Altair Embed Basic®

Model-Based Firmware Development Software

When I used C code to develop and debug my digital control algorithms, it was like I was fumbling around a twisty maze with high walls. When I switched to Embed, I got a bird's eye view of that maze and a clear path to the solution. I will never go back to C coding for my digital power and digital control applications.

Anthony Boon
Chief Engineer
ETA Electronic Design

Introducing Altair Embed Basic. The Low cost version of Altair Embed!! Use Altair Embed Professional Environment without any functionality handicap!! Explore 1000 plus microcontrollers with powerful codegen features

With 100 blocks restrictions, this version is ideal for an organization or individual getting started with model-based development with the intention of graduating to professional version as project matures and more complexity is added.

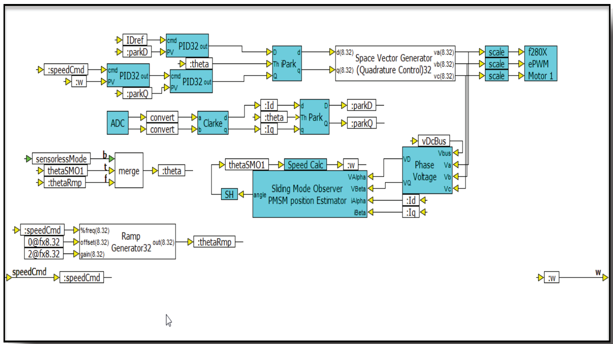
Altair Embed Basic® is a proven tool for developing embedded systems and validating designs via simulation or Hardware-in-the-Loop (HIL). With Altair Embed, you can build the most complex logic quickly, deploy it easily, and be confident it is production ready. Its model-based paradigm ensures easy validation of complex embedded logic, as well as providing deep support for thousands of popular microprocessors (MCUs) from Texas Instruments™, STMicroelectronics®, Arduino®, and Raspberry Pi®. The generated code is highly-optimized and compact, which is essential for low-cost MCUs and high-speed sampling rates.

You can extend Altair Embed to develop embedded systems on hardware not yet supported by Altair Embed using the Core Source Code Library (available separately). The generated fixed-point and floating-point code can be compiled on any platform with an ANSI C compiler

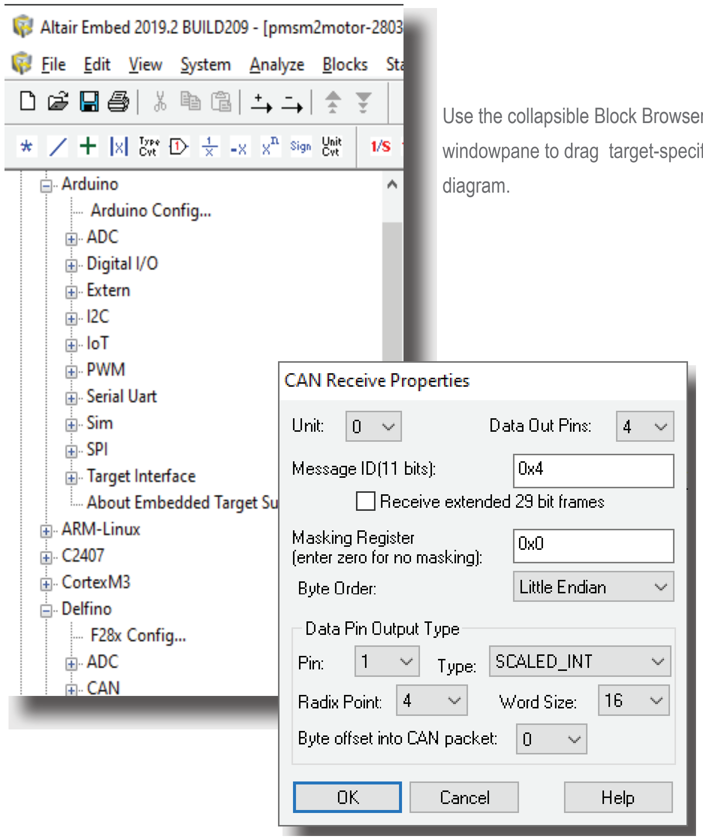
HIGHLIGHTS

- Intuitive graphical interface for simulation of embedded systems
- Rapid prototyping and code generation for Texas Instruments, STMicroelectronics, Arduino, and Raspberry Pi MCUs, DSPs, and DSCs
- Automatic programming of on-chip peripherals
- Production-quality C code with automatic scaling of fixed-point operations
- Algorithm validation using off-line simulation
- Automatic compilation, linking, and download of algorithm to the target
- JTAG hotlink for target-in-the-loop verification
- Retain the Embed GUI while the algorithm executes on the target

Altair Embed expands the power Altair Embed SE-a simulation-only version of Altair Embed-by providing automatic code generation for embedded system development.



Subsystem 1 of two sensorless PMSM motors using sliding mode observer estimation of rotor position. Sample rate is 10 kHz running both motors on a Piccolo F28036 with 50% utilization.



Use the collapsible Block Browser in the left windowpane to drag target-specific blocks to a diagram.

Right-click over a block to access customizable parameters and options.

MODEL-BASED FIRMWARE DEVELOPMENT

Using Embed, you can build a model of your entire system, including the control algorithm and the plant. The controller system can be built in scaled, fixed-point arithmetic, while the plant is built in full-precision, floating point arithmetic.

Model Construction

For model construction, Embed provides over 200 mathematical, engineering, and scientific blocks, allowing you to realize systems of any complexity. The following libraries include blocks that address specific embedded development requirements :

- TI C2000 Motor Control block library
- On-chip peripheral block libraries
- Fixed-point block library
- eMotor block library

TI C2000 Motor Control Blocks

The TI C2000 Digital Motor Control blocks are used to design motion control systems based on AC induction, brushless DC, PMSM, and stepper motors.

Embed provides both 16- and 32-bit digital motor control blocks, including PIDs, 3-Phase PWM drivers, space vector waveform generators, Park and Clarke transforms, volts-to-hertz profiles, sensorless flux and rotor speed estimation, and quadrature-encoder-based speed calculators.

Sample diagrams are included with Embed for sensored and sensorless vector control of PMSM and AC induction motors.

On-Chip Peripheral Block Libraries

The target-specific blocks let you easily program on-chip devices. These blocks include analog ADC, ePWM, eCAP (event capture), SPI, SCI (RS232 serial), I2C, digital GPIO, QEP(quadrature encoder), and CAN 2.0.

CAN Bus Support: CAN bus blocks offer an extensive range of capabilities to support the development of systems with CAN communication.

The CAN transmit and receive blocks support up to 32 CAN mailboxes on the TI C2000 series. Baud rates to 2 megabits are supported. Mailboxes are configurable from 0- to 8-byte data packet size. User-configurable addressing can be 11 or 23 bits. Remote frame requests and auto-answer are also supported.

Scaled, Fixed-Point Algorithms

The Fixed-Point block set lets you perform simulation and efficient code generation of scaled, fixed-point operations.

Overflow and precision loss effects are easily seen and corrected at simulation time. Auto-scaling speeds fixed-point development, while in-line code generation creates fast target code.

eMotor Block Library

The eMotor library consists of over 40 customizable blocks. These blocks include amplifiers, controllers, filters, loads, motors, sensors, sources, and transforms. They are implemented in floating point, making them ideal for simulating a motor and controller in Embed before downloading the code to an embedded target.

Off-line Simulation

During initial simulation of the controller and plant, you can verify, debug, and tune your control algorithms, and view the results interactively in graphical plots.

This step lets you interact with and assess the simulated controller and the simulated plant.

Automatic Code Generation

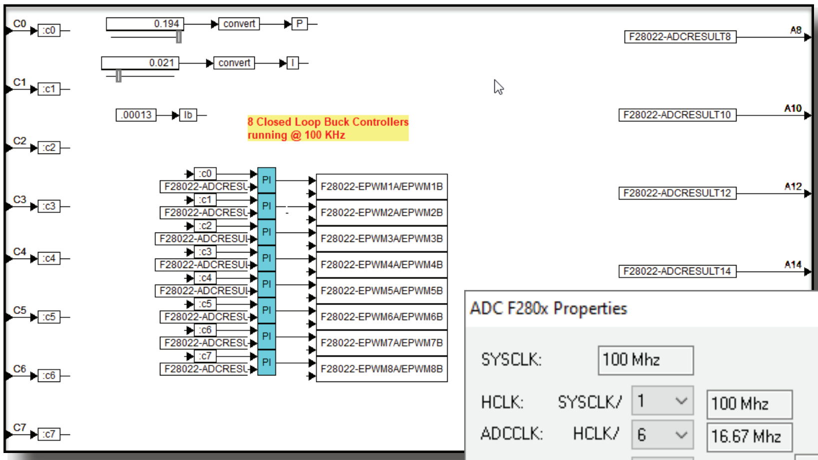
Once the model is verified, you can automatically generate code for the controller and download the code to the target device. The code is optimized for speed and memory usage.

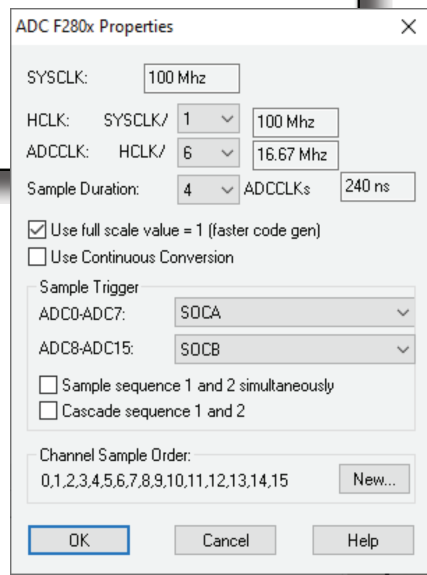
You can execute the generated code with your plant model within Embed to verify successful translation of model to code.

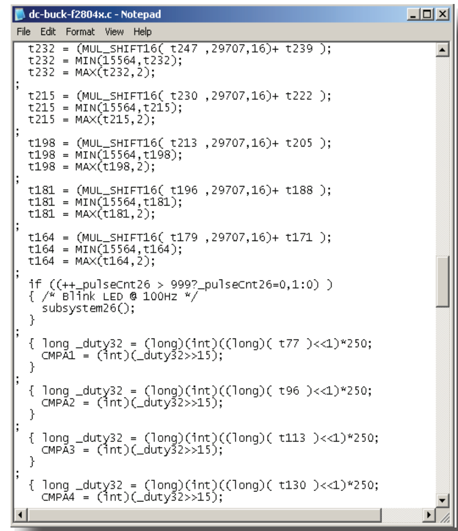
Efficient ANSI C Code

Embed generates efficient and compact ANSI C code for discrete, continuous, and hybrid systems.

Target support includes a report that displays the COFF section sizes of the







Embed model of a 400 kHz digital buck voltage converter that uses a built-in optimizer to tune the PI control against the simulated buck circuit (left). Once the controller gains are tuned, the embedded control is created with ADC input synchronized to the PWM output, along with background tasks to monitor temperature and set status LED bank (below, left). Embed auto-code generation creates the C file used to implement the control on the target (below).

generated execution file. For example, code generated for closed-loop motor control—including, PI controller, digital output, PWM, and encoder peripherals—runs at 300KHz on a 150MHz F28335 DSC. The memory footprint is:

Code size: 2095 bytes

Initialized data: 501 bytes

Uninitialized data: 504 bytes

Processor-in-the-Loop Simulation

During PIL simulation, the controller algorithm is converted to code and executed on a target MCU while the plant diagram remains in the source diagram on the host. Real-time communication between the target and host is performed via a HotLink (JTAG or serial) interface. The Embed GUI is retained while you change controller gains and plot responses from the target.

In most situations, the controller is designed within a compound block with input and output signals flowing through its pins. For most MCUs, a small footprint, low jitter, real-time operating system (RTOS) is automatically generated and included in the executable code. After the executable code with RTOS has been created, it is automatically loaded onto the target. Once loaded, the code can run in either stand-alone mode or under host control.

Hardware-in-the-Loop Simulation

During HIL simulation, the PIL is extended to include the plant hardware, sensors, and actuators. Often, however, it may not be feasible or even possible to include the entire plant, all the actuators, and all the sensors. In these situations, some of the sensors, actuators, and parts of the plant (as they become available) are included as HIL devices with the remainder simulated in models executing on the host.

Like in the PIL phase, automatic code generation of the control algorithm is loaded and executed on the target while the host is used to model the parts of the plant for which hardware is not available, as well as to interactively adjust parameters of the control algorithm and capture and present signal time history data.

The HIL phase always executes in real time. Even though the entire plant may not be actual hardware, the ability to test plant components (hydraulic pumps, lines, accumulators, actuators, electric servos, pneumatic actuators, and so on) in the controlled environment of the HIL greatly improves the level of confidence in meeting the design requirements.

Debugging Models

An Embed diagram can run directly on the PC, or you can generate code from the diagram that is linked with the Altair Visual RTOS to create an executable application that runs on a target MCU. The HotLink bidirectional communication link connects the generated application running on the target with Embed running on the PC providing HIL capability.

Generated code is syntactically error free; however you may have bugs in your diagram, depending on the algorithm memory requirements, target hardware constraints, and throughput requirements of the algorithm. Embed provides a collection of debug tools to investigate and solve each of these issues.

Execution Control

The Start, Stop, Step, and Continue controls work on the diagram executing on the PC. If the target interface block is set to synchronous operation, the target application stops and single steps as well.

Probing Signal Values

Signal values are viewed by hovering over a signal connector, or connecting a display block to any output. Likewise plots can display output traces interactively.

State Chart Breakpoints

You can set breakpoints on state events or transitions. After a breakpoint is hit, the simulation pauses and you can single step through the state chart.

Monitor Register Values

Use the Extern block to display hardware register values on the target. These values can be sent over the HotLink and displayed in plot or display blocks.

Event Statistics

You can record the number of times a compound block is executed in a simulation or target application by creating an iteration counter and sending the counter value over the HotLink.

Execution Timing

Knowing the level of target CPU utilization for each conditionally executed subsystem is important in embedded applications. Applications that consume too much CPU are prone to overframing, when not all the control functions are fully executed and completed in the sample time allotted for the controller. CPU utilization can be measured at the system level or within any conditionally executed compound block.

Digital Scopes

You can capture data on the target and send it to a buffer. Once the buffer is full, the data is sent to the PC where it can be viewed in plots.

Code Analysis Wizard

The Code Analysis wizard analyzes diagrams for issues that can cause performance degradation-including divides, matrix usage, floating-point transfer functions-in the embedded application and provides suggestions on how to improve performance when these type issues are detected.

Heap and Stack Usage and Measurement

After allowing the target to execute with ample runtime to exercise all significant functions, you can report the maximum stack and heap usage encountered during target execution.

Code Placement Control

Code can be placed in specific memory segments, and you can declare data and associate it with a memory segment using C syntax (compiler dependent).

Profile matching

The response profiles produced by applying identical command signals to both the simulation model and the target application can be compared. Any differences indicate the need for further debugging. Target application response profiles collected from the target are transmitted to the host over the HotLink interface and compared with simulation model response profiles.

INTERNET OF THINGS

Embedded development for IoT has unique considerations—peripheral programming, communication protocols, battery life awareness, OTA updates, and security—that are safely addressed with Embed.

Peripheral programming and communication protocols

With built-in support for MQTT, JSON, serial UART, I2C, SPI and HTTP, Embed makes it easy to communicate with other IoT devices. Drivers for these devices are built into the Visual RTOS. This significantly reduce the time and effort needed to develop your IoT algorithms.

Battery Life Awareness

Most IoT devices have batteries, each with their own battery life cycle. Embed has modules that track battery SOC (state-of-charge) and SOH (state-of-health), as well as algorithms to charge the battery in a way to maximize battery life.

OTA

Over-the-air firmware drivers are easily integrated into your IoT application using Embed's Extern block set.

Security

There is an explosion of IoT devices on the internet today and the ease of breaching their security is proving to be a major weakness. Altair Embed's use of protocols-like MQTT-provide encrypted data transmission secured by digital certificates. Additionally, using MCU devices that have no file system reduces the attack surface for hackers.

SOURCE CODE LIBRARIES

You may be required to run verification tools on all the code for your embedded projects. Or, your contractual obligations require 100% ownership of the source code. Embed provides (at an additional cost) two Source Code Libraries:

- The **Core Source Code Library** contains pre-compiled functions for core blocks that are not translated into inline code.
- The **Target Source Code Libraries** contain pre-compiled functions for target-specific blocks that are not translated into inline code. There is a Target Source Code Library for each target family.

These libraries are provided because Embed does not translate all core and target-specific blocks into 100% inline code. The blocks that are not translated are included as pre-compiled functions in the Source Code Libraries.

Using the Core Source Code Library for Unsupported Platforms

You can extend Embed to target unsupported platforms using the Core Source Code Library. The generated fixed-point and floating-point code can be compiled on any platform with an ANSI C compiler.

Embedded Blocks

SUMMARY OF BLOCKS

Fixed Point Blocks

abs
and
atan2
const
convert
cos
div
gain
limit
limitedIntegrator
merge
mu
not
or
PI Regulator
PID Regulator
sampleHold
shift
sign
sin
sqrt
sum
transferFunction
unitDelay
xor
-X
>
<
<=
>=
==
!=

Target-Specific Blocks

ADC10/12
Analog Comparator DAC
Analog In
Analog Input
Analog Out
Analog Output
CAN Transmit
CAN Transmit Ready
CAN Receive
Comparator
DAC
DAC12
Digital In
Digital Input
Digital Out
Digital Output

Target-Specific Blocks

DMA Enabled
eCAP
eCapPW M
ePWM
ePWM Action
ePWM Action Write
ePWM Chopper
ePWM Force Action
ePWM Force Action Write
eQEP
Event Capture
Extern Furnction
Extern Read
Extern Write
fullCompareAction
fullComparePWM
GPIO In
GPIO Input
GPIO Out
GPIO Output
HRCap
I2C Start Communication
I2C Read Buffer
I2C Write Buffer
I/O memoryRead
I/O memoryWrite
JSON
LCD
LCD Control
Monitor Buffer Empty
Monitor Buffer Read
Monitor Buffer Write
MQTT Publish
MQTT Subscribe
opAmp
PWM
QuadratureEncoder
SD16
SD16A
Segment LCD
Serial UART Read
Serial UART Write
Sigma Delta Filter Module
SPI Read
SPI Write
targetInterface
watchDog

Digital Motor Control Blocks

ACI Motor
ACI Speed Estimator
ACI Flux Estimator
Clarke Transform
Current Model
Inverse Clark Transfrom
Inverse Park Transform
Park Transform
Phase Voltage Calc
PID Regulator
QEP Speed
Ramp Generator
Resolver Decoder
SMO Position Estimator
Space Vector Generator (Mag/Freq)
Space Vector Generator (Quad Control)
Space Vector PWM
Speed Calculator
V/Hz Profile Generator

eMotor Blocks

Amplifiers
Controllers
Filters
Loads
Motors
Sensors
Sources
Tools
Transforms

WHERE TO GO FROM HERE

To learn more about Embed or Embed SE, talk to your ESS sales representative or go online to download a free trial of the software.

About Embedded Systems Solutions

Founded in 1996, Embedded Systems Solutions (ESS) has been a leading one-stop provider of hardware and software solutions for the embedded real-time systems market..

ESS is an Electro Systems Associates (ESA) group company. ESA was incorporated in 1986 by a small group of highly qualified technocrats with sound business acumen and strong technical skills.

ESS is particularly well known for in-depth expertise and vast experience with real-time embedded systems and development tools.

ESS partners with technology experts and leaders worldwide to bring together a tools ecosystem of highly integrated embedded hardware and software solutions for the Indian market.

ESS product portfolio has a range of offerings from Embedded Development Suites and Debuggers, Embedded Middleware, Embedded Security Solutions, In-Circuit Debuggers and Emulators, Flashers and Programmers, Connectivity Solutions, Protocol Analyzers, Storage Emulation Tools, Digital Storage Oscilloscopes, JTAG Boundary Scan Tools, Model-Based Design and Development Tools, Hardware Subsystems, and Single-Board Computers, among many others product offerings.

This synergetic tools ecosystem offers comprehensive solutions for design, development, and debugging processes in embedded systems.

Contact us now for more information on the Embed product line.

Phone: 98450 83528

Email: altairsales@embeddedindia.com

http://www.embeddedindia.com/

EMBEDDED SYSTEMS SOLUTIONS