Application Note: AN10002

How to use alias/local pointers

This application note is a short how-to on programming/using the xTIMEcomposer tools. It shows how to use alias/local pointers.

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 use alias/local pointers

Local pointers do not have the same restrictions as restricted pointers. Any pointer declared as a local variable is an *alias* pointer unless specified otherwise.

Alias pointers can be copied and changed to point at different object. In general you can use them like C pointers.

The main restriction on alias pointers is that, although they can be copied, an alias pointer cannot be copied to a pointer with a *larger* static scope. For example the following code would be invalid:

```
int * alias z;

void f() {
{
  int y[10];
  int *x = &y[0];
  z = x;                  <--- invalid since z would point to y
                               after y has been deallocated
}
```

To avoid indirect copying to a larger scope, you cannot have pointers to, arrays containing, or structures containing alias pointers. Alias pointers are also not allowed to be transferred between tasks.