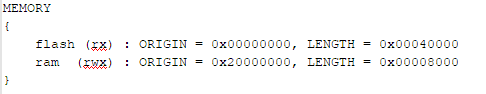
Praktikum 3 – Energia

Aufgabe 2

* Bei der Kompilierung werden zunächst alle Libraries die benutzt werden erkannt, anschließend werden die Funktionsprototypen generiert. Danach wird das Sketch die Libraries und der Kern kompiliert. Zum Schluss wird alles zusammengelinkt
* Nach dem Kompilieren wird der Debugger konfiguriert, dabei wird die Register Database und die CPU (Cortex M4) initialisiert. Zusätzlich werden noch die Startup Skripte der CPU ausgeführt
* Die Memory Map wird initialisiert, das Programm geladen und das Flash-Werkzeug übertragt das Kompilat auf das Target-System, in unserem Fall das LaunchPad
* Die Toolchain besteht aus dem Cross-Compiler und dem Linker, dem Debugger, dem Flash-Werkzeug, dem Board circuit debug interface und den Build-Management-Tools

Aufgabe 3

* C:\Users\Micha2\AppData\Local\Energia15\packages\energia\hardware\tivac\1.0.3\cores\tivac -> Pfad des Linker-Skripts lm4fcpp\_blizzard.ld
* Ein Linker-Skript macht aus einer .o-Datei (Maschinensprache) eine .elf-Datei (Exe-/Linkable Format)



* name [(attr)] : ORIGIN = origin, LENGTH = len
* der Name umfasst die Region des Speichers, in dem Fall den Flash und den RAM-Speicher
* die Attribute wären in dem Fall für den Flash-Speicher r und x, was heißt dass der Bereich lesbar (r) und ausführbar ist (x), die RAM Sektion ist zusätzlich noch überschreibbar
* ORIGIN steht für die Start-Adresse des Speichers und die LENGTH gibt die Größe in Bytes an

Aufgabe 5

* E:\energia-1.6.10E18\hardware\energia\msp430\cores\msp430 -> Pfad für Energia.h und main.cpp
* In der main.cpp werden zunächst die Methoden init und setup aufgerufen, anschließend wird eine for-Schleife aufgerufen die äquivalent zu while (true) ist (for (;;))
* Die for-Schleife enthält dann die loop-Methode und noch eine if-Verzweigung die die Methode serialEventRun aufruft weil serialEventRun true ist
* Setup und loop reichen aus, da init eine leere Methode und so keine Rolle spielt falls sie nicht implementiert wird