Application Note: AN10014

How to examine the stack contents

This application note is a short how-to on programming/using the xTIMEcomposer tools. It shows how to examine the stack contents.

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 examine the stack contents

XGDB can be used to examine the contents of the stack memory at a given point in time. For example, compile the following code ensuring that debug is enabled (-g):

```
int add1(int x) {
  return x + 1;
}

int sub1(int x, int y) {
  return add1(x) - add1(y);
}

int main() {
  sub1(6, 3);
  return 0;
}
```

2 From within xTIMEcomposer Studio

Create a new debug configuration via *Run->debug Configurations->xCORE Applications*. Set a breakpoint at the start of *add1* then start debugging. Execution will now break when *add1* is reached. The contents of the stack can be viewed from the *Debug* view. Note: Clicking on the previous stack frames in the *Debug* view causes the *Variables* view to be updated with the variables in scope for this stack frame.

3 From the command line

On the command line, stack contents can be displayed using the *backtrace* command. For example, start XGDB, connect to the simulator and set a breakpoint on *add1*. When run, execution will break at the start of *add1*. You can display the stack contents using the *backtrace* command as follows:

```
> xgdb a.xe
...etc...
(gdb) connect -s
0xffffc04e in ?? ()
(gdb) b add1
Breakpoint 1 at 0x100b2: file examining_the_stack.xc, line 9.
(gdb) r
...etc...
Breakpoint 1, add1 (x=6) at examining_the_stack.xc:9
9       return x + 1;
(gdb) backtrace
#0  add1 (x=6) at examining_the_stack.xc:9
#1  0x000100ca in sub1 (x=6, y=3) at examining_the_stack.xc:13
#2  0x000100e4 in main () at examining_the_stack.xc:17
```