

Electrical and Computer Engineering

Computer Design Lab – ENCS4110

Introduction to ARM Assembly Language and Keil uVision5

Objectives

 Introduce some of the ARM architecture to students.
 Begin to use the lab tool - Keil uVision 5.
 The students will create a project and write an ARM assembly language program based on a simulated target.

The ARM (Advanced RISC Machine) architecture is introduced in the class (also see <u>http://www.arm.com</u>.) Keil MDK-ARM is a complete software development toolkit for ARM processor-based microcontrollers. Keil uVision5 will be used in the lab. The ARM Cortex-M3 processor will be examined with the STM32VLDISCOVERY board. The following is some important information for you.

Important Information

 In the lab room Masri207, computers are running the operating system Windows 10 Pro, and ARM Software Microcontroller Development Kit Version 5.21a (Keil uVision5) is installed.
 To install it in your home computer, you can download the following files: <u>-ftp/pub/class/301/ftp/uVision5/MDK521a.EXE</u> -ftp/pub/class/301/ftp/uVision5/Keil.STM32F1xx_DFP.2.1.0.pack
 Here is the Link to Keil Tools.
 To know more about Keil, visit <u>http://www.keil.com/</u>
 To see STM32VLDISCOVERY board, visit <u>STM32VLDISCOVERY Board</u>.
 To see The Cortex-M3 Instruction Set, visit <u>Cortex-M3 Devices Generic User Guide</u>.

Create an ARM Assembly Language Program

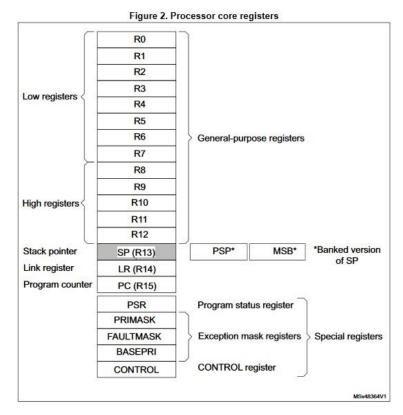
To create an assembly language program, you need to use a text editor such as **NotePad** in Microsoft Windows environment. There is a text edit in the Keil uVision5 for you to use too. The file name must have a **.s** at the end. Let's look at the following program called **FirstArm.s** on a PC. The file **FirstArm.s** contains the source code of the program to load registers and demonstrate a few other operations. We will use Keil uVision5 to create a project and execute this program so that you can get a feel of how Keil uVision5 works.

```
;The semicolon is used to lead an inline documentation.
;This is the first ARM Assembly language program you see in the lab.
;This program skeleton was from Dave Duguid and Trevor Douglas in summer 2013.
;When you write your program, you could have your info at the top document block.
;For Example: Your Name, Student Number, what the program is for, and what it does
etc.
;;; Directives
         PRESERVE8
         THUMB
; Vector Table Mapped to Address 0 at Reset
; Linker requires Vectors to be exported
         AREA
                 RESET, DATA, READONLY
         EXPORT Vectors
 Vectors
       DCD 0x20001000
                           ; stack pointer value when stack is empty
                    ; The processor uses a full descending stack.
                    ;This means the stack pointer holds the address of the last
                    ;stacked item in memory. When the processor pushes a new item
                     ;onto the stack, it decrements the stack pointer and then
                     ;writes the item to the new memory location.
         DCD Reset Handler ; reset vector
         ALIGN
 The program
 Linker requires Reset Handler
```

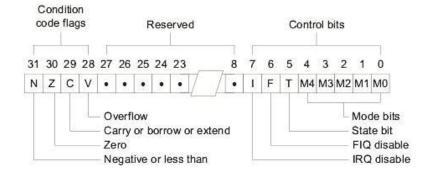
References:

- 1. A complete list of DIRECTIVES from ARM Information Center
- 2. Cortex-M3 Devices Generic User Guide
- 3. Cortex-M3 Programming Manual

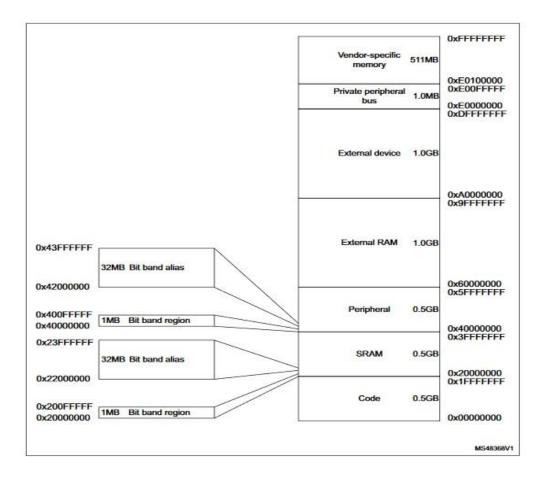
ARM Cortex-M3 Core Registers



Here is the Program Status Register Format:



ARM Cortex-M3 Memory Map



STM32F100xB Memory Map

STM32F100xB Memory Map

STM32F100RB Datasheet

Start up Keil uVision5

Before you start up, you are recommended that you create a folder to hold all your project files.

For example: you can have a folder "FirstARM-Project" ready before hand.



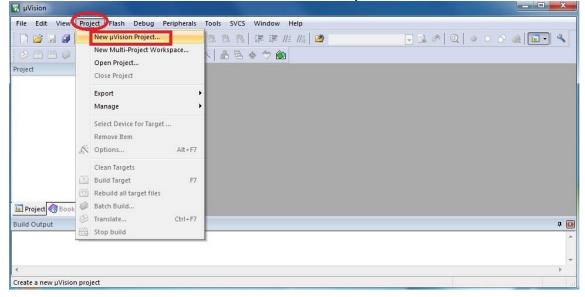
You can start up **uVision5** by clicking on the icon from the desktop or from the "Start" menu or "All Programs" on a lab PC.

The following screen is what you will see.

Wision		_ — X
File Edit View Project Flash Debug	Peripherals Tools SVCS Window Help	
n 🗃 🖬 🖉 8 🔤 🛍 n re	🔗 🌳 巻 巻 巻 巻 澤 澤 ////// 🙆	- i 🥙 🔍 💿 • 🔿 🔬 🔳 - 🔧
	🗔 🔊 🗂 🗟 🗟 🗇 🍘	
Project 📮 🗵		
E Project Books { } Func] 0, Temp]		
Build Output		4 🖬
		*
		*
*		

Create a project

Let's create our first ARM **uVision5** project now. To create a project, click on the "Project" menu from the **uVision5** screen and select "New uVision Project...".



Then, select the folder that you prepared for, give project a name and save it.

rganize 🔻 New folder					1
Downloads ^ Name	Date modified	Туре	Size		
💯 Recent Places	NI- ikaan anak	ch your search.			
-17.92 No.	No items mat	en your search.			
Jibraries					
Documents					
Music					
Pictures Videos					
Videos					
Computer					
Local Disk (C:)					
👝 Local Disk (D:)					
Removable Disk I					
File name: FirstARM					
Save as type: Project Files (*.uvproj; *.uvprojx)					

From the "Select Device for Target" window, select "STMicroelectronics" and then "STM32F1 Series".

📴 I\CS301\ARM5\FirstARM-project\FirstARM.uvproj - µVision	
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help	
□◎日間 2 4 2 2 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	🕘 O O 🔐 🔲 🔧 🛛
🔗 🗂 🗃 🤪 📇 🔤 Select Device for Target 'Target 1'	
Project Device	
Software Packs Vendor: cunknown> Device: cunknown> Toolset: cunknown> Search: Description: Description: STMicroelectronics STMicroelectronics<	
E Project Books D Fun	
Build Output OK Cancel Help	□ □ [∓] ⊠
· · · · · · · · · · · · · · · · · · ·	
4	
	1

click on "+" beside "STM32F100" and then select "STM32F100RB" and click on "OK".

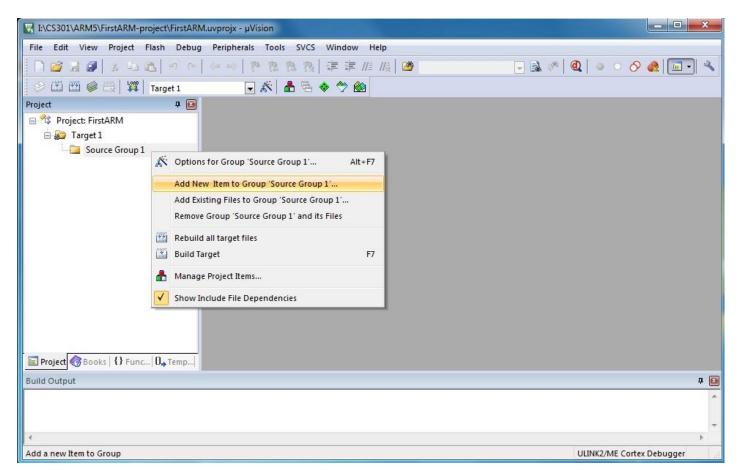
R I\CS301\ARM5\FirstARM-project\FirstARM.uvproj - µVision	- O X
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help	
□ 2 3 3 4 3 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	o 🔗 🏨 🔳 🔧 💧
😢 🖆 🥔 📇 👹 Select Device for Target 'Target 1'	
Project Device	
Software Packs Vendor: STMicroelectronics Device: STM32F100RB Toolset: ARM	
Search:	
Desgription: STM32F100 STM32F100C4 STM32F100C6 STM32F100C8 STM32F100R4 STM32F100R4 STM32F100R6 STM32F100R8 STM32F100R8	
Build Output OK Cancel Help	4
	l l

Make sure you click on "OK" for the following pop up window.

oftware Component	Sel.	Variant		Version	Description	
🛛 🚸 Board Support		MCBSTM32E	-	2.0.0	Keil Development Board MCBSTM32E	
🛛 🚸 CMSIS					Cortex Microcontroller Software Interface Components	
🚸 CMSIS Driver	1.1				Unified Device Drivers compliant to CMSIS-Driver Specifications	
🚸 Compiler					ARM Compiler Software Extensions	
🚸 Device	12		j,		Startup, System Setup	
🚸 File System		MDK-Pro	-	6.8.0	File Access on various storage devices	
🚸 Graphics	1	MDK-Pro	-	5.32.2	User Interface on graphical LCD displays	
🚸 Network		MDK-Pro	-	7.2.0	IPv4/IPv6 Networking using Ethernet or Serial protocols	
🚸 USB		MDK-Pro	-	6.8.0	USB Communication with various device classes	
/alidation Output		Descrip	tion			
alidation Output		Descrip	tion			
/alidation Output		Descrip	tion			
alidation Output		Descrip	tion			
/alidation Output		Descrip	tion			

Create Source File and Add Source File to the Project

Right click on "Source Group 1" and then select "Add New Item to Group 'Source Group 1'...".



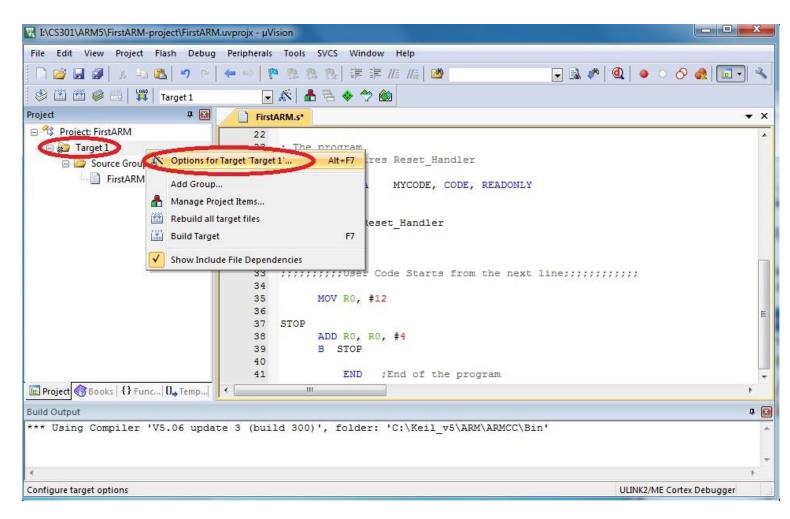
You will see the following window and make the suggested selections to proceed.

I:\CS301\ARM5\FirstA	ARM-project\FirstARM.uvprojx - µVi	sion		3
File Edit View Pro	ject Flash Debug Peripherals	Tools SVCS Window Help		
🗋 🗃 🗃 🎒 🕺	10 12 1 10 10 10 10 10 10 10 10 10 10 10 10 1	<u> </u>	। 🕞 🗞 🖉 🖉 🖉 🖉 🖉	2
🛛 🔗 🖽 🍘 🖶 (Add New Item to Group 'Source Gr	oup 1'		
Project Project: FirstARI Project I Source (C File (c) C File (cpp) Asm File (cpp) Asm File (s) h Header File (h) Text File (bt) Image File (.*) User Code Template	Create a new assembler source file and add it to the project.		
E Project Books 4	Type: Asm File (.s) Name: FirstARM.s Location: I:\CS301\ARM5	FirstARM-project	Help	
<u> </u>			► ULINK2/ME Cortex Debugger	1.2
			OLINK2/WE COTTEX DEDUGGET	1.13

You will see the "FirstARM.s*" text edit window. That is the place you will write your ARM Assembly language program. For a test, you can copy and paste the example program into this window. You can click on the "save" buttom to save your project.

I:\CS301\ARM5\FirstARM-project\FirstARM	l.uvprojx - μVision	×
File Edit View Project Flash Debug	Peripherals Tools SVCS Window Help	
🗍 🎯 🖬 🕼 🛦 🖧 🔊 P		
🔗 🖾 🕮 🥔 🔜 🔤 Target 1	🗔 🔊 📥 🗟 🧇 🍘	
Project 🗣 📧	FirstARM.s*	▼ ×
🖃 🔧 Project: FirstARM		
- Target 1	23 ; The program	
😑 🦢 Source Group 1	24 ; Linker requires Reset Handler	
	25	
FirstARM.s	26 AREA MYCODE, CODE, READONLY	
	27	
	28 ENTRY	
	29 EXPORT Reset_Handler	
	30	
	31	
	32 Reset_Handler	
	<pre>33 ;;;;;;;;User Code Starts from the next line;;;;;;;;;;;</pre>	
	34	
	35 MOV R0, #12	
	36	=
	37 STOP	
	38 ADD R0, R0, #4	
	39 B STOP	
	40	
	41 END ;End of the program	-
E Project		•
Build Output		џ 🛛
*** Using Compiler 'V5.06 updat	te 3 (build 300)', folder: 'C:\Keil_v5\ARM\ARMCC\Bin'	*
		*
4		
	ULINK2/ME Cortex Debugge	r

You can right click on "Target 1" and then select "options for Target 'Target 1'..." the same as the following screen.

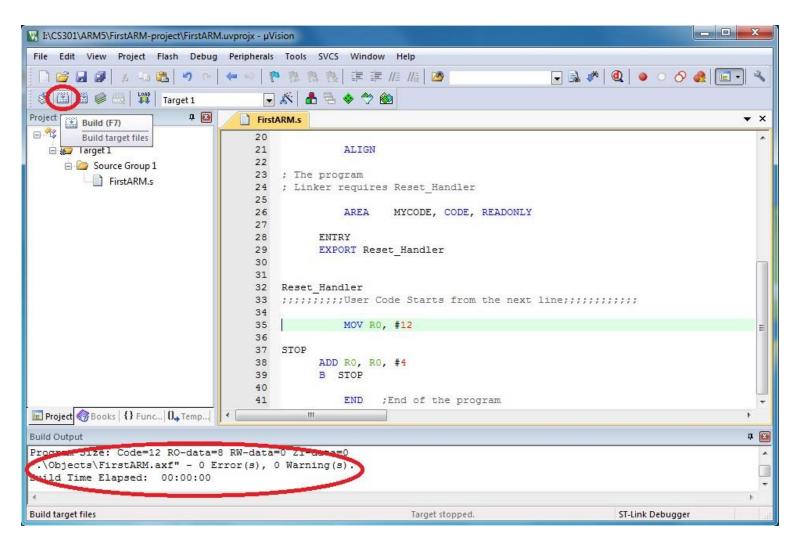


Please click on "Debug" and then select "Use Simulator".

-	to Real-Time	Use: ST-Lir	k Debugger Settings
Load Application File:		Load Applica	tion at Startup 🔽 Run to main()
	E dit		Edit
	Vindows & Performance Analyzer	I Breakpoir I Watch W I Memory [lindows
Memory	Display IV System viewer	I♥ Memory L	Display 🔽 System Viewer
	Parameter:	Driver DLL:	Parameter:
	Parameter:		Parameter:
CPU DLL: SARMCM3.DLI	Parameter:	Driver DLL:	Parameter:
CPU DLL:	Parameter:	Driver DLL:	Parameter:

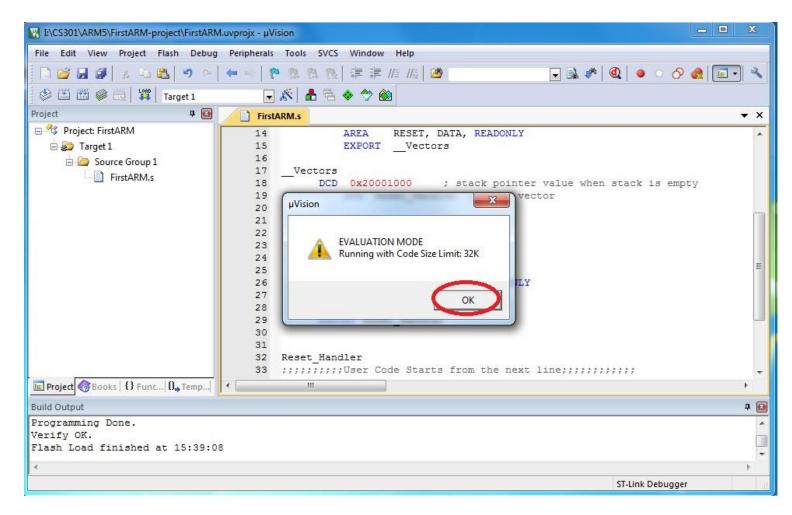
Build your project

Click on the "Build" button or from the "Project" menu, you will see the following screen.



Run the program in your project

When the assembler is happy with the program, we can run the program by selecting "Start/Stop Debug Session" from the "Debug" menu or clicking on the debug button.



Click on "OK" for the pop up window showing "EVALUATION MODE, Running with Code Size Limit: 32K".

Open your uVision5 to full screen to have a better and complete view. The left hand side window shows you the registers and the right side window shows the program code. There are some other windows open. You may adjust the size of them to see better.

Run the program step by step, you can observe the change of the values in the registers.

I:\CS301\ARM5	\FirstARM-project\First	RM.uvprojx - µVision	_ 0 ×
File Edit View	Project Flash Del	ug Peripherals Tools SVCS Window Help	
) x -a 🕰 🔊	> 🖛 -> 🕐 🏡 🏡 譁 譁 //: //: 🖉 📃 🔽 🔜 💐 🔍 🔵 📀 🌢	
RST 🗐 🖾 🕻	(f) ⊕ ⊕ (f)		
legisters	·	Disassembly	4
-	{*} Step (FII)		1
Register	Step one line	ADD NO, NO, #4 C>0x0800000C F1000004 ADD r0,r0,#0x04	
Core R0	0x0000000C	39: B STOP	
RU R1	0x00000003	0x08000010 E7FC B 0x0800000C	
R1 R2	0x40022000	0x08000012 0000 MOVS r0,r0	
R3	0x04C11DB7	0x08000014 FFFFFFF DCD 0xFFFFFFF	
R4	0x00000000	A-AAAAAAAA EEEEEEEE DAD A-EEEEEEEE	
	0x00000000		
R6	0x00000000	FirstARM.s	-
	0x00000000	24 ; Linker requires Reset Handler	
	0x0000000	25	
	0x20000160	26 AREA MYCODE, CODE, READONLY	
R10	0x0000000	27	
R11	0x0000000	28 ENTRY	
R12	0x0000000	29 EXPORT Reset Handler	
R13 (SP)	0x20001000	30	
R14 (LR)	0xFFFFFFFF	31	
R15 (PC)	0x0800000C	32 Reset Handler	
🖻 xPSR	0x01000000	33 ;;;;;;;;User Code Starts from the next line;;;;;;;;;;	
N	0	34	
Z	0 -	35 MOV R0, #12	
с	0	36	
V	0	37 STOP	
Q	0	38 ADD R0, R0, #4	
T	1	39 B STOP	
IT.	Disabled	40	
ISR	0	41 END ;End of the program	
t Banked			
Project Re	egisters	(<u> </u>	
ommand		📮 🔝 Call Stack + Locals	д
**	- 4 **	Name Location/V Type	
	ed version with y used: 20 Bytes	32768 Byte Code Size Limit (0%)	
		•	
>			
SSIGN Break	Disable BreakEna	ble BreakKill BreakList 🔂 Call Stack + Locals 🔲 Memory 1	

Click on the "Start/Stop Debug Session" from the "Debug" menu or click on the debug button to stop executing the program.

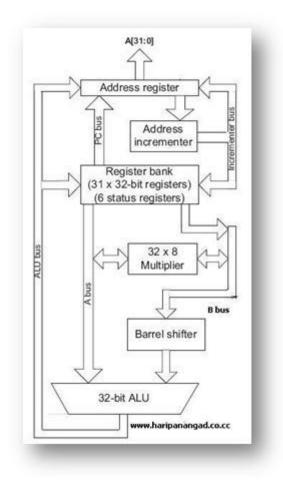
We will analyze the program and see how it works.

It works with both the simulated target and the real circuit board <u>STM32VLDISCOVERY Board</u>. We will demonstrate it in the lab for you.

ARM Architecture

ARM processors are mainly used for low-power and low cost applications such as mobile phones, communication modems, automotive engine management systems, and hand-held digital systems.

Here is a diagram of the ARM architecture for your reference.



```
ARM Architecture is an Enhanced RISC Architecture.It has large uniform Register file and uses Load Store Architecture.i.e. operations operate on registers and not in memory locations.ARM Architecture instructions are of uniform and fixed length.It is a 32 bit processor.It also has 16 bit variant called THUMB.i.e. it can be used as 32 bit and as 16 bit processor.
```

ARM cores are licensed to partners/manufacturers so as to develop and fabricate new microcontrollers around same processor cores. A microcontroller is a small computer on a single integrated circuit containing a processor core, memory, and programmable input/output peripherals.

The ARM Cortex-M3 microcontroller will be used in the lab with the STM32VLDISCOVERY board. For more information, visit <u>STM32VLDISCOVERY Board</u>.

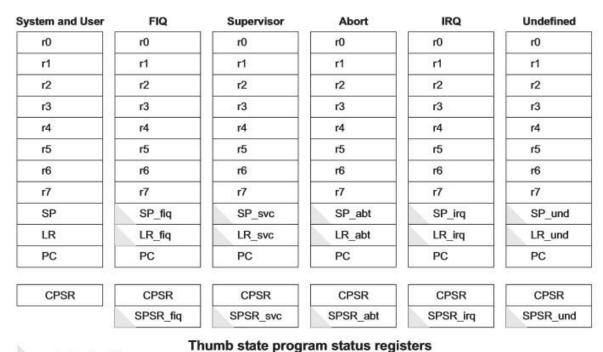
ARM Registers

Here is the Register Organization in ARM State.

	System and User	FIQ	Supervisor	Abort	IRQ	Undefined
	01	rO	ſŨ	On	On	On
	rl	rl	h	rl	rl	r1
	r2	r2	r2	r2	r2	r2
	r3	r3	r3	r3	r3	r3
	r4	r4	r4	r4	r4	r4
	r5	r5	đ	r5	г5	r5
	r6	r6	r6	r6	r6	r6
neral registers	r7	17	r7	r7	r7	17
enerar registers	r8	r8_fiq	r8	r8	r8	81
	en	r9_fiq	en	en	en	en
	r10	r10_fiq	r10	r10	r10	r10
	r11	r11_fiq	r11	r11	r11	r11
	r12	r12_fiq	r12	r12	r12	r12
	r13	r13_fiq	r13_svc	r13_abt	r13_inq	r13_und
	r14	r14_fiq	r14_svc	r14_abt	r14_inq	r14_und
ogram counter	r15 (PC)	r15 (PC)	r15 (PC)	r15 (PC)	r15 (PC)	r15 (PC)
rogram status	CPSR	CPSR	CPSR	CPSR	CPSR	CPSR
registers		SPSR_fiq	SPSR_svc	SPSR_abt	SPSR_irq	SPSR_und

ARM-state general registers and program counter

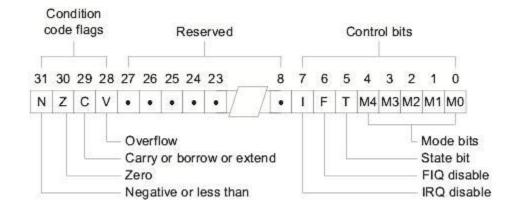
Here is the Register Organization in THUMB State.



Thumb state general registers and program counter

= banked register

Here is the Program Status Register Format:



ARM Instructions

Here are a few sample ARM Instructions for you to test out for this lab:

```
MOV
       R2, #0x76
; Move the 8-bit Hex number 76 to the low portion of R2
     R2, #0x7654
MOV
; Move the 16-bit Hex number 7654 to the low portion of R2
MOVT R2, #0x7654
; Move the 16-bit Hex number 7654 to the high portion of R2
MOV32 R2, #0x76543210
                            ; Move the 32-bit Hex number 76543210 to the R2
LDR R2, = 0x76543210
                            ; Load R2 with the 32-bit Hex number 76543210
     R1, R2, R3
                            ; R1 = R2 + R3
ADD
ADDS R1, R2, R3
                            ; R1 = R2 + R3, and FLAGs are updated
     R1, R2, R3
                            ; R1 = R2 - R3
SUB
SUBS R1, R2, R3
                            ; R1 = R2 - R3, and FLAGs are updated
       LABEL
                             ; Branch to LABEL
В
```

The entire list of the Instructions can be found in the <u>Cortex-M3 Devices Generic User</u> <u>Guide.</u>

OR see The Cortex-M3 Instruction Set in <u>Cortex-M3 Devices Generic User Guide</u>, in Chapter 3: The Cortex-M3 Instruction Set.

Lab Assignment

Write your first ARM assembly language program MyFirstARM.s.

The program will execute the following instructions. You will run the program step by step, observe and answer the question after each statement.

```
#0x01
MOV
       R2,
                            ; R2 = ?
                             ; R3 = ?
       R3,
              #0x02
MOV
;Other examples to move immediate values
                            ; R5 = ?
MOV
       R5,
           #0x3210
MOVT
       R5, #0x7654
                        ; R5 = ?
MOV32 R6, #0x87654321
                            ; R6 = ?
       R7, = 0 \times 87654321
LDR
                            ; R7 = ?
      R1,R2,R3
                            ; R1 = ?
ADD
MOV32 R3, #0xFFFFFFF
                            ; R3 = ?
ADDS
     R1,R2,R3
                             ; R1 = ?
                             ; specify Condition Code updates
                            ; R1 = ?
SUBS
     R1,R2,R3
                            ; specify Condition Code updates
                           ; R4 = ?
MOV
             #0xFFFFFFFF
       R4,
ADD
       R1,R2,R4
                            ; R1 = ?
                     ; How did that operation affect the flags in CPSR?
       R1,R2,R4
                     ; R1 = ?
ADDS
                      ; Please specify Condition Code updates
                      ; and now what happened to the flags in the CPSR?
       R2, #0x0000002
                            ; R2 = ?
MOV
                             ; R1 = ?
ADDS
       R1,R2,R4
                            ; again, what happened to the flags?
MOV
       R2,
              #0x0000001
                           ; R2 = ?
       R3,
MOV
             #0x0000002
                            ; R3 = ?
ADDS R1,R2,R3
                             ; R1 = ?
                             ; Add some small numbers again
                             ; and check the flags again .....
; Add numbers that will create an overflow
MOV
       R2,
             #0x7FFFFFFF; R2 = ?
                            ; R3 = ?
MOV
       R3,
             #0x7FFFFFFF
                             ; R1 = ?
ADDS
      R1,R2,R3
                             ; Check the flags in the CPSR?
```

You will hand in the following:

- 1. The screenshot of the program successfully built in Keil uVision.
- 2. The source code in the file **MyFirstARM.s** with the answers.