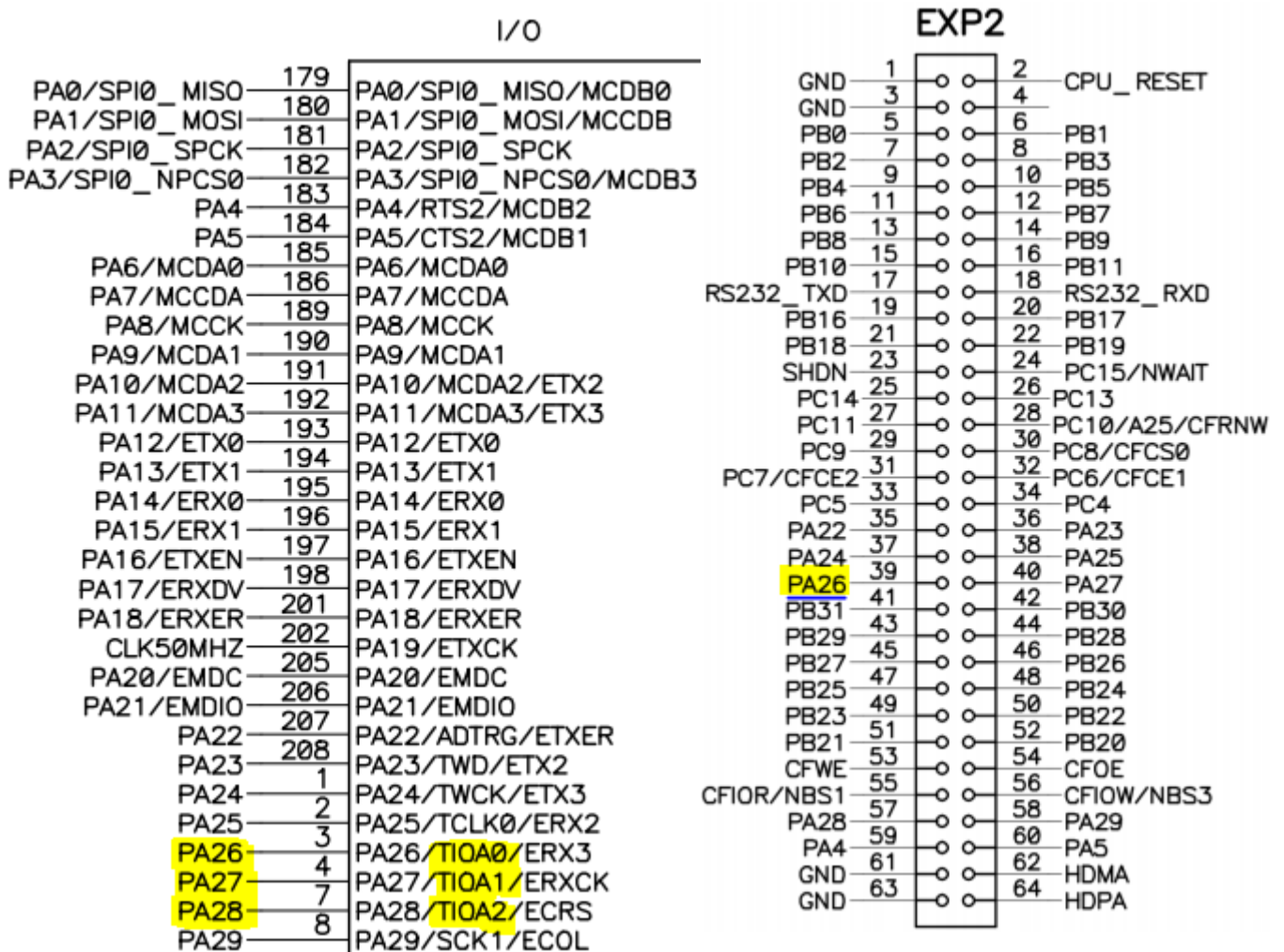


## FRI-SMS generiranje tona

## Priklop in delovanje pinov

Prvo moramo vedeti kateri pini omogočajo izhod ure. To so pini, ki imajo oznako TIOA ali TIOB.



Na levi sliki lahko vidimo, da sta na PA26 pinu lahko TIOA0 in ERX3. Za nas je pomemben TIOA0, katerega upravlja TC0 – Timer/Counter 0.

Velja slednja trditev TCX je vezan na TIOAX in TIOBX in  $X \in \{0,1,2\}$ .

Desna slika nam pa pove na katerem mestu na plošči pri pinih EXP2 je pin PA26.

PA26 pomeni, da je pin krmiljen na 26. bitu PIOA. PB pini so krmiljeni preko PIOB, PC pa preko PIOC. PIO je Peripheral Input/Output Controller.

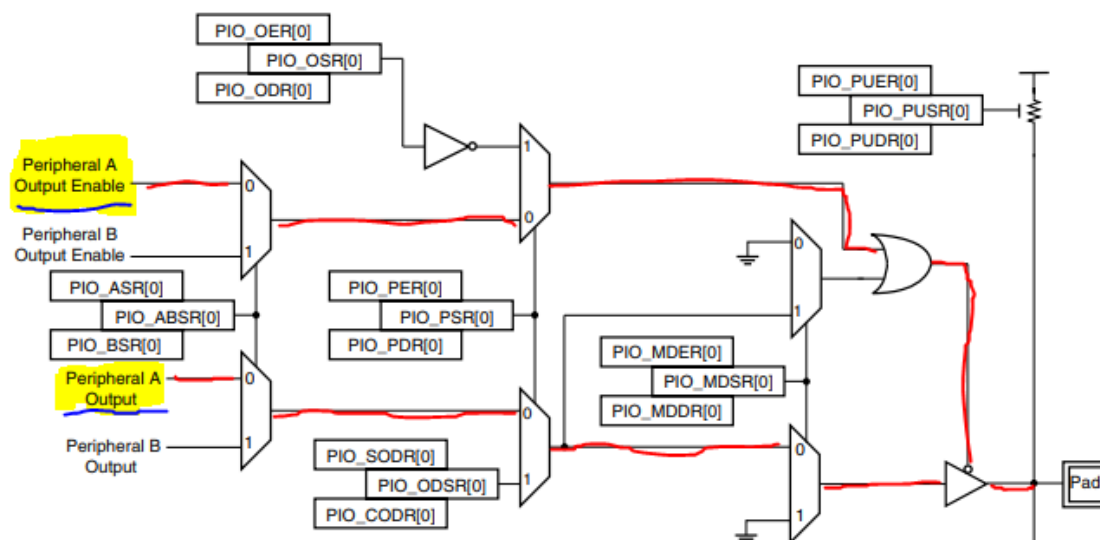
Sedaj vemo, da gre TIOA na + zvočnika medtem, ko gre GND na – zvočnika.

Pomen TIOA in TIOB:

|                             |  |
|-----------------------------|--|
| TIOA – Timer Input Output A | Capture Mode: Timer Counter Input<br>Waveform Mode: Timer Counter Output |
| TIOB – Timer Input Output B | Capture Mode: Timer Counter Input<br>Waveform Mode: Timer Counter Output |

## Nastavljanje vrednosti

Figure 29-3. I/O Line Control Logic



Kot je na sliki prikazano z rdečo barvo, moramo na izhod pripeljati Peripheral A ali B izhod, odvisno ali potrebujemo TIOA ali TIOB izhod. Ali programiramo PAXX/ PCXX / PBXX pine je odvisno od izbire PIO\_BASE naslova.

|             |      |           |
|-------------|------|-----------|
| 0xFFFF F400 | PIOA | 512 Bytes |
| 0xFFFF F600 | PIOB | 512 bytes |
| 0xFFFF F800 | PIOC | 512 bytes |
| 0xFFFF FA00 |      |           |

Da pripeljemo urin signal na pin, moramo nastaviti slednje registre (v našem primeru povsod gledamo A):

- PIO\_ASX / PIO\_BSR -> Že po defaultu je omogočen A, če hočemo omogočiti B moramo zapisati 1 na mesto v PIO\_BSR register.  
**Primer** -> `mov r0, #1 << 26 ; str r0, [r2, #PIO_BSR]` pripeljemo B na 26. Pin.
- PIO\_PER / PIO\_PDR -> PIO\_PER omogoči PIO nadzor nad izhodom, PIO\_PDR pa onemogoči. Če hočemo pripeljati A/B na izhod moramo v PIO\_PDR zapisati 1.  
**Primer** -> `mov r0, #1 << 26 ; str r0, [r2, #PIO_PDR]` onemogočimo PIO nadzor nad pinom 26.
- PIO\_MDER / PIO\_MDDR -> PIO\_MDER omogoči multi drive, PIO\_MDDR pa onemogoči. Če hočemo na izhod pripeljati A/B mora biti 1 v PIO\_MDDR.  
**Primer** -> `mov r0, #1 << 26; str r0, [r2, #PIO_MDRR]` onemogočimo multi drive na pinu 26.

S tem smo sedaj omogočili pot urinega signala na izhodni pin PA26, na katerem ima izhod TIOA0, ki je povezan z TCO.

## Inicializacija TC0

Vse kar še potrebujemo je inicializacija TC0, ko to storimo in če smo v prejšnjih korakih pravilno nastavili, dobimo na izhodu PA26 urin signal, kar je razvidno tudi iz zvoka, ki ga prične zvočnik oddajati. S spreminjanjem razmika med dvema enicama na signalu višamo/nížamo frekvenco tona.

Na začetku moramo omogočiti TC0. Prvo preverimo kam v PMC moramo zapisati 1, da bo TC0 omogočen.

| Peripheral ID | Peripheral Mnemonic | Peripheral Name               | External Interrupt |
|---------------|---------------------|-------------------------------|--------------------|
| 0             | AIC                 | Advanced Interrupt Controller | FIQ                |
| 1             | SYSC                | System Controller Interrupt   |                    |
| 2             | PIOA                | Parallel I/O Controller A     |                    |
| 3             | PIOB                | Parallel I/O Controller B     |                    |
| 4             | PIOC                | Parallel I/O Controller C     |                    |
| 5             | ADC                 | Analog to Digital Converter   |                    |
| 6             | US0                 | USART 0                       |                    |
| 7             | US1                 | USART 1                       |                    |
| 8             | US2                 | USART 2                       |                    |
| 9             | MCI                 | Multimedia Card Interface     |                    |
| 10            | UDP                 | USB Device Port               |                    |
| 11            | TWI                 | Two-wire Interface            |                    |
| 12            | SPI0                | Serial Peripheral Interface 0 |                    |
| 13            | SPI1                | Serial Peripheral Interface 1 |                    |
| 14            | SSC                 | Synchronous Serial Controller |                    |
| 15            | -                   | Reserved                      |                    |
| 16            | -                   | Reserved                      |                    |
| 17            | TC0                 | Timer/Counter 0               |                    |
| 18            | TC1                 | Timer/Counter 1               |                    |
| 19            | TC2                 | Timer/Counter 2               |                    |
| 20            | UHP                 | USB Host Port                 |                    |
| 21            | EMAC                | Ethernet MAC                  |                    |
| 22            | ISI                 | Image Sensor Interface        |                    |
| 23            | US3                 | USART 3                       |                    |
| 24            | US4                 | USART 4                       |                    |
| 25            | US5                 | USART 5                       |                    |
| 26            | TC3                 | Timer/Counter 3               |                    |
| 27            | TC4                 | Timer/Counter 4               |                    |
| 28            | TC5                 | Timer/Counter 5               |                    |

Kot razvidimo je potrebno za TC0 nastaviti pin 17 na ena v registru PMC\_PMCER, ki je Peripheral Clock Enable Register.

**Primer** -> ldr r2, =PMC\_BASE; mov r0, #(1 << 17); str r0, [r2, #PMC\_PCER] omogočimo TC0

Sedaj moramo samo še nastaviti TC0 z vpisom v register TC\_CMRx. Vanj upišemo enice na slednja mesta:

#### 34.6.5 TC Channel Mode Register: Waveform Mode

**Register Name:** TC\_CMRx [x=0..2] (WAVE = 1)

**Access Type:** Read-write

|        |         |        |    |      |    |         |    |
|--------|---------|--------|----|------|----|---------|----|
| 31     | 30      | 29     | 28 | 27   | 26 | 25      | 24 |
| BSWTRG |         | BEEVT  |    | BCPC |    | BCPB    |    |
| 23     | 22      | 21     | 20 | 19   | 18 | 17      | 16 |
| ASWTRG |         | AEEVT  |    | ACPC |    | ACPA    |    |
| 15     | 14      | 13     | 12 | 11   | 10 | 9       | 8  |
| WAVE   | WAVSEL  | ENETRQ |    | EEVT |    | EEVTEDG |    |
| 7      | 6       | 5      | 4  | 3    | 2  | 1       | 0  |
| CPCDIS | CPCSTOP | BURST  |    | CLKI |    | TCCLKS  |    |

**TCCLKS** je namenjen izbiri ure in z kombinacijo 011 izberemo TIMER\_CLOCK4.

##### • TCCLKS: Clock Selection

| TCCLKS |   |   | Clock Selected |
|--------|---|---|----------------|
| 0      | 0 | 0 | TIMER_CLOCK1   |
| 0      | 0 | 1 | TIMER_CLOCK2   |
| 0      | 1 | 0 | TIMER_CLOCK3   |
| 0      | 1 | 1 | TIMER_CLOCK4   |
| 1      | 0 | 0 | TIMER_CLOCK5   |
| 1      | 0 | 1 | XC0            |
| 1      | 1 | 0 | XC1            |
| 1      | 1 | 1 | XC2            |

**WAVSEL** je namenjen izbiri tipa waveforma.

##### • WAVSEL: Waveform Selection

| WAVSEL |   | Effect  |
|--------|---|---|
| 0      | 0 | UP mode without automatic trigger on RC Compare     |
| 1      | 0 | UP mode with automatic trigger on RC Compare        |
| 0      | 1 | UPDOWN mode without automatic trigger on RC Compare |
| 1      | 1 | UPDOWN mode with automatic trigger on RC Compare    |

**WAVE** je namenjen omogočanju waveform načina, ki nam omogoča oddajanje urinega signala.

##### • WAVE

0 = Waveform Mode is disabled (Capture Mode is enabled).

1 = Waveform Mode is enabled.

ACPC je namenjen izbiri obnašanja TIOA glede na register C.

- **ACPC: RC Compare Effect on TIOA**

| ACPC |   | Effect |
|------|---|--------|
| 0    | 0 | none   |
| 0    | 1 | set    |
| 1    | 0 | clear  |
| 1    | 1 | toggle |

**Primer** -> ldr r2, =TC0\_BASE; mov r0, #0b1100110 << 13; add r0, r0, #0b011; str r0, [r2, #TC\_CMR]

Da bi zadeva delovala moramo nastaviti še vrednost v registru C. Ta vrednost nam določa frekvenco tona. Večja kot je vrednost nižja je frekvenca (manjkrat se v določenem časovnem intervalu pojavi menjava med 1 in 0 na izhodu).

**Primer** -> ldr r0, =375; str r0, [r2, #TC\_RC] v register C naložimo vrednost 375.

Sedaj moramo samo še omogočiti uro z vpisom binarne vrednosti 101 v TC\_CCR register.  
TC\_RCC - TC Channel Control Register.

**Register Name:** TC\_CCRx [x=0..2]

**Access Type:** Write-only

|    |    |    |    |    |       |        |       |
|----|----|----|----|----|-------|--------|-------|
| 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     |
| —  | —  | —  | —  | —  | SWTRG | CLKDIS | CLKEN |

- **CLKEN: Counter Clock Enable Command**

0 = No effect.

1 = Enables the clock if CLKDIS is not 1.

- **CLKDIS: Counter Clock Disable Command**

0 = No effect.

1 = Disables the clock.

- **SWTRG: Software Trigger Command**

0 = No effect.

1 = A software trigger is performed: the counter is reset and the clock is started.

S tem omogočimo uro in omogočimo sprožilec ko ura doseže določeno vrednost sproži signal, resetira števec in znova požene uro.

Sedaj bi zadeva morala delovati! ☺