FL 9

Används tillsammans med PPT ”PIS_Periferi_och_io”

Exempelprojekt

PLL och ”timer” inledande exempel

EXEMPEL: Initiering av systemets arbetstakt:
Lägg på OH: CRG: Översikt register, adressdefinitioner i C och assembler

```
// i fil ”crg.c”
#include ”defsCRG.H”
     /* PLL */
     ( ((CRG*) (CRG_BASE))->refdv ) = REFDVVal;
     ( ((CRG*) (CRG_BASE))->synr ) = SYNRVal;
     /* vänta tills PLL låst... */
     while( (( ((CRG*) (CRG_BASE))->crgflg ) & LOCK )== 0);
     /*   växla systemklocka till PLL. */
     ( ((CRG*) (CRG_BASE))->clksel ) |= PLLSEL;
```

Samma sak fast i assembler...

```
; i fil ”crg.s12”

     USE        ”defsCRG.s12”
;    Generisk kod för programmerad arbetstakt...
     MOVB       #REFDVVal,REFDV
     MOVB       #SYNRVal,SYNR
wait:
     BRCLR      CRGFLG,#LOCK,wait     ; vänta tills PLL låst...
     BSET       CLKSEL,#PLLSEL        ; växla systemklocka till PLL.
```

EXEMPEL: Initiering av systemklocka och avbrottsvektor:
Lägg på OH: CRG: Översikt register, adressdefinitioner i C och assembler

```
#include   ”defsCRG.H”
#ifdef SIMTEST
     #define    TIMEBASE 0x10
#else
     #define    TIMEBASE 0x49
#endif

#define    SET_IRQ_VECTOR( interrupt_handler, address ) \
     *(unsigned int *) address = &( interrupt_handler )

     ...
void  timer_interrupt_routine( void );
void  timer_init( void )
{
     /* Skriv tidbas för avbrottsintervall till RTICTL */
     ( ((CRG*) (CRG_BASE))->rtictl ) = TIMEBASE;
     /* Aktivera avbrott från CRG-modul */
     ( ((CRG*) (CRG_BASE))->crgint ) = RTIE;
     /* måste också initiera avbrottsvektor */
     SET_IRQ_VECTOR(timer_interrupt_routine, 0x3FF0 );
}
```

Samma sak fast i assembler...

```
; i fil ”crg.s12”
#ifdef SIMTEST
TIMEBASE   EQU   $10
#else
TIMEBASE   EQU   $49
#endif

     USE        ”defsCRG.s12”
     ...
     IMPORT     _timer_interrupt_routine
timer_init:
; Initiera RTC avbrottsfrekvens
; Skriv tidbas för avbrottsintervall till RTICTL
     MOVB  # TIMEBASE,RTICTL
; Aktivera avbrott från CRG-modul
     MOVB  #RTIE,CRGINT
; måste också initiera avbrottsvektor:
     MOVW  #_timer_interrupt_routine,$3FF0
     RTS
```

Vid AVBROTT från räknarkretsen...

I STANDARD C kan inte avbrottsrutiner implementeras, många kompilatorer erbjuder dock utökningar, här förutsätter vi standard C..

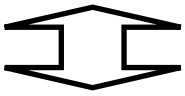
```
      USE         ”defsCRG.s12”
      ...
      EXPORT      _timer_interrupt_routine
_timer_interrupt_routine:
; Kvittera avbrott från RTC
      BSET        CRGFLG,#RTIF
;     ”övrigt” som ska göras vid avbrott
      RTI
```

Alternativt: skriv en "trampolin" i assembler...
Gör grovjobbet i 'C'

```
; i fil ”crg_asm.s12”
      EXPORT      _timer_interrupt_routine
      IMPORT      _C_interrupt_routine
timer_interrupt_routine:
      JSR         _C_interrupt_routine
      RTI


// i fil ”crg.c”
#include   ”CRG.H”
void  C_interrupt_routine( void )
{
      ( ((CRG*) (CRG_BASE))->crgflg ) |= RTIF;
      /* ”övrigt” som ska göras vid avbrott */
}
```

	Clock Reset Generator (CRG)											
	Offset		7	6	5	4	3	2	1	0	Mnemonic	Namn
$34	$00	R	0	0	SYN5	SYN4	SYN3	SYN2	SYN1	SYN0	SYNR	Synthesizer Register
		W										
$35	$01	R	0	0	0	0	REFDV3	REFDV2	REFDV1	REFDV0	REFDV	Reference Divide Register
		W										
$36	$02	R	0	0	0	0	0	0	0	0	CTFLG	*)Test Flags Register
		W										
$37	$03	R	RTIF	PORF	LVRF	LOCKIF	LOCK	SCMIE	SCMIF	SCM	CRGFLG	Flags Register
		W										
$38	$04	R	RTIE	0	0	LOCKIE	0	0	SCMIE	0	CRGINT	Interrupt Enable Register
		W										
$39	$05	R	PLLSEL	PSTP	SYSWAI	ROAWAI	PLLWAI	CWAI	RTIWAI	COPWAI	CLKSEL	Clock Select Register
		W										
$3A	$06	R	CME	PLLON	AUTO	AOQ	0	PRE	PCE	SCME	PLLCTL	PLL Control Register
		W										
$3B	$07	R	0	RTR6	RTR5	RTR4	RTR3	RTR2	RTR1	RTR0	RTICTL	RTI Control Register
		W										
$3C	$08	R	WCOP	RSBCK	0	0	0	CR2	CR1	CR0	COPCTL	COP Control Register
		W										
$3D	$09	R	0	0	0	0	0	0	0	0	FORBYP	*)Force and Bypass Test Register
		W										
$3E	$0A	R	0	0	0	0	0	0	0	0	CTCTL	*)Test Control Register
		W										
$3F	$0B	R	0	0	0	0	0	0	0	0	ARMCOP	COP Arm/Timer Reset
		W	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0		



```
; i fil defsCRG.s12
;( Kretsens BASADRESS är $34 )
SYNR        EQU   $34
REFDV       EQU   $35
CRGFLG      EQU   $37
CRGINT      EQU   $38
CLKSEL      EQU   $39









; Bitdefinitioner
PLLSEL      EQU   $80
LOCK        EQU   8
RTIE        EQU   $80
RTIF        EQU   $80
; Registervärden
REFDVVal:   EQU   1
SYNRVal:    EQU   5
```

```
// i fil ”defsCRG.H”
typedef struct sCRG{
      volatile unsigned char synr;
      volatile unsigned char refdv;
      volatile unsigned char ctflg;
      volatile unsigned char crgflg;
      volatile unsigned char crgint;
      volatile unsigned char clksel;
      volatile unsigned char pllctl;
      volatile unsigned char rtictl;
      volatile unsigned char copctl;
      volatile unsigned char forbyp;
}CRG, *PCRG;
#define     CRG_BASE    0x34

#define     PLLSEL      0x80
#define     LOCK        8
#define     RTIE        0x80
#define     RTIF        0x80

#define     REFDVVal    1
#define     SYNRVal     5
```