Embedded Systems

2. Software Development

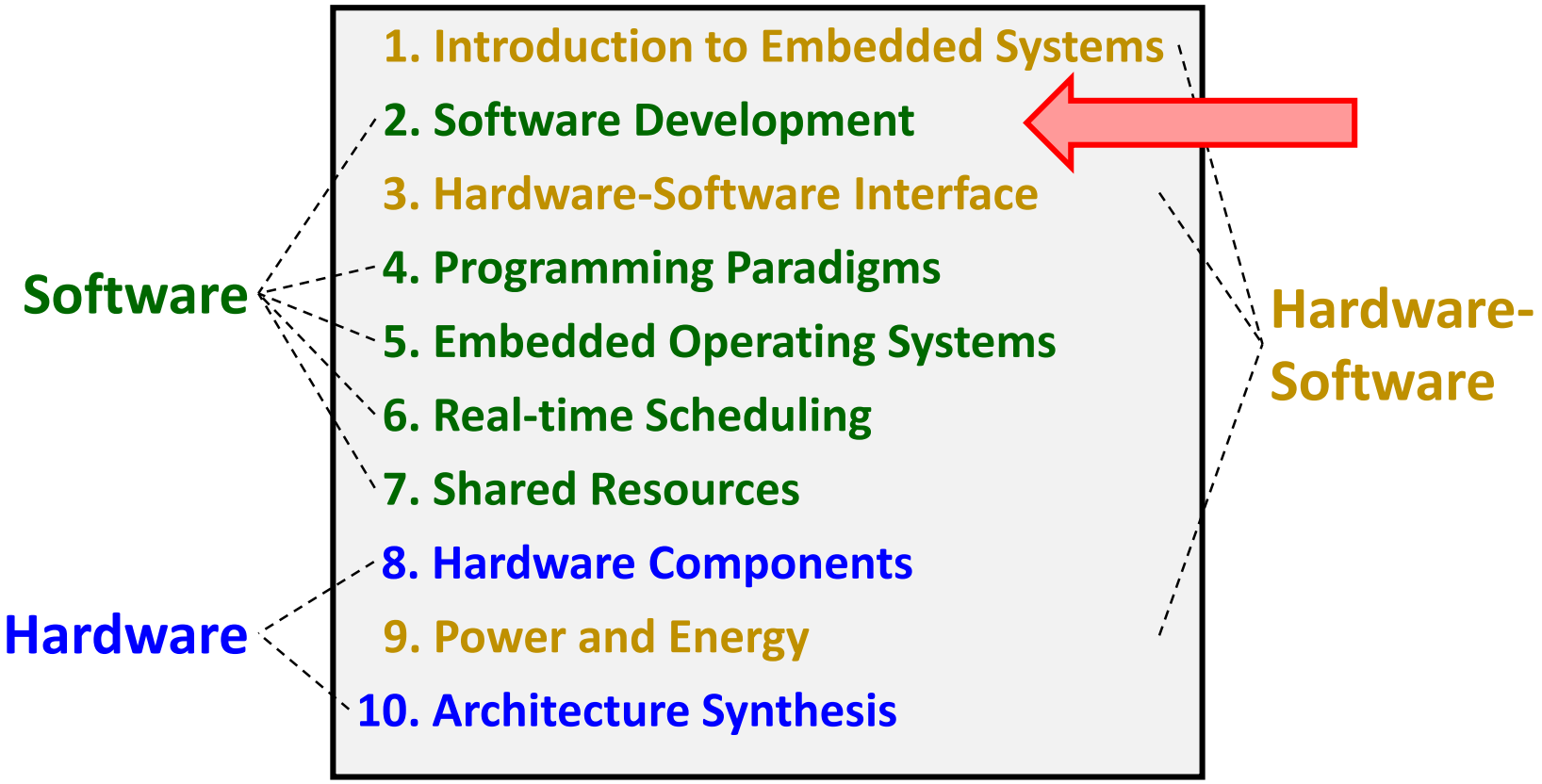
© Lothar Thiele

Computer Engineering and Networks Laboratory

ETH

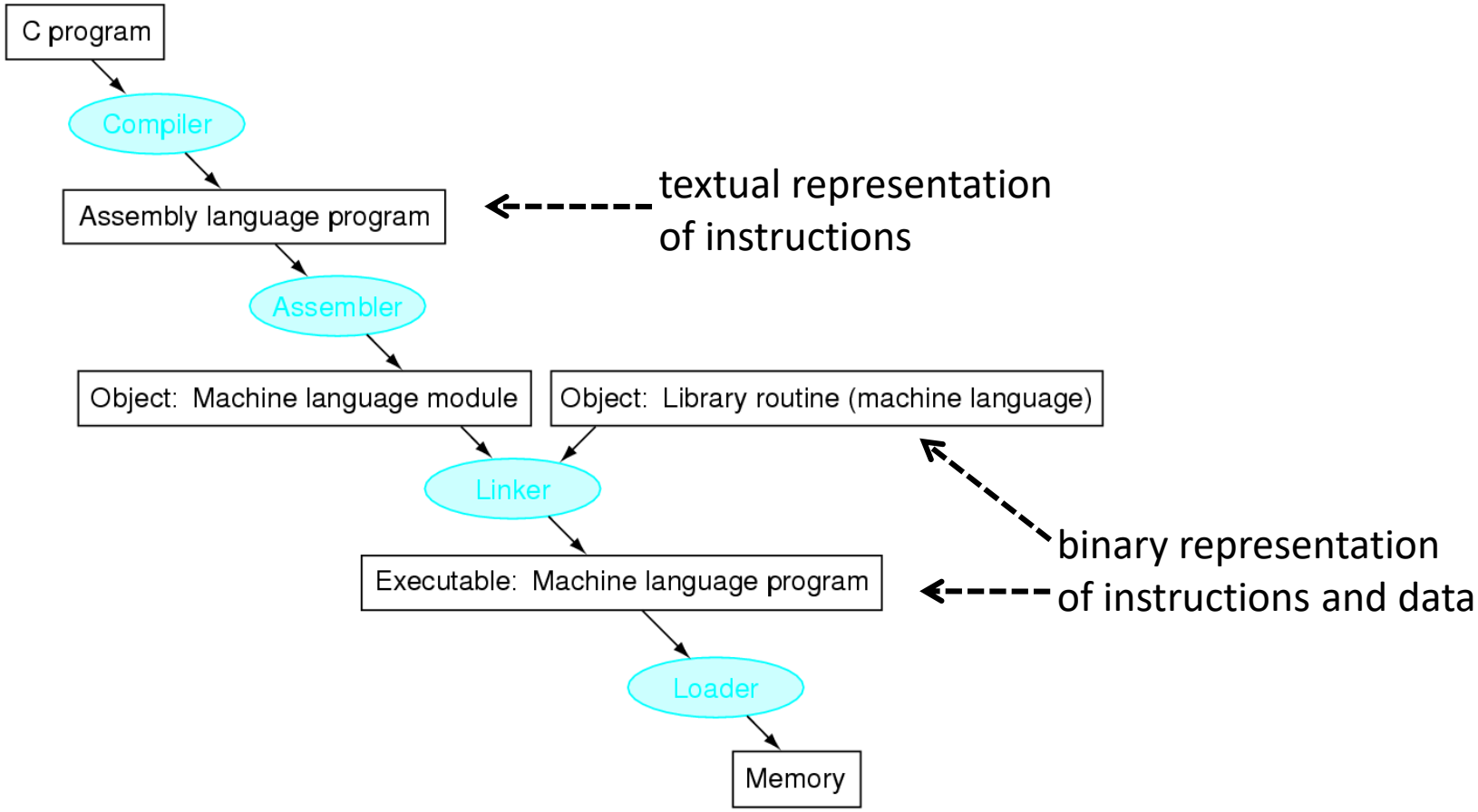
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Where we are …

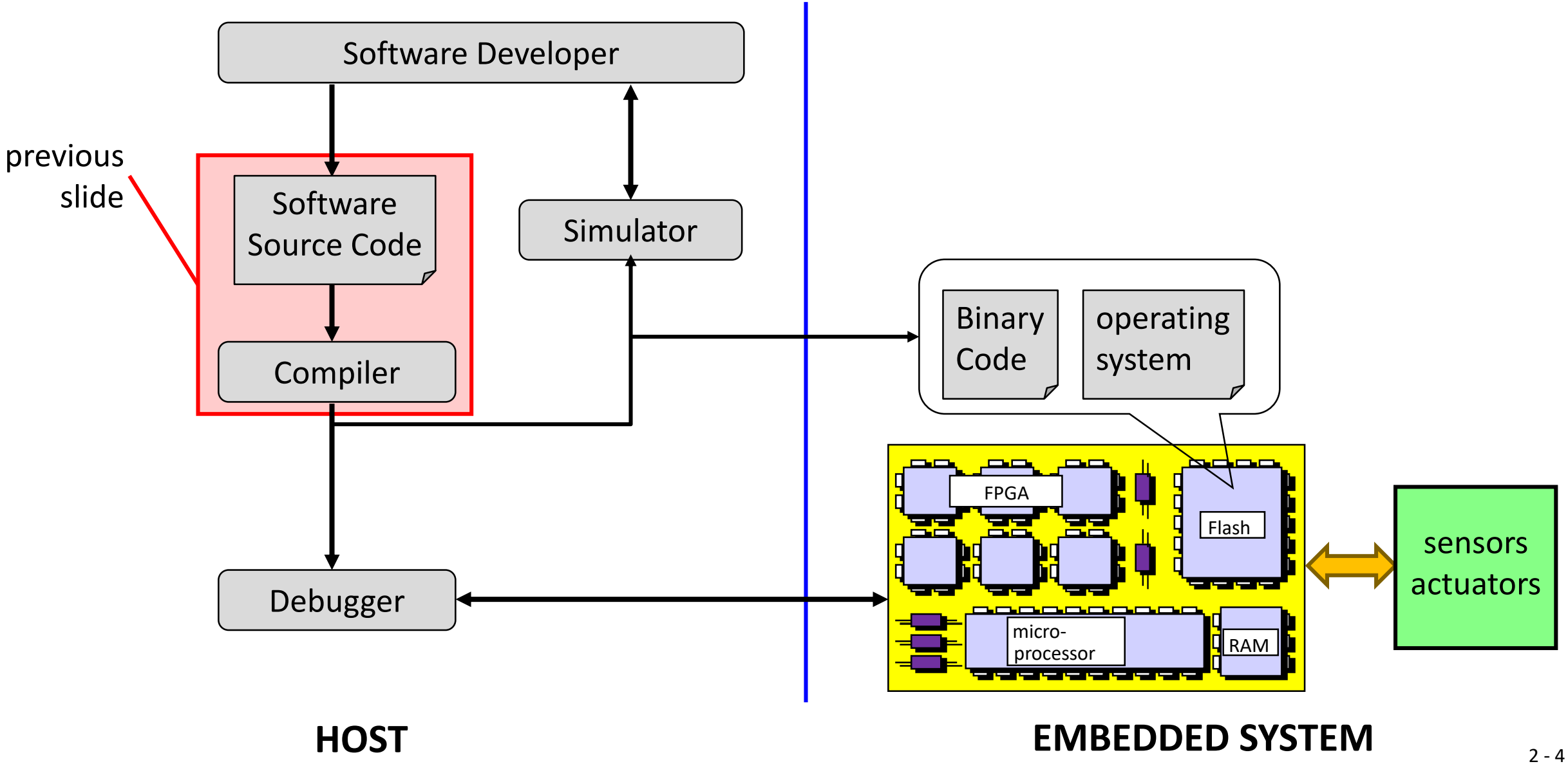

1. Introduction to Embedded Systems
2. Software Development
3. Hardware-Software Interface
4. Programming Paradigms
5. Embedded Operating Systems
6. Real-time Scheduling
7. Shared Resources
8. Hardware Components
9. Power and Energy
10. Architecture Synthesis
Software
Hardware
Hardware-
Software

Remember: Computer Engineering I

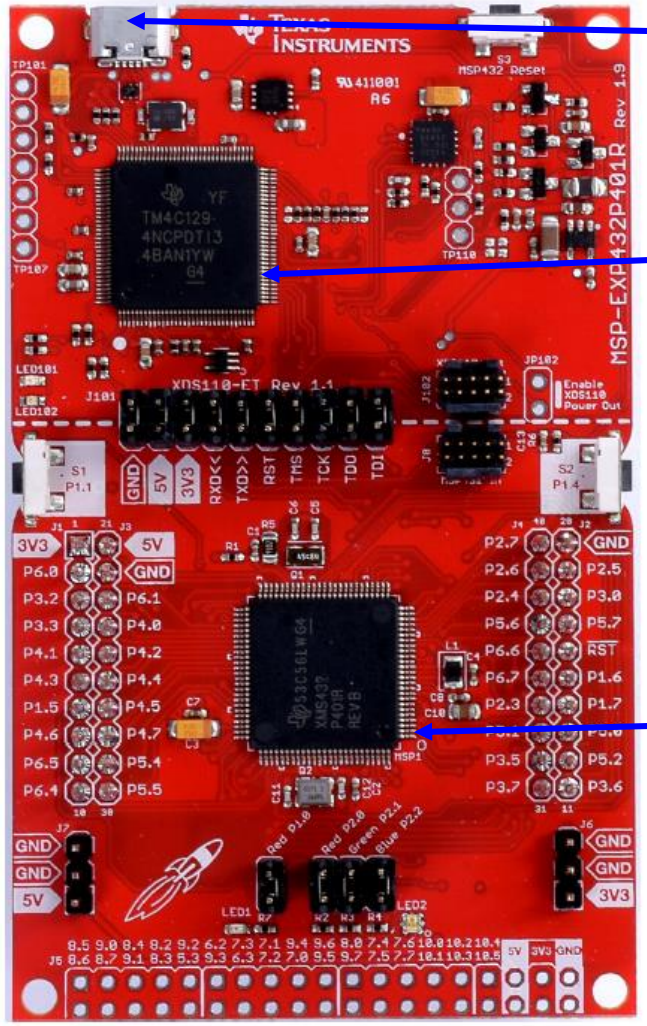
Compilation of a C program to machine language program:

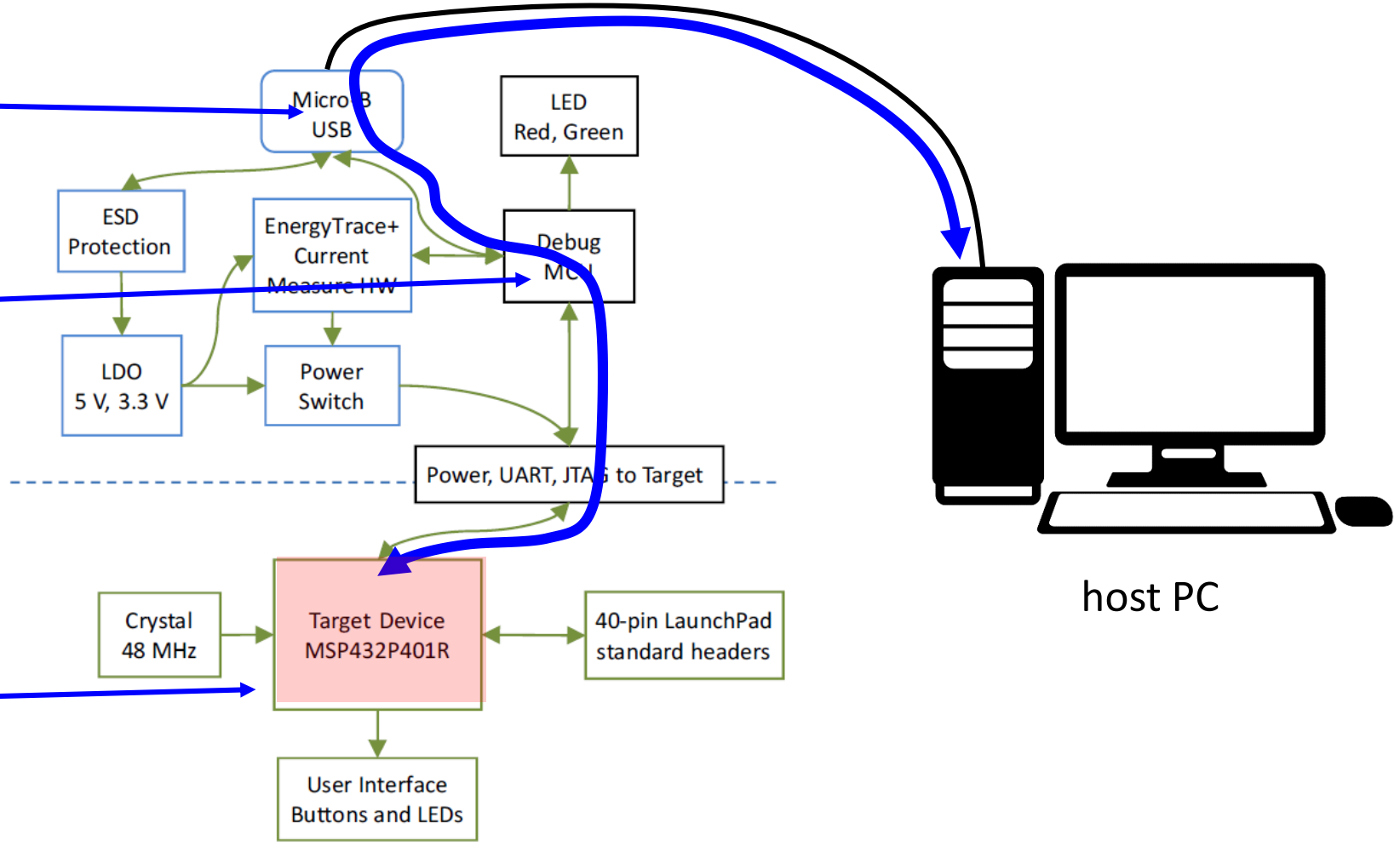


Embedded Software Development



Software Development with MSP432 (ES-Lab)

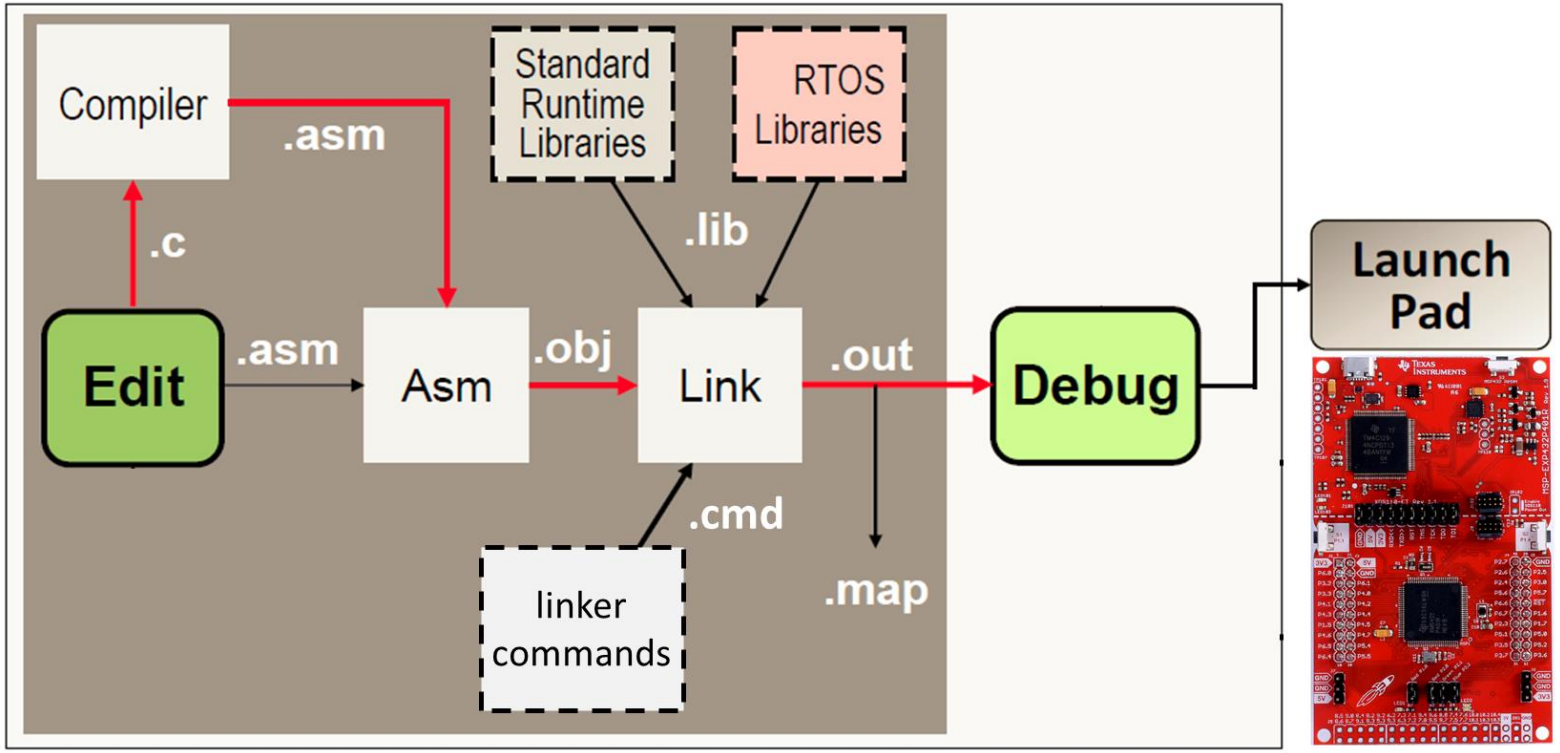




Software Development (ES-Lab)

Software development is nowadays usually done with the support of an IDE (Integrated Debugger and Editor / Integrated Development Environment)

- edit and build the code
- debug and validate



Software Development (ES-Lab)

assembly code

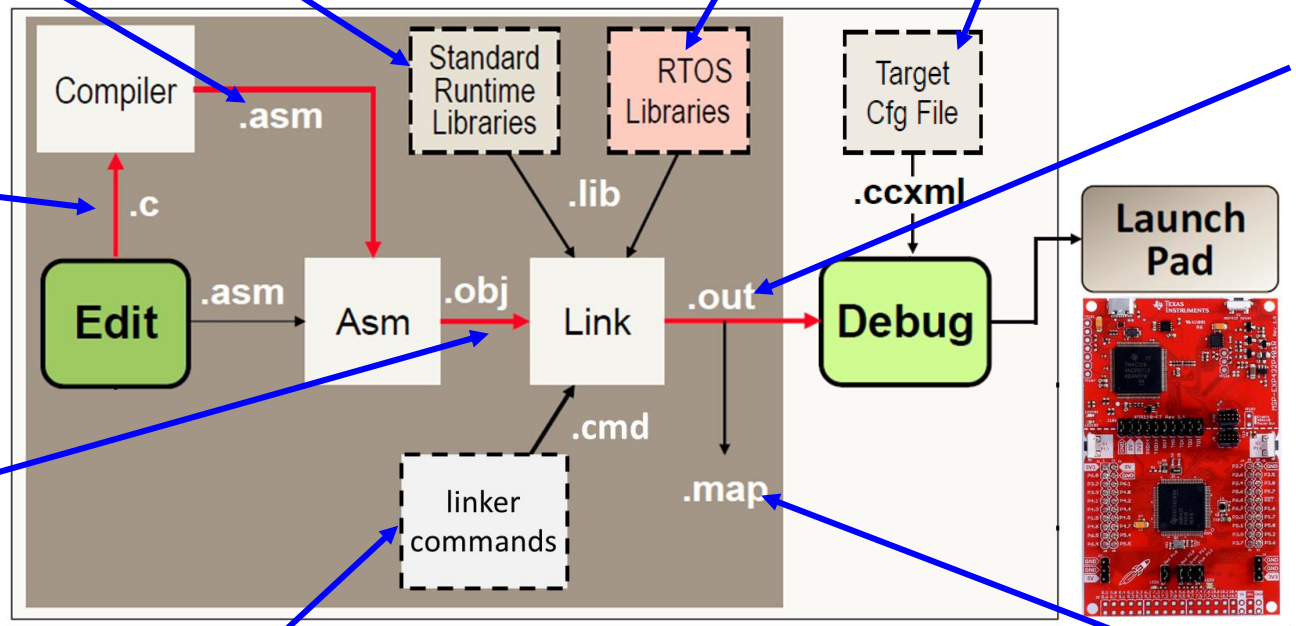
object libraries that are referenced in the code

object libraries that contain the operating system (if any)

target configuration file specifies the connection to the target (e.g. USB) and the target device

source code file in C

the executable output file that is loaded into flash memory on the processor



relocatable object file

Linker command file that tells the linker how to allocate memory and to stitch the object files and libraries together.

report created by the linker describing where the program and data sections are located in memory.

Software Development (ES-Lab)

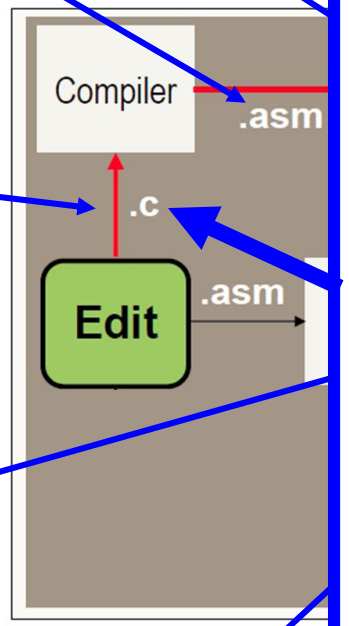
assembly code

object libraries that are refere in the code

source code file in C

relocatable object file

Linker command file tha how to allocate memory the object files and libra




```
...

/*
 * Main function
 */
int main(void)
{
    /* Halting WDT and disabling master interrupts */
    MAP_WDT_A_holdTimer();
    MAP_Interrupt_disableMaster();

    /* Seed the pseudo random num generator */
    srand(TLV->RANDOM_NUM_1);

    /* Set the core voltage level to VCORE1 */
    MAP_PCM_setCoreVoltageLevel(PCM_VCORE1);

    /* Set 2 flash wait states for Flash bank 0 and 1*/
    MAP_FlashCtl_setWaitState(FLASH_BANK0, 2);
    MAP_FlashCtl_setWaitState(FLASH_BANK1, 2);

    /* Default SysTick period for all 4 color states = 0.5s */
    periods[0] = 1500000;
    periods[1] = 1500000;
    periods[2] = 1500000;
    periods[3] = 1500000;

...
```

to the target device

e output file into flash he processor

g

s

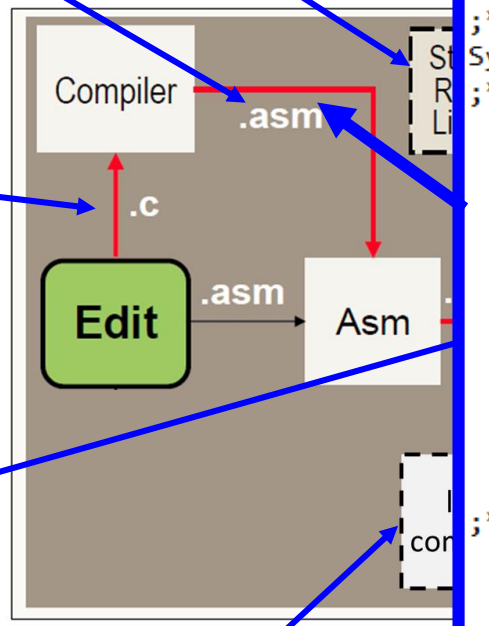
Software Development

assembly code

object libraries that are referenced in the code

source code file in C

relocatable object file



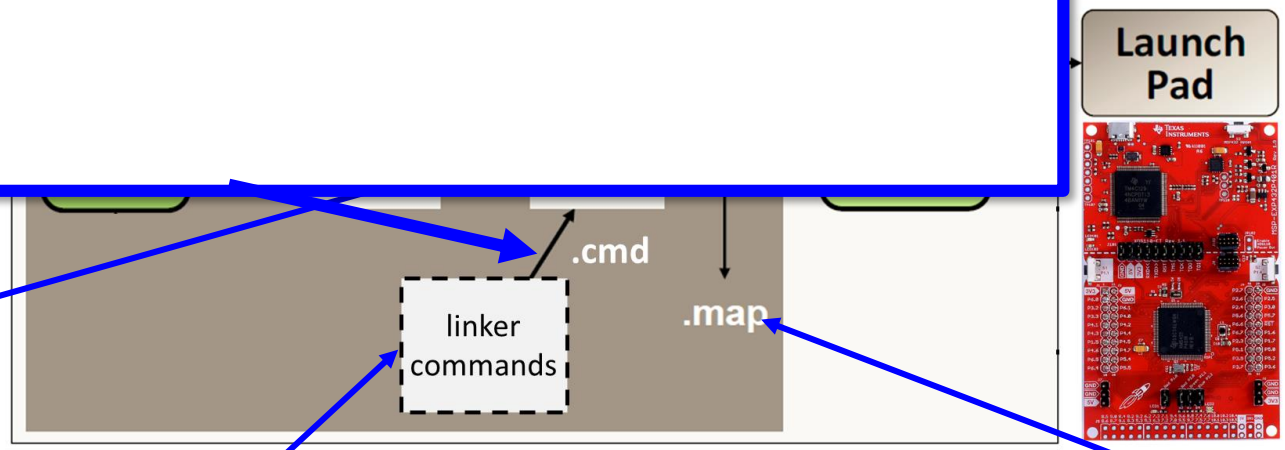
Linker command file that tells how to allocate memory and t the object files and libraries to

```
...

;*****************************************************************************
;* FUNCTION NAME: SysTick_Handler                                            *
;*                                                                           *
;*   Regs Modified     : A1,A2,A3,A4,V9,SP,LR,SR,D0,D0_hi,D1,D1_hi,D2,D2_hi, *
;*                       D3,D3_hi,D4,D4_hi,D5,D5_hi,D6,D6_hi,D7,D7_hi,       *
;*                       FPEXC,FPSCR                                         *
;*   Regs Used         : A1,A2,A3,A4,V9,SP,LR,SR,D0,D0_hi,D1,D1_hi,D2,D2_hi, *
;*                       D3,D3_hi,D4,D4_hi,D5,D5_hi,D6,D6_hi,D7,D7_hi,       *
;*                       FPEXC,FPSCR                                         *
;*   Local Frame Size  : 0 Args + 0 Auto + 4 Save = 4 byte                   *
;*****************************************************************************
SysTick_Handler:
;* ---------------------------------------------------------------------------*
        .dwcfi  cfa_offset, 0
        PUSH      {A4, LR}              ; [DPU_3_PIPE]
        .dwcfi  cfa_offset, 8
        .dwcfi  save_reg_to_mem, 14, -4
        .dwcfi  save_reg_to_mem, 3, -8
        .dwpsn  file "../main.c",line 374,column 5,is_stmt,isa 1
        LDR       A1, $C$CON64          ; [DPU_3_PIPE] |374|
        LDR       A1, [A1, #0]          ; [DPU_3_PIPE] |374|
        CMP       A1, #1                ; [DPU_3_PIPE] |374|
        BNE       ||$C$L20||            ; [DPU_3_PIPE] |374|
        ; BRANCHCC OCCURS {||$C$L20||}   ; [] |374|
;* ---------------------------------------------------------------------------*
        .dwpsn  file "../main.c",line 375,column 9,is_stmt,isa 1
        LDR       A2, $C$CON65          ; [DPU_3_PIPE] |375|
        LDR       A1, [A2, #0]          ; [DPU_3_PIPE] |375|
        ADDS      A1, A1, #1            ; [DPU_3_PIPE] |375|
        STR       A1, [A2, #0]          ; [DPU_3_PIPE] |375|
;* ---------------------------------------------------------------------------*

...
```

```
...

MEMORY
{
    MAIN       (RX) : origin = 0x00000000, length = 0x00040000
    INFO       (RX) : origin = 0x00200000, length = 0x00004000
#ifdef __TI_COMPILER_VERSION__
#if     __TI_COMPILER_VERSION__ >= 15009000
    ALIAS
    {
    SRAM_CODE  (RWX): origin = 0x01000000
    SRAM_DATA  (RW) : origin = 0x20000000
    } length = 0x00010000
#else
    /* Hint: If the user wants to use ram functions, please observe that SRAM_CODE          */
    /* and SRAM_DATA memory areas are overlapping. You need to take measures to separate    */
    /* data from code in RAM. This is only valid for Compiler version earlier than 15.09.0.STS.*/
    SRAM_CODE  (RWX): origin = 0x01000000, length = 0x00010000
    SRAM_DATA  (RW) : origin = 0x20000000, length = 0x00010000
#endif
#endif
}

...
```

target configuration file specifies the connection to the target (e.g. USB) and the target device

the executable output file that is loaded into flash memory on the processor



relocatable object file

Linker command file that tells the linker how to allocate memory and to stitch the object files and libraries together.

report created by the linker describing where the program and data sections are located in memory.

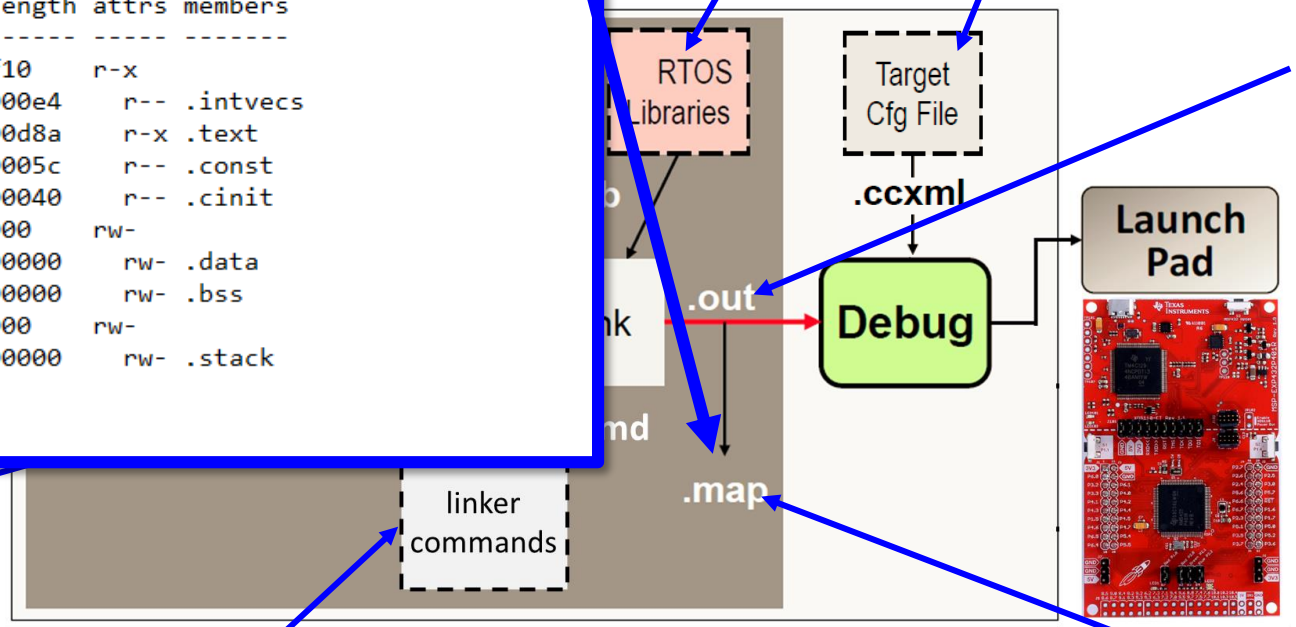
Lab)

```
...

MEMORY CONFIGURATION

         name             origin    length      used     unused   attr    fill
----------------------  --------  ---------  --------  --------  ----  --------
  MAIN                  00000000   00040000  00000f0a  0003f0f6  R  X
  INFO                  00200000   00004000  00000000  00004000  R  X
  SRAM_CODE             01000000   00010000  00000268  0000fd98  RW X
  SRAM_DATA             20000000   00010000  00000268  0000fd98  RW


SEGMENT ALLOCATION MAP

run origin  load origin   length   init length attrs members
----------  -----------  ---------- ----------- ----- -------
00000000    00000000     00000f10   00000f10    r-x
  00000000    00000000     000000e4   000000e4    r-- .intvecs
  000000e4    000000e4     00000d8a   00000d8a    r-x .text
  00000e70    00000e70     0000005c   0000005c    r-- .const
  00000ed0    00000ed0     00000040   00000040    r-- .cinit
20000000    20000000     00000068   00000000    rw-
  20000000    20000000     00000050   00000000    rw- .data
  20000050    20000050     00000018   00000000    rw- .bss
2000fe00    2000fe00     00000200   00000000    rw-
  2000fe00    2000fe00     00000200   00000000    rw- .stack

...
```

ct libraries that
tain the operating
em (if any)

target configuration file specifies the connection to the target (e.g. USB) and the target device

the executable output file that is loaded into flash memory on the processor



relocatable object file

Linker command file that tells the linker how to allocate memory and to stitch the object files and libraries together.

report created by the linker describing where the program and data sections are located in memory.

Software Development (ES-Lab)

assembly code

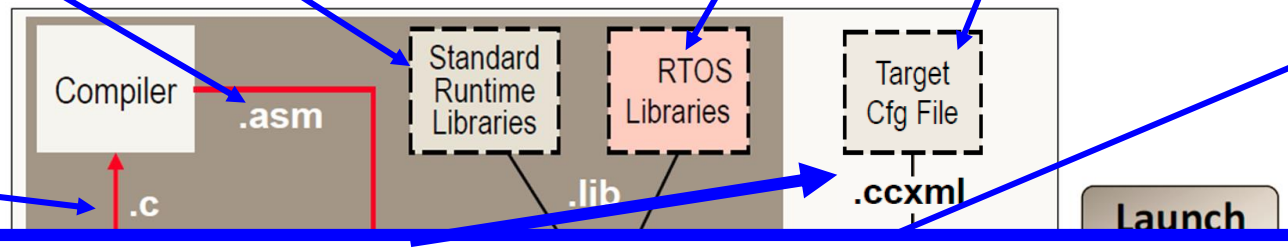
object libraries that are referenced in the code

object libraries that contain the operating system (if any)

target configuration file specifies the connection to the target (e.g. USB) and the target device

source code file in C

the executable output file that is loaded into flash memory on the processor




```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<configurations XML_version="1.2" id="configurations_0">
    <configuration XML_version="1.2" id="configuration_0">
        <instance XML_version="1.2" desc="Texas Instruments XDS110 USB Debug Probe" href="connections/ ...
        <connection XML_version="1.2" id="Texas Instruments XDS110 USB Debug Probe">
            <instance XML_version="1.2" href="drivers/tixds510cs_dap.xml" id="drivers" xml= ...
            <instance XML_version="1.2" href="drivers/tixds510cortexM.xml" id="drivers" xml= ...
            <property Type="choicelist" Value="2" id="SWD Mode Settings">
                <choice Name="SWD Mode - Aux COM port is target TDO pin" value="nothing"/>
            </property>
            <platform XML_version="1.2" id="platform_0">
                <instance XML_version="1.2" desc="MSP432P401R" href="devices/msp432p401r.xml" id= ...
            </platform>
        </connection>
    </configuration>
</configurations>
```

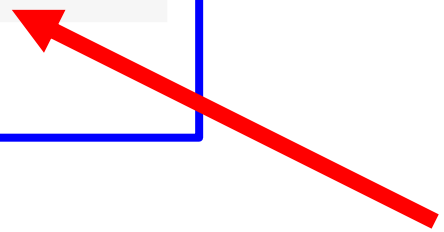
describing a sections

Much more in the ES-PreLab …

- The Pre-lab is intended for students with missing background in software development in C and working with an integrated development environment.

Timetable

Date	Lecture	Exercice	Lab
27.09.2021	1. Introduction 2. Software Development		
29.09./01.10.2021			0. Prelab [MM]
04.10.2021	3. Hardware-Software In-terface		



Much more in the ES-PreLab …

- The Pre-lab is intended for students with missing background in software development in C and working with an integrated development environment.

Embedded Systems 1.0.1 – Filling the gaps

Goals of this Lab

The goal of this lab session is to give a quick crash-course on all necessary background for the following labs. You are expected to have some basic knowledge about programming, but programming an embedded systems is slightly different than Python, Java, or Matlab.

Here are the main topics the pre-lab covers:

- Definitions and keywords – Know what you are talk about
- C programming – Review of the fundamentals
- Embedded systems programming – Specific types and basic operations
- Schematics – Find your way around a processor schematics
- Demo application – If you can make it, you're good to go!