



Středoškolská technika 2023

Setkání a prezentace prací středoškolských studentů na ČVUT

Programujeme „čipy“ pro automotive

Váňa Pavel, Šafler Jakub, Hofman Jakub, Pilař Matěj

Střední průmyslová škola elektrotechnická

Ječná 30, Praha 2

1. Úvod

Výroba aut prochází technologickou revolucí. Rostou požadavky na softwarové vybavení aut, digitalizaci, konektivitu. Mezi zákazníky i investory sílí požadavky na nízkoemisní a udržitelné vozy, ale i na „čistou“ výrobu. Stejným směrem výrobce tlačí přísná unijní regulace.

Podle analýzy Roland Berger/Lazard zvedají ceny nových vozů rostoucí nároky zákazníků, investorů i regulátorů na snižování emisí, cirkulární ekonomiku a diverzitu. Celosvětově sílí trend k elektromobilitě. Dalším zjevným trendem je rostoucí zájem o elektronickou výbavu včetně prvků autonomního řízení, rostou nároky na automobilový software [19]. Např. v [7] je uvedeno, že cena elektronické řídicí jednotky (ECU) se pohybuje mezi 50 000 a 100 000 Kč

Základem ECU je mikrokontrolér, často 32-bit Power Architecture® MCU. Je vyráběn několika velkými výrobci čipů, např. STMicroelectronics či Freescale Semiconductor.

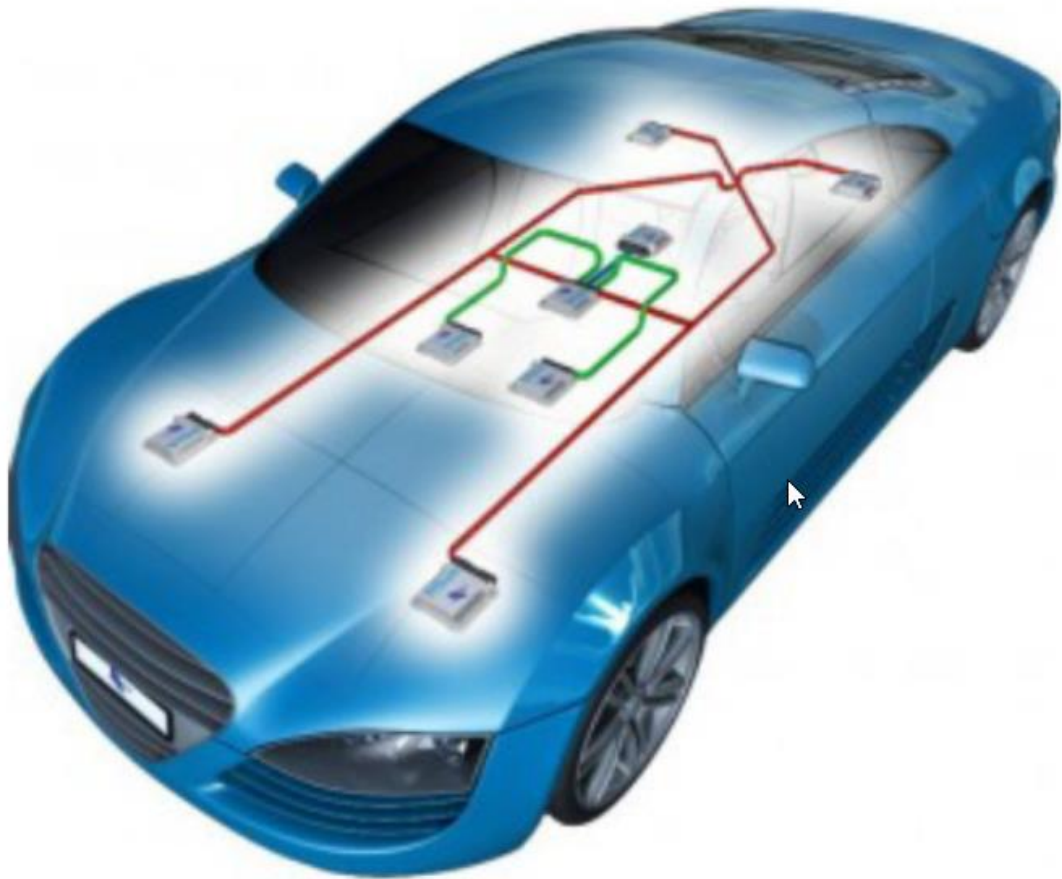
Podstatnou část ceny automobilu tvoří jeho elektrické vybavení, přičemž se stále zvyšuje poměr ceny software vůči ceně hardware. Proto může být pro naše žáky výhodné naučit se programovat i tyto čipy. Úvodem do tohoto programování je kap.5 této práce. Kapitoly 2 až 4 slouží jen jako stručný úvod do problematiky automotive a vycházejí z bakalářských či diplomových prací [4] až [16]. Vřele doporučuji tyto práce si prostudovat, nebo alespoň jen přečíst.

2. Elektrický systém v automobilech

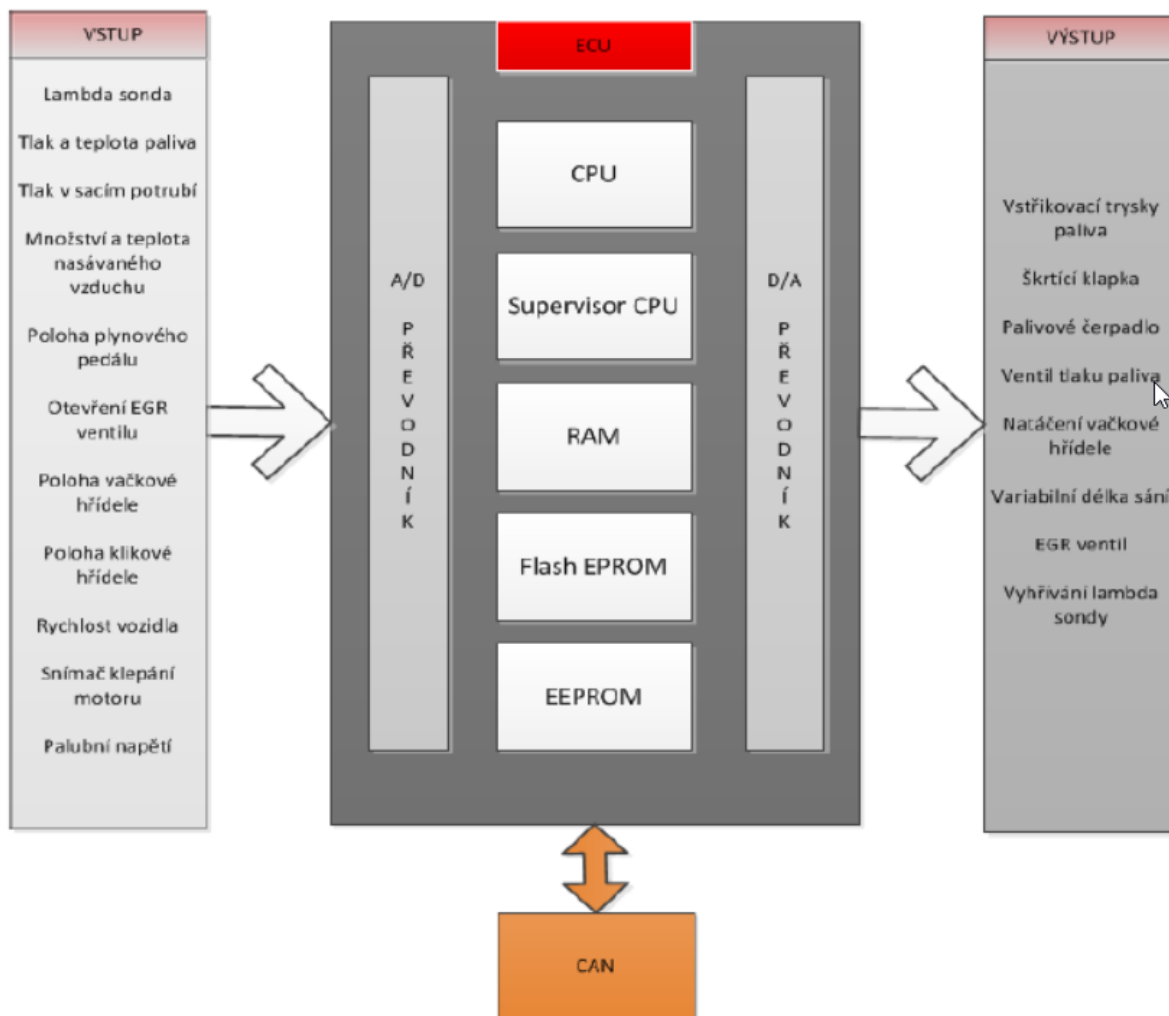
Motorová vozidla byla z počátku plně mechanická a nesledovaly se žádné hodnoty pomocí nichž by se vozidlo mohlo rozhodovat, jak se zachová. Největší změna v tomto odvětví nastala roku 1970, kdy byla představena **elektronická řídicí jednotka (ECU)**, která sleduje hodnoty senzorů v automobilech a na základě dat z nich ovlivňuje fungování určité části automobilu. ECU jsou v dnešních vozidlech samozřejmostí, jedno vozidlo obsahuje až desítky řídicích jednotek, které se starají o chod vozidla od motoru po ohřev sedadel [4]. Jak ECU může vypadat, její umístění v automobilu či její principiální schéma ukazují následující obrázky obr.1 až obr.3.



Obr.1 Electronic Control Unit [7]

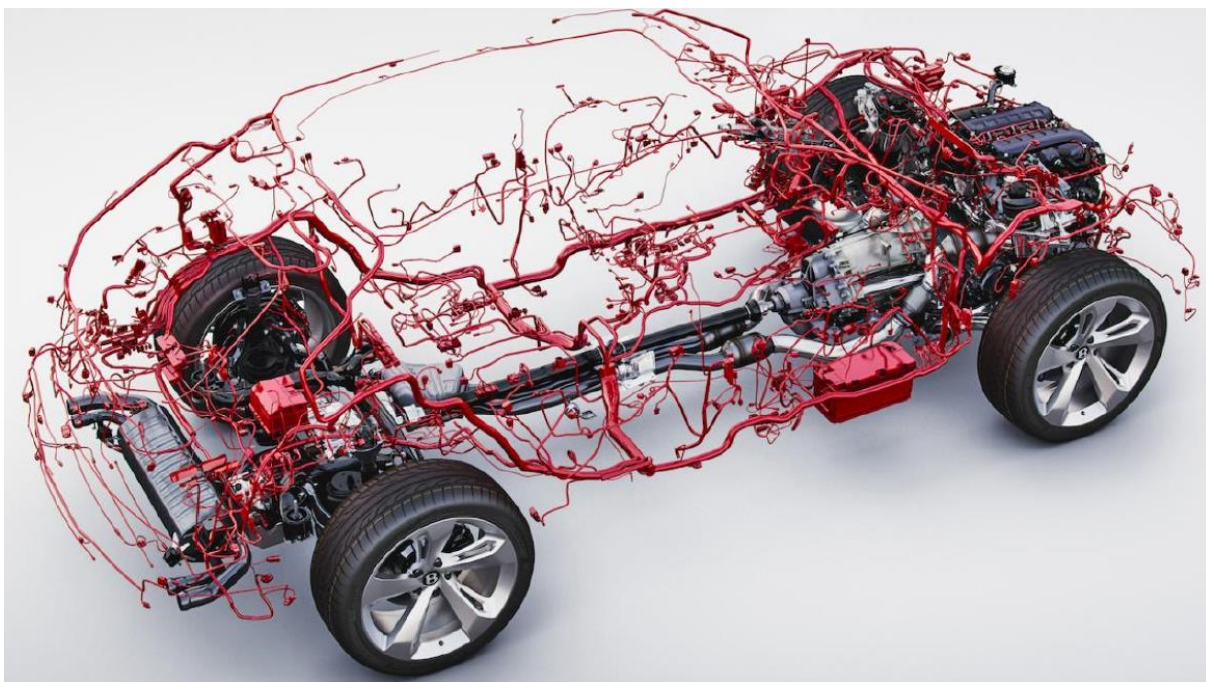


Obr.2 Příklad mikrokontrolérů ve vozidle [3]



Obr.3 schéma ECU [7]

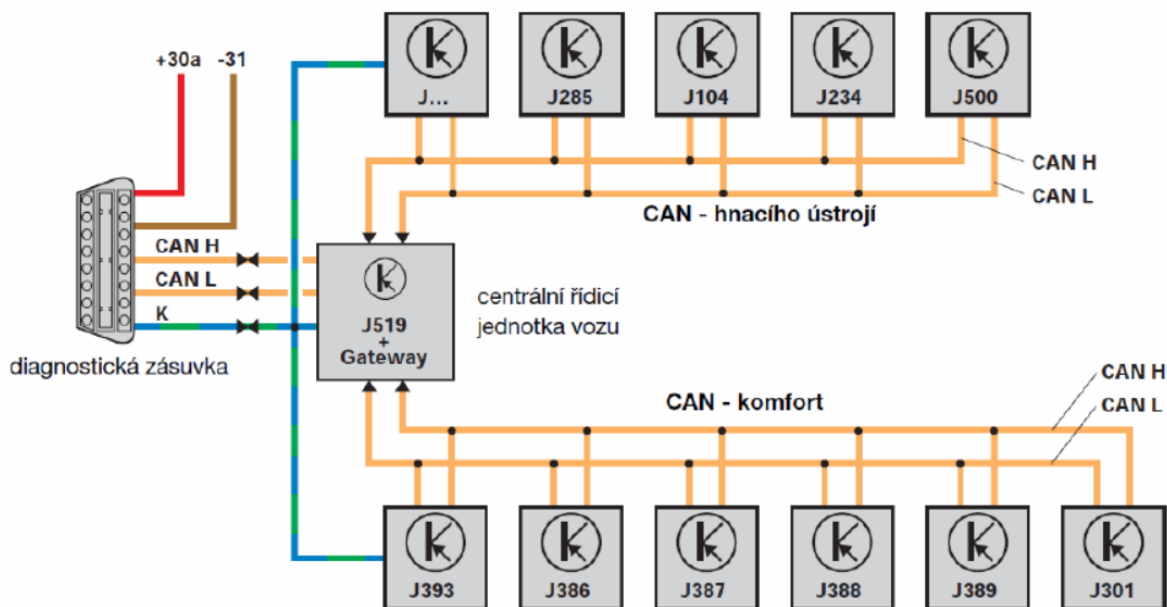
Nezbytnou součástí novodobých vozidel jsou senzory. Zprostředkovávají informace řídicím jednotkám, které následně ovlivňují aktuátory. Data ze senzorů jsou přístupná pomocí standardu OBD, aktuálně OBD-II. Mezi senzory patří např. Snímače polohy, Snímače otáček, Objemové snímače, Snímače tlaku, Snímače teploty, Lambda sonda, Snímače klepání motoru apod. Možné propojení senzorů, řídicích jednotek, komunikačních kanálů ukazují obr.4 a obr.5. Obrázek obr.6 zobrazuje jako příklad propojení řídicích jednotek jejich zapojení v Škoda Felicia.



Obr.4 Grafický model elektroinstalace moderních automobilů [12]



Obr.5 Ilustrační schéma vedení komunikačních kanálů, senzorů a řídicích jednotek ve vozidle Audi A8 2018 [10]



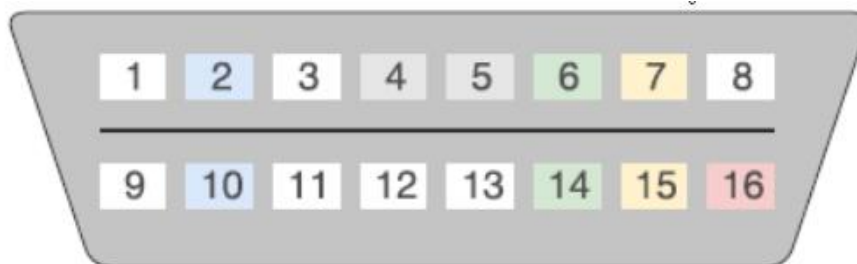
Obr 6. Zapojení řídicích jednotek v automobilu Škoda Fabia, převzato z [5]

3. Palubní diagnostika

Palubní diagnostika, neboli On-Board Diagnostic (zkráceně OBD), je výraz využívaný v automobilovém průmyslu pro schopnost vozidla vlastní diagnostiky a nahlášení chyb. Palubní diagnostika umožňuje mechanikovi nebo vlastníkovi automobilu přístup k informacím o závadách a aktuálním stavu senzorů. Sensory mají za úkol sledovat komponenty, jejichž chybným fungováním by mohlo dojít k nadměrnému vylučování emisí z automobilu. Množství informací, které je uživatel schopen získat, razantně narostl od 80. let minulého století, kdy se objevily první implementace OBD. Nárůst byl způsoben stále se zvyšujícím počtem ECU ve vozidlech a snahou automatizovat různé funkce automobilu. OBD poskytuje mimo jiné i aktuální informace o stavu vozidla (stav palivové nádrže, stav světel apod.) [4]. Aktuálním systémem OBD je OBD-II, který je zabudován přímo ve vozidle, čímž je schopen udržovat stav vozidla, kterého automobil nabýval při výskytu poruchy. Přichází se standardizací diagnostického konektoru a přesně daným popisem pinů. Jedním z nich je pin poskytující energii pro diagnostická zařízení, čímž se eliminuje potřeba připojovat diagnostiku zvlášť do elektřiny. OBD-II určuje, které parametry automobilu je možné sledovat, a jak pro ně kódovat data. Obsahuje také rozšířený seznam chybových kódů s pevně daným formátem zpráv. Díky této standardizaci je možné pomocí jednoho diagnostického zařízení diagnostikovat jakýkoliv automobil, který obsahuje OBD-II. I když prostřednictvím standardizace mohou být přenášeny pouze kódy a data související s emisemi, stal se OBD-II konektor jediným diagnostickým konektorem u mnoha výrobců automobilů, kterému rozšířili jeho diagnostické schopnosti.

3.1. OBD-II konektor

Konektor je umístěn maximálně 60 centimetrů od volantu. Jedná se o 16ti pinový konektor typu samice, jehož vizuální podobu je možné vidět na obr.7. Řídí se standardem SAE J1962, který jasně definuje výstup každého z pinů.



1	podle výrobce	9	podle výrobce
2	J1850 PWM, VPW +	10	J1850 pouze PWM -
3	podle výrobce	11	podle výrobce
4	kostra	12	podle výrobce
5	baterie -	13	podle výrobce
6	CAN-High	14	CAN-Low
7	K-Line (ISO 9141 a 14230)	15	L-Line (ISO 9141 a 14230)
8	podle výrobce	16	baterie +

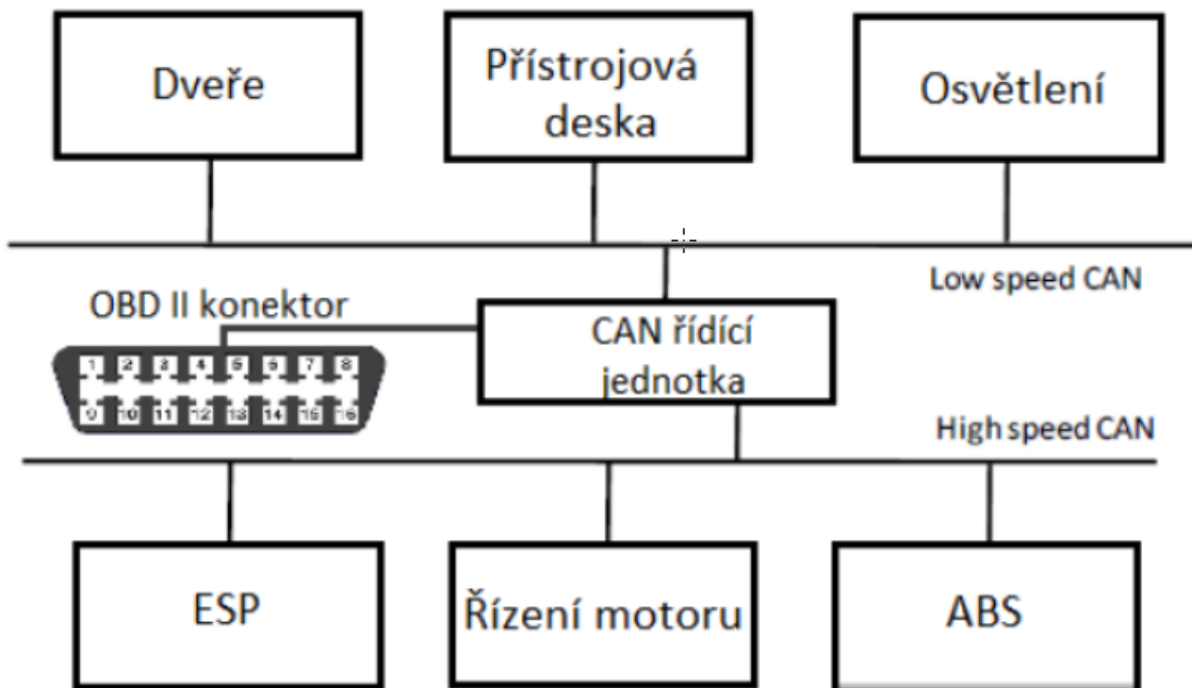
Obr.7 OBD-II port (female) a význam jeho pinu [10]

Vidíme, že piny 1,3, 8, 9, 11, 12 a 13 zde nejsou definovány a jsou definovány výrobcem. Následující tabulka tab.1 ukazuje, jak jsou definovány pro koncernem VW a francouzským koncernem PSA (především Peugeot, Citroën a DS)

Peugeot modely			Škoda Fabia	
Pin	Název pinu	Popis	Název pinu	Popis
2	K-Line	Diagnostika topení a klimatizace	J1850 Bus+	
3		Smysl otáčení motoru	CAN-H	Hnací ústrojí
4	CGND	Kostra vozidla	CGND	Kostra vozidla
5	SGND	Kostra signálu	SGND	Kostra signálu
6	CAN-H	J-2284	CAN-H	Hnacího ústrojí
7	K-Line	Diagnostika motoru a převodovky	K-Line	(ISO 9141-2 a ISO 14230-4)
8			CAN-L	komfort
9			CAN-L	komfort
10	K-Line	Diagnostika modulu volantu	J1850 Bus-	
11	K-Line	Moduly Anti-thief, tlaku vzduchu v pneumatikách, apod.	CAN-L	Hnací ústrojí
12	K-Line	ABS/ESP diagnostika		stínění
13	K-Line	Diagnostika Airbagu		
14	CAN-L	J-2284	CAN-L	Hnací ústrojí
15	L-Line	Diagnostika motoru a převodovky	L-Line	(ISO 9141-2 a ISO 14230-4)
16	+12 V	Akumulátor	+12 V	Akumulátor

Tab.1 zapojení pinů konektoru koncernu PSA a koncernu VW [6]

V této tabulce vidíme, že piny definované výrobcí souvisí s K-Line, L-Line či CAN. Jejich popis najedeme v [6]. Pro komunikaci s řídicími jednotkami jsou totiž předepsány komunikační protokoly. Ty jsou dány normami ISO 9141 a ISO 14230. Podle nich ECU musí mít jednu nebo dvě komunikační linky, K nebo K a L. Připojení linek K a L z jedné nebo více ECU dohromady tvoří sběrníkový systém. Signály používají kódování NRZ a jsou vztaženy ke 12 V, což je napětí autobaterie. Další možností komunikace je využití sběrnice CAN-BUS. CAN-BUS vyvinula firma Bosch. Jedná se o protokol pro sériovou multiplexní komunikaci s vysokou přenosovou rychlostí, zjednodušeným propojením a tím větší přehledností a kvalitním zajištěním dat. Výhodou CAN je, že se využívá stejná sběrnice jako pro normální komunikaci mezi řídicími jednotkami. Sběrnici CAN BUS principiálně ukazuje následující obr.8.



Obr.8 Sběrnice CAN BUS, převzato z [12]

3.1. Prostředky pro komunikaci s řídicí jednotkou [14]

3.1.1. Interpret OBD II

K realizaci komunikace mezi OBD II a digitální linkou na straně PC slouží převodník, který v nejjednodušším případě konvertuje napěťové úrovně signálu mezi sériovým portem a TTL logikou. Tato varianta je sice jednodušší na hardwarové provedení převodníku, ovšem celou komunikaci je nutné řešit programově. To znamená, že je nutné sběrnici iniciovat a budit v definovaných intervalech. Dále se komunikuje s jednotkou pomocí kompletních paketů podle normy SAE. (posílat hlavičky paketu).

Další variantou je převodník – interpret. Tento interpret je již hardwarově složitější a obsahuje mikroprocesor, který se stará jednak o komunikaci s počítačem po sériovém portu, tak o komunikaci s řídicí jednotkou podle normy SAE. Jádro převodníku tvoří mikroprocesor firmy Microchip s firmwarem starajícím se o komunikaci. Jedná se o komerční obvod nesoucí označení ELM XXX, např. ELM327 viz obr.9. Fyzicky je převodník řešen tak, že obsahuje konektor pro propojení s diagnostickou zásuvkou a přímo v jeho těle je integrována veškerá elektronika. Komunikace s počítačem je bezdrátově pomocí bluetooth.



Obr.9 Mobilly OBD-II BT [17]

Komunikace převodníku s počítačem může být i pomocí USB, obr.10.



Obr.10 Mobilly USB VAG OBD-II kabel [18]

Pokud jde o sw vybavení počítače, informaci o různých aplikacích najdeme např. v [4] nebo [10].Práce [10] se navíc zabývá návrhem vlastní aplikace OBD Robot pomocí Android Studia.

4.Chiptuning

Lokalizace dat v řídicí jednotce není standardizovaná, tak pouze výrobce ví, kde jsou uložena konkrétní data. Tyto informace se tedy mohou dostat k některým „VIP klientům“ prostřednictvím originální dokumentace. V praxi získat tento dokument obzvláště u novějších automobilů je takřka nemožné a je nutné postupovat metodou „reverse engineering“. Tato metoda je založena na diagnostických testech vstupů a výstupů řídicí jednotky (dále jen ECU), podle nichž se určuje, které paměťové prostory jsou k jakým účelům využívány. V posledních letech výrobci automobilů podnikají kroky zaměřené proti neautorizovanému přístupu, které mají snahu zkomplikovat komunikaci externího zařízení s automobilem. Tímto se snaží, aby majitelé automobilů využívali pouze autorizované servisy. Bez profesionálního vybavení lze provádět libovolné úpravy ECU dvěma způsoby. Je možné komunikovat přes OBD2 diagnostickou zásuvku, kde je nutné u nových generací ECU překonat ochranné algoritmy. Nebo pomocí BDM rozhraní přistupovat přímo ke konkrétnímu integrovanému obvodu (EEPROM). Tato druhá varianta vyžaduje speciální zařízení. Nevýhodou BDM přístupu je nutnost demontovat ECU z automobilu. Výhodou je jednak snadnější přístup k datům, ale především při přerušení během aktualizace firmware ECU, nebo při zapsání chybných dat, řídicí

jednotka přes OBD2 nekomunikuje. Ke stavu, kdy ECU se chybně přeprogramuje, může dojít, když dojde k poklesu napětí nebo při silném rušení [5].

Existuje celá řada úprav, které se ukrývají pod pojmem chiptuning. Důležité je poznamenat, že chiptuning lze provádět pouze u vozů vybavených ECU. Dále platí, že jak šel technický pokrok kupředu a do automobilů se dostávalo stále více elektroniky, otevřely se dveře také širokému spektru parametrů, které lze na vozidle sledovat a měnit. ECU jsou stále propracovanější, takže u současných vozidel lze sledovat a měnit daleko více parametrů a hodnot než u starších vozidel, ve kterých bylo implementováno mnohem méně elektroniky a snímačů.

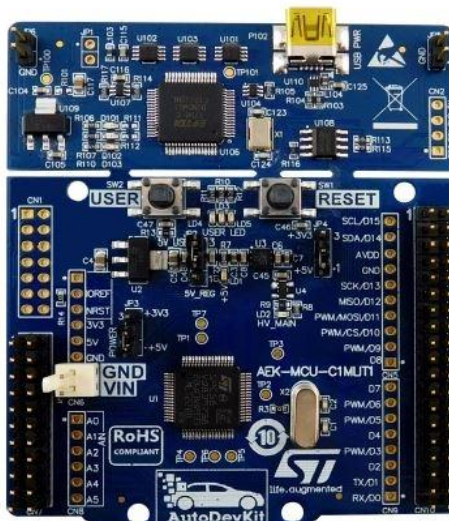
5. Programování Power Architecture MCU pro automotive

V roce 2015 se NXP a Freescale spojují ve 4. největší světovou společnost vyrábějící polovodiče a největšího dodavatele pro automobilový průmysl [29]. Její MCU pro řídicí jednotky jsou založeny na IBM POWER architektuře. Jejich vývoj usnadnilo to, že společně s STMicroelectronics vytvořili společný návrhářský tým a sladili výrobní technologie [30]. Např. Freescale a STMicroelectronics vyvinuly dvoujádrový mikrokontrolér Power Architecture pro aplikace kritické z hlediska bezpečnosti v automobilech [31]. Např. na [32] najdeme článek NXP MPC5xxx / ST SPC5 Microcontrollers začínající větou: Technologie Power Architecture®, společný vývoj ST a NXP, lze nalézt v 32bitových automobilových mikrokontrolérech poskytovaných oběma společnostmi. Dále popisuje řadu MPCxxx firmy Freescale konkrétně MPC56xx, MPC57xx a MPC58xx a řadu SPC5 firmy STMicroelectronics, mající podřady SPC56, SPC57 a SPC58.

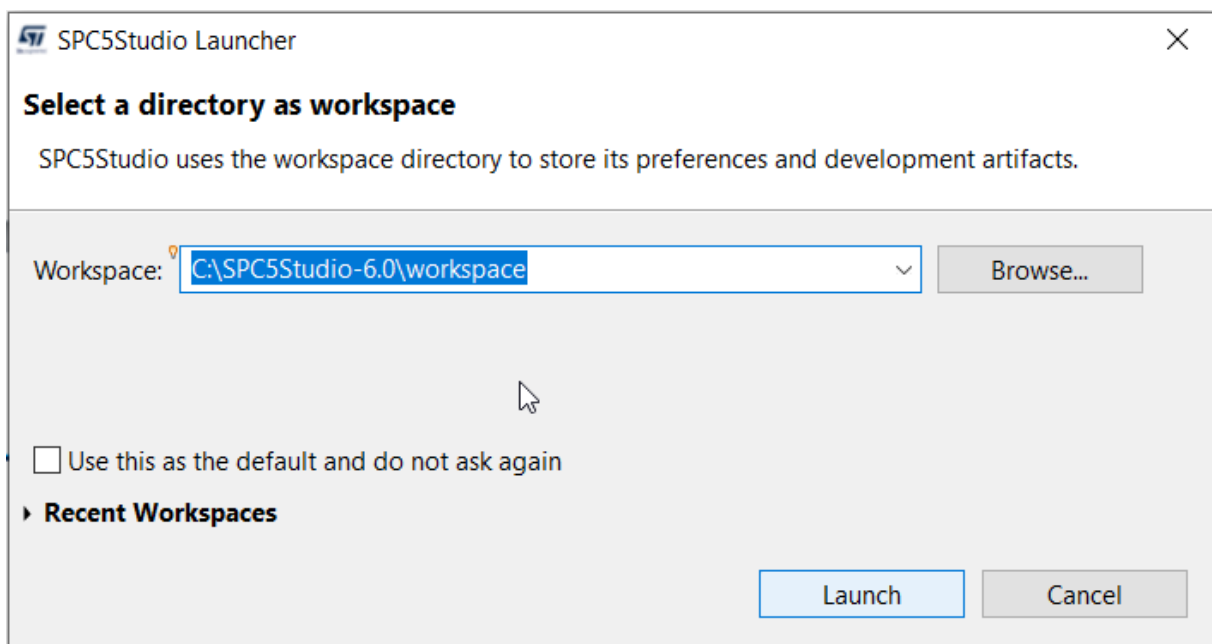
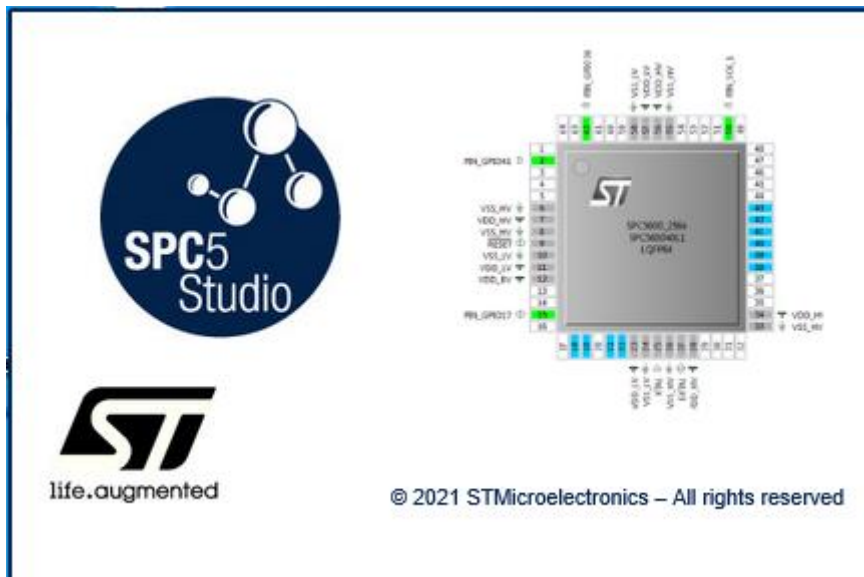
Posledně jmenovanou využijeme nyní při našich prvních krocích při programování MCU pro automotive. Máme totiž k dispozici startkit AEK-MCU-C1MLIT1 s MCU SPC582B60E1 a startkit AEK-MCU-C4MLIT1 s MCU SPC58EC80E5. Řada SPC58 se vyznačuje i hardwarovou podporou bezpečnosti, viz např. článek na HW serveru [33].

Pro tvorbu našich projektů použijeme free IDE SPC5Studio popř. AutoDevKit Studio. AutoDevKit™ je ekosystém zahrnující softwarové a firmwarové komponenty pro nastavení prototypu aplikace. Umožní vývojáři programování aplikace na vysoké úrovni, aniž by musel znát technické detaily hardwaru. Lze však i přistupovat k pokročilým funkcím a funkcím na nízké úrovni [20]. Následující linky [21] až [28] jsou linky na osm kapitol **AutoDevKit™ detailed tutorial**. Užitečné informace při tvorbě našich prvních projektů pro automotive najdeme v italských magisterských pracích [2] a [3].

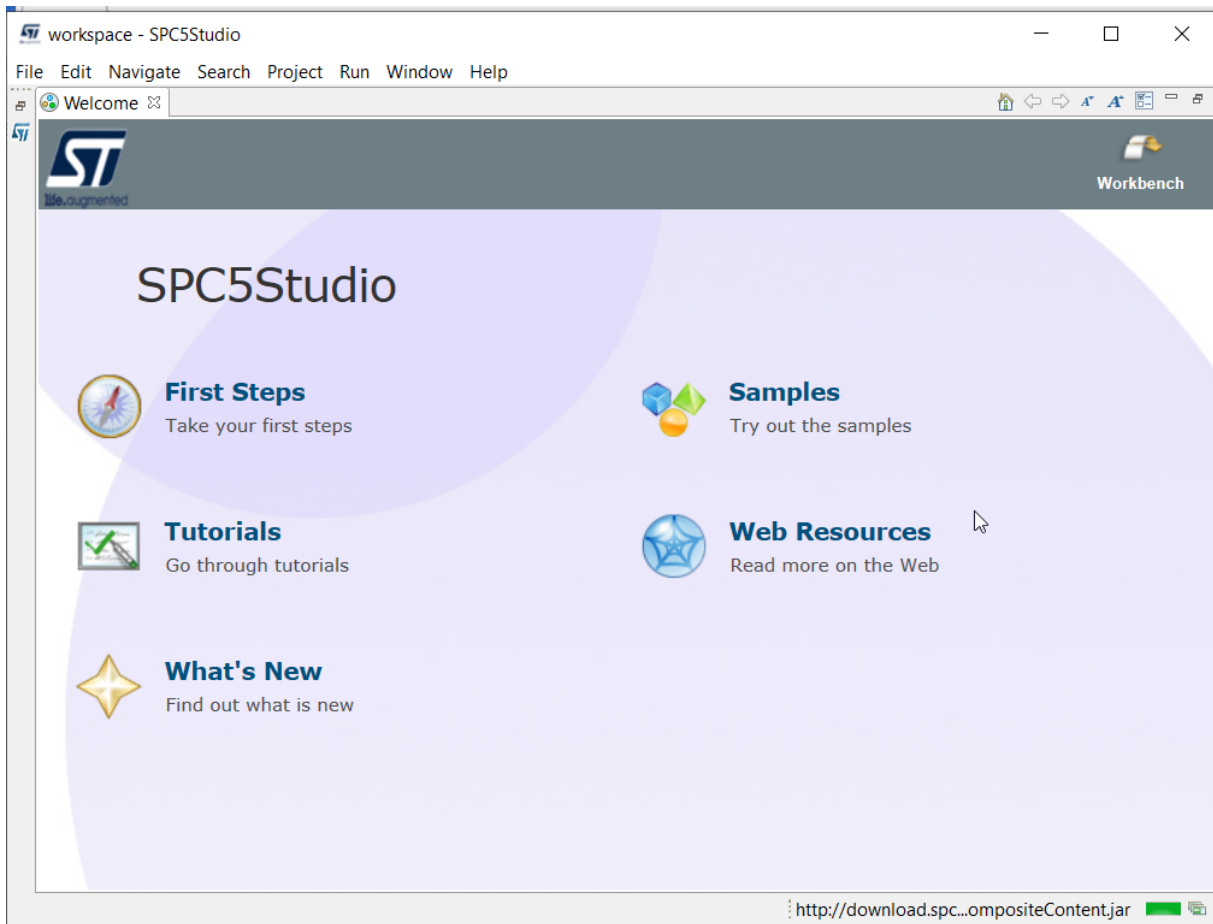
Práce s startkitem AEK-MCU-C1MLIT1



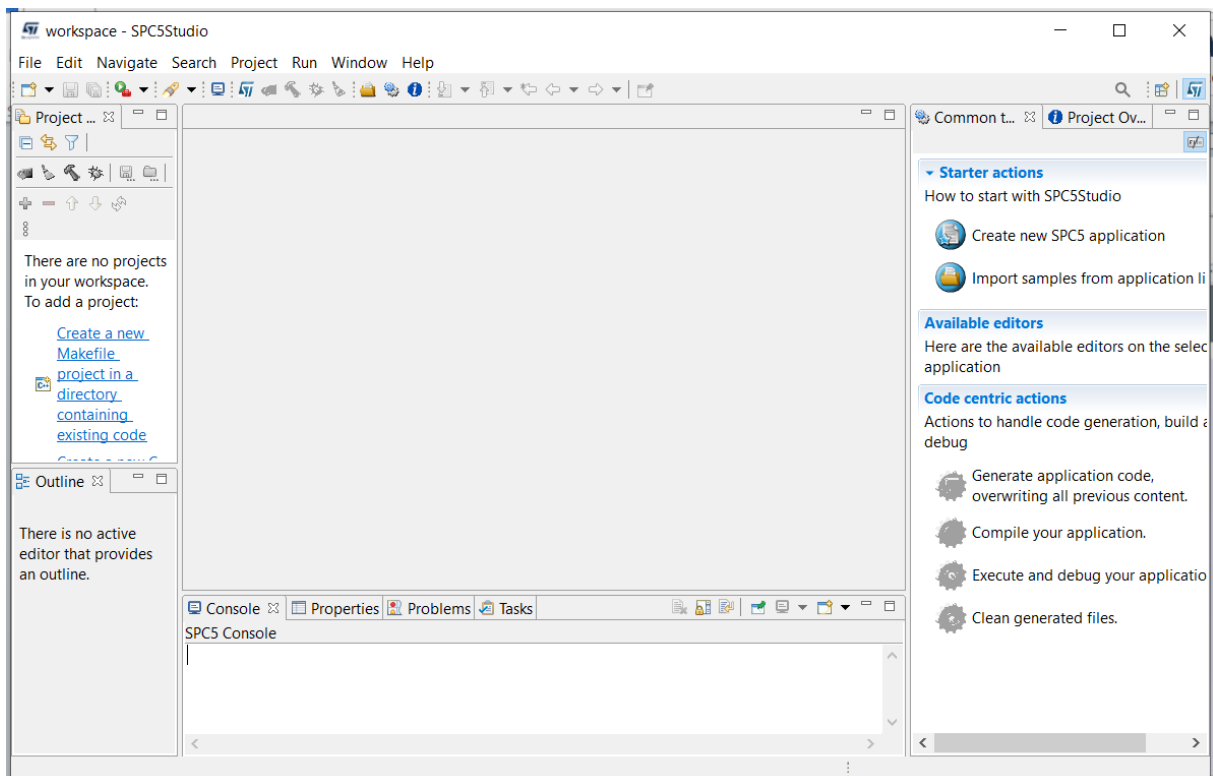
Nejprve si vyzkoušíme příklad od výrobce. Spustíme **SPC5studio**



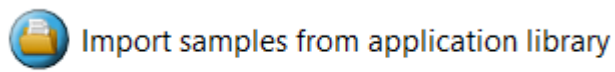
Klikneme na **Launch**



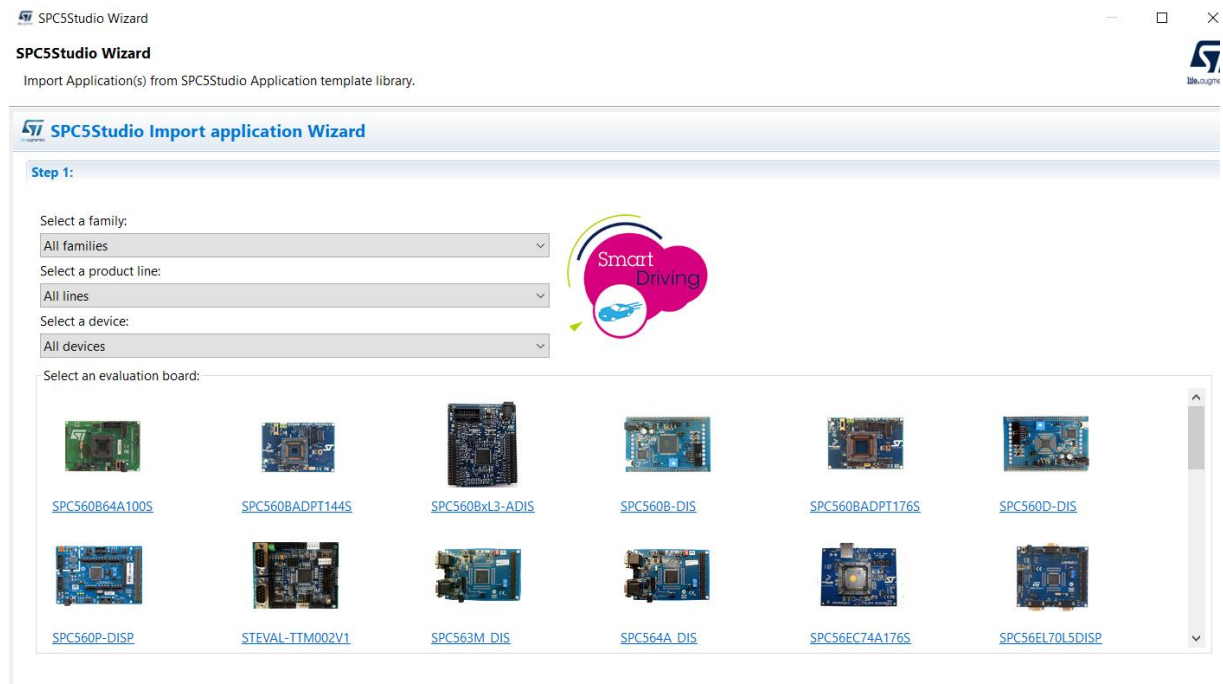
Zavřeme **Welcome** panel.



Klikneme na

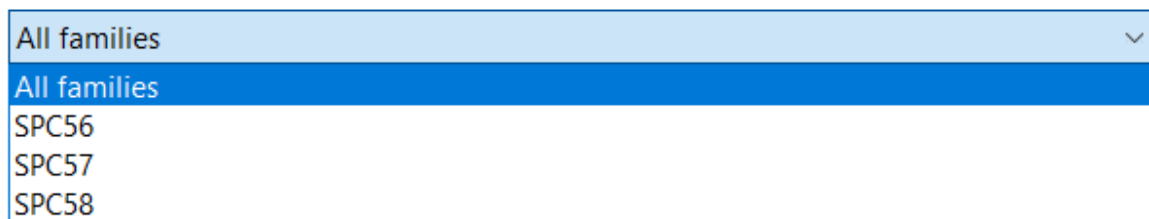


Dostaneme



Nejprve vybereme **family**

Select a family:



Zvolíme **SPC58**

Step 1:

Select a family:

SPC58

Select a product line:

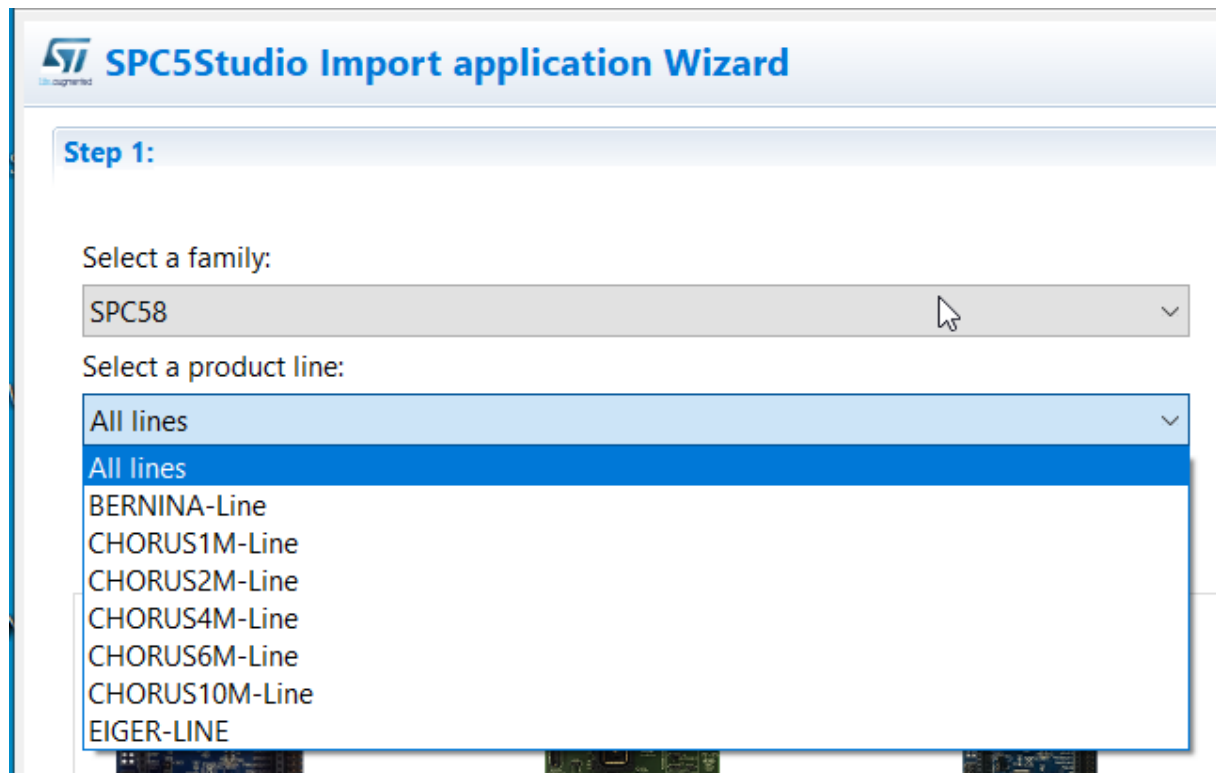
All lines

Select a device:

All devices

Select an evaluation board:


Potom zvolíme **product line**




Vybereme **CHORUS1M-Line**

Step 1:


Select a family:

All families 

Select a product line:

CHORUS1M-Line 

Select a device:

All devices 

Select an evaluation board:



[SPC582B-DIS](#)

A nakonec vybere **device**, zvolíme **SPC582B-DIS**. Má stejný MCU SPC58B60E1 a stejné připojení tří barevných LED a usr tlačítka jako náš startkit **AEK-MCU-C1MLIT1**, který v nabídce bohužel ještě není

SPC5Studio Wizard

Import Application(s) from SPC5Studio Application template library.

SPC5Studio Import application Wizard

Step 1:

Select a family:

Select a product line:

Select a device:

Select an evaluation board:
[SPC582B-DIS](#)

A následně klikneme na tlačítko **Next**.

Objeví se

SPC5Studio Import application Wizard

Step 2:

Template library for selected lines / evaluation boards.

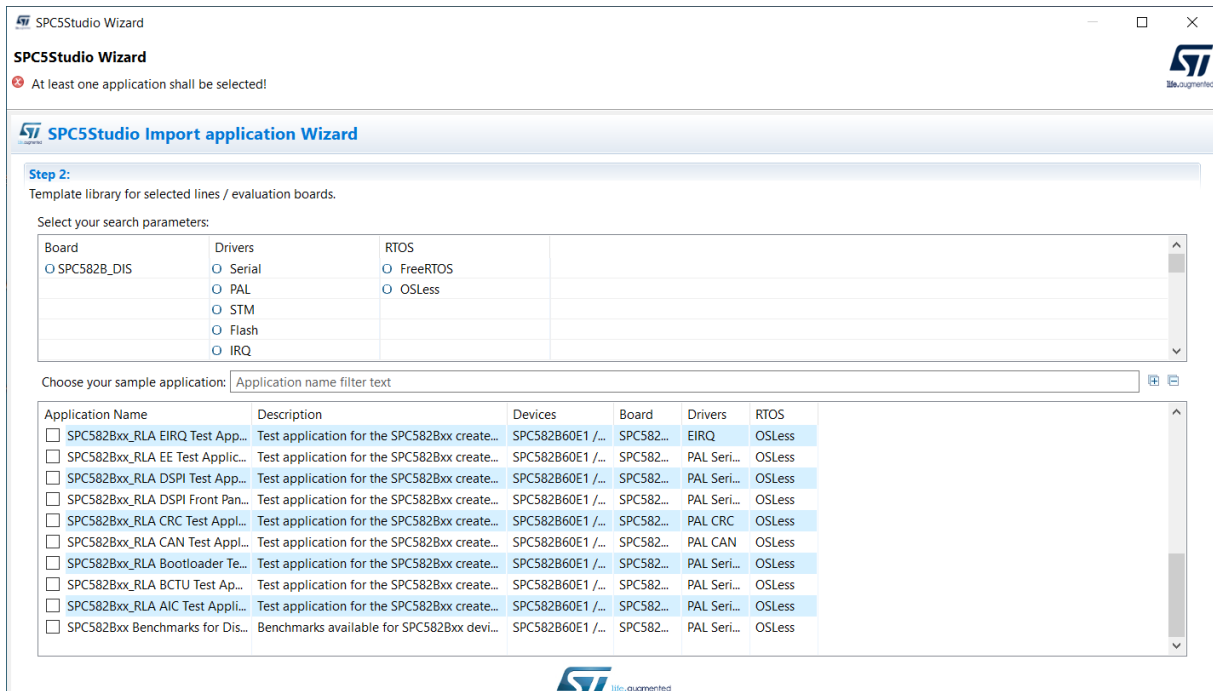
Select your search parameters:

Board	Drivers	RTOS
<input type="radio"/> SPC582B_DIS	<input type="radio"/> Serial <input type="radio"/> PAL <input type="radio"/> STM <input type="radio"/> Flash <input type="radio"/> IRQ	<input type="radio"/> FreeRTOS <input type="radio"/> OSLess

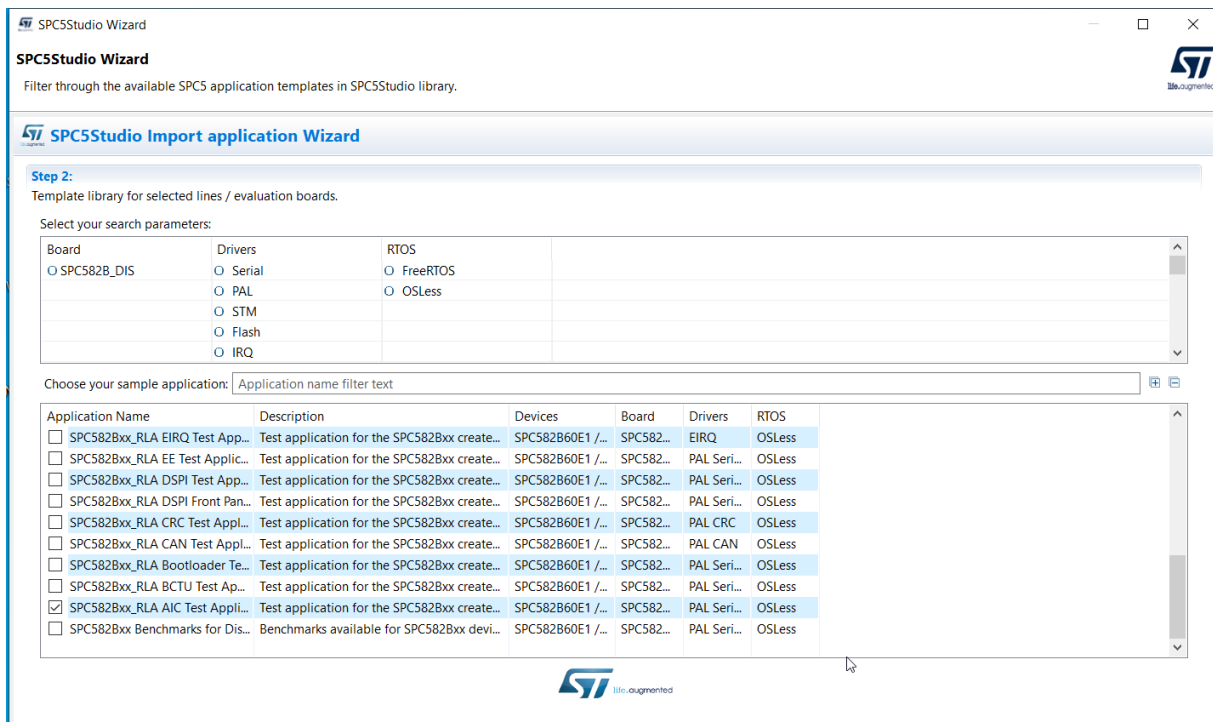
Choose your sample application:

Application Name	Description	Devices	Board	Drivers	RTOS
<input type="checkbox"/> SPC582Bxx_RLA WKPU Test Ap...	Test application for the SPC582Bxx create...	SPC582B60...	SPC582...	PAL WK...	OSLess
<input type="checkbox"/> SPC582Bxx_RLA SWT Test Appl...	Test application for the SPC582Bxx create...	SPC582B60...	SPC582...	PAL SWT	OSLess
<input type="checkbox"/> SPC582Bxx_RLA STM Test Appl...	Test application for the SPC582Bxx create...	SPC582B60...	SPC582...	PAL Seri...	OSLess
<input type="checkbox"/> SPC582Bxx_RLA SERIAL Test A...	Test application for the SPC582Bxx create...	SPC582B60...	SPC582...	PAL Seri...	OSLess
<input type="checkbox"/> SPC582Bxx_RLA SERIAL DMA T...	Test application for the SPC582Bxx create...	SPC582B60...	SPC582...	PAL Seri...	OSLess
<input type="checkbox"/> SPC582Bxx_RLA SARADC Test ...	Test application for the SPC582Bxx create...	SPC582B60...	SPC582...	PAL Seri...	OSLess
<input type="checkbox"/> SPC582Bxx_RLA RTC Test Appli...	Test application for the SPC582Bxx create...	SPC582B60...	SPC582...	PAL Seri...	OSLess
<input type="checkbox"/> SPC582Bxx_RLA PWM-ICU Test...	Test application for the SPC582Bxx create...	SPC582B60...	SPC582...	PAL ICU ...	OSLess
<input type="checkbox"/> SPC582Bxx_RLA PIT Test Appli...	Test application for the SPC582Bxx create...	SPC582B60...	SPC582...	PAL PIT	OSLess
<input type="checkbox"/> SPC582Bxx_RLA LIN Test Appli...	Test application for the SPC584Bxx create...	SPC582B60...	SPC582...	PAL LIN ...	OSLess
<input type="checkbox"/> SPC582Bxx_RLA IRQ Test Appli...	Test application for the SPC582Bxx create...	SPC582B60...	SPC582...	IRQ	OSLess

Posuneme

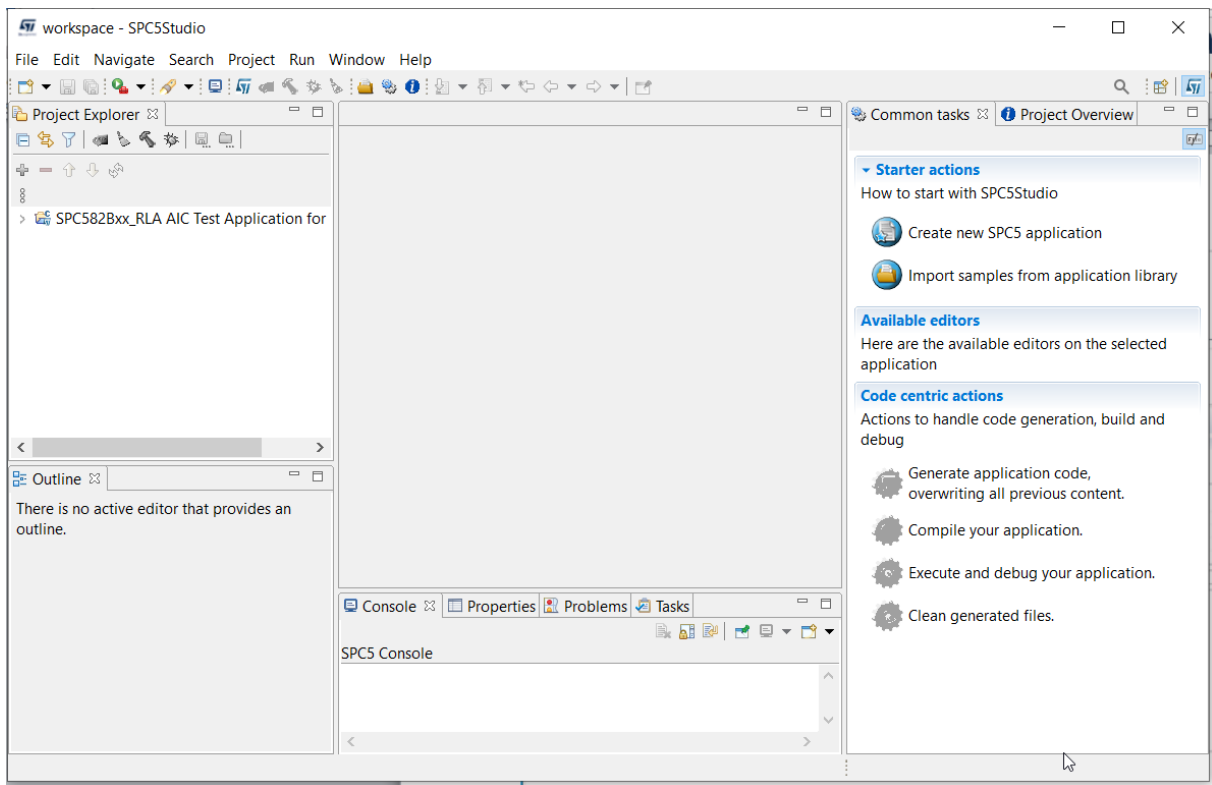


A vybereme předposlední příklad

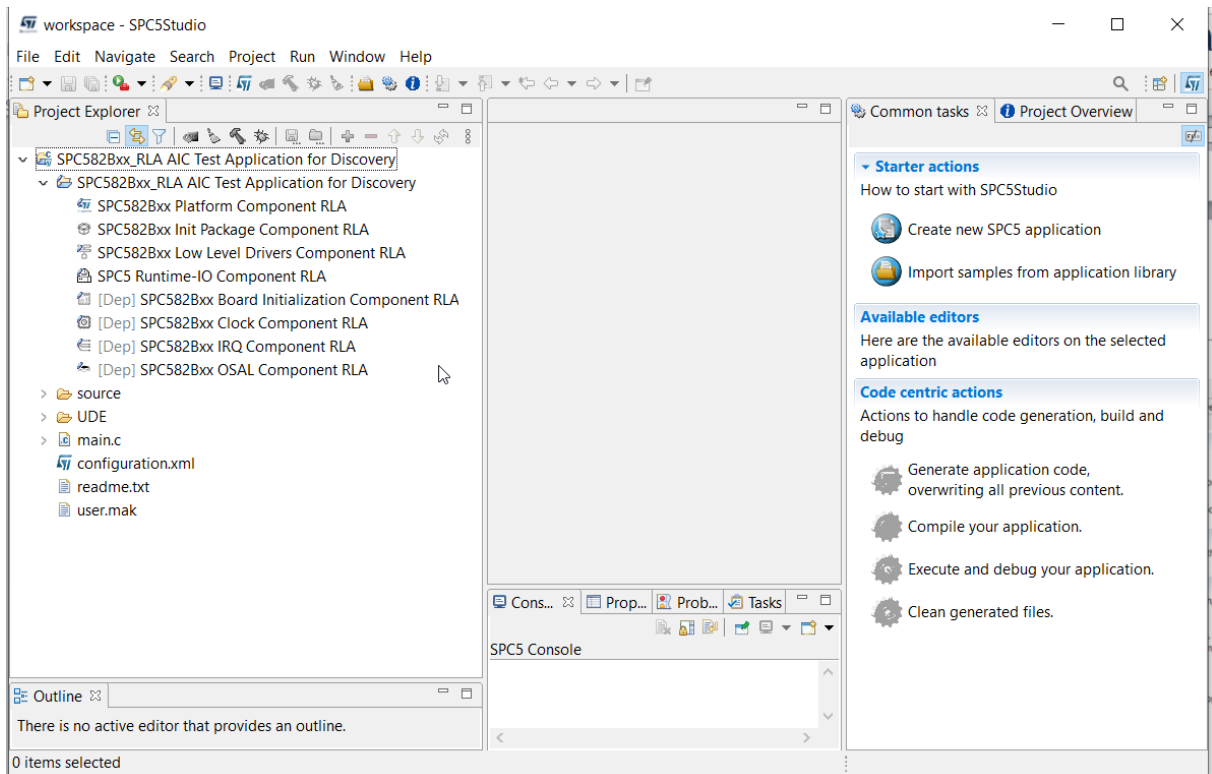


A klikneme na tlačítko **Finish**.

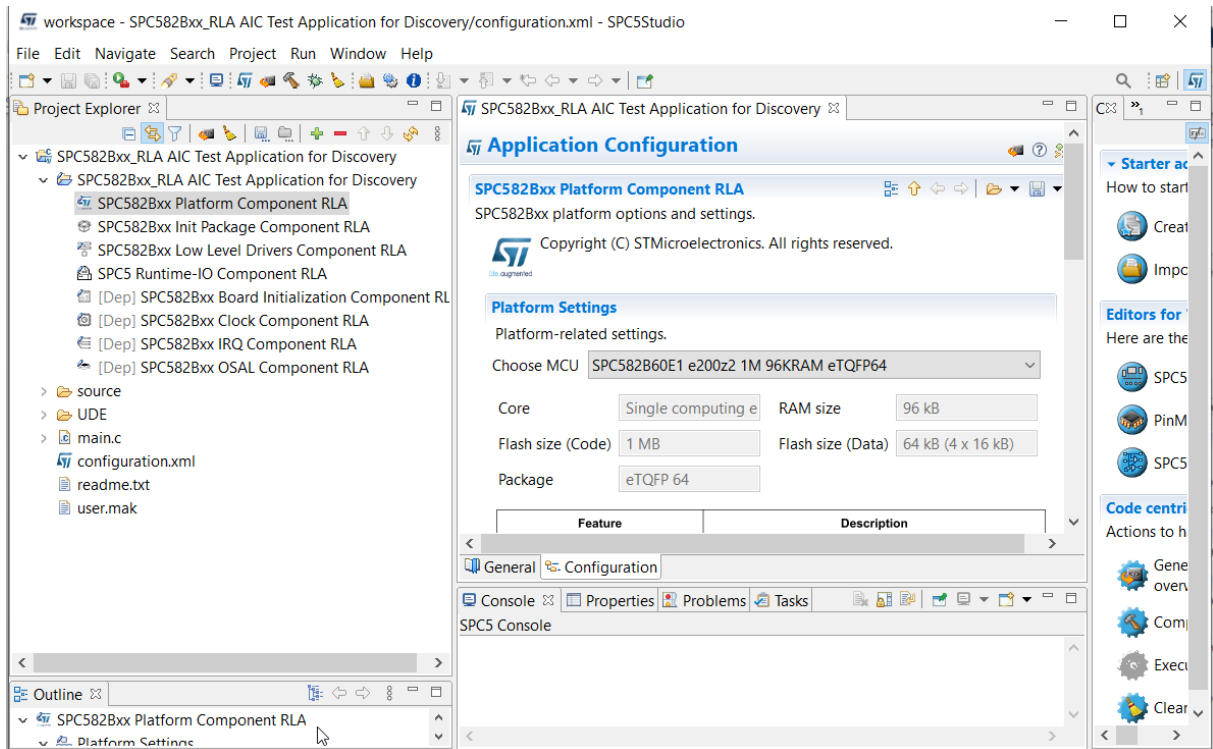
(pozn. V posledním příkladě Benchmarks je zřejmě chyba – místo SPC582B totiž používá SPC)



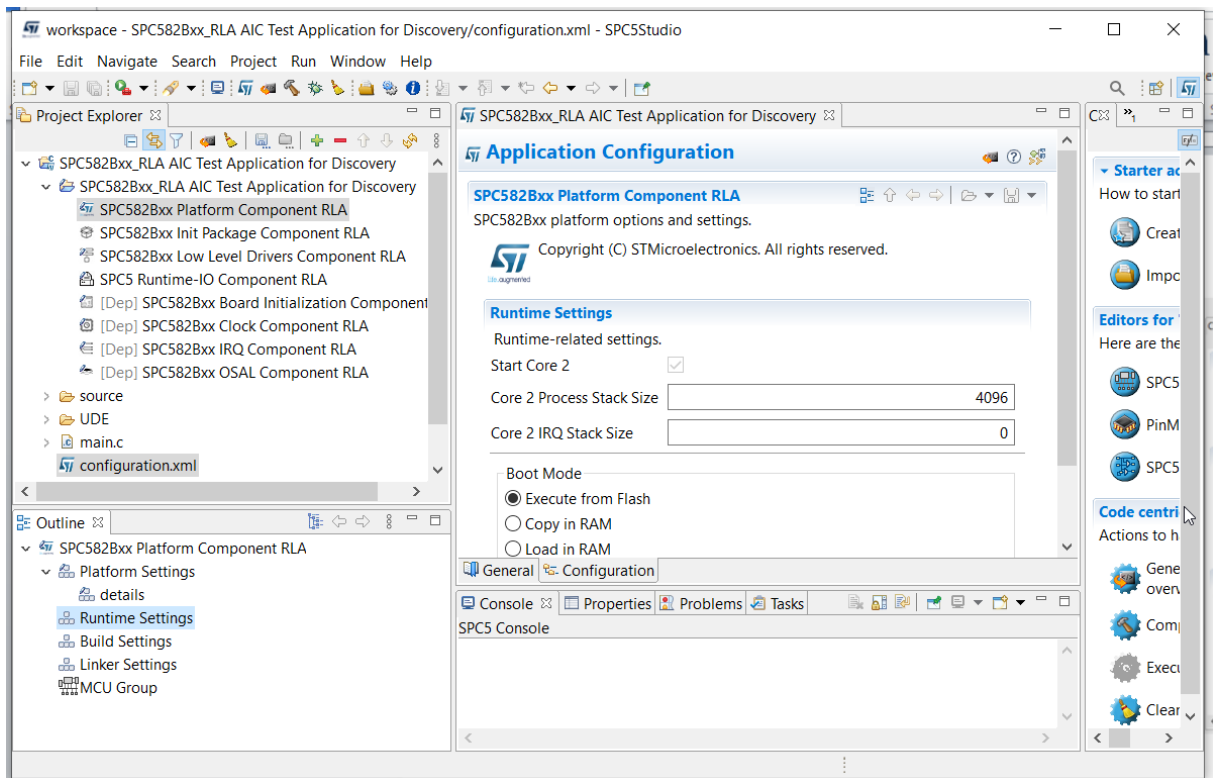
Rozklikneme jméno programu v **Project Exploreru**



A označíme položku **SPC582Bxx Platform Component RLA**

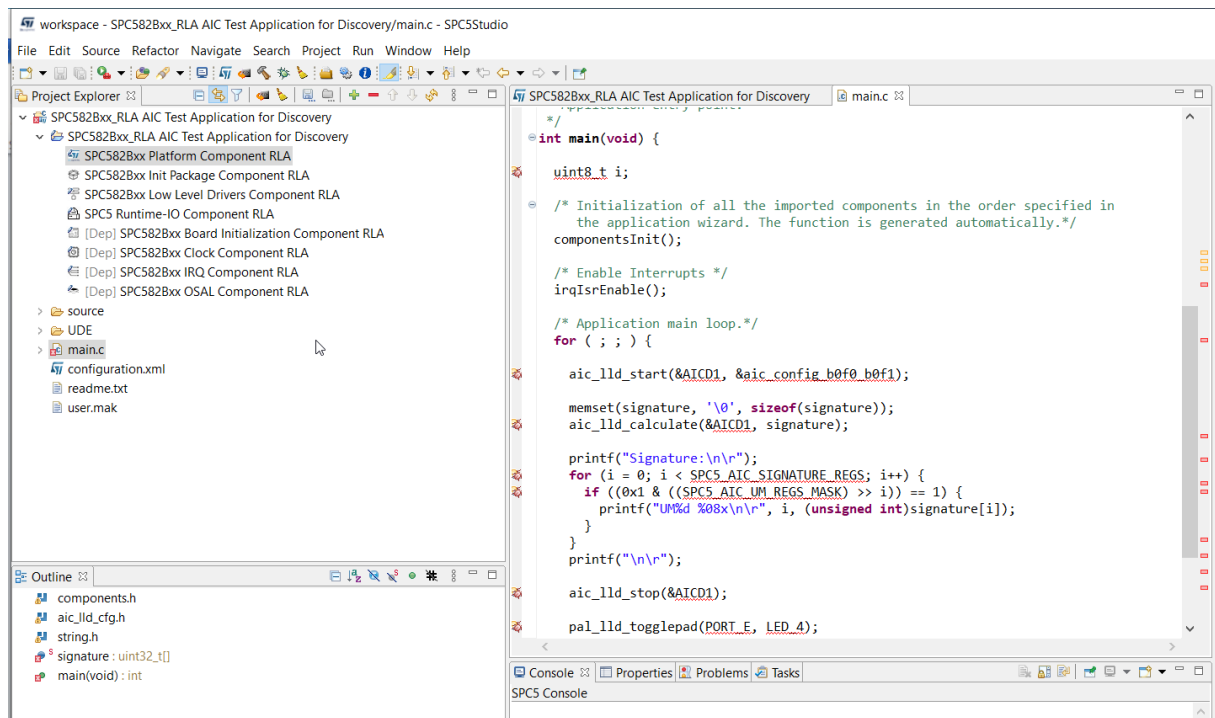


A v **Outline** označíme **Runtime Settings**



Poté v **Boot Mode** místo **Execute from Flash** vybereme **Load in Ram** . (je to důležité !)

Nyní zbývá upravit soubor **main.c** Proto ho otevřeme



V tomto souboru dáme do poznámek // několik řádků

```
SPC582Bxx_RLA AIC Test Application for Discovery *main.c
the application wizard. The function is generated automatically.*/
componentsInit();

/* Enable Interrupts */
irqIsrEnable();

/* Application main loop.*/
for ( ; ; ) {

    // aic_lld_start(&AICD1, &aic_config_b0f0_b0f1);

    // memset(signature, '\0', sizeof(signature));
    // aic_lld_calculate(&AICD1, signature);

    printf("Signature:\n\r");
    // for (i = 0; i < SPC5_AIC_SIGNATURE_REGS; i++) {
    //     if ((0x1 & ((SPC5_AIC_UM_REGS_MASK) >> i)) == 1) {
    //         printf("UM%d %08x\n\r", i, (unsigned int)signature[i]);
    //     }
    // }
    printf("\n\r");

    // aic_lld_stop(&AICD1);

    pal_lld_togglepad(PORT_E, LED_4);
    osalThreadDelayMilliseconds(100);
    pal_lld_togglepad(PORT_A, LED_3);
    osalThreadDelayMilliseconds(100);
    pal_lld_togglepad(PORT_D, LED_5);
    osalThreadDelayMilliseconds(100);
}
}
```

Takže vlastně zůstane k provedení jen

```
int main(void) {

    uint8_t i;

    /* Initialization of all the imported components in the order specified in
       the application wizard. The function is generated automatically.*/
    componentsInit();

    /* Enable Interrupts */
    irqIsrEnable();

    /* Application main loop.*/
    for ( ; ; ) {
        printf("Signature:\n\r");
        printf("\n\r");
        pal_lld_togglepad(PORT_E, LED_4);
        osalThreadDelayMilliseconds(100);
        pal_lld_togglepad(PORT_A, LED_3);
        osalThreadDelayMilliseconds(100);
        pal_lld_togglepad(PORT_D, LED_5);
        osalThreadDelayMilliseconds(100);
    }
}
```


Ještě postupně spustíme



Clean generated files.

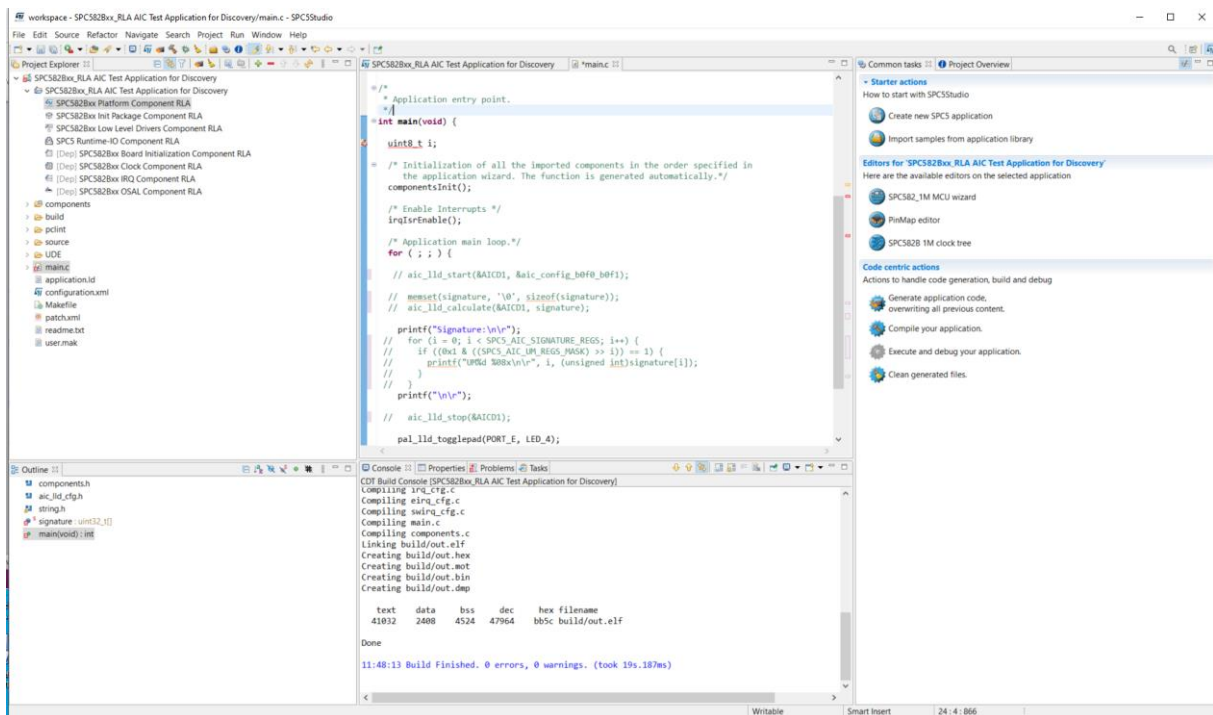


Generate application code,
overwriting all previous content.



Generate application code,
overwriting all previous content.

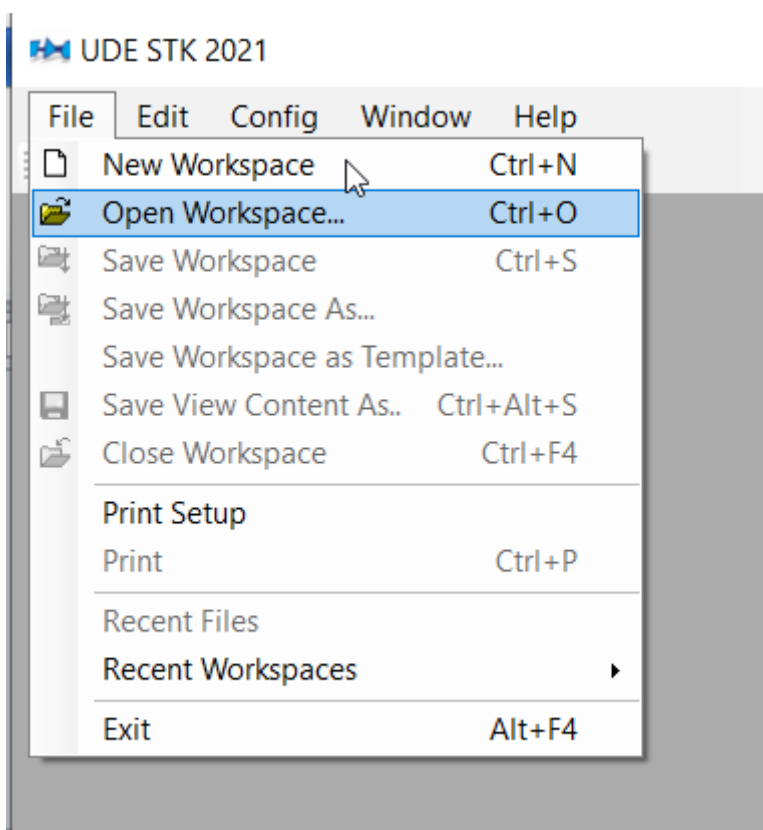
Po překladu dostaneme



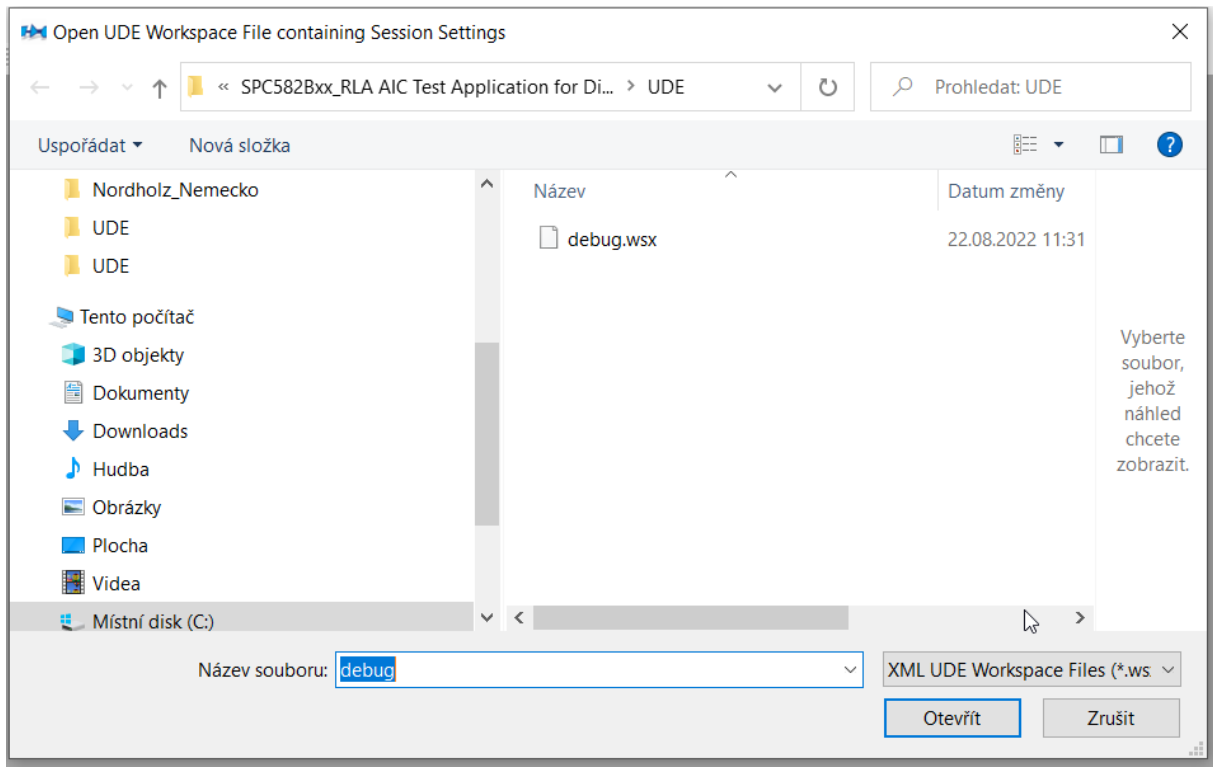
Zbývá nahrát program do RAM startkitu. Startkit připojíme pomocí usb kabelu k PS a spustíme program **UDE starterkit 2021**



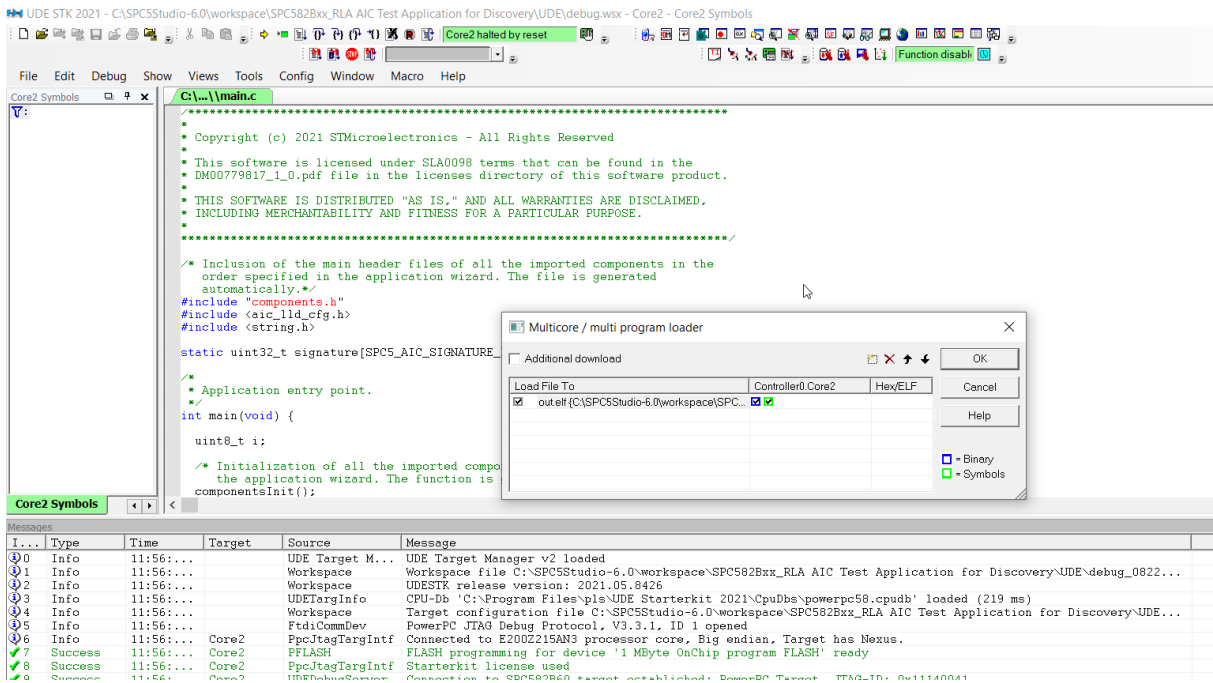
V jeho menu



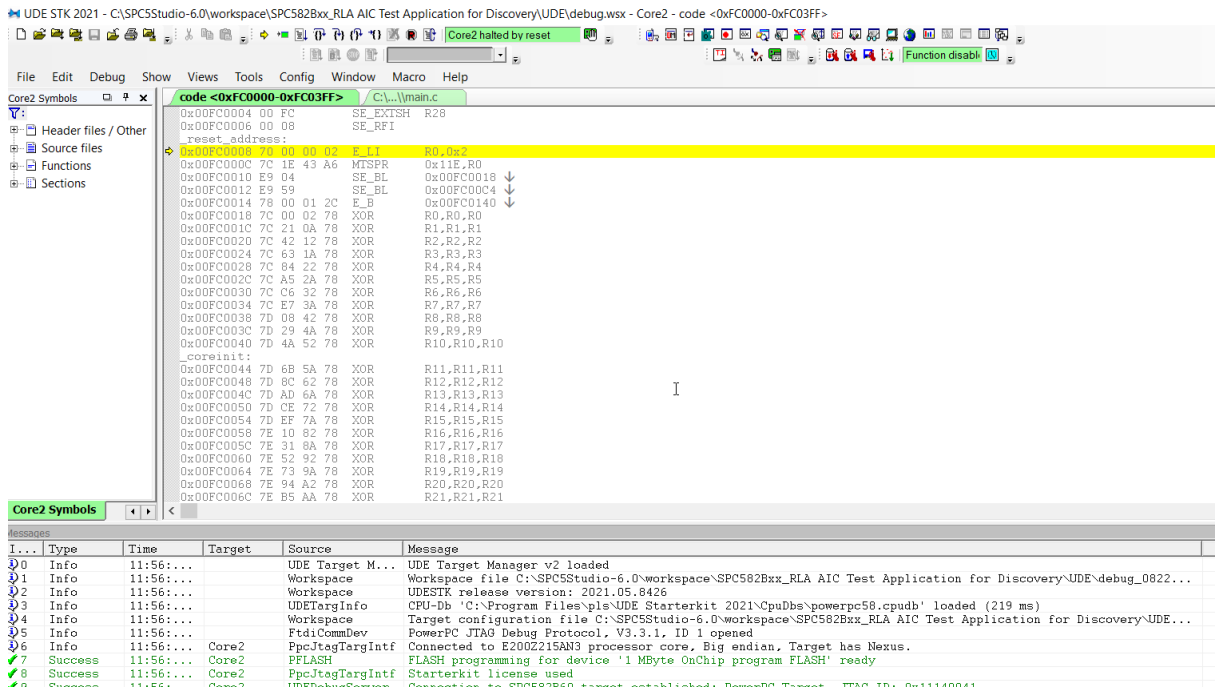
Vybereme položku **Open Workspace**



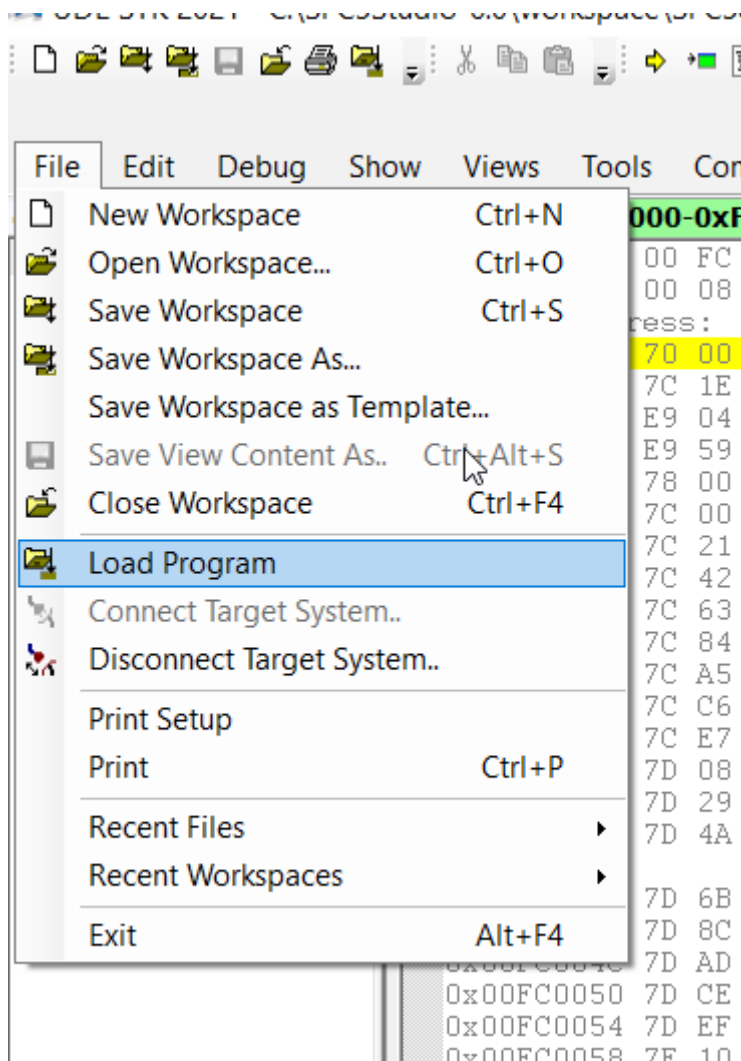
A vybereme soubor debug.wsx uložený v podadresáři UDE adresáře našeho projektu



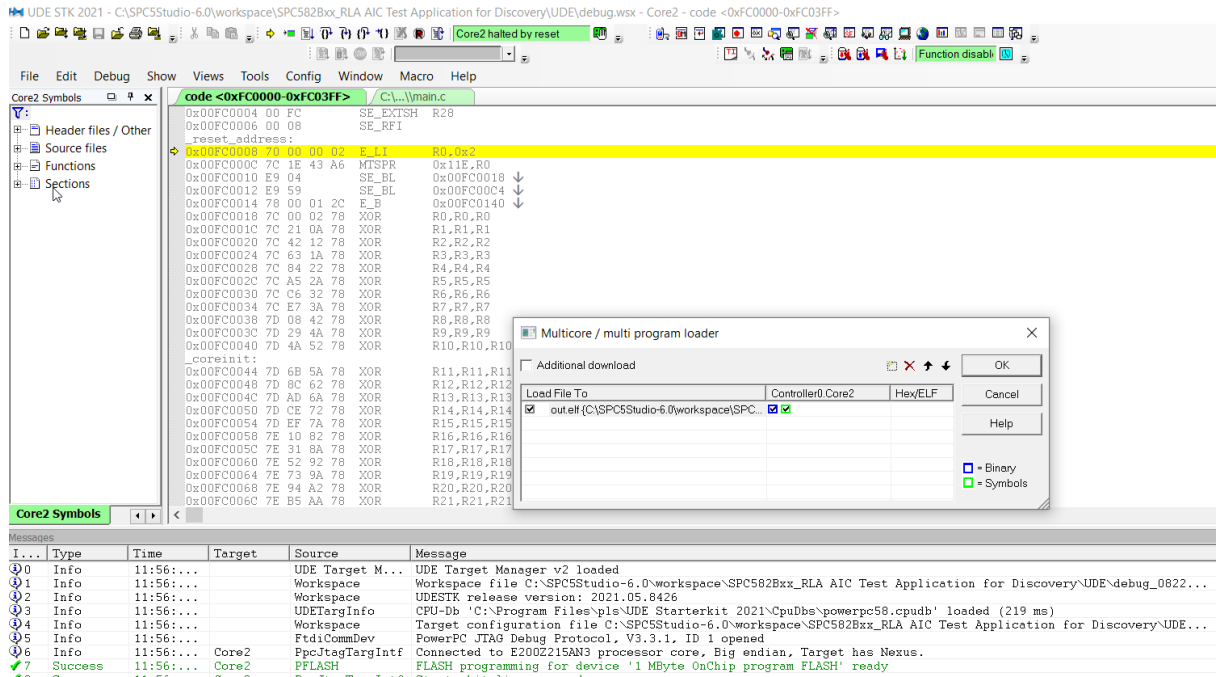
Klikneme na OK



A rozvineme menu



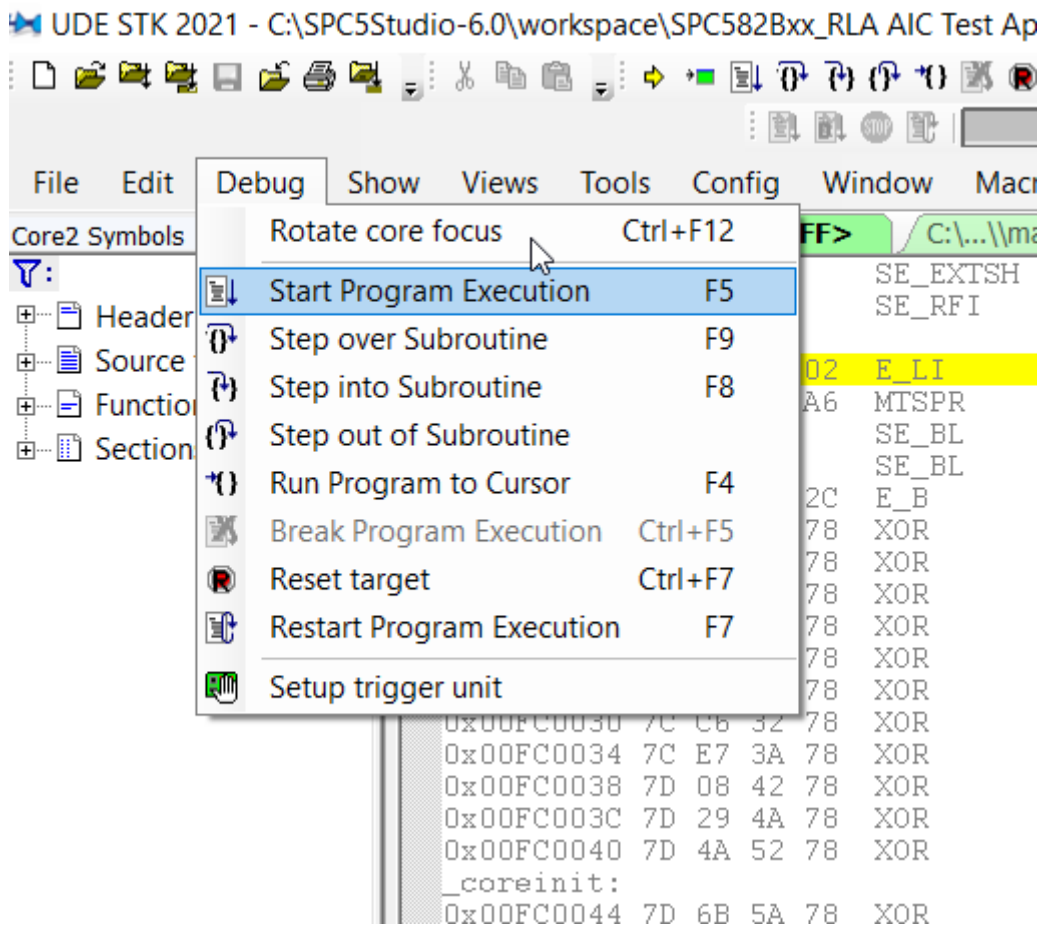
A vybereme položku Load Program



The screenshot shows the UDE STK 2021 IDE interface. The main window displays assembly code for the file 'C:\...\main.c'. The code includes instructions like SE_EXISH, SE_RF1, F.LI, and various XOR operations. A dialog box titled 'Multicore / multi program loader' is open, showing a table with columns for 'Load File To', 'Controller', and 'Hex/ELF'. The file 'out.elf' is selected in the 'Load File To' column, and 'Controller0 Core2' is selected in the 'Controller' column. The 'Hex/ELF' column has a checked box. The dialog also has 'Additional download' and 'OK' buttons. At the bottom, the 'Messages' window shows a log of events, including 'FLASH programming for device '1 MByte OnChip program FLASH' ready'.

I...	Type	Time	Target	Source	Message
0	Info	11:56:...		UDE Target M...	UDE Target Manager v2 loaded
1	Info	11:56:...		Workspace	Workspace file C:\SPC5Studio-6.0\workspace\SPC582Bxx_RLA AIC Test Application for Discovery\UDE\debug_0822...
2	Info	11:56:...		Workspace	UDESTRK release version: 2021.05.8426
3	Info	11:56:...		UDETargInfo	CPU-Db 'C:\Program Files\p1e\UDE Starterkit 2021\CpuDbs\powerpc58.cpubd' loaded (219 ms)
4	Info	11:56:...		Workspace	Target configuration file C:\SPC5Studio-6.0\workspace\SPC582Bxx_RLA AIC Test Application for Discovery\UDE...
5	Info	11:56:...		FtdiCommDev	PowerPC JTAG Debug Protocol, V3.3.1, ID 1 opened
6	Info	11:56:...	Core2	EpoJtagTargIntf	Connected to E200Z215AN3 processor core, Big endian, Target has Nexus.
7	Success	11:56:...	Core2	PFLASH	FLASH programming for device '1 MByte OnChip program FLASH' ready

Klikneme na **OK** a poté program spustíme

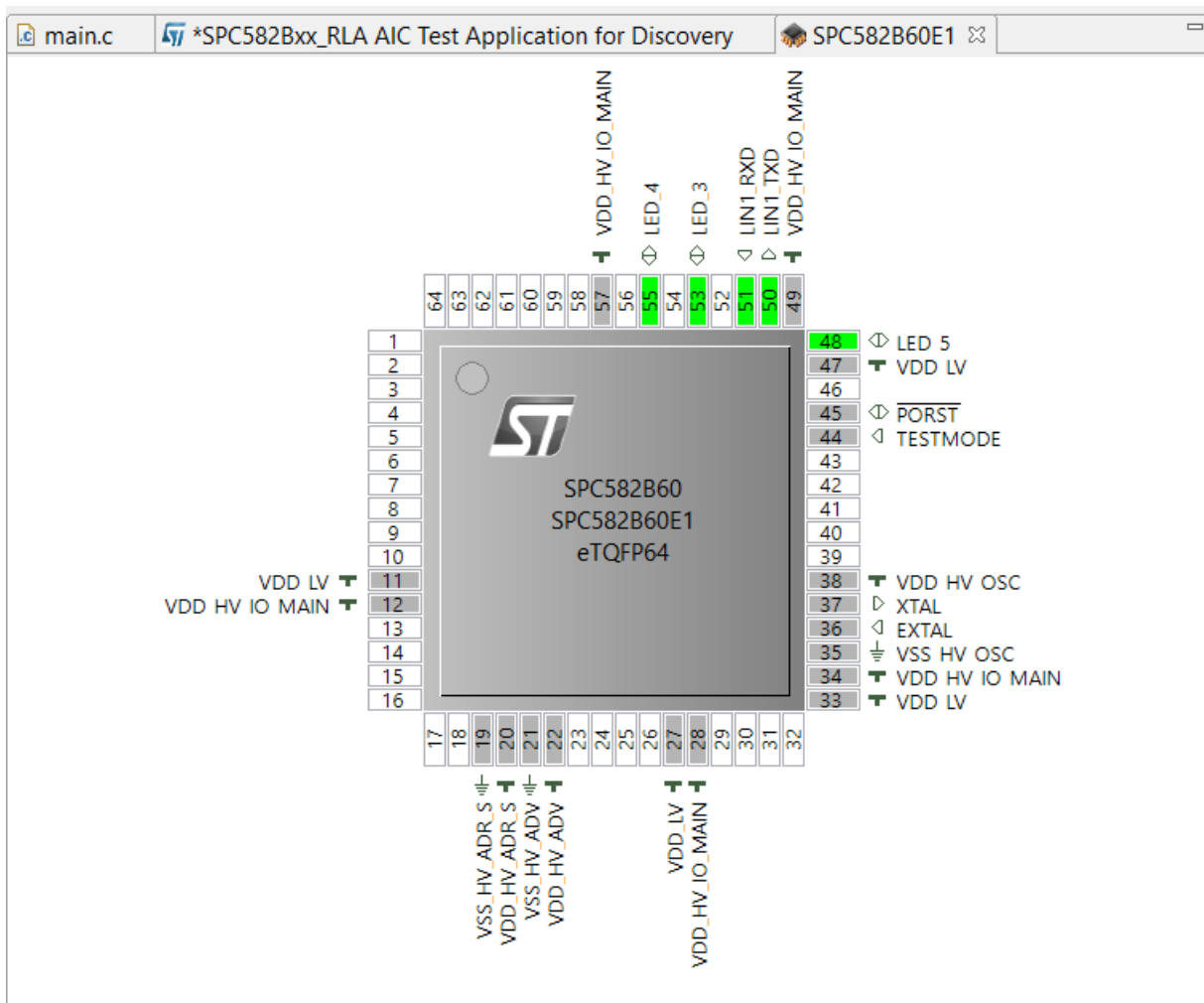


Postupně se budou po 0,1 s rozsvěcovat červená, žlutá a zelená LED. Při tvorbě programu ovšem potřebujeme znát jejich připojení k pinům MCU. K tomu se bude hodit následující tabulka tab.2.

pin	Označení pinu	Označení periferie na startkitu	Popis periferie
64	PF2	User	Uživatelské tlačítko
53	PA11	LD3	Žlutá LED
55	PE11	LD4	Červená LED
48	PD5	LD5	Zelená LED

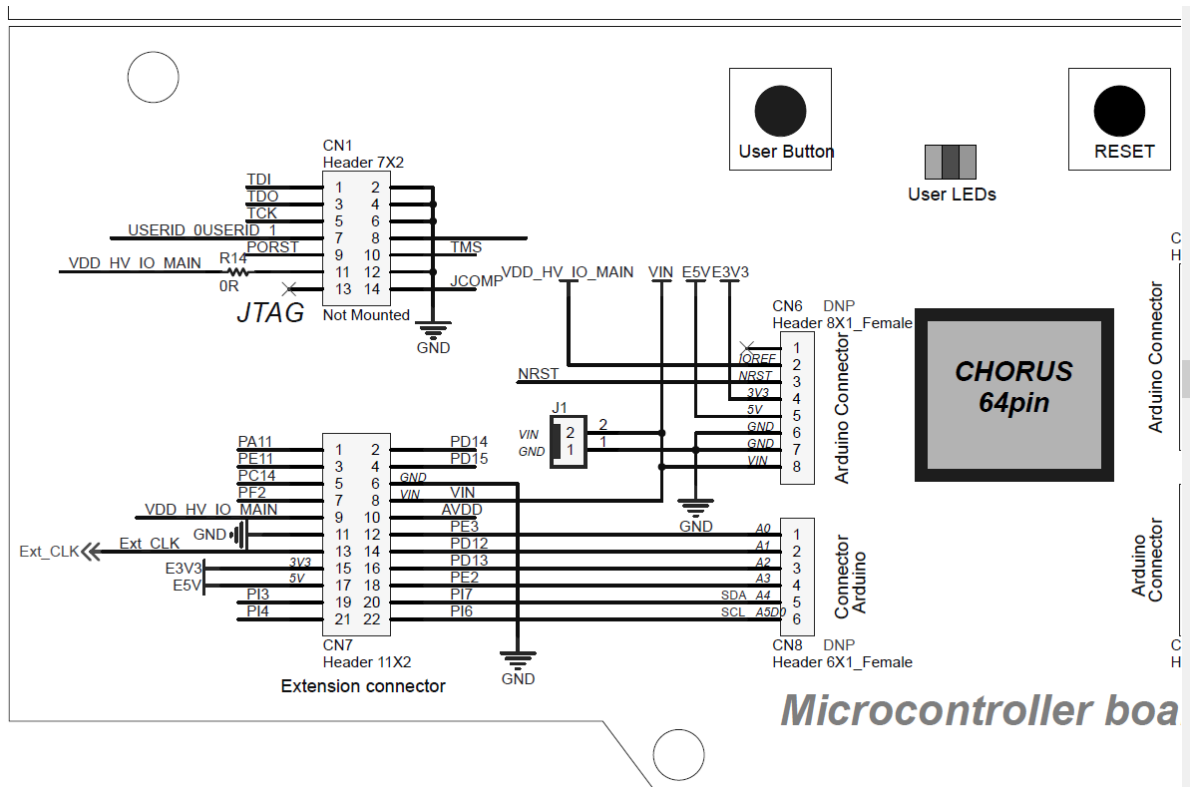
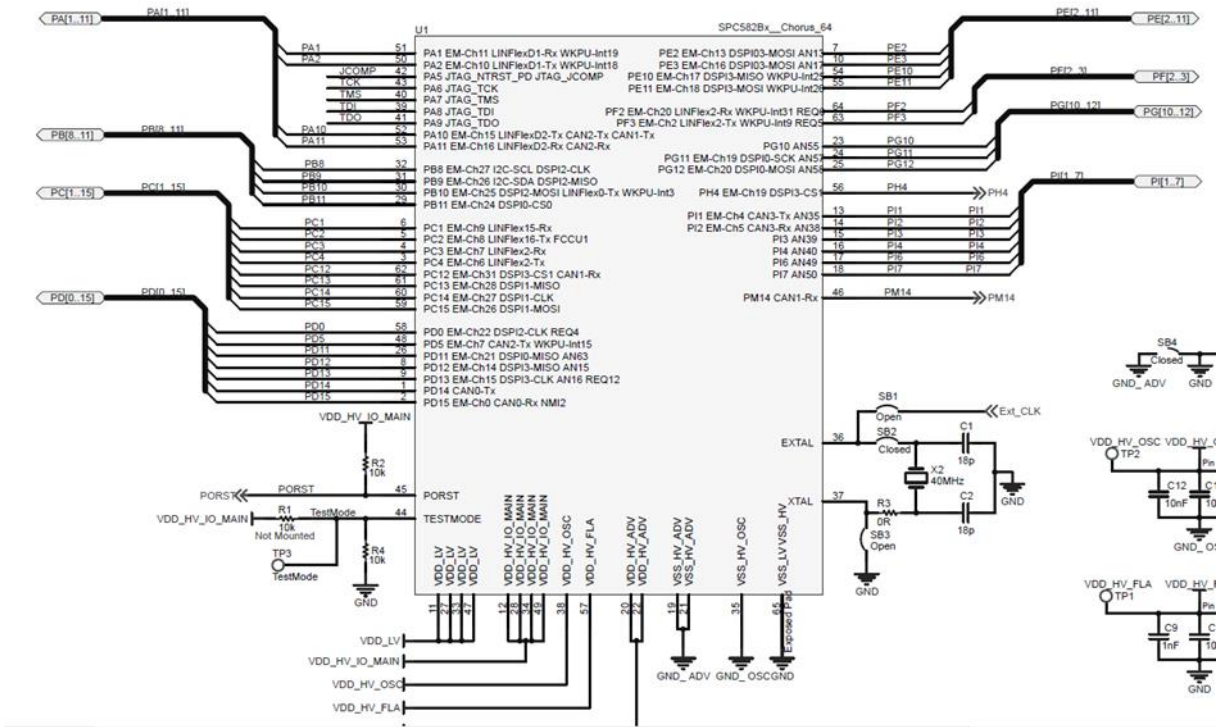
Tab.2 periferie startkitu AEK-MCU-C1MLIT1

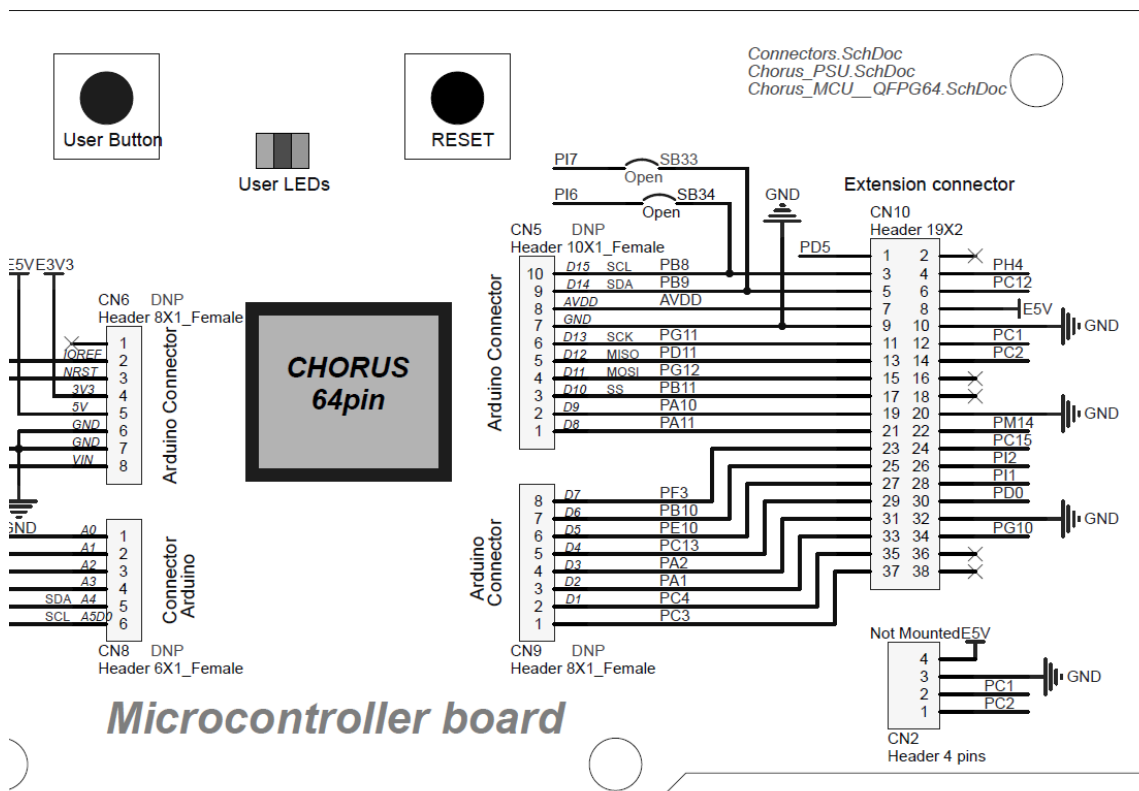
Popř. také obrázek namapování pinů MCU:



Zdrojem k vytvoření této tabulky tab.2 bylo schéma startkitu (viz následující tři obrázky):

Figure 1. AEK-MCU-C1MLIT1 circuit schematic (1 of 4)





Po tomto úvodním projektu si odzkoušíme postupy při programování předvedené v osmi kapitolách **AutoDevKit™ detailed tutorial** [21] až [28].

6. Závěr

Pokud se Vám podařilo naprogramovat aplikaci pro MCU SPC58, je cíl této úvodní práce splněn. Teď by již pro Vás neměl být problém vytvářet alespoň jednoduché vlastní aplikace.

Literatura:

[1] Váňa Vladimír: Práce se startkity AutoDevKit – úvod, moodle kurz CIT_programování ARM a SPC58 jednočipových počítačů C4abc, SPŠE Ječná 2022

[2] Souoush Arab: ALGORITHM ANALYSIS FOR AN AUTOMOTIVE ADAPTIVE FRONT LIGHT SYSTEMS, Master of Science Thesis in Electronic Engineering (Embedded Systems) , POLYTECHNIC OF TURIN The Department of Electronics and Telecommunications (DET), 2020

[3] Emanuele Di Miceli: Implementation of cost effective solutions for testing in automotive environment, LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA, UNIVERSITÀ DEGLI STUDI DI PALERMO FACOLTÀ DI INGEGNERIA, 2020

[4] PATRIK PIHRT: APLIKACE PRO EXTRAKCI A ANALÝZU JÍZDNÍCH DAT Z OBD-II NA IOS, diplomová práce VUT Brno, 2021

[5] ONDŘEJ HÁJEK: DIAGNOSTIKA DAT V MODERNÍCH ŘÍDÍCÍCH JEDNOTKÁCH MOTORU, diplomová práce VUT Brno, 2010

- [6] Barbora Chmelíková: Diagnostické systémy OBD, bakalářská práce ZÁPADOČESKÁ UNIVERZITA V PLZNI FAKULTA ELEKTROTECHNICKÁ, 2013
- [7] Jan Berg: Chiptuning a kontrola namáhání klikového ústrojí vozu Volkswagen CC, bakalářská práce DOPRAVNÍ FAKULTA UNIVERZITA PARDUBICE, 2017
- [8] Martin Ložek: Konektivita vozů ŠKODA AUTOa.s., bakalářská práce Technická univerzita Liberec, 2021
- [9] Jiří Očenášek: Návrh software pro jednotku řízení akumulátorů (BMS), diplomová práce, ZČU Plzeň, fak. Elektrotechnická, 2020
- [10] Tomáš Zimmerhakl: Sledování stavu vozidla prostřednictvím mobilní aplikace, diplomová práce, ČVUT FIT Praha, 2019
- [11] JAROSLAV FADRŇÝ: BEZDRÁTOVÝ SBĚR DIAGNOSTICKÝCH DAT Z AUTOMOBILŮ PODPORUJÍCÍCH OBD-II, diplomová práce, VÚT Brno, 2014
- [12] Michal Hajda: Diagnostické nástroje motorových vozidel a jejich bezpečnostní funkce, bakalářská práce, Univerzita Tomáše Bati ve Zlíně, 2018
- [13] Jan Hrdina: Diagnostika automobilu, bakalářská práce, Jihočeská univerzita v Českých Budějovicích, 2014
- [14] Petr Karafiát: Interní diagnostické nástroje osobního automobilu, jejich aplikace při zkoušení, diplomová práce, Mendelova zemědělská a lesnická univerzita v Brně, Agronomická fakulta 2009
- [15] Petr Beneš: Programování řídicích jednotek automobilů, diplomová práce, Bankovní institut vysoká škola Praha, Katedra matematiky, statistiky a informačních technologií, 2011
- [16] Lukáš Karaffa: DIAGNOSTIKA V AUTOMOBILOCH, BAKALÁRSKA PRÁCA, SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVĚ FAKULTA ELEKTROTECHNIKY A INFORMATIKY 2010
- [17] <https://www.alza.cz/mobilly-obd-ii-bt-d4624328.htm>
- [18] <https://www.alza.cz/mobilly-usb-vag-obd-ii-kabel-d4633878.htm>
- [19] https://www.seznamzpravy.cz/clanek/ekonomika-firmy-byznys-zachrana-podnikani-firem-cesi-v-ohrozeni-pokud-se-autoland-nezmeni-nema-sanci-prezit-228876#dop_ab_variant=909531&dop_source_zone_name=hpfeed.sznhp.box&utm_source=www.seznam.cz&utm_medium=sekce-z-internetu
- [20] AutoDevKit™ a new development approach to Automotive & Transportation applications, STMicroelectronic
https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwjL7pD9zZ - AhURhv0HHbITAP4QFnoECBEQAQ&url=https%3A%2F%2Fwww.st.com%2Fresource%2Fen%2Fbrochure%2Fbrautodevkit0220.pdf&usg=AOvVaw3DipHeJFoZuK8j9HP0_QJb

- [21] Part 1 - AutoDevKit™ detailed tutorial: how to install
<https://www.youtube.com/watch?v=WDp4XmmwEwc&t=1s>
- [22] Part 2 - AutoDevKit™ detailed tutorial: creating your project application
<https://www.youtube.com/watch?v=pQeyJQBHV1c&t=6s>
- [23] Part 3 - AutoDevKit™ detailed tutorial: how to add a component in the SPC5Studio application
<https://www.youtube.com/watch?v=3iCMWjYalwg>
- [24] Part 4 - AutoDevKit™ detailed tutorial: Pinmap Editor
<https://www.youtube.com/watch?v=UvLChQnFWS8&t=2s>
- [25] Part 5 - AutoDevKit™ detailed tutorial: Board View
<https://www.youtube.com/watch?v=EJelkEq-1gQ>
- [26] Part 6 - AutoDevKit™ detailed tutorial: APIs <https://www.youtube.com/watch?v=ky3-y3QEzMO>
- [27] Part 7 - AutoDevKit™ detailed tutorial: Debug and release of firmware created with AutoDevKit™
https://www.youtube.com/watch?v=4Vn_cRUfRag
- [28] Part 8 - AutoDevKit™ detailed tutorial: user support
<https://www.youtube.com/watch?v=Nrjq0pS2d-c>
- [29] <https://www.nxp.com/company/about-nxp/history:NXP-HISTORY>
- [30] <https://www.digitimes.com/news/a20060207PR207.html?mod=3&q=semiconductor>
- [31] <https://www.electronicweekly.com/news/products/micros/freescale-st-develop-dual-core-mcu-for-safety-critical-automotive-apps-2009-10/>
- [32] <https://www.isystem.com/chip-search/nxp-mcu-overview/mpc5xxx-spc5xxx-family.html>
- [33] <https://vyvoj.hw.cz/anatomie-bezpecnosti-v-mikrokontroleru-pro-aplikace-internetu-veci.html>