Budapesti Műszaki és Gazdaságtudományi Egyetem
Méréstechnika és Információs Rendszerek Tanszék

Szoftverfejlesztés a Formula Student versenyautóhoz

Tóth János (VHH4K3), I. évf, (MSc) mérnök inf. szakos hallgató
Konzulens: Scherer Balázs, MIT
Szolgáltatásbiztos rendszertervezés szakirány
Önálló laboratórium 2 összefoglaló
2012/13. I. félév

A Formula Student egy nemzetközi konstruktőri versenysorozat, ahol a csapatoknak egy együléses versenyautót kell megtervezniük és megépíteniük szabályok alapján. 2007-ben a műegyetemista diákok is alapítottak egy Formula Student csapatot BME Formula Racing Team néven, e csapathoz csatlakozva végeztem az önálló laboratórium feladatomat.

A feladatom az elektronika csoportban használt virtuális környezet új verziójának az elkészítése volt, valamint a kommunikációs réteg kiegészítés CAN signal kezeléssel, ehhez szükség volt egy kódgenerátor készítésére.

Először a virtuális környezetet készítettem el és konfiguráltam. Fontos volt, hogy a virtuális környezetet a fejlesztők többféle host platformon is futtatni tudják, ezért esett a választás egy VMware alapú virtuális gépre. Először telepítésre került egy Windows Xp virtuális gép, ez adja a virtuális környezet alapját. Ezután következett a fejlesztésre elengedhetetlen GCC fordító, amely biztosítja, hogy a megírt C kódok megfelelően lefordúljanak a mikrovezérlőkre. A mikrovezérlők ARM platformra épülnek, ezért esett a választás a MentorGraphics Sourcery CodeBench Lite 2012.09-63 for ARM EABI fordítócsomagjára. Ez a csomag tartalmazza a fordításhoz és linkeléshez szükséges könyvtárat. Az előző években a kódok elkészítéséhez az Eclipse fejlesztőkörnyezetet használtuk, így az idei verzióban is megmaradt. A fejlesztőeszközök után következett a fejlesztéshez szükséges könyvtárak összeállítása. Az ST minkrovezérlőkhöz kiadott függvénykönyvtárból is a legújabb került telepítésre, valamint az ütemezést biztosító FreeRTOS operációs rendszerből is. Az előző években ezek a könyvtárak egy közös mappán belül helyezkedtek el, idén már megfelelő struktúrában rendezve lehet őket megtalálni. Ehhez szükséges volt kiegészítő makefile-ok elkészítésére. Az idei évtől a hibakeresésre JTAG modulokat használunk, így az Eclipse környezeten belül, konfigurálásra kerültek az ehhez szükséges elemek. Legvégül telepítésre került a Subclipse SVN plugin, amivel a megírt kódok verziókövetését lehet egyszerűen megoldani.

A virtuális környezet elkészítése után következett a kommunikációs réteg kiegészítése. A versenyautóban lévő összes eszköz CAN buszon keresztül kommunikál, az adatokat CAN üzenetekben küldik el. Egy üzenetben úgynevezett signalokat küldenek, amik lehetnek valamilyen mért szenzorértékek vagy vezérlőjelek. Ezek értelmezéséhez szükséges egy adatbázis, amely tartalmazza, hogy melyik signal melyik üzenetben kerül elküldésre, valamint a bináris értékhez milyen faktorizációs és offset érték tartozik. Itt a Vector cég által használt CAN DBC adatbázist lett alkalmazva. Ahhoz, hogy a fejlesztők ne csak üzeneteket, hanem signalokat is tudjanak küldeni, szükséges ezt az adatbázist letárolni a mikrovezérlőben, valamint kiegészítő függvényeket is implementálni kell. Ehhez elkészült egy kódgenerátor, ami az adatbázisból, futtatható C és H fájlokat generál. Ezeket a fájlokat hozzá kell adni a meglévő könyvtárstruktúrához, és a fejlesztők már képesek signalokat küldeni és fogadni.

Felmerülő probléma volt, hogy az adatbázis szöveges formátumáról nincs semmiféle dokumentáció, így az adatbázisba beírt adatok alapján kellett kitalálni, hogy melyik érték pontosan hol helyezkedik el egy adott sorban.

A félév során elkészült a fejlesztésekhez használható virtuális környezet. Ezt a fejlesztők a következő félévben fogják használni az idén elkészült modulok kódjainak a megírásához. Az adatbázis összeállítását és abból a fájlok konvertálását, csak egy megbízott ember végzi, az esetleges hibák elkerülése végett. A kódgenerátor által generált kód hibamentes, a használatát meg kell mutatni majd a fejlesztőknek.