Externa avbrott

Ur innehållet:

- Avbrottsvippa med programstyrd återställning
- Pulsgenerator för periodiska avbrott
- EXTI - External Interrupt Configuration
- NVIC - Nested Vectored Interrupt Controller

Läsanvisningar:

- Arbetsbok kap 6.5-6.7

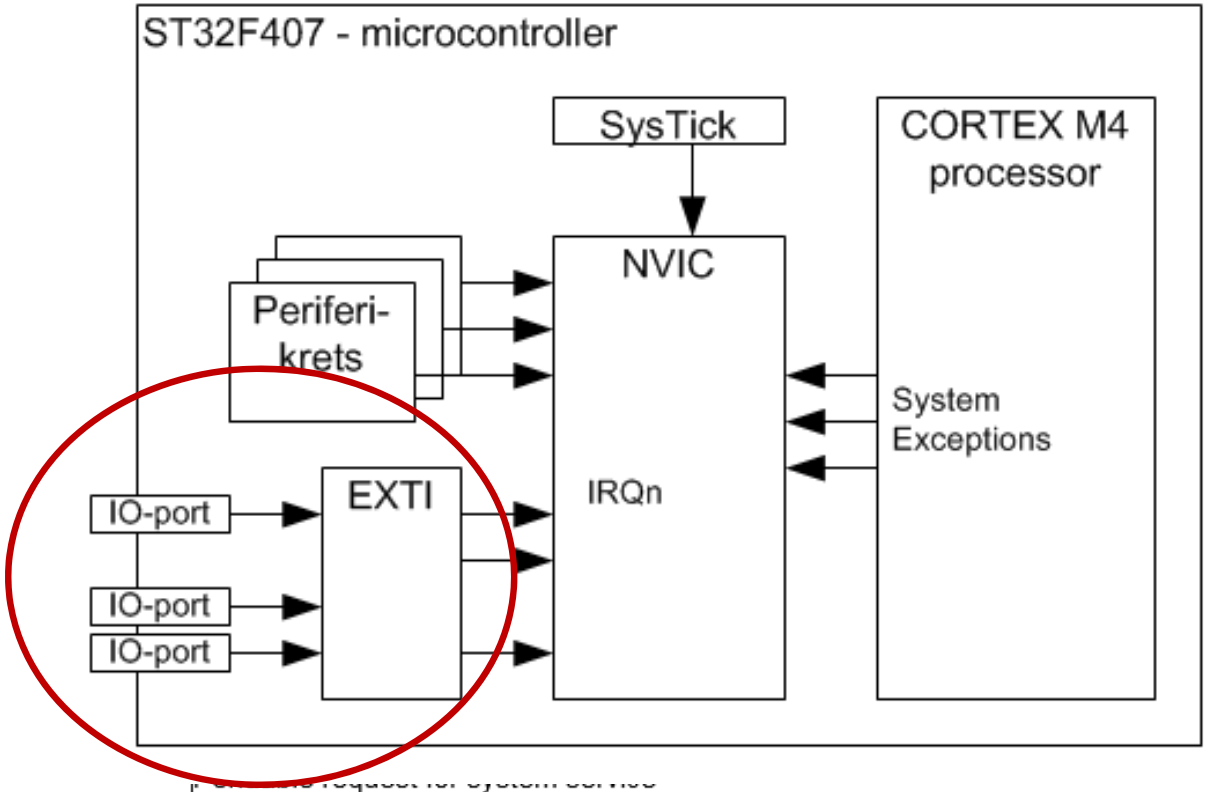
Målsättningar:

- Konfigurera portpinne som en avbrottsingång
- Programmera extern avbrottsvippa, och hantera vektoriserade avbrott

Externa avbrott

En portpinne kan konfigureras som avbrottsingång via EXTI-modulen (External interrupt/event controller)

Detta möjliggör avbrott från enheter utanför enchipsdatorn, *externa avbrott*

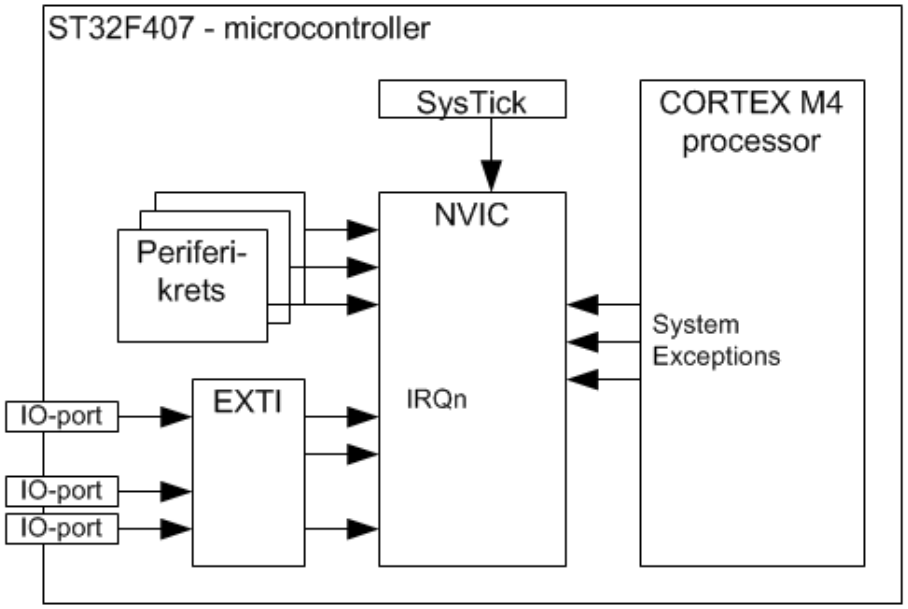


	6	settable	SysTick	System tick timer
0	7	settable	WWDG	Window Watchdog interrupt
1	8	settable	PVD	PVD through EXTI line detection interrupt
2	9	settable	TAMP_STAMP	Tamper and TimeStamp interrupts through the EXTI line
3	10	settable	RTC_WKUP	RTC Wakeup interrupt through the EXTI line
4	11	settable	FLASH	Flash global interrupt
5	12	settable	RCC	RCC global interrupt
6	13	settable	EXTI0	EXTI Line0 interrupt
7	14	settable	EXTI1	EXTI Line1 interrupt
8	15	settable	EXTI2	EXTI Line2 interrupt
9	16	settable	EXTI3	EXTI Line3 interrupt
10	17	settable	EXTI4	EXTI Line4 interrupt
11	18	settable	DMA1_Stream0	DMA1 Stream0 global interrupt
12	19	settable	DMA1_Stream1	DMA1 Stream1 global interrupt

Externa avbrottskällor

Avbrottsingångar konfigureras i tre steg

- SYSCFG-modulen används för att koppla en GPIO-pinne från någon port till en specifik avbrottslina (1 av 16) .
- EXTI-modulen programmeras.
- NVIC-modulen (Nested vectored interrupt controller) programmeras för att hantera avbrott från denna avbrottsvektor (NVIC_ISERx).



Irq	Hanteringsrutin	Beskrivning
EXTI0_IRQn	EXTI0_IRQHandler	för IO-pinnar anslutna till avbrottslina 0
EXTI1_IRQn	EXTI1_IRQHandler	för IO-pinnar anslutna till avbrottslina 1
EXTI2_IRQn	EXTI2_IRQHandler	för IO-pinnar anslutna till avbrottslina 2
EXTI3_IRQn	EXTI3_IRQHandler	för IO-pinnar anslutna till avbrottslina 3
EXTI4_IRQn	EXTI4_IRQHandler	för IO-pinnar anslutna till avbrottslina 4
EXTI9_5_IRQn	EXTI9_5_IRQHandler	för IO-pinnar anslutna till avbrottslinor 5 till 9
EXTI15_10_IRQn	EXTI15_10_IRQHandler	för IO-pinnar anslutna till avbrottslinor 10 till 15

SYSCFG

EXTICRx används för att koppla IO-pinnar till processorns avbrottssystem

offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Register
0																																	SYSCFG_MEMRMP
4																																	SYSCFG_PMC
8																																	SYSCFG_EXTICR1
0xC																																	SYSCFG_ EXTICR2
0x10																																	SYSCFG_ EXTICR3
0x14																																	SYSCFG_ EXTICR4
0x20																																	SYSCFG_ CMPCR

EXTICRx är fyra kontrollregister med samma struktur:

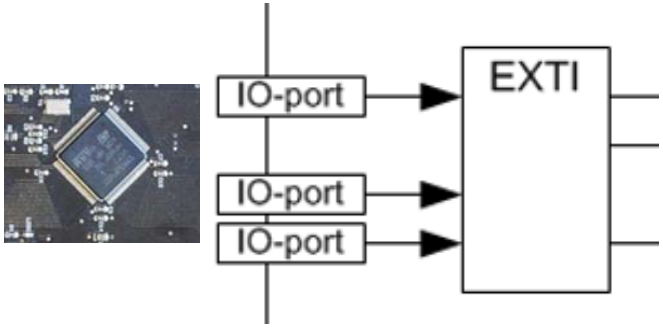
offset	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Register
8	EXTI3[3:0]				EXTI2[3:0]				EXTI1[3:0]				EXTI0[3:0]				SYSCFG_EXTICR1
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

Bitarna i EXTICRx bestämmer hur en IO-pinne dirigeras till någon av de 16 avbrottslinorna EXTI0..EXTI15 genom att motsvarande fält skrivs med fyra bitar.
Undantag: PK[15:8] används ej.

0000: PA[x]	0001: PB[x]	0010: PC[x]	0011: PD[x]	0100: PE[x]	0101: PF[x]
0110: PG[x]	0111: PH[x]	1000: PI[x]	1001: PJ[x]	1010: PK[x]	

Exempel:

Koppla pinne PC0 till “Extern avbrott 0”



5	12	settable	RCC	RCC global interrupt	0x0000 0054
6	13	settable	EXTI0	EXTI Line0 interrupt	0x0000 0058
7	14	settable	EXTI1	EXTI Line1 interrupt	0x0000 005C

offset	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Register
8	EXTI3[3:0]				EXTI2[3:0]				EXTI1[3:0]				EXTI0[3:0]				SYSCFG_EXTICR1
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

0000: PA[x]	0001: PB[x]	0010: PC[x]	0011: PD[x]	0100: PE[x]	0101: PF[x]
0110: PG[x]	0111: PH[x]	1000: PI[x]	1001: PJ[x]	1010: PK[x]	

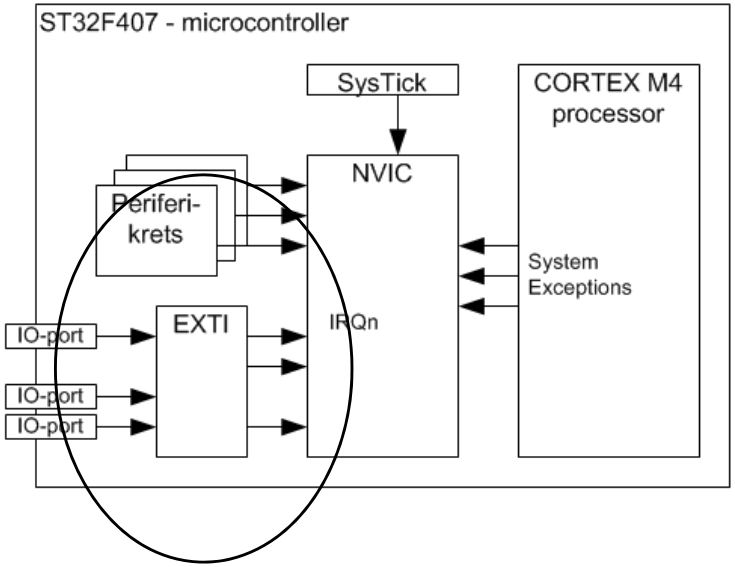
```
*SYSCFG_EXTICR1 &= 0xFFF0;      /* Nollställ bitfältet */
*SYSCFG_EXTICR1 |= 0x0002;      /* 0010 -> EXTI0[3:0] */
```

EXTI-modul

Totalt finns det 23 olika möjliga avbrottslinor i EXTI-modulen, i STM32F407 används de 16 första för externa avbrott. Några av dessa har gemensam avbrottsvektor.

Irq	Hanteringsrutin	Beskrivning
EXTI0_IRQn	EXTI0_IRQHandler	för IO-pinnar anslutna till avbrottslina 0
EXTI1_IRQn	EXTI1_IRQHandler	för IO-pinnar anslutna till avbrottslina 1
EXTI2_IRQn	EXTI2_IRQHandler	för IO-pinnar anslutna till avbrottslina 2
EXTI3_IRQn	EXTI3_IRQHandler	för IO-pinnar anslutna till avbrottslina 3
EXTI4_IRQn	EXTI4_IRQHandler	för IO-pinnar anslutna till avbrottslina 4
EXTI9_5_IRQn	EXTI9_5_IRQHandler	för IO-pinnar anslutna till avbrottslinor 5 till 9
EXTI15_10_IRQn	EXTI15_10_IRQHandler	för IO-pinnar anslutna till avbrottslinor 10 till 15

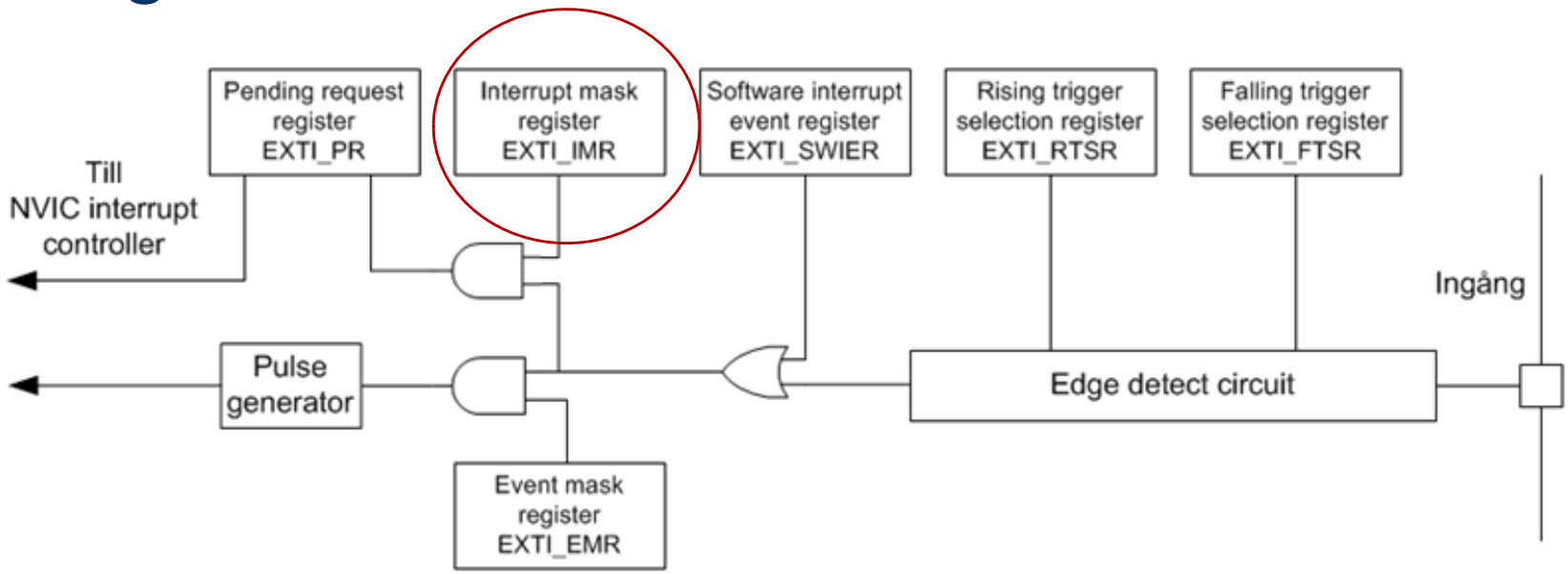
Irq	Beskrivning
EXTI16	PVD utgång fast ansluten till EXTIO16
EXTI17	RTC "Alarm event" fast ansluten till EXTIO17
EXTI18	USB OTG FS "Wakeup event" fast ansluten till EXTIO18
EXTI19	Ethernet "Wakeup event" fast ansluten till EXTIO19
EXTI20	USB OTG HS "Wakeup event" fast ansluten till EXTIO20
EXTI21	RTC "Tamper and TimeStamp" fast ansluten till EXTIO21
EXTI22	RTC "Wakeup event" fast ansluten till EXTIO22



offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Register
0																																	EXTI_IMR
4																																	EXTI_EMR
8																																	EXTI_RTSR
0xC																																	EXTI_FTSR
0x10																																	EXTI_SWIER
0x14																																	EXTI_PR

Port pinnar Px 15 .. Px 0 (x=A,B,C,D,E,F)

EXTI– registren har samma struktur



EXTI_IMR (0x40013C00) Interrupt Mask Register

offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	mnemonic
0										rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	EXTI_IMR

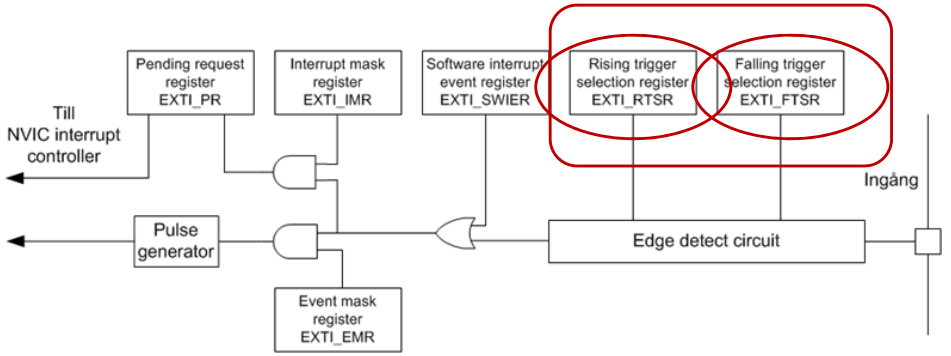
Kontrollerar EXTI 15

Kontrollerar EXTI 0

Exempel: Tillåt avbrott från EXTI0

```
*EXTI_IMR |= 1;
```

EXTI– triggfunktioner



EXTI_RTSR (0x40013C08) Rising Trigger Selection Register

offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	mnemonic
8										rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	EXTI_RTSR

Exempel: Generera avbrott då insignalen från den pinne som "routats" till EXTI6 går från 0 till1, dvs "positiv flank":

```
*EXTI_RTSR |= 1<<6;
```

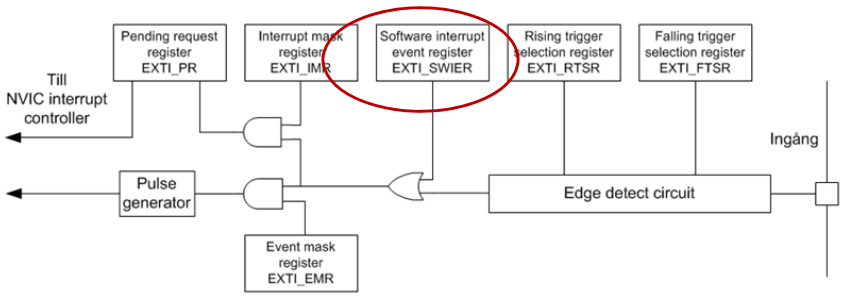
EXTI_FTSR (0x40013C0C) Falling Trigger Selection Register

offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	mnemonic
0xC										rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	EXTI_FTSR

Exempel: Generera avbrott då insignalen från den pinne som "routats" till EXTI2 går från 1 till 0, dvs "negativ flank":

```
*EXTI_FTSR |= 1<<2;
```

EXTI– ”software interrupt”



Avbrott kan också genereras av programvara.

Om avbrott är tillåtet för lina *x*, kan programvara ge detta avvaktande tillstånd genom att skriva '1' till motsvarande bit i EXTI_SWIER.

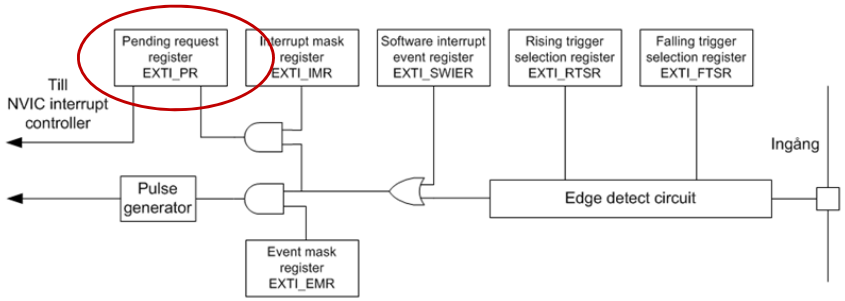
EXTI_SWIER (0x40013C10) Software Interrupt Event Register

offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	mnemonic
0x10										rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	EXTI_SWIER

Exempel: Generera avbrott hos EXTI3

```
*EXTI_SWIER |= 1<<3;
```

EXTI– ”interrupt pending”



Om ett avbrott avvaktar är motsvarande bit 1 i EXTI_PR.

Men registret är också skrivbart och biten återställs till 0 genom att 1 skrivs till bitpositionen.

EXTI_PR (0x40013C14) Pending Register

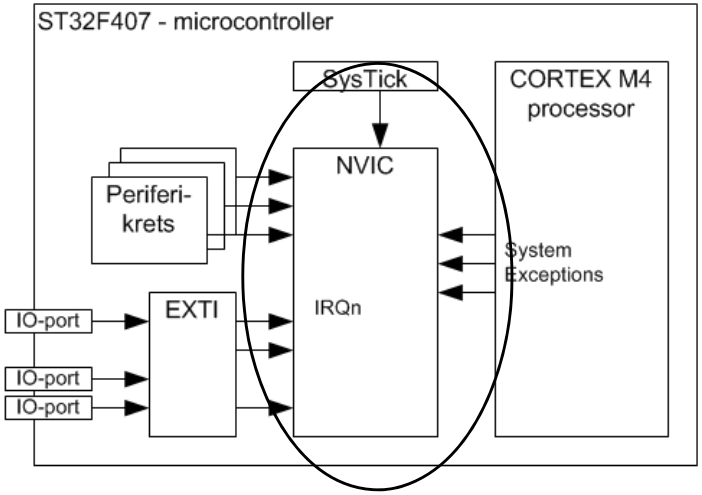
offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	mnemonic
0x14										rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	EXTI_PR

Exempel: Om ett avbrott avvaktar för EXTI15, "kvittera" (återställ) då detta.

```
if( *EXTI_PR & (1<<15) )
    *EXTI_PR |= 1<<15;
```

NVIC – Kontrollerar alla avbrott

offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	mnemonic
0x000	SETENA[31:0]																																NVIC_ISER0
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	
0x004	SETENA[63:32]																																NVIC_ISER1
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	
0x008	Reserverat																SETENA[81:64]																NVIC_ISER2
																	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	



Fem uppsättningar register för olika funktioner, men med samma struktur. Vår processor har 82 avbrottsgenererande enheter, alltså har varje registeruppsättning 82 bitar.

- Interrupt Set Enable Registers, NVIC_ISERx
- Interrupt Clear Enable Registers, NVIC_ICERx
- Interrupt Set Pendig Registers, NVIC_ISEPRx
- Interrupt Clear Pending Registers, NVIC_ICPRx
- Interrupt Active Bit Registers, NVIC_IABRx

Bestäm bitnummer i NVIC

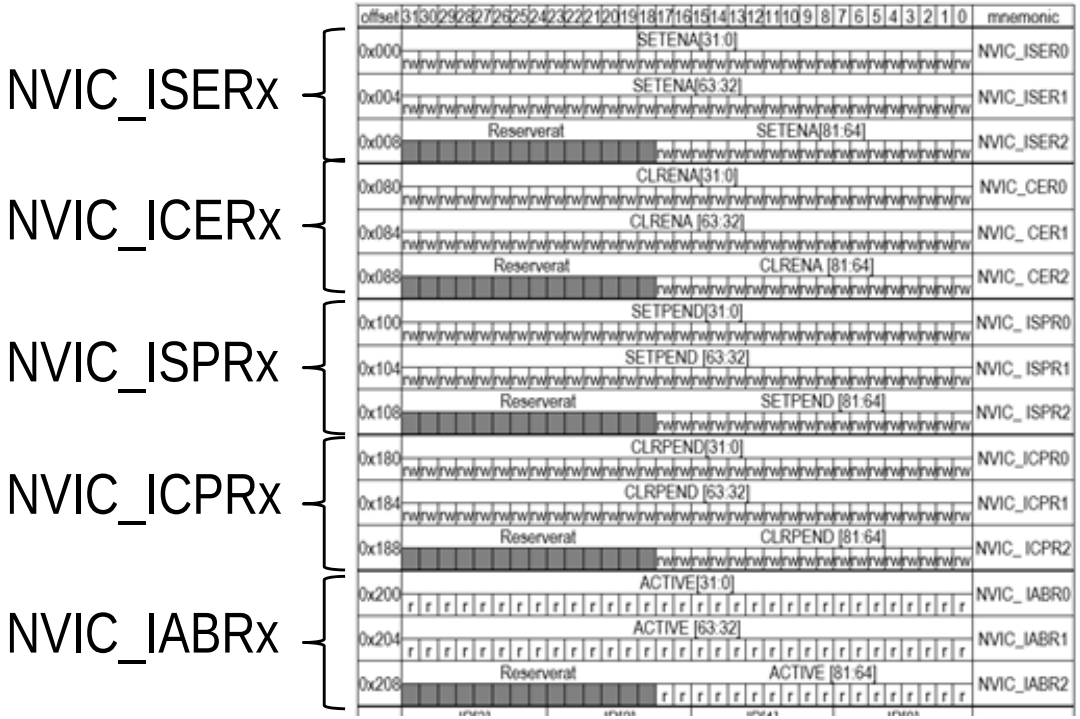
För vektor IRQ=n (n = 0..81):

x = n/32

dvs. x kan vara 0,1,2

bit = n – (32*x),

dvs. bit kan vara 0..31



offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	mnemonic
0x000	SETENA[31:0]																																NVIC_ISER0
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	
0x004	SETENA[63:32]																																NVIC_ISER1
	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	
0x008	Reserverat													SETENA[81:64]																			NVIC_ISER2
														rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	

Exempel:
Avbrottsvektor och bitnummer i NVIC för EXTI15_10

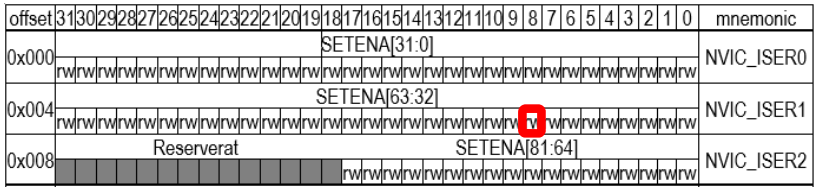
Vi letar upp EXTI15_10 i vektortabellen. Den första kolumnen anger avbrottets nummer i NVIC, dvs. 40. Kolumnen längst till höger anger offset för avbrottsvektorns placering i vektortabellen:

39	settable	USART3	USART3 global interrupt	0x0000 00DC
40	settable	EXTI15_10	EXTI Line[15:10] interrupts	0x0000 00E0
41	settable	RTC_Alarm	RTC Alarms (A and B) through EXTI line interrupt	0x0000 00E4

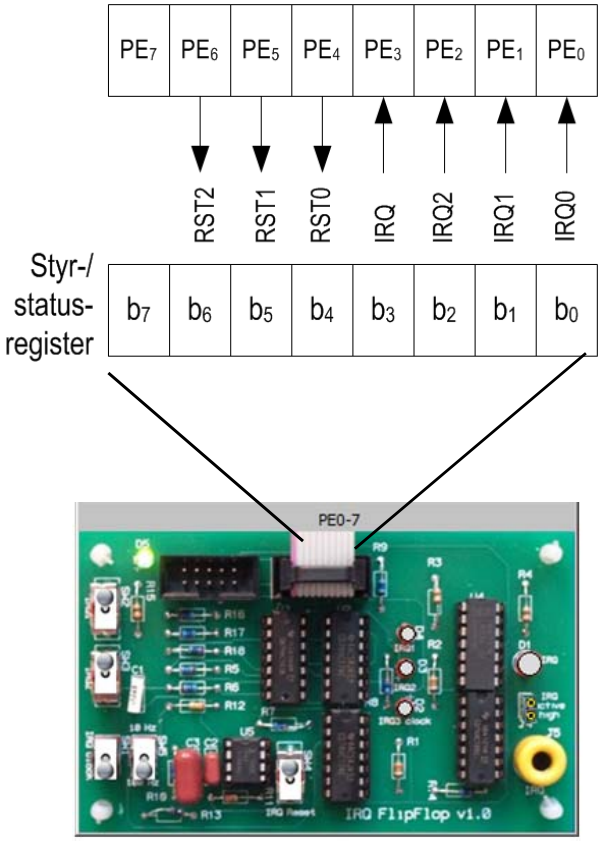
Vi har alltså avbrottsnummer 40 och får:
x = 40/32 = 1
Bit = 40 -32 = 8
Dvs: bit 8 i register NVIC_ xxxx1

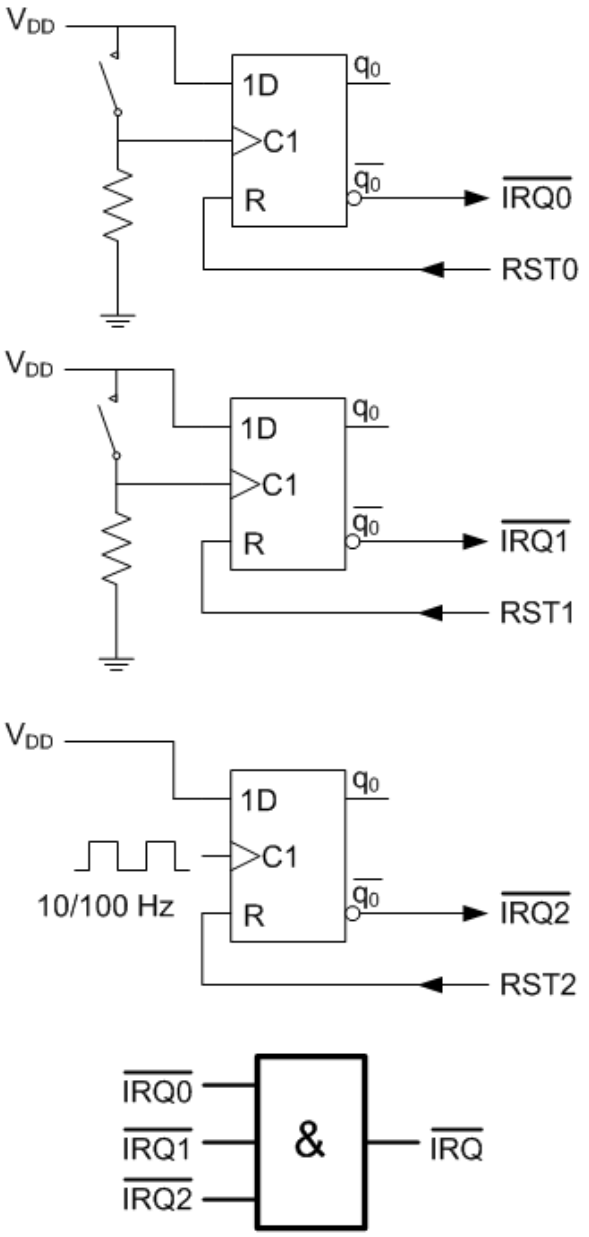
Exempel: Möjliggör avbrott EXTI15_10

```
*NVIC_ISER1 = (1<<8);
```

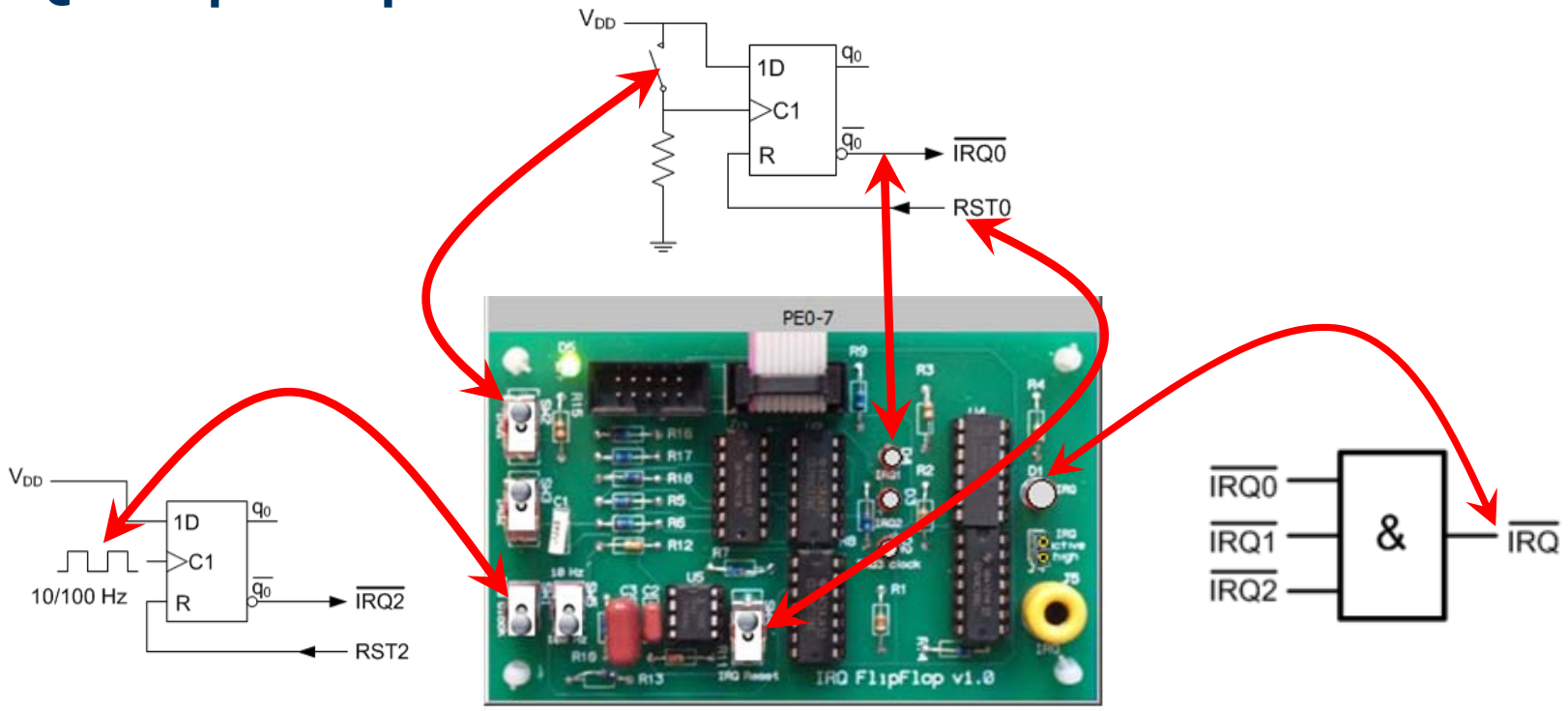


IRQ-FlipFlop - avbrottsvippor

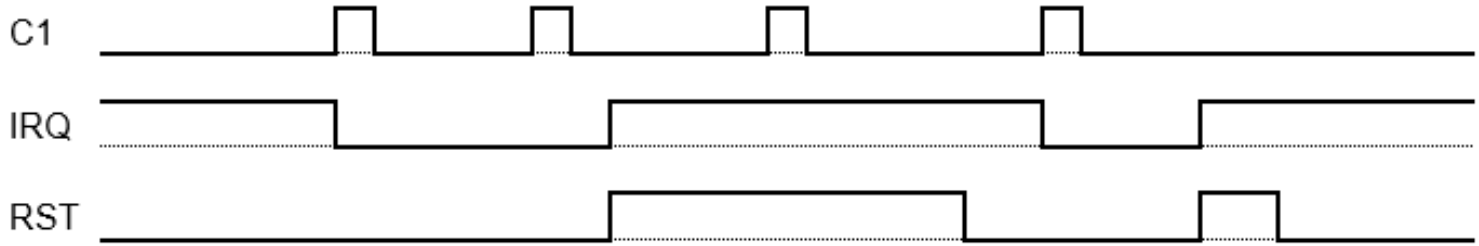




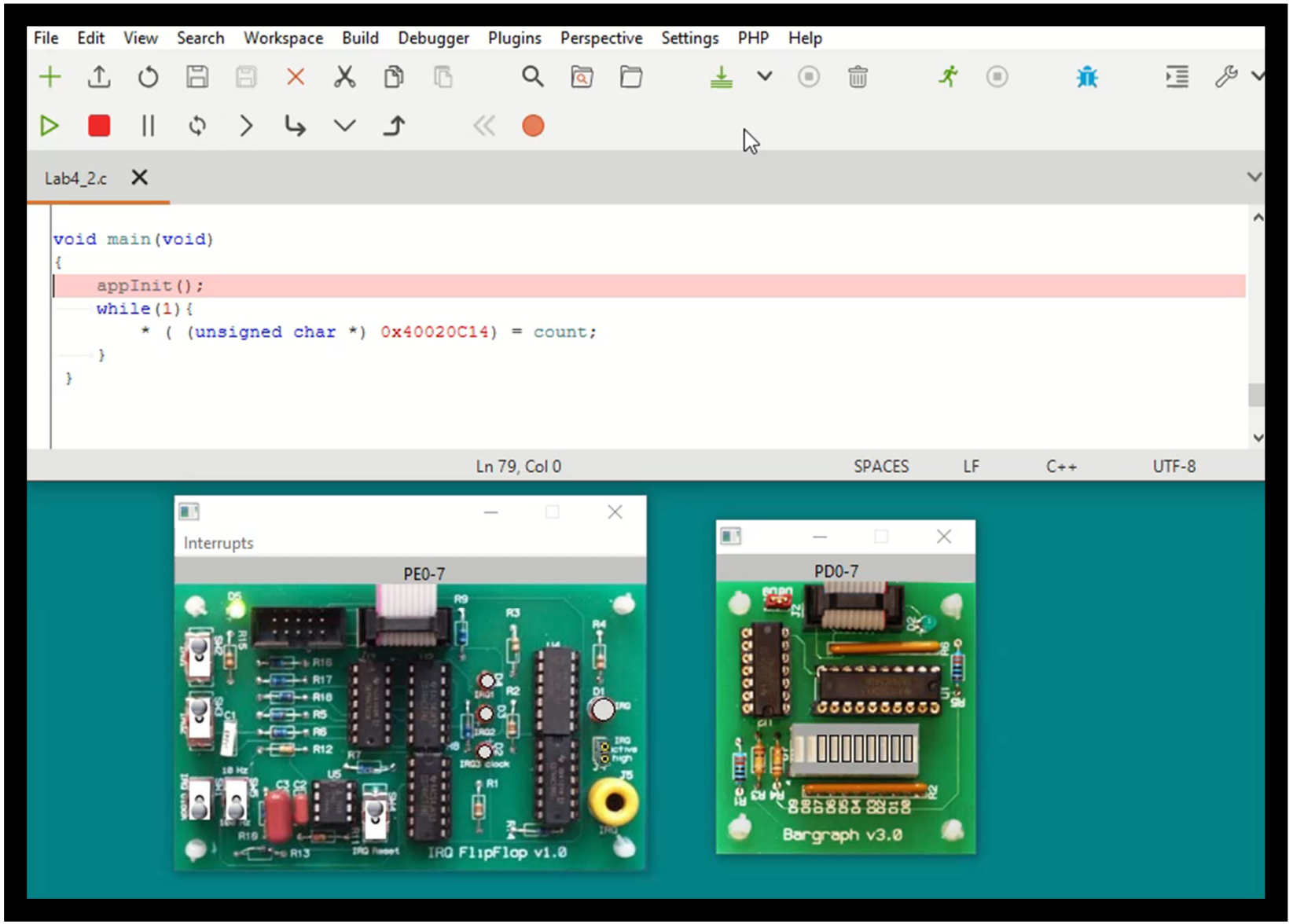
IRQ-FlipFlop - funktion



Exempel: Aktivering och återställning av IRQ-signal (aktiv låg)



Inför laborationsuppgifter 4.2-4.4...



Pseudoparallell exekvering - en enkel ”TASK-switch”

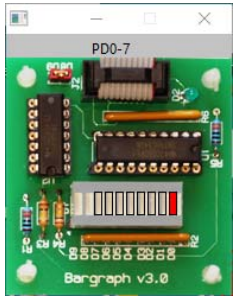
En enkel applikation med fyra olika program, (*processerna*) p0, p1, p2 och p3

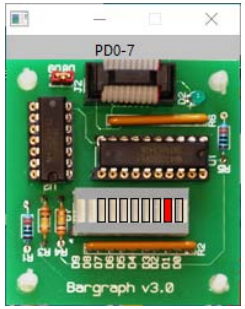
Växling mellan programmen utförs av ett överordnat program SV(*Supervisor*) vid ett avbrott från EXTI0 (bit 0).

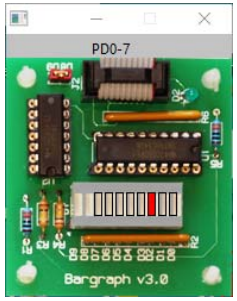
”Processerna”...

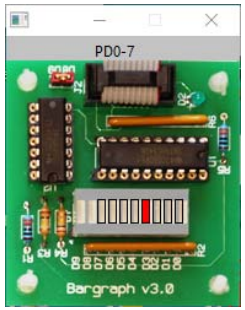
```
void p0( void )
{
  while(1)
    *GPIO_ODR_LOW = 1;
}
void p1( void )
{
  while(1)
    *GPIO_ODR_LOW = 2;
}
void p2( void )
{
  while(1)
    *GPIO_ODR_LOW = 4;
}
void p3( void )
{
  while(1)
    *GPIO_ODR_LOW = 8;
}
```

Processerna tänder var sin diod i diodrampen, vi kan då tydligt se vilken process som körs...

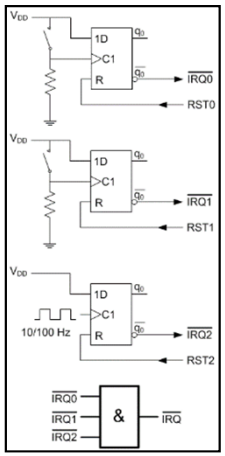








Processtillstånd



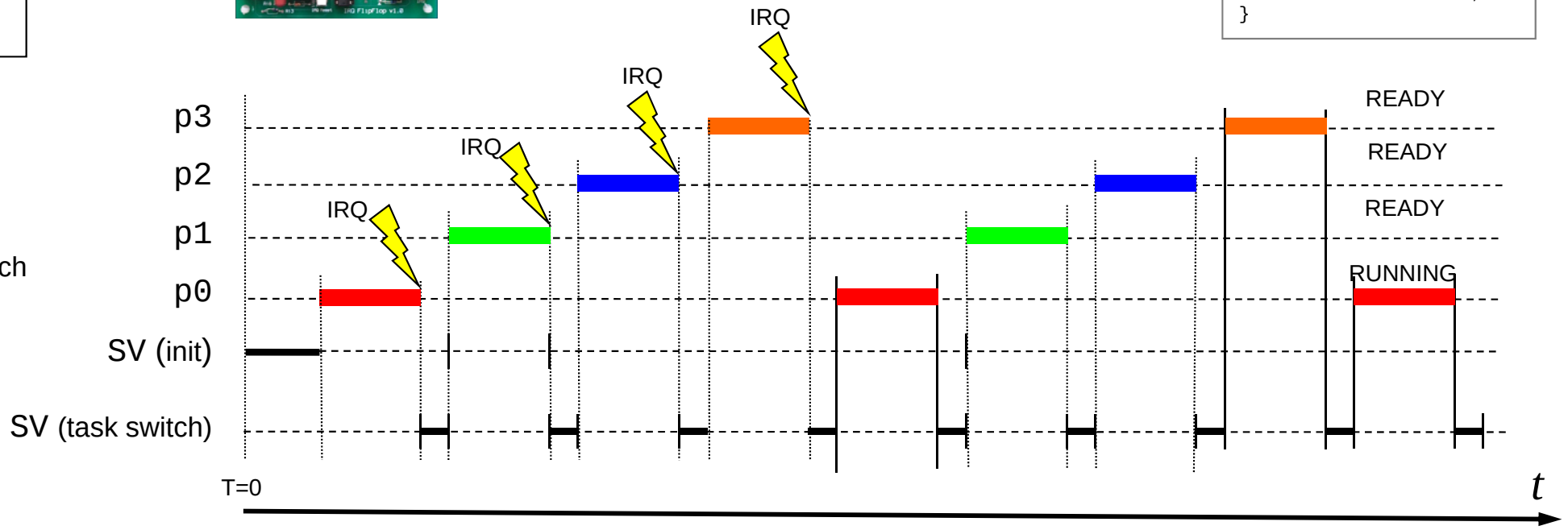
Avbrotten kan också triggas manuellt eller periodiskt från FLIP-FLOP-kortet...

Vi har bara en CPU, som vi växlar mellan processerna. Processen kan därför befinna sig i olika *tillstånd.*

"Processerna"...

```
void    p0( void )
{
  while(1)
    *GPIO_ODR_LOW = 1;
}
void    p1( void )
{
  while(1)
    *GPIO_ODR_LOW = 2;
}
void    p2( void )
{
  while(1)
    *GPIO_ODR_LOW = 4;
}
void    p3( void )
{
  while(1)
    *GPIO_ODR_LOW = 8;
}
```

SV sköter initiering och växling



Task-switch

"Processerna"...

```
void    p0( void )
{
  while(1)
    *GPIO_ODR_LOW = 1;
}
void    p1( void )
{
  while(1)
    *GPIO_ODR_LOW = 2;
}
void    p2( void )
{
  while(1)
    *GPIO_ODR_LOW = 4;
}
void    p3( void )
{
  while(1)
    *GPIO_ODR_LOW = 8;
}
```