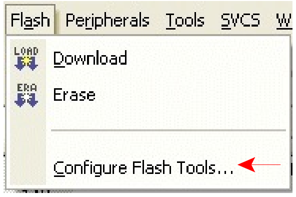
Downloading software to an ARM embedded board

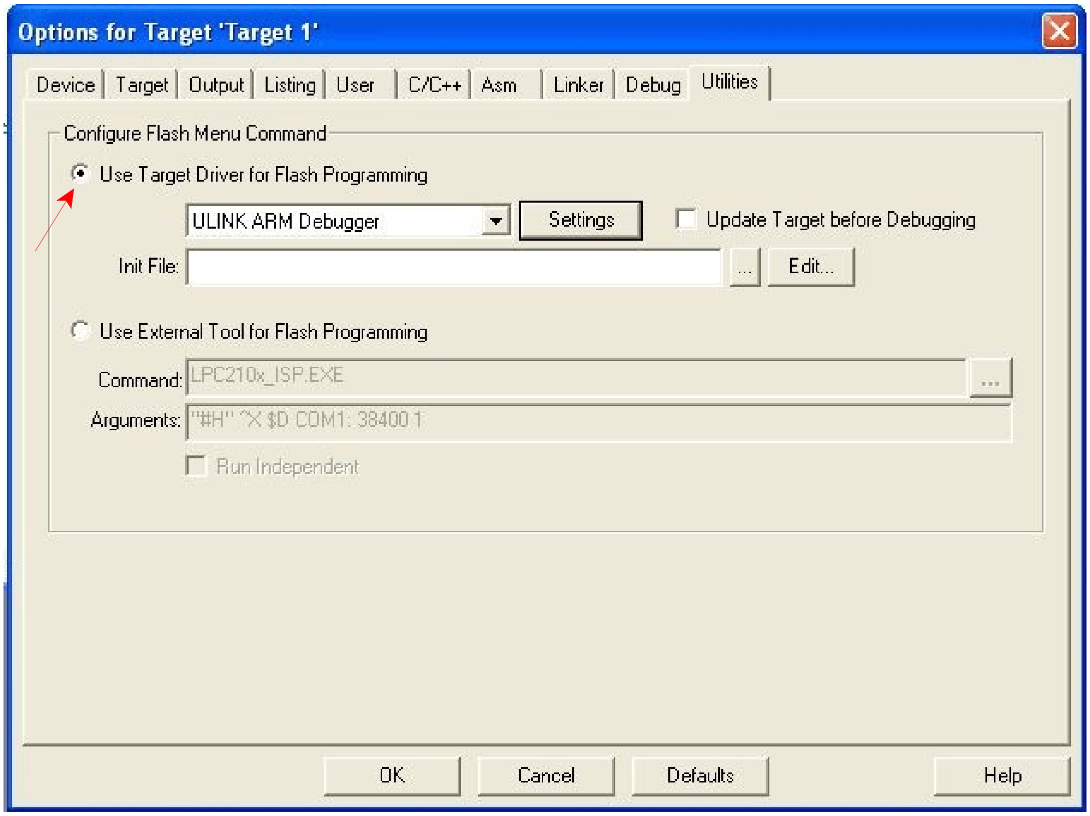
An assumption here is that you know how to create ARM software with the Keil development environment and how to test it using the simulator. The details covered here are about downloading the software to a hardware board and using the hardware debugger.

After writing the software and successfully “building the target” the program is ready for downloading. Note that at the start of the program code there will be many lines of equates that define standard values or addresses followed by LDR and other instructions that set up exception handling and interrupts (topics that have not yet been covered in class).

On the *Flash* drop down menu select *Configure Flash Tools*.



This will open up a menu that has several tabs. Select *Utilities* by clicking on it:



Configure Flash Menu Command

● Use Target Driver for Flash Programming

ULINK ARM Debugger

□ Update Target before Debugging

Init File:

○ Use External Tool for Flash Programming

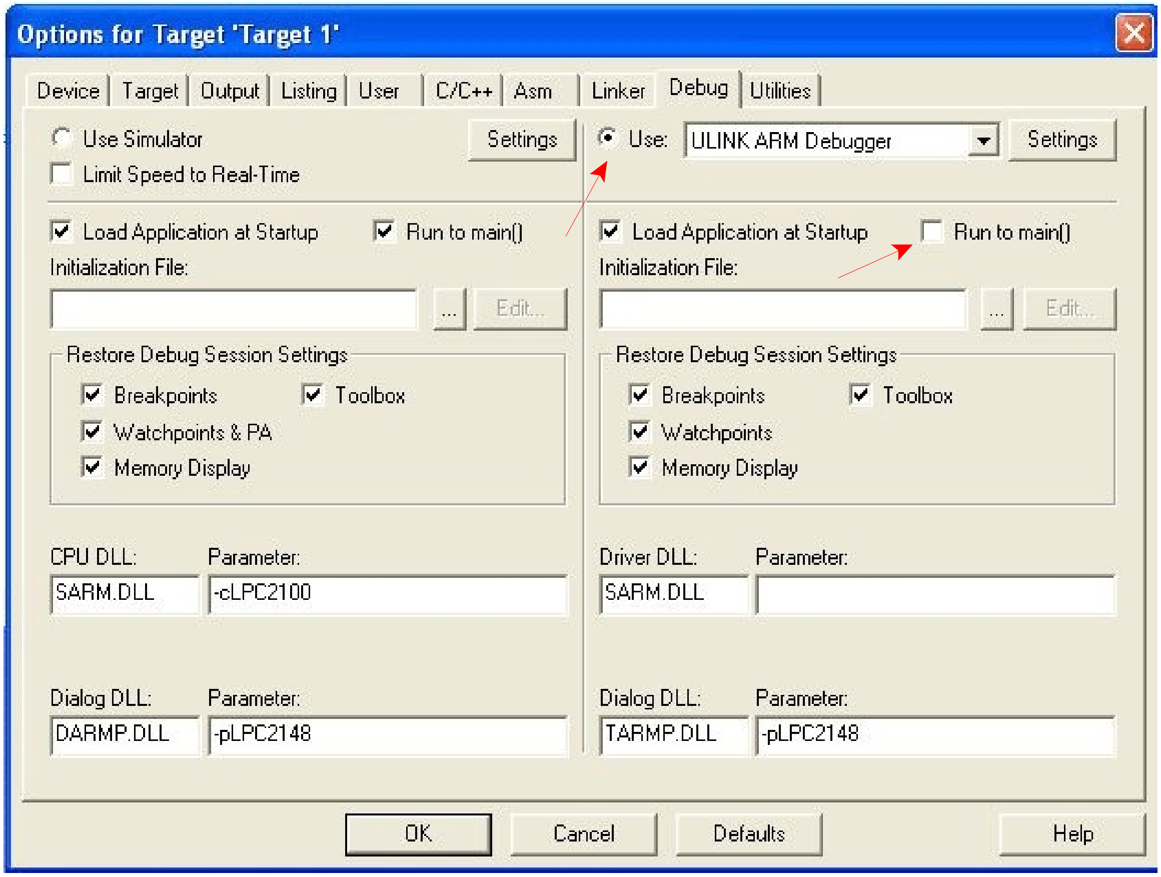
Command: LPC210x_ISP.EXE

Arguments: "#H" ^X $D COM1: 38400 1

□ Run Independent

Click *Use Target Driver for Flash Programming* if it isn’t already selected. ULINK ARM Debugger should be visible just below as shown above.

Then select the *Debug* tab and click *Use: ULINK ARM Debugger* if it isn't already selected. Also, unselect Run to main().



Options for Target 'Target 1'

Device | Target | Output | Listing | User | C/C++ | Asm | Linker | Debug | Utilities

○ Use Simulator Settings
□ Limit Speed to Real-Time

☑ Load Application at Startup ☑ Run to main()
Initialization File:

Restore Debug Session Settings
☑ Breakpoints ☑ Toolbox
☑ Watchpoints & PA
☑ Memory Display

CPU DLL: SARM.DLL Parameter: -cLPC2100

Dialog DLL: DARMP.DLL Parameter: -pLPC2148

● Use: ULINK ARM Debugger Settings

☑ Load Application at Startup □ Run to main()
Initialization File:

Restore Debug Session Settings
☑ Breakpoints ☑ Toolbox
☑ Watchpoints
☑ Memory Display

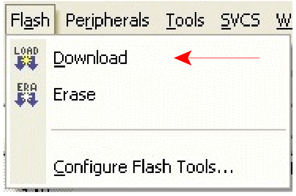
Driver DLL: SARM.DLL Parameter:

Dialog DLL: TARMP.DLL Parameter: -pLPC2148

Default parameters on the other tabs should work ok. Click OK.

Make sure the USB cable is connected between the small debug interface board that is plugged into the ARM CPU board and the PC's USB port.

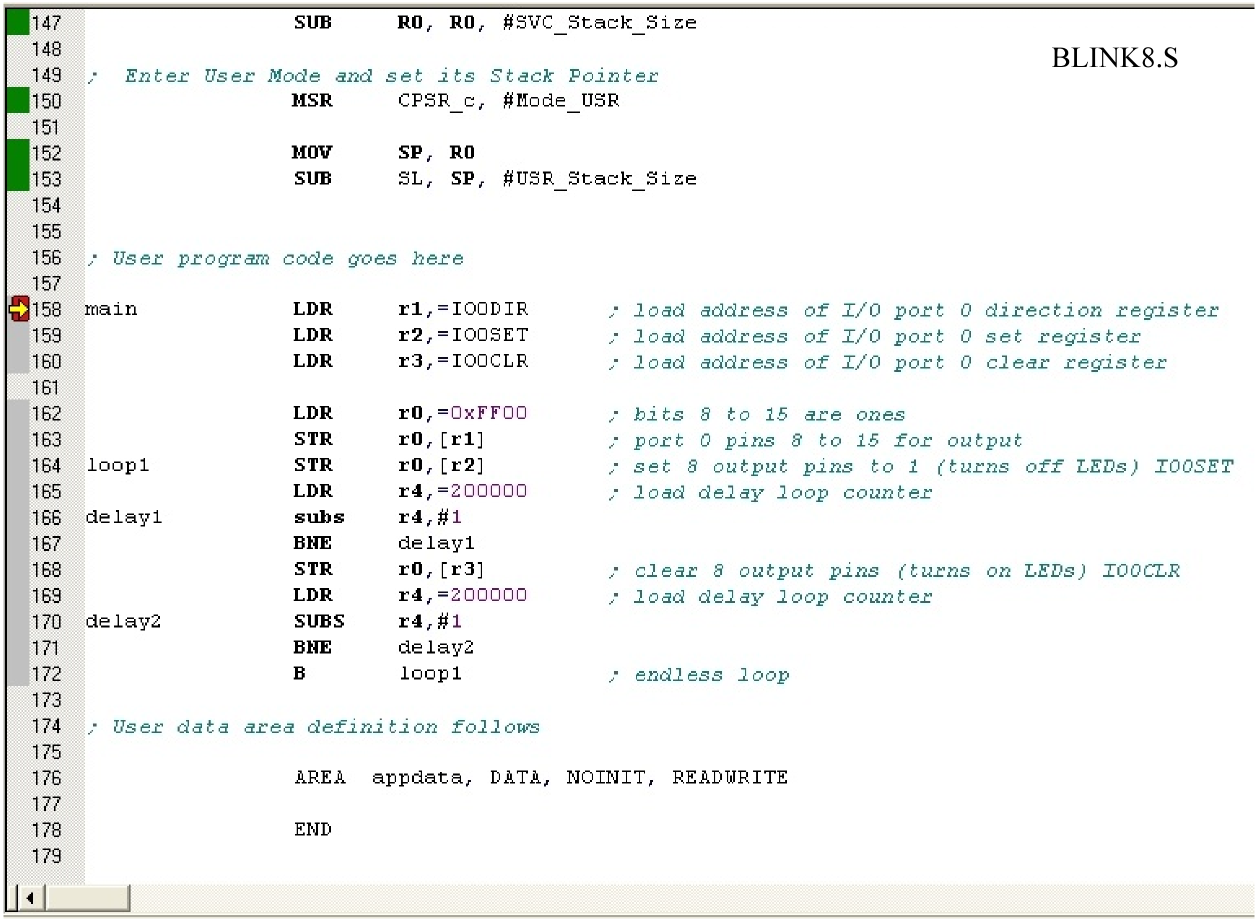
Again click on *Flash* from the tool bar and *Download*.



Downloading to the CPU board should begin and complete quickly. A blue LED on the Debug board should light (possibly fading in/out) and a green power light on the CPU board should be lit. Once the program is downloaded it will immediately begin running. If there are no bugs it will operate as you desire. If it isn't working properly debugging is in order.

To start the debugger click on Debug on the main tool bar and then Start Debugging (or press Cntrl-F5). The program counter will have zero in it and thus will be pointing to the first instruction in your program. This is an LDR instruction that is part of setting up what we call the run-time environment. Scroll down the source window until you come to the first instruction in the main part of the program. Double click on that statement to set a break point (a red block will appear at the start of that program line). Click run to execute to that point. You cannot single step through some of the instructions that set up interrupts.

With the execution pointer set to the first instruction in main, debugging can commence with single stepping or setting break points and running to them.



BLINK8.S

```
                SUB     R0, R0, #SVC_Stack_Size

; Enter User Mode and set its Stack Pointer
                MSR     CPSR_c, #Mode_USR

                MOV     SP, R0
                SUB     SL, SP, #USR_Stack_Size


; User program code goes here

main            LDR     r1,=IO0DIR      ; load address of I/O port 0 direction register
                LDR     r2,=IO0SET      ; load address of I/O port 0 set register
                LDR     r3,=IO0CLR      ; load address of I/O port 0 clear register

                LDR     r0,=0xFF00      ; bits 8 to 15 are ones
                STR     r0,[r1]         ; port 0 pins 8 to 15 for output
loop1           STR     r0,[r2]         ; set 8 output pins to 1 (turns off LEDs) IO0SET
                LDR     r4,=200000      ; load delay loop counter
delay1          subs    r4,#1
                BNE     delay1
                STR     r0,[r3]         ; clear 8 output pins (turns on LEDs) IO0CLR
                LDR     r4,=200000      ; load delay loop counter
delay2          SUBS    r4,#1
                BNE     delay2
                B       loop1           ; endless loop

; User data area definition follows

                AREA  appdata, DATA, NOINIT, READWRITE

                END
```

Note that the contents of memory can be viewed by clicking *View* on the tool bar and selecting *View Memory*. The desired starting address is entered in the view memory window that opens.