STM32 Discovery Board Projects

Notes:

- This tutorial was written for the STM32F3348-Discovery board and its STM32F334C8T6 microcontroller. If using a different STM Discovery or Nucleo board, basic project setup is the same, except for the microcontroller to be selected.
- 2. This document assumes that the ARM Keil MDK (Version 5.xx) has been installed, along with the STM32F3xx device family pack and ST-Link USB driver. Refer to the tutorial "Installing MDK-ARM" on the course web page.

ARM Keil Microcontroller Development Kit (MDK)

As illustrated in Figure 1, MDK-ARM Version 5.xx (*µVision*) comprises a set of core functions:

- Integrated Development Environment (IDE) with project manager and editor to create projects
- *ARM C/C++ compiler, assembler, and linker* to build projects (any combination of languages)
- Debugger to interactively debug projects on the target board or simulator
- Pack installer

One or more software packs must be installed, each of which provides resources (system/startup code, device drivers, etc.) for a specific family of microcontrollers. The appropriate software pack must be installed for each microcontroller family to be used.

• The required pack for these projects is *Keil::STM32F3xx_DFP*, which supports the *STM32F334C8T6* microcontroller on the *STM32F3348-Discovery* board.



Figure 1. MDK-ARM V5.xx (µVision) comprises the MDK Core and one or more Software Packs



ARM University Program Copyright © ARM Ltd 2013

Creating a New Project for the Discovery Board

Open µVision and from the menu bar select: Project → New µVision Project. In the Create New
Project Window (Figure 1), navigate to the folder in which you want to create the project (create the
folder if necessary), enter a project name (ex. Project1), and click Save.

🐺 Create New Project					×
← → ✓ ト 🔤 « CPSC 3300 → Pro	grammingProjects	> Sample 🗸 진	,⊂ Sea	irch Sample	
Organize 🔻 New folder		Project Folder			?
 ProgAssign3 ProgAssign4 ProgAssign5 ProgAssign6 ProgAssign7 RTE 	↑ Name	A No items match y	Status your search.	Date modified	
Sample	~ <				>
File <u>n</u> ame: Sample Proj Save as <u>type</u> : Project Files (*.uvproj	ect File ; *.uvprojx)				~
∧ Hide Folders			<u>S</u> av	re Cancel	

It is recommended that a separate folder be created for each project.

Figure 1. Creating a new µVision project "Sample"

2. In the Select Device for Target window (Figure 2), select microcontroller STM32F334C8T6.

Select Device for Target 'Target 1'		×
Device Software Packs Vendor: STMicroelectronics Device: STM32F334C8Tx Toolset: ARM Search:	_	
Image: Stm32F303 Image: Stm32F373 Image: Stm32F334 Image: Stm32F334C4 Image: Stm32F334C6 Image: Stm32F334C6 Image: Stm32F334C8 Image: Stm32F334C8 Image: Stm32F334K4 Image: Stm32F334K4	Description: STMicroelectronics' STM32F3 family offers an extensive range of enhanced I/Os and peripherals connected to two APB buses. - SRAM with HW parity checking - Timer with quadrature (incremental) encoder input - Timer with deadtime generation and emergency stop - 5 Vtolerant I/Os - Rail to-rail analog comparators - Operational amplifiers - 96-bit unique ID	~
	OK Cancel Help	,

Figure 2. Select the STM32F334C8T6 microcontroller.



ARM University Program Copyright © ARM Ltd 2013

3. In the *Manage Run-Time Environment* window (Figure 3), you can add microcontroller header and startup files, device drivers, and other software pack components to the project by checking the corresponding boxes. (These can also be added later, if desired.)

CPSC 3300 - Introductory assembly language programs:

CMSIS Core and Startup programs are unnecessary for testing simple assembly language programs. **Click OK, without checking any boxes,** to omit these from the project. The assembled program code will be placed beginning at the first byte of program memory.

OTHER PROJECTS:

For other projects: check the boxes for **CMSIS** → **CORE** and **Device** → **Startup**. (CMSIS = Cortex *Microcontroller Software Interface Standard*), plus boxes for any drivers or other software components you wish to include in the project. If you see STM32CubeMX in the Variant column, next to Device, click on STM32CubeMX and change it to Standalone.

The assembled/compiled code for your programs will be placed <u>after</u> that of the device startup program in program memory. The startup program is required for initializing the stack, clocks, and some other system resources, after which it jumps to the first instruction of your application program.



Figure 3. Include the CMSIS CORE and Startup files in the project. (Unless testing simple assembly language programs in CPSC 3300.)



ARM University Program Copyright © ARM Ltd 2013

- 4. Create and add source file(s) to the project in the **Project** pane of the μ *Vision* window (Figure 4). You may do this in either or both of the following ways.
 - Right-click on *Source Group 1*, select *Add New Item to Group 'Source Group 1'*, select a file type (*Asm File (.s) or C File (.c)*), enter the file name (Example: *Program1.s*), and click *Add*. This creates a blank file in the editor pane, into which you can type or paste your source program.
 - For files created outside of μVision, right-click on Source Group 1, select Add Existing Files to Group 'Source Group 1', locate and select file(s) to be added to the project and click Add.
 - Click *Close* after you've created/added all desired source files.



Figure 4. Add source file(s) to Source Group 1 in the Project pane, and build the project. (Note project pane above for simple assembly projects, without startup and CMSIS.)



ARM University Program Copyright © ARM Ltd 2013 The Architecture for the Digital World[®] Before going to Step 5, click the "Options for Target" button, and select the Linker tab as shown below. Check the box "Use Memory Layout from Target Dialog" (if not already checked) and click OK to close the window.

Options for Target 'Target 1'	Х
Device Target Output Listing User C/C++ (AC6) Asm Linker Debug Utilities	
✓ Use Memory Layout from Target Dialog X/O Base: ✓ Make RW Sections Position Independent B/O Base: ✓ Make RO Sections Position Independent B/O Base: ✓ Dont Search Standard Libraries ✓ ✓ Report 'might fail' Conditions as Errors disable Warnings:	
Scatter File	
Misc controls	
String OK Cancel Defaults Help	

- 5. After creating and saving source files, the project must be "built" by compiling/assembling the source files, correcting any syntax errors as necessary, and then linking the files into one downloadable program. This is performed by:
 - Click on the *Build* icon above the project pane (or from the menu bar select *Project* → *Build Project*). This will recompile/assemble only source file(s) that have changed.
 - OR: Click on the *Rebuild* icon above the project pane (or from the menu bar select *Project* → *Rebuild all target files*). This will compile/assemble all source files.
 - Read the messages in the Build Project pane. You must see the message

ARM University Program

Copyright © ARM Ltd 2013

".\MyProject.axf" - Error(s)", k Warning(s) before you can debug the program. (*MyProject.axf* is the object file (binary machine code) to be programmed into the microcontroller. If there are any errors, this file will NOT be created, and attempting to enter the debugger will fail.)

- If there are one or more errors/warnings, scroll up through the messages in the Build Project pane to identify the reason for each and where they are located in your program. Any errors must be corrected before the object file will be created.
- **NOTE:** The project must be rebuilt if you change any device, target, compiler, assembler, or linker options.



- 6. There are several options for testing and debugging projects with the μ *Vision* debug tools.
 - a) Load the program into the **simulator** for testing, rather than the physical microcontroller (*See* to Step 7 below.)
 - b) Download the program to the **Discovery board**, program it into the on-chip flash memory of the microcontroller and execute the program in the target hardware. (*See Step 8 below*.)
 - c) Execute a program that was **previously programmed** into the flash memory of the target microcontroller. (*See Step 9 below.*)
- 7. To download the compiled program to the <u>simulated</u> microcontroller's memory and initiate a debug session with the simulator, set up the project Debug options as follows.
 - From the menu bar, select Project → Options for Target 'Target 1' (or click on the Options for Target icon next to the target name in the menu bar.)
 - Select the *Debug* tab (Figure 5) and check the *Use <u>Simulator</u>* button.
 - If you wish to use a debug initialization file (described in Step 10 below), enter the file name in the *Initialization File* box below the *Use Simulator* button.
 - Click OK to close the Options for Target window.

Options for Target 'Target 1'	×			
Device Target Output Listing User C/C++ Asm Image: Settings Image: Settings Image: Limit Speed to Real-Time	Linker Debug Utilities C Use: ST-Link Debugger Settings			
✓ Load Application at Startup ✓ Run to main() Initialization File:	Initialization File:			
Restore Debug Session Settings Image: Breakpoints Image: Toolbox Image: Watch Windows & Performance Analyzer Image: Memory Display Image: System Viewer	Restore Debug Session Settings Image: Breakpoints Image: Toolbox Image: Breakpoints Image:			
CPU DLL: Parameter: SARMCM3.DLL -MPU -REMAP	Driver DLL: Parameter: SARMCM3.DLL -MPU			
Dialog DLL: Parameter: DCM.DLL PCM4	Dialog DLL: Parameter: TCM.DLL -pCM4			
OK Cancel Defaults Help				

Figure 5. Target Options Debug tab. "Use Simulator" selected, and debug initialization file specified.



ARM University Program Copyright © ARM Ltd 2013

- 8. To download the compiled program to the <u>Discovery board</u>, program it into the microcontroller's flash memory, and initiate a debug session, set up the project Debug options as follows.
 - From the menu bar, select Project → Options for Target 'Target 1' (or click on the Options for Target icon next to the target name in the menu bar.)
 - Select the *Debug* tab (Figure 6), check the *Use* button and select *ST-Link Debugger*.
 - If you wish to use a debug initialization file (described in Step 10 below), enter the file name in the *Initialization File* box below the *Use ST-Link Debugger* button.

Coptions for Target 'Target 1'				
Device Target Output Listing User C/C++ Asm	Linker Debug Utilities			
C Use Simulator Settings				
✓ Limit Speed to Real-Time				
✓ Load Application at Startup ✓ Run to main()	Load Application at Startup Run to main()			
Initialization File:	Initialization File:			
Edit	debug.ini Edit			
Restore Debug Session Settings	Restore Debug Session Settings			
I Breakpoints I Toolbox	I Breakpoints I Toolbox			
✓ Watch Windows & Performance Analyzer	✓ Watch Windows			
Memory Display System Viewer	Memory Display System Viewer			
CPU DLL: Parameter: SARMCM3.DLL -MPU -REMAP	Driver DLL: Parameter: SARMCM3.DLL -MPU			
Dialog DLL: Parameter: DCM.DLL pCM4	Dialog DLL: Parameter: TCM.DLL -pCM4			
OK Cancel Defaults Help				

Figure 6. Target Options Debug tab. "Use ST-Link Debugger" selected.

(Continue on next page for setting up the ST-Link Debugger)



ARM University Program Copyright © ARM Ltd 2013

 Click on Settings next to ST-Link Debugger to produce the Cortex-M Target Driver Setup Window (Figure 7), and select Port SW (Single-Wire debug) under Target Com.
 If Serial Number is blank, install/reinstall the ST-Link USB driver on your computer.

Debug Adapter Unit: ST-LINK/V2	SW Dev		Douioo Namo	Move
Shareable ST-Link	SWDIO	0x2BA01477	ARM CoreSight SW-DP (ARM Core	Up
Serial Number: 30FF6E065647373721571043 Version: HW: V2 FW: V2J3750 Check version on start Target Com Port: SW Clock Req: 10 MHz Selected: 1.800 MHz	C Aut	omatic Detection nual Configuration	ID CODE: Device Name: pdate IR Ien:	AP: 0
Debug Connect & Reset Options Connect: Normal Reset: Autodetect Reset after Connect Stop after Reset	t	Cache Optic	ons Download Options Code Verify Code Download Memory Download to Flash	ŧ

Figure 7. Single-Wire (SW) Debug Adapter port selected.

 Click on the *Flash Download* tab of the Cortex-M Target Driver Setup Window (Figure 8) and under *Programming Algorithm*, click the *ADD* button and select *STM32F3xx 256kB Flash*.

Then you can close the setup and options windows.

Cortex-M Target Driver Setup				×
Debug Trace Flash Download	Pack			
Download Function C Erase Full Chipi Frase Full Chipi Frase Sectors Do not Erase	 ✓ Program ✓ Verify ✓ Reset and F 	RAM for A Start: 0	Ngorithm x20000000 Size: 0x00001000	
Description	Device Size	Device Type	Address Range	_
STM32F3xx Flash	256k	On-chip Flash	0800000H - 0800FFFFH	_
		Start:	Size:	
	Add	Remove		
			OK Car	cel <u>Apply</u>

Figure 8. STM32F3xx flash programming algorithm selected



ARM University Program Copyright © ARM Ltd 2013

- 9. To initiate a debug session for a program previously programmed into the microcontroller's flash memory on the Discovery board, set up the project Debug options as in step 8 above, but on the Debug tab (Figure 6) uncheck the button *Load Application at Startup*, located beneath the *Use ST-Link Debugger* button. This will cause the debugger to be entered without downloading the program, thereby using the program currently in the microcontroller's flash memory.
- 10. A **debug initialization file** is useful to set up any desired initial register and memory values and other debug options when the debugger is first opened. This is a plain text file that contains one or more debugger commands; these commands may also be entered in the Debugger's Command Window (see Figure 9).

The following is a sample initialization file (*debug.ini* in Figures 5 and 6)

```
xPSR=0x01000000 //Set T bit of xPSR Register to Thumb Mode
PC=0x08000000 //Set starting program counter address
E INT 0x20000000 = 5,23,-100,200,201,-200,0,1000,2000
```

- The first line initializes the T (Thumb) bit to 1 in the Processor Status Register (xPSR) to force the processor into Thumb mode. The Cortex-M4 processor in the STM32F334C8T6 microcontroller supports only Thumb mode instructions, and its T bit is hard-wired to 1. However, the simulator initializes xPSR to all 0's, thereby setting T=0, which selects ARM mode. Attempting to execute a Thumb instruction in ARM mode will trigger an error exception. Therefore, the user must initialize T=1 at the start of simulation when using the simulator.
- The second line initializes the program counter (PC) to the address of the first instruction to be executed. This will normally be the first byte of flash memory at address 0x08000000. If using the device startup code, the initial PC will be loaded from a vector table and therefore this debug command should be removed.
- The third line enters a sequence of 32-bit data values (separated by commas) into consecutive memory words, beginning at address 0x20000000 (the first byte SRAM). The "INT" indicates that the data are to be encoded and written as 32-bit values. Type "CHAR" would indicate 8-bit data. This E command is useful for entering test data for small programs.
- 11. To initiate a debug session, click the **Debug** icon in the μ Vision window (Figure 4), or select **Debug** \rightarrow Start/Stop Debug Session. Depending on how you set up the debug options, this will open the Debug Window (Figure 9) after:
 - downloading the project code to the simulator memory (if set up via Step 7),
 - downloading the project code to the target board and programming it into the microcontroller's flash memory (if set up via Step 8),
 - skipping the download (if set up via Step 10)

ARM University Program

Copyright © ARM Ltd 2013

At this point, you may run/stop/step the program, set breakpoints, select watch variables, etc. Click on the **Debug** icon to stop the debug session and return to the μ Vision IDE.



The Architecture for the Digital World[®]



Figure 9. MDK uVision Debug Window.

Use of the debugger is described in the document **Debugging Projects with MDK-ARM**, available on the course web page.

For further information, refer to "*Keil uVision MDK - uVision Lab for the STM32F4 Discovery Board*" (<u>http://www.keil.com/appnotes/docs/apnt_230.asp</u>): App Note 230, Section 20 (Creating your own project from scratch) This refers to the STM32F4 family of devices, but is mostly applicable to STM32F3 devices as well.



ARM University Program Copyright © ARM Ltd 2013