STM32-MAT/TARGET

Hands On

Rev 1.0

life.augmented

Objectives

- Hands-on workshop to show you the steps needed to quickly simulate and develop STM32 graphical applications using MATLAB® Simulink environment.
- Know tools installations and settings to be able to start development.
- Know Simulation and « C » Code Generation possibility
- Know how to develop application from scratch
- Know where to obtain additional technical support

Systems Check

- Mandatory Software
  - From Mathworks
    - MATLAB®, Simulink and Embedded Code 
  - From STMicroelectronics
    - STM32CubeMX 
  - One of following Toolchain
    - EWARM from IAR 
    - MDK-ARM from Keil 
    - TrueSTUDIO from Atollic 
    - SW4STM32 from STMicroelectronics 
  - STM32-MAT/TARGET toolkit to develop STM32 applications

- Hardware
  - Any electronic application board with STM32 and SWD/JTAG connection.
  - STLinkV2 or 3rd parties dongle if not integrated to STM32 application board.
  - USB to Serial adapter.

Hardware setup

Step #1 – Hardware selection

- Use one of STM32 boards including STLinkV2
  - Nucleo, Discovery, EvaluationBoard etc…
  - STM32F3348-DISCO and STM32F429i-DISCO

  will be used during examples.



- Or STM32 application board connected to SWD (Single Wire Debug)/JTAG dongle
  - STLinkV2, ULink2, Jlink etc..

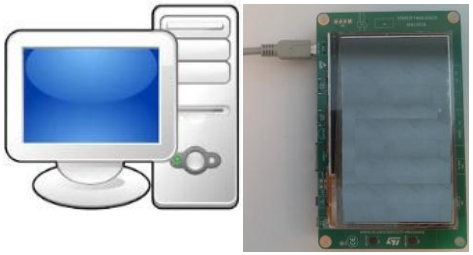

- For PIL (Processor In the Loop) communication.
  - USB to Serial adapter is needed or board integrating VCP(Virtual COM Port)

Step #2 – Hardware connection

- Connect USB dongle port to PC USB port

- And connect STM32 HE10 20 pins dongle connector to STM32 target board

- Or connect PC USB port to embedded STLinkV2



Step #3 – Hardware connection

- As soon as you are using ST-LINK/V2
  - look at http://www.st.com/web/catalog/tools/FM146/CL1984/SC720/SS1450/PF251168?s_searchtype=partnumber
  - « Related Tools and Software » section to check or update firmware

Related Tools and Software

Related Tools and Software	
Part Number	Description
STSW-LINK004	STM32 ST-LINK utility
STSW-LINK005	ST-LINK/V2 firmware upgrade
STSW-LINK009	ST-Link, ST-Link/V2, ST-Link/V2-1 USB driver signed for XP, Windows7, Windows8

Software setup

Quick description of tools

MATLAB®

High level language for complex calculation

Simulink

Graphical development environment
Complete environment of simulation and implementation of embedded systems.

Embedded Coder

C code generation for embedded system.
Embedded system interface

STM32Cube Embedded Software

Collection of embedded software components, highly portable from one STM32 to another

STM32CubeMX

Configuration software tool on the PC, able to generate initialization C code versus user choices

Toolchain

One toolchain from partners or ST is required to compile and link C code generated by Embedded Coder, STM32CubeMX and STM32Cube embedded software

IAR SYSTEMS

KEIL™ Tools by ARM

atollic

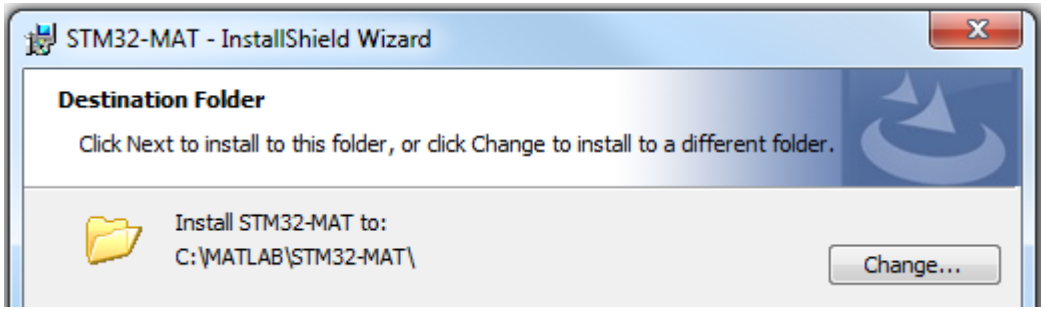
STM32

Step #1 – Software installation

- Install MathWorks software (R2015b or later)
  - MATLAB®, Simulink, Embedded Coder are mandatory
  - Add-ons : SimPowerSystems, Simscape and Staflow for motor control applications if needed.
  - http://www.mathworks.com
- Install STM32CubeMX
  - Download and documents available from : *www.st.com/microxplorer*
- Install toolchain (Cf Slide 3 : « Systems Check »)
  - Cf Slide 3 « Systems Check » to get link to supported 3rd parties download area.

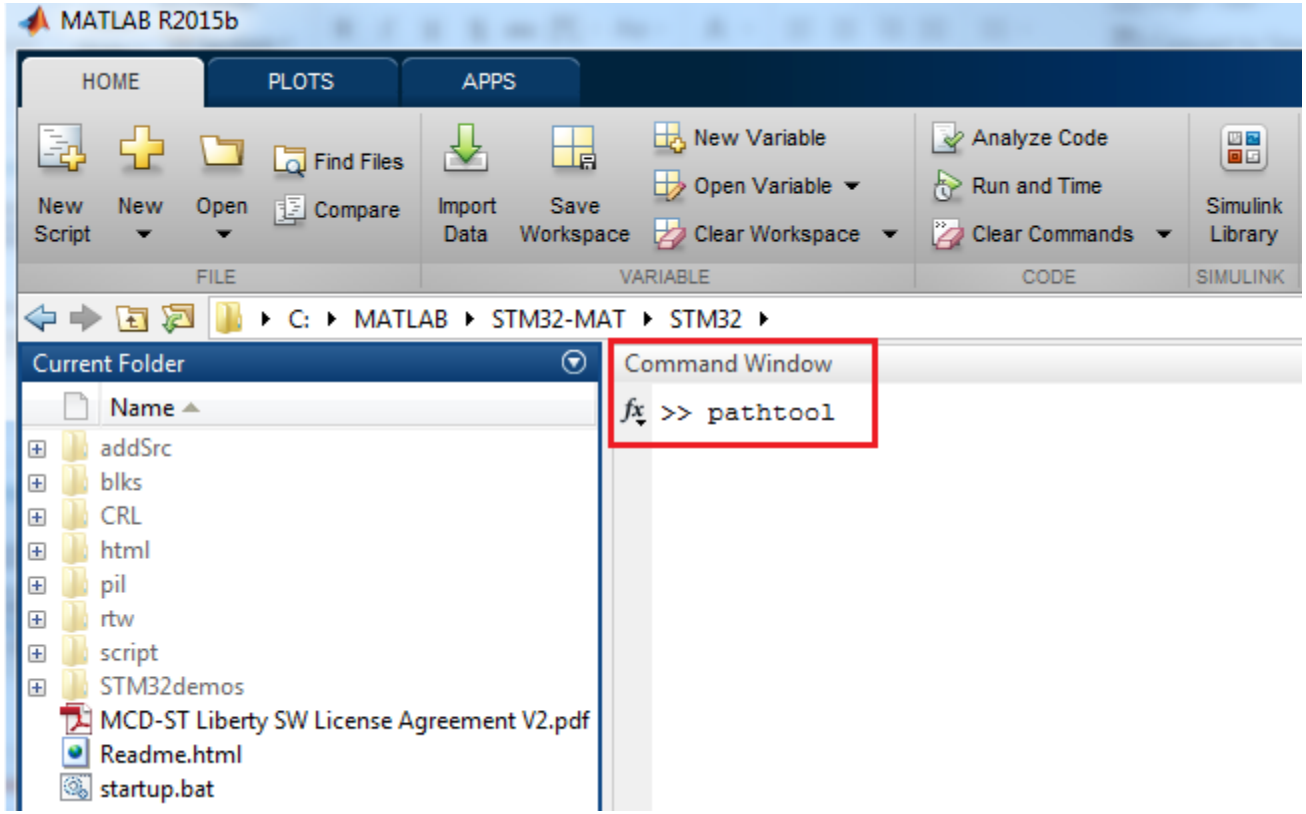
Step #2 – Software installation

- Install STM32-MAT/TARGET
  - STM32 embedded target for MATLAB® and Simulink
  - This toolkit is mandatory to integrate STM32 to MathWorks tools.
  - Download and documents available from : click here
  - Unzip « stm32-mat_target.zip »
  - Run «STM32MatTarget_xx_setup.exe » (xx = software version)
  - Default installation path is « C:\MATLAB\STM32-MAT » repository



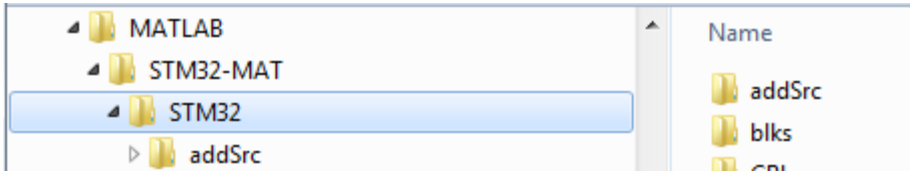
Step #3 – STM32-MAT/TARGET integration

- Integrate STM32-MAT/TARGET to MathWorks flow
  1. Open MATLAB®
  2. Enter « pathtool » command from CommandWindow

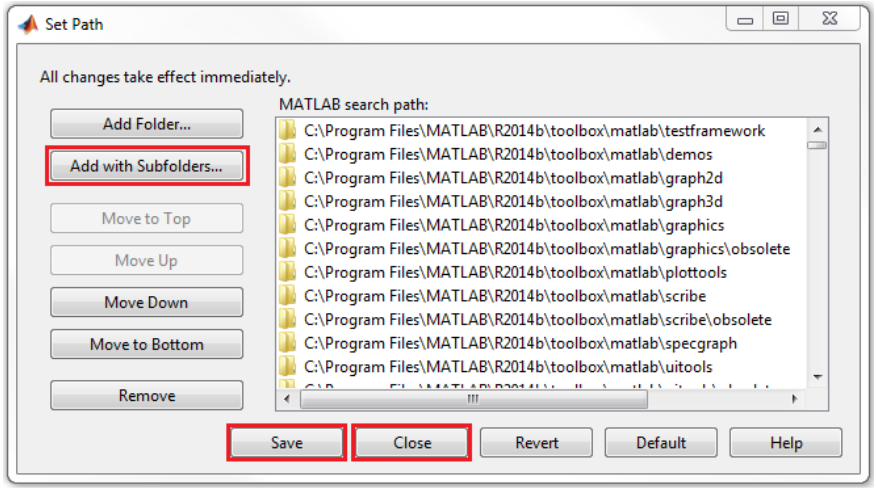


Step #4 – STM32-MAT/TARGET integration

3. Click « Add with Subfolders… » button on « Set path » window.
4. Select STM32 installed path
   - Default path :**C:\MATLAB\STM32-MAT\STM32**



5. Click « Save » then « Close » button.



6. Simulink is then ready to design STM32 application.

Scenarios and Tools usage

Simulation / PIL / Code Generation

- Step 1: Pure simulation
  - Everything done on the PC. Only MATLAB® and Simulink are needed

MATLAB®/ Simulink

- Step 2: Processor-in-the-loop (PIL)
  - Algorithm fully executed on STM32
  - Data (Input or output) exchanged between MATLAB®/Simulink and STM32 via UART

MATLAB®/ Simulink

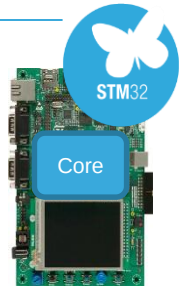
Embedded Coder

STM32Cube Embedded Software

STM32CubeMX

*: used only for UART

Toolchain



- Step 3: Everything on STM32
  - Data (input or output) obtained within STM32 through its peripherals (ADC, Timers, …) and algorithm fully executed on STM32

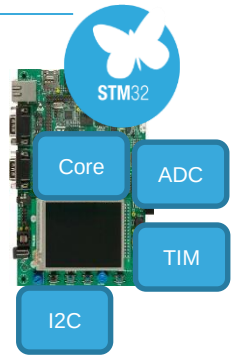
MATLAB®/ Simulink

Embedded Coder

STM32Cube Embedded Software

STM32CubeMX

Toolchain



- Scenarios are independent and can be done individually

Scenarios

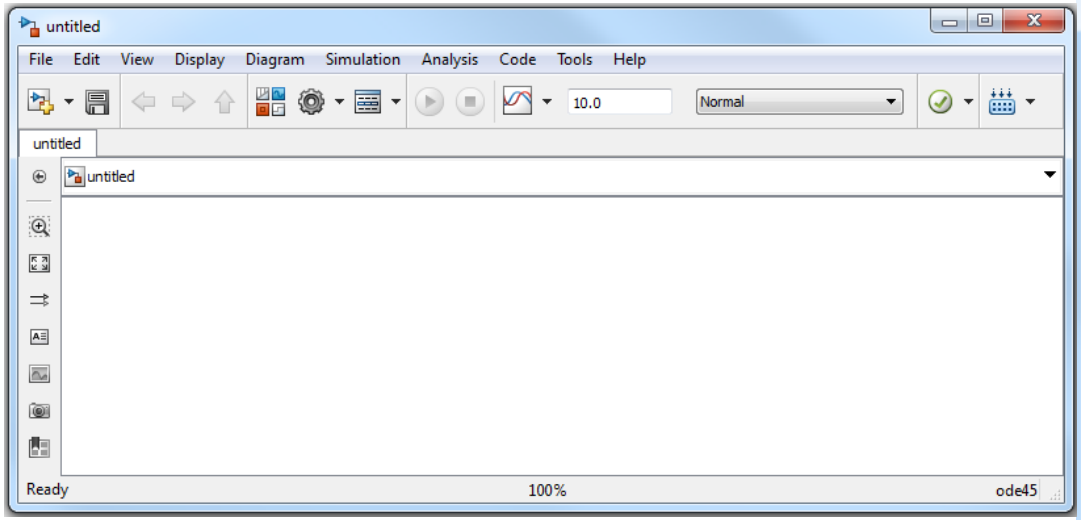
- Simulink graphical applications can :
  - Be Simulated on PC.
    - MATLAB® and Simulink needed only.
  - Process Simulink data on STM32 target
    - MATLAB®, Simulink, STM32CubeMX, one toolchain and STM32-MAT/TARGET must have been installed.
  - Generate C code project for this application
    - MATLAB®, Simulink, STM32CubeMX, one toolchain and STM32-MAT/TARGET must have been installed.

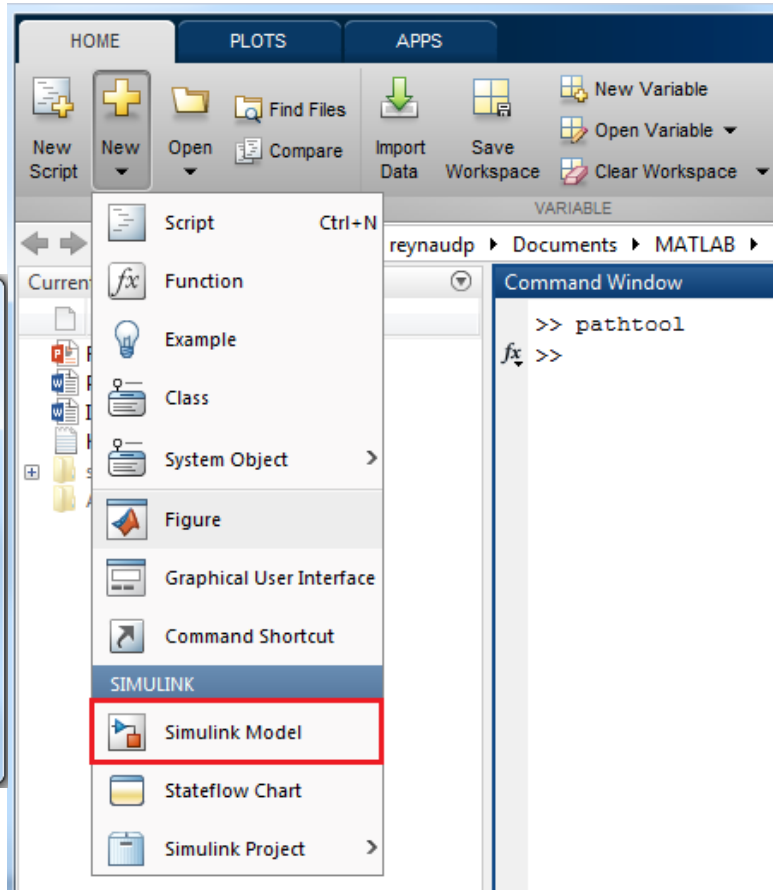
- Simulink model must be created and configured to be able to be able to develop STM32 graphical applications.

Simulink model configuration

Simulink Model Setting 1/4

- From MATLAB® IDE click New>Simulink Model to open a new Simulink graphical application model.

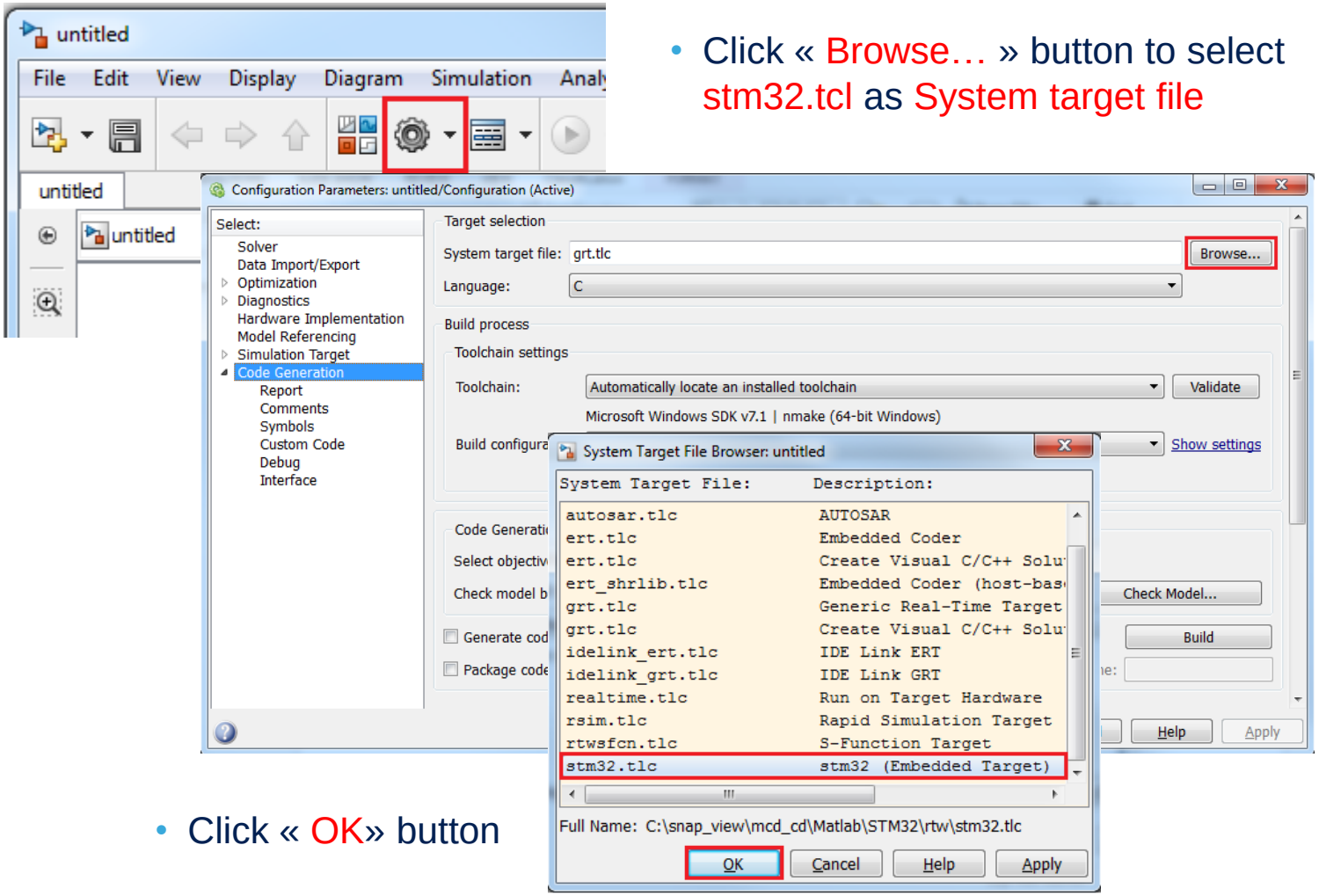




- Then application parameters must be set for STM32 target…..

Simulink Model Setting 2/4

- Open Configuration Parameters window and select Code Generation

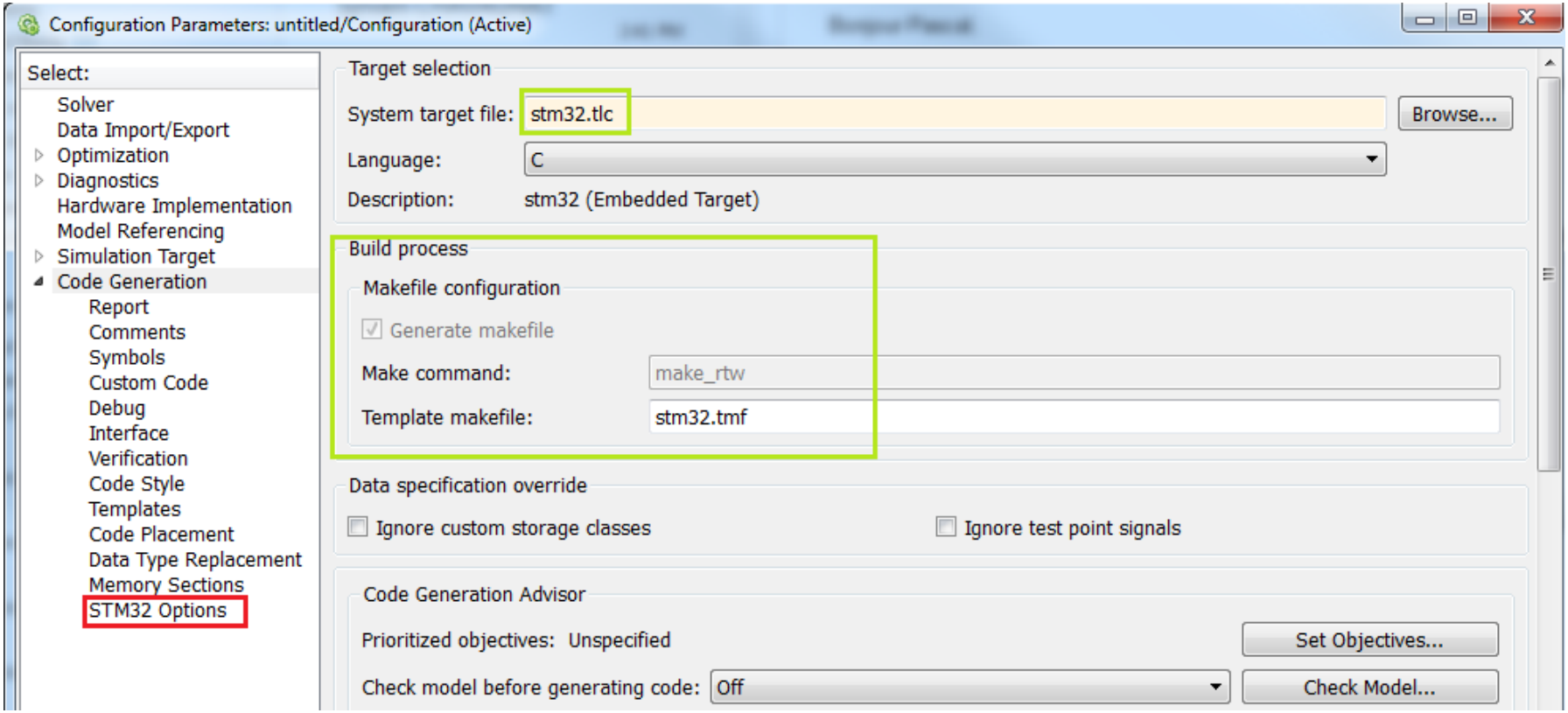


- Click « Browse… » button to select stm32.tcl as System target file

- Click « OK» button

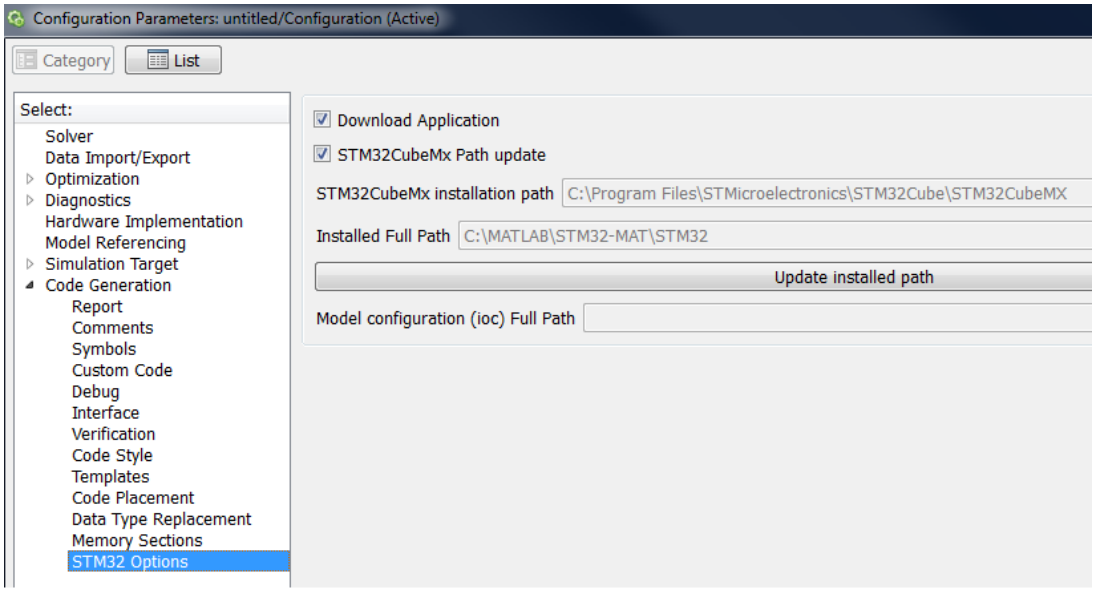
Simulink Model Setting 3/4

- stm32.tlc has been selected, Build process parameters has changed
  - Select STM32 Options



Simulink Model Setting 4/4

- STM32CubeMX Path update is used to automatically update STM32CubeMX installation path.
  - Path selected during STM32CubeMX installation
- Download Application is used to start STM32CubeMX to generate project.
  - Uncheck for code generation only
- Installed Full path is STM32-MAT/TARGET installation path.
  - Default path : C:\MATLAB\STM32-MAT\STM32
- Update installed path to update path when it has changed
- Model configuration (ioc) Full path is hardware configuration file path created using STM32CubeMX for this Simulink application
  - Read only, updated from STM32_Config model.

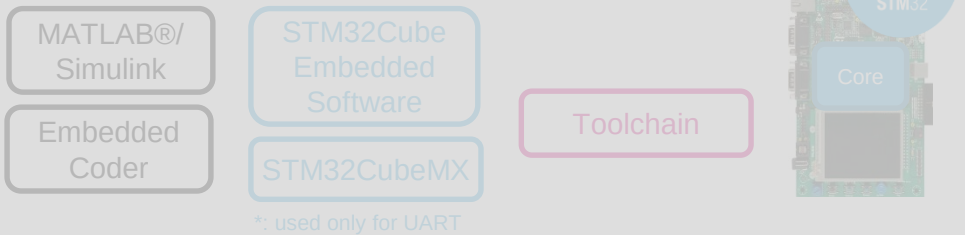


PC pure simulation

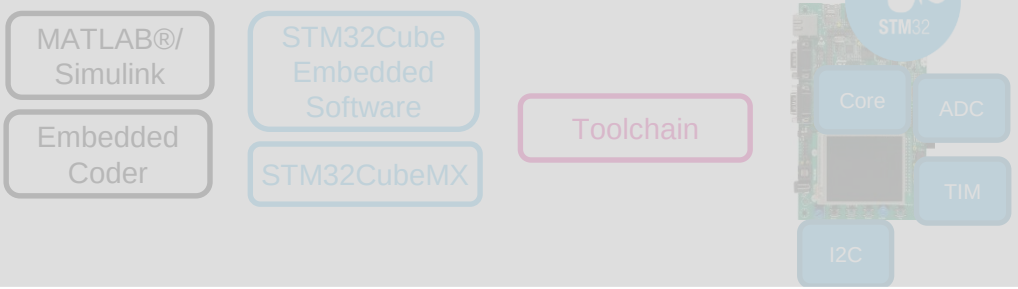
Tools usage

- Step 1: Pure simulation
  - Everything done on the PC

- Step 2: Processor-in-the-loop (PIL)
  - Algorithm fully executed on STM32
  - Data (Input or output) exchanged between MATLAB®/Simulink and STM32 via UART



- Step 3: Everything on STM32
  - Data (input or output) obtained within STM32 through its peripherals (ADC, Timers, …) and algorithm fully executed on STM32



MATLAB® & Simulink

- Not a subject for this HandsOn
- General purpose for MATLAB®/Simulink
- Doesn’t need STM32-MAT/TARGET toolkit.
- Contact MathWorks for MATLAB®, Simulink trainings

Processor in The Loop (PIL)

Tools usage

- Step 1: Pure simulation
  - Everything done on the PC

MATLAB®/ Simulink

- Step 2: Processor-in-the-loop (PIL)
  - Algorithm fully executed on STM32
  - Data (Input or output) exchanged between MATLAB®/Simulink and STM32 via UART

MATLAB®/ Simulink

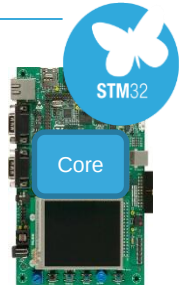
Embedded Coder

STM32Cube Embedded Software

STM32CubeMX

*: used only for UART

Toolchain



- Step 3: Everything on STM32
  - Data (input or output) obtained within STM32 through its peripherals (ADC, Timers, …) and algorithm fully executed on STM32

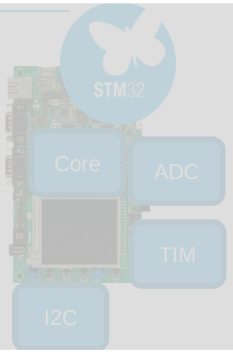
MATLAB®/ Simulink

Embedded Coder

STM32Cube Embedded Software

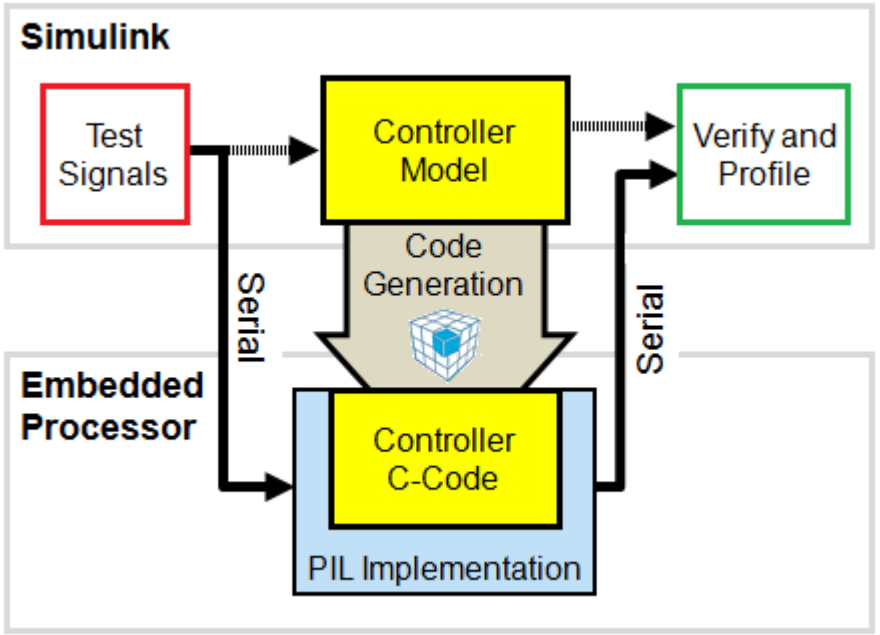
STM32CubeMX

Toolchain



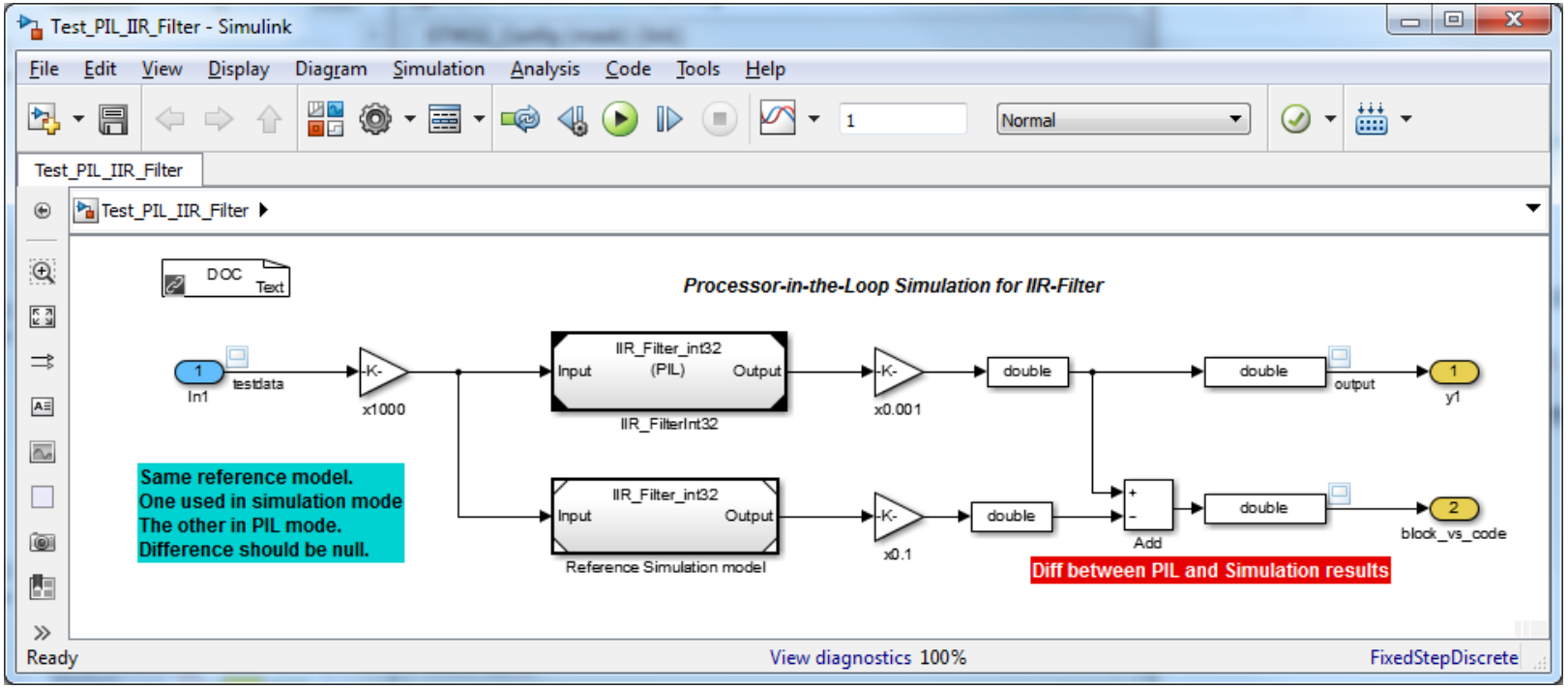
PIL Overview

- PIL is used to run MATLAB® algorithm on STM32
  - To see if STM32's computational results are numerically equivalent to PC-based simulation results.
  - To measure directly the time to execute the model steps (**profiling**)
- PIL simulation uses serial port to send data from Simulink to STM32 and receive Simulink processed data back from STM32
- PIL simulation doesn't run in « real-time »
  - Simulink is « master » sending asynchronously data to STM32 through serial port.
  - STM32 waits for data from Simulink, process data (in real-time) and send it back to Simulink through serial-port.
- PIL simulation doesn't process real data from STM32 peripherals.
  - STM32 peripherals (ADC etc…) are not used.
  - Only STM32 USART peripheral is used to communicate with Simulink.



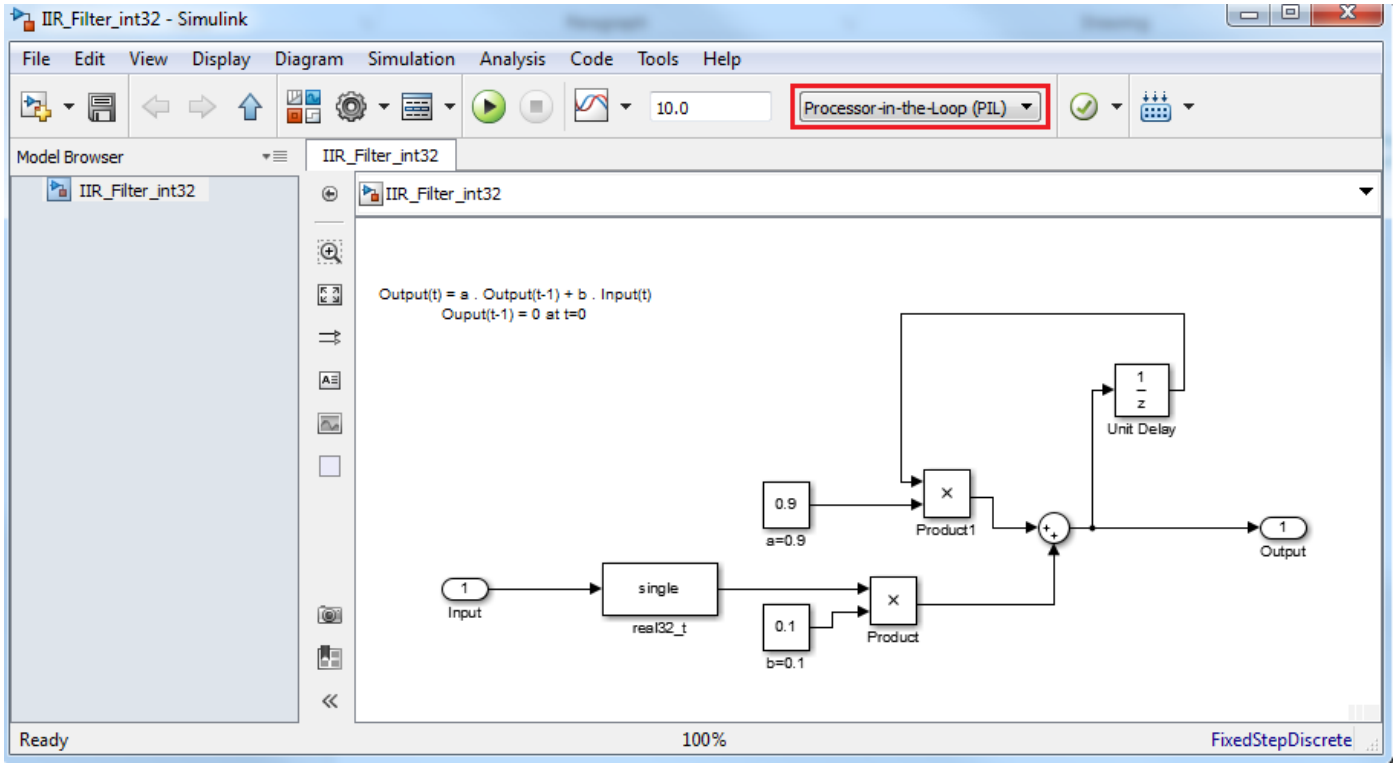
PIL Example

- IIR Filter example
  - Open Test_PIL_IIR_Filter.mdl model example from C:\MATLAB\STM32-MAT\STM32\STM32demos\PIL\Filter (Default STM32-MAT/TARGET installation path)



Reference Model for PIL

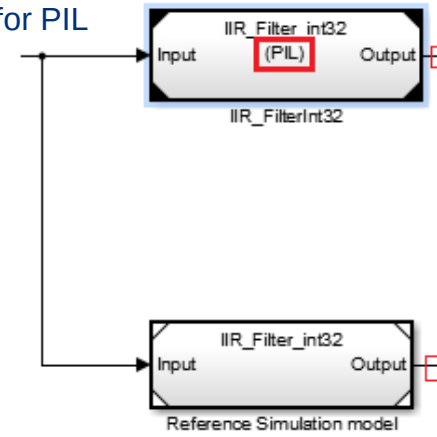
- It uses IIR_Filter_int32.mdl as reference model.
- IIR_Filter_int32.mdl is a simple algorithm for one order filtering.
- IIR_Filter_int32.mdl is set to be used for Processor-in-the-loop (PIL)



PIL setting vs Simulation

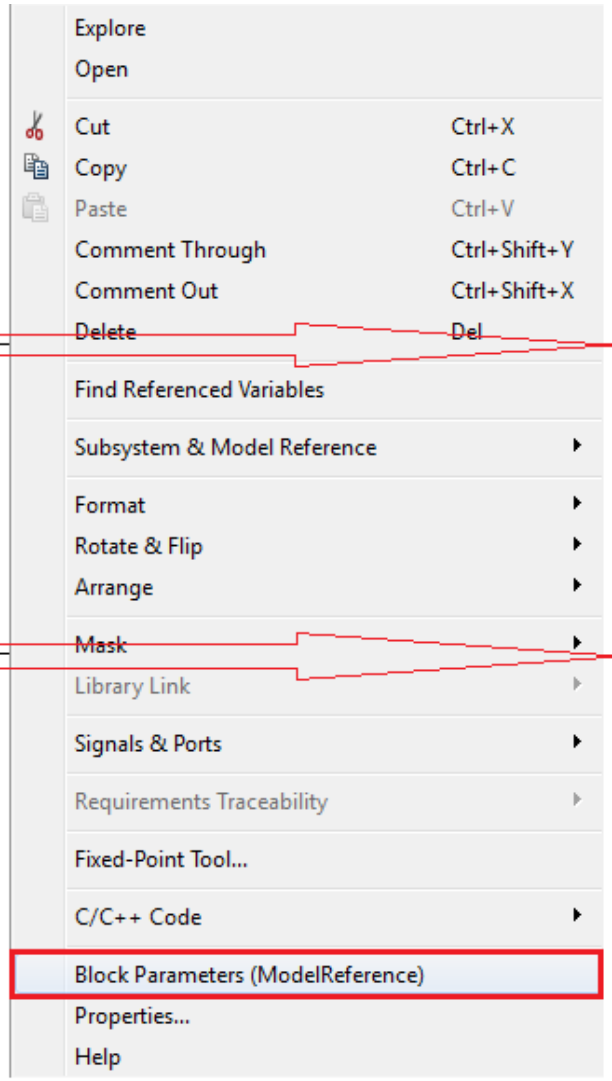
- IIR_filter_int32 model is used twice :

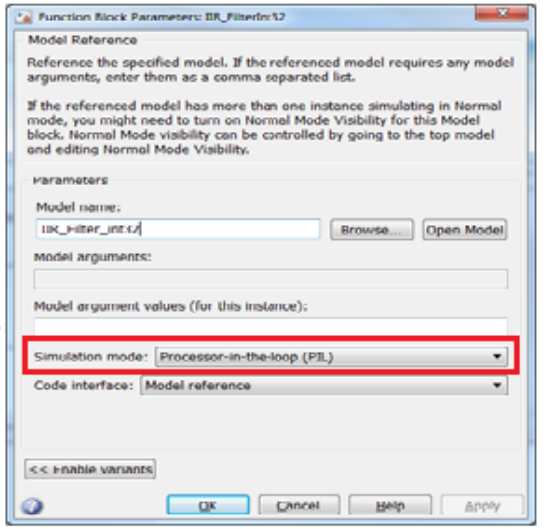
- One is set for PIL

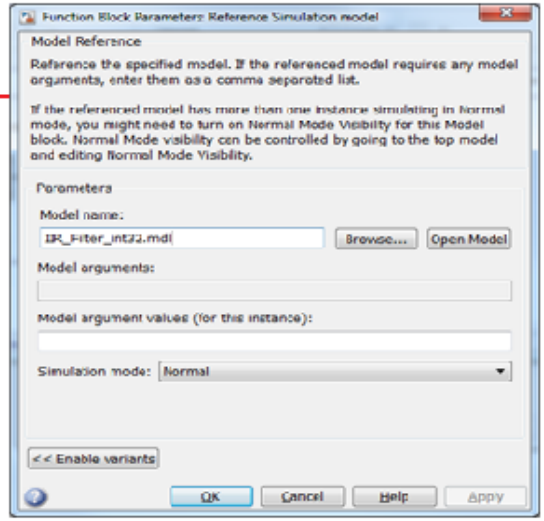


- One is set for simulation model (Normal)

- Click right mouse button on each reference model to open select Block Parameters



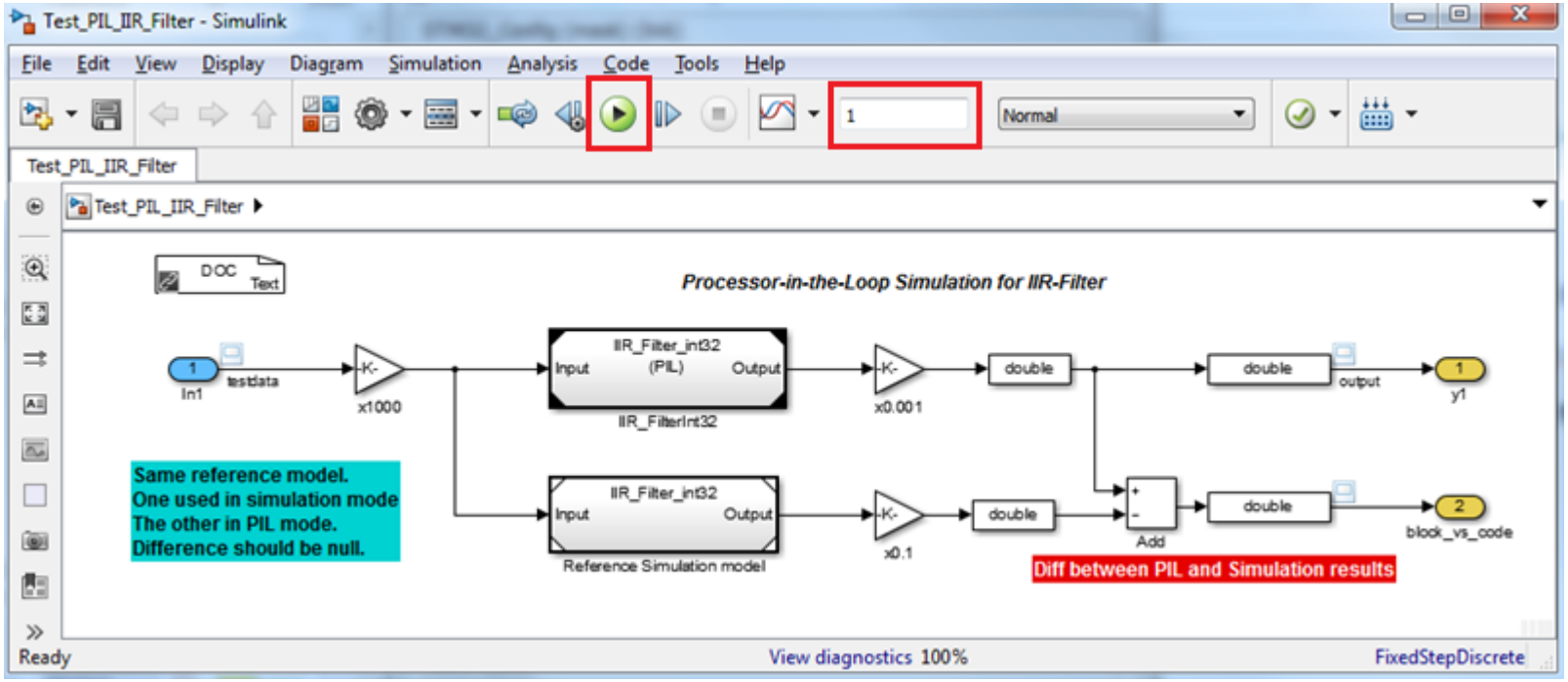




life.augmented

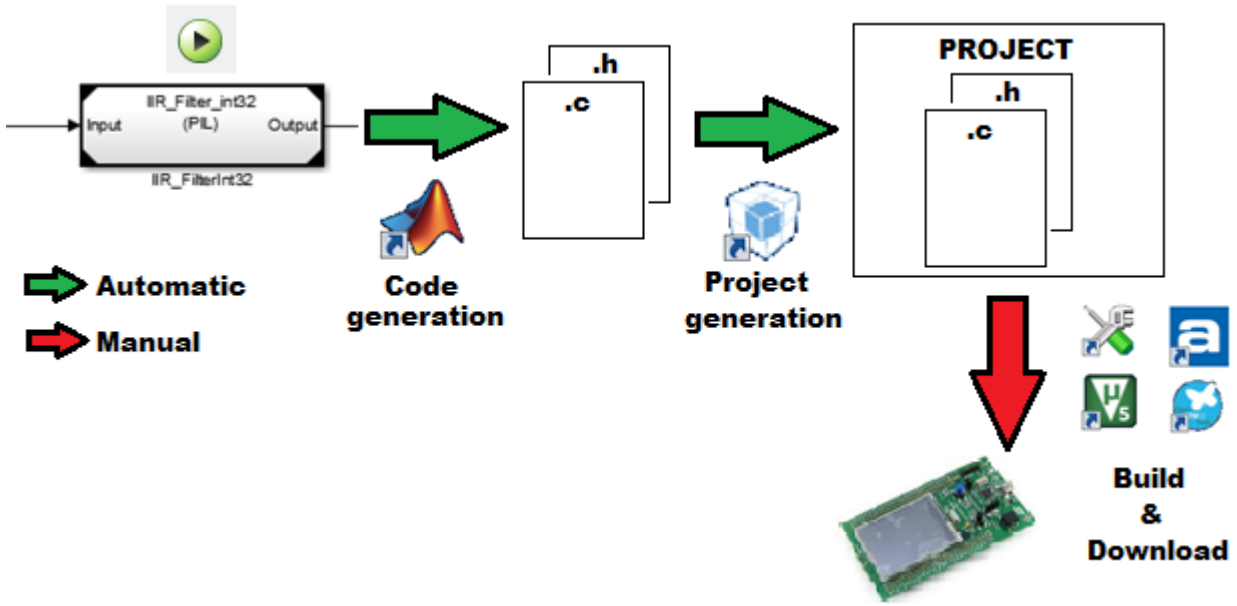
Simulink PIL start

- Set simulation duration time and click run simulation green button
  - Example : Simulate 1second



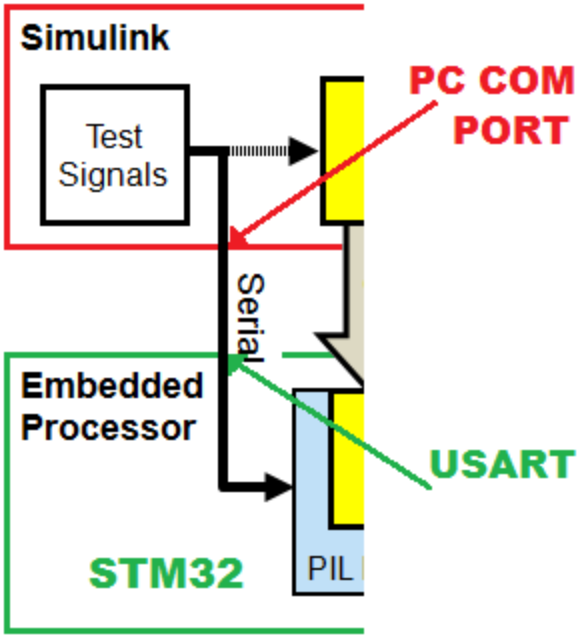
Code generation overview

- PIL reference model runs into STM32 target as simulation model runs on PC
  - Automatic convertion of PIL model to .c/.h files
  - Automatic call to STM32CubeMX to create project
  - Manually build and download project to STM32 target from selected toolchain



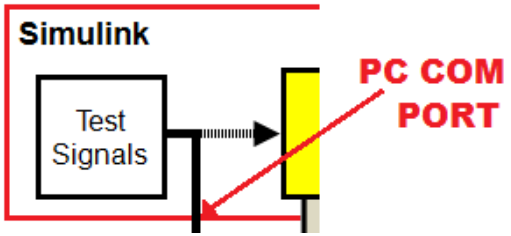
PC/STM32 communication overview

- Some parameters are requested when you start PIL
  - STM32 Target
    - STM32 device that will run the application
  - PC Communication Port
    - COMx
      - Default parameters : 115200bds, 8b, no parity, 1 stop
  - STM32 Communication Port
    - USARTx
      - USARTx, Port and Pins for Rx/Tx
      - Same parameters as PC COM Port selection

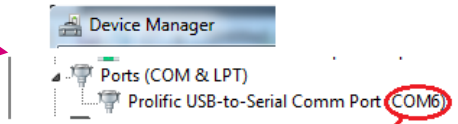


PIL Processing 1/5

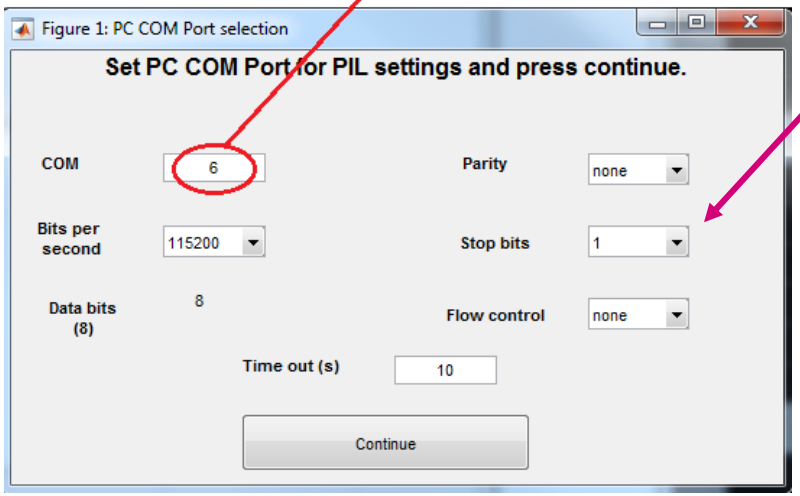
- PC COM Port from Device Manager
  - 8b is fixed, other parameters can be changed
  - Time Out: Simulink message error after 10 seconds without communication.



Look at Windows «Device Manager» to find PC COM Port



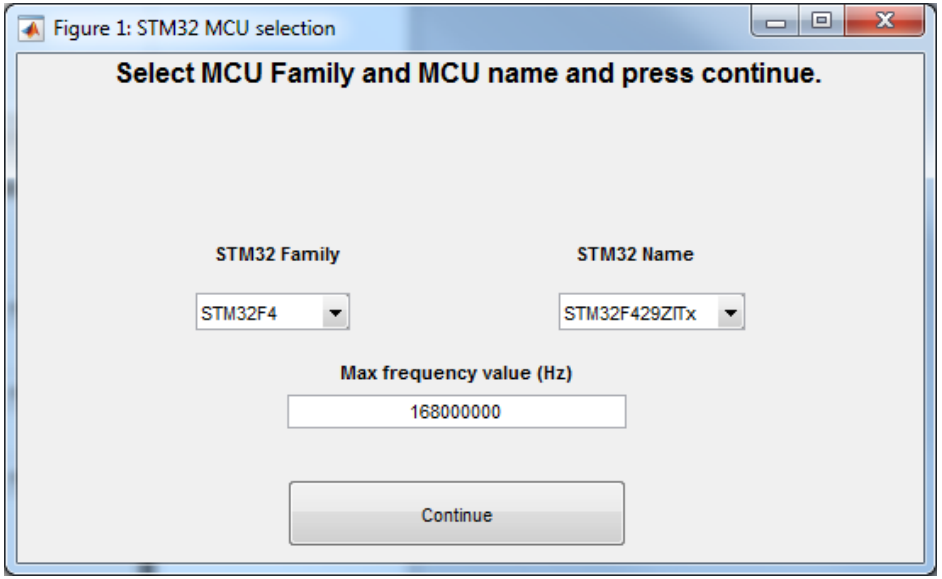
Set parameters



PIL Processing 2/5

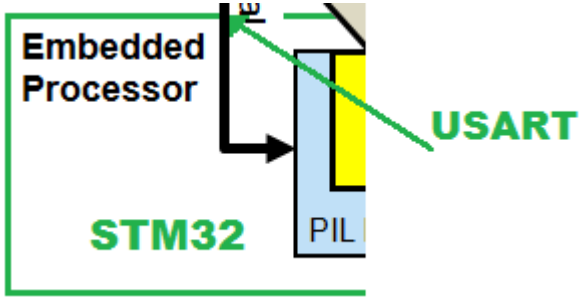
- STM32 Selection
  - Example with STM32F429i-DISCO board
    - Family : STM32F4
    - Name: STM32F429ZITx
    - Frequency:168MHz (by default STM32max speed is selected)

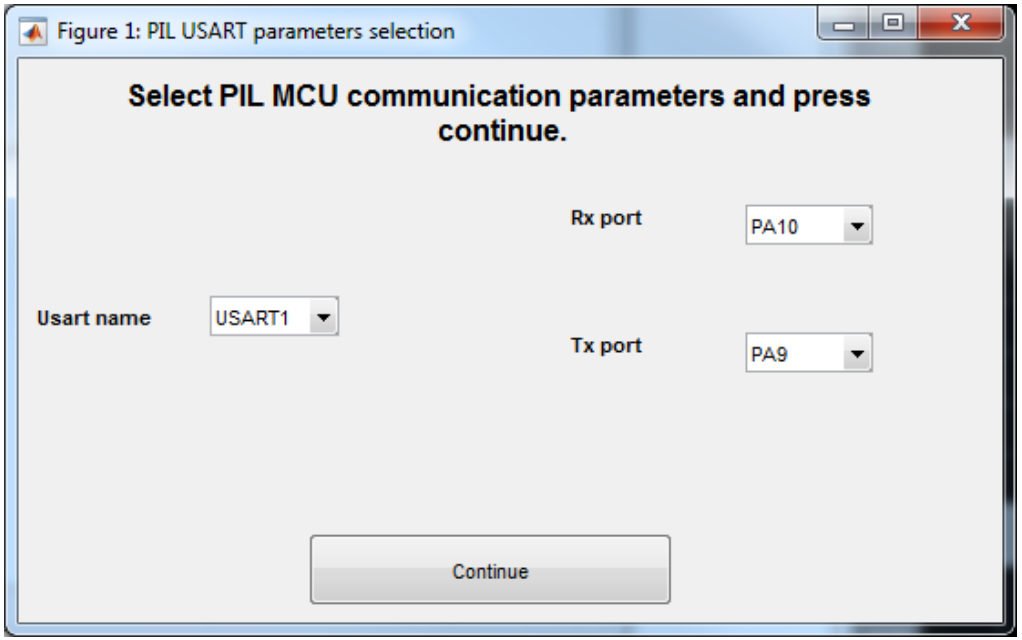
Frequency value is used for Profiling.
To compute STM32 processing time.



PIL Processing 3/5

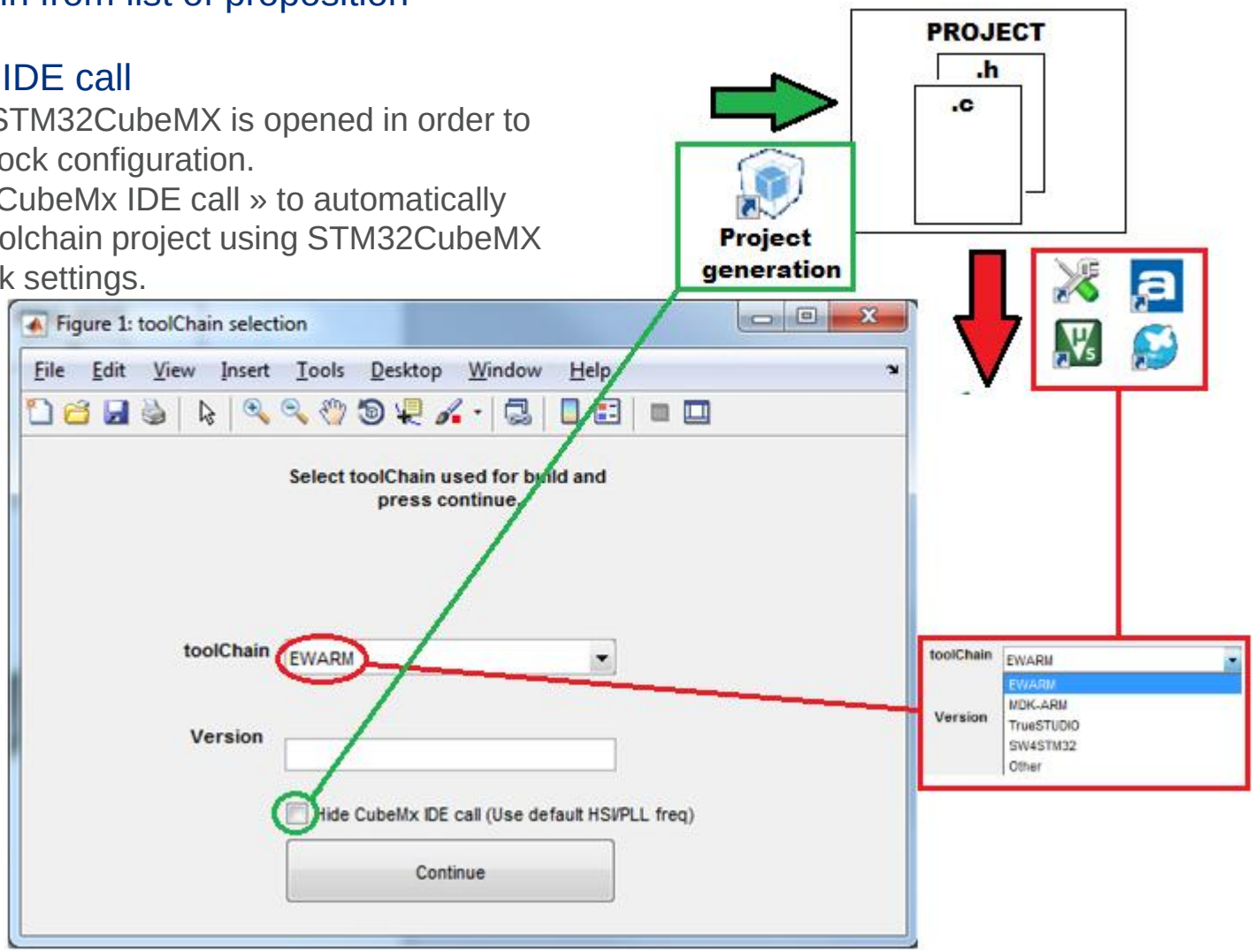
- STM32 COM Port
  - Example
    - USART1
    - Rx: PA10 & Tx: PA9





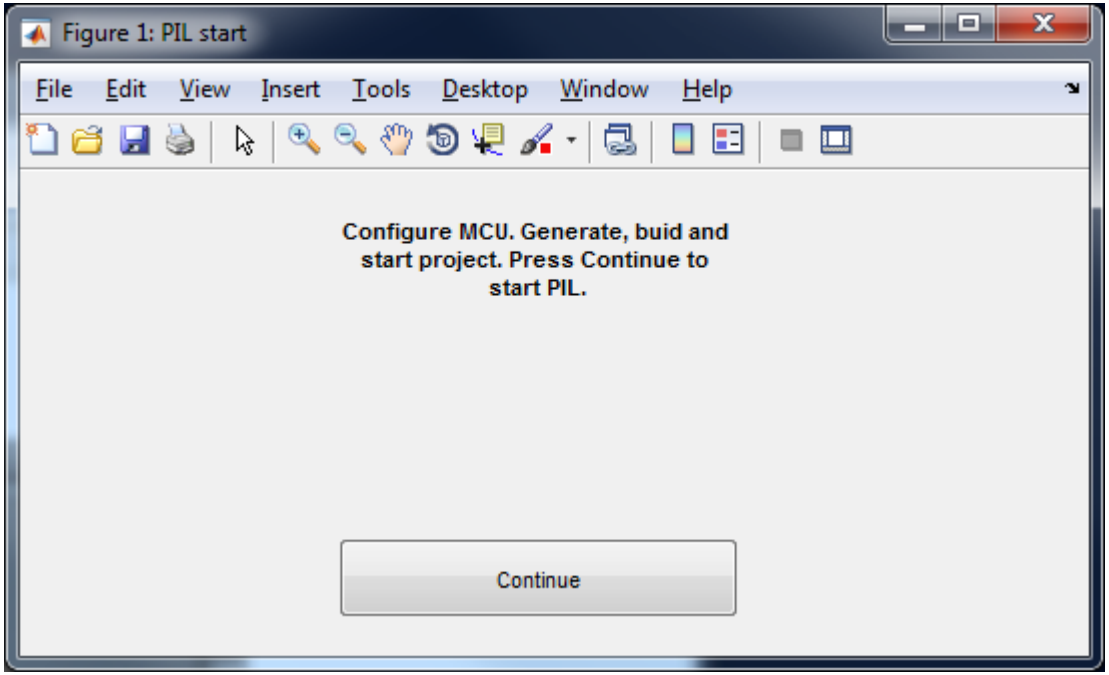
PIL Processing 4/5

- Toolchain Selection
  - Select toolchain from list of proposition
  - Hide CubeMx IDE call
    - By default STM32CubeMX is opened in order to set/verify clock configuration.
    - Set « Hide CubeMx IDE call » to automatically generate toolchain project using STM32CubeMX default clock settings.



PIL Processing 5/5

- It is an asynchronous process.
  - Simulink is waiting.
  - Simulink must send data through COM port, only when project is built and downloaded to STM32 target.
  - Then, you will press « Continue » button to start data flow PC/STM32.
  - STM32CubeMX is automatically opened and you can verify or modify STM32 settings.

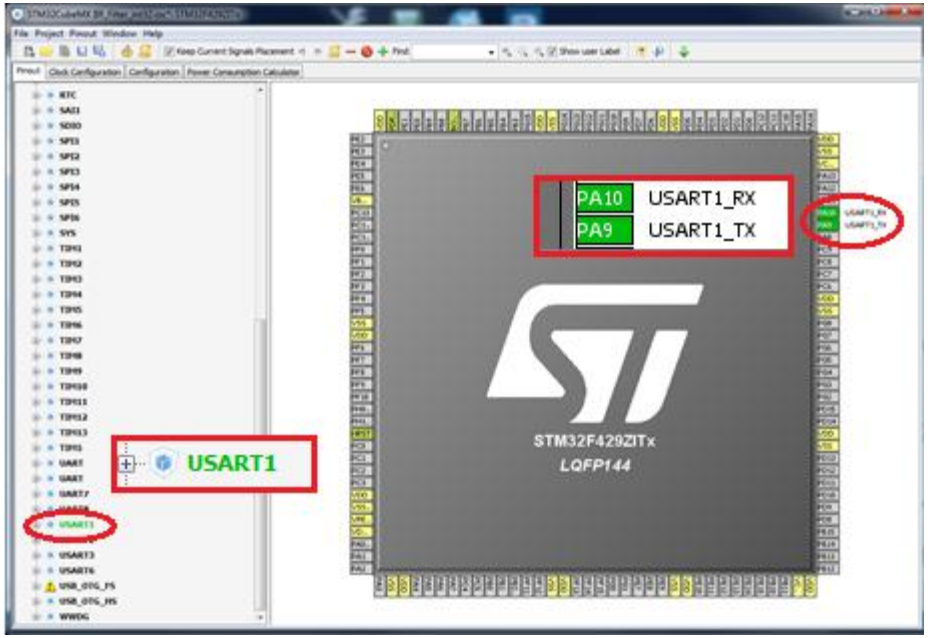


Look at STM32CubeMX User Manual to know how to deeply use it.

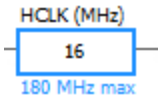
STM32CubeMX & PIL 1/2

- STM32CubeMX settings done
  - Pinout :
    - Usart1 , Rx/Tx
  - USART1 Configuration

Basic Parameters	
Baud Rate	115200 Bits/s
Word Length	8 Bits (including Parity)
Parity	None
Stop Bits	1



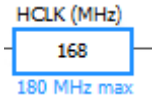
- STM32CubeMX settings to do
  - Clock Configuration
    - Default value



  - Reference value already selected for Profiling

Max frequency value (Hz)
168000000

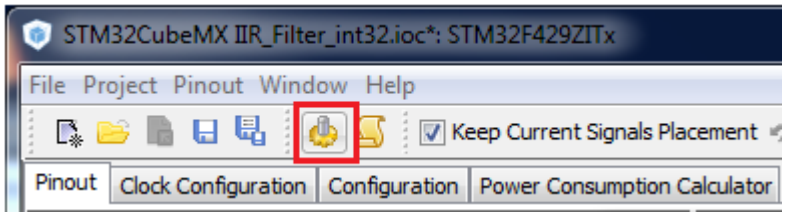
  - Clock setting : Enter 168 and return
    - Then Clock tree is automically updated



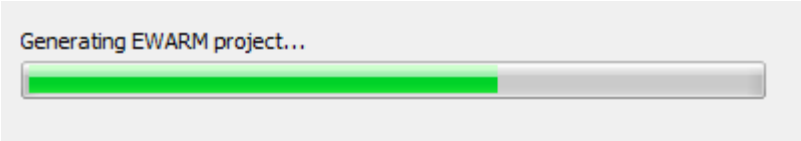
life.augmented

STM32CubeMX & PIL 2/2

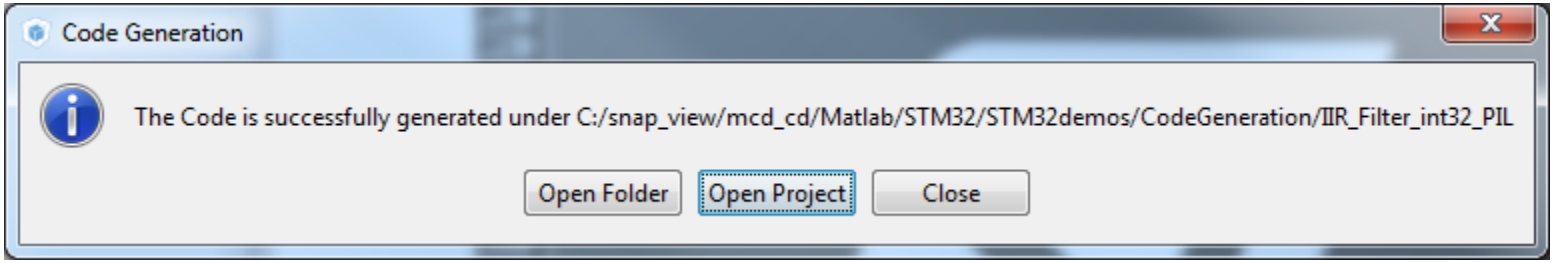
- STM32CubeMX project generation
  - Generate source code based on user settings
    - Press Project Settings « OK »



  - Generate Project



  - Open Project

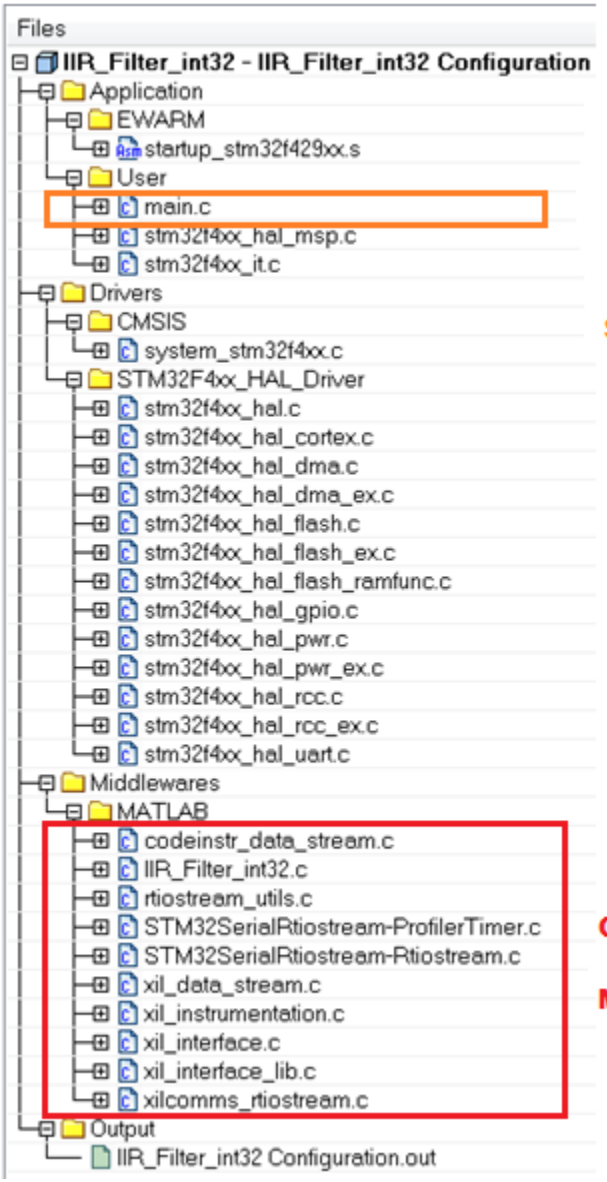


  - Close STM32CubeMX

Look at toolchain User Manual to know how to deeply use it.

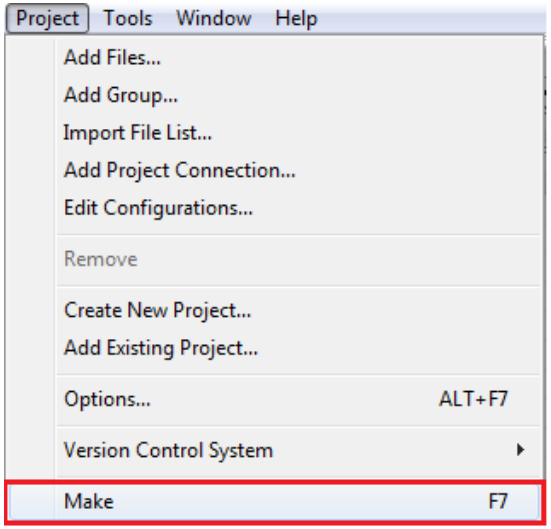
Toolchain & PIL 1/2

- Toolchain project
  - Generated from STM32CubeMX
  - Includes
    - Application files generated from MATLAB®
    - main.c generated from MATLAB® and modified by STM32CubeMX
    - HAL mandatory peripherals drivers



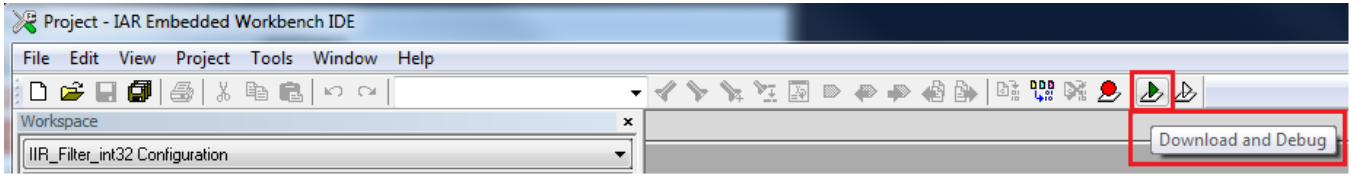
Toolchain & PIL 2/2

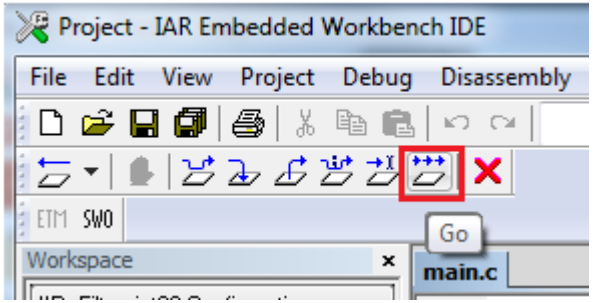
- Build Toolchain project
  - Make (F7)



IAR EWARM toolchain used for this example

- Download project

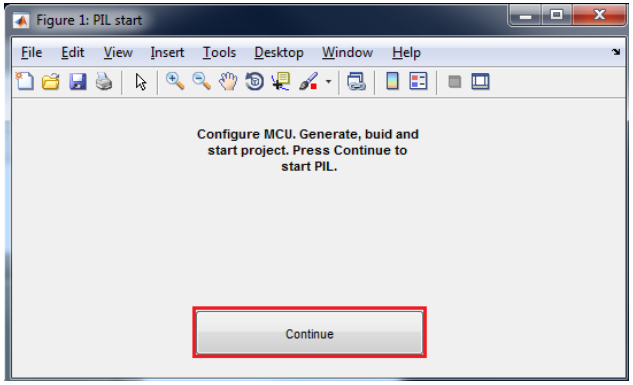




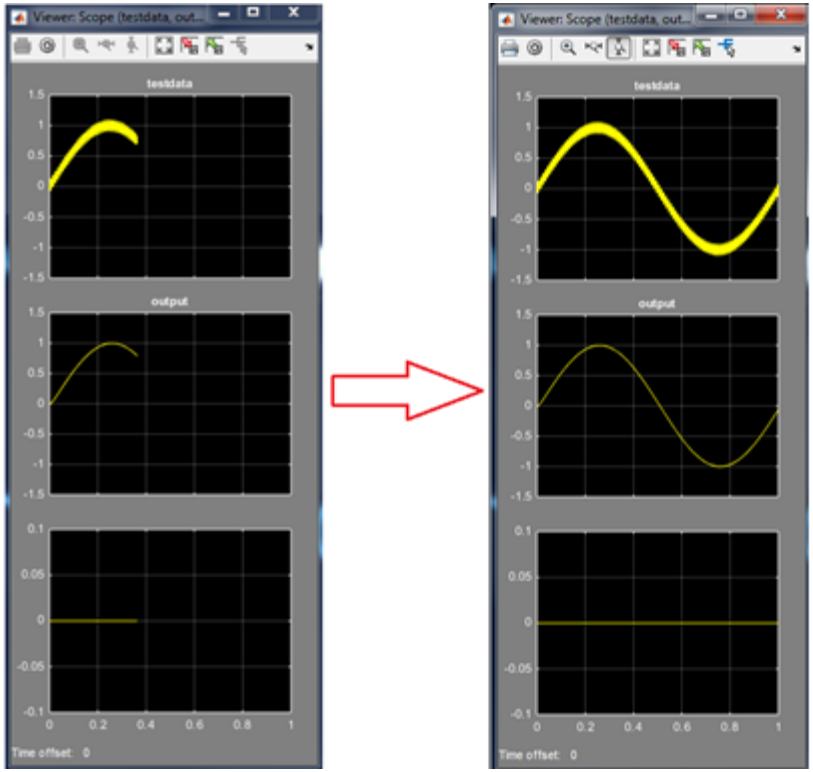
- Run project
  - Then, STM32 is running and waits for data from Simulink.

Running PIL

- Start Simulink data flow
  - « Continue » button of PIL start window



- PIL results



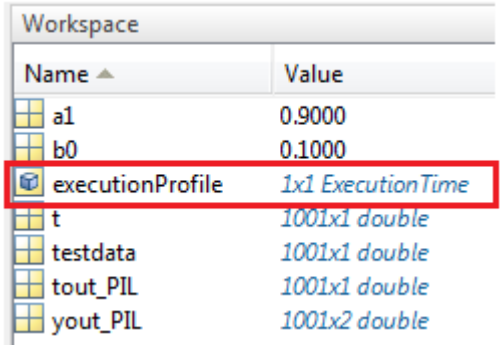
Input signal

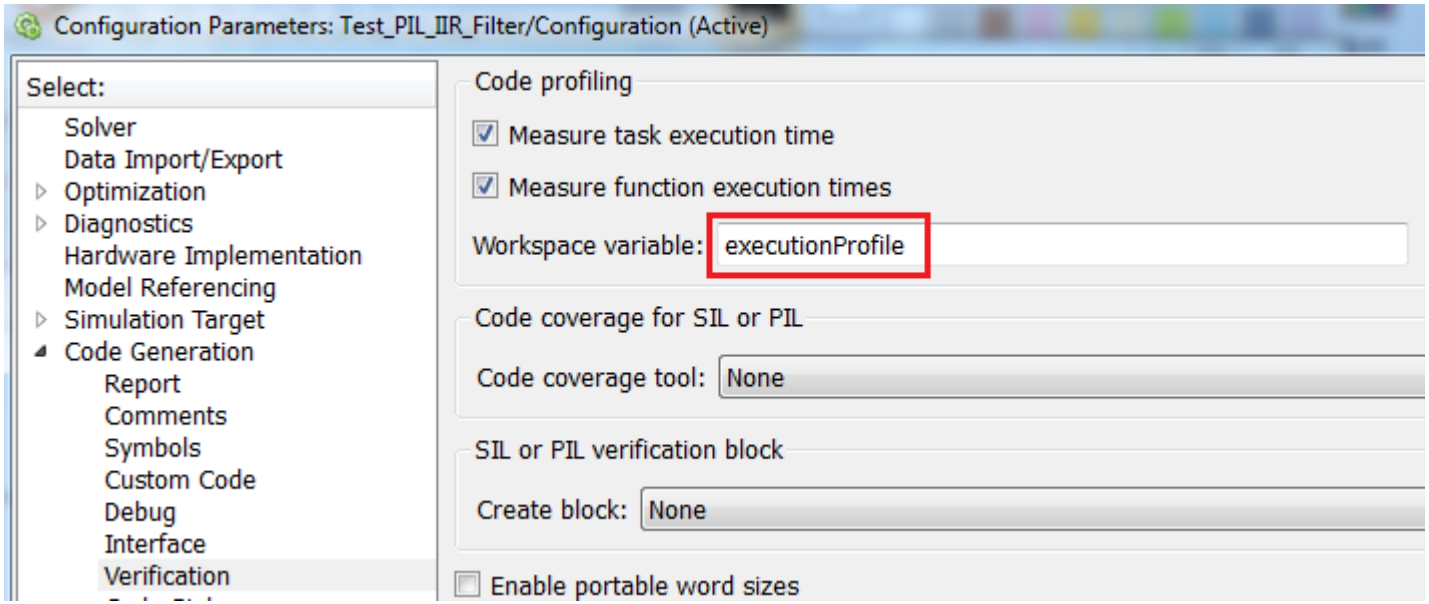
STM32 filtered signal

Difference between PC & STM32 filters

PIL Profiling 1/3

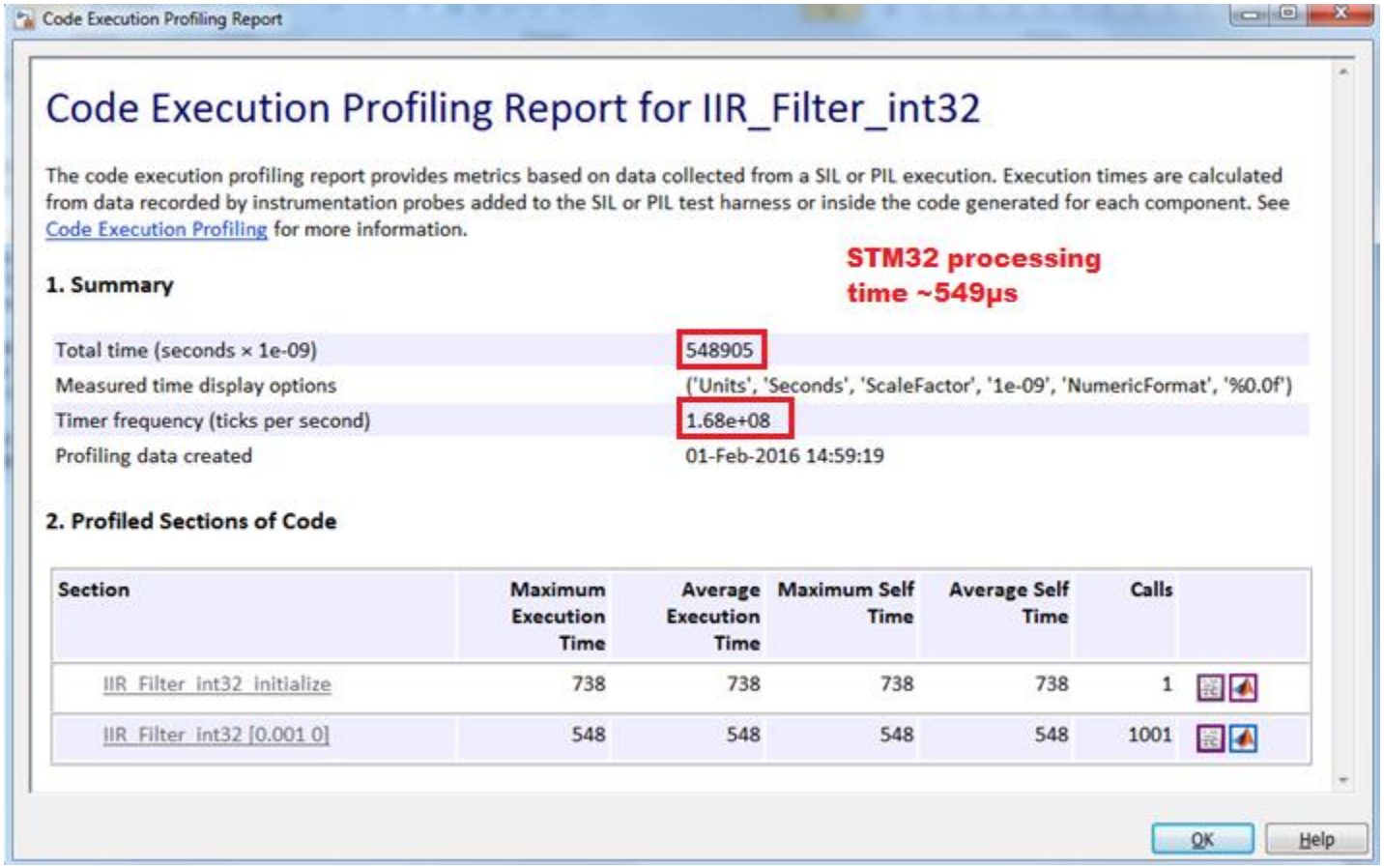
- Profiling = PIL Timing analysis
  - executionProfile has been generated in MATLAB® Workspace
  - « executionProfile » default Workspace variable name
    - Can be modified from « Configuration Parameters » window Code Generation > Verification of the model.





PIL Profiling 2/3

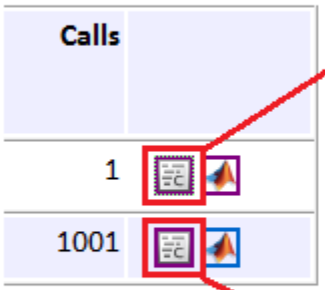
- Enter «executionProfile.report» on MATLAB® command window to open profiling report window to know
  - STM32 processing time at selected frequency
  - Execution time and number of calls per function



life.augmented

PIL Profiling 3/3

- Click on C code generated
  - To see MATLAB® generated code for STM32
  - or profiled sections




```
125   /* Single In-the-Loop Component */
126   if (xilFcnId == 0) {
127     taskTimeStart_IIR_Filter_int32(1U);
128     IIR_Filter_int32_initialize(rt_errorStatus, &(RTModel), &(localDW));
129     taskTimeEnd_IIR_Filter_int32(1U);
130   } else {
```

```
202   switch (xilTID) {
203    case 1:
204     taskTimeStart_IIR_Filter_int32(2U);
205     IIR_Filter_int32(&(i_Input), &(o_Output), &(localDW));
206     taskTimeEnd_IIR_Filter_int32(2U);
207     break;
```

Application Code Generation

Tools usage

- Step 1: Pure simulation
  - Everything done on the PC

MATLAB®/ Simulink

- Step 2: Processor-in-the-loop (PIL)
  - Algorithm fully executed on STM32
  - Data (Input or output) exchanged between MATLAB®/Simulink and STM32 via UART

MATLAB®/ Simulink

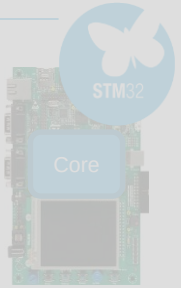
Embedded Coder

STM32Cube Embedded Software

STM32CubeMX

*: used only for UART

Toolchain



- Step 3: Everything on STM32
  - Data (input or output) obtained within STM32 through its peripherals (ADC, Timers, …) and algorithm fully executed on STM32

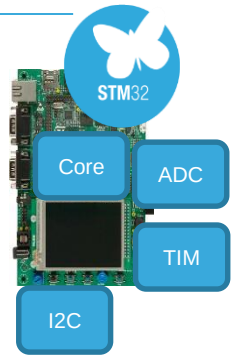
MATLAB®/ Simulink

Embedded Coder

STM32Cube Embedded Software

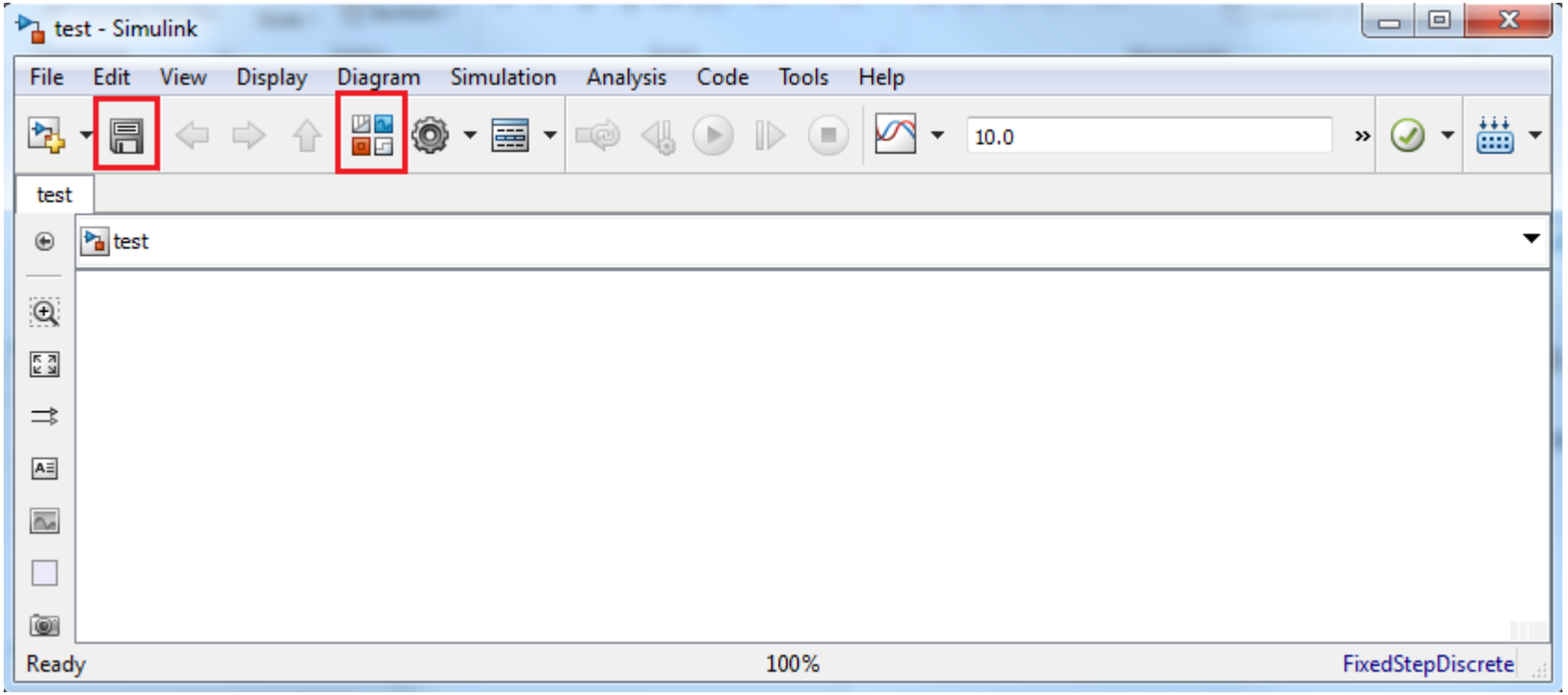
STM32CubeMX

Toolchain



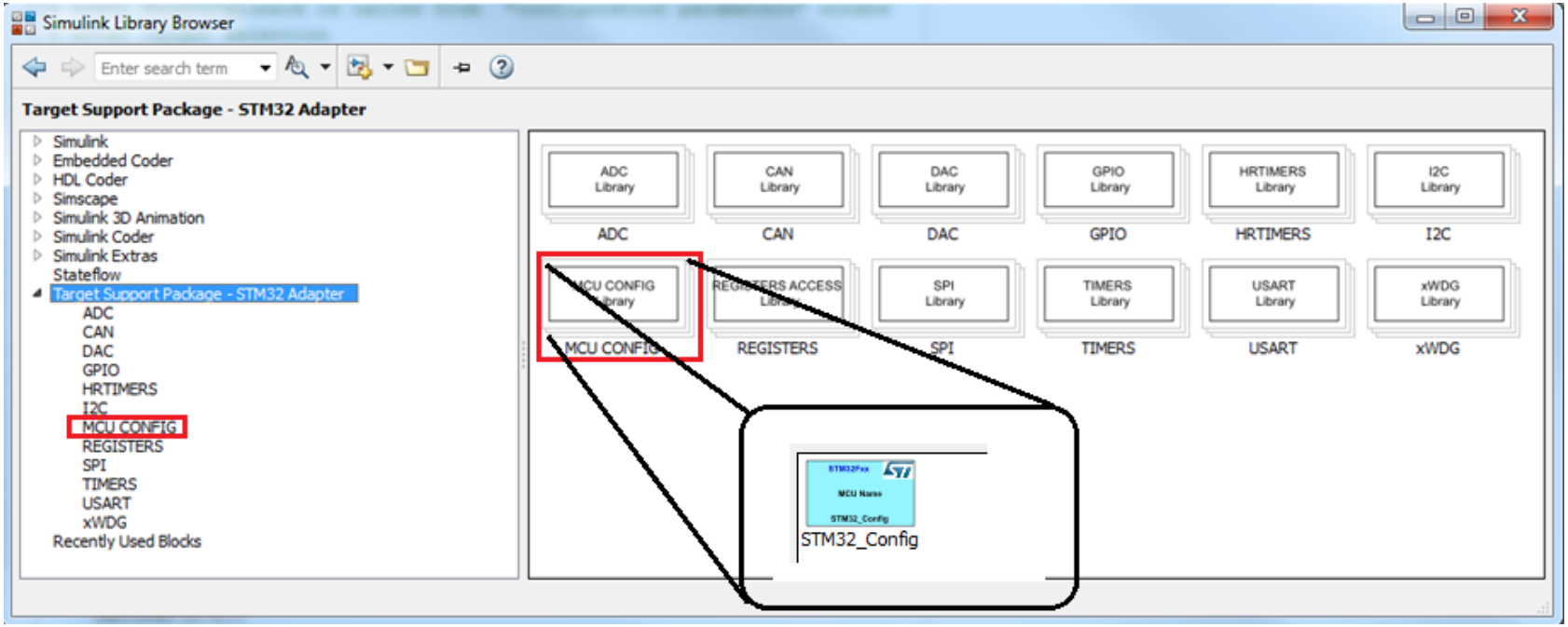
Simulink application development

- Look at Simulink model Setting 1/4 to 4/4 to open and configure new model.
- Save Simulink model and open Library Browser
  - For example: Save model as test.slx into C:\TEMP\test repository



STM32 Configuration 1/3

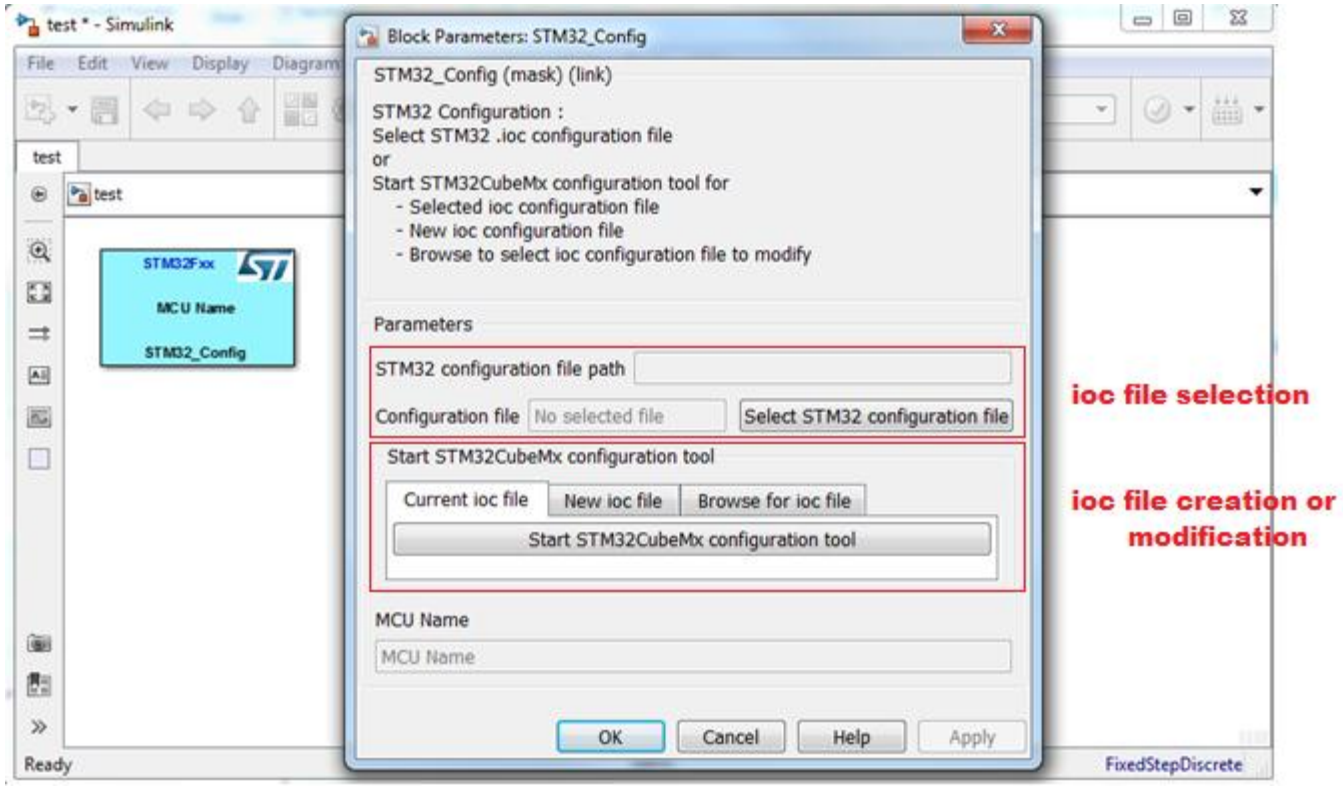
- STM32 Drivers Library
  - Several STM32 peripheral drivers are available.
- MCU Configuration
  - « MCU CONFIG » is the first library to open and use
  - Drag&Drop STM32_Config to your model.
  - STM32_Config is used to select STM32 configuration through STM32CubeMX configuration ioc file.



STM32 Configuration 2/3

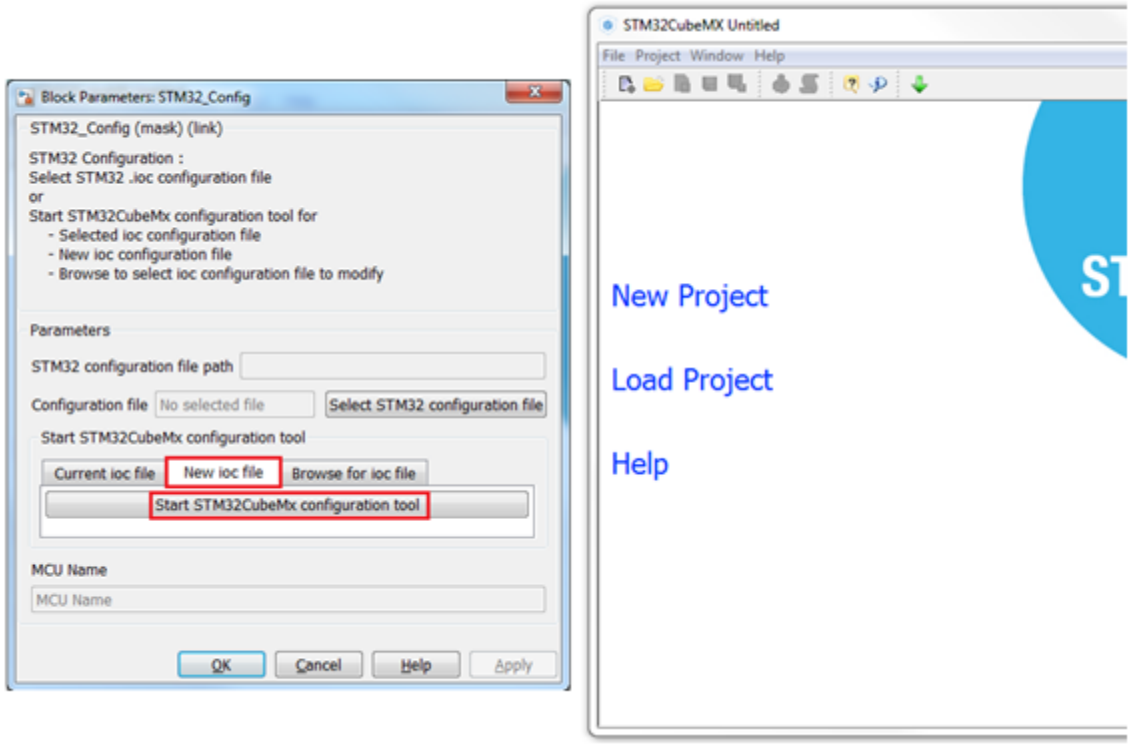
- IOC file selection
  - .ioc file is a text file created and used by STM32CubeMX as STM32 descriptor.
  - .ioc file contains STM32 configuration (pinout, peripheral selection & configuration)
  - Double click STM32_Config to open block parameters window

- Select ioc file describing STM32 you want to use for your application with « Select STM32 configuration file »
  - It is the only and mandatory way to select .ioc file for Simulink application.
- You can also modify ioc file or create a new one and STM32CubeMX is automatically opened.



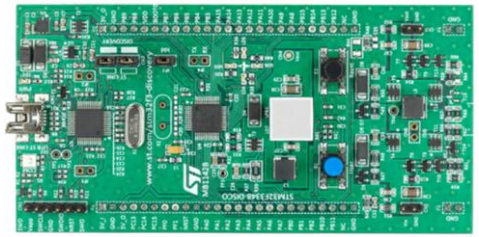
STM32 Configuration 3/3

- IOC file creation
  - Select « New ioc file » and click « Start STM32CubeMX configuration tool » button.
  - STM32CubeMX opens.
  - Look at http://www.st.com/web/catalog/tools/FM147/CL1794/SC961/SS1533/PF259242?s_searchtype=partnumber to get STM32CubeMX User Manual.
  - Save STM32CubeMX ioc configuration file and select it using « Select STM32 configuration file » button.
    - ioc file is not automatically selected for Simulink application from STM32CubeMX



Simulink application example

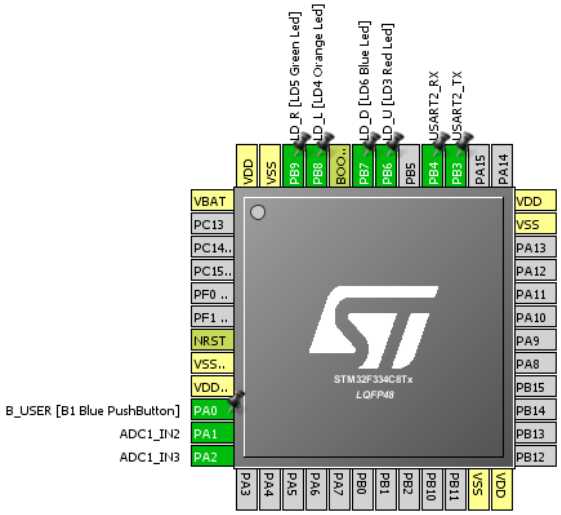
- Hardware :
  - Example based on STM32F3348-DISCO
  - Configuration :
    - Leds (LED3/4/5/6)
    - Push Button (User blue button)
    - USART2 Virtual Com Port (SB14&SB16 soldered)
    - ADC1
    - TIM1 & TIM6



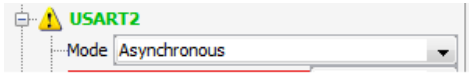
- Software application :
  - Use TIM1 to blink LED3 at 1Hz
  - Use TIM6 to blink LED4 at 2Hz
  - Use TIM6 to trig ADC1 channels 2&3 conversion
  - Blink Led6 when user push button is pressed
  - Send ADC1 channel 3 values on USART2 when user push button is pressed

STM32CubeMX STM32F3348 Pinout

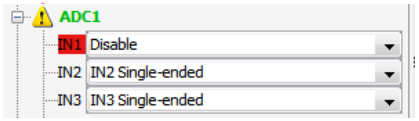
- Hardware pinout configuration
  - PA0 : GPIO_EXTI0
  - PA1 : ADC1_IN1
  - PA2 : ADC1_IN2
  - PB3 : Usart2_Tx
  - PB4 : Usart2_Rx
  - PB6 to PB9 : GPIO_Output



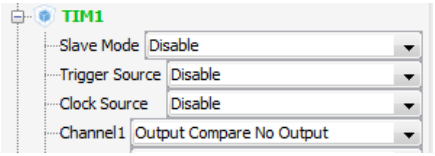
- Hardware setting
  - USART2 is Asynchronous



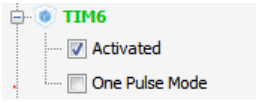
  - ADC1 IN2 & IN3 Single-ended



  - TIM1 Channel1 as Ouput Compare No output

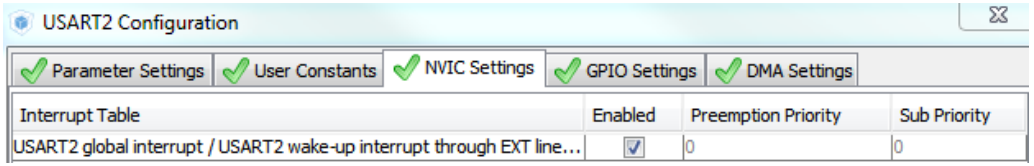


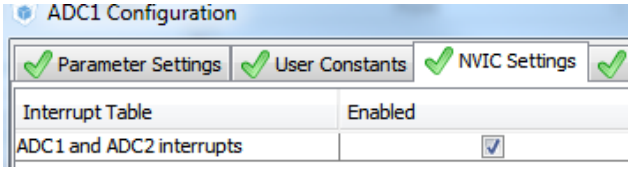
  - TIM6 Activated (No Output)

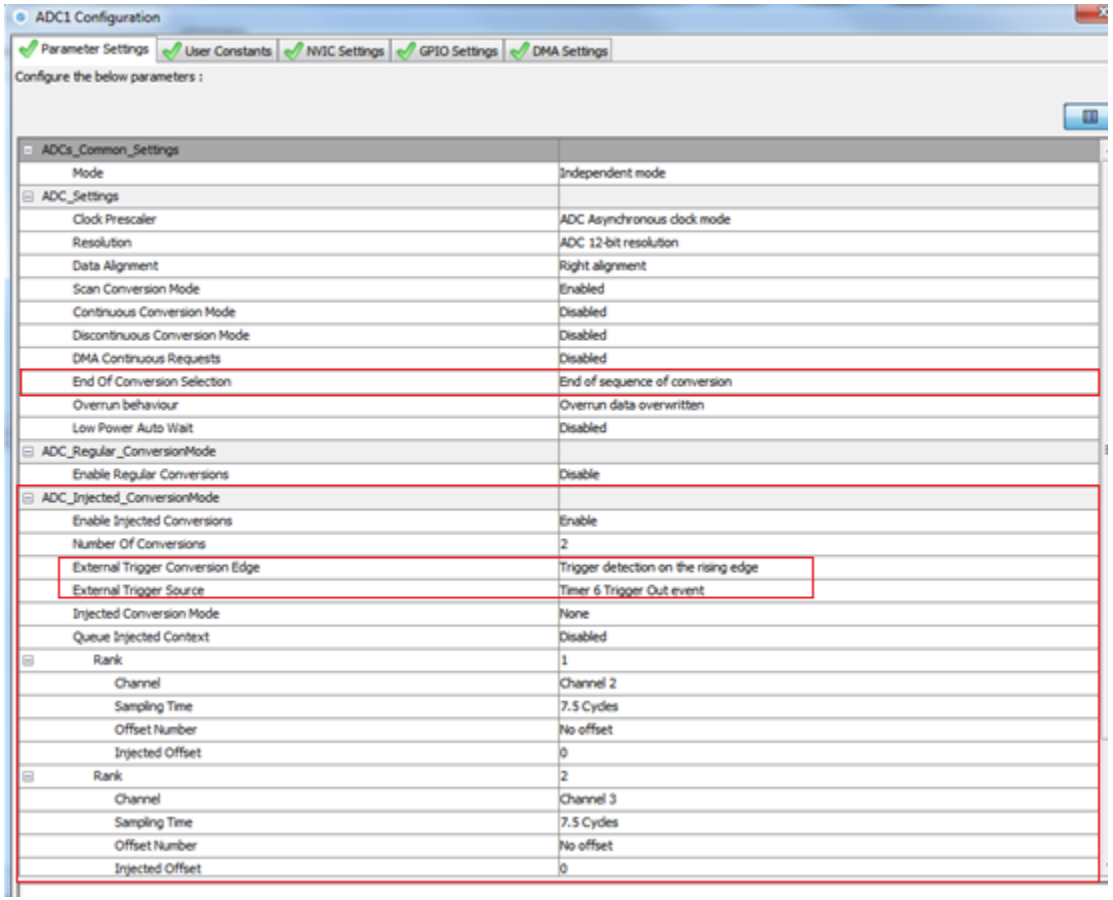


STM32CubeMX Peripheral settings 1/2

- Peripheral configuration :
  - USART2
    - Baud Rate : 115200
    - Word Length: 8 Bits
    - Parity: None
    - Stop Bits: 1
    - Enable global interrupt
  - ADC1
    - Injected Channels 2&3
    - Interrupt at end of sequence of conversion
    - Conversion triggered from Timer6
    - Interrupt Enabled

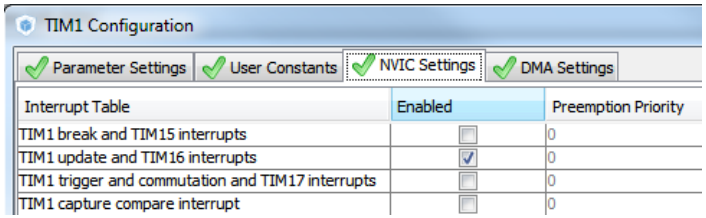




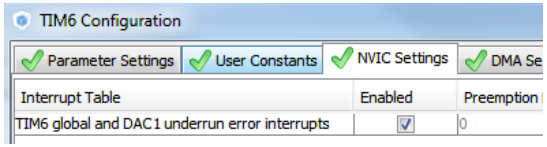


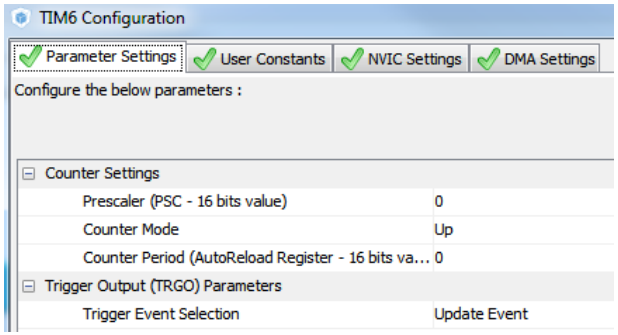
STM32CubeMX Peripheral settings 2/2

- Peripheral configuration :
  - TIM1
    - Default configuration
    - TIM1 Update interrupt enabled

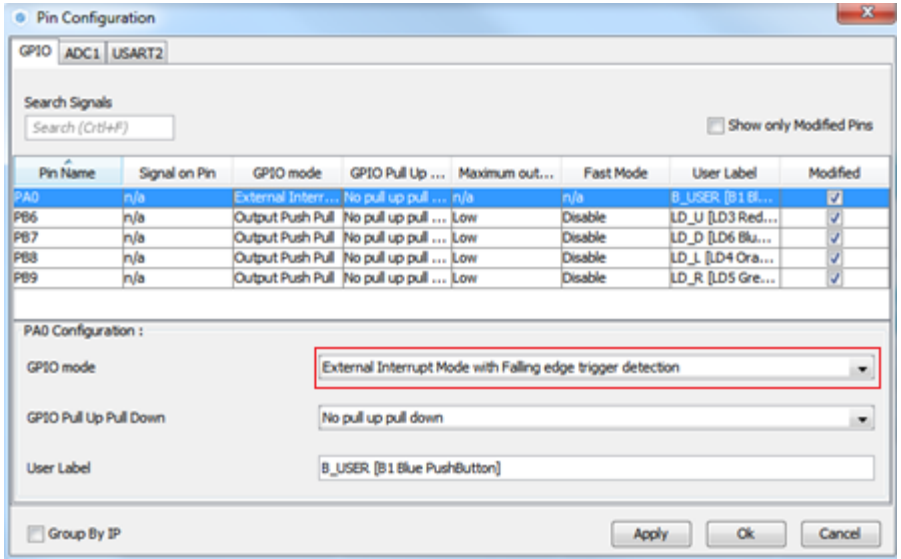


  - TIM6
    - Trigger event :Update Event
    - TIM6 global interrupt enabled



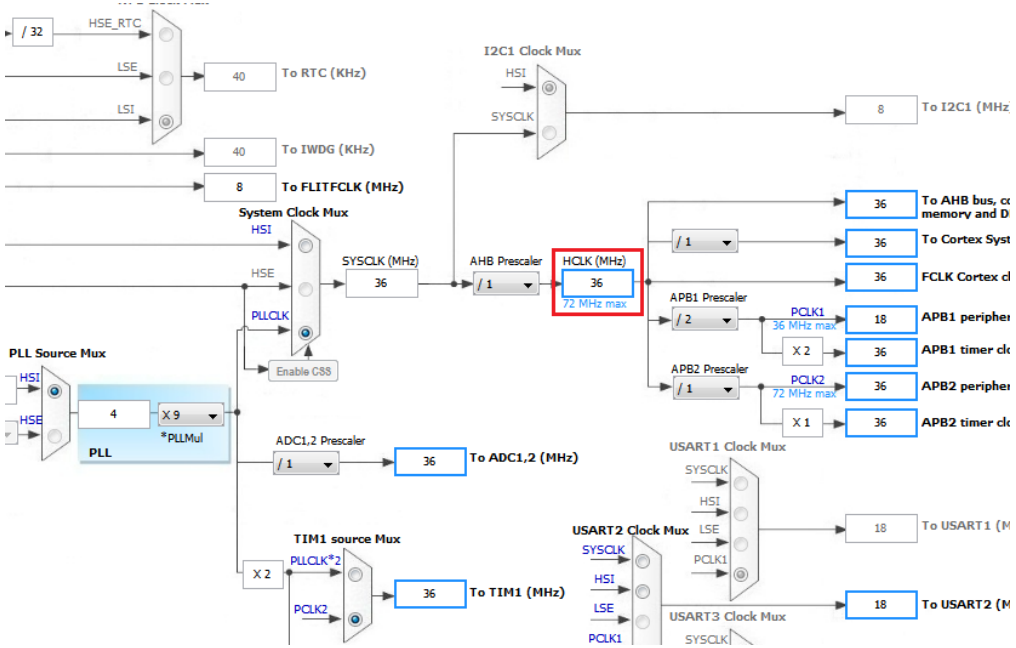


  - GPIO External interrupt
    - External Interrupt Mode with Falling edge trigger detection



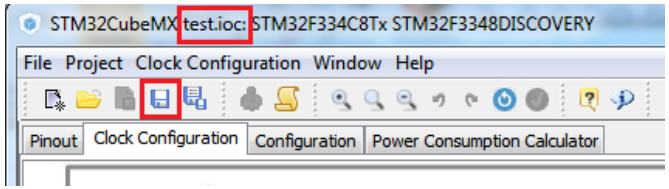
STM32CubeMX Clock Configuration

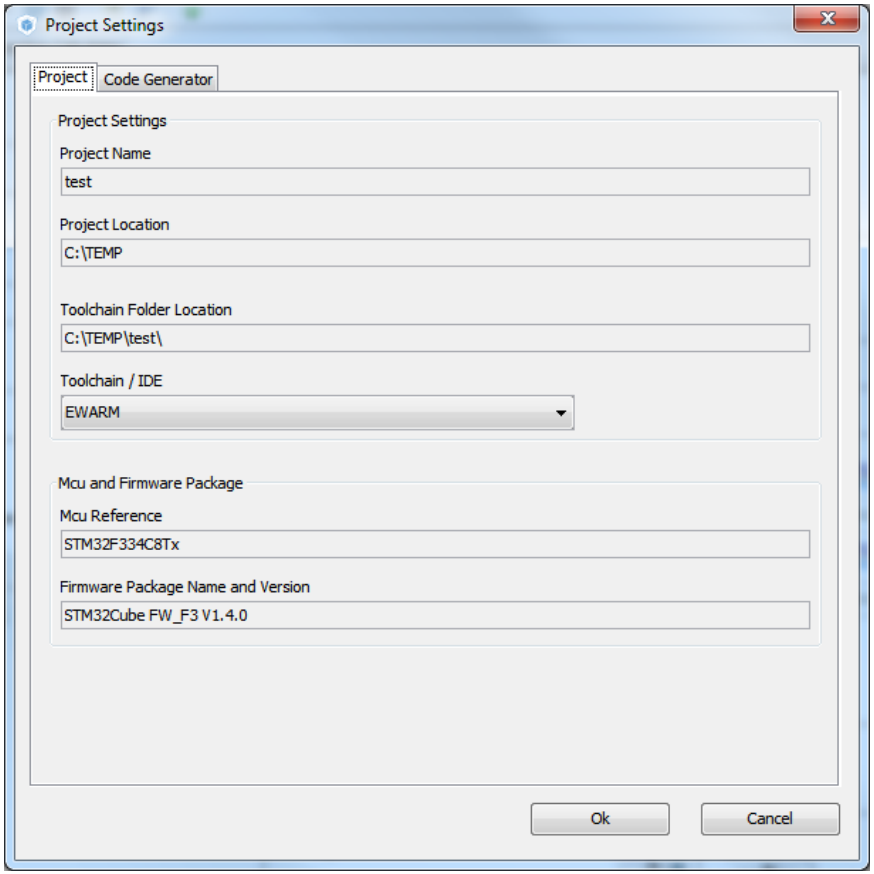
- Clock Default Configuration:
  - 16 MHz
- Modification not mandatory
  - Can be 36 MHz for example



STM32CubeMX project Settings

- Project Name:
  - «test» for this example
- Project Location :
  - C:\TEMP for this example
- Save the current project
  - test.ioc file is available from c:\TEMP\test repository





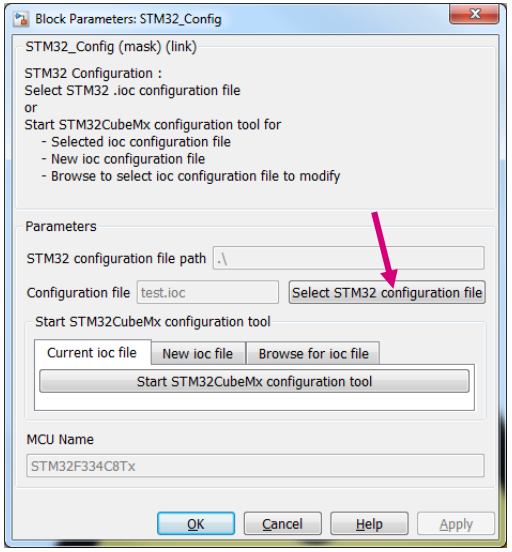
- You can save ioc file anywhere.
- It is preferable to save ioc file in same repository as model that will use it. Repository and ioc file must have same name.

life.augmented

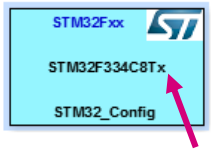
Simulink application IOC file selection

- IOC file selection
  - Ioc file has been created and can be selected
  - Open (double click) STM32_Config block parameters window
  - STM32 configuration path is related to model
    - .\ because test.ioc and test.slx in same repository

Double click



«Select STM32 configuration file »
Browse and select test.ioc file



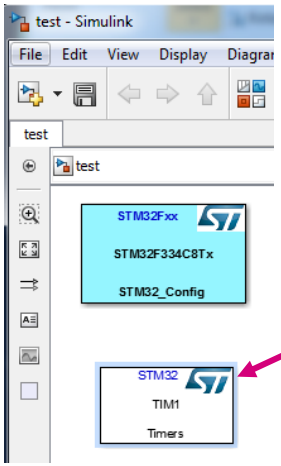
Simulink application for
STM32F334C8Tx MCU

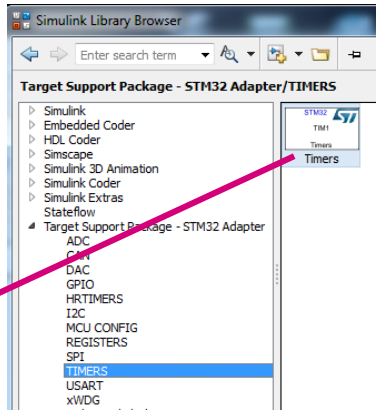
USE TIM1 to Blink LED3 at 1Hz

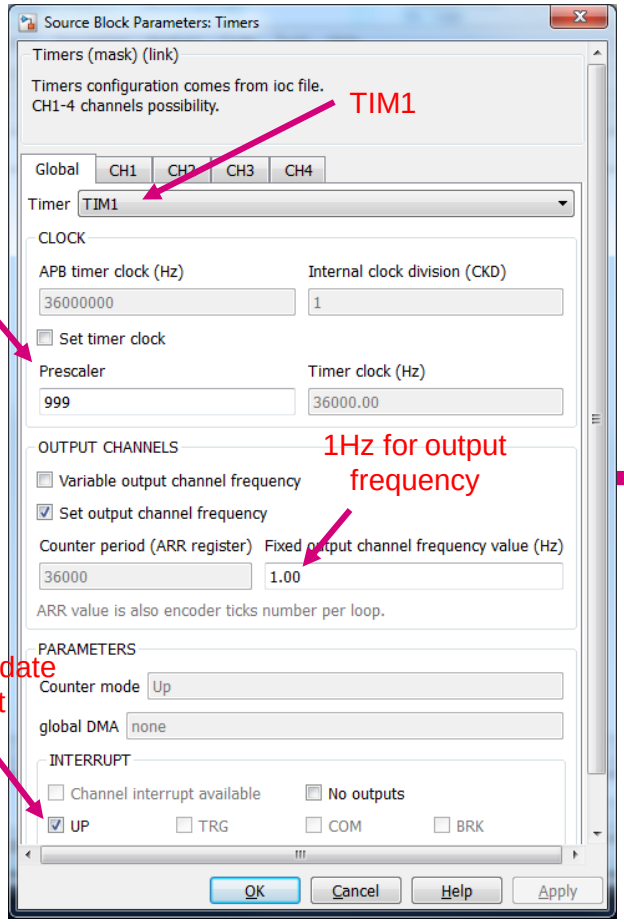
- Software application example:
  - **Use TIM1 to blink LED3 at 1Hz**
  - Use TIM6 to blink LED4 at 2Hz
  - Use TIM6 to trig ADC1 channels 2&3 conversion
  - Blink Led6 when user push button is pressed
  - Send ADC1 channel 3 values on USART2 when user push button is pressed

TIM1 Selection & Configuration

- TIM1 Selection
  - Drag&Drop Timers block from Simulink Library Browser
- TIM1 Configuration
  - Open (double click) Timers block parameters window
  - Select TIM1 and set parameters



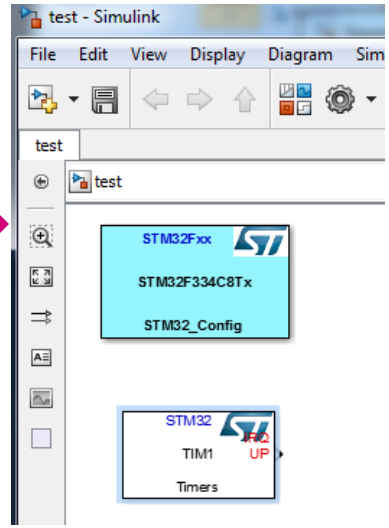




Prescaler or Timer Clock

1Hz for output frequency

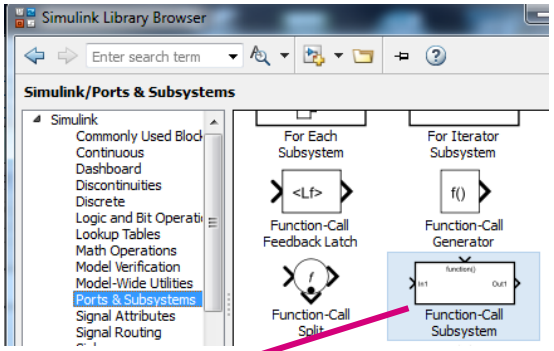
Validate Update interrupt

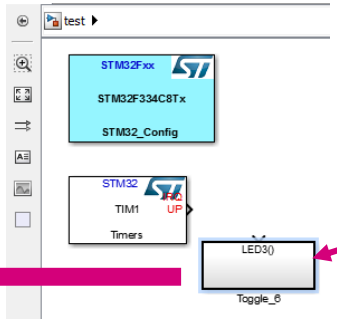


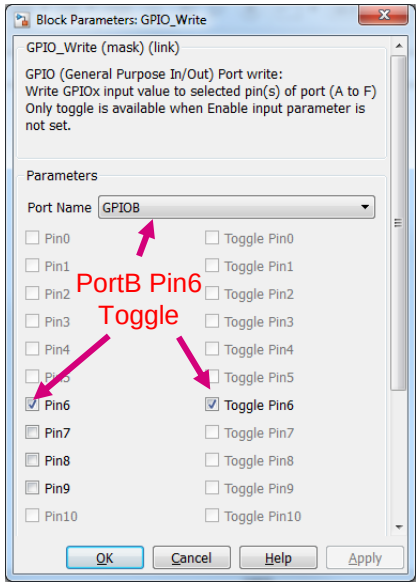
life.augmented

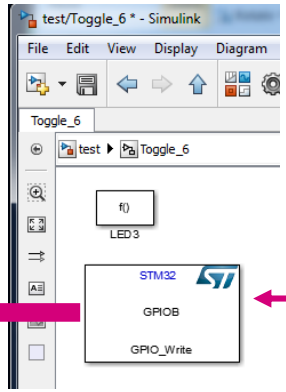
TIM1 Application

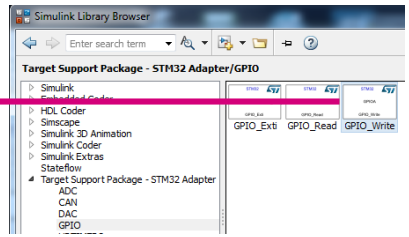
- TIM1 toggle LED3 at 1Hz
  - Drag&Drop Function-Call Subsystem from Simulink Library Browser (Simulink>Ports & Subsystems)
  - Open (double click) Function-Call block and Drag&Drop GPIO_Write block
  - Open (double click) GPIO_Write block parameters window and select Pin6 (LED3 is connected to Pin6)

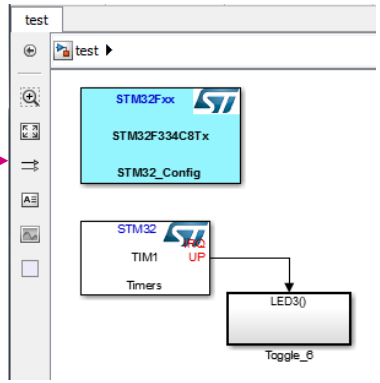












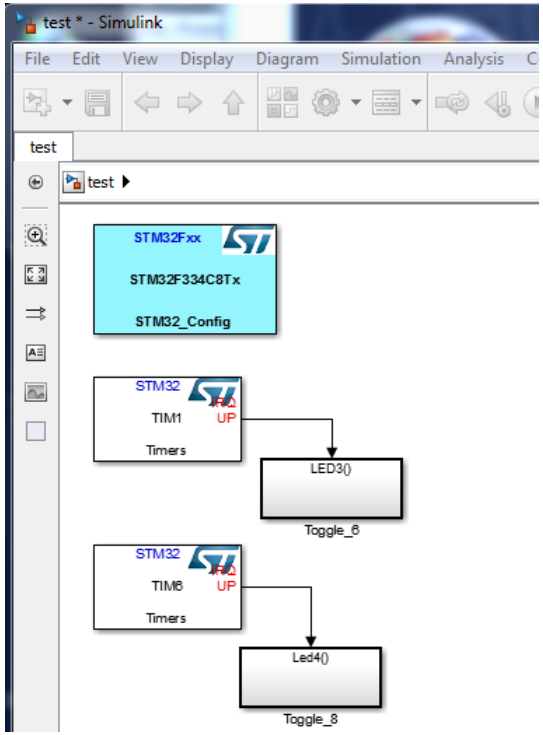
LED3 will blink when TIM1 update interrupt occurs. Every second at 1Hz

USE TIM6 to Blink LED4 at 2Hz

- Software application example:
  - Use TIM1 to blink LED3 at 1Hz
  - **Use TIM6 to blink LED4 at 2Hz**
  - Use TIM6 to trig ADC1 channels 2&3 conversion
  - Blink Led6 when user push button is pressed
  - Send ADC1 channel 3 values on USART2 when user push button is pressed

TIM6 Application

- TIM6 toggle LED4 at 2Hz
  - Make the same thing as for TIM1 but frequency is 2Hz and PortB Pin8 toggle as it is connected to Led4



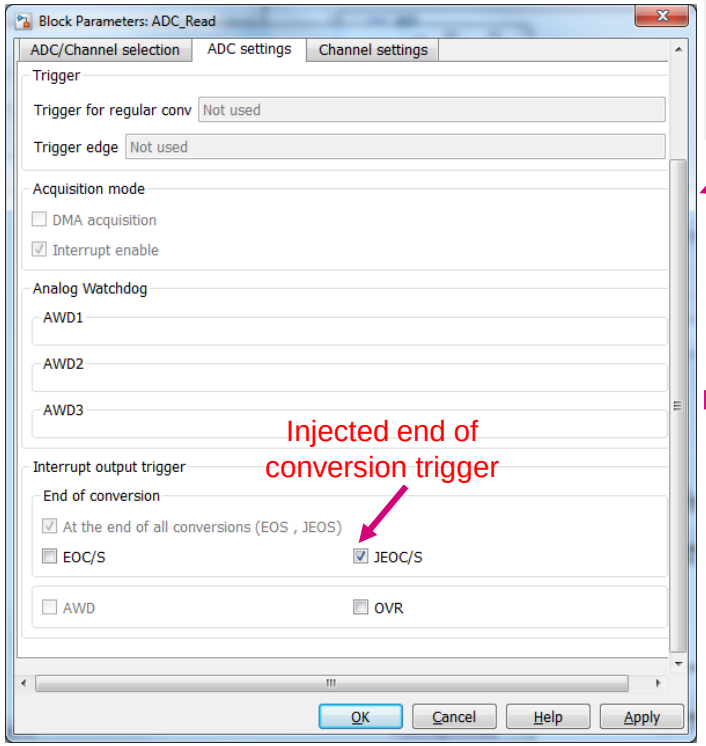
LED4 will blink when TIM6 update interrupt occurs. Every 0.5 second at 2Hz

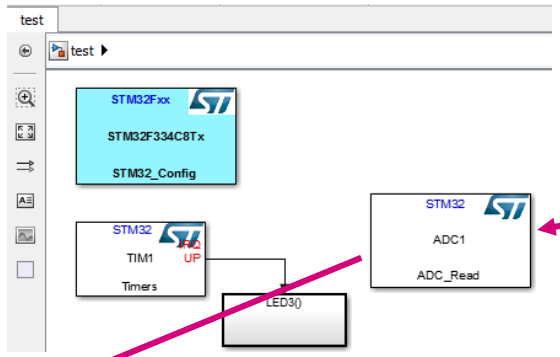
USE TIM6 to trig ADC1 channels 2&3

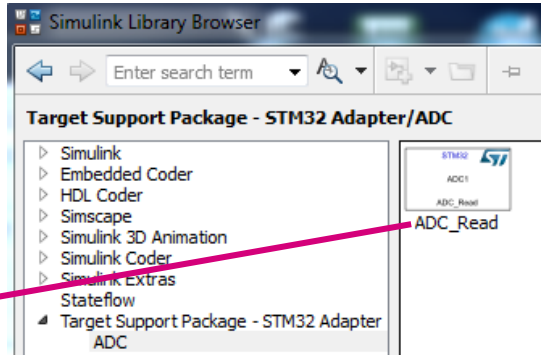
- Software application example:
  - Use TIM1 to blink LED3 at 1Hz
  - Use TIM6 to blink LED4 at 2Hz
  - **Use TIM6 to trig ADC1 channels 2&3 conversion**
  - Blink Led6 when user push button is pressed
  - Send ADC1 channel 3 values on USART2 when user push button is pressed

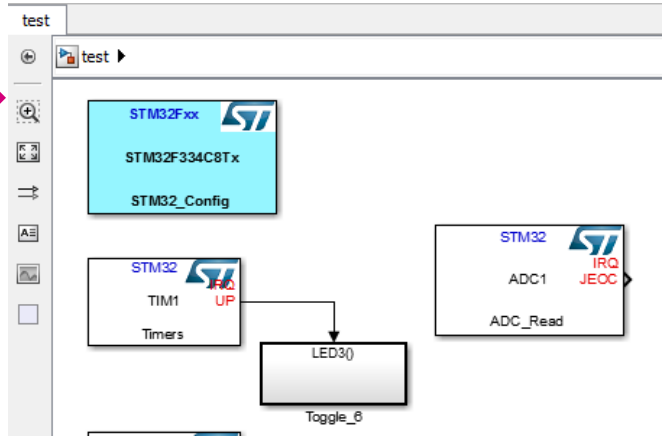
ADC1 Selection & Configuration

- ADC1 Selection
  - Drag&Drop ADC_Read block from Simulink Library Browser
- ADC1 Configuration
  - We don't need ADC values
  - Select JEOC/S as interrupt output trigger





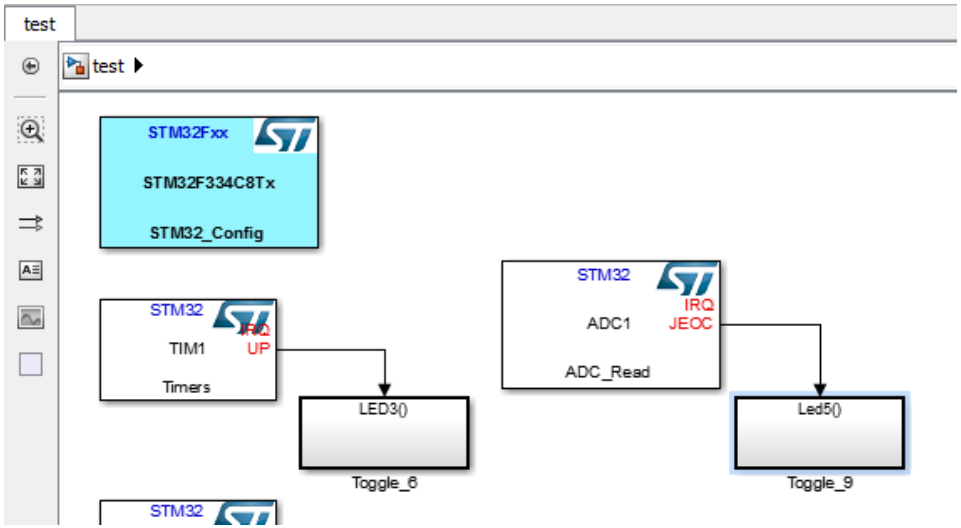




life.augmented

ADC1 Application

- TIM6 trig ADC1 channels conversion
  - Blink LED5 at end of ADC1 conversion to verify that TIM6 has triggerer it.
  - Drag&Drop Function-Call Subsystem from Simulink Library Browser (Simulink>Ports & Subsystems) and add GPIO_Write block in.
  - Set GPIO_Write block parameters window to toggle Pin9 (LED5 is connected to Pin9)



LED5 will blink when ADC1 injected channels 2&3 has been converted.

Start of Conversion is triggered from TIM6

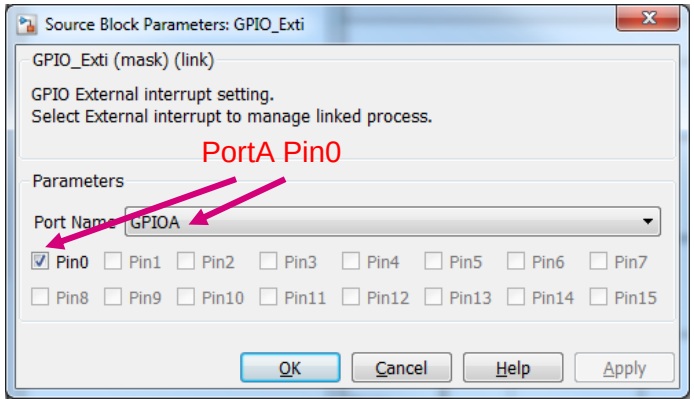
Channels 2&3 values are available at the end of conversion

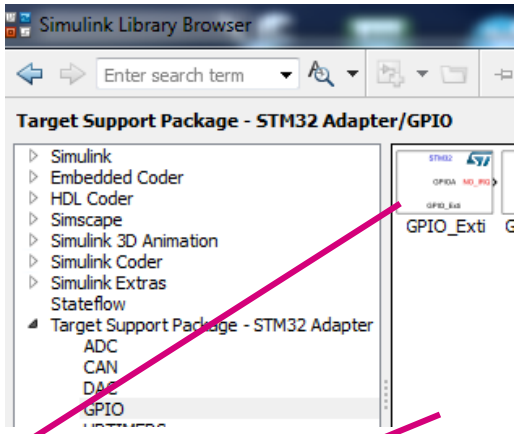
Push Button functions

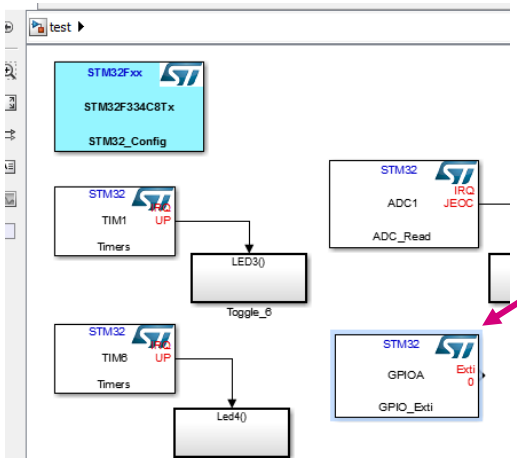
- Software application example:
  - Use TIM1 to blink LED3 at 1Hz
  - Use TIM6 to blink LED4 at 2Hz
  - Use TIM6 to trig ADC1 channels 2&3 conversion
  - **Blink Led6 when user push button is pressed**
  - **Send ADC1 channel 3 values on USART2 when user push button is pressed**

EXTI0 Selection & Configuration

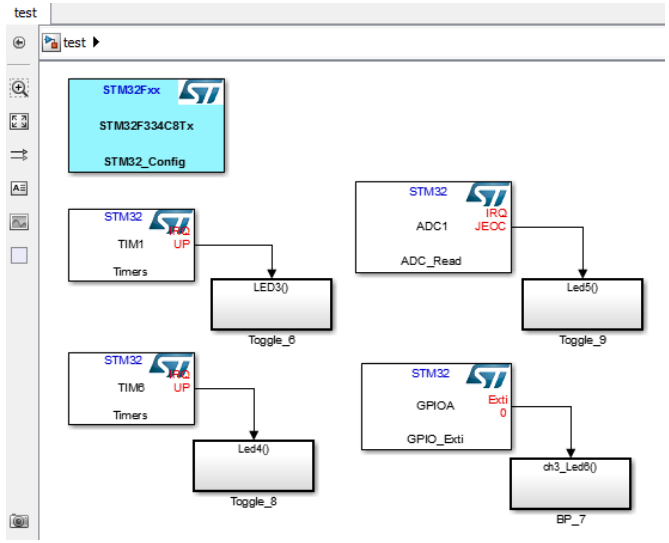
- EXTI0 Selection
  - Push Button is connected to External Interrupt 0 on PA0
  - Drag&Drop GPIO_Exti block from Simulink Library Browser
  - Open (double click) GPIO_Exti block parameters window and select GPIOA pin0







- Drag&Drop Function-Call Subsystem from Simulink Library Browser (Simulink>Ports & Subsystems) and connect it to Exti0 output



Every models in Function-Call connected to GPIO_Exti output will be executed for every action on push button.

We want to blink LED6 and send ADC1 ch3 value on USART2

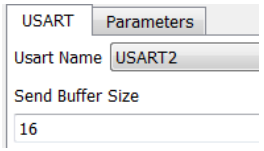
Push Button Action 1/2

- Blink LED6
  - Open (double click) Function-Call block and Drag&Drop GPIO_Write block
  - Open (double click) GPIO_Write block parameters window and select Pin7 (LED6 is connected to Pin7)

- Send ADC1 Ch3 value on USART2
  - Drag&Drop ADC_Read block from Simulink Library Browser
  - Drag&Drop USART_Send block from Simulink Library Browser

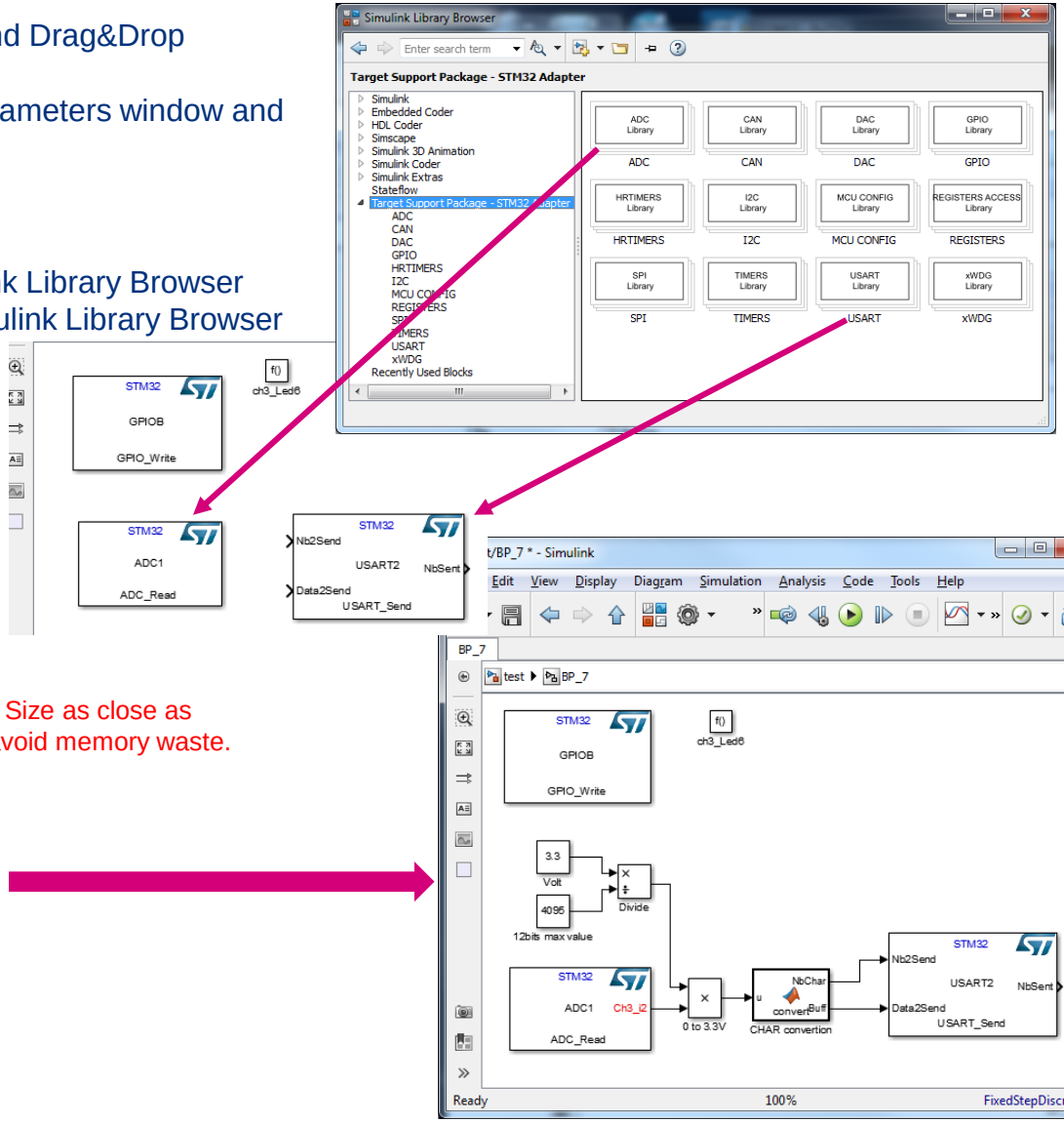
- ADC1 Settings
  - Open (double click) ADC_Read block parameters window and select ADC1 Ch3

- USART2 Settings
  - Open (double click) USART2_Send block parameters window and set buffer size.



It is mandatory to set Buffer Size as close as messages sent in order to avoid memory waste.

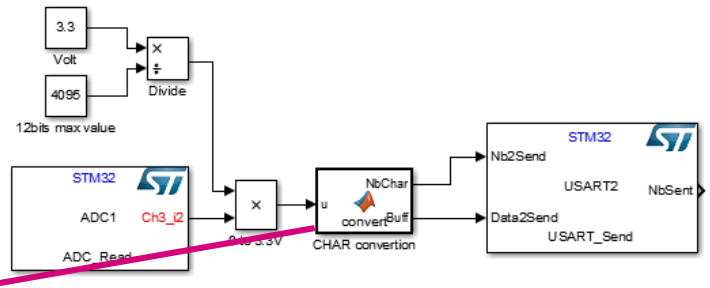
- Add signal processing.
- MATLAB® code can be added in « MATLAB Function » from Simulink Library Browser>User-Defined Functions>MATLAB Function



life.augmented

Push Button Action 2/2

- MATLAB Function
  - Add MATLAB® code to convert ADC1 Ch3 value to Buffer of characters
  - MATLAB® function is converted to C code and integrated to generated project during « build » process.



Function convert : 1 input and 2 outputs

BP_7/CHAR convertion

```
1   function [NbChar, Buff] = convert(u)
2   - if coder.target('Sfun')
3   -    % Executing in MATLAB, Buff is null
4   -    Buff = uint32(0);
5   -    NbChar = uint16(0);
6   - else
7   -    % Executing in the generated code.
8   -    buffer = zeros(1,15,'uint8');
9   -    coder.ceval('sprintf',coder.wref(buffer),['%2.2f',0],u);
10  -    tmp = uint8('Volt');
11  -    y = [buffer(1:4), tmp, 13, 10, 0];
12  -    Buff = coder.ceval('getBuffPtr',coder.ref(y));
13  -    NbChar = uint16(size(y,2));
14  - end
15  end
```

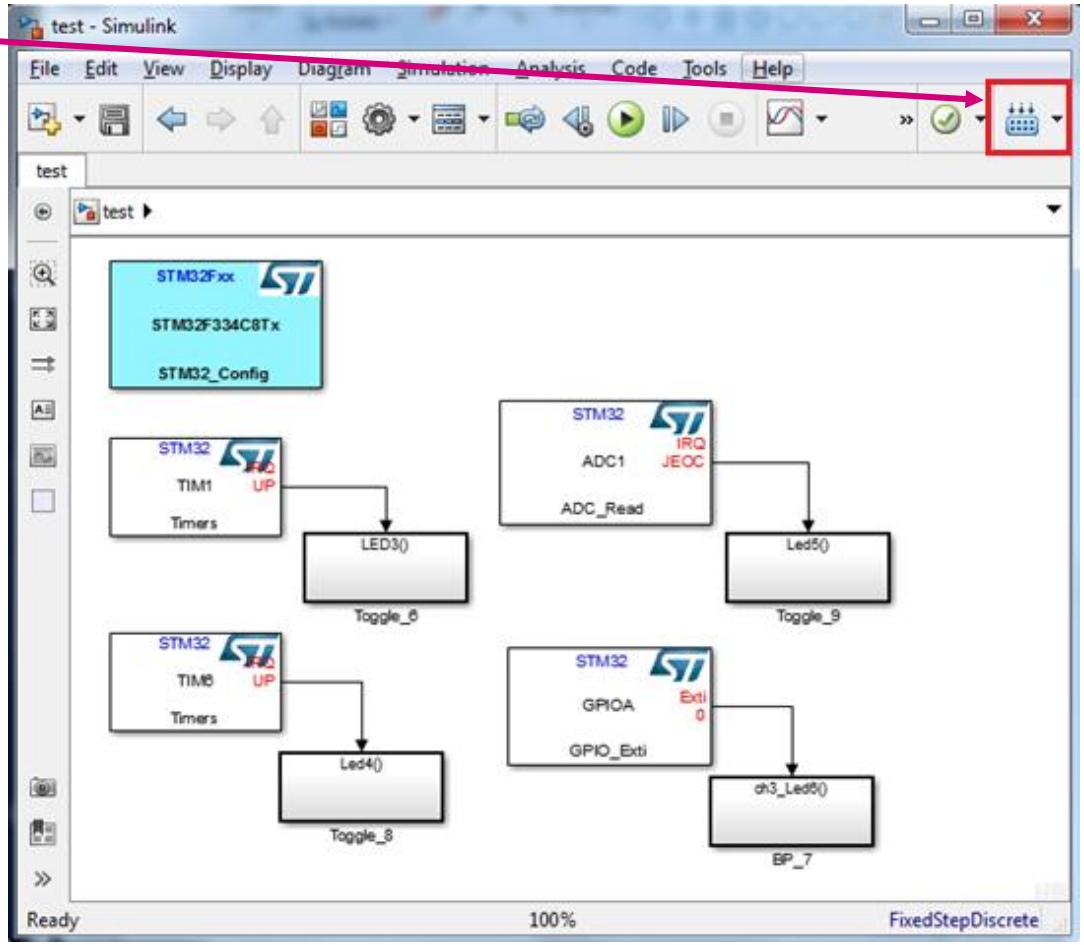
Nothing to do for Simulation

« xx.x Volt » Buffer to send throught USART2

«getBuffPtr» is a C function provided with STM32-MAT/TARGET that convert MATLAB® array to C pointer.

Build Application

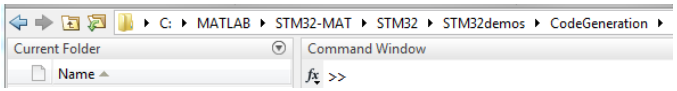
- Generate code for created application
  - Press « Build Model » to automatically generate C code and toolchain project.



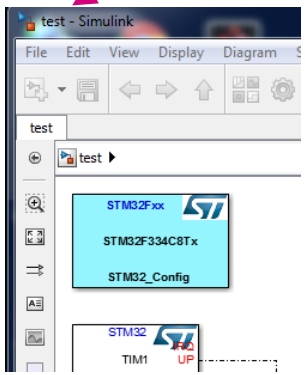
MATLAB® Code Generation 1/2

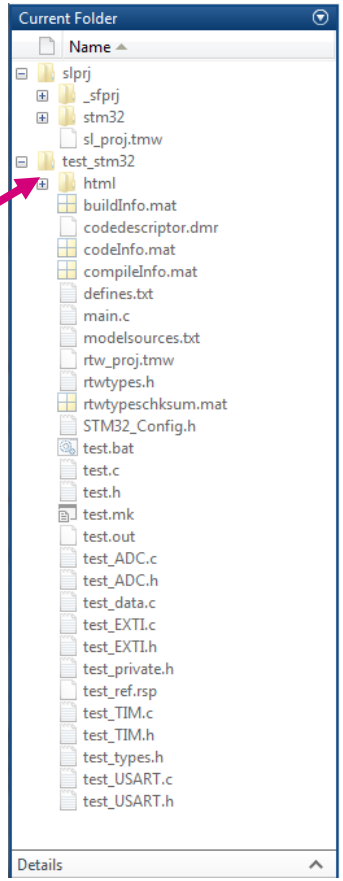
- Diagnostic Viewer
  - Diagnostic Viewer window appears when you click on « View diagnostics» at the bottom of built model. It gives « Code Generation » information about Build process.

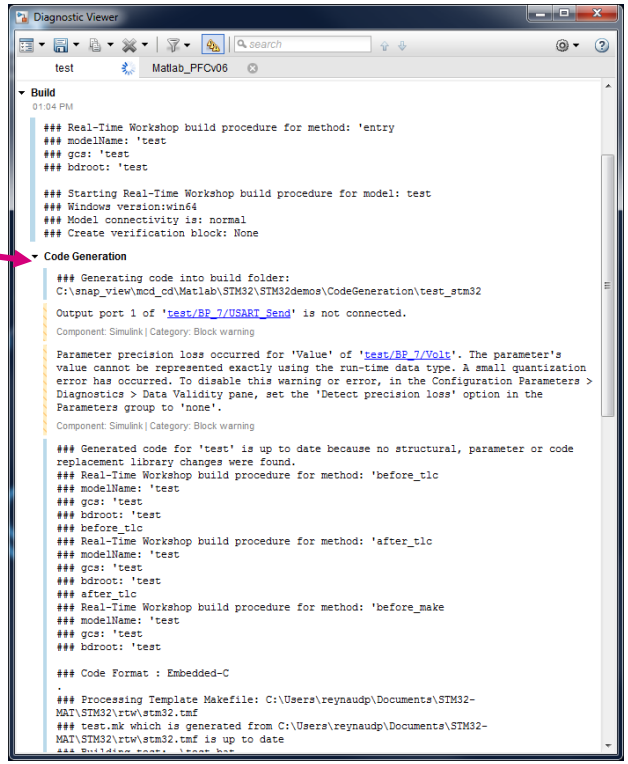
- Generated Code
  - MATLAB® generates C code corresponding to Simulink model in selected Current Folder



  - STM32 code is generated in repository wich name is created using name of Simulink model with « _stm32»

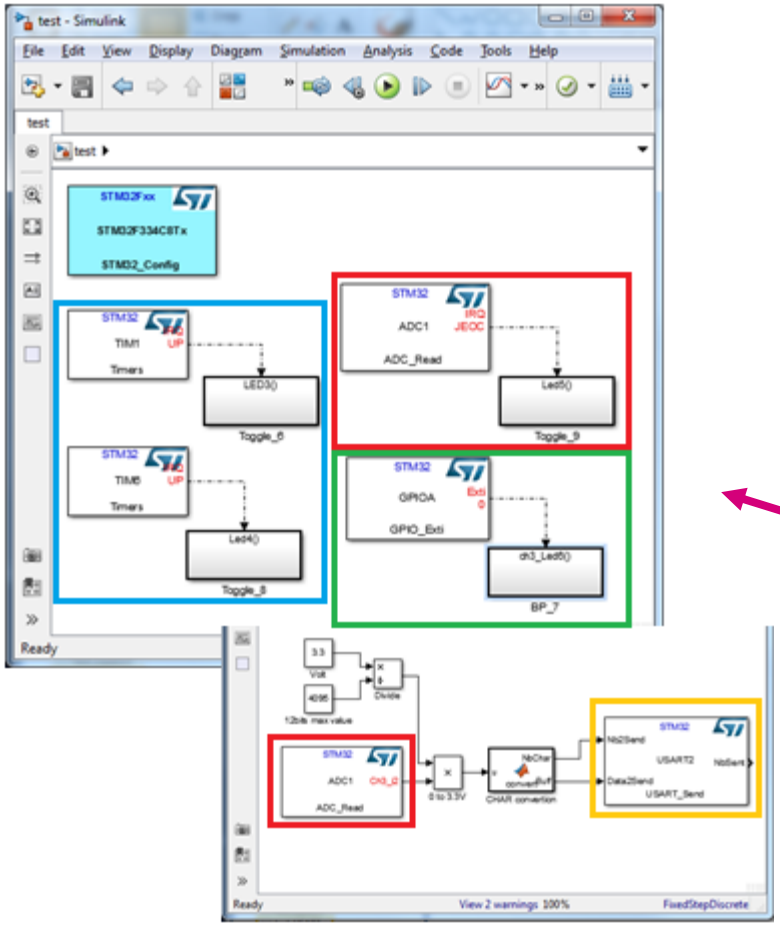






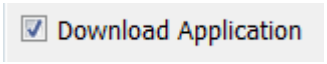
MATLAB® Code Generation 2/2

- STM32 peripherals driver code is generated in .c/.h files wich name is created using name of Simulink model and peripheral name.

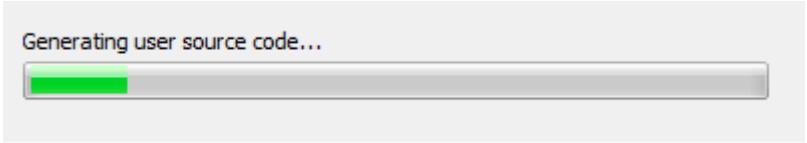


STM32CubeMX Code Generation 1/2

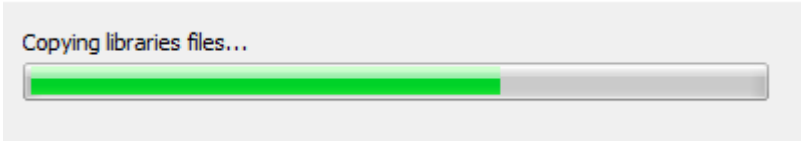
- STM32CubeMX process
  - STM32CubeMX is automatically called from MATLAB® when « Download Application » has been selected from Model Configuration window.



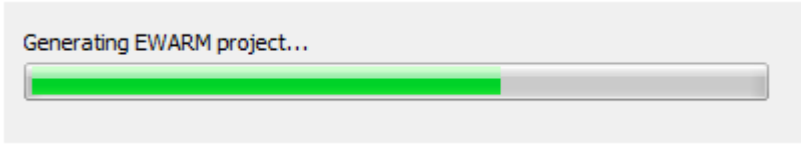
  - STM32CubeMX generates configuration code.



  - STM32CubeMX adds necessary library files. .c/.h library files from HAL STM32 libraries.

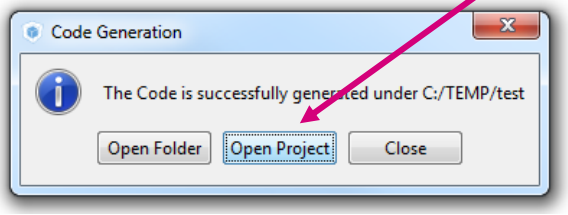


  - STM32CubeMX generates toolchain project including files generated from MATLAB®



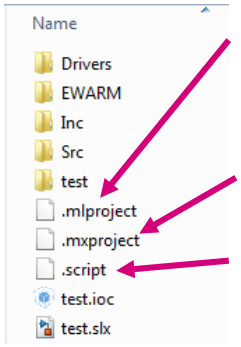
  - STM32CubeMX generated project can be open

Click « Open Project » to automatically open project using selected toolchain.



STM32CubeMX Code Generation 2/2

- STM32CubeMX project generation
  - Project is generated in same repository as ioc file. (Example with Simulink « test » project)

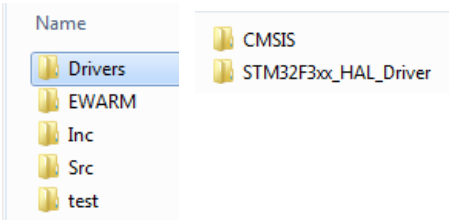


.mlproject : Generated from MATLAB®. Contains information about .c/.h files to add to project from MATLAB®

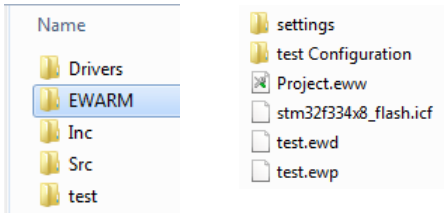
.mxproject : Generated from STM32CubeMX. Contains information about .c/.h files generated from STM32CubeMX

.script : Generated from MATLAB®. Contains STM32CubeMX command to generate project.

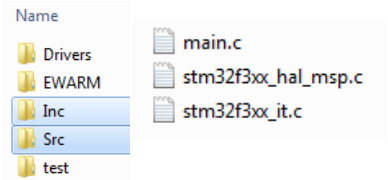
- STM32CubeMX project contains
  - Drivers : Contains STM32 selected library and CMSI files



  - EWARM: Contains toolchain project files (IAR for example)

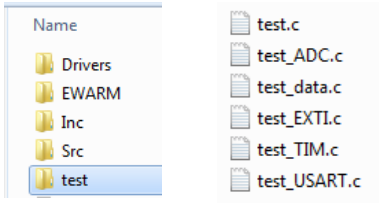


- Inc & Src: Contains STM32CubeMX generated or modified files.



main.c : Generated from MATLAB®. It has been modified by STM32CubeMX to add project configuration.
_hal_msp.c: Peripherals configuration
_it.c : Interrupt handlers for configured interrupt only.

- test (Simulink project name): Contains all .c/h files generated from MATLAB®



It is required to know toolchain functionalities.

Toolchain Project

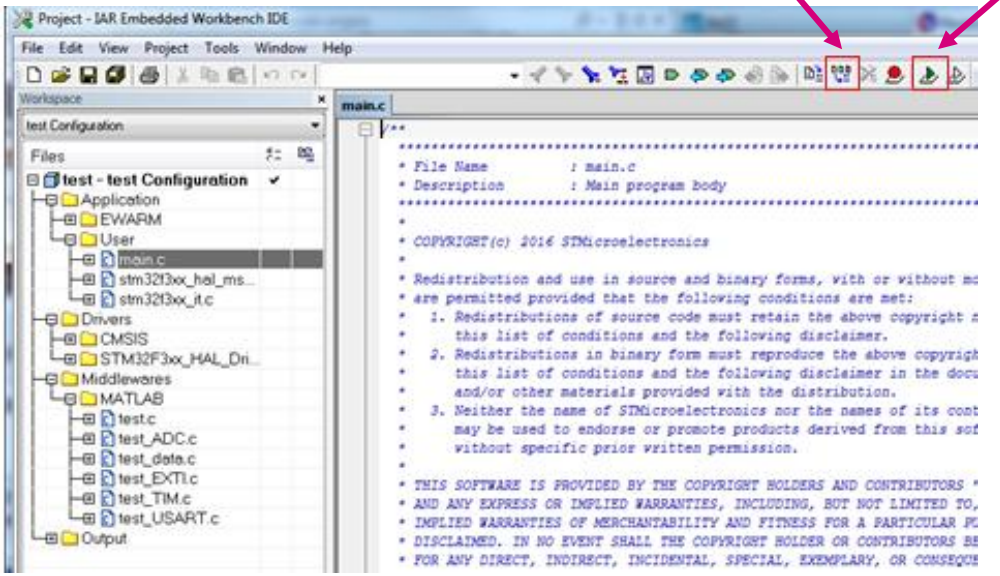
- Toolchain settings
  - STM32CubeMX has automatically generated project including mandatory settings. It is exactly same project at it should be generated « by hand ». Possibility to tune all settings.

Example using EWARM (IAR) toolchain for Simulink test project

- Toolchain Actions
  - Build project
  - Download and Debug project

BUILD project

Download&Debug project

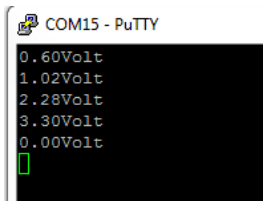


STM32 board must be connected to PC when you click « Download&Debug »

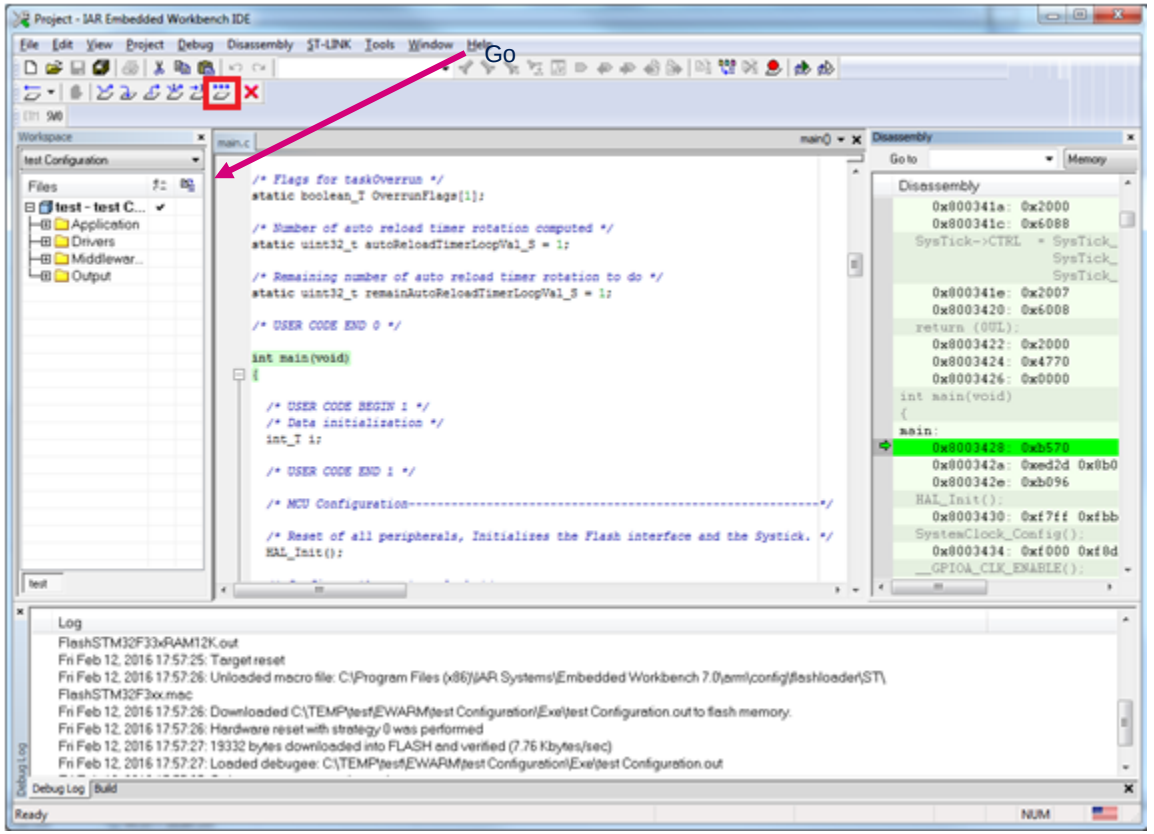
It is required to know toolchain functionalities.

Run Project

- Simulink « test» example results
  - Project is started and waits at 1st main instruction.
  - Click «Go»
  - LD3/LD4/LD5 are blinking
  - LD6 is alternatively ON and OFF when you press User button. ADC value set on PA2 (ADC1 Ch3) is sent to PC through USART.
    You can see ADC value on PC using PuTTY for example.



Example using EWARM (IAR) toolchain for Simulink test project



life.augmented

Releasing your creativity with the STM32

/STM32 @ST_World st.com/e2e

www.st.com/stm32

life.augmented