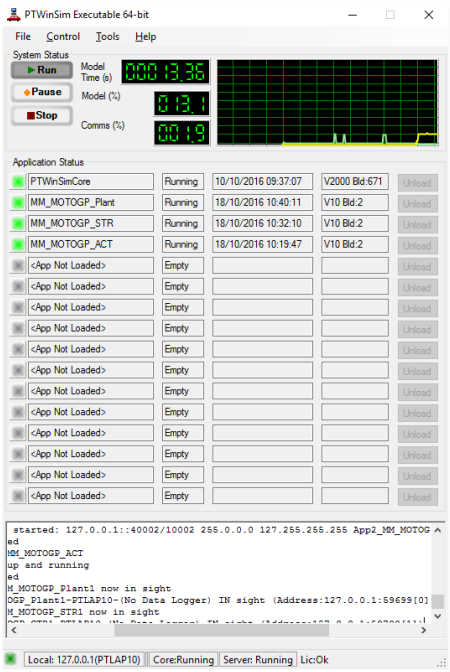
PTWinSim

Windows Based Co-Simulation Environment



PTWinSim provides a framework and supporting functionality enabling several applications to execute together in a coherent timeframe in a variety of hosts. Applications can consist of just about any functionality from full vehicle models, to hardware interfaces through to complex control algorithms or ECU code. The host system controls the execution; therefore, the models can operate in real-time or at a time to suit the host. A host may be one of the supplied windows applications (right), run as a plugin to a graphical environment or a data analysis package processing a real time telemetry stream, where the applications will execute in bursts as and when new data packets are available.

Examples of PTWinSim's implementation are:

- Creating a Virtual HIL, where models, normally executed in a separate HIL machine, execute in PTWinSim along with the ECU model(s)
  - Simulated driver model
  - Vehicle/Plant model
  - Vehicle ECU(s) - ECU models compiled for PTWinSim (Software-in-the-Loop – SIL)
- Executing control systems in real-time. For example, a driver-in-the-loop (DIL) system may consist of the following applications:
  - Hardware interface for driver controls
  - Hardware interface for driver display systems (e.g. via CAN)
  - Vehicle/Plant model
  - Vehicle ECU(s) - ECU models compiled for PTWinSim (SIL)
  - Interface to HIL system attached to external real ECUs.
  - Interface to the motion system
  - Interface to the graphics system – sending vehicle state and motion information for display, receiving terrain information for tyre contact patches.
- Executing compiled ECU control systems in Simulink (perhaps where the source models are not available), e.g. for parameter optimisation.
  - Vehicle/Plant model
  - Vehicle ECU(s) - ECU models compiled for PTWinSim (SIL)
- Executing compiled ECU control systems in MATLAB (perhaps where the source models are not available), e.g. for parameter optimisation. MATLAB solution can be compiled for external use.
  - Vehicle/Plant model
  - Vehicle ECU(s) - ECU models compiled for PTWinSim (SIL)

Sprytown, Lifton, Devon, PL16 0AY, UK. Tel:+44(1566)784508. Email: enquiries@podiumtechnology.co.uk

- Running real-time performance analysis on telemetry streams with the use of Magneti Marelli VMS in WinTax.
    - Tyre performance analysis
    - Gear-shift quality metrics
    - Interfacing to race strategy systems

One of the key advantages of PTWinSim is the ability to reuse your applications in different environments. In the above examples, the Vehicle ECU models are being used by the software developers in a virtual HIL, the vehicle dynamics engineers in the simulator, and systems/control engineers in Simulink to find optimal controller settings.

The partitioning of the applications enables a plug and play approach to selecting which applications to execute together. For the DIL example above, while keeping all other applications the same, the interface to the motion system could be removed to make it suitable for a static/workstation solution, or replaced with a different one, to interface to a different platform.

PTWinSim provides the following key services for all applications:

- Data exchanges between applications – linking the outputs of applications to the inputs of others, as well as allowing parameters to be shared between applications.
- The automotive XCP protocol over Ethernet interface: Enabling a variety of standard application tools to monitor, make tune changes and log data. These tools would already be familiar to vehicle engineers as they commonly used with automotive ECUs, such as Vector CANape, ETAS Inca, ATI Vision, dSPACE ControlDesk, Bosch Motorsport Modas etc.
- Marelli MTP Protocol over Ethernet for live monitoring and tuning of the models with Vision or Sysma.
- Logging to comma separated text files (CSV)
- Logging to Marelli WinTAX ZTX files (requires TelDataZTX licence from MM)
- Logging to remote computers via UDP packets. A sample application is provided for logging to CSV and ZTX and a C library for developing your own output formats.
- Logging to McLaren Applied's ATLAS using PTATLASRecorder.
- Logging to Cosworth Electronics' Diablo/Toolbox for live logging and analysis.
- Logging to Marelli's WinTAX Telemetry Server using PTTelRTClient for live streaming and recording.
- Logging output formats can be developed to suit your needs, sample Visual Studio projects are supplied for your own development.
- Parameter importing and exported via text files, to enable the models to be easily parameterised.
- Log file of parameter changes and/or logging of the parameter states at the start and end of execution.
- Virtual CAN buses and virtual FlexRay bus.
- Ability to break a model link at a signal write/read block and inject source data to the middle of a model.
- Proprietary API over the network to monitor, parameterise, control and log data over the network. The API is provided as a .NET library to enable users to create their own user interface.

PTWinSim is delivered with host programs to execute the applications as well as monitoring and programming tools when the host is provided by another vendor.

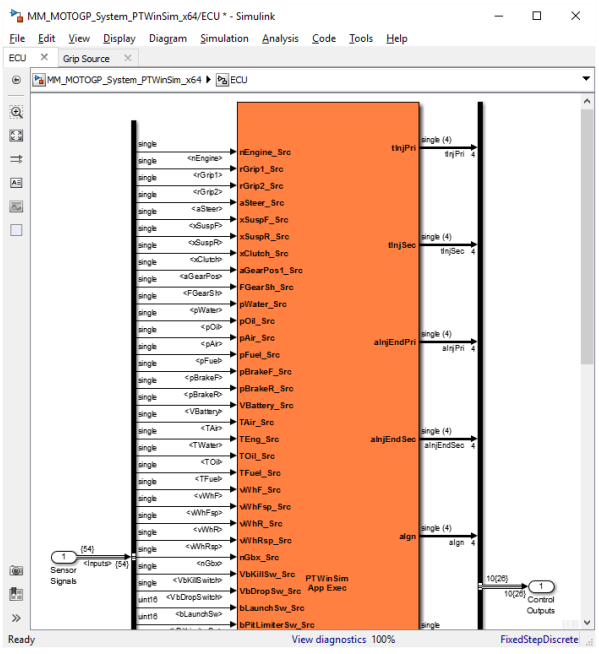
PTWinSim applications are Windows DLLs with defined entry points and description structures, they can be generated in several ways:

- From Simulink models using the Podium Technology @Source Simulink Coder target.
- Claytex's Simulator Library for Dymola models
- Bosch Motorsport's CCA/Simulation Packages
- Microsoft Visual C++, based on a supplied template.
- Functional Mock-up Interface – FMU packages
- vTAG applications (32-bit GDE/@Source and 64-bit when generated by @Source)

PTWinSim is implemented in such a way to allow all the functionality to be integrated in different host systems with the minimum effort. Some descriptions of actual implementations have been given below, please contact us if you wish to discuss how a bespoke host may be of use to your project.

Some Customer Uses of PTWinSim

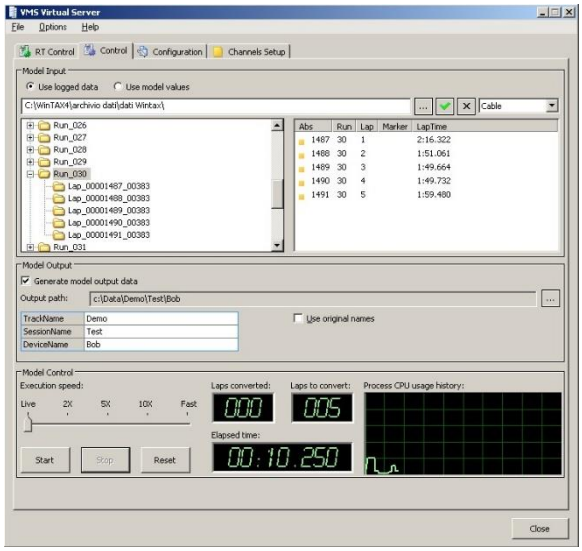
Magneti Marelli develop the unified software that run on all MotoGP bikes using Podium Technology's @Source in Simulink. Much of the development is done using PTWinSim as a virtual HIL system. The control code is built for PTWinSim along with a plant model to exercise the strategies and loaded into the PTWinSim GUI program and executed in real time. As PTWinSim supports the same MTP protocol used by the real ECU, the engineers are using the same Sysma tool chain to test, tune and verify the software in this virtual environment as they do when they finally test with a real ECU on a HIL system or track. The result is well proven and verified software without the need for multiple expensive HIL systems.

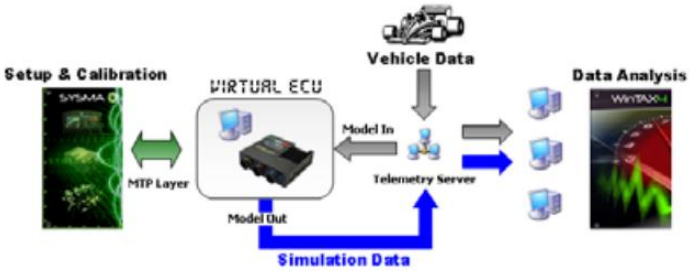


The source Simulink models for MotoGP are proprietary to Magneti Marelli and are not available to the teams, however, Marelli do release PTWinSim versions of the models to the teams, for use in their simulations. Teams will typically use the PTWinSim Simulink block that allows for the ECU strategies to be executed. The team then wrap their own bike/systems models around the block and execute it multiple time to optimise whatever system they are focussed on. Prior to each run, they would be updating the file the block uses to parameterise the model each time.

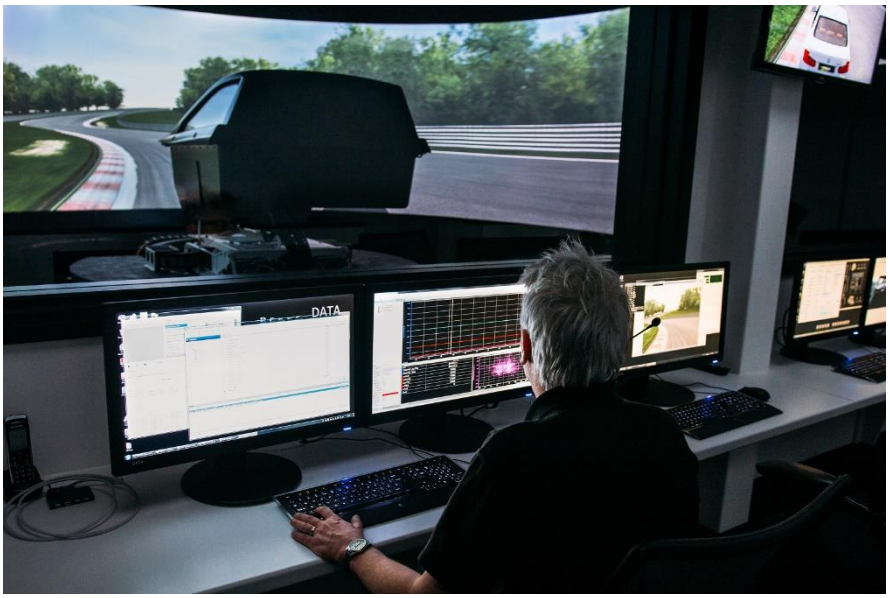
PTWinSim applications can execute in the rFpro driver-in-the-loop system using the plugin developed by Podium Technology. The plugin controls PTWinSim's execution for starting, running the model and stopping as well as giving PTWinSim applications access to the complete API provided by rFpro through a series of data structures and function call-backs. The tight integration ensures the graphics, terrain and vehicle model are always perfectly synchronised.

Marelli provide the ability to perform extensive analysis on live telemetry streams or logged data with PTWinSim applications using VMS as the host. VMS provides the inputs to the applications from the source data and then merges the application outputs to make both the source and calculated values available in WinTAX for combines viewing and analysis. As VMS controls the execution of the PTWinSim applications it runs the applications as and when data is available. Therefore, with a live telemetry stream, this is likely to be in bursts when the data reception is good. With logged data from disk, the applications are executed as fast as the data can be read from disk.





Ansible Motion design and build driver in the loop (DIL) simulators for leading automotive manufacturers and motorsport teams. PTWinSim is used in multiple instances throughout the system, the applications are providing cueing algorithms, vehicle models, hardware interfaces (CAN bus, medical, audio etc.) and interfaces to the graphics system. The use of PTWinSim has produced several benefits, among them is the ability to handle different configurations of system easily by mixing and matching applications to suit the installation. Further benefits have come from the ability to swap in and out different vehicle models from different providers for comparison and evaluation purposes.



Ansible Motion's approach allows for customers to take ownership of certain parts of the system (e.g. their own vehicle model or interfaces to the dashboard), while leaving the core functionality to be delivered and maintained by Ansible Motion.