Topics

- We are AAM
- Background
- Conventional Method
- Auto Code Generation Method
- Comparison
AAM was founded in 1994, with manufacturing expertise rooted in more than 90 years of experience. Today, AAM is a leading, Tier-One global automotive supplier of driveline and drivetrain systems and related components for light trucks, SUVs, passenger cars, crossover vehicles and commercial vehicles.

Experience the AAM Advantage:

- **Operational Excellence**
- **Global Market Cost Competitiveness**
- **Technology Leadership**

**Facts and Figures:**

- **2012 Sales:** $2.93 Billion
- **Employees:** Approximately 11,700 associates globally
- **Locations:** More than 33 locations in 13 countries
- **Customers:** More than 100 customers
Driveline Product Portfolio

Light Truck or SUV
- Front Axle
- Front & Rear Propeller shafts
- Rear Beam Axle
- Transfer Case

Passenger Car or Crossover Vehicle
- Front Axle
- Front & Rear Propeller shafts
- Rear Beam Axle
- Multi-piece Propeller shaft
- Power Transfer Unit

Commercial Vehicle Truck
- Front Axle
- Propeller shaft
- Rear Drive Module
- Rear Beam Axle
Drivetrain Systems

- Transfer cases
- Power Transfer Units
- I-Ride™ Suspension Modules
- Smart Bar™ Electronic Stabilizer Systems
- Front & Rear Wheel Drive eLSD

For more information, please visit our website, www.aam.com
In Automotive domain ADC are used extensively to receive the feedback from different sensors and process that data to ensure the proper functionality of the system.

Processing of the data often done in MATLAB and the ADC driver code is written in C.

Now a day’s most of the developers are used to MATLAB and they need some graphical interface in MATLAB to configure the ADC and read the ADC channel.
Conventional Method

- Conventionally, ADC driver is written with all possible configurations.
- “ADC_Cfg.h” file is used to configure ADC driver.
- To read the ADC channel, ADC read function is called in C code.
- Data Store block is used to access the data in MATLAB from C code.
Limitation for MATLAB Environment

- Configuring the ADC is cumbersome using the “ADC_Cfg.h” files for MATLAB user.

- C code is not readable because of too many precompiler switches.

- Too many global variables required to access the data in MATLAB.
Auto Code Generation Method

 invocation TLC code

 Code generation using TLC by MATLAB

 ATD Driver Code

 User

 ADC Config:
 Frequency: 24 MHz
 Resolution: 8 Bit
 Interrupt: Enable
 Mux: Enable
 Ext Trig Source: Enable
 Ext Trig Channel: 0
 Left Justification: Enable
 Discharge Before Sample
 Spec Channel Conv: Disable
 ADC Junc Temp: Disable
 ADC Band Gap: Disable
The ATD block provides read access to the ATD input channels available in the board support package (BSP).

Parameters

ADC Channel: MotTTD_Current

ADC Read:
ADC Channel: MotTTD_Current
ATD0 AN1 Pin 69
Auto Code Generation Method

- AAM has written a <BSP>.m file which has the information about the ADC channel name and their connection to the micro pin.
- AAM has written 2 S-function blocks.
  - One block to configure the ADC as per user requirement.
  - Second block to read the ADC channel value.
- AAM has written two TLC function for code generation.
  - One uses the ADC configuration block parameter value to generate the configured ADC code without any dead code.
  - Second uses the ADC read channel block to generate the ADC read function which will be accessed in MATLAB.
Code Comparison

Auto Generated code

Conventional code

Lines in circles are present in convention code
But not in Auto Generated code
Advantages

- A graphical interface is convenient to configure the ADC for MATLAB users.

- Only required code is generated which is easy to understand as precompiler switches are not used.

- No global variable sharing is required as we can directly read the channel value from MATLAB.