1 Laboration 3: Hårdvarunära programmering

1.1

Skapa en funktion för att läsa av värdet från knapparna.
`void read_BUTTON (int *nB1,int *nB2)`. nBx är 1 om knapp nedtryckt annars 0. Anmärkning: Varför måste man skicka med pekare till funktionen?

1.2

Skapa en funktion för att tända och släcka lysdioderna.
`void set_LED(int nLD1,int nLD2,int nLD3,int nLD4,)`. Om nLDx är 1 skall dioden lysa om 0 skall den vara släckt.

1.3

Skapa en funktion för att blinka med lysdioderna.
`void blink_LED(int nBlinkfreq, int nBlinktime)`. Där nBlinkfreq är blinkfrekvensen och nBlinktime är tiden för hur länge blinkningen skall pågå.
Krav: Använd en timer.

1.4

Skapa ett program där man med knapparna väljer vilket beteende lysdioderna har.

Vänster knapp	Tända alla dioder.	`void LEDS_on(void)`
Höger knapp	Släcka alla dioder.	`void LEDS_off(void)`
Båda knapparna	Rinnande ljus.	`void Walking_LEDS(void)`

Programmet skall vara så att en flagga ändras var femte sekund. När flaggan är ett läses knapparna av och när den är noll ändras beteendet på dioderna.
Krav: Använd funktionspekare för att köra de olika funktionaliteterna.

1.5

Utöka ovanstående program så att det kan läsa av och presentera temperaturen samtidigt som diodstyrningen bevaras.
Krav: Sammanlagt behövs alltså två timers. Använd avmaskning vid avbrottshanteringen.

2 Skapa ett projekt i IAR

Förutom de filer som ni själva skapar krävs en länkfil, en startupfil och ett nedladdningsmacro. Dessutom för att underlätta programmeringen inkluderas tre header-filer: hwoption.h, option.h, och s3c3410.h. Dessa headerfiler innehåller bl a adresser till kontroll-, adressregister och portar.

I I cstartup görs de grundläggande initieringar som krävs för att köra ett C-program. Denna fil hittas och länkas med automatiskt. (I de fall då man vill skriva en egen så måste man markera `Projekt->options->XLINK`: Ignore Cstartup). Startup programsnutten körs vid reset och därefter sker ett hopp till main() funktionen, där ert C-program börjar.

II .xcl-Länkningsfil: När ni trycker på byggknappen så kompileras varje programfil till objektfiler var för sig (och läggs i debugger obj katalogen). Dessa länkas sedan samman till en exkverbar binärfil. Men för att länkaren skall veta var i minnet programmet, variabler och stacken etc. skall placeras så definieras detta i länkfilen. Dessutom sätts här storleken på heap och stack(-ar).

III .mac är en macrofil som används av debuggern när du laddar ner koden till SDRAMET. I detta macro så bankas Bank7 till adress 0 och er kod läggs sedan ner dit. Dessutom så initieras Bank3 till 0x06000000. Kontrollera i projektets inställningar `Projekt->options->CSPY` att rätt setup fil (.mac) används!

Inställningar i projektet

Under rubriken `Project->options` finns inställningarna för projektet. Av grundinställningarna behöver följande förändras: `General Processor`: *variant ARM7TDMI samt Arm mode och Big Endian.*

Länkningen ställs in under `XLINK`: med *Output format* till Debug info. Vilken länkfil som används sätts med `XLINK->Include`: *XCL-filename.*

Vid avlusning av sitt program kan en .map-fil vara användbar. Genom att klicka i `XLINK->List`: *Generate linker listing*, *Segment map*, *Module map* så skapas en .map-fil i list biblioteket. I denna fil visas bl a var i minnet funktioner och globala variabler placeras. En avlusningsmetod kan vara att skapa en variabel och sedan mha map-filen titta i minnet hur den ändrats.

3 Lite om avbrottshantering i C

Vid avbrott så hoppar programräknaren till avbrottsvektorn. Positionen i avbrottsvektorn beror på vilken typ av avbrott som skett: mjukvaruavbrott, hårdvaruavbrott etc. I assembler krävs att man bl a sparar undan register i början av avbrottet och att dessa återställs i slutet av funktionen. I C däremot så kan man definiera en funktion av sorten avbrott enligt:

`__irq __arm void IRQ_handler(void)`. Namnet på funktionen är det samma som det i avbrottsvektorn. Kompilatorn lägger då till registerhantering, inladdning av återhoppsadress och initiering av stacken använd vid avbrott. Men, något som inte hanteras automatiskt är bland annat: av och påställning av avbrott, identifiering av vilken avbrottskälla mm.

4 Övrigt

Exempelkod finns i hw_cprog.c och vecs.s79 vilka finns i den komprimerade filen hw_Cprog.zip. Denna innehåller också övriga nödvändiga filer. Hur temperturgivaren fungerar finns i MAX6575L/H databladen (finns på kursens hemsida). För användandet av timers se kap 8 och avbrott kap 11 och 15 i *S33410X Users manual.*