Eclipse + Codesourcy + OpenOCD + GCC, for develop and debug on STM32Fo family using LINUX.

By: Nicolas Fillon <u>ST</u> FAE (USA) – Santa Clara (California).





INTRODUCTION

This guide's intent and purpose is to allow a user with minimal Linux experience to successfully setup an **absolutely free** development environment to program the STM₃₂Fo microcontroller with full debugging capability. Other STM₃₂F families can be substituted.

PREREQUISITES

- 1. A Linux distribution, this example uses Linux Mint 13 (Ubuntu works nicely too)
- 2. An Internet connection
- 3. At least 2GB of spare hard drive space
- 4. Familiarity with TERMINAL
- 5. An STM32Fo Discovery Board!

PART 1 – Install Codesourcy

If you are using 64bit Linux, install ia32-libs with TERMINAL by input ing:

sudo apt-get install ia32-libs

Download and install Sourcery G++ Lite. In this example, we use Sourcery G++ Lite 2011.03-42 for ARM EABI. Newer versions might not work properly. https://sourcery.mentor.com/sqpp/lite/arm/portal/subscription?@template=lite

sh arm-2011.03-42-arm-none-eabi.bin

If the error message "ERROR: DASH shell not supported as system shell" is displayed, use this command to change the shell:

sudo dpkg-reconfigure -plow dash

It can now be installed by using the command:

~/Desktop/arm-2011.03-42-arm-none-eabi.bin

Follow the prompts and configure as listed below

- 1. Choose install set
 - Minimal Installation
- 2. Choose Install Folder
 - Default folder (/home/zach/CodeSourcery/Sourcery_G++_Lite)
- 3. Add to Path?
 - Check Modify path for user
- 4. Choose Link Folder
 - Check Don't create links



<u>T 2 – Install OpenOCD</u>

Open a terminal and run each command.

sudo apt-get install build-essential sudo apt-get install git sudo apt-get install libtool sudo apt-get install libftdi1 sudo apt-get install texinfo git clone http://repo.or.cz/r/openocd.git/ cd '/home/zach/openocd/' ./configure --prefix=/usr --enable-maintainer-mode --enable-stlink^C make git clone https://github.com/szczys/stm32fo-discovery-basic-template.git git clone git://github.com/texane/stlink.git

This should create a directory named *openocd*, a directory named *stm32fo-discovery-basic-template*, and *stlink* all in the local directory (/home/zach/).

Install the udev rules for our st-link device, so that a device "/dev/stlink" is accessible non-root. The needed rules are part of the sources that come with the file "10-stlink.rules".

cd ~/stlink/ sudo install -m 644 49-stlinkv2.rules /etc/udev/rules.d/49-stlinkv2.rules

To activate the new rules either reboot, or:

sudo udevadm control --reload-rules

PART 3 – Install Eclipse

- 1. Download Eclipse C/C++ Development Edition . The specific edition used here is: Eclipse IDE for C/C++ Developers Version: Indigo Release Build id: 20110615-0604
- 2. Download/Install GNU ARM Eclipse plug-in, go to indigo "Help"->"Install New software", Work with: GNU ARM Eclipse Plug-in - <u>http://gnuarmeclipse.sourceforge.net/updates</u>

Available Software Check the items that you wish to install.	
Work with: GNU ARM Eclipse Plug-in - http://gnuarmecl	ipse.sourceforge.net/updates Add software by working with the <u>"Available Software Sites"</u> preferences.
type filter text	2
Name	Version
🔻 🗹 🚥 CDT GNU Cross Development Tools	
🖙 🆚 GNU ARM C/C++ Development Support	0.5.4.201202210114
Select All Deselect All 1 item selected Details The GNU ARM Eclipse Pluq-in is an Eclipse CDT Manged I	Build Extension
	What is already installed?
Group Items by category	What is aneady instaned?
Show only software applicable to target environment	
$\begin{tabular}{ c c c c c c c c c c c c c c c c c c c$	software
(3)	< Back Next > Cancel Finish

PART 4 – Setup File Directory

- Download the STM32FoDiscovery Firmware package <u>http://www.st.com/internet/com/SOFTWARE_RESOURCES/SW_COMPONENT/FIRMWARE/</u> <u>stm32fodiscovery_fw.zip</u>
- 2. Extract the contents and copy the top level, Libraries folder to the workspace/project folder i.e /home/zach/workspace/Template/
- 3. Copy *stm32foxx_conf.h* from an example project in the STM32fo firmware package
- 4. In the Template folder create a new folder called *src*. Save all your source files in that location.
- 5. Copy and paste the files located in /home/zach/stm32fo-discovery-basictemplate/Device/ldscripts to your project's src folder (~/workspace/Template/src/)
- 6. Open the *stm32fo.ld* file which now resides in your src folder, and modify the blue highlighted lines to match below

/*

Default linker script for STM32F051R8T6 64k 8k Copyright RAISONANCE S.A.S. 2007

!!! This file is automatically generated by RIDE !!! Do not modify it, as it will be erased at every link. You can use, copy and distribute this file freely, but without any warranty. */

/* include the memory spaces definitions sub-script */ INCLUDE "/home/zach/workspace/Template/src/stm32fodiscovery_def.ld"

/* include the sections management sub-script for FLASH mode */ INCLUDE "/home/zach/workspace/Template/src/sections_flash.ld"

- 7. Rename *startup_stm32foxx.s*, located in ~/stm32fo-discovery-basic-template/and change the .s to .S
- 8. Copy the renamed file and paste it into the *src* directory.
- 9. Copy the ~/stm32fo-discovery-basic-template/extra/ to ~/workspace/Template/

PART 5 – Setup Eclipse

1. Create an empty C Project

Project name: Template	
✓ Use default location	
Location: /home/zach/workspace2/Templat	e Browse
Choose file system: default 🛔	
Project type:	Toolchains:
🔻 ≽ GNU Autotools	ARM Linux GCC (GNUARM)
Empty Project	ARM Linux GCC (Sourcery G++ Lite)
Hello World ANSI C Autotools Project	ARM Linux GCC (Summon)
🔻 📂 ARM Cross Target Application	ARM Linux GCC (devkitPro)
🟺 Empty Project	
ARM Cross Target Static Library	
🕨 🗁 Executable	
Shared Library	
🕨 🗁 Static Library	
🕨 🗁 Makefile project	
Show project types and toolchains only	if they are supported on the platform

type filter text	Tool Chain Editor	♥ ⇒
 Resource Builders C/C++ Build Build Variables 	Configuration: Debug [Active]	Manage Configurations
Discovery Options Environment Logging	Display compatible toolchains only Current toolchain: ARM Linux GCC (Sourcery G++ Lite)	* *
Settings Tool Chain Editor C/C++ General	Current builder: ARM Sourcery Linux GNU Make builder Used tools	÷
 Project References Run/Debug Settings Task Repository WikiText 	ARM Sourcery Linux GCC Assembler ARM Sourcery Linux GCC C Compiler ARM Sourcery Linux GCC C++ Compiler ARM Sourcery Linux GCC C+Linker ARM Sourcery Linux GCC C++ Linker ARM Sourcery Linux GCC Archiver ARM Sourcery Linux GNU Create Flash Image	Select Tools
		Restore Defaults Apply
(?)		Cancel

. Change Processor to cortex-mo

type filter text 🛛 🕿	Settings $rightarrow rightarrow r$	*
 Resource Builders 	Configuration: Debug [Active]	.]
▼ C/C++ Build		
Build Variables	🛞 Tool Settings 🎤 Build Steps 🚇 Build Artifact 🗟 Binary Parsers 😣 Error Parsers	
Environment	Target Processor Cortex-m0	
Logging	Debugging If the function of the func	
Settings	Additional Tools Thumb interwork (-mthumb-interwork)	

4. Paths and Symbols (Set for both Assembly SourceFile & GNU C). To add each path quickly, add every path as a workspace path /Template/Libraries/CMSIS/Include /Template/Libraries/STM32Foxx_StdPeriph_Driver/inc /Template/src /Template/Libraries/CMSIS/ST/STM32Foxx/Include



 Symbols in Settings USE_STDPERIPH_DRIVER USE_STM32Fo_DISCOVERY STM32FoXX

type filter text 🛛 🖾	Settings	↓ ↓
 Resource Builders C/C++ Build 	Configuration: Debug [Active]	↓ Manage Configurations
Build Variables Discovery Options	Tool Settings	Binary Parsers 😣 Error Parsers
Environment	🖉 Target Processor	Do not search system directories (-nostdinc)
Logging	🖉 Debugging	Preprocess only (-E)
Settings	🖄 Additional Tools	Defined symbols (-D)
Tool Chain Editor	SARM Sourcery Linux GCC Assembler	USE STDPERIPH DRIVER
C/C++ General	🖄 Preprocessor	USE STM32F0 DISCOVERY
Project References	🖄 Directories	STM32F0XX
Run/Debug Settings	🖄 Warnings	
Task Repository	🖄 Miscellaneous	
WikiText	🔻 🛞 ARM Sourcery Linux GCC C Compiler	
	🖄 Preprocessor	
	A Directories	

6. Linker setting. Check the Remove unused sections. Change the textbox so as to match the location of your stm32fo.ld file: /home/zach/workspace/Template/src/stm32fo.ld .



PART 6 – Configure the gdb/OpenOCD

1. Open Debug Configurations... by clicking on the down arrow of the green bug, and selecting Debug Configurations.



GDB (DSF) Hardware Debugging Launcher. Click OK.

This dialog allows you to specify which launcher to use when multiple launchers are available for a configuration and launch mode.
✓ Use configuration specific setting Change Workspace Settings
Launchers:
Standard GDB Hardware Debugging Launcher
GDB (DSF) Hardware Debugging Launcher
Description
Jtag hardware debugging using the Debugger Services
Cancel OK

5. Under the Debugger Tab uncheck everything and add

	Name: OpenOCD Debug
•	Image: Control of the second of the secon
	Remote Target Use remote target JTAG Device: Generic TCP/IP Host name or IP address: localhost Port number: 10000
	 Force thread list update on suspend Using GDB (DSF) Hardware Debugging Launcher - <u>Select other</u> Apply Revert
	Close Debug

GDB Command: /home/zach/CodeSourcery/Sourcery_G++_Lite/bin/arm-none-eabi-gdb

6. Under the Startup Tab uncheck everything. In the Run Commands: you should have something similar. target remote localhost:3333

monitor reset init

monitor stm_flash ~/workspace/Template/Debug/Template.bin

load ~/workspace/Template/Debug/Template.elf

```
symbol-file ~/workspace/Template/Debug/Template.elf cont
```

Name: OpenOCD Debug	
🗋 Main 🏇 Debugger 🍉 Startup 😼 Source 🗔 Common	
Initialization Commands	
Reset and Delay (seconds): 3	
Halt	
1	
Load Image and Symbols	
Load image	
• Use project binary: Template.elf	
O Use file: Workspace File System	
Image offset (hex):	
Load symbols	
• Use project binary: Template.elf	
O Use file: Workspace File System	
Symbols offset (hex):	
Runtime Options	
Set program counter at (hex):	
Set breakpoint at:	
Resume	
Run Commands	
target remote localhost:3333 monitor reset init monitor stm_flash ~/workspace/Template/Debug/Template.bin	
Using GDB (DSF) Hardware Debugging Launcher - <u>Select other</u> Apply Revert	



7. Press Apply. Under External Tools Configurations, add the Location:/home/zach/openocd/src/openocd Working directory: /home/zach/openocd/tcl

Arguments: -f./board/stm32fodiscovery.cfg -f./extra/stm32fo-openocd.cfg

	Name: OpenOCD	
	📄 Main 🔗 Refresh 🗟 Build 🌄 Environment 🖾 <u>C</u> ommon	
🔻 💁 Program	Location:	
💁 Flash	/home/zach/openocd/src/openocd	
Q gdb_server	Browse Workspace Browse File System Variables	
CopenOCD	Working Directory:	
	/home/zach/openocd/tcl	
	Browse Workspace Browse File System Variables	
	Arguments:	
	-f ./board/stm32f0discovery.cfg -f ./extra/stm32f0-openocd.cfg	
	Variables	
	Note: Enclose an argument containing spaces using double-quotes (").	

APPENDIX A – Erroneous Errors

If for some reason you build too soon, errors will always appear in your project even though your project compiles fine. CDT is unfortunately at fault and this problem can easily be fixed.

<pre>/* dirlist.cpp * fifd dirlist.cpp,v 1.1 2006/06/08 02:14:18 zwm Exp 5 * stood dirlist.cpp,v 5 * Revision 1.1 2005/06/08 02:14:18 zwm * Initial revision * / finclude <dorent.be finclude </dorent.be finclude finclude <</pre>	dirlist.cpp 🛛 🔀 👔 makemake.cpp	- C
<pre>\$Id: dirlist.cpp,v 1.1 2006/06/08 02:14:18 gem Ean \$ \$Log: dirlist.cpp,v 5 Nuclude citrat.cpp,v 5 Nuclude cit</pre>	/* dirlist.cpp	
<pre>* Story dirlist.cpp,v \$ * Revision 1.1 2006/06/08 02:14:18 sem * Initial revision * // einclude citype.h> einclude citype.</pre>	* * \$Id: dirlist.cpp,v 1.1 2006/06/08 02:14:18 swm Exp \$	
<pre>* Blog: drlat.cp,v 5 * Revision 1: 2006/06/06 02:14:18 sem * Tritial revision * * * * * * * * * * * * * * * * * * *</pre>	*	
<pre>* Initial revision */ */ */ */ */ */ */ */ */ */ * include :: */ * include :: *: * include :: *: * include :: *: *: *: * of streams *: * of streams *: * of streams *: * issue: *: *: *: *: *: *: *: *: *: *: *: *: *:</pre>	* \$Log: dirlist.cpp,v \$	
<pre>*/ */ */ */ */ */ */ */ */ */ */ */ */ *</pre>	* Initial revision	
<pre>*/ */ */ */ */ */ */ */ */ */ */ */ */ *</pre>	x	
<pre>#include <ctype.h> #include <cttype.h> #include <cttype.h< ttp=""> #include <cttype.h> #include <cttype.h> #include <cttype.h> #include <cttype.h> #include <cttype.h> #include <cttype.h< ttp=""> #include <cttype.h< ttp=""> #include <cttype.h> #include <cttype.h> #include <cttype.h< ttp=""> #include <cttype.h< ttp=""></cttype.h<></cttype.h<></cttype.h<></cttype.h<></cttype.h<></cttype.h<></cttype.h<></cttype.h<></cttype.h<></cttype.h<></cttype.h<></cttype.h<></cttype.h></cttype.h></cttype.h<></cttype.h<></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h<></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></cttype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></ctype.h></pre>	*/	
<pre>#include <dirent.h> #include <dirent.h> #include <dirent.h> #include <fstream> #include <fstream>#include <fstream>#</fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></fstream></dirent.h></dirent.h></dirent.h></pre>	<pre>#include <ctype.h></ctype.h></pre>	
<pre>#include <lostream> #include <lostream &="" &<="" <lostream="" include="" td=""><td><pre>#include <dirent.h></dirent.h></pre></td><td></td></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></lostream></pre>	<pre>#include <dirent.h></dirent.h></pre>	
<pre>#include cstream> #include cstream> #includ</pre>	#include <lostream></lostream>	
<pre>#include "dirlist.h" using namespace gtd; /* Constructor for Inserter object that just initializes fields * of struct */ FilelistInserter:iFileListInserter(const get<string> of, const char *suf) fileg(f), guffix(suf) {} /* Insertion operator for the previous inserter object. Writes a * space-separated list of file names in f. */ ostream & operator<(ostream &os, const FileListInserter &ost) { for(setString2::const.iterator, it = ins.files.begin();</string></pre>	#include <stream></stream>	
<pre>#include "dirlist.h" using namespace gtd; /* Constructor for Inserter object that just initializes fields * of struct */ FileListInserter:ifileListInserter(const set<string> ôf, const char *suf) : fileg(f), suffix(suf) {} /* Insertion operator for the previous inserter object. Writes a * space-sparated list of file names in f. */ ostream ô operator<(ostream ôos, const FileListInserter ôins) { for(setsstring::const_iterator.it = ins.files.begin();</string></pre>		
<pre>using namespace gtd; /* Constructor for Inserter object that just initializes fields * of struct */ FileListInserter::FileListInserter(const set<string> &f, const char *suf) : fileListInserter::FileListInserter(const set<string> &f, const char *suf) : fileListInserter::FileListInserter(const set<string> &f, const char *suf) : fileListInserter::FileListInserter(const set<string> &f, const char *suf) : fileListInserter::FileListInserter(const set<string> &f, const char *suf) : fileListInserter::FileListInserter(const set<string> &f, const char *suf) : fileListInserter::FileListInserter(const set<string> &f, const char *suf) : fileListInserter::FileListInserter(const set<string> & space-separated list of file names in f. */ /* Insertion operator</string></string></string></string></string></string></string></string></pre>	#include "dirlist.h"	
<pre>/* Constructor for Inserter object that just initializes fields * of struct */ FileListInserter::FileListInserter(const set<string> &f, const char *suf) files(f), suffix(suf) {} /* Insertion operator for the previous inserter object. Writes a * space-separated list of file names in f. */ Ostream & operator<<(ostream & ost, const FileListInserter & & &) { for(setsetring: const iterator, it = ins.files.begin(); it != ins.files.begin()) { // don't put space before first item os << " "; if (ins.suffix) { os << 0 DirList::basename(*it) << ins.suffix; } else { Problems @ Tasks @ Console [SuperMakemake] ** Build of configuration Debug for project SuperMakemake **** ke all </string></pre>	using namespace std;	
<pre>/* Constructor for Inserter object that just initializes fields * of struct * of struct * IllelistInserter::EllelistInserter(const set<string> &f, const char *suf) files(f), suffix(suf) {} /* Insertion operator for the previous inserter object. Writes a * space-separated list of file names in f. */ Stream & operator<<(ostream &os, const FileListInserter &ins) { for(setsstring::const iterator it = ins.files.begin(); it != ins.files.begin() } /// don't put space before first item os << " "; } if(ins.suffix) { os << " "; } if(ins.suffix) { os << " "; } if(ins.suffix) { os << " "; } if(ins.suffix) { os << " "; } if(ins.suffix) { os << " "; } if(ins.suffix) { os << " "; } if(ins.suffix) { os << " "; } if(ins.suffix) { os << " "; } if(ins.suffix) { visions of the properties T Build Console [SuperMakemake] ** Build of configuration Debug for project SuperMakemake **** ke all</string></pre>		
<pre>*/ FileListInserter::FileListInserter(const set<string> &f, const char *suf) : files(f), suffix(suf) {} /* Insertion operator for the previous inserter object: Writes a * space-separated list of file names in f. */ ostream & operator<(optream & os, const FileListInserter & ins) { for (setsstring: const iterator it = ins.files.begin(); it != ins.files.eng(); it! != ins.files.eng() / // don't put space before first item os << " "; } if(ins.suffix) { os << DirList::basename(*it) << ins.suffix; } else { Problems ② Tasks ③ Console ② Properties TBuild Console [SuperMakemake] ** Build of configuration Debug for project SuperMakemake **** ke all </string></pre>	/* Constructor for Inserter object that just initializes fields	r
<pre>FileListInserter::FileListInserter(const set<string> &f, const char *suf) : files(f), suffix(suf) {} /* Insertion operator for the previous inserter object. Writes a * space-separated list of file names in f. */ ostream & operator<(optream & os, const FileListInserter & ins) { for(setstring> const iterator it = ins.files.begin()); it != ins.files.eng(); +wit) { if(ins.files.eng()) { // don't put space before first item os << "; } if(ins.suffix) { os << Otherwise iterator it = ins.suffix; } else { /* Problems Tasks Console S Properties /* Properties /** Build of configuration Debug for project SuperMakemake **** ke all /** /**</string></pre>	*/	1
<pre>: files(f), suffix(suf) {} /* Insertion operator for the previous inserter object. Writes a * space-separated list of file names in f. */ ostream & operator<(ostream &os, const FileListInserter &os) { for(setSstring::const.iterator it = ins.files.begin(); it != ins.files.begin() ; ++it) { if (in != ins.files.begin()) { // don't put space before first item os << " "; } if (ins.suffix) { os << 0.01List::basename(*it) << ins.suffix; } else { // Problems ② Tasks ③ Console ③ Properties // Properties // Properties // //</pre>	FileListInserter::FileListInserter(const set <string> &f, const char *suf)</string>	
<pre>/* Insertion operator for the previous inserter object. Writes a * space-separated list of file names in f. */ ostream & operator<<(ostream &ost, const FileListInserter &ins) { for(set(string)::const iterator it = ins.files.begin();</pre>	: files(f), suffix(suf) {}	
<pre>* space-separated list of file names in f. */ ostream & operator<<(ostream & cost FileListInserter & ins) { for(setstring)const literator it = ins.files.begin(); it != ins.files.end(); ++it) { if(it != ins.files.begin()) { // don't put space before first item os << " "; } if(ins.suffix) { os << DirList::basename(*it) << ins.suffix; } else { // Problems ② Tasks ③ Console ※ Properties // //</pre>	/* Insertion operator for the previous inserter object. Writes a	r i i i i i i i i i i i i i i i i i i i
<pre>*/ ostream & operator<<(ostream &os, const FileListInserter &ins) { for(setsstring::const iterator it = ins.files.begin(); it != ins.files.begin()) { // don't put space before first item os << " "; } if(ins.suffix) { os << DirList::basename(*it) << ins.suffix; } else { // Problems Properties // Tasks @ Console S3 Properties // Properties // Properties // Build Console [SuperMakemake] *** Build of configuration Debug for project SuperMakemake **** ke all // // //</pre>	* space-separated list of file names in f.	
<pre>Optrace of optrace of solution of the second of the s</pre>	*/	
<pre>it != ins.files.end(); ++it) { if(it != ins.files.begin()) { // don't put space before first item os << " "; } if(ins.suffix) { os << DirList::basename(*it) << ins.suffix; } else { // Problems 2 Tasks Console 2 Properties // Properties // Properties // // //</pre>	Setteen a operator<<(Setteen aos, const ritelistinserter ains) i for(set <strino>::const iterator it = ins files begin():</strino>	1
<pre>++it) { if(if != ins.files.begin()) { // don't put space before first item os << " "; } if(ins.suffix) { os << DirList::basename(*it) << ins.suffix; } else { // Problems 2 Tasks @ Console 23 Properties // Properties // Problems // Properties // // //</pre>	it != ins.files.gnd();	
<pre>if(if != ins.files.begin()) { // don't put space before first item os << " "; } if(ins.suffix) { os << DirList::basename(*it) << ins.suffix; } else { // Problems 2 Tasks 2 Console 2 Properties // Problems 2 Tasks 2 Console 2 Properties // Problems 2 Tasks 2 Console 2 Properties // // //</pre>	++it) {	
<pre>} if(ins. suffix) { os << DirList::basename(*it) << ins. suffix; } else { roblems 2 Tasks 2 Console 2 Properties TBuild Console [SuperMakemake] *** Build of configuration Debug for project SuperMakemake **** ke all </pre>	if(if != ins.files.begin()) { // don't put space before first item	
<pre>if(ins. suffix) { os << DirList::basename(*it) << ins. suffix; } else { roblems 2 Tasks 2 Console 2 Properties TBuild Console [SuperMakemake] ** Build of configuration Debug for project SuperMakemake **** ke all </pre>	3	
os << DirList::pasename(*it) << ins. suffix; } else { Problems 2 Tasks 2 Console 2 Properties IT Build Console [SuperMakemake] *** Build of configuration Debug for project SuperMakemake **** ke all	if(ins.suffix) {	
Image: start Image: start	os << DirList::basename(*it) << ins.suffix;	
Problems Image: Console Signer Makemake IT Build Console [SuperMakemake] *** Build of configuration Debug for project SuperMakemake **** ke all		
Problems 2 Tasks Console 2 Properties T Build Console [SuperMakemake] *** Build of configuration Debug for project SuperMakemake **** ke all		
DT Build Console[SuperMakemake] *** Build of configuration Debug for project SuperMakemake **** ke all	Problems 🕢 Tasks 📮 Console 🛛 🗌 Properties	
*** Build of configuration Debug for project SuperMakemake **** ke all	T Build Console [SuperMakemake]	
ke all	** Build of configuration Debug for project SuperMakemake ****	
ike all	en anna an ann ann ann an an an an ann an a	
	ke all	
+ -ggdb -Wall - c makemake.cpp	⊦-ggdb -Wall -c makemake.cpp + -qqdb -Wall -c dirlist cpp	

**** Build Finished ****

(Example of error-just not with gcc)

Right click on your Template project Highlight index Click Rebuild

If this doesn't work delete your errors on the problems tab (right click, select delete) and try the Appendix A process again.

LICENSE

*You may not sell this file without the original author's permission

*Accreditation must be given to the original author/authors when republishing this document, or modified versions of this document

*The original author/authors retain ownership of this document

*Unless required by applicable law or agreed to in writing, this guide is distributed on an "AS IS"

*BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.