Application Note: AN10045

How to periodically perform an action using a timer

This application note is a short how-to on programming/using the xTIMEcomposer tools. It shows how to periodically perform an action using a timer.

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 periodically perform an action using a timer

Timers can be used to periodically perform an action. First input the current time from the timer:

```
t :> time;
```

The following loop uses the `timerafter` predicate to print a message once a second 100 times:

```
for (int i = 0; i < 100; i++) {
  t when timerafter(time) :> void;
  time += XS1_TIMER_MHZ * 1000 * 1000;
  printstr("This message is printed once a second\n");
}
```

If you want to periodically perform an action while also responding to other events, you can use a loop containing a `select` statement:

```
for (int i = 0; i < 100; i++) {
  select {
  case t when timerafter(time) :> void:
    time += XS1_TIMER_MHZ * 1000 * 1000;
    printstr("This message is printed once a second\n");
    break;
  // Insert cases to handle other events here...
  }
}
```

Note in the above examples the time input from the timer is discarded in the loop by inputting it to `void`. To see why this form is the preferred solution, suppose you had instead written the following:

```
for (int i = 0; i < 100; i++) {
  t when timerafter(time) :> time;
  time += XS1_TIMER_MHZ * 1000 * 1000;
  printstr("This message is printed once a second\n");
}
```

Because the processor completes the input shortly after the time specified is reached, the input in the loop may actually increment the value of time by a small amount. This amount may be compounded over multiple loop iterations, leading to drift over time.