Accelerating Innovative Design Using Low-Cost Hardware

Andrew Bennett

MATLAB EXPO 2015
UNITED KINGDOM
Introduction

- Interfacing to the real-world
- How to do this with MATLAB and Simulink
- Choosing most appropriate solution
  - Live demonstrations of examples
- Take home information:
  - How you can accelerate your commercial designs using low-cost hardware
  - Where to find out more
Workflows covered today

- Access real-world data on desktop
  - Developing or prototyping algorithms
  - Read data from sensors
  - Send data to servers
  - Control actuators

- Deploy algorithms
  - Read data from sensors
  - Control actuators
  - Send data to servers
Sensors

- Digital
  - On/off
- Analogue
  - Continuously varying
  - e.g. position, temperature
- Cameras
  - USB cameras
  - Hardware specific/bespoke
Sensors – Analogue

- **Acceleration**
  - Accelerometer
  - X, Y, and Z, m/s²
- **Magnetic Field**
  - Magnetometer
  - X, Y, and Z, μT
- **Orientation**
  - Gyroscope
  - Roll, Pitch, and Yaw degrees
- **Angular velocity**
  - Derivative of orientation
  - X, Y, and Z, rad/s

- **Global Positioning System (GPS)**
  - Latitude, degrees
  - Longitude, degrees
  - Speed, m/s
  - Course, degrees
  - Altitude, m
  - Horizontal accuracy, m

18 analogue values…
… accessible from an everyday device
… mobile phone
Actuators

- **Digital**
  - On/off
- **Analogue**
  - Continuously varying
- **Somewhere in the middle**
  - Pulse-Width Modulation (PWM)
Target Support Packages from MathWorks

- Commercial off the shelf software is released twice per year
- Target Support Packages are updated two or more times per year to keep pace
- Available from within MATLAB
- Available from MathWorks web-site
- Easy to download
- Automated install – as long as you have administrator rights
Installing Hardware Support Packages
Access real-world data on desktop

- MATLAB Support Packages
- Key benefits of using MATLAB to read, write, and analyze:
  - Don’t wait for your code to compile: read and write sensor data interactively
  - Analyze and visualize your data using thousands of built-in math, engineering, and plotting functions
  - Share code and discuss project ideas with over one million MATLAB users around the world
Deploy algorithms

- **Simulink Support Packages**
- Key benefits of using Simulink to develop algorithms:
  - Develop and simulate your algorithms in Simulink and use automatic code generation to run them on the device
  - Incorporate signal processing, control design, state logic, and other advanced math and engineering in your hardware projects
  - Interactively tune and optimize parameters as your algorithm runs on the device
  - Easily modify algorithms to run on other low cost and commercial hardware platforms
If your application needs...

- **Challenge**
  - Automotive – Driving quality metrics e.g. RAC
  - Aerospace – Sensor fusion, GNC
  - Medical – Motion, movement or tremor analysis

- **Solution**
  - Consider MATLAB App phone and tablet sensors

- **Benefit**
  - Small form factor
  - Self-contained
  - On-board battery for power
Example: MATLAB App: Sensors

- MATLAB App
- MATLAB App sensors
- Stream data into MATLAB
If your application needs…

- **Challenge**
  - IO including
    - Digital
    - SPI
    - I²C

- **Solution**
  - Consider Arduino
    - MATLAB Support Package
    - Simulink Support Package

- **Benefit**
  - Lightweight
  - Lots of IO – digital and analogue
# MATLAB Support Package for Arduino

## Setup and Configuration

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>supportPackageInstaller</td>
<td>Install support for third-party hardware or software</td>
</tr>
<tr>
<td>arduino</td>
<td>Connect to Arduino hardware</td>
</tr>
</tbody>
</table>

## Read and Write Data

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>configurePin</td>
<td>Arduino pin mode</td>
</tr>
<tr>
<td>readDigitalPin</td>
<td>Read data from digital pin on Arduino hardware</td>
</tr>
<tr>
<td>writeDigitalPin</td>
<td>Write to digital pin on Arduino hardware</td>
</tr>
<tr>
<td>writePWMVoltage</td>
<td>Write digital pin PWM voltage value</td>
</tr>
<tr>
<td>writePWMDutyCycle</td>
<td>Set digital pin PWM duty cycle</td>
</tr>
<tr>
<td>playTone</td>
<td>Play tone on piezo speaker using digital pin</td>
</tr>
<tr>
<td>readVoltage</td>
<td>Read Arduino analog pin voltage</td>
</tr>
<tr>
<td>configureAnalogPin</td>
<td>Set analog pin mode</td>
</tr>
<tr>
<td>configureDigitalPin</td>
<td>Set digital pin mode</td>
</tr>
</tbody>
</table>

## SPI Devices

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>spiDev</td>
<td>Connect to SPI device</td>
</tr>
<tr>
<td>writeRead</td>
<td>Read and write data from SPI sensor</td>
</tr>
<tr>
<td>arduino</td>
<td>Connect to Arduino hardware</td>
</tr>
</tbody>
</table>

## Servo Motors

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>servo</td>
<td>Create connection to servo motor</td>
</tr>
<tr>
<td>readPosition</td>
<td>Read servo motor position</td>
</tr>
<tr>
<td>writePosition</td>
<td>Write position of servo motor</td>
</tr>
<tr>
<td>arduino</td>
<td>Connect to Arduino hardware</td>
</tr>
</tbody>
</table>

## Add-On Devices

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>addOn</td>
<td>Create add-on device connection</td>
</tr>
<tr>
<td>dcmotor</td>
<td>Attach DC motor to Adafruit motor shield</td>
</tr>
<tr>
<td>servo</td>
<td>Create add-on servo motor connection</td>
</tr>
<tr>
<td>start</td>
<td>Start DC motor</td>
</tr>
<tr>
<td>stop</td>
<td>Stop DC motor</td>
</tr>
<tr>
<td>stepper</td>
<td>Attach stepper motor to Adafruit motor shield</td>
</tr>
<tr>
<td>move</td>
<td>Move stepper motor</td>
</tr>
<tr>
<td>release</td>
<td>Release stepper motor</td>
</tr>
</tbody>
</table>

---

Device connected on Arduino I2C bus.

Read from I2C device register.

Write to I2C device register.

Scan Arduino hardware for I2C bus address.

Read I2C data.

Write data to I2C bus.
Simulink Support Package for Arduino
Example: Arduino

- Run on Target Hardware
  - External mode – use Dashboard widgets and tune parameters on-line
  - Deployment – deploy to hardware to run immediately at power on
If your application needs…

- **Challenge**
  - High-sample rate data
  - Data processing on edge-node

- **Solution**
  - Consider Raspberry Pi
    - MATLAB Support Package
    - Simulink Support Package

- **Benefit**
  - Full Linux Operating System
  - Floating point mathematics
MATLAB Support Package for Raspberry Pi

Installation and Setup

- **supportPackageInstaller**: Install support for third-party hardware or software
- **raspi_examples**: Open featured examples for this support package
- **targetupdater**: Setup support package that is already installed
- **matlabshared.supportpkg.checkForUpdate**: List of support packages that can be updated
- **matlabshared.supportpkg.getInstalled**: List of installed support packages

Connection to Raspberry Pi Hardware

- **raspi**: Create connection to Raspberry Pi hardware

LEDs

- **raspi**: Create connection to Raspberry Pi hardware
- **writeLED**: Turn LED on or off
- **showLEDs**: Show location, name, and color of user-controllable LEDs

GPIO Pins

- **raspi**: Create connection to Raspberry Pi hardware
- **configureDigitalPin**: Configure GPIO pin as input or output
- **readDigitalPin**: Read logical value from GPIO input pin
- **writeDigitalPin**: Write logical value to GPIO output pin
- **showPins**: Show diagram of GPIO pins

Serial Port

- **raspi**: Create connection to Raspberry Pi hardware
- **serialdev**: Create connection to serial device
- **read**: Read data from serial device
- **write**: Write data to serial device

I2C Interface

- **raspi**: Create connection to Raspberry Pi hardware
- **scanI2CBus**: Scan I2C bus device addresses
- **i2cdev**: Create connection to I2C device
- **read**: Read data from I2C device
- **write**: Write data to I2C device
- **readRegister**: Read from register on I2C device
- **writeRegister**: Write to register on I2C device
- **enableI2C**: Enable I2C interface
- **disableI2C**: Disable I2C interface

SPI Interface

- **raspi**: Create connection to Raspberry Pi hardware
- **spiDev**: Create connection to SPI device
- **writeRead**: Write data to and read data from SPI device
- **enableSPI**: Enable SPI interface
- **disableSPI**: Disable SPI interface

Camera Board

- **raspi**: Create connection to Raspberry Pi hardware
- **cameraBoard**: Create connection to Raspberry Pi Camera Board Module
- **snapshot**: Capture RGB Image from Camera Board
- **record**: Record video from Camera Board
- **stop**: Stop video recording from Camera Board

Linux

- **raspi**: Create connection to Raspberry Pi hardware
- **system**: Run command in Linux shell on Raspberry Pi hardware
- **openShell**: Open terminal on host computer to use Linux shell on Raspberry Pi hardware
- **getFile**: Transfer file from Raspberry Pi hardware to host computer
- **putFile**: Transfer file from host computer to target hardware
- **deleteFile**: Delete file on target hardware
Simulink Support Package for Raspberry Pi
Example: Raspberry Pi

- Raspberry Pi Camera
- Model-view-control application with seven-segment display
If your application needs…

- **Challenge**
  - Automotive – sensing and steering
  - Robotics – actuation and control of joints and mechanisms

- **Solution**
  - Consider LEGO Mindstorms EV3

- **Benefit**
  - Wide range of shrink wrapped sensors and actuators
  - Real-time control algorithms are technically feasible
  - Can easily be extended using components like those used by Gabrielle
LEGO MINDSTORMS EV3

- **EV3 Intelligent Brick**
- **4 Actuator Ports**
- **Actuators: Large or Small Motors**
- **USB Port: For Wi-Fi Dongle**
- **4 Sensor Ports**
- **Sensors: Ultrasonic, Touch, Color, IR, Gyro, etc.**
# MATLAB Support Package for LEGO MINDSTORMS EV3

## Installation and Setup
- **Using legoev3 Objects**: Connection to EV3 brick
- **supportPackageInstaller**: Install support for third-party hardware or software
- **legoev3**: Create connection to EV3 brick

## Read and Write Data
- **Using legoev3 Objects**: Connection to EV3 brick
- **legoev3**: Create connection to EV3 brick
- **writeLCD**: Write characters to LCD on EV3 brick
- **clearLCD**: Clear characters from LCD on EV3 brick
- **playTone**: Play tones from speaker on EV3 brick
- **beep**: Play beep from speaker on EV3 brick
- **readbutton**: Read whether button on EV3 brick is being pressed
- **writeStatuslight**: Control color and mode of status light on EV3 brick

## IR Sensors
- **Using irSensor Objects**: Connection to infrared sensor
- **irSensor**: Create connection to infrared sensor
- **readProximity**: Read distance from infrared sensor to object
- **readBeaconProximity**: Read distance and heading from infrared sensor to beacon
- **readBeaconButton**: Read number of pressed button on infrared beacon

## Motors
- **Using motor Objects**: Connection to motor
- **motor**: Create connection to motor
- **readRotation**: Read rotation from motor
- **resetRotation**: Reset rotation count to zero
- **start**: Start motor
- **stop**: Stop motor

## Color Sensors
- **Using colorSensor Objects**: Connection to color sensor
- **colorSensor**: Create connection to color sensor
- **readColor**: Read color of object in front of color sensor
- **readLightIntensity**: Read intensity of light that reaches color sensor

## Gyro Sensors
- **Using gyroSensor Objects**: Connection to gyroscopic sensor
- **gyroSensor**: Create connection to gyroscopic sensor
- **readRotationAngle**: Read rotation angle from gyroscopic sensor
- **readRotationRate**: Read rotation rate from gyroscopic sensor
- **resetRotationAngle**: Reset rotation angle to zero

## Sonic Sensors
- **Using sonicSensor Objects**: Connection to ultrasonic sensor
- **sonicSensor**: Create connection to ultrasonic sensor
- **readDistance**: Read distance from ultrasonic sensor to object

## Touch Sensors
- **Using touchSensor Objects**: Connection to touch sensor
- **touchSensor**: Create connection to touch sensor
- **readTouch**: Read touch sensor value
Simulink Support Package for LEGO MINDSTORMS EV3
Example: EV3

- Run motors open-loop, accumulate angular velocity to give angular displacement
- Run motors closed-loop – robot drives in a straight line
If your application needs…

- **Challenge**
  - Bespoke interfacing
  - Extended hardware warranty

- **Solution**
  - Consider other available support packages already available for commercial off the shelf hardware
  - Consider custom target for commercial off the shelf hardware

- **Benefit**
  - Volume pricing
  - Commercial hardware support
Extending or writing own target

- **MathWorks products:**
  - MATLAB Coder
  - Simulink Coder
  - Embedded Coder

- Use or write printf debug function that can easily be enabled/disabled using compiler switches

- Interface code which compiles on **both** host and target

- Create S-Functions to call interface code using either:
  - Legacy Code Tool
  - Or write by hand
void dbgPrintf( const char* format, ... ) {
#ifndef NDEBUG
#define BUFSIZE 1024
    va_list args;
    char buf[BUFSIZE];

    va_start( args, format );
    vsnprintf( buf, BUFSIZE, format, args );
    va_end( args );

    mexPrintf( "%s", buf );
#endif
}

MATLAB EXPO 2015
UNITED KINGDOM
Interface code

```c
switch ( inputOutput )
{
    case 1: /* GPIO input to Raspberry Pi */
        dbgPrintf( "INPUT " );
    #ifndef MATLAB_MEX_FILE
        bcm2835_gpio_fsel(*bcmPin, BCM2835_GPIO_FSEL_INPT);
    #endif
        dbgPrintf( "set\n" );
...
```
Hand written S-Function

```c
static void mdlOutputs(SimStruct *S, int_T tid)
{
    int_T inputOutput = (int_T) *mxGetPr( ssGetSFcnParam(S,0) );
    int_T raspPiGpioPin = (int_T) *mxGetPr( ssGetSFcnParam(S,1) );
    InputBooleanPtrsType uPtrs;
    boolean_T *y = malloc( sizeof( boolean_T ) );
    int_T status;

    dbgPrintf( "%f ", ssGetT( S ) );
    switch ( inputOutput )
    {
        case 1: /* Input to Raspberry Pi */
            y = (boolean_T *) ssGetOutputPortRealSignal(S,0);
            status = piInput( raspPiGpioPin, y );
            break;
        case 2: /* Output from Raspberry Pi */
            uPtrs = (InputBooleanPtrsType) ssGetInputPortSignalPtrs(S,0);
            status = piOutput( raspPiGpioPin, *uPtrs[0] );
            break;
    }
    ...
```
Legacy Code Tool

- Use a Legacy Code Tool data structure to specify
  - A name for the S-function
  - Specifications for the existing C functions
  - Files and paths required for compilation
  - Options for the generated S-function

- Use the legacy_code function to
  - Initialize the Legacy Code Tool data structure for a given C function
  - Generate an S-function for use during simulation
  - Compile and link the generated S-function into a dynamically loadable executable
  - Generate a masked S-function block for calling the generated S-function
  - Generate a TLC block file and, if necessary, an rtwmakecfg.m file for code generation (Simulink Coder product license required)
Tips for working with…

- **Phone/tablet**
  - Same Wi-Fi network
  - Antivirus/Firewall

- **Arduino**
  - Use external power supply for add on boards

- **Raspberry Pi**
  - Get a good power supply
    - $\geq 1A$, preferably 2A
  - USB cable
    - 20 AWG or larger cross-sectional area & $<3m$ works best
  - HDMI to DVI converter or cable

- **LEGO Mindstorms EV3**
  - Install provided software to reprogram the Brick
  - Consider Accelerometer
  - WiFi or LAN adaptor is necessary
MATLAB EXPO

Presentations
- MATLAB and the Internet of Things (IoT): Collecting and Analysing IoT Data
- Simplifying the Development of Computer Vision Systems
- Development and Testing of Robotic Applications Using MATLAB and Simulink

Exhibition:
- MathWorks Hardware partners
- Inverted pendulum - LEGO
Intel has come out guns a blazin' these last few months to reconquer the embedded and maker market with new products. Each generation of Atom processors has also popped up in new devices, and now Quark has also popped up...

**Projects and Tutorials**

**Communicate with LEGO MINDSTORMS EV3 Hardware**

In this example, you learn an easy way to communicate with LEGO MINDSTORMS EV3 hardware. You...

**Build a Motion Sensor Camera**

In this project, you learn to build a motion sensor camera by combining a passive infrared (PIR) sensor...

**Build a Digital Voltmeter Using Raspberry Pi**

This example makes it easy to create your own digital voltmeter application. Digital voltmeters need...

**Receiving UDP data with Arduino Due and Ethernet Shield**

This example shows how to send and receive TCP/IP or UDP messages with a remote host using an Ar...

**Receiving TCP/IP data with Arduino Due and WiFi Shield**

If you are running an Arduino Due board with WiFi Shield, you might need to send and receive TCP/IP...
Get Inspired

MATLAB and Simulink are the tools of inspiration and innovation, used at more than 5000 universities worldwide. Explore further to learn why.

For Students

For Educators

Most Popular Student Resources
- MATLAB Student
- Tutorials
  » Explore more resources for students

Most Popular Educator Resources
- MATLAB Courseware
- Webinars for instructors
  » Explore more tools for educators
Conclusion

- Interfacing to the real-world
- How to do this with MATLAB and Simulink
- Choosing most appropriate solution
  - Live demonstrations of examples
- Take home information:
  - How you can accelerate your commercial designs using low-cost hardware
  - Where to find out more
Questions?

MATLAB EXPO 2015
UNITED KINGDOM
Additional Links

Run Simulink Models on Low-Cost Embedded Hardware

Applying MATLAB and Simulink
- http://www.mathworks.com/academia

Legacy Code Tool

MAKERZONE
- http://makerzone.mathworks.com/

Other Hardware
- http://www.mathworks.com/hardware-support/home.html