DAT320
Operating Systems

Fall 2014
Introduction to C

University of
Stavanger

University of Stavanger

Today

- History of C
- Hello World
- Toolchain

University of Stavanger

History

- Initially developed by Dennis Ritchie between 1969 and 1973 at AT&T Bell Labs
- Closely related to development of UNIX
- Inspired by ALGOL

University of Stavanger

History

- C was derived from B
- B from BCPL
- Rumors say that the name B comes from the name of the inventors wife, Barbara

University of Stavanger

First program

```
int main( void )
{
    return 0;
}
```

$gcc first.c –o first

$ ./first

$

University of Stavanger

First program

- Every full C program begins inside a function called "main“

University of Stavanger

Parts of a C-program

```
int main( void )
{
   return 0;
}
```

University of Stavanger

Hello world

```
#include <stdio.h>


int main(void)
{
    /* The hello world program*/
    printf("hello, world\n");
}
```

mortenm@badne7:~$ gcc hello.c -o hello

mortenm@badne7:~$ ./hello

hello, world

University of Stavanger

- The first line of the program *#include <stdio.h>* is a preprocessor command, which tells a C compiler to include stdio.h file before going to actual compilation.
- The next line *int main(void)* is the main function where program execution begins.
- The next line /*...*/ will be ignored by the compiler and it has been put to add additional comments in the program. So such lines are called comments in the program.
- The next line *printf(...)* is another function available in C which causes the message "Hello, World!" to be displayed on the screen.
- The next line **return 0;** terminates main()function and returns the value 0 to the OS.

University of Stavanger

The preprocessor,

- Executed **before** compiler
- Include other files:
  #include<…>, #include "…."
- Simple macros:
  #define BUFFER_SIZE 100
- Simple control structures:
  #if
  #endif

University of Stavanger

Functions in C

```
return_type function_name( parameter list )
{
    body of the function
}
```

University of Stavanger

Functions in C

```
#include <stdio.h>
int my_add(int, int);

int main(void)
{
    int sum;
    sum = my_add( 3,5);
    printf("hello, world\n");
    printf("%d + %d = %d\n", 3,5, sum );
}

int my_add( int a, int b)
{
     return a+b;
}
```

University of Stavanger

Guide

http://www.tutorialspoint.com/cprogramming/c_quick_guide.htm

University of Stavanger

Make

“Tool to help automate build of software”

target: dependencies
[tab] system command

University of Stavanger

Make

“Tool to help automate build of software”

target: dependencies
[tab] system command

University of Stavanger

Make

Makefile:

```
hello: hello.c
        gcc hello.c -o hello
```

```
mortenm@badne7:~/OpSys2014$ make
gcc hello.c -o hello
mortenm@badne7:~/OpSys2014$
.
.
mortenm@badne7:~/OpSys2014$ make
make: `hello' is up to date.
mortenm@badne7:~/OpSys2014$
```

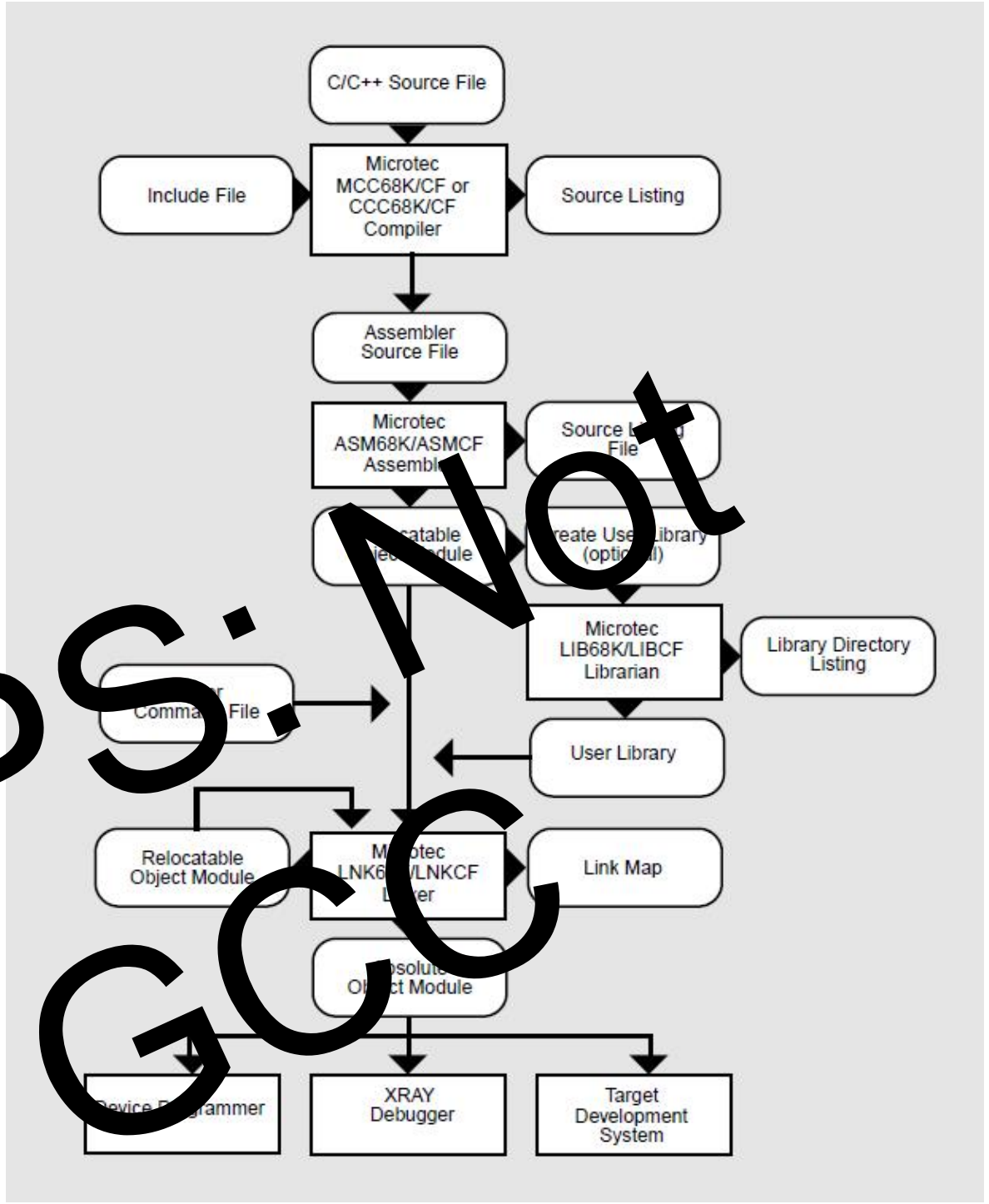
Behind the scene of gcc.
gcc is actually a collection of tools:

University of Stavanger

- powerpc-linux-gnu-addr2line.exe
- powerpc-linux-gnu-ar.exe
- powerpc-linux-gnu-as.exe
- powerpc-linux-gnu-c++.exe
- powerpc-linux-gnu-c++filt.exe
- powerpc-linux-gnu-cpp.exe
- powerpc-linux-gnu-g++.exe
- powerpc-linux-gnu-gcc-4.3.2.exe
- powerpc-linux-gnu-gcc.exe
- powerpc-linux-gnu-gcov.exe
- powerpc-linux-gnu-gdb.exe
- powerpc-linux-gnu-gprof.exe
- powerpc-linux-gnu-ld.exe
- powerpc-linux-gnu-nm.exe
- powerpc-linux-gnu-objcopy.exe
- powerpc-linux-gnu-objdump.exe
- powerpc-linux-gnu-ranlib.exe
- powerpc-linux-gnu-readelf.exe
- powerpc-linux-gnu-size.exe
- powerpc-linux-gnu-strings.exe
- powerpc-linux-gnu-strip.exe

PS:GCC for PPC

University of Stavanger



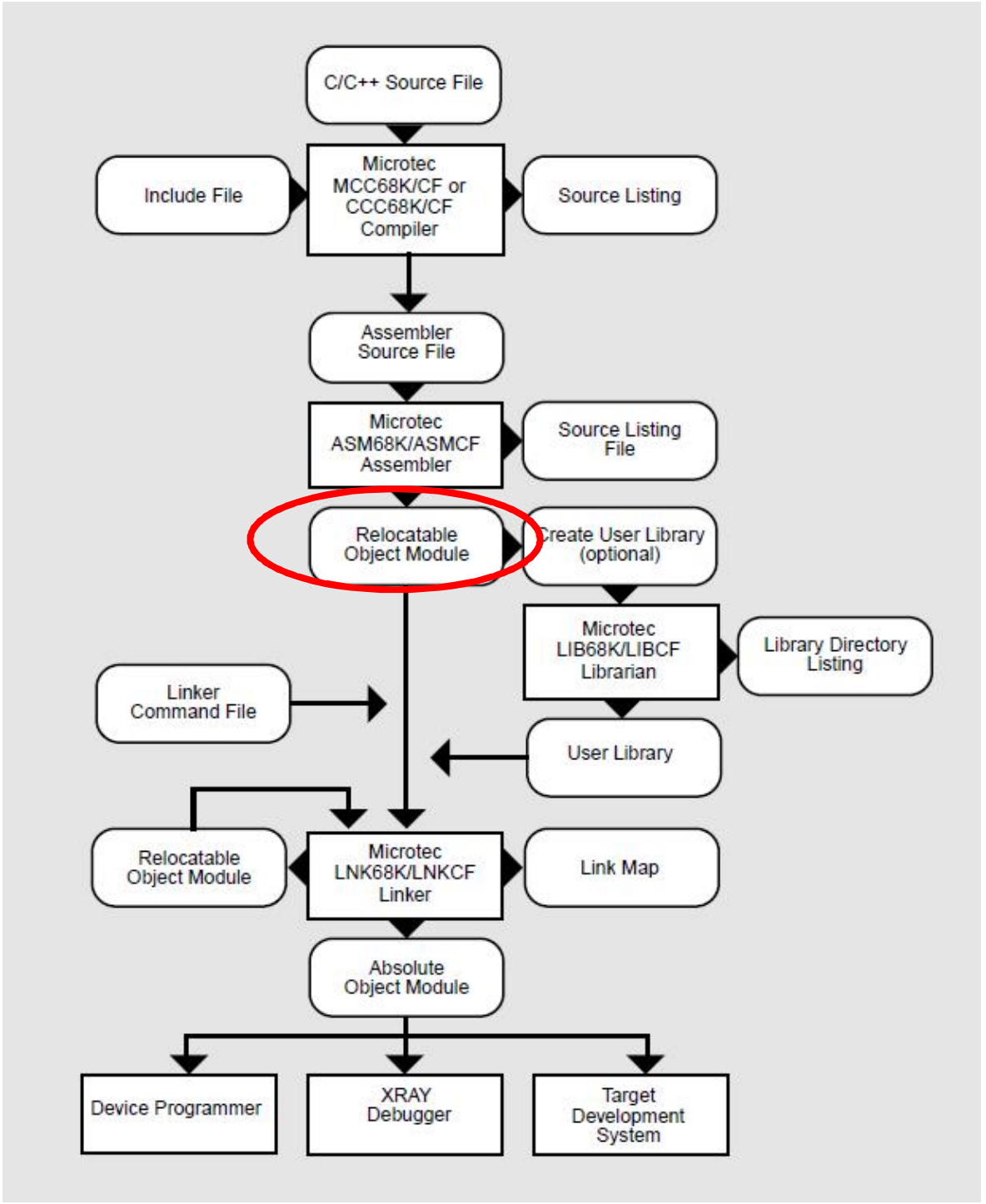
PS: Not GCC

University of Stavanger

Q1

- How do you use gcc to only produce the .o file? What is the difference between generating only the .o file, and building the hello executable done in the previous compilation above?

University of Stavanger



University of Stavanger

```
mortenm@badne7:~$ gcc –help
Usage: gcc [options] file...
Options:
  -pass-exit-codes         Exit with highest error code from a phase
  --help                   Display this information
   .
   .
 -c                      Compile and assemble, but do not link

mortenm@badne7:~/OpSys2014$ gcc -c hello.c -o hello.o
```

University of Stavanger

Q2 and Q3

Give the command for compiling with debug enabled instead of normal compilation for the two examples shown in Listing 2 and Listing 3. Explain how to turn debugging on/off for the two cases.

Give a brief pros and cons discussion for the two methods to add debug code shown in Listing 2 and Listing 3.

University of Stavanger

gcc -D defines a macro to be used by the preprocessor:

```
gcc -DDEBUG myfile.c -o myfile
```

University of Stavanger

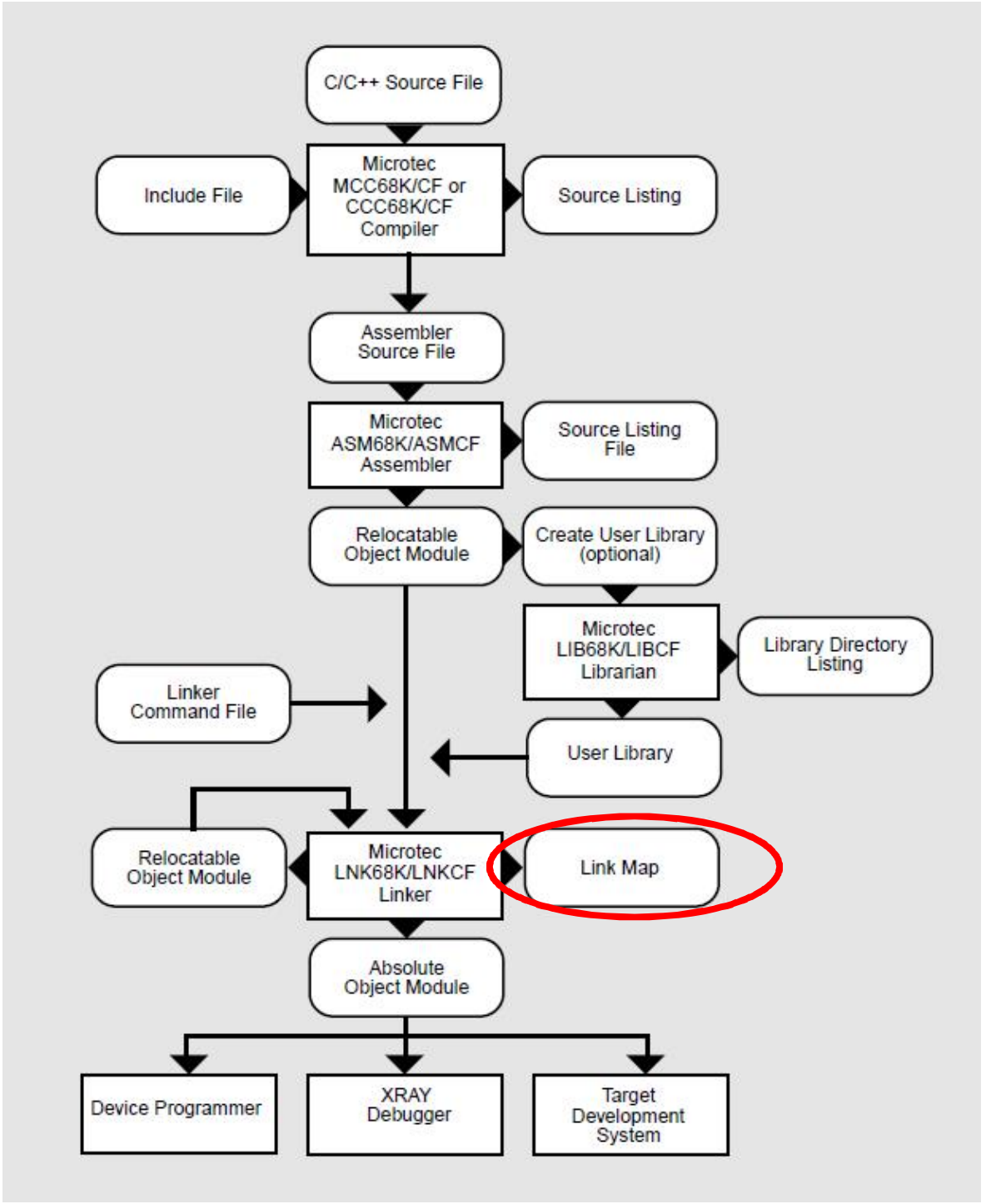
```
#include <stdio.h>
int main(void)
{
#ifdef MYSYMBOL
   printf("hello, world\n");
#endif
    printf("%d + %d = %d\n", 3,5,my_add(3,5) );
}

int my_add( int a, int b)
{
        return a+b;
}

mortenm@badne7:~/OpSys2014$ gcc  hello.c -o hello
mortenm@badne7:~/OpSys2014$ ./hello
3 + 5 = 8
mortenm@badne7:~/OpSys2014$ gcc  -DMYSYMBOL hello.c -o hello
mortenm@badne7:~/OpSys2014$ ./hello
hello, world
3 + 5 = 8
```

University of Stavanger

Map file

```
gcc hello.c -o hello -Wl,-Map=mymap.txt
```

University of Stavanger

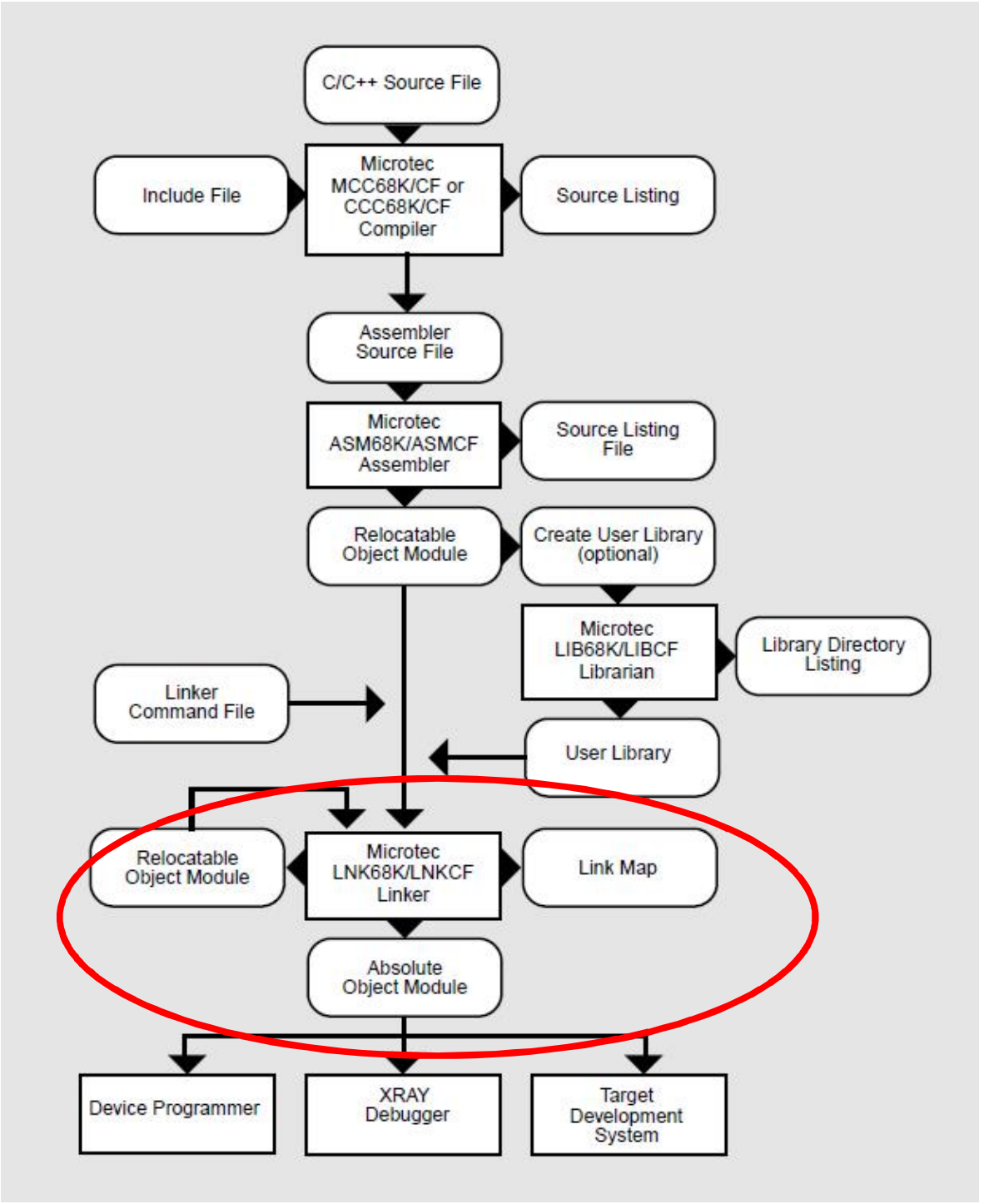


University of Stavanger

- .data - **initialized data**.
- .bss - uninitalized data.
- .rodata – read only

```
#include <stdio.h>
int myvar1=0;
int myvar2=1;
const int myvar3=1;
int main(void)
{
        …
```

University of Stavanger

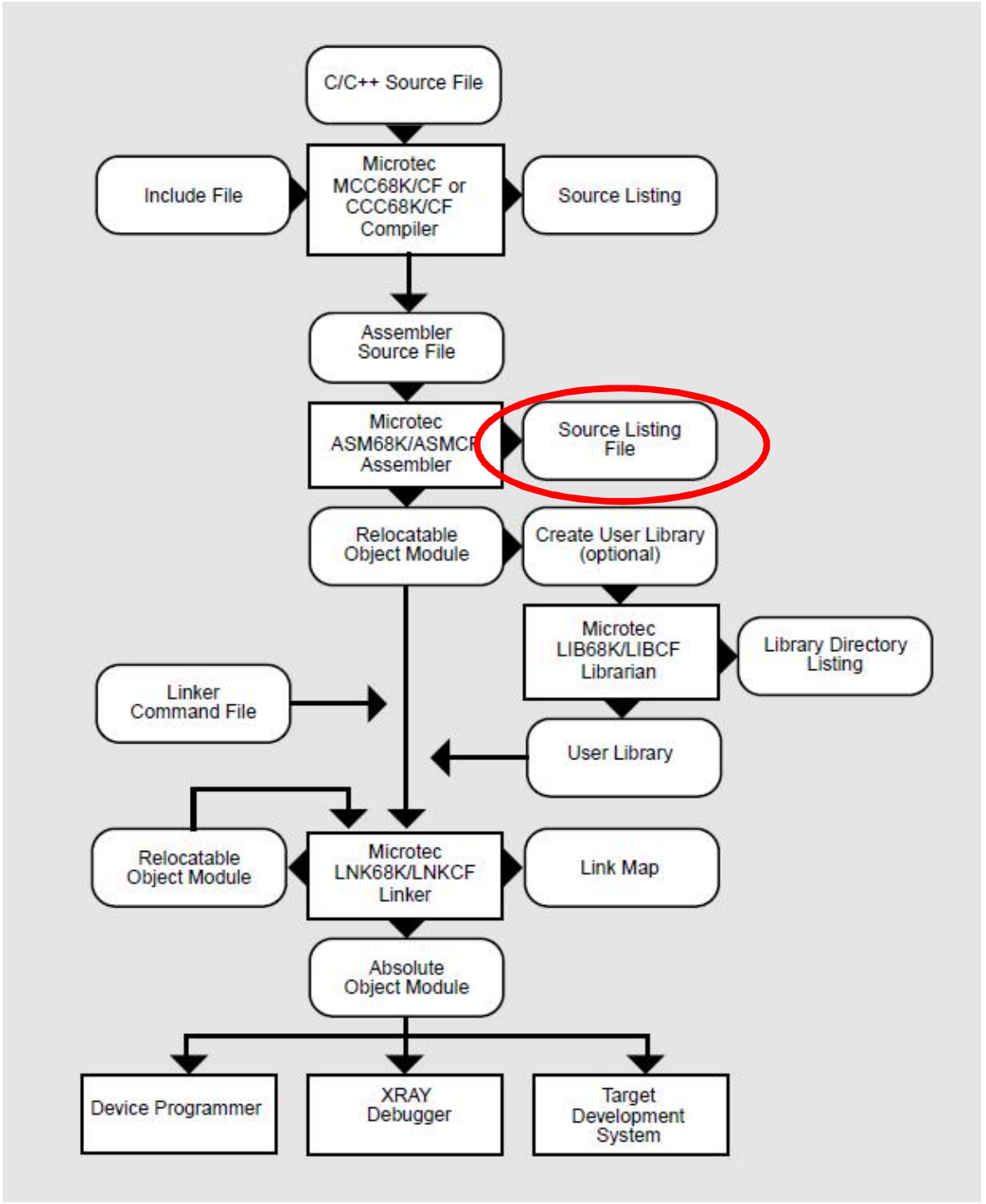


University of Stavanger

float vs double multiply

```
mortenm@badne7:~$ gcc –help
Usage: gcc [options] file...
Options:
  -pass-exit-codes         Exit with highest error code from a
phase
  --help                   Display this information
   .
   .
  -S                       Compile only; do not assemble or link
```

University of Stavanger



University of Stavanger

```
float mult( float a, float b)
{
        return a*b;
}

double multd(double a, double b)
{
        return a*b;
}

mortenm@badne7:~/OpSys2014$ gcc hello.c -S -o hello.asm

Hint: look at hello.asm. Search for ‘mult’ and ‘multd’
```

University of Stavanger

Make a lib

http://www.adp-gmbh.ch/cpp/gcc/create_lib.html

University of Stavanger

Makefile for exe + lib

```
main.exe: main.c l1.h libl1.a
        gcc main.c -o main.exe -L. –ll1

libl1.a: l1.c l1.h
        gcc -c l1.c -o l1.o
        ar rcs  libl1.a l1.o

clean:
        rm main.o l1.o libl1.a
```

University of Stavanger

Pointers in C

Get the address to a variable by putting a ‘&’ in front:

```
int I = 2;
printf(“Address of I=%p\n”, &I );
```

University of Stavanger

- To be continued….