SoCkit – instalace Linuxu

Vladislav Mlejnecký

5. června 2016

Obsah

Seznam obrázků

1 Nastavení Kitu

Na kitu se nachází několik přepínačů a jumperů které je potřeba před zahájením práce mít správně nastavené. Jmenovitě se jedná o přepínač SW6, SW4 a jumpery J15 až J19.

Jumpery J15 a J16 ovládají nastavení hodin, je potřeba mít u obou propojené piny 2 a 3. Jumpery J17 až J19 vybírají umístění ze kterého bude bootovat hardwarový procesor. My budeme bootovat z Micro SD karty a této volbě odpovídá konfigurace 101.

Přepínač SW4 ovládá rozhraní JTAG. Pro nás je potřeba nastavit SW4.1 na zapnuto a SW4.2 na vypnuto. Tímto zapneme JTAG pro hardwarový procesor a odpojíme JTAG na HSMC. Poslední přepínač SW6 vybírá mód práce FPGA. Nastavme výchozí mód ASx4 volbou 10010.

Tato uvedená konfigurace odpovídá výchozí konfiguraci kitu. Bližší informace hledejte v manuálu kitu.

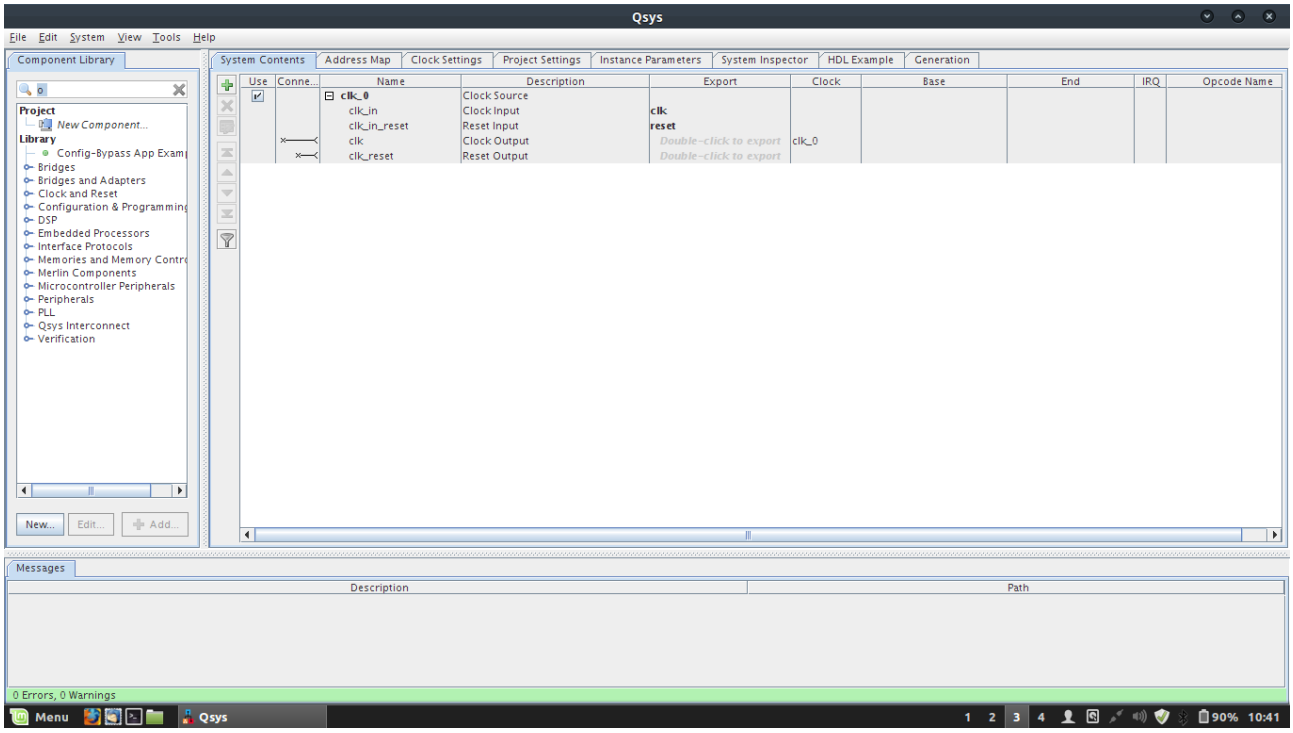
2 Konfigurace FPGA

2.1 Vytvoření projektu

Dalším krokem je konfigurace FPGA. K tomuto potřebujeme využít programu dodávaného firmou Altera s názvem Quartus. Začneme tím že vytvoříme nový projekt, vybereme správné FPGA použito v SoCkitu, tedy 5CSXF6D6F31C8ES. Následně je vhodné provést obvyklá nastavení projektu, nepoužité piny nakonfigurovat jako vstupy a piny s dvojím užitím (dual-purpose pins) nastavit jako regulérní vstupy/výstupy. Následně ještě Block Design Naming nastavit na Quartus II.

2.2 Konfigurace HPS

FPGA použité v SoCkitu obsahuje hardwarový procesorový systém na kterém Linux spustíme. Tento hardwarový procesor je doplněn 1GB paměti DDR3 což je pro potřeby Linuxu více než dostatečné.



Obrázek 1: Hlavní okno průvodce Qsys.

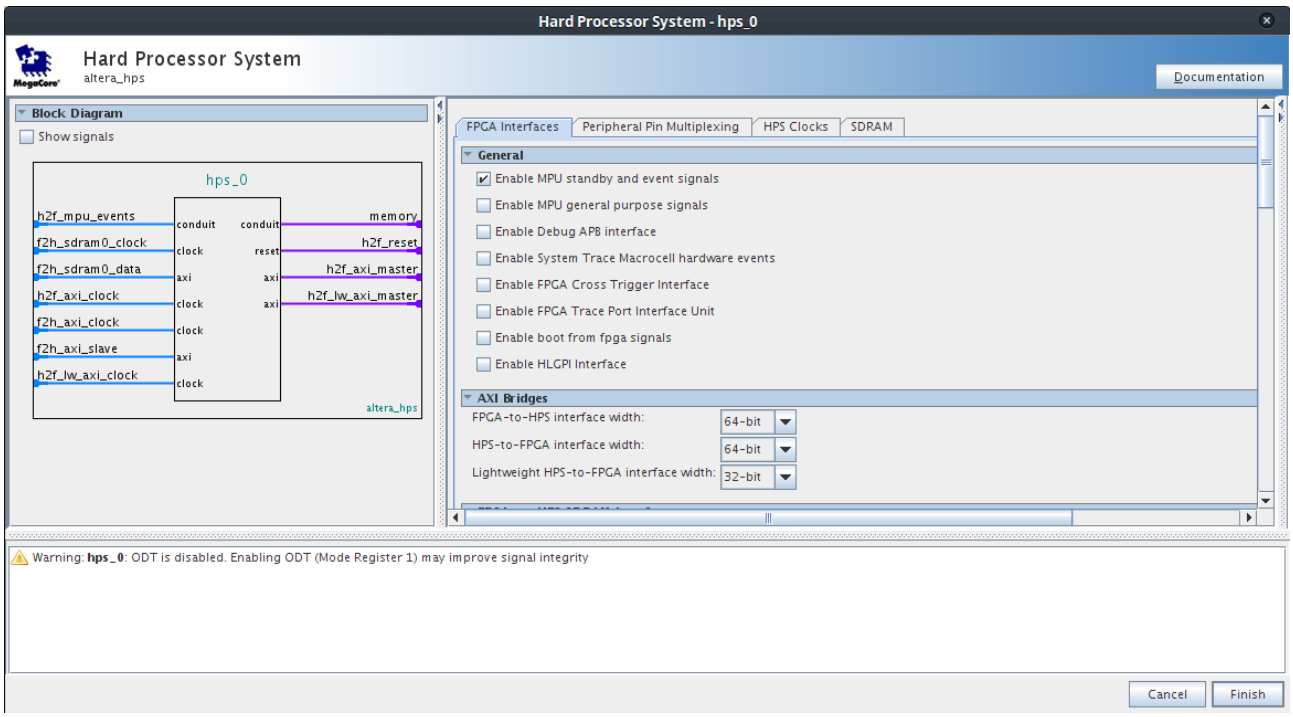
Pro konfiguraci hardwarového procesoru nabízí Quartus průvodce, který práci velice usnadňuje a zpřehledňuje. Tento průvodce se nazývá Qsys a je ho možné spustit z menu Tools. Obrázek číslo 1 zachycuje hlavní okno programu Qsys.

Ve výchozím stavu máme již vloženou komponentu clk_0 která se stará o distribuci hodin a generování signálu reset.

Dále přidáme komponentu hardwarového procesoru. To provedeme kliknutím do levého sloupce, tedy do knihovny komponent. V záložce Embedded processors nalistujeme položku Hard Procesor System a dvakrát na ni klikneme.

Po krátké chvíli se ukáže průvodce vložením komponenty kde procesorový systém nakonfigurujeme podle našich požadavků. Úvodní okno průvodce pro hardwarový procesor je na obrázku číslo 2.

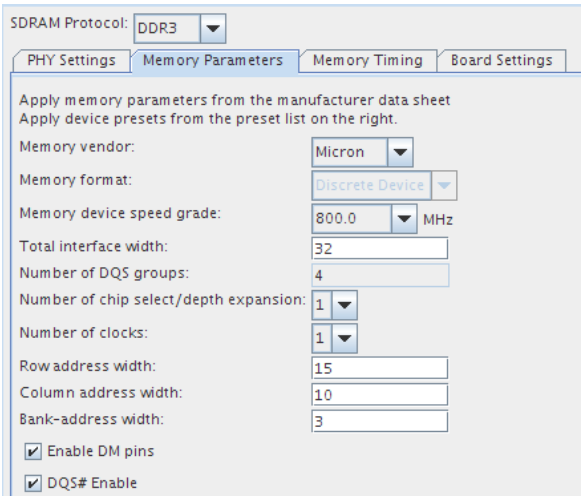
Pro nejjednodušší konfiguraci budeme postupovat následovně. V záložce FPGA interfaces, v oddílu General, zrušíme označení položky „Enable MPU standby and event signals“. V oddílu AXI Bridges nastavíme FPGA-TO-HPS interface a HPS-TO-FPGA interface na unused. Tyto rozhraní jsou určeny pro připojení složitějších obvodů a pro začátek nebudou použity. Pouze se ujistíme že lightweight HPS-TO-FPGA interface má nastavenou šíři 32bitů. Toto rozhraní, pro jeho jednoduchost, použijeme. V dalším oddílu, FPGA-to-HPS SDRAM interface se ujistíme že nic není. V oddílu Resets by též nemělo být nic povolené, stejně tak oddíl DMA peripheral request i oddíl Interrupts.



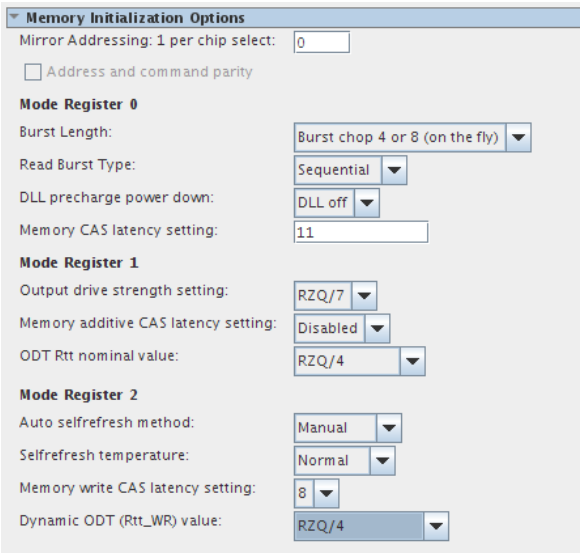
Obrázek 2: Okno pro nastavení komponenty HPS.

Nyní se přepněme do záložky Peripherial Pin Multiplexing, kde to již začne býti zajímavější. Zde máme možnost nastavit jednotlivá rozhraní hardwarového procesoru. Pro naše potřeby bude bohatě stačit SDIO a UART, ovšem praktický může být i Ethernet. Proto v oddílu Ethernet Media Access Controller zvolíme EMAC1 pin multiplexing na HPS I/O Set 0 a EMAC1 mode na RGMII. Následně v oddílu SDMMC/SDIO Controller nastavíme „HPS I/O set 0“ a „4-bit Data“. Následně ještě v oddílu UART Controllers nastavíme UART0 na „HPS I/O set 2“ a „no flow control“.

V záložce HPS Clocks by nemělo být třeba nic měnit. Kdežto v záložce SDRAM musíme nastavit několik dalších věcí. V této záložce se konfiguruje rozhraní pro SDRAM. Tyto údaje se vyplní podle použité SDRAM paměti v kitu. Protokol se proto nastaví na DDR3, frekvence paměti se nastaví na 400MHz a referenční frekvence pro PLL na 25MHz. V podzáložce Memory Parameters je potřeba vyplnit poměrně velké množství údajů, správné údaje jsou na obrázcích 3 a číslo 4.

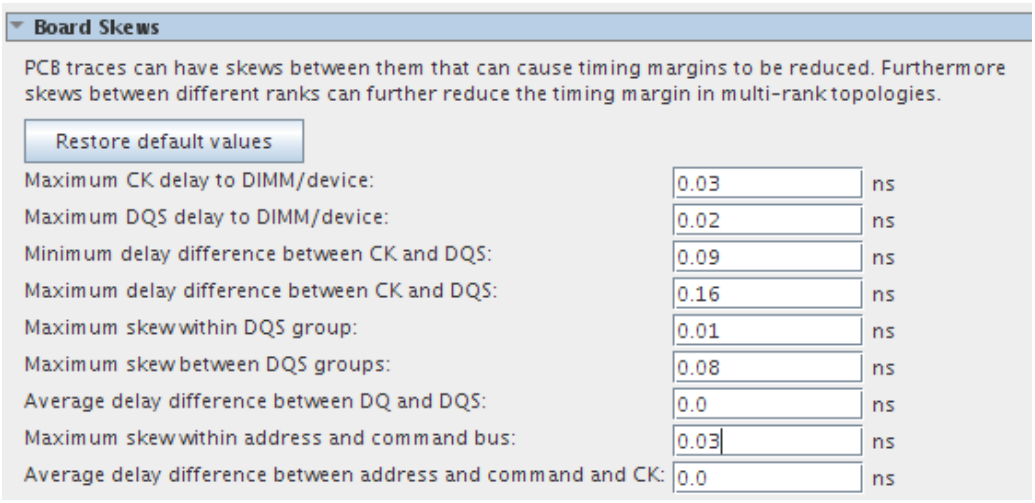


Obrázek 3: Parametry pro SDRAM první část.



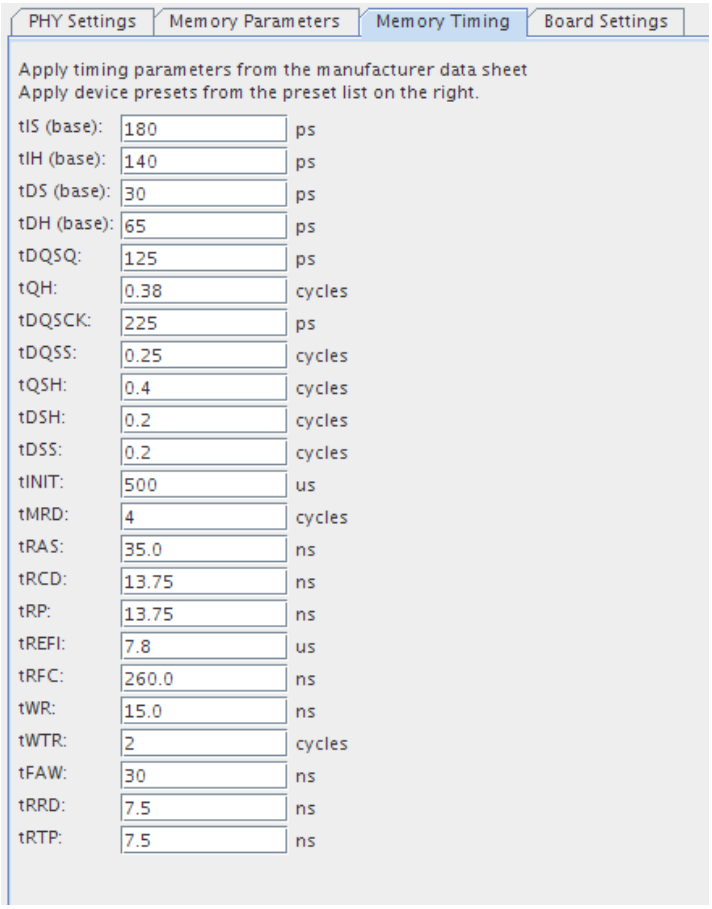
Obrázek 4: Parametry pro SDRAM druhá část.

V pod-záložce Memory Timing se nastavují parametry použité SDRAM paměti, tyto hodnoty se



Obrázek 6: Nastavení board skews.

nastaví podle datasheetu. Pro SoCkit jsou vhodné hodnoty zachyceny na obrázku číslo 5.



Obrázek 5: Nastavení časování paměti.

V poslední pod-záložce board settings je opět potřeba nastavit několik konstant. Pro SoCkit se v oddílu Setup and Hold Derating použije výchozí nastavení, stejně tak v oddílu intersymbol interface. Posledním oddílem je Board Skews který se pro SoCkit nastaví podle obrázku číslo 6.

Pokud máme veškeré nastavení hotové stačí kliknout na tlačítko finish a komponenta se vloží do systému.

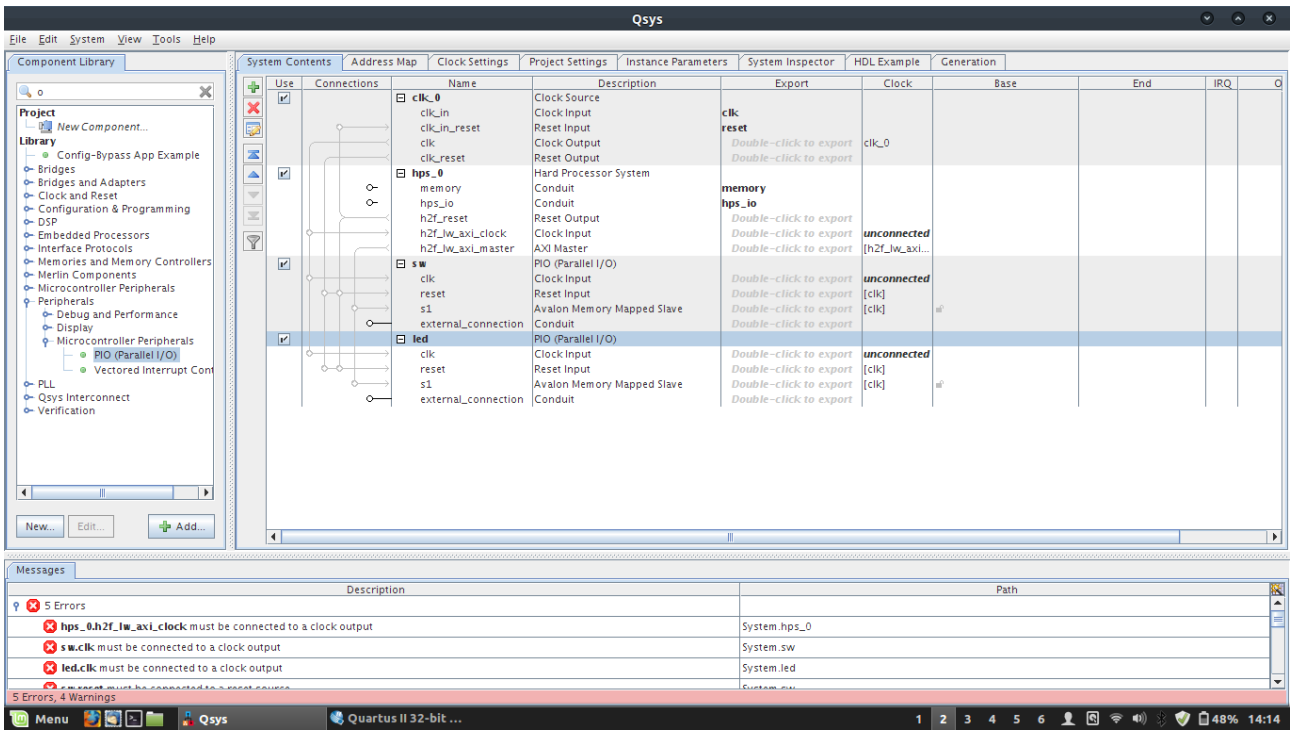
Obdobným způsobem vložíme do systému i další dvě komponenty, dva paralelní porty, na které připojíme LED diody a tlačítka s přepínači. Jeden by tedy měl být vstupní a druhý výstupní.

Pro paralelní porty slouží komponenta PIO kterou najdeme v levém sloupci v knihovně, ve složce Peripherials a následně Microcontroller peripherials. Tyto komponenty nepotřebují bližší popis. Šířka pro vstupní i výstupní port by měla být 8 bitů. Žádné přerušení ani další možnosti nepotřebujeme.

Po vložení paralelních portů je vhodné je přejmenovat pro větší přehlednost. První z nich, výstupní pojmenujeme kupříkladu led a vstupní třeba sw.

Obrázek číslo 7 zachycuje Qsys po vložení komponenty hardwarového procesoru a paralelních portů. Vidíme že krom bloku clk_0 je zde i blok hps_0 což je právě náš procesor. Nyní potřebujeme jednotlivé komponenty propojit. Jak již bylo letmo zmíněno, pro připojení komponent použijeme lightweight sběrnici. V hlavním okně Qsys tato sběrnice vystupuje jako h2f_lw_axi. Dále připojíme hodinový signál a signál reset.

Propojení uvnitř systému provedeme ve sloupci Connections, klikáním na jednotlivé uzly propoju-



Obrázek 7: Qsys po vložení všech komponent systému.

jeme signály. Hodinový výstup z bloku clk_0 tedy přivedeme na všechny komponenty. Signál reset na paralelní porty přivedeme z bloku hps_0. Stejně tak oba tyto porty připojíme na sběrnici h2f_ls_axi.

Další důležitou částí je export signálů ven ze systému. Toto se provede ve sloupci Export. Důležité je vyexportovat signály clk z bloku clk_0, memory a hps_io z bloku hps_0. Protože ale chceme použít i paralelní porty tak jejich externí rozhraní taktéž musíme exportovat, dvojím kliknutím ve sloupci Export a v řádku external conection nám to umožní. Je vhodné exportované signály opět dobře pojmenovat.

Protože na sběrnici h2f_lw_axi může být připojeno více slave uzlů, je potřeba každému nastavit jeho rozsah adres. To lze provést ve sloupci Base. Nastavme adresu portu pro diody LED na 0x000 0010.

Nakonfigurovaný systém je zachycen na obrázku číslo 8.

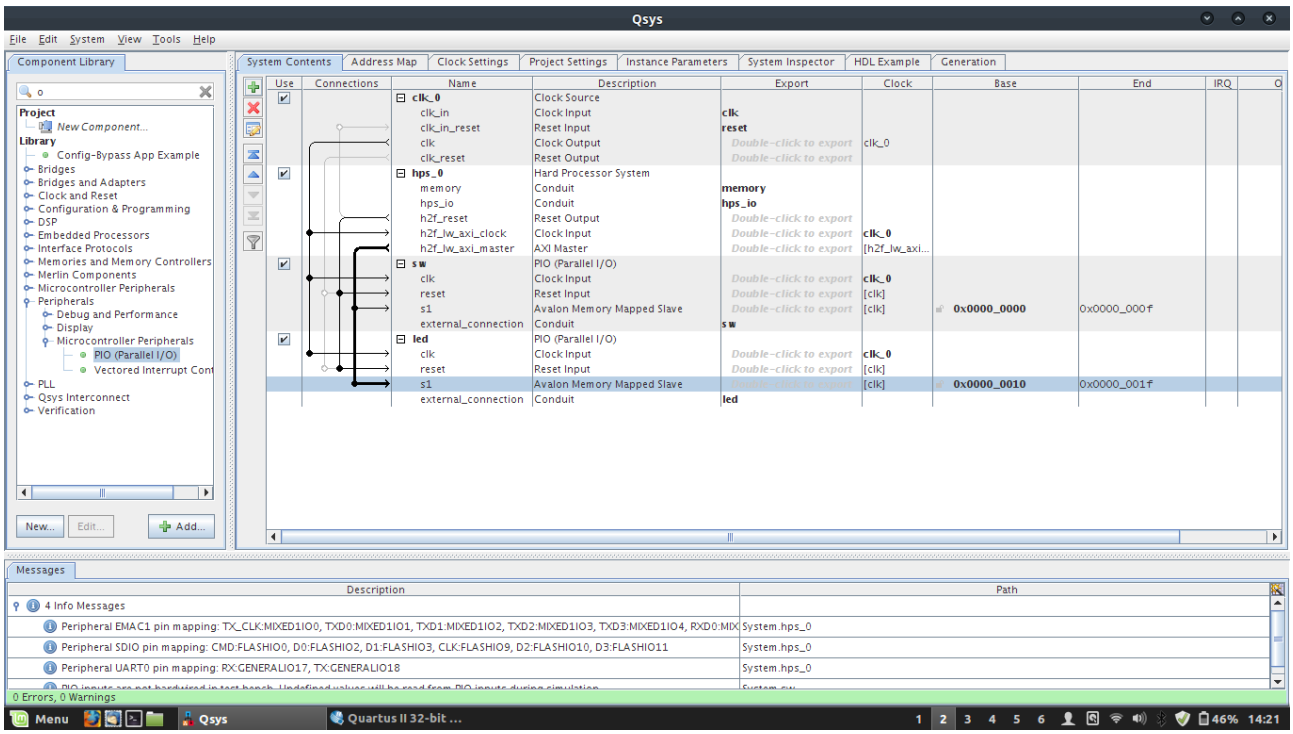
Nyní již můžeme přistoupit ke generování komponenty. Vybereme poslední záložku s názvem Generation ve které nastavíme jazyk pro syntézu VHDL a odškrtneme vytváření souboru bdf. Uložíme projekt kupříkladu pomocí klávesové zkratky CTRL+S a klikneme na tlačítko Generate.

Generování může trvat delší dobu v závislosti na HW vašeho počítače. Po ukončení generování přepněte na záložku HDL Example, jazyk zvolte VHDL a zkopírujte celý vygenerovaný kód. Tento kód využijeme až budeme vytvářet hlavní entitu, je to v podstatě komponenta právě vytvořeného systému.

2.3 Top level entity

Dalším krokem v návrhu je vytvoření hlavní entity. V prostředí Quartus klikneme na nabídku File, následně vybereme New a zvolíme VHDL File. Otevře se nám okno editoru do kterého umístíme náš kód. Vložíme kód zkopírovaný v předchozí části. Poté je potřeba vložit do projektu propojení na patřičný soubor qip. Klepnutím na menu Project, Add/Remove Files in Project se nám otevře okno kde vybere soubor qip který bude uložen v adresáři se jménem systému pod kterým jsme uložili výstup Qsysu a dále ve složce synthesis. Klepneme na add a apply.

Nyní nás čeká asi nejhorší část práce, je potřeba kód hlavní entity upravit a připojit signály na správná místa. Naše entita bude mít stejné rozhraní jako má komponenta z Qsysu, plus navíc bude mít hodinový vstup 50MHz, vstupy pro tlačítka. přepínače a výstupy pro led diody.



Obrázek 8: Nakonfigurovaný systém pro hardwarový procesor.

Zdrojový kód hlavní entity je možno najít v `https://github.com/VladisM/sockit_linux/tree/manual`. Vývody entity pro jednotlivá rozhraní HPS jako jsou ethernet, paměť a podobně se budou jmenovat stejně jako je vyexportoval Qsys.

Posledním krokem je připojení pinů k vývodům entity. Z části nám pomůže Quartus, spustíme analýzu a syntézu, poté otevřeme menu Tools a TLC Scripts. V tomto okně spustíme script "hps sdram p0 pin assigments.tcl"který automaticky přiřadí piny pro HPS. Zbytek pinů, tedy LED diody, tlačítka, přepínače a hodinové signály musíme přiřadit ručně podle schématu kitu a nebo podle návodu.

2.4 Kompilace a nakonfigurování FPGA

Nyní již můžeme celý projekt zkompilovat. Toto opět může trvat déle v závislosti na hardwarové výbavě vašeho počítače. Po úspěšné kompilaci můžeme přistoupit k nahrání konfigurace do kitu. V menu Tools otevřeme okno programátoru příkazem Programmer. Vybereme patřičný soubor .sof a nahrajeme. Tímto je SoCkit připravený bootovat, stačí vložit paměťovou kartu.

3 Příprava paměťové karty

Pro spuštění Linuxu na SoCkitu budeme potřebovat paměťovou kartu, postačí jakákoliv která je po ruce, s kapacitou alespoň 2GB. Ovšem je vhodné pokud se použije paměťová karta rychlejší. Pomalá paměťová karta se nepříjemně promítne do odezvy celého operačního systému.

3.1 Použití předpřipravené distribuce

Jako nejjednodušší možnost se jeví použití předpřipravené distribuce, která je na přiloženém CD ke kitu, případně ji lze stáhnout ze stránek firmy Terasic.

Po stažení archivu "SoCkit_SD.rar"s obrazem disku, nebo jeho vyhledání na CD jej rozbalme do vhodného adresáře, kupříkladu příkazem unrar.

```
$ unrar e ./SoCKit_SD.rar
```

Rozbalený soubor je poté snadné naklonovat na paměťovou kartu pomocí příkazu dd. Nejprve je ale vhodné ujistit se, že přesně známe název zařízení do kterého chceme soubor naklonovat. K tomuto je velice užitečný příkaz lsblk. Následující příklad předpokládá že zařízení na které se bude instalovat je /dev/mmcblk0.

```
$ lsblk
$ sudo dd if=./SoCKit.img of=/dev/mmcblk0
```

Nyní máme velice holý ale funkční Linux na paměťové kartě, byli bychom schopni nabootovat ale nejspíše by nám nefungovala ani síť. Je nanejvýš vhodné nahradit tuto distribuci nějakou lepší, například distribucí Debian.

3.2 Nahrazení předpřipravené distribuce distribucí Debian

Pro nahrazení distribuce prakticky stačí změnit rootfs na paměťové kartě. Kromě rootfs je na kartě ještě několik věcí, jmenovitě je to jádro Linuxu, dále device tree a zavaděč. Toto vše může zůstat stejné.

Nový rootfs budeme generovat přímo nám na míru, budeme si jej takto tedy moci snadno uzpůsobit k obrazu svému. Ke generování použijeme nástroj debootstrap.

Prvním krokem je instalace potřebných programů. Budeme potřebovat hlavně debootstrap a qemu. Jednoduše je nainstalujeme pomocí.

```
$ sudo apt-get install qemu-user-static debootstrap binfmt-support
```

Druhým krokem je identifikace oddílu na paměťové kartě kde sídlí starý rootfs. Kartu připojíme, prohledáme a zjistíme přípojný bod oddílu s rootfs. Rootfs oddíl poznáme podle toho že bude obsahovat složky jako bin, etc, opt, home a podobně. Cestu k tomuto oddílu budeme potřebovat. Je nutné tento oddíl naformátovat a znovu připojit. Na takto připravený oddíl již můžeme vygenerovat vlastní rootfs.

Pro snadnější a přehlednější práci si nastavíme pár proměnných. Předpokládejme že oddíl s rootfs se připojil do /media/rootfs.

```
$ rootdir=/media/rootfs
$ distro=wheezy
```

Spustíme první krok generování rootfs. Poté nakopírujeme některé důležité soubory pro chod qemu a následně provedeme chroot do cílového rootfs. Stačí postupně zadat následující příkazy.

```
$ sudo debootstrap --arch=armhf --foreign $distro $rootdir
$ sudo cp /usr/bin/qemu-arm-static $rootdir/usr/bin/
$ sudo cp /etc/resolv.conf $rootdir/etc
$ sudo chroot $rootdir
```

Nyní se nacházíme v prostředí našeho rootfs. Tedy, ještě ne zcela vygenerovaného. Nastavíme opět pár proměnných pro přehlednost a spustíme druhý krok v generování rootfs.

```
$ distro=wheezy
$ export LANG=C
$ /debootstrap/debootstrap --second-stage
```

Dalším rozumným krokem je konfigurace programu apt. Tedy, přidání zrcadel pro stahování programů.

```
$ cat <<EOT > /etc/apt/sources.list
deb http://ftp.uk.debian.org/debian $distro main contrib non-free
deb-src http://ftp.uk.debian.org/debian $distro main contrib non-free
deb http://ftp.uk.debian.org/debian $distro-updates main contrib non-free
deb-src http://ftp.uk.debian.org/debian $distro-updates main contrib non-
    free
deb http://security.debian.org/debian-security $distro/updates main
    contrib non-free
deb-src http://security.debian.org/debian-security $distro/updates main
    contrib non-free
EOT
```

Nyní máme přidány zrcadla a jen aktualizujeme databázi s balíčky. Můžeme zrovna i aktualizovat.

```
$ apt-get update
$ apt-get upgrade
```

Velice užitečné by bylo nainstalovat i několik balíčků. SSH server je velice praktický. Locales je též vhodný, ten ale musíme i nakonfigurovat.

```
$ apt-get install locales dialog openssh-server ntpdate
$ dpkg-reconfigure locales
```

Samozřejmě nesmíme zapomenout nastavit heslo pro uživatele root.

```
$ passwd
```

A když už máme nainstalovaný a k FPGA připojený ethernet, tak nakonfigurujeme i toto rozhraní. Takto aktivujeme hotplug a dhcp na rozhraní eth0.

```
$ echo <<EOT >> /etc/network/interfaces
auto eth0
allow-hotplug eth0
iface eth0 inet dhcp
EOT
```

Dále nastavíme hostname našeho systému. Použijeme například jméno sockit. A taktéž aktivujeme konzoli pro ovládání Linuxu přes sériový port.

```
$ echo sockit > /etc/hostname
$ echo T0:2345:respawn:/sbin/getty -L ttyS0 57600 vt100 >> /etc/inittab
```

A tímto jsme hotovi. Stačí již jen opustit prostředí chrootu, smazat pomocné soubory a odpojit paměťovou kartu.

```
$ exit
$ sudo rm $rootdir/etc/resolv.conf
$ sudo rm $rootdir/usr/bin/qemu-arm-static
$ sync
$ sudo umount $rootdir
```

3.3 Testovací program

Pro otestování funkčnosti systému použijeme jednoduchý program který párkrát blikne LED diody na kitu. Tento program bude napsán v jazyce C. Vzorové programy lze opět dohledat na adrese `https://github.com/VladisM/sockit_linux/tree/manual`.

Ze všeho nejdříve je nutné nainstalovat překladač. Ve většině linuxových distribucí je vhodný překladač již v oficiálních repositářích a stačí tedy provést jednoduchý příkaz.

```
$ sudo apt-get install gcc-arm-linux-gnueabihf
```

Po instalaci překladače můžeme přistoupit k napsání kódu pro náš program. Program bude jednoduchý, pouze přistoupí k paměťovému místu na kterém je umístěn výstupní registr pro LED diody a zapíše do něj vhodné hodnoty.

Architektura ARM má adresní prostor 2^{32}. Přičemž periferie jsou mapovány do společného prostoru s pamětí. Můžeme si tedy dovolit udělat ukazatel na určité adresové místo které obsahuje náš registr a jednoduše do něj zapsat. Ovšem, musíme vzít v úvahu Linuxové jádro, které má správu paměti a jednotlivým procesům přiřazuje určitý paměťový rozsah ve kterém mohou pracovat. Pokud bychom se pokusili zapsat mimo tento rozsah, program by skončil s chybou. Linuxové jádro musíme požádat o přidělení příslušné části paměti, která obsahuje náš registr, našemu procesu.

Náš registr jsme připojili na sběrnici h2f_lw_axi. Tato sběrnice začíná na adrese 0xFF200000. Offset registru je 0x00000010. Tedy výsledná adresa, kde budeme registr hledat, je 0xFF200010.

Následující ukázka kódu je založena na článku `https://zhehaomao.com/blog/fpga/2013/12/27/sockit-3.html`. Ukázka je dostatečně sebe-vysvětlující.

```
#include <sys/mman.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <stdint.h>

#define PAGE_SIZE 4096
#define LWHPS2FPGA_BRIDGE_BASE 0xff200000
#define LED_OFFSET 0x10

volatile unsigned char *leds_register;
void *bridge_map;

int main(int argc, char *argv[]){

        off_t BASE = LWHPS2FPGA_BRIDGE_BASE;

        // open the memory device file
        int fd = open("/dev/mem", O_RDWR|O_SYNC);
        if (fd < 0) {
                perror("open");
                exit(EXIT_FAILURE);
        }

        // map the LWHPS2FPGA bridge into process memory
        bridge_map = mmap(NULL, PAGE_SIZE, PROT_WRITE, MAP_SHARED, fd,
            BASE);
        if (bridge_map == MAP_FAILED) {
                perror("mmap");
```

```
                close(fd);
                exit(EXIT_FAILURE);
        }

        // get the leds address
        leds_register = (unsigned char *) (bridge_map + LED_OFFSET);

        // blink with leds
        *leds_register = 0x00;
        sleep(1);
        *leds_register = 0x0A;
        sleep(1);
        *leds_register = 0x05;
        sleep(1);
        *leds_register = 0x00;

        // clean mapped memory
        if (munmap(bridge_map, PAGE_SIZE) < 0) {
                perror("munmap");
                close(fd);
                exit(EXIT_FAILURE);
        }

        // normall exit
        close(fd);
        return EXIT_SUCCESS;
}
```

Tímto způsobem jsme schopni jednoduše číst i zapisovat data z registrů. Pokud bychom data chtěli i číst musíme jen argument PROT_WRITE funkce mmap, přepsat na PROT_READ.

Program můžeme přeložit pomocí make, což je elegantní cesta, stačí vytvořit ve složce se zdrojovým kódem, soubor Makefile a vložit do něj následující kód.

```
CC=arm-linux-gnueabihf-gcc
CFLAGS=-Wall -O2
OUT=led_test

all: main.c
        $(CC) $(CFLAGS) $(LDFLAGS) $< -o $(OUT)

clean:
        rm -f $(OUT)
```

A poté již jen spustit příkaz pro překlad.

```
$ make
```

Jiná možnost je nevytvářet Makefile ale spustit překladač přímo. Pro takto malý projekt to není nic složitého. Následující příkaz právě toto provede.

```
$ arm-linux-gnueabihf-gcc -Wall -02 main.c -o led_test
```

Přeložený program poté stačí již jen nahrát na paměťovou kartu, do oddílu s rootfs, do nějakého vhodného adresáře. Kupříkladu do adresáře home.

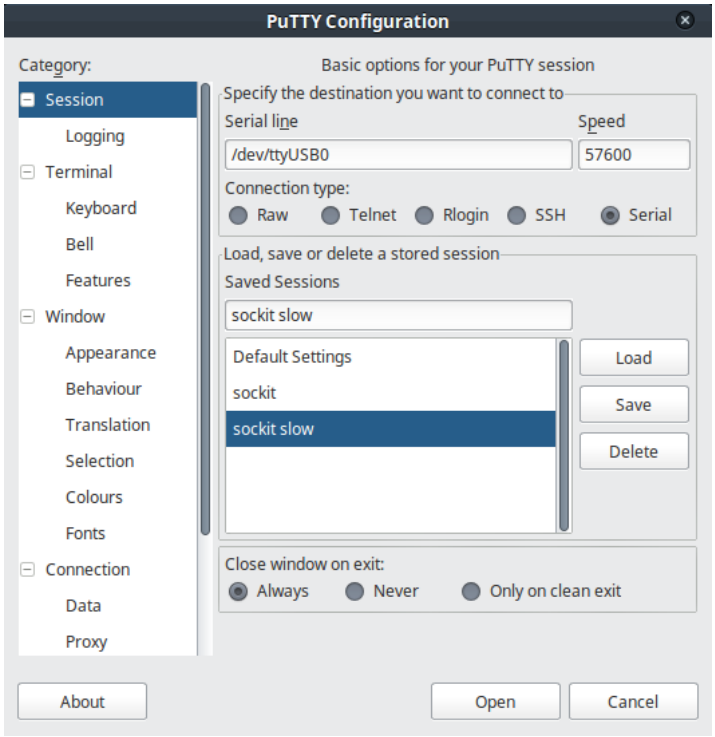
4 Spuštění Linuxu

4.1 První boot

Pokud máme již připravenou paměťovou kartu, můžeme přejít k prvnímu bootu. Začneme tím že připojíme SoCkit ke zdroji napájení. Pomocí kabelů USB A - USB micro B propojíme počítač s USB Blasterem a USB-UART převodníkem, případně můžeme připojit i ethernet.

Dalším krokem je vložení paměťové karty do slotu na spodní straně kitu. Po vložení zapneme program Putty abychom mohli komunikovat s Linuxem. Pro nastavení spojení v programu Putty potřebujeme znát minimálně dvě věci. Rychlost komunikace rozhraní na kterém se kit připojil. Rychlost komunikace jsme nastavili při generování systému na 57600 baudů a rozhraní, na kterém je kit připojen, můžeme zjistit pomocí následujícího příkazu.

```
$ dmesg | grep tty
```



Obrázek 9: Ukázka nastavení programu putty.

Ukázkové nastavení komunikace v programu Putty můžeme vidět na obrázku číslo 9. Na obrázku je port pro komunikaci nastaven na /dev/ttyUSB0 a rychlost 57600 baudů.

Zmáčkneme tlačítko číslo 6 kterým resetujeme HPS a měli bychom vidět proces bootování. Po nabootování se ocitneme v bodě pro přihlášení. Přihlasme se jako root s heslem které jsme navolili při tvorbě rootfs.

Právě máme zprovozněný Linux běžící na HPS. Můžeme klasicky spouštět programy a ovládat je jak jsme zvyklí.

4.2 Test diod LED

Pokud jsme postupovali přesně podle kapitoly 3.3 měli bychom mít v adresáři /home program led_test. Tento program tedy zkusíme spustit a pokud bude vše fungovat správně, měli bychom vidět krátké probliknutí LED diod. Program tedy spustíme následovně.

```
$ chmod +x ./led_test
$ ./led_test
```

První řádek, s příkazem chmod, přidává našemu programu práva pro spuštění. Toto je standardní bezpečnostní opatření v Linuxu. Druhým řádkem už jen program zapínáme.

4.3 Připojení pomocí SSH

Ovládání přes sériový port je nepohodlné. Velice příjemnější možností je ovládání přes internet pomocí SSH. Za tímto účelem jsme v kapitole 3.2 nainstalovali SSH server. SSH server se spouští automaticky a funguje už ve výchozí konfiguraci. Vezměte ale prosím na vědomí že tento návod se nezabývá zabezpečením komunikace a SSH serveru, proto je vhodné tento postup aplikovat pouze na soukromé a zabezpečené síti.

Abychom byli schopni se připojit ke kitu pomocí protokolu SSH, musíme jej nejprve připojit do sítě a zjistit jeho IP adresu. Kabelem připojme kit do sítě. Pro jednoduchost budeme předpokládat že

tato síť je typu LAN. Pokud nám na síti běž DHCP server, měl by kit obdržet IP adresu automaticky, pokud DHCP server na síti není, je nutné konfiguraci provést ručně v souboru /etc/network/interfaces.

IP adresu kterou kit obdržel je možno zjistit pomocí následujícího příkazu.

```
$ ifconfig
```

Řekněme že IP adresa kterou kit obdržel je 10.0.0.32. Z jiného počítače se ke kitu připojíme pomocí jednoduchého příkazu.

```
$ ssh 10.0.0.32 -l root
```

Program nás následně vyzve k zadání k zadání hesla pro uživatele root na kitu. Zadejme jej. Nyní bychom měli již být připojeni a můžeme kit ovládat úplně stejně jako bychom byli připojeni ke konzoli přes sériový port.