Cambridge Consultants

XAP

Getting Started with xIDE

Cambridge Consultants Limited
Science Park
Milton Road
Cambridge
England
CB4 0DW

+44 (0) 1223 420024

xap@CambridgeConsultants.com

www.CambridgeConsultants.com

ASICs-TM-002 v1.7
8 December 2014

Revision History

Revision	Date	Details
v0.1	4 January 2008	Initial release
v1.0	23 January 2008	Minor corrections and changes following review
v1.1	3 July 2008	Minor corrections and update to Legal Notices
v1.2	7 July 2008	Replaced outdated screen shot
v1.3	8 July 2008	Added copyright notice to all pages
v1.4	9 July 2008	Added copyright notice to section 10
v1.5	28 July 2008	Added notes and pitfalls about turning debug off and XAP5 far mode. Also copyright message on front page
v1.6	8 October 2014	Updated to include XAP6
v1.7	8 December 2014	Formatting changes

Legal Notices

This is an Application Note for XAP processors. You may use this if you accept the following conditions. If you do not accept these conditions, you must delete or dispose of this copy of the Application Note immediately.

Copyright: This Application Note is © Copyright Cambridge Consultants 2005-2014. You are authorised to open, view and print any electronic copy we send you of this Application Note within your organisation. Printouts of this Application Note must be kept within your organisation. Distribution of this Application Note, in whole or in part, to anyone outside your organisation is prohibited without prior written permission from Cambridge Consultants Ltd.

Fit for purpose: The XAP processors and the xIDE software development environment must not be used for safety critical and/or life support applications without a specific written agreement from Cambridge Consultants Ltd.

Liability: Cambridge Consultants Ltd. makes no warranty of any kind with regard to the information contained in this Application Note, including but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Cambridge Consultants Ltd. shall not be liable for omissions or errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material. The information contained herein may be updated from time to time.

Ownership and Licence: The XAP4 and XAP5 processors (and associated technologies including SIF and xIDE), XAP4 and XAP5 documentation and the xIDE software development environment and all intellectual property rights associated with them are proprietary to and owned by Cambridge Consultants Ltd and are protected by copyright law and international copyright treaties, patent applications, letters patent and other intellectual property laws and treaties. You must not attempt to use the XAP4 and XAP5 processors or the xIDE developer's environment unless you have a valid licence agreement with Cambridge Consultants Ltd. You must not use the information within this Application Note or other items supplied to you for the purposes of designing, developing or testing a device or simulator which is, or is intended to be, wholly or partly instruction set compatible with the XAP4 or XAP5 processor.

Trademarks: Cambridge Consultants, the Cambridge Consultants logo and XAP are registered trademarks and XAP1, XAP2, XAP3, XAP4, XAP5, SIF and xIDE are trademarks of Cambridge Consultants Ltd. All other trademarks mentioned herein are the property of their respective owners.

Cambridge Consultants Ltd
Science Park
Milton Rd
Cambridge CB4 0DW
England

www.CambridgeConsultants.com

Project Team

With thanks to the XAP4, XAP5 and XAP6 project teams: Alistair Morfey, Karl Swepson, Derek Henderson, Peter Lloyd, Chris Roberts, Martin Cooper, Alan Egan, Eric Wieser, Chris Turner, Saajan Chana, Hereward Mills, Nimrod Gileadi, Rodrigo Queiro, James Crosby and Jerome Woodwark.

Table of Contents

1 Introduction

1.1 This Document

This Application Note describes how to create, run, debug and evaluate your first project using xIDE for XAP4, XAP5 and XAP6. The aim of this document is to help potential users to evaluate the XAP4, XAP5 and XAP6 processors and the associated tools.

This document provides a brief tour of the xIDE Integrated Development Environment for XAP4, XAP5 and XAP6. Further documents describe XAP4, XAP5 and XAP6 in depth. Refer to section 1.3, "Other Documentation" for details.

1.2 Assumptions

The reader is assumed to be familiar with microprocessor architectures in general, and to have a good understanding of the C language. A detailed understanding of digital electronics is not required.

1.3 Other Documentation

The following documents describe the xIDE integrated development environment:

Document	Contains
xIDE User Manual C7066-TM-001	Describes the general features of xIDE.
xIDE Python Object Model C7066-TM-003	Describes the Python object model used within xIDE. This is essential for developers wanting to use the scripting features of xIDE. This allows developers to perform automated testing or to model other parts of a system during simulation.

The following documents describe the XAP4 software tools and xIDE integrated development environment:

Document	Contains
xIDE for XAP4 C7066-TM-017	Describes specific features of the XAP4 plugin for xIDE.
XAP4 GNU Binutils Manual C7432-UM-003	Describes the XAP4 Assembler, Linker and other binary utilities.
XAP4 GCC Compiler Manual C7432-UM-004	Describes the XAP4 ANSI C compiler
XAP4a Instruction Set Quick Reference Card C7432-UM-005	A short guide to the XAP4 instruction set.

The following documents describe the XAP5 software tools and xIDE integrated development environment:

Document	Contains
xIDE for XAP5 C7066-TM-019	Describes specific features of the XAP5 plugin for xIDE.
XAP5 GNU Binutils Manual C7508-UM-003	Describes the XAP5 Assembler, Linker and other binary utilities.
XAP5 GCC Compiler Manual C7508-UM-004	Describes the XAP5 ANSI C compiler
XAP5a Instruction Set Quick Reference Card C7508-UM-005	A short guide to the XAP5 instruction set.

The following documents describe the XAP6 software tools and xIDE integrated development environment:

Document	Contains
xIDE for XAP6 C7066-TM-027	Describes specific features of the XAP6 plugin for xIDE.
XAP6 Binutils Manual C7920-UM-003	Describes the XAP6 Assembler, Linker and other binary utilities.
XAP6 GCC Manual C7920-UM-004	Describes the XAP6 ANSI C compiler
XAP6a Instruction Set Quick Reference Card C7920-UM-005	A short guide to the XAP6 instruction set.

For details of the XAP4 hardware, refer to these documents:

Document	Contains
XAP4a Hardware Reference Manual C7432-UM-011	Information needed by digital designers using the XAP4a in an FPGA or ASIC
XAP4a Hardware Quickstart Guide C7245-UM-001	A short guide on how to get started in synthesis and simulation of XAP4a.
XAP4 Datasheet ASICs-SB-011	A short overview of the XAP4 architecture.

For details of the XAP5 hardware, refer to these documents:

Document	Contains
XAP5a Hardware Reference Manual C7508-UM-011	Information needed by digital designers using the XAP5a in an FPGA or ASIC
XAP5a Hardware Quickstart Guide C7508-UM-001	A short guide on how to get started in synthesis and simulation of XAP5a.
XAP5 Datasheet ASICs-SB-012	A short overview of the XAP5 architecture.

For details of the XAP6 hardware, refer to these documents:

Document	Contains
XAP6a Hardware Reference Manual C7920-UM-006	Information needed by digital designers using the XAP6a in an FPGA or ASIC
XAP6a Quickstart Guide C7920-UM-001	A short guide on how to get started in synthesis and simulation of XAP6a.
XAP6 Datasheet ASICs-SB-017	A short overview of the XAP6 architecture.

1.4 For Further Help

The XAP documentation set provides answers to most questions.

If a problem remains unsolved, contact Cambridge Consultants technical support at:

xap@CambridgeConsultants.com

2 Installation

You will need to install xIDE and GNU packages for the processor you going to evaluate.

For the rest of this document we use XAP to mean XAP4,XAP5 or XAP6.

xIDE for XAP

This package contains the Integrated Development Environment, the instruction set simulator, a standard C library and some example projects.

The xIDE application is protected by a license manager. You will need to obtain a licence file from Cambridge Consultants before you can use xIDE.

You can tell if you have a licence by launching xIDE. If the main application starts you have a valid licence. You can see details of the licence in the About dialog.

If you don't have a licence, a wizard will appear to guide you through requesting and installing the licence file. At the end of the wizard, your email client will be opened containing a new message. Some email clients will contain the text for the message. If you client does not contain the text, you will need to paste the message from the clipboard.

After you have sent the email, we will send you a licence file. You will then install the licence file in the `licence` directory of the xIDE for XAP installation.

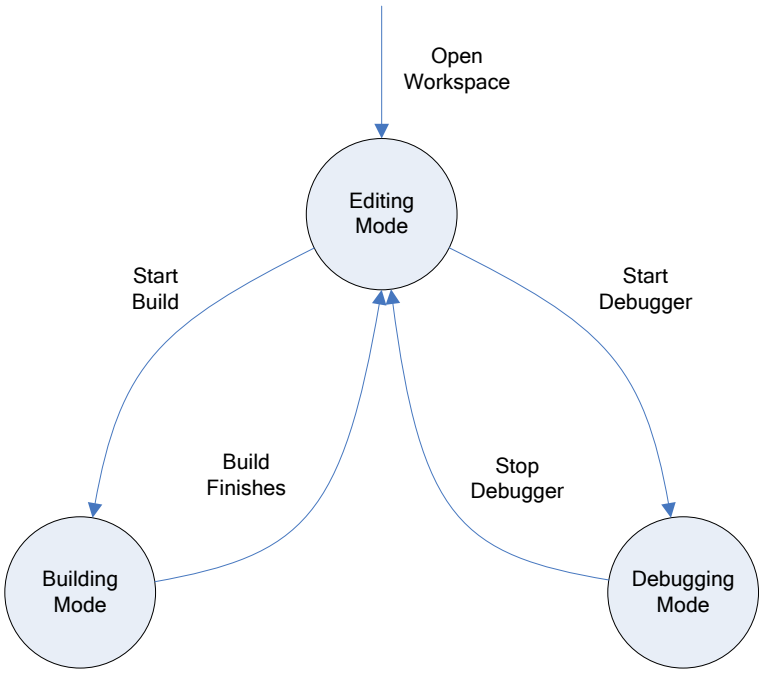
GNU for XAP

This package contains ports of GNU GCC and GNU binutils. These provide the complier, assembler and linker functions used by xIDE for XAP.

3 xIDE Overview

The xIDE application is used to edit, build and debug applications that run on XAP processors. A project contains one application. The project contains a list of source files, settings for building and settings for debugging. Projects are contained with xIDE workspaces.

The xIDE application can be in one of three modes.



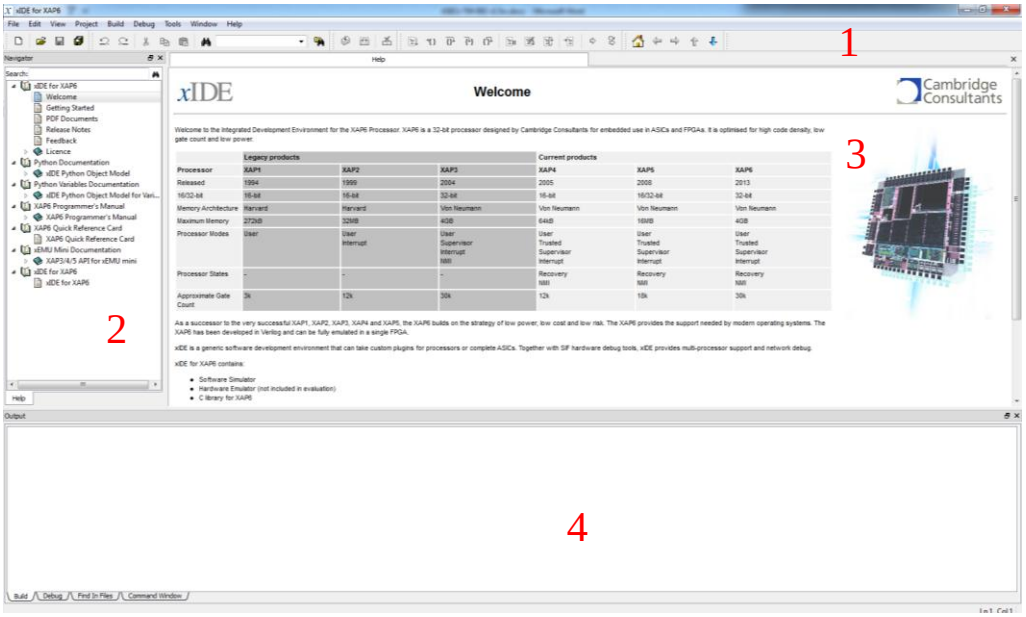
Mode	Description
Editing	From this mode source code can be modified, project properties can be changed, builds can be started and the debugger can be started
Building	xIDE is in this mode when the build is running. When a build is in progress it is not possible to start a debug session or to modify project settings.
Debugging	xIDE is in the mode when the debugger is active. In this mode project properties cannot be changed and builds cannot be started.

In Debugging mode, xIDE will be either communicating with real XAP hardware or with xIDE's simulation of a XAP. The same xIDE is used for debugging simulations, hardware emulators, ASICs, FPGAs and final products.

4 Open a Project

Some example projects are installed as part of xIDE for XAP. Some of these projects run on hardware and some run on the instruction set simulator that is part of xIDE for XAP.

When you launch xIDE you will see an application similar to the one shown below.



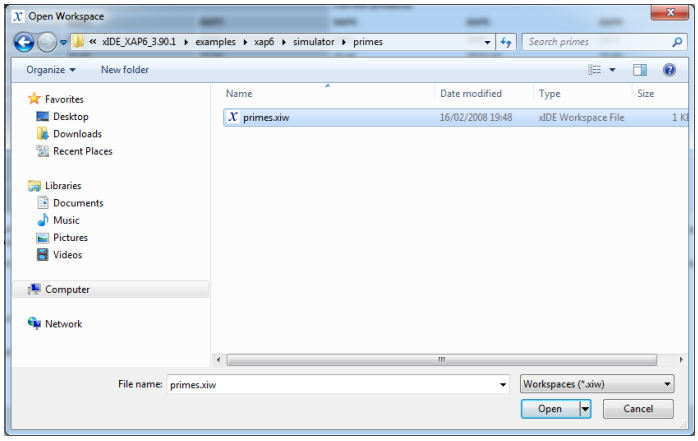
We use the word widget to describe a region in the GUI.

1	Menus and toolbars widgets. These provide access to the commands that control the application. The toolbars present a subset of the functionality of the menus.
2	The navigator widget. This provides access to the online documentation. When a workspace or project is opened a new tab appears at the bottom of the navigator widget providing access to the project files and settings.
3	The workspace widget. This widget displays help files and the text editor for editing source files. Additional tabs appear at the top of the workspace widget as new documents are opened.
4	The output widget. There are four tabs at the bottom of this widget. These tabs

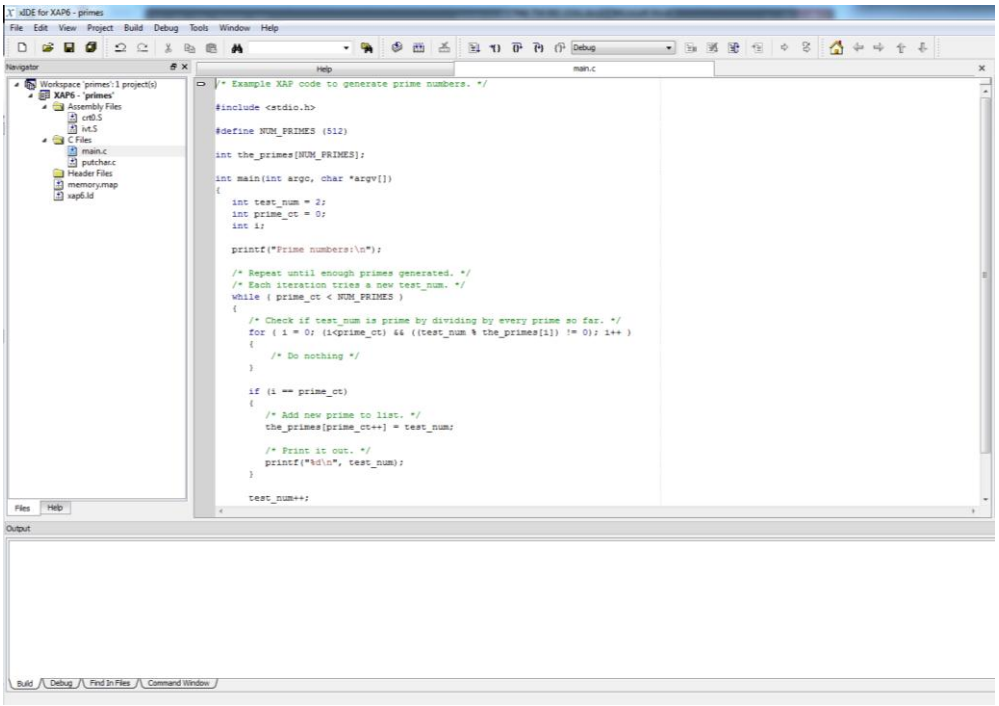
	provide access various output information from xIDE, such as build and debug messages. The command tab provides access to the embedded Python command line interpreter, which can be used to automate the xIDE.

We are now going to open a project that runs on the instruction set simulator.

From the Project menu choose Open Workspace. This opens a standard file chooser dialog like the one below. Within your "My Documents", browse to `Cambridge Consultants\xIDE_XAPn_{version}\examples\xapn\simulator\primes` and open the `primes.xiw` file.



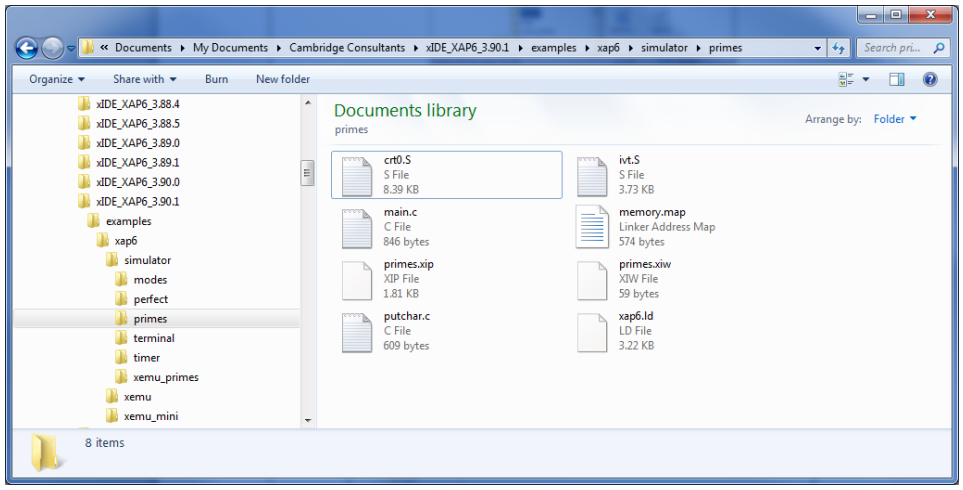
The loaded workspace contains one project as shown in the Navigator widget below. Expand the folders in the project and double click on `main.c` to open the file in the editor.



When the project is first opened xIDE is in Editing mode. For more information see section 3, "xIDE Overview".

The files in the project can be opened by double clicking on the icons in the Navigator widget.

The folder icons in the Navigator provide a way of grouping related files. These need not be the same as the layout in the file system. The contents of the project directory are shown below.



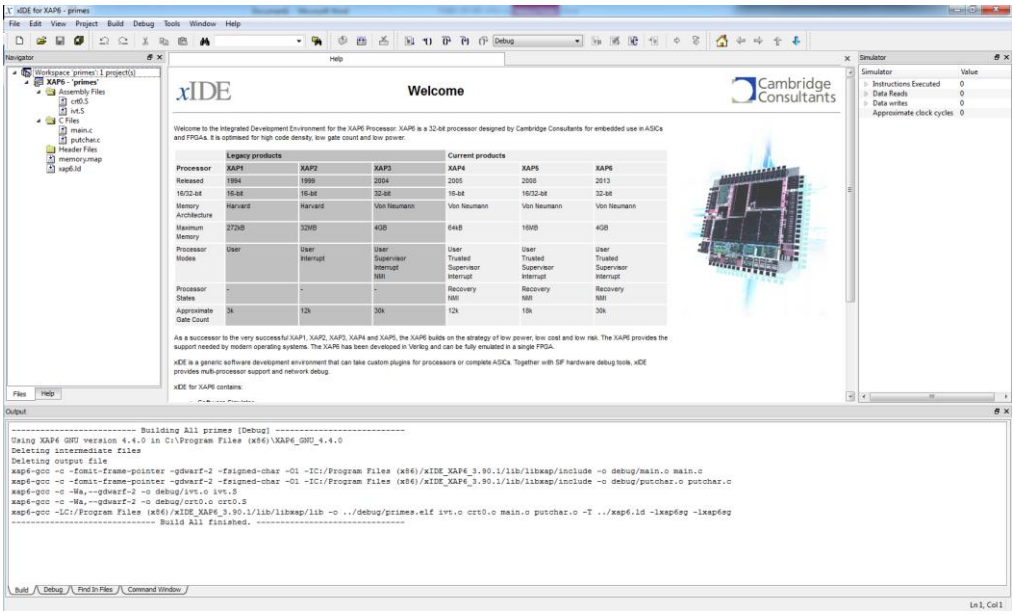
The purposes of the files in the project directory are explained below.

File	Function
`primes.xiw`	xIDE Workspace file
`primes.xip`	xIDE Project file
`crt0.S`	Assembly file. This file contains the standard code that runs on the XAP before the C entry point, `main()` is called. This file is usually in all XAP projects.
`ivt.S`	Assembly file. Defines the functions used for interrupts and exceptions. This file is usually in all XAP projects.
`main.c`	C file. This contains the main application code.
`putchar.c`	C file. Provides the `putchar()` function used by the application.
`xap4.ld`	Linker script. This file is used by the linker and defines where in memory the project will be loaded. All projects require a linker script.
`memory.map`	Memory map. This file is used by the simulator and defines the sizes and positions of memory available to the XAP simulation. All projects require memory map file.

5 Build a Project

From the Build menu choose Rebuild All to build the project. Details of the build steps are reported in the Output widget as shown below.

Any error messages will also be reported here. Double clicking on an error message that contains source file and line number information causes the offending line to be displayed in the editor.



When a build is in progress xIDE is in Building mode. For more information see section 3, "xIDE Overview".

Projects contain at least one configuration. Configurations provide a way of storing multiple sets of project properties in a single project. The example projects in xIDE contain two configurations: Debug and Release. The Debug configuration produces code that is easy to debug because much of the compiler optimisation is disabled. The Release configuration produces code that is smaller and faster because compiler optimisation is enabled.

The active configuration can be changed either by using the combo box on the tool bar or from the Build menu select Configurations.

6 Debug a Project

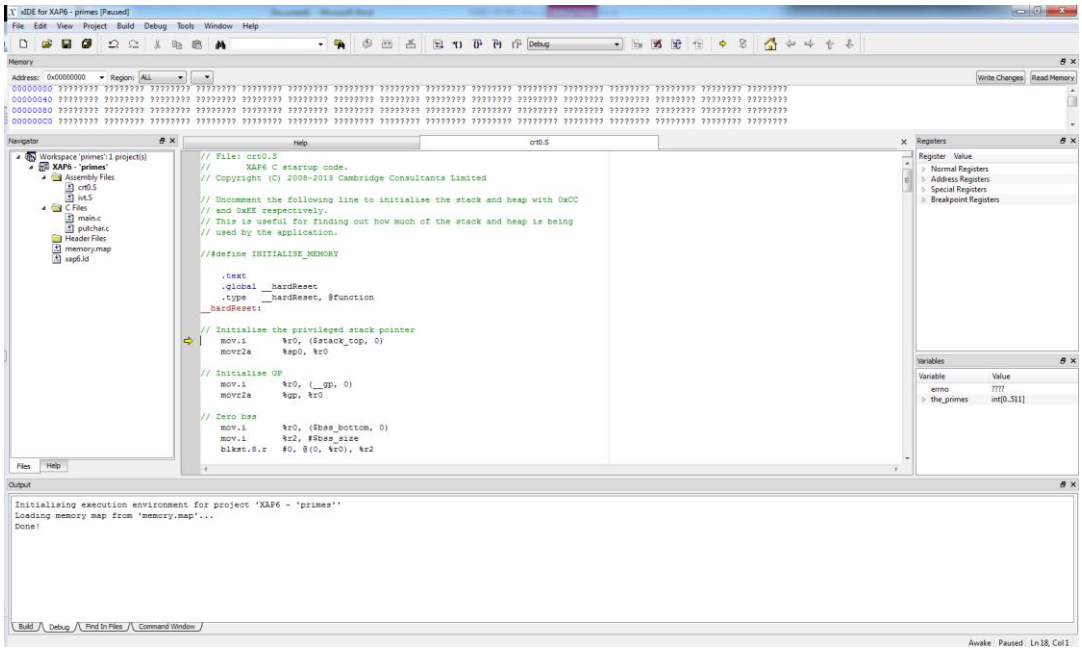
From the Debug menu choose Restart to begin a debug session. This initialises the processor simulation by loading the project into memory and resetting the processor.

When a debug session is active xIDE is in Debugging mode. For more information see section 3, "xIDE Overview".

The yellow arrow in the Marker widget to the left of the editor indicates the next line of code to be executed. In this example the yellow arrow is showing the very first instruction to be executed after reset, which is part of the C initialisation code. This code eventually calls the C entry point, `main()`.

In the picture below you can see two new widgets: Memory and Registers. If these are not visible on your system you can make them visible: from the View menu choose Debug Windows.

This menu contains more widgets you can use to help debug your project.



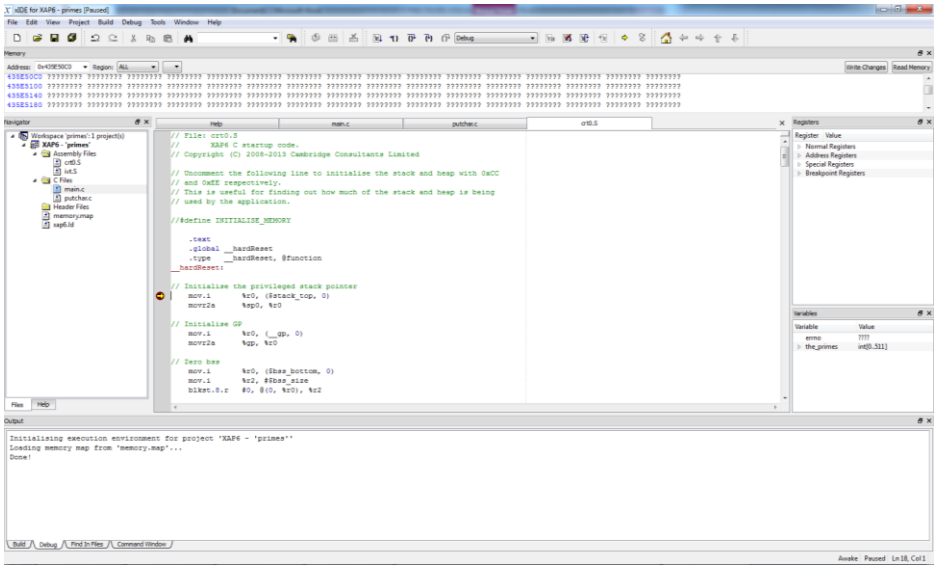
In this example, the `main()` function is in the `main.c` file. We are going to set a breakpoint at the beginning of `main()` and make the processor run to the breakpoint.

Open the `main.c` file by double clicking the `main.c` icon in the Navigator widget. Below is a list of the marker icons used in the editor pane.

- All breakpoints at this source line are enabled, one of which is in the current project
- All breakpoints at this source line are disabled, one of which is in the current project
- There is a combination of enabled and disabled breakpoints at this source line, one of which is in the current project
- All breakpoints at this source line are enabled, none of which are in the current project
- All breakpoints at this source line are disabled none of which are in the current project
- There is a combination of enabled and disabled breakpoints at this source line, none of which are in the current project
- Next statement marker used during debugging
- Cursor position marker showing the line currently containing the cursor

If you click on a source line that hasn't generated any XAP instructions, such as a comment, xIDE will automatically insert a breakpoint lower down the source file.

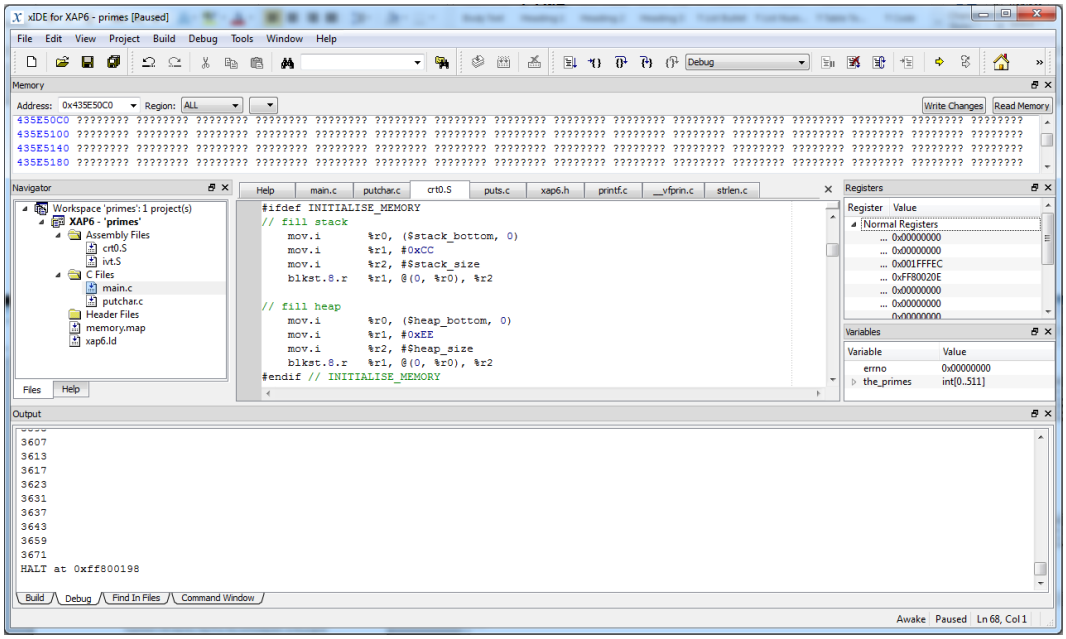
From the Debug menu choose Run. The processor will now execute the C initialisation code and stop at our breakpoint. This is shown below.



The Debug menu also contains Step Into and Step Over. These can be used to manually step through the source lines. Step Into will take you into any called functions as they are encountered. Step Over will not step into any called functions.

Experiment with Step Into, Step Over, Run and breakpoints. As the program continues you will see a list of prime numbers being displayed in the Output widget.

The program finishes back in the C startup file that we started in.

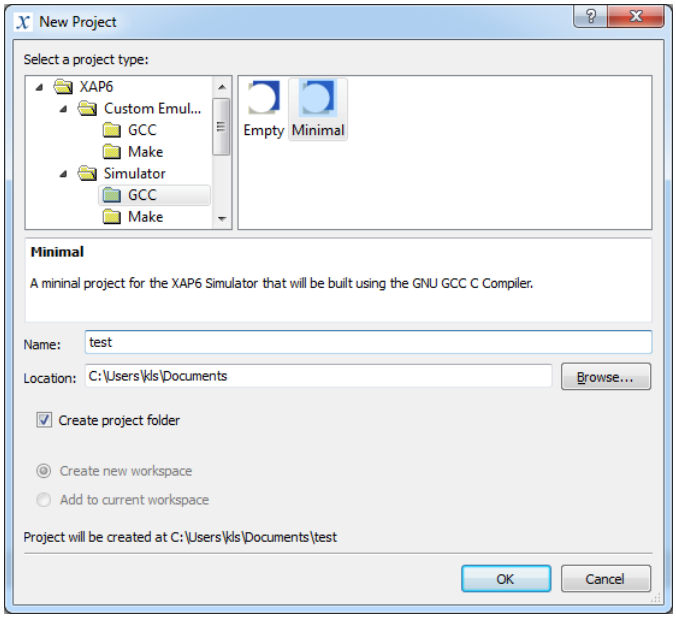


From the Debug menu choose Stop to finish the debug session.

7 Create a Project

From the Project menu choose New. Then click on the GCC folder icon under Simulator and select the Minimal icon. This selects a project that will run on the instruction set simulator and will be built by xIDE using the default GCC tool chain.

Select a location and name for your project and click OK.



The newly created project will be loaded into xIDE and is ready for editing and building. The new project will contain a complete XAP application including a `main.c` file that contains the standard C entry point, `main()`.

Add some new code to `main.c`. For example:

```
int main(int argc, char* argv[])
{
    int i;
    for (i=0; i<100; i++)
    {
        printf ("Number is %d\n", i);
    }
    return 0;
}
```

You should now be able to build and debug your new project.

8 Evaluate a Project

Measurements of application size and execution speed are significantly affected by the compiler optimisation settings. The default projects created by xIDE have two configurations: Debug and Release. The Release configuration uses compiler optimisation settings that give a good trade-off between code size and execution speed. The Debug configuration uses a less aggressive optimisation setting that results in larger code, but produces better debug information.

8.1 Measure Memory Usage

The code and data for a XAP program is grouped into sections. The sections have defined purposes and can be used to calculate ROM and RAM requirements.

For the purposes of this section we are going assume a simple memory model where all code and constants are stored in a ROM and RAM is used for variables.

Other configurations are possible and supported by the tools. For example, some systems may run entirely from RAM, which is initialised at power up by a second processor.

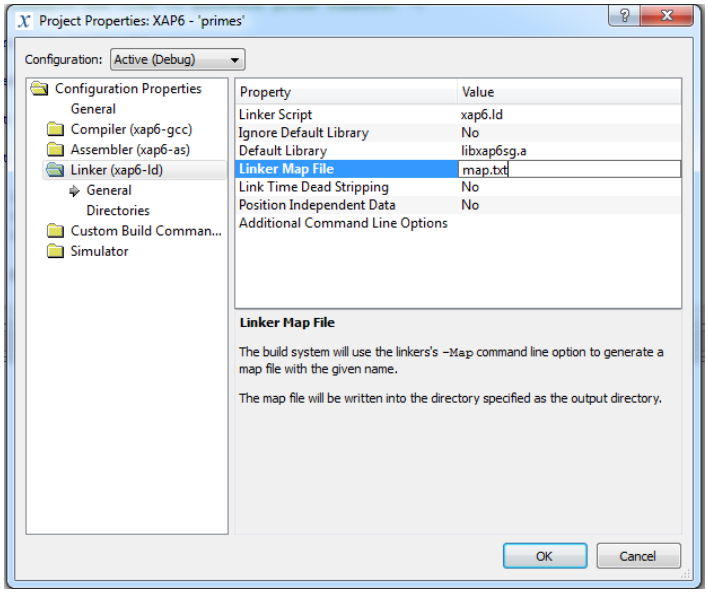
Section	Function	ROM	RAM
`.vp`	Vector pointer initialisation value. This defines where the `.vectors` section is located.	Yes	
`.vectors`	Vector table for interrupts and exceptions.	Yes	
`.text`	XAP instructions.	Yes	
`.rodata`	Statically allocated constants.	Yes	
`.data`	Initialisation constants for initialised statically allocated variables.	Yes	Yes
`.bss`	Zero initialised statically allocated variables.		Yes

Two ways of measuring code size are described below; both will produce the same result.

Linker Map File

Make sure the debug session is not active.

From the Project menu choose Properties. This opens the dialog that can be used to control all project settings.



Add a `map.txt` to the Linker Map File property as shown above. This instructs xIDE to generate a file called `map.txt` containing detailed information about memory usage each time the application is built.

For the default project settings for the Release configuration, the file will be placed in the `release` directory within the project directory.

From the Debug menu choose Rebuild All. From the File menu choose Open. Select `*.*` as the file type then browse into the `release` directory and open `map.txt`.

A complete linker map file is given in section 10, "Linker Map File". The results contained in the file are:

Section	ROM	RAM
.vp	0x02	
.vectors	0x60	
.text	0x2EF0	
.rodata	0x5A	
.data	0x12	0x12
.bss		0x4
TOTAL	0x2FBE	0x16

Command Line Tools

Before using the command line tools you will need to make sure that two environment variables are set correctly.

Variable	Requirement
`PATH`	This system variable contains a semicolon separated list of directories to search for executables called from the command line. This list needs to include the `bin` directory of the GNU for XAP installation.
`XAP4GCC_ROOT` or `XAP5GCC_ROOT`	This variable contains the installation directory of GNU for XAP4 or GNU for XAP5 as appropriate. GNU for XAP6 does not require an equivalent environment variable.

Further information is in the Installation section of the XAP GNU Binutils Manual.

Open a command prompt and change to the output directory of the project. For the Release configuration the default setting is the `release` directory in the project directory.

The `xapn-objdump` program can be used to display the sizes of the sections in the project.

The section names and sizes are highlighted below in **bold**. Equivalent values are highlighted in the linker map file in section 10, "Linker Map File". Both methods produce the same results.

```
C:\>xap6-objdump -h myproject.elf

myproject.elf:     file format elf32-xap4

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .vectors      00000060  00008400  00008400  00001400  2**1
                  CONTENTS, ALLOC, LOAD, READONLY, DATA
  1 .rodata       0000005a  00008460  00008460  00001460  2**1
                  CONTENTS, ALLOC, LOAD, READONLY, DATA
  2 .data         00000012  00000002  000084ba  00002002  2**2
                  CONTENTS, ALLOC, LOAD, DATA
  3 .bss          00000004  00000014  00000014  00001014  2**2
                  ALLOC
  4 .text         00002ef0  000084cc  000084cc  000024cc  2**1
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  5 .vp           00000002  0000fffe  0000fffe  00005ffe  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  6 .comment      00000444  00000000  00000000  00006000  2**0
                  CONTENTS, READONLY
```

8.2 Measure Execution Time

From the View menu choose Debug Windows then Simulator. This widget contains details of the number of instructions, clock cycles and memory accesses made by the instruction set simulator as it executes its program.

The values in the widget can be set by double clicking on the number and entering a new value, say zero. This is useful for measuring how long a key area of code takes to execute.

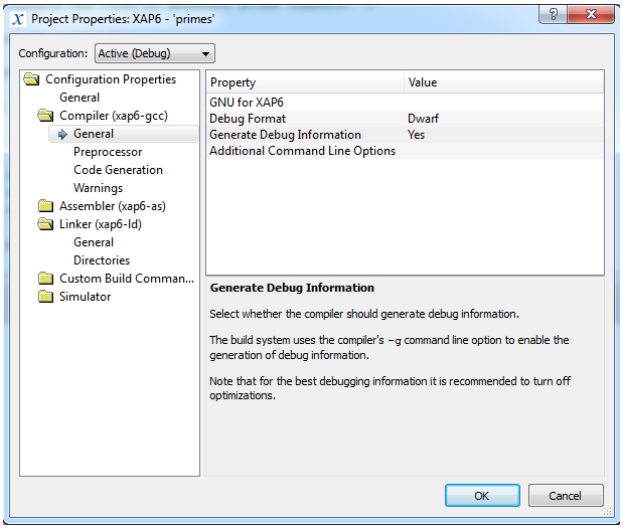
The Release configuration does not contain any debug information, so working out where to measure to and from can be tricky.

Three techniques that can be used to measure execution time are described below.

Enable debugging information

By default, Release builds do not contain debugging information which means that it is only possible to step through the code at the disassembly level rather than at the source level.

We can override this setting. From the Project menu choose Properties. Set the Generate Debug Information property to Yes.



Stepping through code that is built with the optimisation enabled may produce erratic behaviour. This is because the compiler may merge instructions from multiple source lines meaning that a single XAP instruction is associated with more than one source line.

If we structure our test code so that the measurement we want to take is for a complete function then we can write a simple harness that the debugger will be able to step through.

```
/* must not be static to avoid the compiler optimising the function
 * into main() */
void test1(void)
{
    int i;
    for (i=0; i<100; i++)
    {
        printf ("Number is %d\n", i);
    }
}

int main(int argc, char* argv[])
{
    test1();
    return 0;
}
```

We can measure the execution time of the `test1()` function by running to a breakpoint at the `test1()` call in `main()`, setting the counts in the Simulator widget to zero, and then stepping over. The results are shown below.

Insert breakpoints in the code

The XAP processors include a `brk` instruction that can be inserted into the code from C.

We can measure the execution time of the loop by running to the first embedded `brk` instruction, setting the counts in the Simulator widget to zero, then running to the second embedded `brk` instruction.

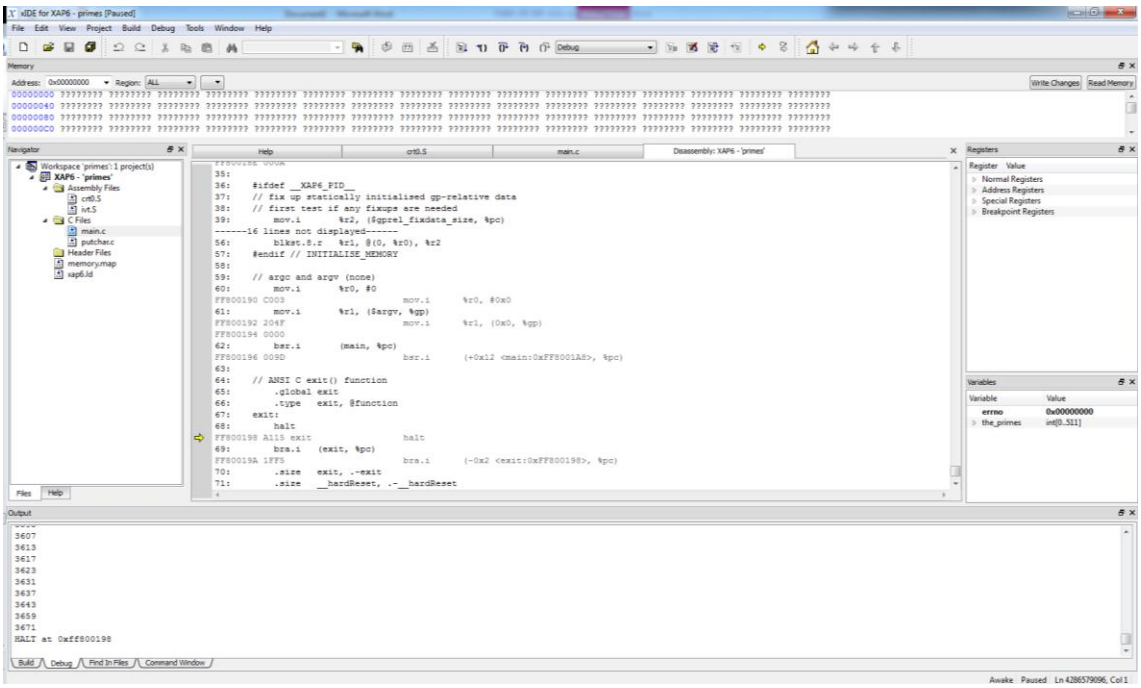
```
#include <machine/xap.h>
int main(int argc, char* argv[])
{
    int i;
    xap_brk();
    for (i=0; i<100; i++)
    {
        printf ("Number is %d\n", i);
    }
    xap_brk();
    return 0;
}
```

Examine the Disassembly

The third approach is more difficult and requires a detailed understanding of the XAP instruction set. It is possible to inspect the generated code in the Disassembly view to decide where to take the measurements.

Start the debug session. Then, from the Debug menu choose View Disassembly. Note: if debugging information is disabled, the disassembly will appear automatically.

As before, insert breakpoints at the beginning and end of the region of interest.



9 Notes and Pitfalls

You should now be in a position to quickly and easily evaluate the performance of the XAP processor with your own application. Further information about XAP and the associated tools is available in the documentation listed in section 1.3, "Other Documentation".

Selecting your benchmark code carefully is important. Below are some of the common pitfalls.

9.1 Compiler flags

Make sure you have optimised for size or speed as required for your evaluation and turn off debug information for best results.

9.2 Use of float, double and long long

Some benchmarks may make excessive use of features that occur much less frequently in real code. In particular:

- Floating point numbers: `float` and `double`
- 64-bit integers: `long long`

9.3 16-bit XAP

Some benchmarks may not be suitable for the 16-bit XAP4 and XAP5 because the code has been written assuming the C type `int` is 32-bit.

Others may make excessive use of `long` where `int` is more appropriate.

9.4 XAP5's Far mode

For evaluations of XAP5 a program may require more than 64 kBytes of data in which case a Far mode program must be created, which will generate larger code and run more slowly than the equivalent Near mode program. However, the full 16

MByte of XAP5 address space is always available regardless of mode and you may be able to restore Near mode performance by changing the way data is accessed, for example with some Assembler code.

9.5 printf()

Use of the function `printf()` may cause the entire floating point library to be included. If you know that the floating point library is not required you can replace `printf()` with `_printf()`, which doesn't include the floating point library.

10 Linker Map File

The section names and sizes are highlighted below in **bold**. Equivalent values are highlighted in the output from `xapn-objdump` in section 8.1, "Measure Code Size".

```
Archive member included because of file (symbol)

C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(printf.o)
                              main.o (printf)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__fpdisp.o)
                              C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(printf.o) (__fpdisp)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__vfprin.o)
                              C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(printf.o) (__vfprin)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(frexp.o)
                              C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__fpdisp.o) (frexp)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(isdigit.o)
                              C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__vfprin.o) (isdigit)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(ldexp.o)
                              C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__fpdisp.o) (ldexp)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(strlen.o)
                              C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__vfprin.o) (strlen)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dadd_gcc.o)
```

```
                                C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__fpdisp.o) (__gcc_dadd)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__ddiv_gcc.o)
                                C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__fpdisp.o) (__gcc_ddiv)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__deq_gcc.o)
                                C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__fpdisp.o) (__gcc_deq)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dfix_gcc.o)
                                C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__fpdisp.o) (__gcc_dfix)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dflt_gcc.o)
                                C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__fpdisp.o) (__gcc_dflt)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dgeq_gcc.o)
                                C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__fpdisp.o) (__gcc_dgeq)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dgr_gcc.o)
                                C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dgeq_gcc.o) (__gcc_dgr)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__div_gcc.o)
                                C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__fpdisp.o) (__gcc_div)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__udiv_gcc.o)
                                C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__vfprin.o) (__gcc_udiv)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dleq_gcc.o)
                                C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__fpdisp.o) (__gcc_dleq)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dls_gcc.o)
                                C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__fpdisp.o) (__gcc_dls)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dmul_gcc.o)
                                C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__fpdisp.o) (__gcc_dmul)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dsub_gcc.o)
                                C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__fpdisp.o) (__gcc_dsub)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__setremain4.o)
                                C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__div_gcc.o) (__setremain)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dcreate.o)
                                C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dadd_gcc.o) (__dcreate)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__huge_v.o)
                                C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(ldexp.o) (__huge_v)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__ovfl.o)
                                C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__div_gcc.o) (__ovfl)
C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(errno.o)
                                C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(ldexp.o) (errno)
```

```
Memory Configuration

Name             Origin             Length             Attributes
mem              0x00000000         0x00010000
*default*        0x00000000         0xffffffff

Linker script and memory map

LOAD ivt.o
LOAD crt0.o
LOAD main.o
LOAD putchar.o
LOAD C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a
LOAD C:/Program Files/XAP4_GNU_1.14.1/lib/gcc/xap4/3.4.2\libgcc.a
LOAD C:/Program Files/XAP4_GNU_1.14.1/lib/gcc/xap4/3.4.2\libgcc.a
                0x00000002                $ram_bottom = 0x2
                0x00008400                $rom_bottom = 0x8400

.vectors        0x00008400       0x60
 *ivt.o(.vectors)
 .vectors       0x00008400       0x60 ivt.o
 *(EXCLUDE_FILE(*ivt.o) .vectors)

.rodata         0x00008460       0x5a
 *(.rodata)
 .rodata        0x00008460        0x4 crt0.o
                0x00008460                __heapsize
 .rodata        0x00008464        0xe main.o
 .rodata        0x00008472        0x8 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__fpdisp.o)
 .rodata        0x0000847a       0x38 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__vfprin.o)
 .rodata        0x000084b2        0x8 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__huge_v.o)
                0x000084b2                __huge_v

.data           0x00000002       0x12 load address 0x000084ba
                0x00000002                $data_bottom = .
 *(.data)
```

```
 *fill*          0x00000002        0x2 00
 .data           0x00000004        0x8 crt0.o
                 0x00000008                __heap
 .data           0x0000000c        0x8 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__setremain4.o)
                 0x0000000c                $remain

.bss             0x00000014        0x4
                 0x00000014                $bss_bottom = .
 *(.bss)
 .bss            0x00000014        0x4 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(errno.o)
                 0x00000014                errno

.text            0x000084cc     0x2ef0
 *(.text)
 .text           0x000084cc       0x2a crt0.o
                                  0x38 (size before relaxing)
                 0x000084cc                $crt0
                 0x000084f4                $nullHandler
                 0x000084f0                exit
 .text           0x000084f6       0x18 main.o
                                  0x1c (size before relaxing)
                 0x000084f6                main
 .text           0x0000850e        0x6 putchar.o
                 0x0000850e                putchar
 .text           0x00008514       0x1a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(printf.o)
                 0x00008514                printf
 .text           0x0000852e      0xb50 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__fpdisp.o)
                                 0xc42 (size before relaxing)
                 0x00008d0a                __fpdisp
 .text           0x0000907e      0x7fe C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__vfprin.o)
                                 0x944 (size before relaxing)
                 0x00009098                __vfprin
 .text           0x0000987c       0x58 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(frexp.o)
                                  0x5c (size before relaxing)
                 0x0000987c                frexp
 .text           0x000098d4       0x10 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(isdigit.o)
```

```
                                0x12 (size before relaxing)
                0x000098d4              isdigit
.text           0x000098e4       0x7c C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(ldexp.o)
                                0x86 (size before relaxing)
                0x000098e4              ldexp
.text           0x00009960        0xe C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(strlen.o)
                                0x10 (size before relaxing)
                0x00009960              strlen
.text           0x0000996e      0x40a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dadd_gcc.o)
                               0x468 (size before relaxing)
                0x0000996e              __gcc_dadd
.text           0x00009d78      0x314 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__ddiv_gcc.o)
                               0x36c (size before relaxing)
                0x00009d78              __gcc_ddiv
.text           0x0000a08c       0x82 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__deq_gcc.o)
                                0xa8 (size before relaxing)
                0x0000a08c              __gcc_deq
.text           0x0000a10e       0x76 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dfix_gcc.o)
                                0x82 (size before relaxing)
                0x0000a10e              __gcc_dfix
.text           0x0000a184       0x80 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dflt_gcc.o)
                                0x8a (size before relaxing)
                0x0000a184              __gcc_dflt
.text           0x0000a204       0x66 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dgeq_gcc.o)
                                0x7a (size before relaxing)
                0x0000a232              __gcc_dgeq
.text           0x0000a26a       0xdc C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dgr_gcc.o)
                               0x118 (size before relaxing)
                0x0000a26a              __gcc_dgr
.text           0x0000a346      0x242 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__div_gcc.o)
                               0x2a4 (size before relaxing)
                0x0000a346              __gcc_div
.text           0x0000a588      0x1f4 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__udiv_gcc.o)
                               0x254 (size before relaxing)
                0x0000a588              __gcc_udiv
.text           0x0000a77c       0x5a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dleq_gcc.o)
```

```
                                0x6c (size before relaxing)
                0x0000a7aa              __gcc_dleq
 .text          0x0000a7d6       0x68 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dls_gcc.o)
                                 0x7a (size before relaxing)
                0x0000a804              __gcc_dls
 .text          0x0000a83e      0x390 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dmul_gcc.o)
                                0x3ce (size before relaxing)
                0x0000a83e              __gcc_dmul
 .text          0x0000abce      0x5e6 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dsub_gcc.o)
                                0x66e (size before relaxing)
                0x0000abce              __gcc_dsub
 .text          0x0000b1b4        0x6 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__setremain4.o)
                0x0000b1b4              __setremain
 .text          0x0000b1ba      0x1fe C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dcreate.o)
                                0x24c (size before relaxing)
                0x0000b1ba              __dcreate
 .text          0x0000b3b8        0x4 C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__ovfl.o)
                                  0x6 (size before relaxing)
                0x0000b3b8              __ovfl
                0x00000004              $bss_size = SIZEOF (.bss)
                0x00000012              $data_size = SIZEOF (.data)
                0x000084ba              $data_src = (($rom_bottom + SIZEOF (.vectors)) + SIZEOF (.rodata))

.vp             0x0000fffe        0x2
                0x0000fffe        0x2 SHORT 0x8400 ADDR (.vectors)
                0x00008000              $stack_top = 0x8000
                0x00000018              $heap_bottom = (($ram_bottom + SIZEOF (.data)) + SIZEOF (.bss))
                0x00003ff4              $heap_size = (($stack_top - $heap_bottom) / 0x2)
                0x0000400a              $heap_top = (($heap_bottom + $heap_size) - 0x2)
                0x0000400c              $stack_bottom = ($heap_top + 0x2)
                0x00003ff4              $stack_size = ($stack_top - $stack_bottom)
OUTPUT(../release/myproject.elf elf32-xap4)

.comment        0x00000000      0x444
 .comment       0x00000000       0x2a main.o
 .comment       0x0000002a       0x2a putchar.o
```

```
.comment        0x00000054        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(printf.o)
.comment        0x0000007e        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__fpdisp.o)
.comment        0x000000a8        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__vfprin.o)
.comment        0x000000d2        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(frexp.o)
.comment        0x000000fc        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(isdigit.o)
.comment        0x00000126        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(ldexp.o)
.comment        0x00000150        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(strlen.o)
.comment        0x0000017a        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dadd_gcc.o)
.comment        0x000001a4        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__ddiv_gcc.o)
.comment        0x000001ce        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__deq_gcc.o)
.comment        0x000001f8        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dfix_gcc.o)
.comment        0x00000222        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dflt_gcc.o)
.comment        0x0000024c        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dgeq_gcc.o)
.comment        0x00000276        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dgr_gcc.o)
.comment        0x000002a0        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__div_gcc.o)
.comment        0x000002ca        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__udiv_gcc.o)
.comment        0x000002f4        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dleq_gcc.o)
.comment        0x0000031e        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dls_gcc.o)
.comment        0x00000348        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dmul_gcc.o)
.comment        0x00000372        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dsub_gcc.o)
.comment        0x0000039c        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__dcreate.o)
.comment        0x000003c6        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__huge_v.o)
.comment        0x000003f0        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(__ovfl.o)
.comment        0x0000041a        0x2a C:/Program Files/xIDE_XAP4_3.33.0/lib/libxap/lib\libxap4s.a(errno.o)
```