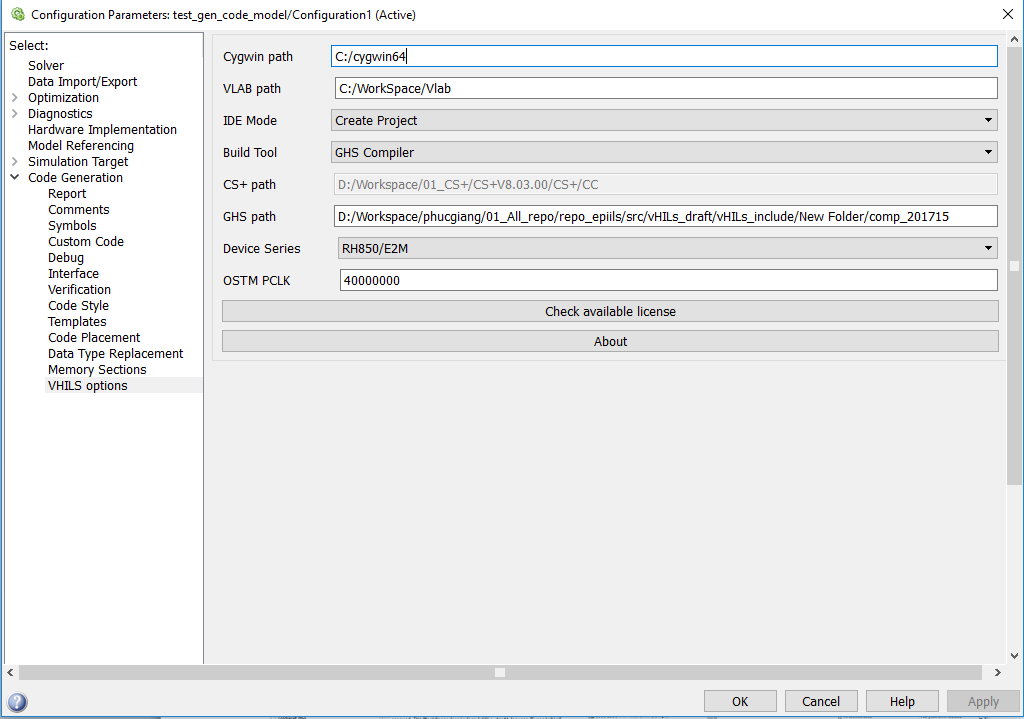


**4.** Trên MATLAB 2014a, mở mô hình test\_gen\_code\_model.slx trong thư mục <TestModel>.

Hệ thống con test\_gen\_code\_model/Target\_Block\_vHIL bao gồm ADC và khối chức năng cổng S (Có thể chọn tên cổng khác).



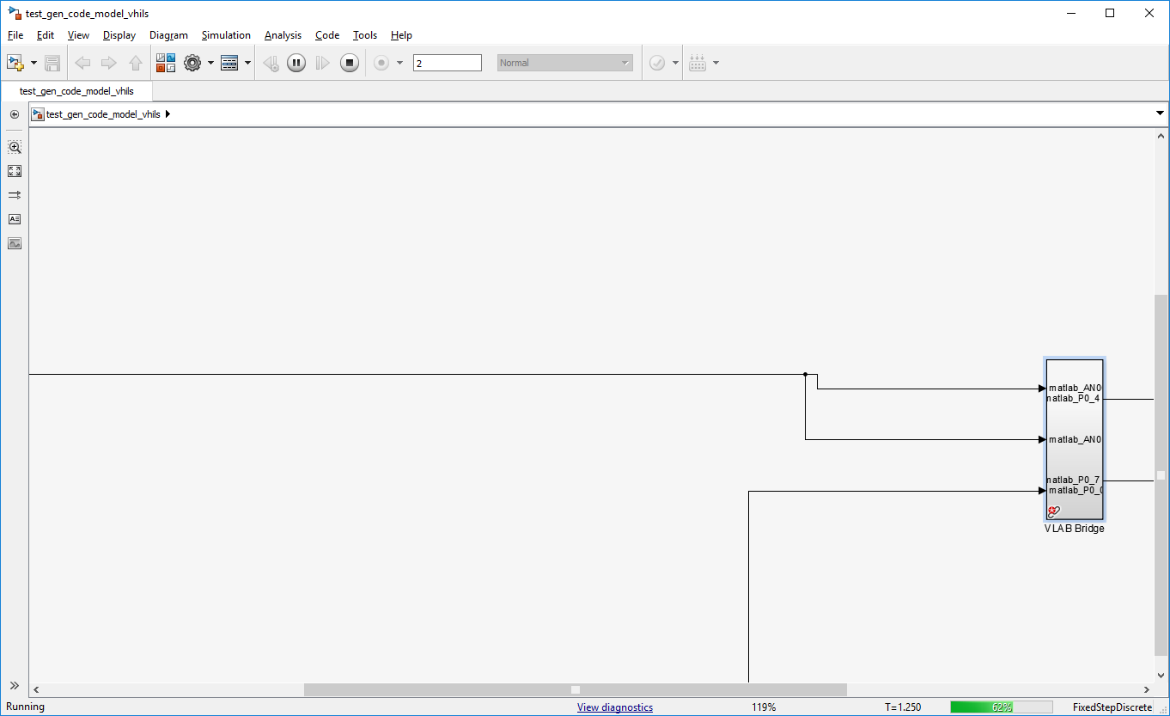
**5.** Mở hộp thoại Cấu hình vHILS và cài đặt:

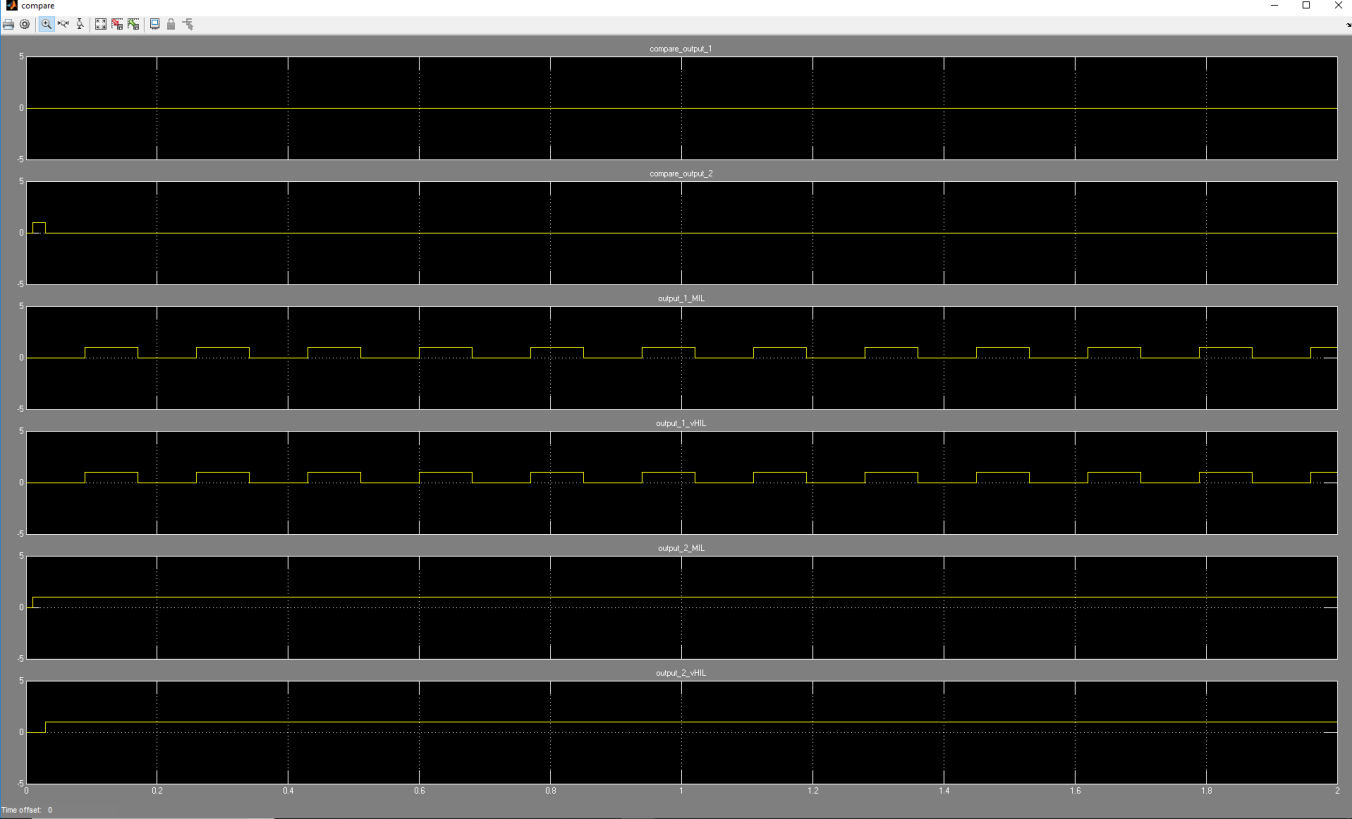


**6.** Tiếp tục tạo mã mục tiêu (trong mô hình này, đó là khối “Target\_Block\_vHIL”). Chạy lệnh MATLAB:

>> run\_vlab [Nhập]

Quá trình tạo mã và mô phỏng sẽ bắt đầu mà không cần mở cửa sổ VLAB.

**7.** Quá trình mô phỏng bắt đầu, đợi đến khi kết thúc và xác minh kết quả.

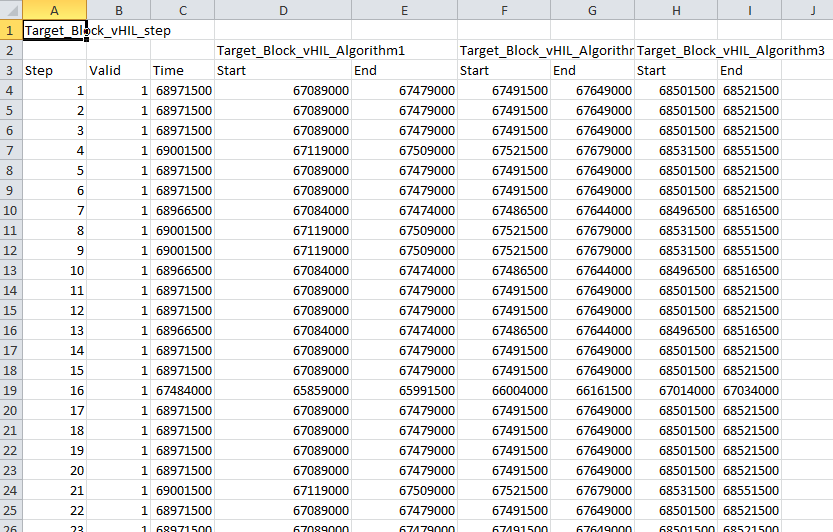
Kết quả của phạm vi bên dưới, kết quả của MIL và vHIL giống nhau.

Lưu ý: Vui lòng bỏ qua bước đầu tiên. Bởi vì, tại thời điểm đó, MATLAB và VLAB đã được kết nối và giá trị giả được gửi đến MATLAB bằng Đồng mô phỏng MATLAB.

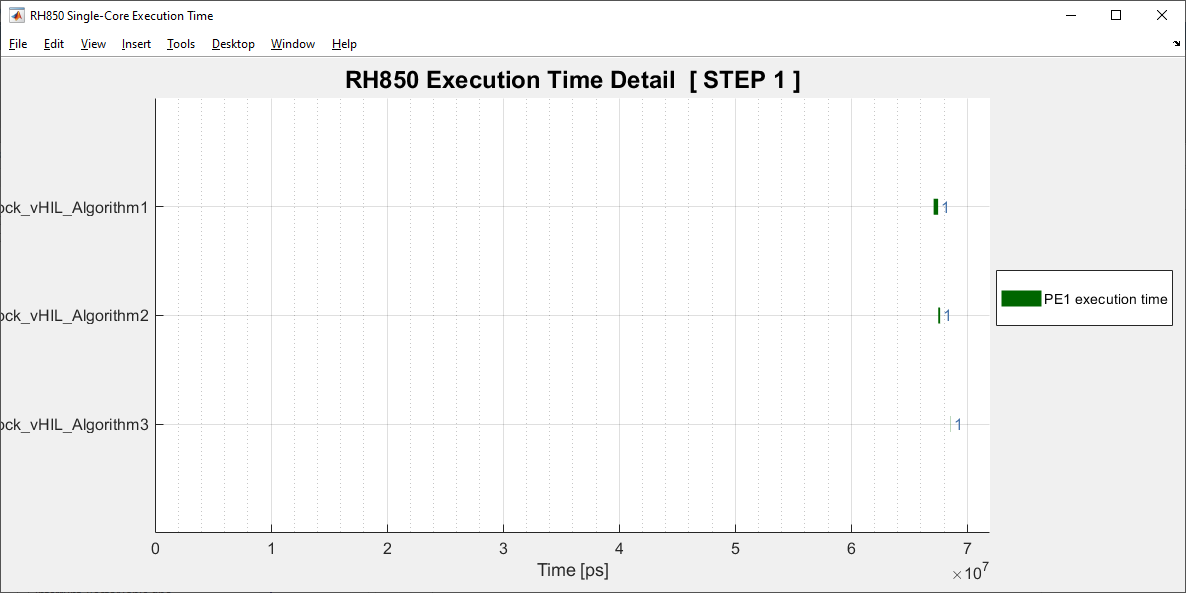
**số 8 .** Sau khi mô phỏng kết thúc, exec\_data.csv sẽ được tạo.

Đó là thời gian thực thi của mã nguồn người dùng.

Dưới đây là ví dụ:



Ngoài ra, Trình xem biểu đồ sẽ được mở sau khi đóng mô hình thực thi.



9. Thực thi lệnh bên dưới để kiểm tra bước tối đa, tối thiểu và thời gian thực hiện trung bình.

>> check\_execution\_data

# tôi nghĩ :

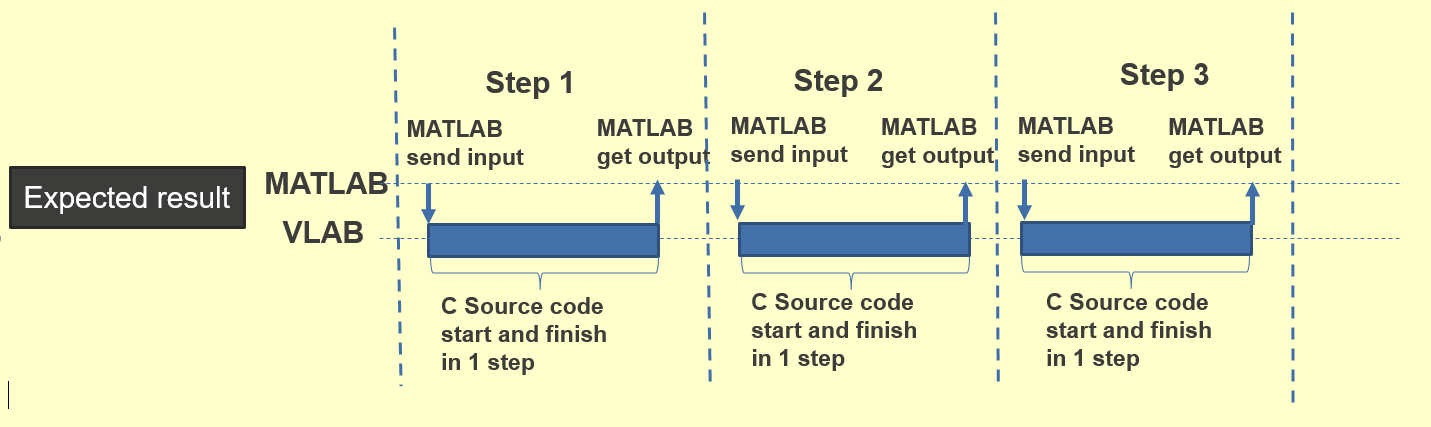
## Không đồng bộ hóa ở bước đầu p (Vấn chưa được giải quyết)

## Không đồng bộ hóa với mã thực thi thời gian dài (Đã giải quyết)

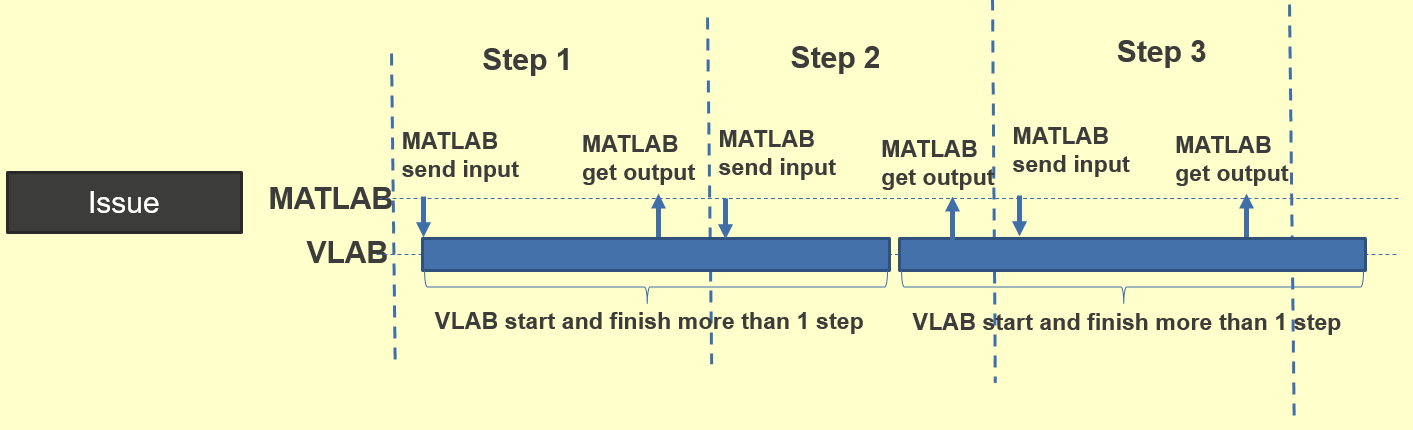
Vấn đề này Phúc-san đã báo cáo, REL đã đưa ra giải pháp đồng bộ lượng tử. Chúng tôi đã lên kế hoạch áp dụng giải pháp và xác minh trong bước tiếp theo (trước khi có thể bàn giao)

### Về vấn đề

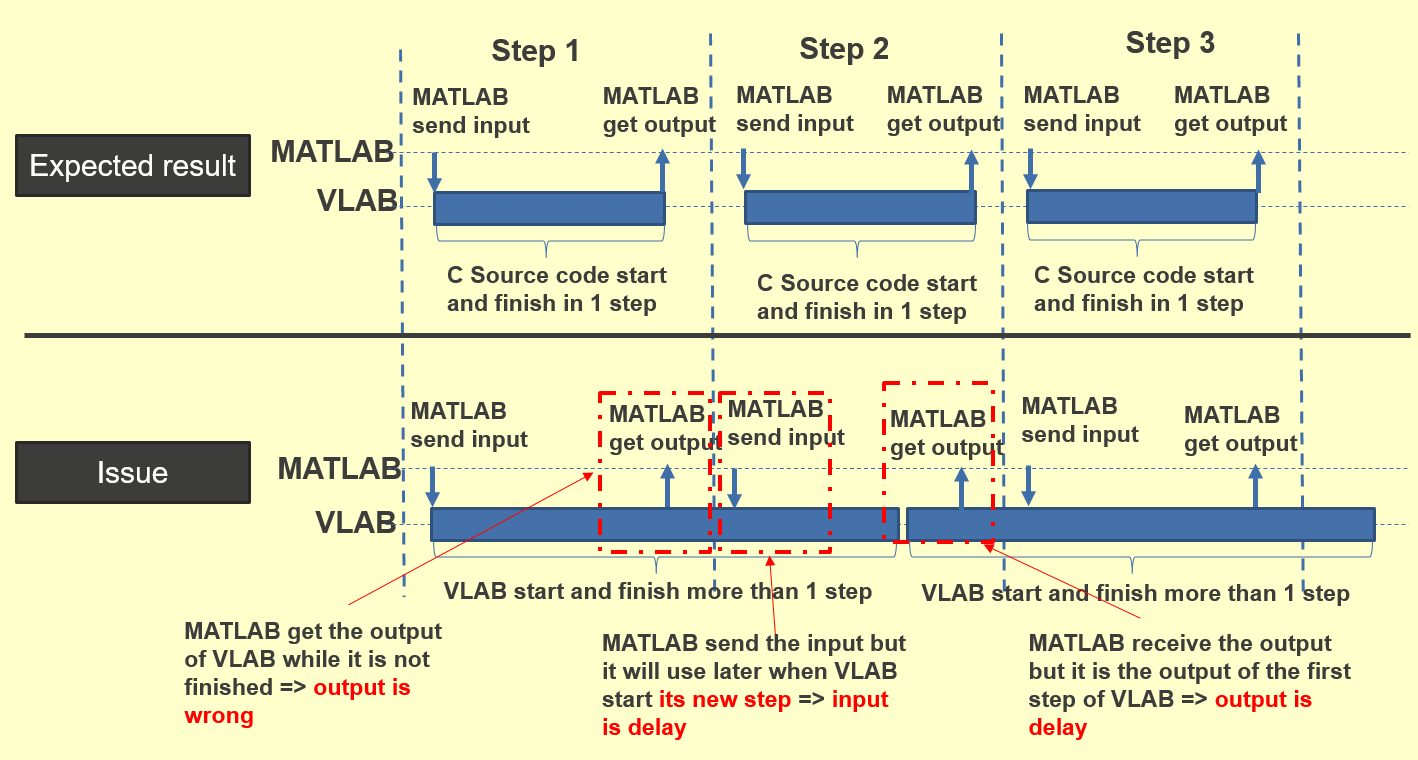
#### Sự mong đợi



#### Sự thật



#### Error

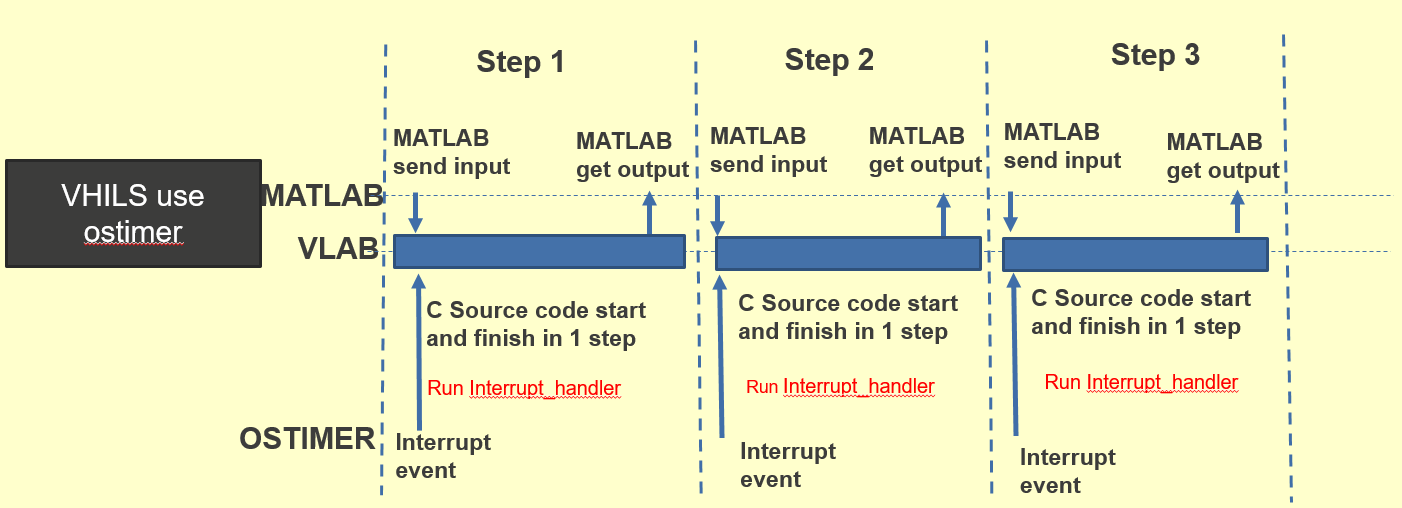


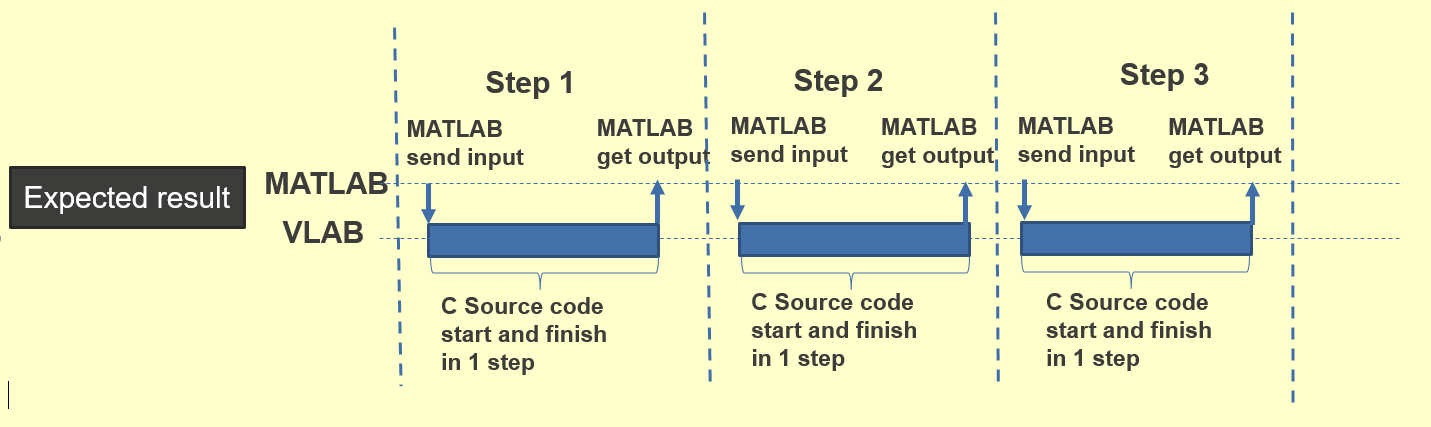
### Biện pháp đối phó - Cài đặt để sử dụng OS TIMER và thời gian lấy mẫu hợp lý.

#### tổng quan

Mục đích sử dụng OS Timer: Với khoảng thời gian ngắt quãng bằng thời gian lấy mẫu, OS Timer sẽ làm cho phần mềm điều khiển (chạy trên VLAB) chạy định kỳ (mỗi lần lấy mẫu).

Khi sử dụng OS Timer (bằng cách cài đặt trong phần 6.2.2.2 Cài đặt ghi âm ) và thời gian lấy mẫu hợp lý (do người dùng cài đặt), kết quả khi sử dụng OS TIMER so với mong đợi.





#### setting thanh ghi

1. **OSTM0**

(Tham khảo phần “ **Bộ hẹn giờ hệ điều hành (OSTM) Phần 29”** trong r01uh0641ej0110-rh850e2x-fcc1\_General.pdf )

|  |  |
| --- | --- |
| **Đăng ký** | **Mục đích** |
| OSTMnTT | Stop đếm khi bắt đầu |
| OSTMnTS | Bắt đầu đếm sau khi cài đặt xong |
| OSTMnCTL | **-** Kích hoạt ngắt.  - Cài đặt sử dụng chế độ hẹn giờ ngắt quãng.  - Nạp giá trị đếm khi bắt đầu đếm.  - Cài đặt cho phép ngắt khi bắt đầu đếm. |
| OSTMnCMP | Set time interval  Công thức là:  OSTMnCMP value = time interval (giây)/PCLK |

1. **INTC**

Theo tệp r01uh0641ej0110-rh850e2x-fcc1\_General.pdf .

* Hình 6.1 Sơ đồ khối của các đơn vị ngắt:

Đối với OSTM (Số ngắt = 18) phải dùng INTC1 để điều khiển.

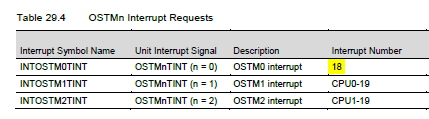
Giới thiệu về cách sử dụng thanh ghi INTC1 (tham khảo phần “ **6.3.2 EIC0 đến EIC511 — Thanh ghi điều khiển ngắt điện EI từ 0 đến 511** ” trong r01uh0641ej0110-rh850e2x-fcc1\_General.pdf )

|  |  |
| --- | --- |
| **bit name** | **Mục đích** |
| EICTn | Set 0 to play by Cận cảnh |
| EIRFn | Use to wipe off (đặt 0) |
| EIMKn | Sử dụng để bật hoặc tắt ngắt |
| EITBn | Cài đặt sử dụng bảng ngắt. |
| EIPn | Sử dụng để đặt mức ưu tiên của ngắt |

1. **Bảng ngắt mạch:**

Trong bảng vector bổ sung hàm xử lý ngắt vào **vị trí 18** (bắt đầu từ 0).

#pragma ghs phần sdata=".inttable\_PE0"

khoảng trống **(\*** IntVectors **[])(** void **)** **=**

**{**

eiint0\_handler **,** /\* EIINT0 \*/

.

.

.

giả **,** /\* EIINT10 \*/

giả **,** /\* EIINT11 \*/

giả **,** /\* EIINT12 \*/

giả **,** /\* EIINT13 \*/

giả **,** /\* EIINT14 \*/

giả **,** /\* EIINT15 \*/

giả **,** /\* EIINT16 \*/

giả **,** /\* EIINT17 \*/

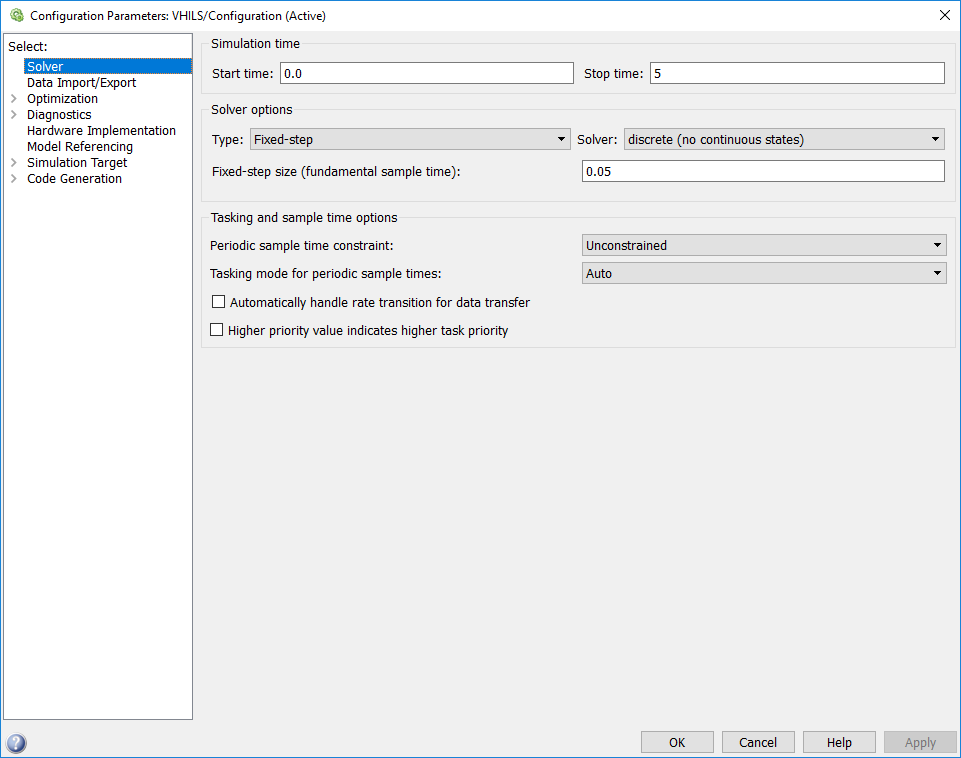
ostm\_interrupt\_handler **,** /\* OSTM0 \*/

giả **,** /\* EIINT19 \*/

#### Kết quả

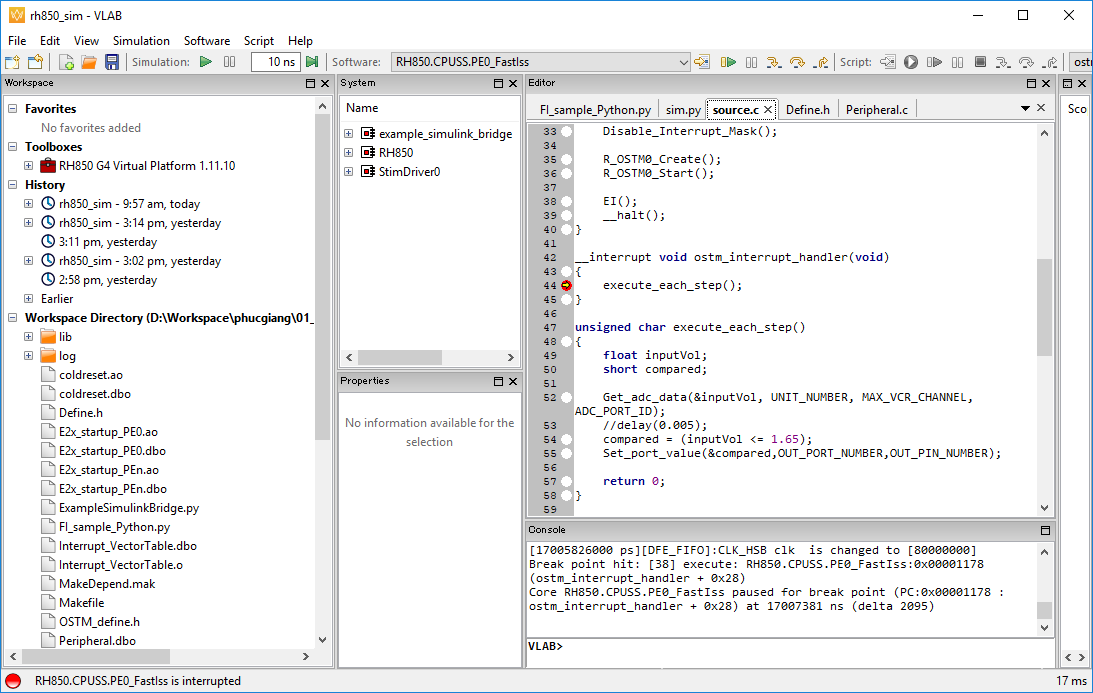
Trong kết quả mẫu này, thời gian lấy mẫu được đặt thành 50ms trên hộp thoại “ **Tham số cấu hình** ”.

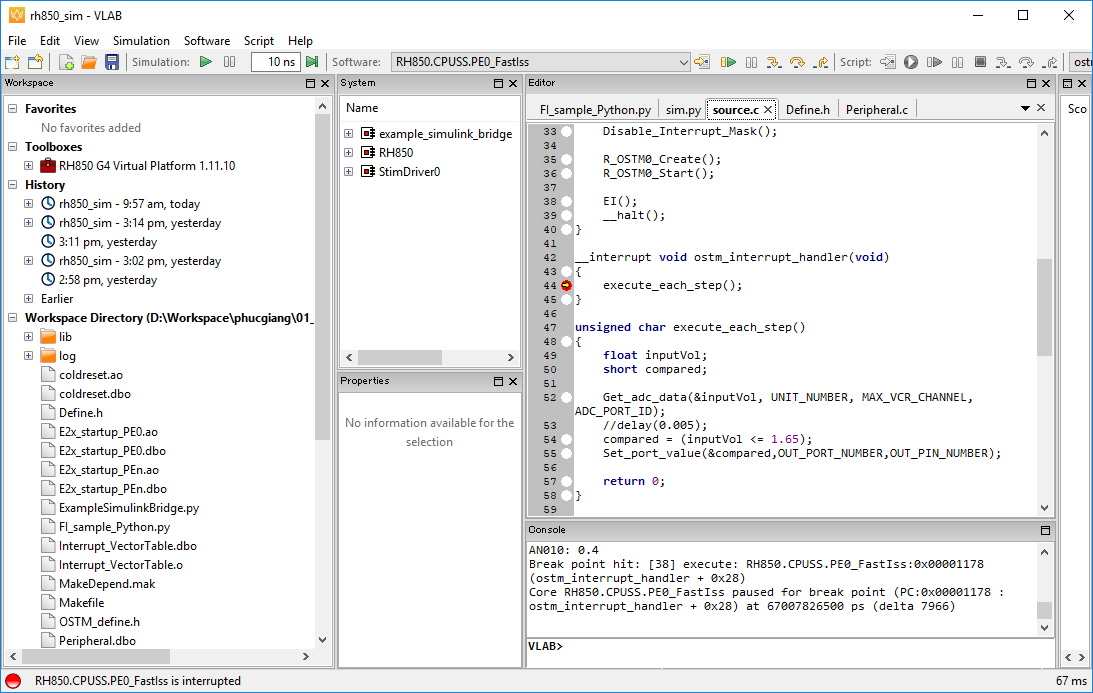
Nó được đặt tự động thành giá trị điện tử và khoảng thời gian.

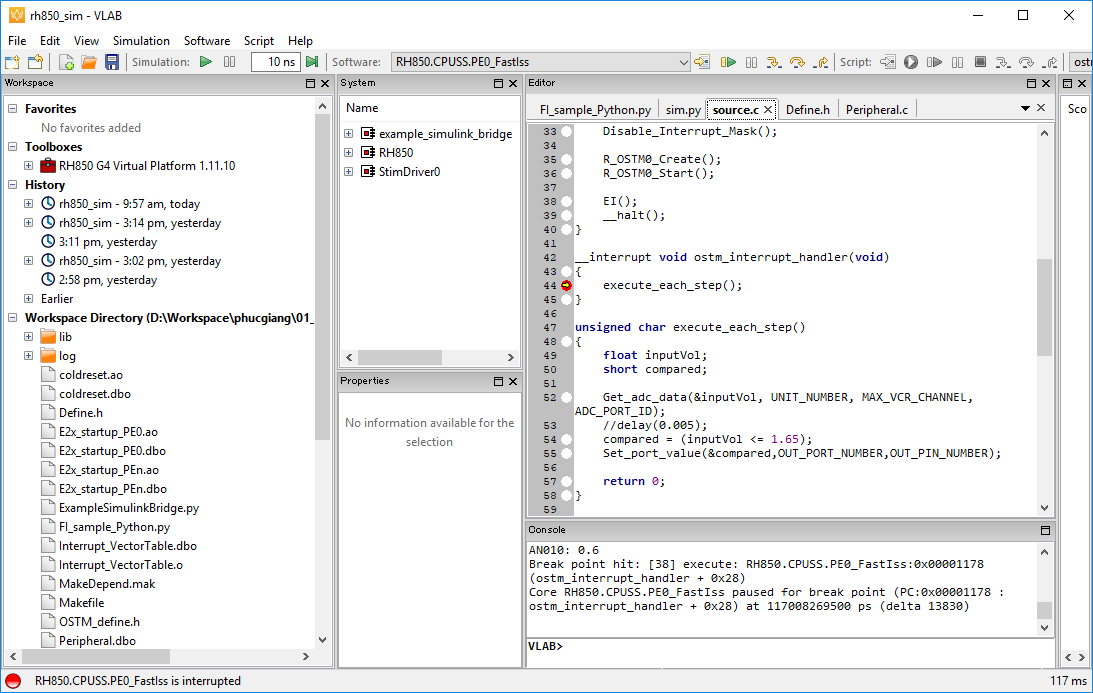


Sau khi chạy, nó sẽ giữ mã nguồn trong VLAB đang chạy định kỳ (50ms một lần).

Tham khảo 3 hình bên dưới. Mã nguồn người dùng sẽ được chạy cứ sau 50 mili giây (17 mili giây -> 67 mili giây -> 117 mili giây)

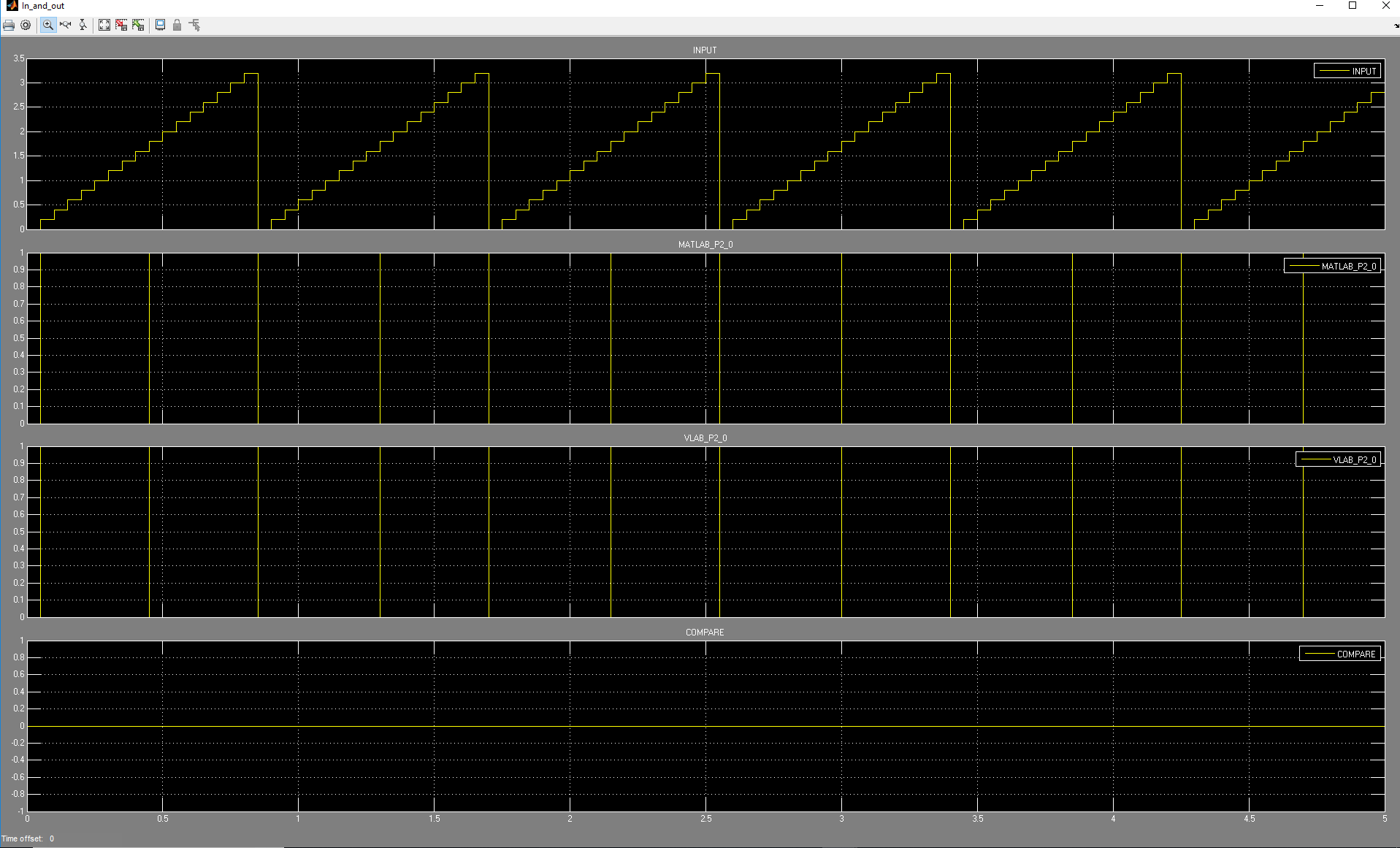






Đối chiếu với Chế độ xem biểu tượng

Kết quả MIL và vHILS trùng nhau.



# Công việc hạng mục cho giai đoạn tiếp theo (quy tắc tới)

- Hỗ trợ module CSIH, CANFD.

- Tra cứu hỗ trợ một thiết bị ngoại vi khác.