Embedded Systems - Notes Week 1

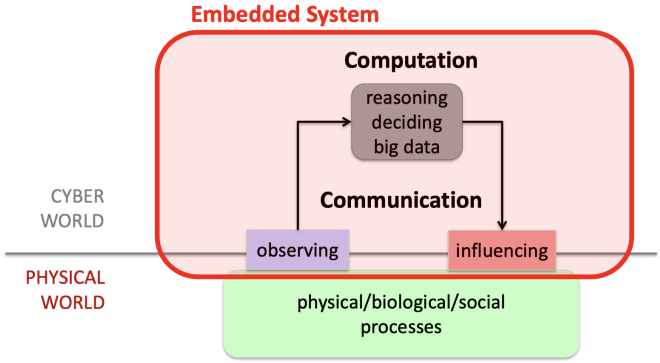
Ruben Schenk, ruben.schenk@inf.ethz.ch

January 13, 2022

Chapter 1: Introduction

1.1 Impact

Embedded systems are information processing systems embedded into a larger product. Usually they use feedback to influence the dynamics of the physical world by taking smart decisions in the cyber world.



1.2 Facts

Embedded systems are often *reactive:* reactive systems must react to stimuli from the system environment.

> "A reactive system is one which is in continual interaction with its environment and executes at a pace determined by that environment" - Bergé, 1995

ES often must meet *real-time constraints:* For hard real-time systems, right answers arriving too late are wrong. All other time-constraints are called soft. A *guaranteed system response* has to be explained without statistical arguments.

> "A real-time constraint is called hard, if not meeting that constraint could result in a catastrophe" - Kopetz, 1997

It is essential to *predict* how a cyber-physical system (CPS) is going to behave under any circumstances before it is deployed. CPS must *operate dependably,* safely, securely, efficiently and in real-time.

ES must be *efficient:*

- Energy efficient
- Code-size and data memory efficient
- Run-time efficient
- Weight efficient
- Cost efficient

ES are often *specialized* towards a certain application or application domain: Knowledge about the expected behavior and the system environment at design time is exploited to *minimize resource usage* and to *maximize predictability and reliability.*

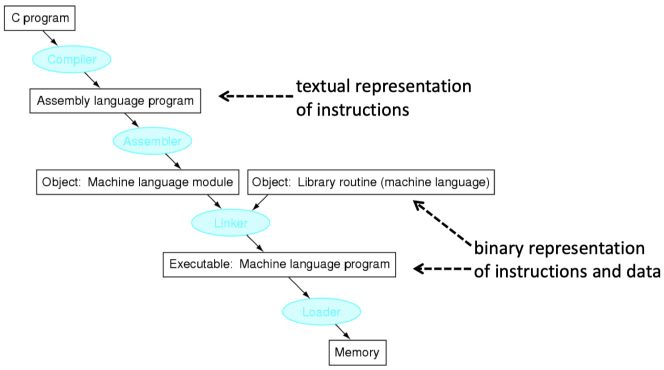
1.3 Trends

Some trends of embedded systems:

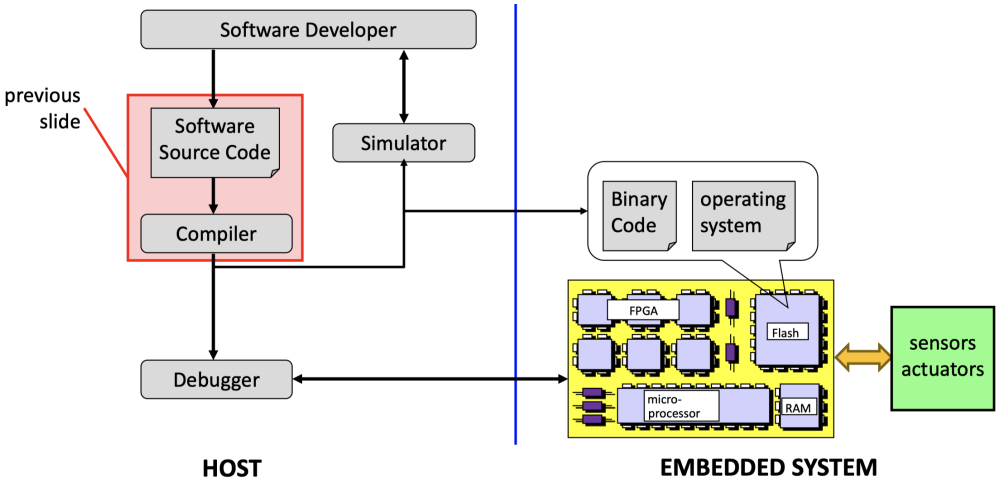
- ES communicating with each other, with servers or with the cloud. Communication is increasingly.
- Higher degree of integration on a single chip or integrated components:
  - Memory + processor + I/O units + communication
  - Use of networks-on-chip for communication between units
  - Use of homogeneous or heterogeneous multiprocessor system on a chip (MPSoC)
- Low power and energy constraints (especially for portable or unattended devices) are increasingly important, as well as temperature constraints
- There is increasing interest in energy harvesting to achieve long term autonomous operation

Chapter 2: Software Development

Reminder: Compilation of a C program to machine language works as follows:



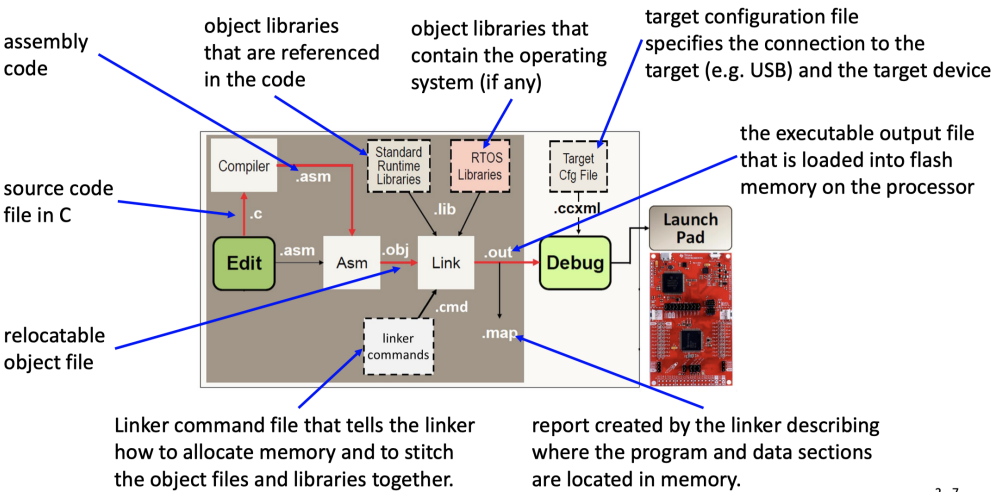
The main chain-of-events for **embedded software developments** is given by the following diagram:



Software development is nowadays usually done with the support of an IDE:

- Edit and build the code
- Debug and validate the code

A better overview on how this works with embedded systems is given below:


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
the executable output file that is loaded into flash memory on the processor
source code file in C
relocatable object file
Compiler
.asm
Standard Runtime Libraries
RTOS Libraries
Target Cfg File
.c
.lib
.ccxml
Launch Pad
Edit
.asm
Asm
.obj
Link
.out
Debug
.cmd
.map
linker commands
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.