Lab 09 - LEDs

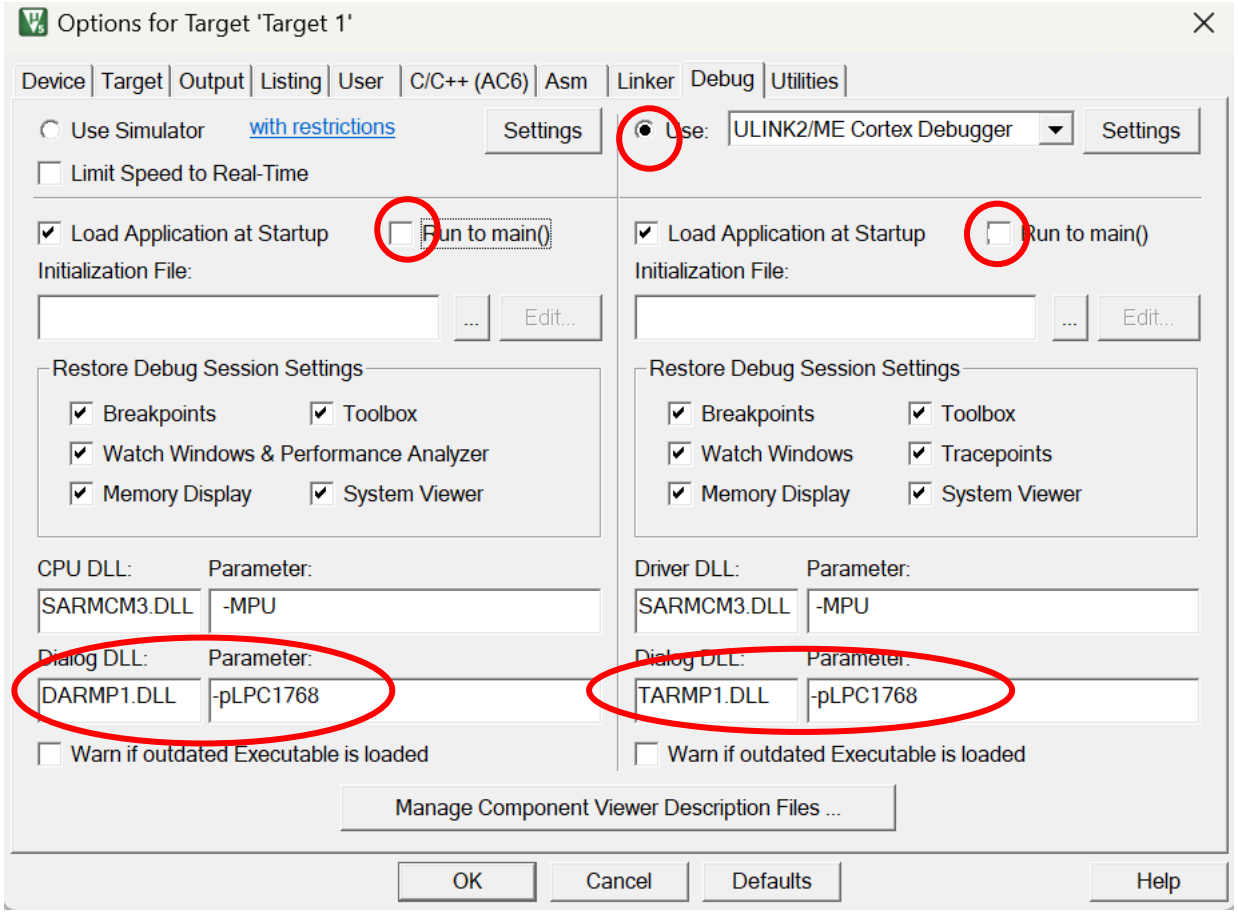
R. Ferrero, M. Russo

Politecnico di Torino

Dipartimento di Automatica e Informatica (DAUIN)

Torino - Italy

Setup


Options for Target 'Target 1'
Device
Target
Output
Listing
User
C/C++ (AC6)
Asm
Linker
Debug
Utilities
Use Simulator
with restrictions
Settings
Use:
ULINK2/ME Cortex Debugger
Settings
Limit Speed to Real-Time
Load Application at Startup
Run to main()
Initialization File:
Edit...
Restore Debug Session Settings
Breakpoints
Toolbox
Watch Windows & Performance Analyzer
Memory Display
System Viewer
Watch Windows
Tracepoints
CPU DLL:
Parameter:
SARMCM3.DLL
-MPU
Driver DLL:
Dialog DLL:
DARMP1.DLL
-pLPC1768
TARMP1.DLL
Warn if outdated Executable is loaded
Manage Component Viewer Description Files ...
OK
Cancel
Defaults
Help

Exercise 1

- Add to led.h file the prototype:
  `void led4and11_On(void);`
- Add to ‘led’ group the file funct_led.c
- Implement in funct_led.c the function `led4and11_On(void)`, powering on the LEDs 4 and 11 acting on the FIOSET register.
- Note: the state (on/off) of the other LEDs must not be modified.
- Test the function calling it from the main.

Exercise 2

- Add to led.h file the prototype:
  `void led4_Off(void);`
- Implement in funct_led.c the function `led4_Off(void)`, switching off LED 4 acting on FIOCLR register.
- Note: the state (on/off) of the other LEDs must not be modified.
- Test the function calling it from the main.

Exercise 3

- Add to led.h file the prototype:
  `void ledEvenOn_OddOff(void);`
- Implement in funct_led.c the function `ledEvenOn_OddOff(void)`, powering on the LEDs with even index number and powering off odd ones, acting on FIOPIN register.
- Test the function calling it from the main.

Exercise 4

- Add to led.h file the prototype:
  `void LED_On(unsigned int num);`
- Implement in funct_led.c the function `void LED_On(unsigned int num)` powering on the LED corresponding to the parameter passed:
  - num = 0 -> LED 4
  - num = 1 -> LED 5
  - num = 7 -> LED 11
- Test the function calling it from the main.

Exercise 5

- Add to led.h file the prototype:
  `void LED_Off(unsigned int num);`
- Implement in funct_led.c the function `void LED_Off(unsigned int num)` powering off the LED corresponding to the parameter passed: num = 0 -> LED 4
  - num = 1 -> LED 5
  - num = 7 -> LED 11
- Test the function calling it from the main.

What LED is on?

- To know which LED is on you can:
  - Read content of `LPC_GPIO2->FIOPIN`
  - Read content of `LPC_GPIO2->FIOSET`
  - *define* a global variable in funct_led.c:

```
unsigned int led_value;
```

`led_value` stores the on LED.

In the other files you can access the variable *declaring*:

```
extern unsigned int led_value;
```