Application Note: AN10106

How to embed XTA commands in your program

This application note is a short how-to on programming/using the xTIMEcomposer tools. It shows how to embed XTA commands in your program.

Required tools and libraries

This application note is based on the following components:

- xTIMEcomposer Tools - Version 14.0.0

Required hardware

Programming how-tos are generally not specific to any particular hardware and can usually run on all XMOS devices. See the contents of the note for full details.

1 How to embed XTA commands in your program

The XTA can be used interactively to investigate and measure the timing properties of your program. Once satisfied, it is often a requirement to capture these timing requirements thus allowing the timing of your program to be automatically re-validated on future modifications. This can be achieved by embedding XTA commands into your source files using the *xta command* pragma.

For example, compile the following code:

```
#include <stdlib.h>
#include <xs1.h>

port p1 = XS1_PORT_1A;
port p2 = XS1_PORT_1B;

int main() {
  int x;

  #pragma xta endpoint "input"
  p1 :> x;

  // Checks for errors..
  if (x == 1) {
    #pragma xta label "error_case"
    exit(1);
  }

  // do some computation here..

  #pragma xta endpoint "output"
  p2 <: 0;
  return 0;
}
```

Assume that there is a timing requirement between the *input* and the *output* of 100.0 ns. Assume also that you are not interested in the timing of the *error_case*.

Add the following lines to the source file:

```
#pragma xta command "analyze endpoints input output"
#pragma xta command "set exclusion - error_case"
#pragma xta command "set required - 100.0 ns"
```

The above commands will get run when the binary is next loaded into the XTA, both from the command line or from within the xTIMEcomposer Studio. Also, if a binary contains embedded XTA commands, then the XTA is automatically called by XCC (the compiler driver) as part of the build.