Introduction to Microchip-SIMULINK Blocksets and MATLAB Plug-in for MPLAB® IDE

Produced by
Murali Manohara Chembarpu

 WebSeminar Title Slide 1

Welcome to "Introduction to Microchip-SIMULINK Blocksets and MATLAB Plug-in for MPLAB® IDE"

MICROCHIP WebSeminars

Introduction & Agenda

- Introduction to Microchip Device Blocksets
- Introduction to MATLAB Plug-In in MPLAB® IDE
- Creating a SIMULINK model with Microchip Blockset and building them
- Learning Goal: To create a SIMULINK model using Microchip Blockset, generate C code from it and build it

 WebSeminar Title Slide 2

MATLAB is a very widely used model based design tool. Microchip's MPLAB® IDE has a Plug-In capable of importing source files generated by RTW component of MATLAB. Users can configure the peripherals of dsPIC® DSC devices in MATLAB/SIMULINK environment using **Microchip Device Blockset** which can be located in SIMULINK library tree after installation.

Application models can be created using **Microchip Device Blockset** and generic SIMULINK blocks. These models can be built either independently from MATLAB or using **MATLAB Plug-in.** The hex/cof files generated from MATLAB can be programmed using Microchip or third party programmers. When MATLAB Plug-in is used, code generated can be modified and debugged from MPLAB environment.

This session will last about 15 minutes, we will go through the steps to use **Microchip Device Blockset** and **MATLAB Plug-In**. This will also guide you through SIMULINK Model creation, configuring RTW parameters, generate code, build and debug code generated from SIMULINK models.

MICROCHIP WebSeminars

Vocabulary

- MATLAB/SIMULINK
- SIMULINK Model
- RTW- Real Time Workshop
- RTW Embedded Coder

 WebSeminar Title Slide 3

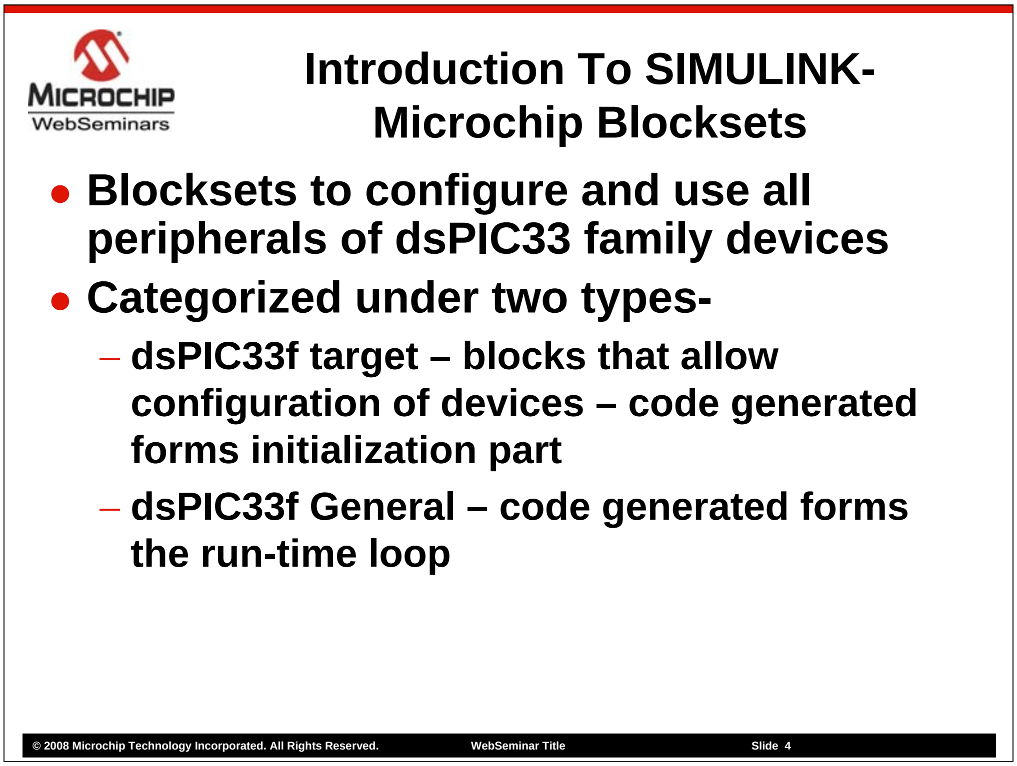
Let’s look at some of the words we will encounter in this session.

MATLAB/SIMULINK This is popularly used tool for mathematical computation and model based design.

SIMULINK Model This is in general a pictorial representation of an application. Created using blocks provided by MATLAB as well as third parties.

RTW- Real Time Workshop This is a optional product of MATLAB which can generate ANSI C code from SIMULINK models.

RTW Embedded Coder This is an optional add-on to RTW. Code generated by RTW is generic in nature. Embedded coder generates optimized code suitable for embedded targets.



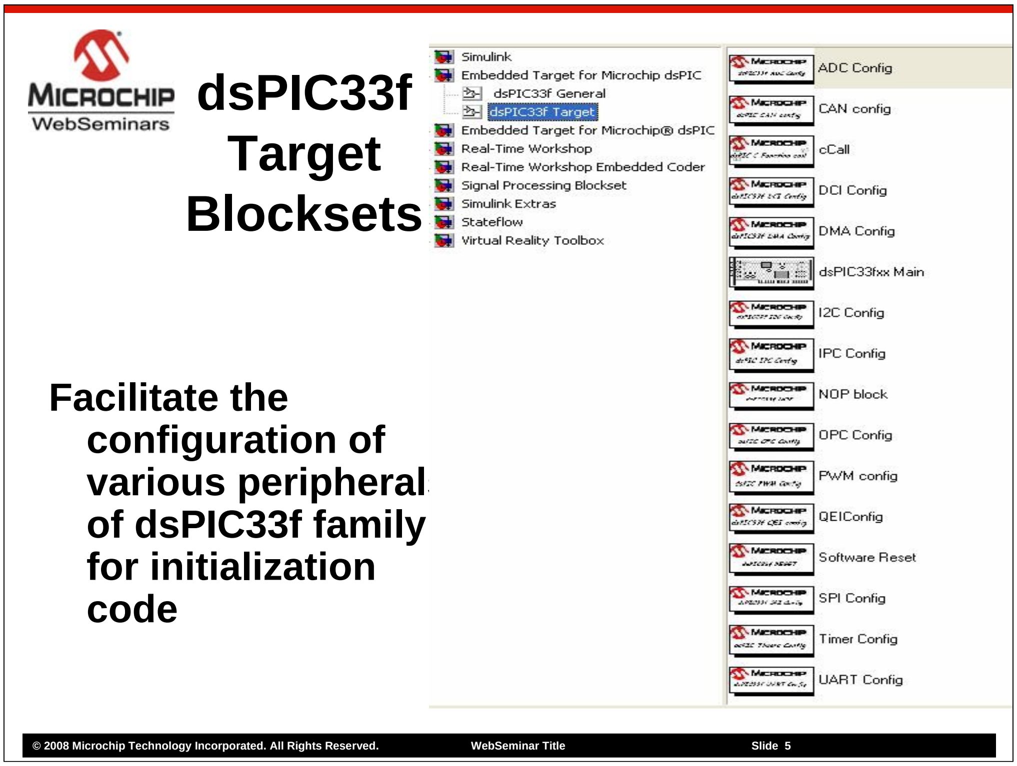
The SIMULINK-Microchip blocksets run with Real-Time Workshop and RTW Embedded Coder of MATLAB.

These blocks enable you to do model-based design of your applications and offload you from the hassle of going through datasheets and configuring peripherals. The Auto-code generation feature of RTW Embedded Coder coupled with drivers behind the microchip blocksets generate efficient C code for your models.

These blocksets appear under two main categories:

•**dsPIC33f Target-** These bocks allow you to configure various peripherals for initialization purposes. You can configure by double clicking on these blocks. These configurations would reflect in the initialization code that is generated from these models e.g., : UART Config

•**dsPIC33f General**- These bocks allow you to configure various peripherals for run-time purposes. You configure by double clicking on these blocks. These configurations would reflect in run-time code generated from these models. E.g.,: UART transmit

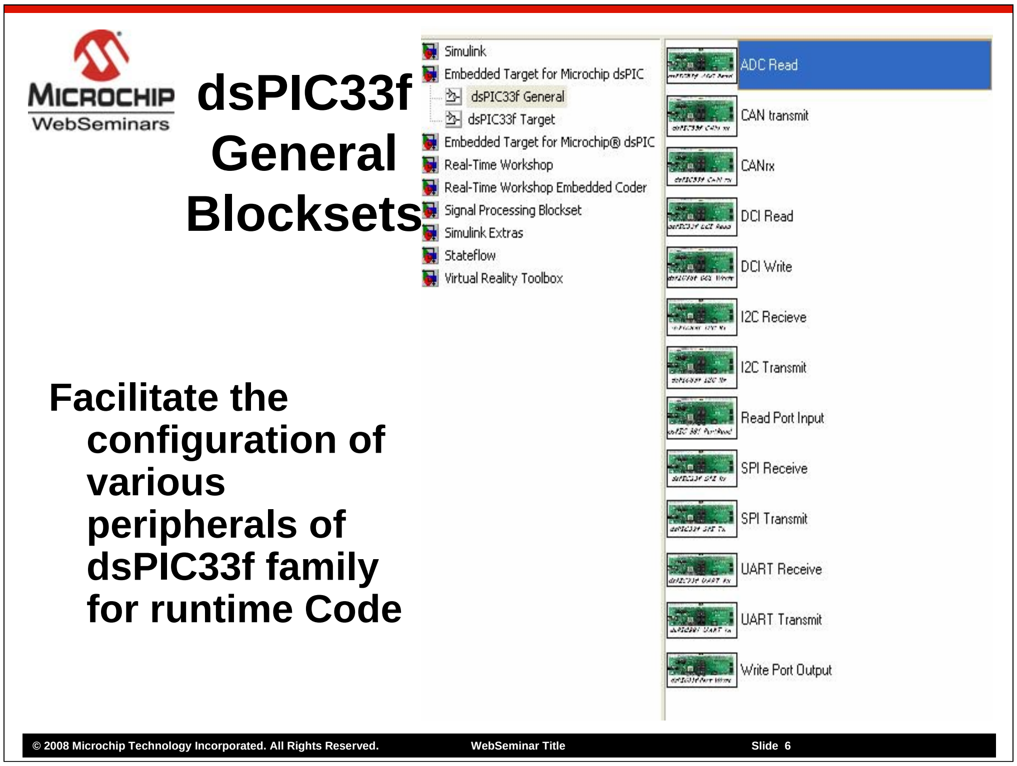


The picture here shows **dsPIC33f Target** library.

You can pull these blocks into your models and configure them as per your requirements.

Peripherals supported here include, but are not limited to: ADC, UART, SPI, Output Compare, CAN, Timers, DMA, I^2C™, DCI, motor Control PWM etc,. This library also has a block ‘cCall’ to import user developed C routines into SIMULINK models. These routines would be compiled and built along with files generated from the model.

‘dsPIC33fxx Main’ block found here is essential component for every application model that you create. This allows you to configure clocks, Oscillators etc.

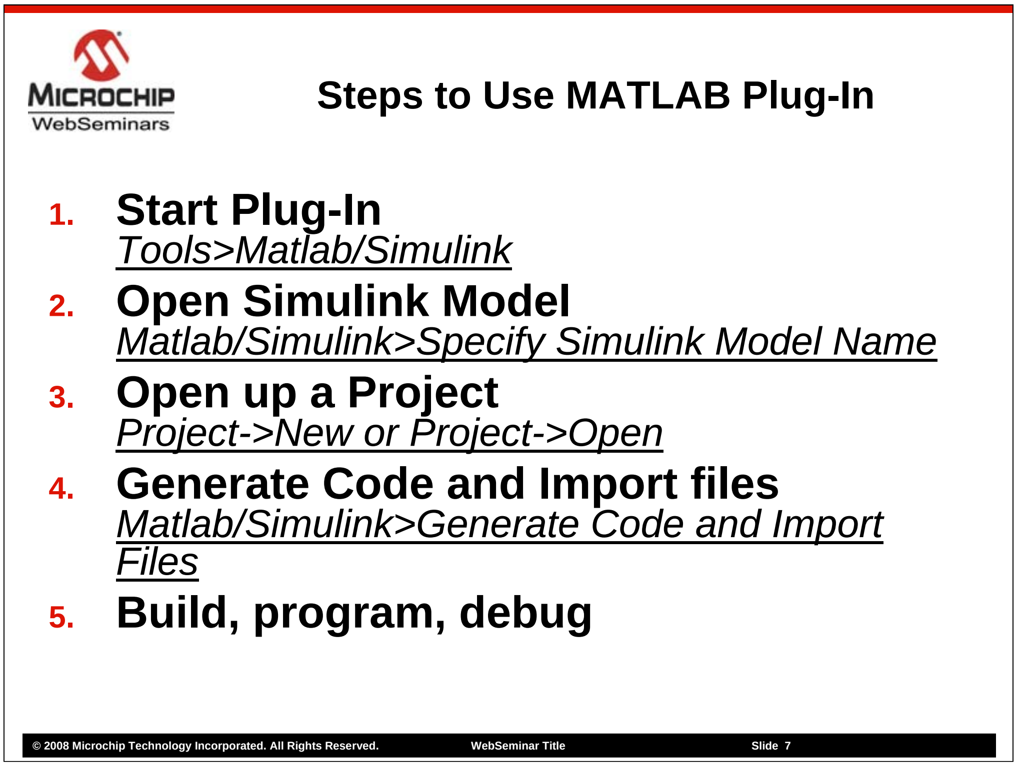


The picture here shows **dsPIC33f General** library.

You can pull these blocks into your models and configure them as per your requirements.

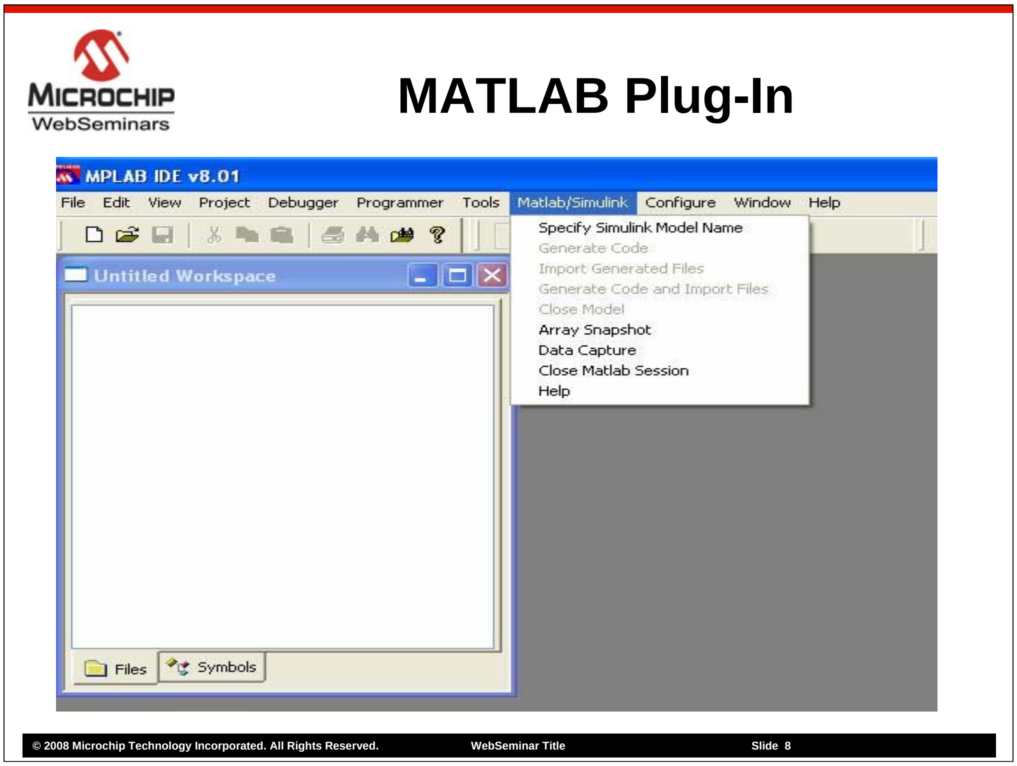
Blocks found under this library generally have one or more input/output ports, which have to be connected to appropriate source/sink blocks.

E.g.,: UART transmit block has an input port which accepts ‘uint16/int16’ data type and transmits it on every timer expire event. This timer has to be configured in dspic33f Main block.



Here is an overview of the steps to use MATLAB Plug-In for MPLAB.

1. Select ***Tools>Matlab/Simulink*** from the menu.
2. Select ***Matlab/Simulink>Specify Simulink Model Name*** to specify the SIMULINK model to be opened for code generation.
3. Open up a Project, either fresh one or an existing one. The code generated will be added to this project.
4. Click on Generate Code. The plug-in will generate code through RTW embedded coder.
5. Click on Import Generated files. The plug-in will include all generated files into the active project.
6. You can edit (if required) and build the code generated. You can program the hex/cof through MPLAB® IDE. You can also do source-level debugging in the same way you would do with hand written code.

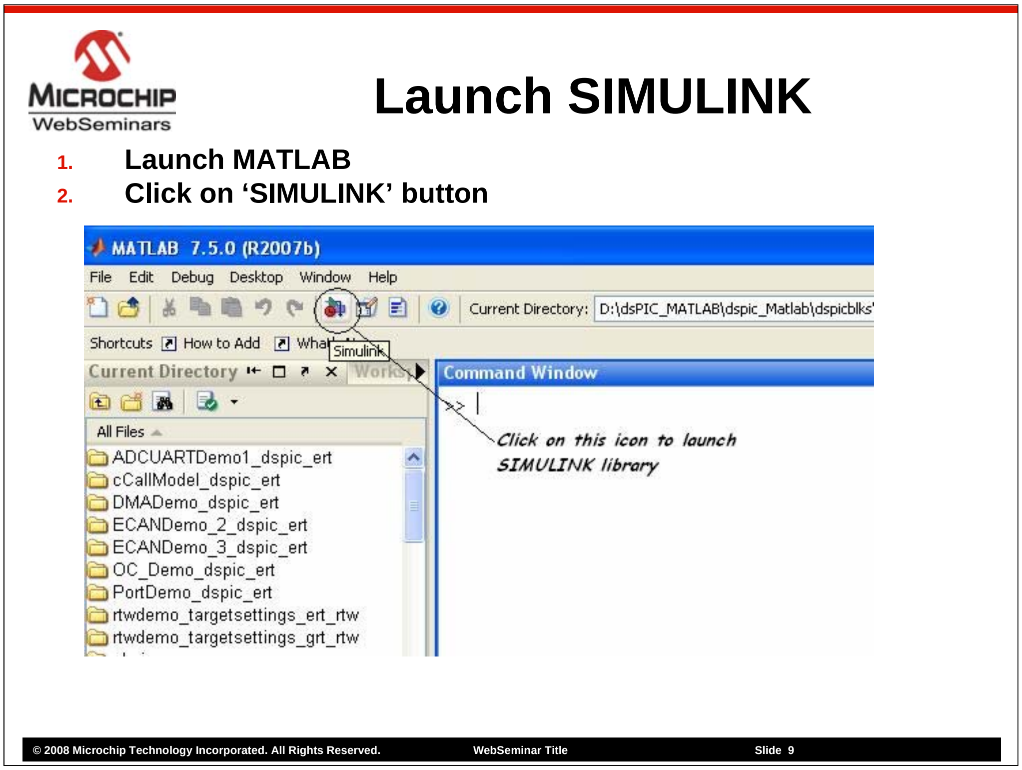


The screen shot shows the MPLAB® IDE with MATLAB/SIMULINK menu. It appears when user clicks on MATLAB/SIMULINK in 'Tools' drop down menu of MPLAB® IDE. When you click on 'Specify Simulink Model name', it will prompt the user to choose a SIMULINK model to open.

If it is the first time you are working with MATLAB, you need to launch MATLAB and create and save a SIMULINK model separately so that you can specify that model name in the plug-in.

Once you choose a SIMULINK model, MPLAB will launch MATLAB and open up the SIMULINK model that you specified. This model can be modified and saved.

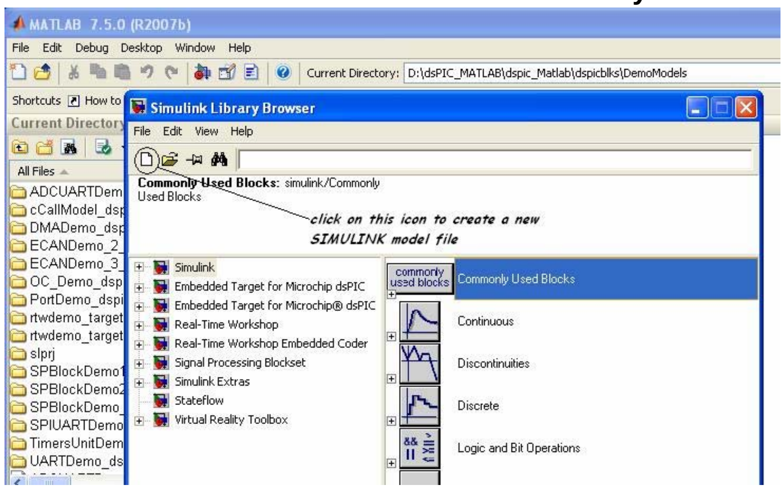
It is mandatory to have a project in MPLAB IDE before you generate code from a model.



1. Launch MATLAB
2. Click on SIMULINK icon as shown in the picture. This would open up SIMULINK library browser. This will have default SIMULINK blocks and different blocksets that you might have bought from Mathworks.
3. Make sure that 'Embedded Target for Microchip dsPIC' appears in SIMULINK library tree. If it does not exist, then you need to install it or add path of installation to MATLAB paths. Refer to '***Device Blocks Deployment details.doc***'

Create a new SIMULINK model-1

Click on ‘Create a new Model’ in SIMULINK Library browser

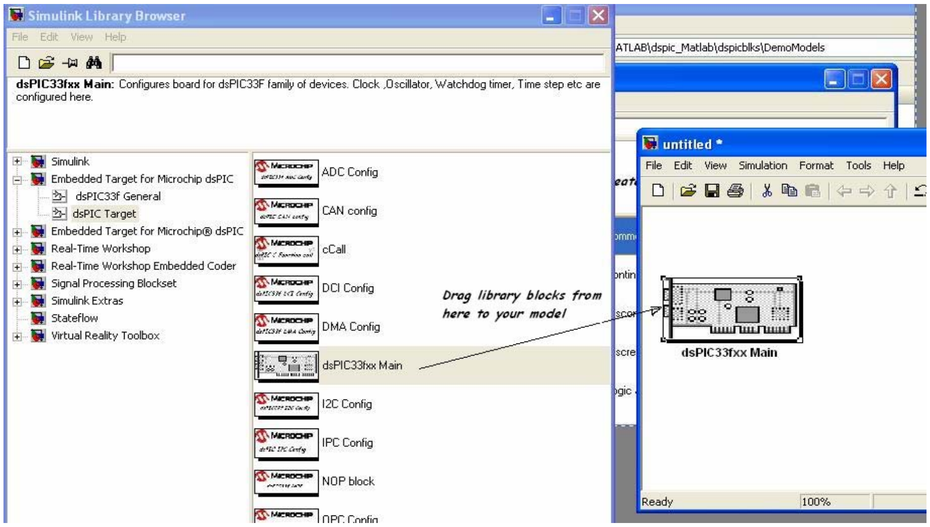


 WebSeminar Title Slide 10

1. Click on ‘Create a new Model’ in SIMULINK Library browser. This would open up an empty model file. You can populate this model file with different blocks from SIMULINK library and create an application model.

Create a new SIMULINK model-2

Populate your model file with required blocks

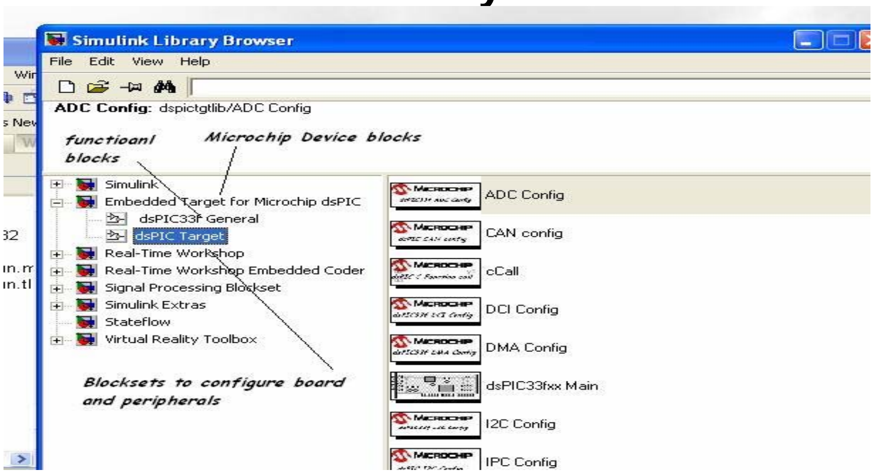


 WebSeminar Title Slide 11

Populate your model file with required blocks. You can use Generic SIMULINK blocks which support fixed point data types along with microchip blocks.

Generic SIMULINK blocks can be checked for compatibility by clicking on ‘Signal Attributes’ tab in their respective masks (the pop-up GUI that appears when you double click on these blocks). If data types like ‘uint16’, int8’ etc are listed there, these blocks can be used with microchip blocksets.

Microchip blocksets in SIMULINK Library browser



Use these blocks with any fixed point generic blocks of SIMULINK to create application models

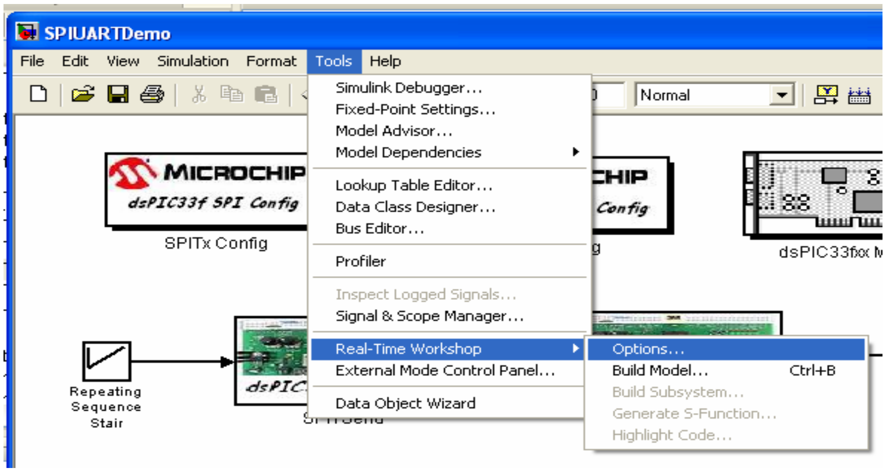
 WebSeminar Title Slide 12

This diagram shows ‘Embedded Target for Microchip dsPIC’ contents in SIMULINK library browser. These blocks can be pulled to the model that you are creating. Double clicking on these blocks (after placing them in models) allows you to configure these blocks.

You can use all generic SIMULINK blocks which support fixed point operations.

Blocks which have input/output ports need to be connected with appropriate sources/sinks. For e.g.,: for UART transmit, you need to connect a block which outputs a uint16/int16 datatype.

RTW Settings

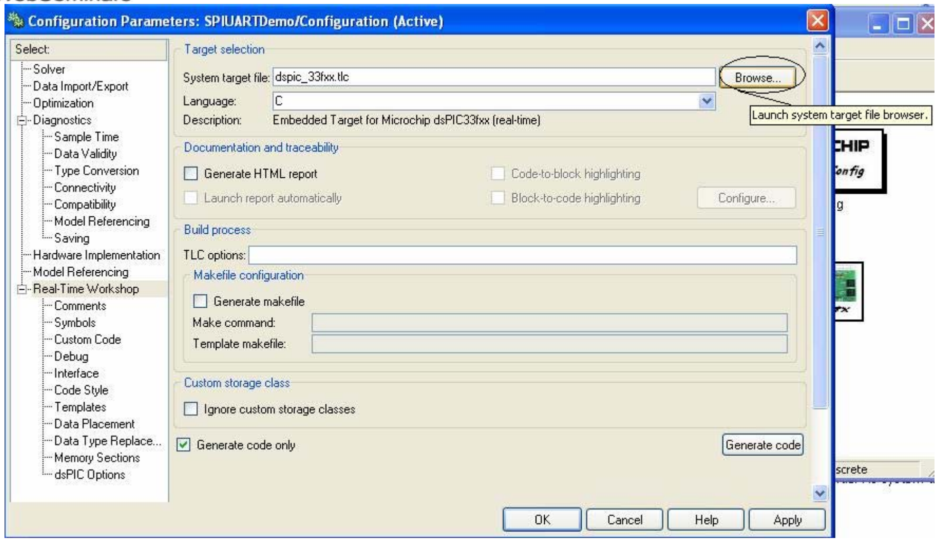


- **Real Time Workshop Options**

 WebSeminar Title Slide 13

This picture shows how to launch Real Time workshop Configuration dialog. Once the SIMULINK block is designed and ready, you need to do some configuration settings for each model. These settings are used in configuring generated code from the respective models.

RTW configuration



- **Select system target file**

 WebSeminar Title Slide 14

Here we have view of ‘Real Time Workshop’ pane of RTW configuration dialog. You have to choose ‘**dspic_33fxx.tlc’** as system target file. You can choose this file by clicking ‘Browse’ button which is circled in the picture shown. This will pop up a list of system target files that are registered in your MATLAB installation.

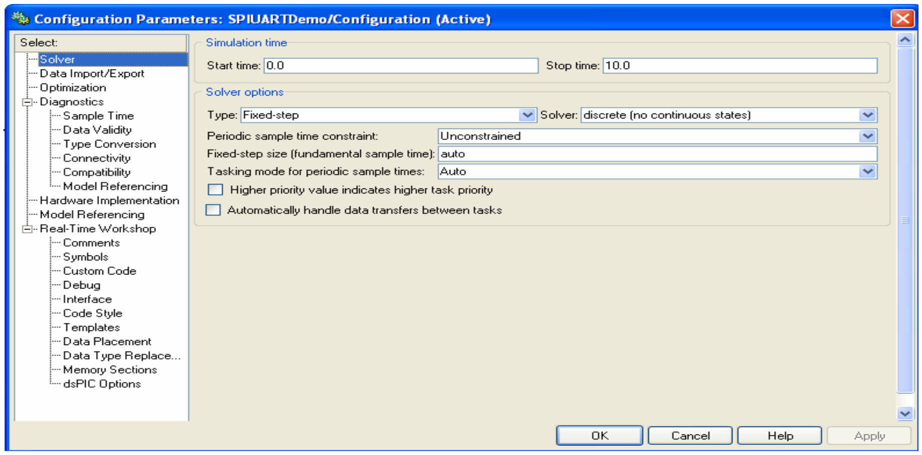
If you would be using MATLAB plug-In and MPLAB® IDE to build,

- ‘ert.tlc’ also can be used as system target file
- ‘Generate makefile’ option can be unchecked
- ‘Generate code only’ Option can be checked

If you would like to build this model without the help of plug-In in MPLAB® IDE,

- You should check ‘Generate Makefile’ Option
- ‘Generate code only’ Option should be unchecked

RTW Solver Configuration

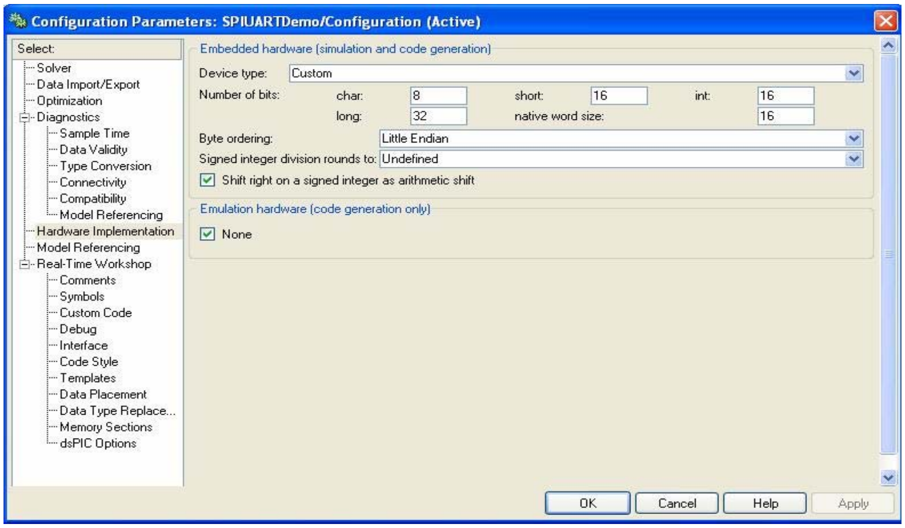


- Select fixed point solver

 WebSeminar Title Slide 15

Here we have view of ‘Solver’ pane of RTW configuration dialog. Since Microchip blocksets do not have simulation capabilities, you need to select ‘**fixed step**’ type ‘**discrete (no continuous state)**’ solver. All other fields can be left to default.

RTW Hardware Implementation



- **Hardware properties**

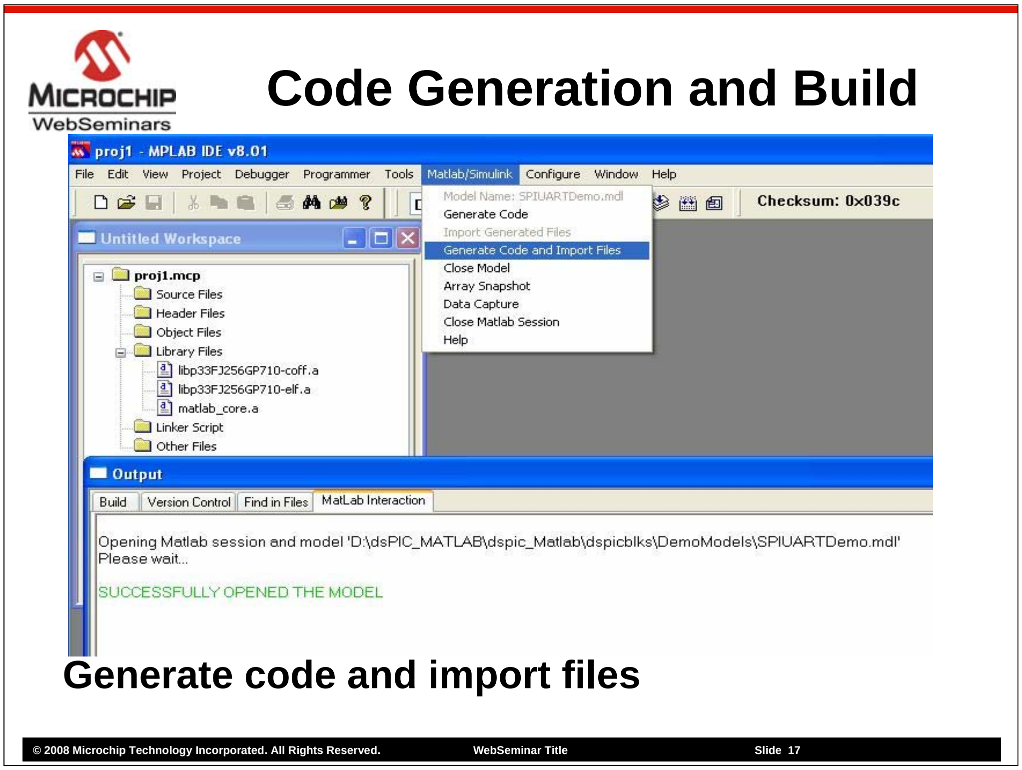
 WebSeminar Title Slide 16

Here we have view of ‘Hardware Implementation’ pane of RTW configuration dialog of MATLAB R2007a.

Select ‘**custom**’ device type (R2007b onwards, there will be an additional field, where you can choose ‘**generic**’)

Select number of bits as follows:
Char- 8, short- 8, int-16, long- 32, native word size -16
Byte ordering- Little Endian

Emulation Hardware- check ‘none’



Picture here shows how to generate and import files from a SIMULINK model once you have configured RTW settings and saved the model.

It is mandatory to have a project in MPLAB IDE before you generate code from a model.

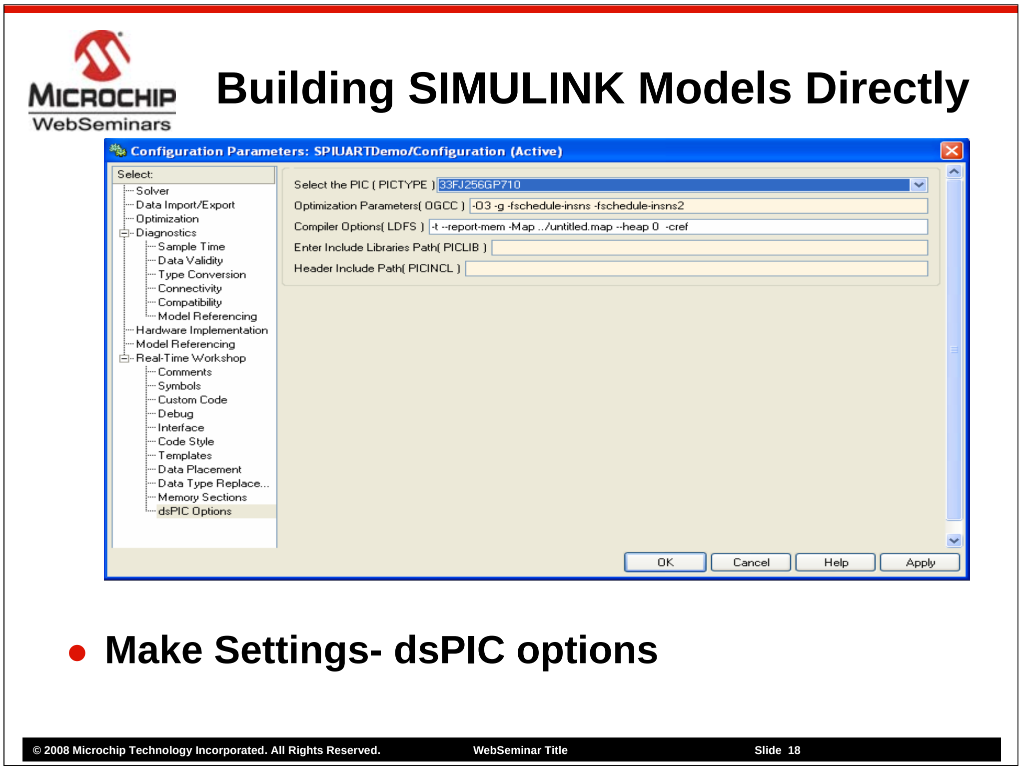
‘Generate Code’ generates code from the model specified. ‘**Import Generated Files**’ imports all the files that are generated from the model into the active project. All other dependent files are compiled and object files are archived and imported as ‘matlab_core.a’

By clicking ‘**Generate Code and Import Files**’ both operations of generating files and importing them into project in a single step.

Add library files ‘libp33FJxxxxxxx-coff.a’ and ‘libp33FJxxxxxxxx-elf.a’ to the project and build.

To trace arrays when the MPLAB REAL ICE in-circuit emulator or MPLAB SIM simulator is the debugger, select ‘**Array Snapshot**’. Tracing array variables occurs on a Halt. The data can be loaded in the MATLAB workspace.

To use the data capture facility of the MPLAB REAL ICE in-circuit emulator, select ‘**Data Capture**’. The data can be loaded in the MATLAB workspace.



Here we have view of ‘dsPIC Options’ pane of RTW configuration window. This will appear only if you have chosen ‘**dspic_33fxx.tlc**’ as system target file.

These settings will be effective only if you build SIMULINK models without using **MATLAB plug-in**.

Select the PIC – Select the target dsPIC device here. Default is dsPICJ256GP710

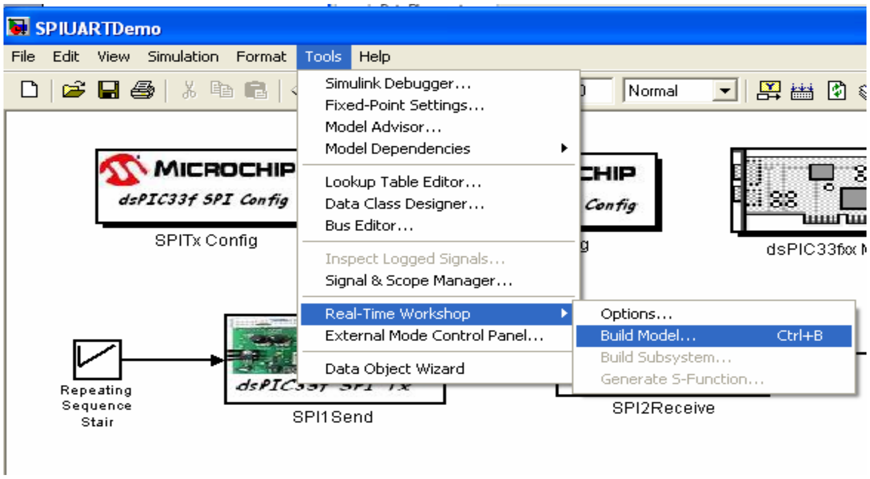
Optimization Parameters – You can enter optimization switches here. Leave it to default if you are not sure as what to use.

Compiler Options - You can enter compiler switches here. Leave it to default if you are not sure as what to use.

Enter Include Libraries path – If you are linking from any specific archives or libraries, you can enter their path here separated with white spaces

Header Include Path - If you want to include any headers specifically, you can enter their path here separated with white spaces

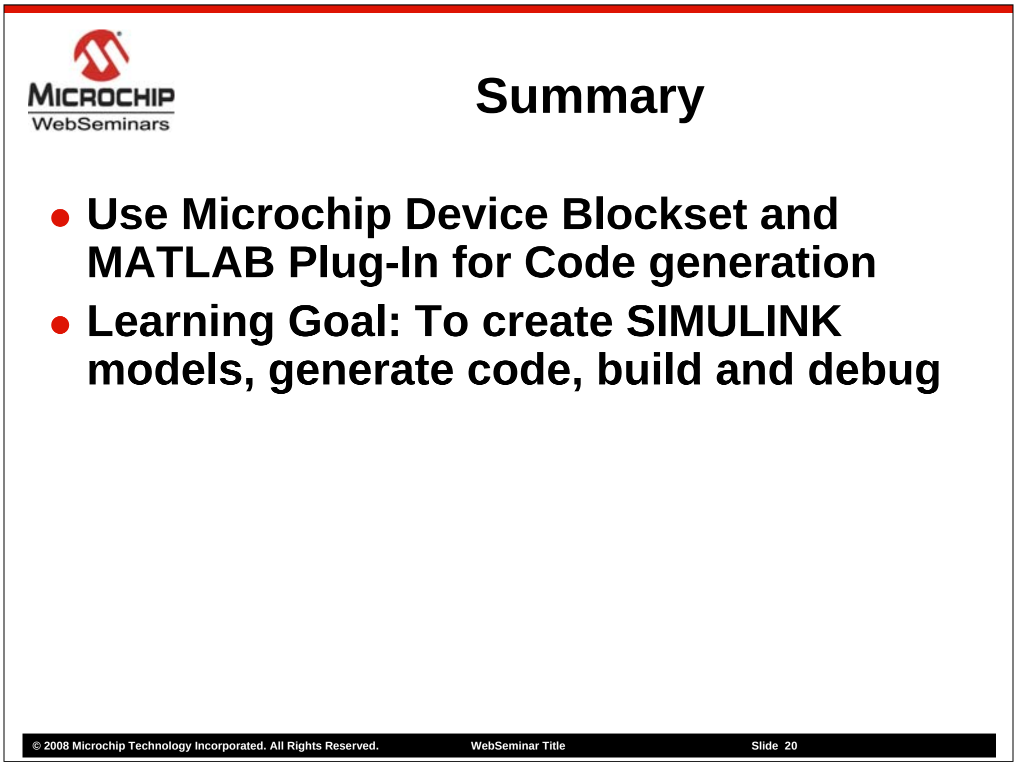
Building SIMULINK Models Directly



- Build the model to generate hex/cof file

 WebSeminar Title Slide 19

Here we show how to directly generate hex/cof files from your SIMULINK model without using the **MATLAB Plug-in**. Hex/cof files generated can be imported into MPLAB® IDE (or any third party programmers) and programmed into your target devices.



That's all you need to use Microchip Device Blocksets and MATLAB Plug-in for MPLAB® IDE!

Let's summarize what we have learned in this session. We have outlined the steps to use the Microchip Device Blocksets and MATLAB Plug-in for MPLAB® IDE.

You can use these independently or together. We have demonstrated how you can use them to make your application development faster.

Please provide us feedback by visiting our webinar website and click on the Feedback link.

What other webinars could be useful to you?

Where to Get More Information

MICROCHIP WebSeminars

- Other webinars http://www.microchip.com/webinar
- Discussion board http://forum.microchip.com > Development Tools > MATLAB
- Support (Knowledge Base and Help Ticket) http://support.microchip.com
- Collaboration WIKI http://www.microchip.com/wiki

 WebSeminar Title Slide 21

If you are interested in other webinars, please go to http://techtrain.microchip.com/webseminars/Archived.aspx for more. You may select “Development Tools” in the Category for development tools specific webinars.

We have a discussion board at http://forum.microchip.com where you can post questions in the user community. You can drill down to the Development Tools then MPLAB® IDE Simulator to see questions and answers regarding the simulator and stimulus.

You could also go to our support page to see published knowledge base articles and submit a web ticket for our support staff.

Lastly, you may collaborate with our customers by visiting our WIKI page on topics related to application development.

Thank you for your interest.