Introduction to VBCC and Cross Compilation

Karl Jeacle

Amiga Ireland 2020

CONTENTS

- Cross compilation
- VBCC
- Installing VBCC + Amiga NDK
- Compiling with VBCC
- Writing Amiga C code

CROSS COMPILATION

- Compile code on host platform
  - e.g. Windows, MacOS or Linux
- Create binaries for different target platform
  - e.g. Amiga/68K, OS4/PPC, MorphOS/PPC
- AmigaOS created using Sage-II/68K and Sun/68K
  - Cross-compiled with Greenhills C compiler

WHY CROSS COMPILE?

- Modern development tools
- Faster compile times
- Large display
- Can emulate target machine in a window
- Test crash doesn't kill dev environment

RUNNING PROGRAMS ON YOUR AMIGA

1. UAE
   - Run on virtual Amiga

2. Network
   - Run on real Amiga via Samba/FTP/Serial

3. Sneakernet
   - Run on real Amiga via CF/SD/USB

VBCC

- C Compiler written by Volker Barthelmann
- Supports C89 (ANSI C) and subset of C99
- Actively maintained and (mostly) open source
- Hosts: Win, Mac, Unix, Amiga (just need C99 cc)
- Targets: Amiga 1.2/1.3, 2.x/3.x, 4.x, MorphOS
- Amiga support by Frank Wille (phx)

INSTALLING VBCC

- Install binaries on your host platform
  - Amiga binaries for 0.9g on web site
  - Win/Mac binaries for 0.9f now, 0.9g soon
  - Compile from source on Linux
- Google: vbcc amiga cross compile
- Install platform-specific target files

VBCC TARGET CONTENTS

1. Config file for target
   - `kick13, aos68k, aosppc, morphos`

2. C include files
   - `stdio.h, math.h, proto/*.h`

3. C link libraries
   - `vc.lib, mieee.lib, amiga.lib`

AOS68K CONFIG FILE

```
-cc=vbccm68k -quiet -hunkdebug %s -o= %s %s -O=%ld -I$VBCC/tar
-ccv=vbccm68k -hunkdebug %s -o= %s %s -O=%ld -I$VBCC/targets/m
-as=vasmm68k_mot -quiet -Fhunk -nowarn=62 %s -o %s
-asv=vasmm68k_mot -Fhunk -nowarn=62 %s -o %s
-rm=rm -f %s
-rmv=rm %s
-ld=vlink -bamigahunk -x -Bstatic -Cvbcc -nostdlib -mrel $VBCC
-l2=vlink -bamigahunk -x -Bstatic -Cvbcc -nostdlib -mrel %s %s
-ldv=vlink -bamigahunk -t -x -Bstatic -Cvbcc -nostdlib -mrel $
-l2v=vlink -bamigahunk -t -x -Bstatic -Cvbcc -nostdlib -mrel %
-ldnodb=-s -Rshort
-ul=-l%s
-cf=-F%s
-ml=1000
```

$VBCC PATH

- VBCC will look for config files here
- (vbcc: on AmigaOS and MorphOS)
- Config file contains:
  - Path to include files
  - Path to target libraries

NDK + SDK

- VBCC allows you to compile generic C code
- Need NDK to write Amiga-specific code
- (Native/Software Development Kits)
- Amiga NDK
- Roadshow SDK

AMIGA NDK

- To use AmigaOS functions, the NDK is required
  - C include files for Exec, DOS, Intuition, etc
  - Without this would be limited to basic I/O
- Last updated in 2001 after OS3.9 release
- Includes documentation (Autodocs) and examples
- Look online for HTML versions of Autodocs

INSTALLING AMIGA NDK

- haage-partner.de/download/AmigaOS/NDK39.lha
- Simply unpack and browse docs and examples
- Need C include files
  - NDK_3.9/Include/include_h/
- Don't need C link libraries
  - NDK_3.9/Include/linker_libs/

INSTALLING ROADSHOW SDK

- Download 1.4:
  - `amigafuture.de/app.php/dlext/?view=detail&df_id=3658`
- Unpack and browse documentation
  - `doc/bsdsocket.doc`
- Need C include files
  - `netinclude/`

COMPILING WITH VBCC

- Command line examples
- Maths
- Optimisation
- Warnings
- Verbosity

HELLO.C

```
#include <stdio.h>

main()
{
    printf("Hello Amiga Ireland!\n");
}
```

VC FRONTEND

- vc is frontend for vbcc, vasm and vlink
- vc +aos68k hello.c
- vc +aos68k -o hello hello.c
- vc +aos68k -k -o hello hello.c

HELLO.ASM

```
        idnt    "hello.c"
        opt o+,ol+,op+,oc+,ot+,oj+,ob+,om+
        section "CODE",code
        public  _main
        cnop    0,4
_main
        movem.l l3,-(a7)
        pea     l2
        jsr     ___v0printf
        addq.w  #4,a7
l1
l3      reg
l5      equ     0
        rts
        cnop    0,4
```

HELLO.C C99

```c
#include <stdio.h>
#include <stdint.h>

int main()
{
    // C++ comments are part of C99
    // -cpp-comments or -+ if using C89
    uint16_t year = 2020;
    printf("Hello Amiga Ireland %hu!\n", year);
    return 0;
}
```

- `vc +aos68k -c99 hello.c`

MATHS.C

```c
#include <stdio.h>

main()
{
    double x = 1.5;
    x *= 2.5;
    printf("%0.2f\n", x);
}
```

- `vc +aos68k maths.c -lmieee`
- (no `%f` in `vc.lib printf` so need `-lmieee`)

OPTIMISATION

- `-O0` is equivalent to `-O=0`
- `-O1` usually generates smallest code
- `-O2` will activate more optimisations
- `-O3` will activate all optimisations
- Can optimise for `-speed` or `-size`

WARNINGS

- Useful for correcting your code
- `-warn=-1` to enable all warnings (too many)
- use `-dontwarn=XX` to selectively disable

```
-warn=-1
-dontwarn=81  -dontwarn=163 -dontwarn=166
-dontwarn=167 -dontwarn=306 -dontwarn=307
```

VERBOSITY

- `-v` shows programs called by vc
- `-vv` shows libraries being linked

-V

```
vc frontend for vbcc (c) in 1995-2016 by Volker Barthelmann
vbccm68k -quiet -hunkdebug "hello.c" -o= "/tmp/fileRjd8Z1.asm"
vasmm68k_mot -quiet -Fhunk -nowarn=62 "/tmp/fileRjd8Z1.asm" -o
vlink -bamigahunk -x -Bstatic -Cvbcc -nostdlib -mrel $VBCC/tar
rm -f "/tmp/fileRjd8Z1.asm"
rm -f "/tmp/fileRjd8Z1.o"
```

-VV

```
vc frontend for vbcc (c) in 1995-2016 by Volker Barthelmann
flags=1414 opt=1 len=1426
Argument 6:hello.c
File "hello.c"=2
add_name: "/tmp/file66Xn8b.asm"
vbccm68k -hunkdebug "hello.c" -o= "/tmp/file66Xn8b.asm"  -O=1
vbcc V0.9g (c) in 1995-2019 by Volker Barthelmann
vbcc code-generator for m68k/ColdFire V1.13 (c) in 1995-2019 b
add_name: "/tmp/file66Xn8b.o"
add_name: "/tmp/file66Xn8b.o"
vasmm68k_mot -Fhunk -nowarn=62 "/tmp/file66Xn8b.asm" -o "/tmp/
vasm 1.8g (c) in 2002-2019 Volker Barthelmann
vasm M68k/CPU32/ColdFire cpu backend 2.3f (c) 2002-2019 Frank
vasm motorola syntax module 3.13 (c) 2002-2019 Frank Wille
vasm hunk format output module 2.11 (c) 2002-2019 Frank Wille
```

WRITING AMIGA C CODE

- amiga.lib
- Autodocs
- Prototypes
- Ctrl-c
- Stack size
- Version string
- Types

AMIGA.LIB

```c
#include <stdio.h>
#include <proto/dos.h>

int main() {
    puts("1"); Delay(50);
    puts("2"); Delay(50);
    puts("3"); Delay(50);
}
```

- `vc +aos68k -c99 -I$NDK_INC delay.c -lamiga`
- Can add -I$NDK_INC to aos68k config file

```
NAME
        Open -- Open a file for input or output

    SYNOPSIS
        file = Open( name, accessMode )
        D0           D1    D2

        BPTR Open(STRPTR, LONG)

    FUNCTION
        The named file is opened and a file handle returned.
        [...]

    INPUTS
        name       - pointer to a null-terminated string
        accessMode - integer

    RESULTS
        file - BCPL pointer to a file handle

    SEE ALSO
        Close(), ChangeMode(), NameFromFH(), ParentOfFH()
```

PROTOTYPES

- Amiga parameters in registers
  - C parameters are on stack
- Original solution
  - `#include <clib/dos_protos.h>`
  - Stub C function in amiga.lib
- Modern solution
  - `#include <proto/dos.h>`
  - Direct call with inline assembly or pragmas

CTRL-C (STDIO)

```
#include <stdio.h>
#include <proto/dos.h>

void cleanUp()      { puts("1"); }
void _EXIT_9_nine() { puts("2"); }
void _EXIT_0_zero() { puts("3"); }

int main() {
    atexit(cleanUp);
    puts("A"); Delay(50);
    puts("B"); Delay(50);
    puts("C");
}
```

CTRL-C (AMIGA)

```c
#include <stdio.h>
#include <proto/dos.h>

int _chkabort(void) { return(0); } // disable ctrl-c

void checkCtrlC() {
    if (CheckSignal(SIGBREAKF_CTRL_C)) {
        puts("cleanup"); exit(1);
    }
}

int main() {
    checkCtrlC(); puts("A"); Delay(50);
    checkCtrlC(); puts("B"); Delay(50);
    checkCtrlC(); puts("C");
}
```

STACK SIZE

- Default OS3 stack is 4000 bytes
  - Take care to avoid stack overflow
- Avoid large local variables
  - Declare as static or global
  - Use AllocMem() / FreeMem()
- Increase stack size

```
size_t __stack = 8000; // OS3
static const char USED min_stack[] = "$STACK:102400";
```

VERSION STRING

- Identify program using version command
- ```
  UBYTE *Version =
      "\0$VER: MyProgram 1.0 (18.1.2020)\0";
  ```
- ```
  Work:> version MyProgram full
  MyProgram 1.0 (18.1.2020)
  ```

EXEC/TYPES.H VS STDINT.H

Amiga	C99
BYTE	int8_t
SHORT / WORD	int16_t
LONG / BPTR	int32_t
STRPTR	char *
APTR	void *

USEFUL TOOLS

- MuForce, MuGuardianAngel
- Scout, Stackmon
- SnoopDOS
- make, cppcheck

The End

karl@jeacle.ie