

Linux board bringup

Előző előadás emlékeztető

Linux alapok

– Alapfogalmak:

- Operációs rendszer rétegei
- Kernel és user space
- Device tree

– Driverfejlesztés alapok

- Kernel modul
- Hardverelérés
 - Platform eszközvezérlők
- Egyszerű karakteres eszközök

Nem volt szó a Linux „élesztéséről”.

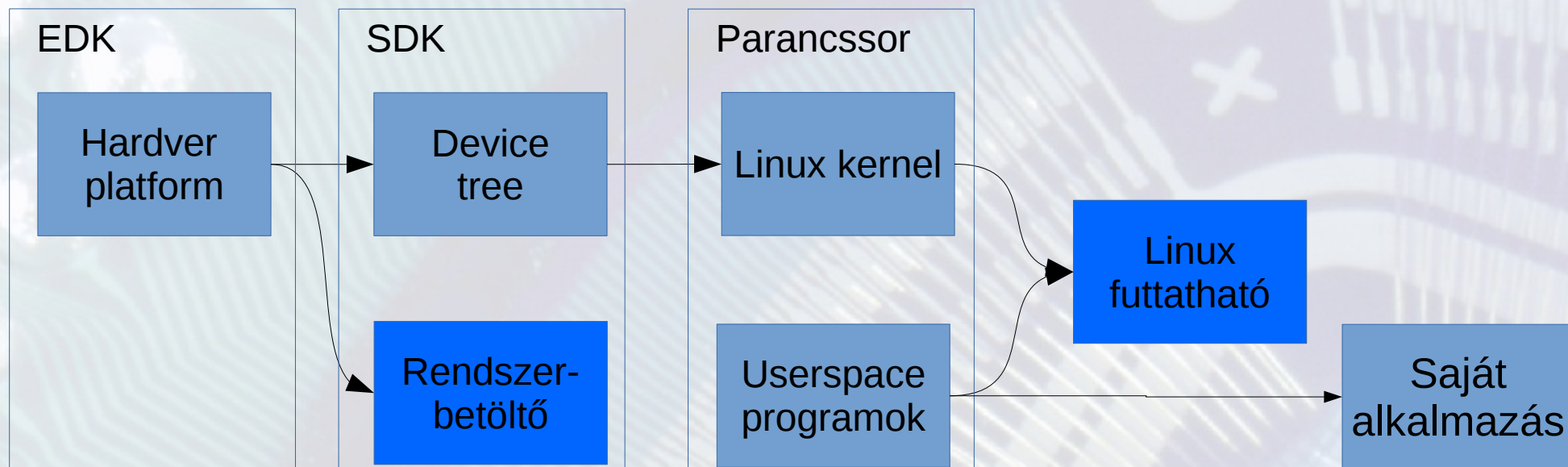
Mai előadás

Cél

- Egy olyan, Digilent Atlys panelen futó hardverkörnyezet létrehozása, mely rendelkezik
 - soros porttal
 - hálózat támogatással
 - GPIO portokkal
 - SD kártya olvasóval
- és ezeken túl Linux operációs rendszert futtat.
 - Rendszerbetöltővel nem fogunk foglalkozni.
- Ráadásként szeretnénk rá könnyen szoftvert fejleszteni.

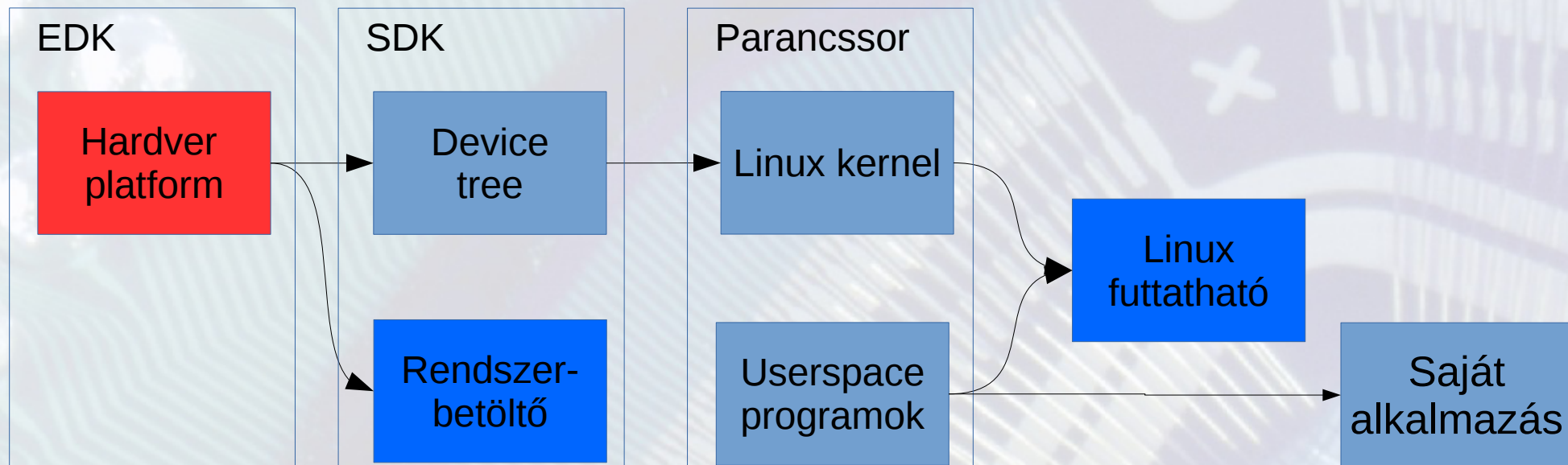
MicroBlaze alapú Linux rendszer

Linux rendszer létrehozásának lépései



MicroBlaze alapú Linux rendszer

Linux rendszer létrehozásának lépései



Hardver platform létrehozása

Új projekt

- Ugyanúgy létrehozhatjuk a BSB wizard segítségével a projektünket, amire oda kell figyelni:
 - Legyen timer a rendszerben
 - Legyen külső (DDR vagy SD RAM) memória
 - Cache nem árt
 - Blokk RAM memória nem lényeges a Linux szempontjából, a bootloadernek, ha használunk, viszont kell!
 - Minden egység megszakításos legyen
 - A MicroBlaze Debug Module kivételével
 - Hogy lássuk a boot folyamatot, soros portra szükségünk van

Hardver platform létrehozása

MicroBlaze beállítások

- A MicroBlaze beállításoknál „Linux with MMU” alapbeállítás
 - Ez megdobja az erőforrásigényt!
 - „Megjegyzendő” adatok
 - Memória fizikai címe
 - Bekapcsolt processzortulajdonságok (osztó, barrel shifter, stb.)
- ezeket később használni kell

Hardver platform létrehozása

SD kártya

- Az SD kártyát SPI módban fogjuk használni, ehhez SPI vezérlőt helyezünk el a projektben
 - A szokásos módon történik a bekötés
 - Fontos paraméter, hogy az SPI órajele mekkora legyen a rendszer órajeléhez képest, ehhez konzultálni kell az SD kártya adatlapjával

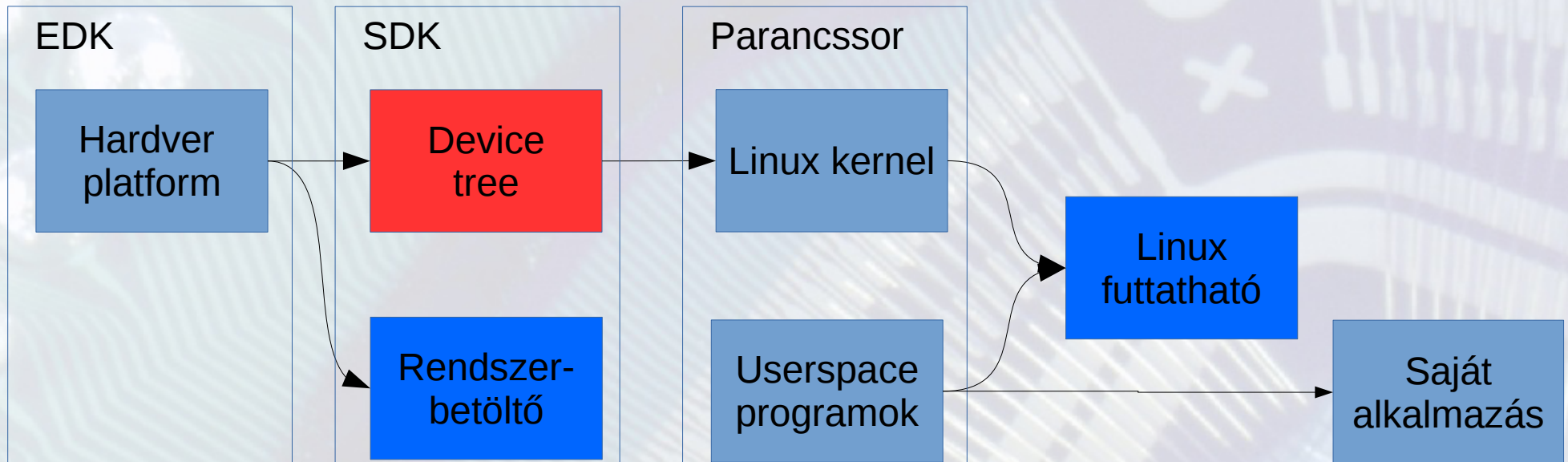
Hardver platform létrehozása

Szintézis

- A következő lépést már SDK-ban hajtjuk végre →
Export to SDK
 - Célszerű először bitfájl nélkül exportálunk, és úgy indítani az SDK-t
 - Mert utána már tudunk SDK-zni, míg szintetizálódik a projekt

MicroBlaze alapú Linux rendszer

Linux rendszer létrehozásának lépései



Az eszközfá generálása

Eszközfá exportált hardver projektből

- A device tree olyan BSP, mint a „standalone” vagy a „xilkernel”, csak az alaptelepítés nem tartalmazza, külön kell letölteni, git verziókezelővel
 - *git clone git://github.com/Xilinx/device-tree.git bsp/device-tree_v0_00_x*
- Hozzá kell majd adni a Xilinx SDK elérési útvonalához
- Automatikusan generál a leírásból egy device tree-t.
 - Ami utólag módosítható
 - Maga a generáló program TCL programnyelven íródott, adott esetben kézzel továbbfejleszthető

Az eszközfá generálása

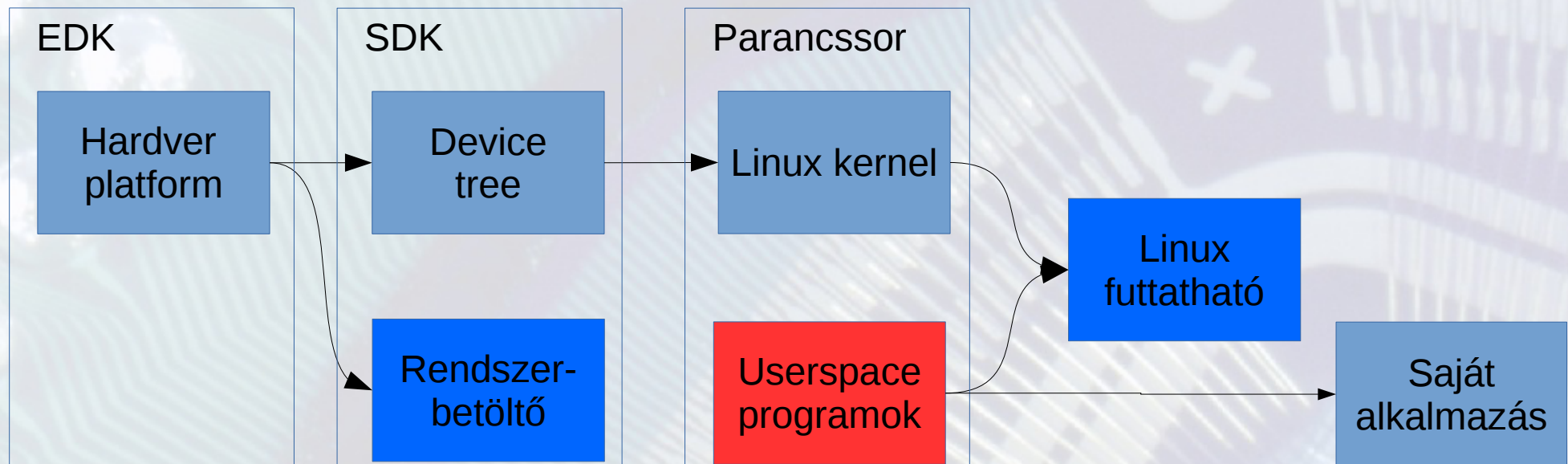
Paraméterek

- Boot console: hova írja ki a kernel a boot üzeneteket
 - Xilinx Uartlite, Xilinx MDM vagy Xilinx Uart16550
- Build project hatására az eszközfá *xilinx.dts* néven előáll
 - Ebbe az SPI-ra illesztett SD-kártya miatt kézzel bele kell nyúlni és az SPI vezérlő csomópontjába befűzni a következőt:

```
mmc_spi@0{  
    compatible = "mmc-spi-slot";  
    spi-max-frequency = < 2000000 >;  
    reg = < 0x00000000 >;  
    voltage-ranges = < 3300 3300 >;  
};
```

MicroBlaze alapú Linux rendszer

Linux rendszer létrehozásának lépései



Root fájlrendszer fordítása

A root fájlrendszer elkészítésének általános szempontjai

- Mekkora méretű lesz a kész fájlrendszer?
 - Sokszor nincs adathordozó (pl. SD kártya) a hardverben, így a fő memóriába töltődik a root lemez (ramdisk)
- Milyen alkalmazásokat tartalmazzon?
 - Sokféle alkalmazás – nagy méret
 - Kevés alkalmazás – kevesebb szolgáltatás
- Milyen formátumban készüljön el?
 - Pl. ISO CD-lemez kép, vagy ext2 (szabványos Linux fájlrendszer)
- Hogyan tudunk majd a kész userspace alkalmazásokhoz saját alkalmazást hozzátenni?

Root fájlrendszer fordítása

A root fájlrendszer elkészítésének szoftveres szempontjai

– Kiemelkedően fontos a C programkönyvtár

- Hiszen ez lesz, ami a userspace alkalmazás és a kernel közötti interfészt kezeli
- Sokféle megvalósítása létezik, szolgáltatásban és optimalizáltságban különbözőek
 - glibc – legösszetettebb, asztali számítógépeken
 - eglibc – nagyobb beágyazott rendszereken
 - uclibc – mikrokontrollereken
- A választás sokszor azon múlik, hogy melyik van kéznél :-)

Root fájlrendszer fordítása

A root fájlrendszer elkészítésének szoftveres szempontjai

– Eszközkezelés

- Alapértelmezésben a különböző hardveres eszközökhöz nem jönnek létre a fájlrendszer bejegyzések
 - Kézzel kéne létrehozni
- Létezik viszont olyan szoftver, amely értesül a kerneltől ha egy új eszközt csatlakoztatnak, és akkor létrehozza a fájlrendszerbejegyzéseket
 - devtmpfs – kernel hozza létre a bejegyzéseket, nagyon kis méretű, nem konfigurálható
 - mdev – beágyazott, kis méretű
 - udev – nagyobb méretű, „okosabb”
 - Választási szempont: van-e elég szabad helyünk?

Root fájlrendszer fordítása

A root fájlrendszer elkészítésének szoftveres szempontjai

- *init*: első processz, ami elindul
 - Ez indítja el a userspace szolgáltatásokat (hálózatot konfigurál, stb.)
 - Itt is több választási lehetőség van, a „klasszikus” init a System V (sysv) init
 - A választási szempont itt is lényegében a méret
- Felhasználói segédprogramok
 - *busybox*: az általában használt userspace programok

Root fájlrendszer fordítása

Fordítás módszerei

– Kézzel

- Ember letölti a forráskódokat, beállítja, mi szükséges neki, aztán lefordítja → nehézkes

– Buildroot

- Egy konfigurációs felületen megadjuk, mire van szükségünk, aztán automatikusan letölti a forrást, fordít, installál

– Petalinux

- Xilinx specifikus, kernel és root fájlrendszer automatizált fordítása, viszont kevésbé konfigurálható

– Yocto

- Hasonló a Buildroot-hoz, de sokkal nehezebben tanulható (viszont egyre többen támogatják)

Root fájlrendszer fordítása

Buildroot – <http://buildroot.uclibc.org/>

- Átlátható, menüvezérelt konfiguráció
- Automatikus letöltés és fordítás
 - Automatikusan letölti az architektúrához szükséges fordítóprogramot
 - Több architektúrára, nem csak MicroBlaze-re
 - ! A MicroBlaze támogatás a 2014-es release-ekben hibás, lefordul, de a hardveren futás közben „elszáll” !
- Kimeneti formátum választható:
 - ISO image, ext2, CPIO (standard kernel root fájlrendszer formátum)
- Eclipse támogatás: a lefordított root fájlrendszerhez Eclipse-ből könnyű saját alkalmazást fejleszteni

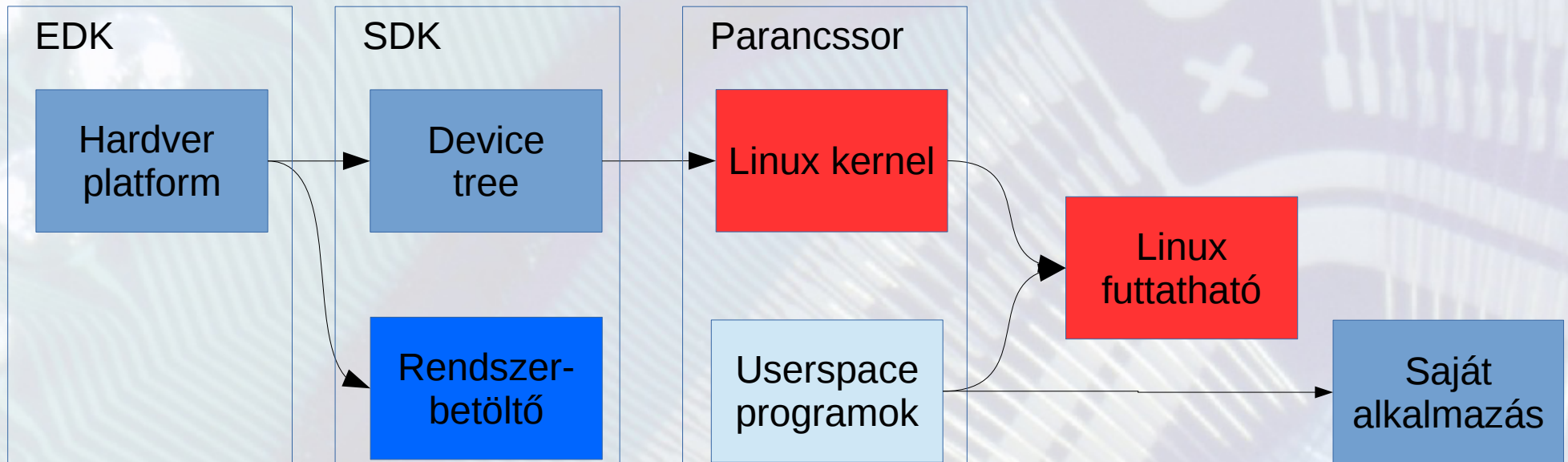
Root fájlrendszer fordítása

A fordítás lépései

- A fordítás alapvetően parancssorból történik :-(
- Konfiguráció: letöltött és kitömörített Buildroot forrásban a *make menuconfig* paranccsal
- Fordítás: konfiguráció után a *make* paranccsal történik.
- A lefordított kimenet az *output/images* könyvtárban jelenik meg

MicroBlaze alapú Linux rendszer

Linux rendszer létrehozásának lépései



Linux kernel fordítása

A Linux kernel fordításának általános szempontjai

- Milyen architektúrára fordítunk?
 - Esetünkben *microblaze*
- Milyen fordítóprogrammal végezzük a fordítást?
- Milyen hardvereket szeretnénk támogatni?
 - Mihez szeretnénk drivert fordítani?
 - Milyen platformra fejlesztünk?
 - Device tree
- Milyen egyéb kernel oldali szolgáltatásokat szeretnénk?

Linux kernel fordítása

A kernel forrása

- A Linux kernel forrása letölthető a www.kernel.org weboldalról
 - A Xilinx IP-k drivereit csak részben tartalmazza (GPIO, UART, intc, ethernet), a Xilinx driverek egy része még nem vált „hivatalossá”
- Xilinx-módosított kernel források verziókezelőzve
 - *git clone git://github.com/Xilinx/linux-xlnx.git*
 - A főbb Xilinx IP-khez (GPIO, UART, DMA, intc, ethernet) teljes körű támogatás

Linux kernel fordítása

Kicsomagolt forráskód főbb alkönyvtárai

- *arch* – alacsony szintű támogatás (megszakításkezelés, boot inicializálás, stb.)
- *arch/microblaze* – a MicroBlaze-specifikus alacsony szintű rutinok
- *arch/microblaze/boot* – a bootolható kimenet ide generálódik
- *arch/microblaze/boot/dts* – ide szükséges helyezni a device tree fájlt, ami a platformot írja le

Linux kernel fordítása

Kicsomagolt forráskód főbb alkönyvtárai

- *Documentation*: egész használható leírások az egyes részekről
- *drivers*: meghajtóprogramok
- *fs*: fájlrendszer kezelő részek
- *net*: hálózati protokollok
- *samples*: néhány példaprogram

Linux kernel fordítása

Előkészítés

- A fordítás előkészítéseként szükséges
 - a device tree
 - Az *arch/microblaze/boot/dts* könyvtárba helyezendő
 - a root fájlrendszer CPIO formátumban
 - Tipikusan a kitömörített kernel forrás könyvtárába helyezhető
 - C fordító, amely az adott processzorhoz fordít
 - Például a Xilinx MicroBlaze fordító

Linux kernel fordítása

Konfiguráció

- A szükséges szolgáltatások és driverek kiválasztása
- A root fájlrendszer kiválasztása
- A platform specifikus tulajdonságok beállítása
 - Memória báziscíme
 - Barrel shifter, osztó, hardveres szorzó, FPU
 - Ezek a tulajdonságok a device tree-ben megtalálhatóak, azokkal kell egyeznie

Linux kernel fordítása

Konfiguráció

- A konfiguráció a Buildroot-hoz hasonlóan menüvezérelt
- *make ARCH=microblaze
CROSS_COMPILE=microblaze-xilinx-linux-gnu-
menuconfig*
- Itt kiválaszthatóak a támogatott tulajdonságok, elmenthető és betölthető a konfiguráció

Linux kernel fordítása

Konfiguráció

- Általában nehéz „nulláról” konfigurálni egy rendszert
- Érdeemes egy „default” konfigurációt módosítani
 - MicroBlaze-nél *mmu_defconfig*
- *make ARCH=microblaze*
CROSS_COMPILE=microblaze-xilinx-linux-gnu-
mmu_defconfig

Linux kernel fordítása

Fordítás

– Fordítani a

make ARCH=microblaze

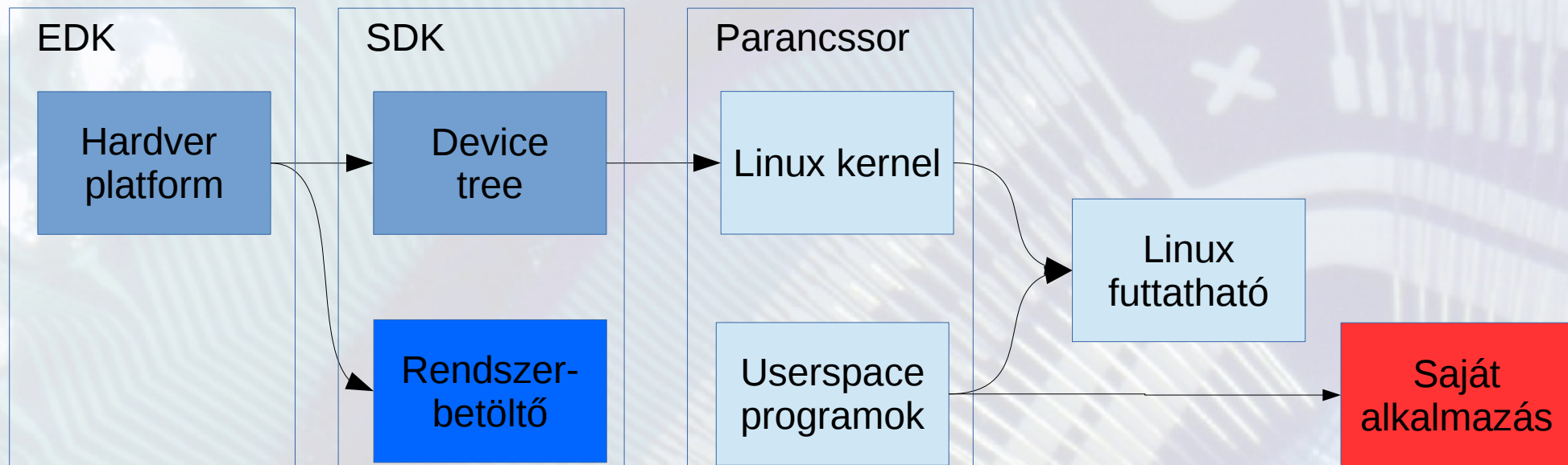
*CROSS_COMPILE=microblaze-xilinx-linux-gnu-
simpleImage.[devicetree neve]*

paranccsal lehet.

– A simpleImage magában hordozza majd a device tree-t és a root fájlrendszert, letöltve a panelra a Linux bebootol

MicroBlaze alapú Linux rendszer

Linux rendszer létrehozásának lépései



Saját alkalmazás fejlesztése

Saját alkalmazás fejlesztése

- Ez az leggyakoribb feladat
- Szükséges követelmények
 - Hol található meg a fordítóprogram?
 - Milyen programkönyvtárak találhatóak meg?
 - Hol érhetőek el ezek?
- Hogyan töltjük fel az alkalmazást?
 - Belefordítva a root fájlrendszerbe
 - Hálózatról
 - SD kártyáról
 - Stb.

Saját alkalmazás fejlesztése

Segédeszközök

- A legtöbb board bringup eszköz Eclipse-integrációval rendelkezik
 - Kiválasztható, hogy melyik platformhoz szeretnék fordítani
- Az Eclipse támogat ún. remote target-eket, tehát hálózaton megtalálható eszközre telepíthető az alkalmazás

Saját alkalmazás fejlesztése

Buildroot Eclipse plugin

- Telepíthető:
<http://buildroot.org/downloads/eclipse/kepler-sr2/>
- Új projekt létrehozásakor kiválasztható a létrehozott Buildroot rendszer
- Innentől kezdve „klasszikus” szoftverfejlesztés
 - Debugger nélkül...