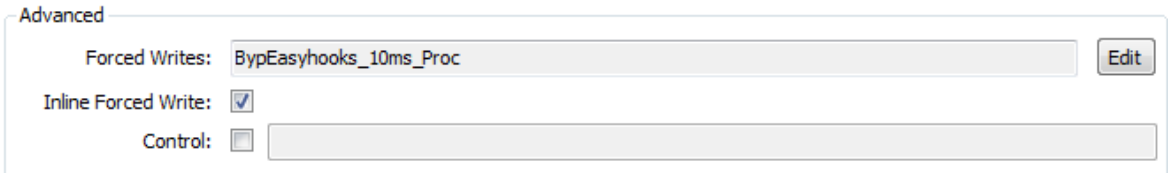
EHOOKS Forced Write

Question:

What exactly does the Forced Write option in EHOOKS?

For Variable Hooks the Forced Write option can be activated and attached to an ECU function.



Answer:

Forced Write can ensure the correct functionality of the Variable hook if the normal patch does not work like expected

The normal Patch

The EHOOKS write to an ECU variable is intended to be performed only if the ECU code would write to this variable as well. In that way we can obtain the know-how the ECU manufacturer placed here to ensure consistent values.

Example:

```
void Process(void)
{
if (condition)
{
X = 1; &lt;-- Patch

}
}
```

When EHOOKS hooks x it puts a patch at the point where x is written (the patch calls the hook function). If condition is always false the patch will never be executed and therefore the hook function will never run and so the bypass value will never be written to x.

The indicator for x is updated in the hook function. So if the indicator never changes it means that the hook function is not running.

The forced write

The first patch calls the hook function which writes to x at the beginning of the process specified in EHOOKS-DEV.

So even if condition is always false the bypass value is still written to x.

```
void Process(void)
{
&lt;-- Patch
if (condition)
{
X = 1; &lt;-- Patch
}
}
 
```

The inline forced write

For a normal forced write the user specifies the process in the GUI.

For an inline forced write EHOOKS automatically adds a patch to the start of any process that writes to the hooked variable.

The patch used for a normal forced write and an inline forced write is exactly the same. The only difference is whether the user specifies the process to be patched or EHOOKS works it out by looking for writes.

```
void Process(void)
{
uint16 t;
t = x + 1;
if (condition)
{
x = t;
}
else
{
y = t;
}
}
```

Using an inline forced write would mean that the bypass value was written to x before the statement t = x + 1. This **may** not be what the user wants.

The SIMULINK option 'forced write at the end of a dispatch point'

This option is available for each ECU Function Call Trigger Source.

The read of ECU variables and the calculation is done each time the ECU source is calculated, however the write to the ECU would be done only if the original ECU software would write to the used variables as well. This option can be used to ensure that the calculated signals are written back to the ECU at the end of each SIMULINK function-call.

Additional information:

Limitation of Forced Write

In the EHOOKS Build log a warning may indicate that a hook is possibly not working as expected:

```
WARNING(): The value written to <labelname> is cached in a register by the ECU code and
this cached value is used in a subsequent calculation. It is therefore possible that
if <labelname> is used as aninput to a calculation then the ECU calculated value may be
used as the input instead of the bypass value.
```

or

```
WARNING(0): No write information found for measurement in configuration <labelname>', the
hook has not been applied (the measurement can only be hooked using a forced write)
```

The Forced Write option increases the chance that the hook is working even though it still cannot guarantee this. Whether the hook works as expected is highly dependent on implementation of the ECU software itself and may need adaptation by the ECU supplier.

In case of further questions:

Please feel free to contact our Support Center, if you have further questions.

Here you can find all information: http://www.etas.com/en/hotlines.php