Ultra Low Power 8-bit FLASH Microcontroller

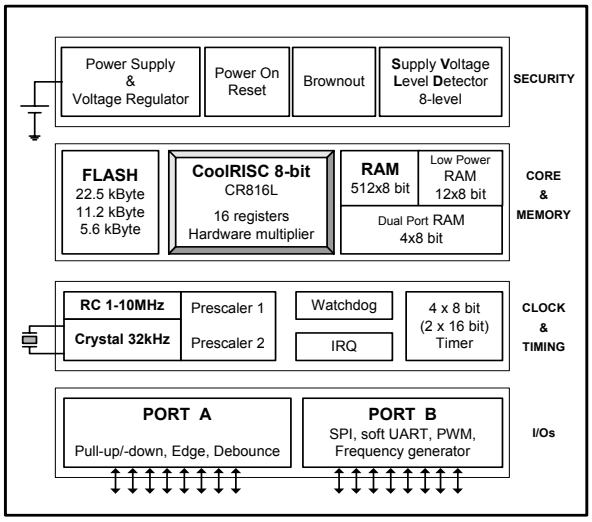
Description

The EM6812 is designed to be battery operated for extended lifetime applications. Brownout and powercheck functions ensure reliable operation at or near undervoltage conditions, offering greater reliability in complex operation modes. Each of the 16 I/Os is freely programmable and the microcontroller has a dual quartz and trimmable RC oscillator up to 10MHz. It has an 8-bit RISC architecture specially designed for very low power consumption. With 2 clocks per instruction, EM6812 executes up to 2.5 MIPS at 5MHz and achieves an astonishing 2200 MIPS/Watt.

Features

- Green mold / leadfree package
- True low current: 120 µA active mode
  6 µA standby mode, RC on
  0.8 µA standby mode, RC off
- Up to 2.5 MIPS at 5MHz
- On-chip brownout detection
- Powercheck functions at start-up
- 8-level Supply Voltage Level Detection (SVLD)
- Fast wake-up from standby mode
- 16 fully configurable I/Os
  - Input / Output
  - Pull-up, Pull-down
  - CMOS, N-channel open drain
- 6 high currents outputs, up to 20 mA
- Wide supply voltage range 2 V – 5.5 V
- Flash read monitor (allows save instruction execution at lowest voltages)
- Dual mode quartz and RC oscillators:
  - *1 MHz – 10 MHz RC*
  - *32768 Hz crystal or external clock source*
- 8-bit CoolRISC architecture
  - *16 registers*
  - *2 clock per instruction*
  - *8x8bit hardware multiplier*
- Power-On-Reset and watchdog
- Various Flash memory size:
  - *2k x 22 bit (5.6k Byte)*
  - *4k x 22 bit (11.2k Byte)*
  - *8k x 22 bit (22.5k Byte)*
- Fully static 512B or 256B RAM, Low power 12B RAM,
- Dual port 4B RAM
- Internal and external interrupts
- Frequency generator
- PWM functions
- 8/16-bit timers
- Prescaler:
  - 10-bit RC divider
  - 15-bit crystal divider
- SPI interface, UART programmable by software
- Small 24-pin TSSOP and SO packages (leadfree)

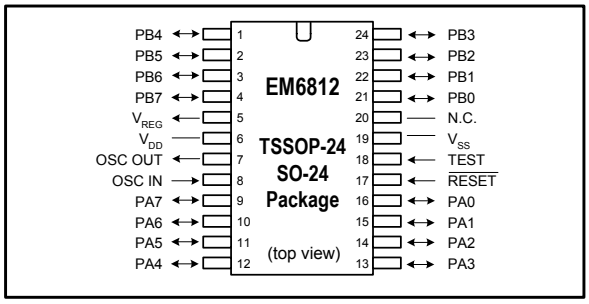
Block Diagram



Tools & Services

- Easy to use emulator with full debug functions, full peripheral integration, C-compiler
- Windows-based software programs
- Programmer from different vendors
- Dedicated team of engineers for outstanding support

Pinout Configuration



Typical Applications

- Metering
- Heat Cost Allocation
- Smoke detector
- Security
- Body care
- Sports
- Computer peripherals, Bluetooth chipset

A COMPANY OF THE SWATCH GROUP

1 EM6812 at a glance

Power supply
- Low power architecture
- Voltage regulator for internal logic supply
- External regulator capacitor

CPU
- 8 bit CoolRisc 816L Core
- 16 internal registers
- 4 hardware subroutine stacks
- 8 bit hardware multiplier
- refer also to the CR816L reference manual

ROM / Flash
- ROM 4096 Instructions = 11.26 Kbytes
- Flash 8192 Instructions = 22.5 Kbytes

RAM
- 512 x 8 bit static SRAM (for 8k Instructions)
- 256 x 8 bit static SRAM (for < 4k Instructions)
- low voltage ram data retention

Low power RAM, 12 Byte
- for lowest power calculations

Dual Port RAM, 4 Byte
- Data IO on port B, Control on port A

Operating modes
- Active mode: CPU and peripherals are running
- Standby mode: CPU halted, peripherals on
- Sleep mode: no clocks, reset state
- Wake Up from port A inputs

Resets
- Power On Reset
- Reset from watchdog timer
- External Reset Input
- Brown Out
- Reset with Port A reset combination
- Reset Flags to identify the reset source

Oscillator XTAL 32kHZ
- Oscillation clock pre-divider (1 sec)
- External clock low frequency input

Oscillator RC
- internal RC oscillator
- External clock high frequency input
- Freq. Trimming register
- 1MHz or 10MHz Clocks
- stable over temperature and voltage

Prescaler's
- 2 Prescaler for RC and Xtal Oscillators
- input clock software selectable
- fix interval IRQ's (RTC and others)
- clock source to other peripherals
- Divider capture, 8 MSB's

Parallel In/Output Port A
- 8 bit wide direct input read
- all functions bit-wise configurable
- Input , output
- debouncer
- IRQ on pos. or neg. edge
- Pull-up, pull-down or no pull selectable
- Freq. Input for timer
- Input combination reset
- CMOS or NCH. Open Drain outputs

Parallel In/Output Port B
- 8 multipurpose I/O's
- 8 bit wide direct input read
- all functions bit-wise configurable
- 4 high current outputs
- Input , output
- Pull-up, pull-down or no pull selectable
- CMOS or NCH. Open Drain outputs
- special function: Serial Interface I/O's, DP RAM

Serial Interface SPI
- 3 wire serial Interface, Sclk, Din, Dout

Timer (4 x 8 bit, or 2 x 16 bit)
- 8 (16) bit wide, Zero-Stop and Auto-Reload mode
- External signal pulse width measurement
- PWM generation
- Event Counter
- IRQ requests

Watchdog timer
- generation of watchdog reset after time out

Interrupt
- external IRQ's from Port A, Comparator
- internal IRQ's from Timer, Prescaler

SVLD
- 8 levels supply voltage level check

Brown Out
- On-chip Brown-Out detection, reset state
- Power check at Startup

Table of contents

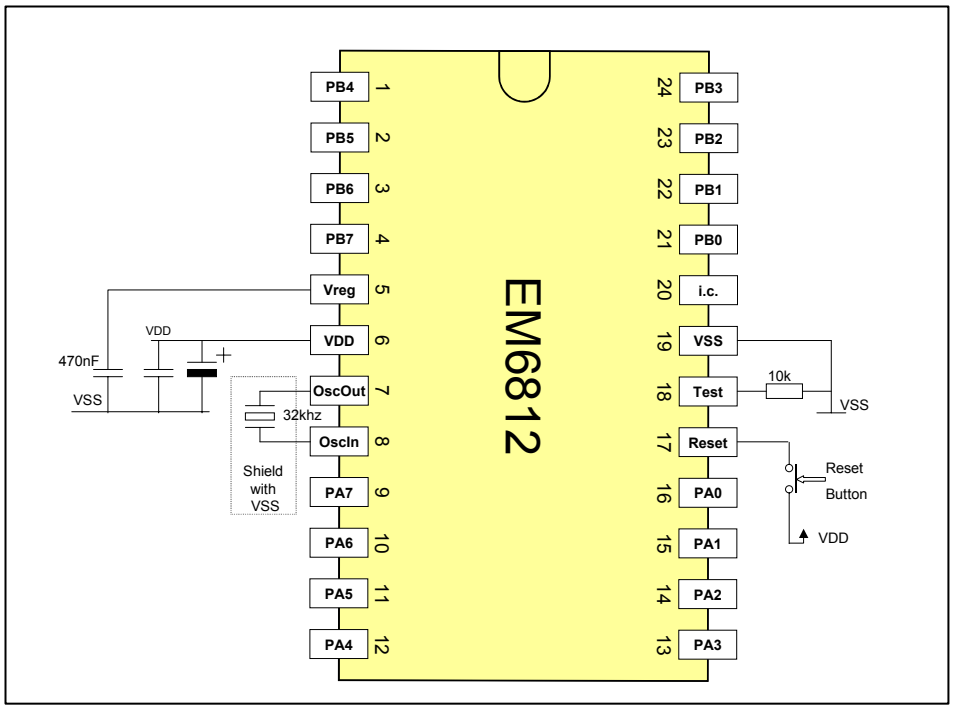
2 Circuit Connectivity

The EM6812 has the same pin-out in both the SO24 and TSSOP24 pin package.
Minimum connectivity includes the power supply on V_{SS} and V_{DD}, a capacitor on on Vreg, and de-coupling capacitance on V_{DD}. Circuit reference terminal (substrate) is on V_{SS}.

The 32kHz XTAL is only needed for systems requiring low frequency Crystal operation.

The integrated supply voltage regulator filters supply noise and allows lowest power peripheral operations. For proper operation, a capacitor (470nF minimum) must be connected to the regulator's VREG terminal. This terminal must not be used for any other outside connection.

Figure 1: Sample minimum connectivity



Note:

- ALL circuit IO's (except OscIn) are on V_{DD} level. OscIn terminal is only used in conjunction with an active Crystal oscillator. Its input voltage must never exceed the Vreg voltage.
- The crystal oscillator should be shielded with V_{SS} to keep noise away.
- When using the Crystal oscillator PA[7] and PA[6] should preferably used as static inputs only to avoid noise coupling on the OscIn and OscOut high impedance inputs.

2.1 Terminal usage

Table 1. Circuit terminals

Pin	Name		Description	SPI & PWM	Dual Port RAM	Programming connections
1	PB4	IO	Standard IO		DPRData[4]	
2	PB5	IO	Standard IO	SCLK	DPRData[5]	SCLK
3	PB6	IO	Standard IO	SOUT	DPRData[6]	
4	PB7	IO	Standard IO	SIN	DPRData[7]	SDIO
5	Vreg	Sup	Connect min 470nF			
6	V_{DD}	Sup	Main power supply			V_{DD}
7	OscOut	In	Crystal, External LF Clock input, Data input			
8	OscIn	In	Crystal only connection			
9	PA7	IO,	Standard IO, IRQ, timer start & clock,			
10	PA6	IO,	Standard IO, IRQ, timer start & clock			
11	PA5	IO,	Standard IO, IRQ, timer start & clock			
12	PA4	IO	Standard IO, IRQ, timer start & clock			
13	PA3	IO	Standard IO, IRQ, timer start & clock		ExtAdr[1]	
14	PA2	IO	Standard IO, IRQ, timer start & clock		ExtAdr[0]	
15	PA1	IO	Standard IO, IRQ, timer start & clock		ExtWEn	
16	PA0	IO	Standard IO, IRQ, timer start & clock		ExtCen	
17	Reset	In	Reset input, active high with internal pull-down resistor			
18	Test	In	EM test and Program high Voltage See note.			VPP
19	V_{SS}	Sup	Reference terminal			V_{SS}
20	i.c.		Used for EM test purposes, internally connected. Must not be connected externally			
21	PB0	IO	Standard IO, drive 2	PWM	DPRData[0]	
22	PB1	IO	Standard IO, drive 2	PWM	DPRData[1]	
23	PB2	IO	Standard IO, drive 2	PWM	DPRData[2]	
24	PB3	IO	Standard IO, drive 2	PWM	DPRData[3]	

Notes:

Connection on Test pin:

- On Flash device, either connect to V_{SS} via a 10kOhm resistor (as close as possible to V_{SS} pad) or foresee a jumper for programming (V_{SS} or VPP connection)

Connection on pin i.c. (i.c stands for internally connected).

- This pin is used in EM test modes. No external connection must be made on this pin.

2.2 Programming connections

The EM6812 embedded Flash program memory is programmed using standard microcontroller programmers available from 3rd parties.

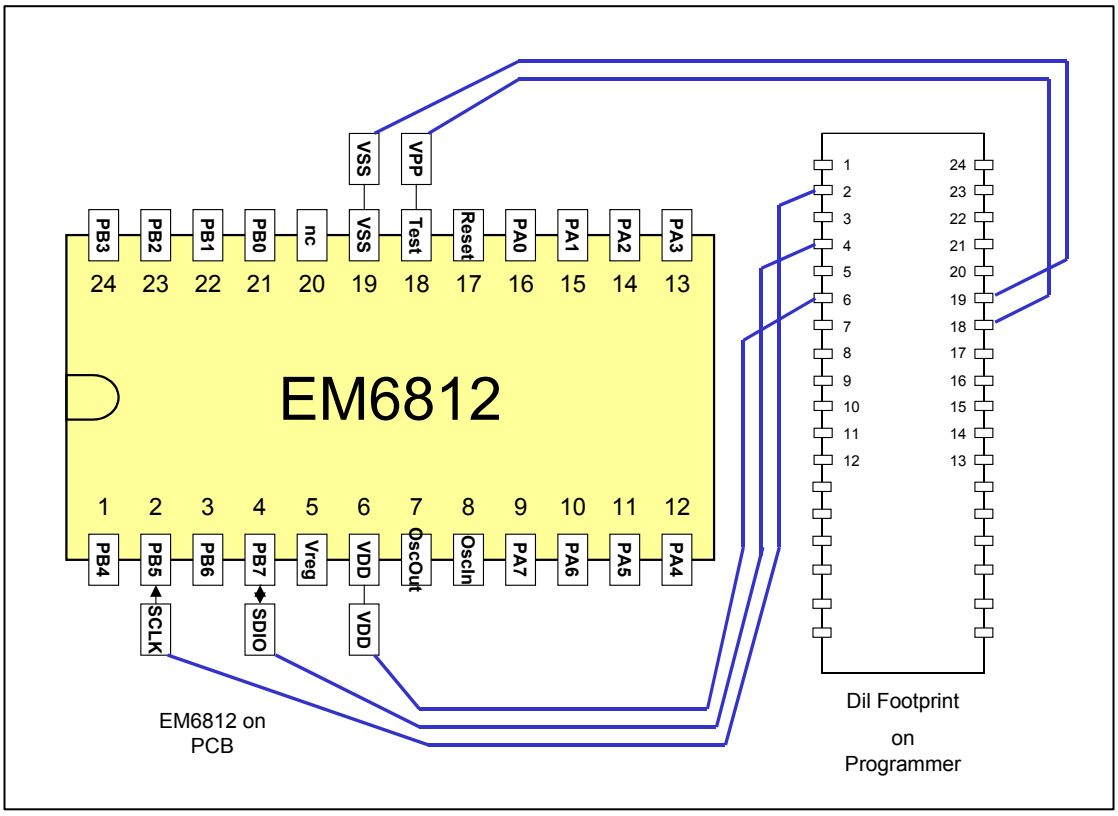
Programmers which currently support the EM6812:

- ELNEC (SmartProg / LabProg / JetProg) www.elnec.com
  For an updated list please consult: www.elnec.com/sw/dev_html/em_microelectronic_dev.htm

Erase/Write:The programmer allows to erase/write the whole program memory at once (bulk erase). Typical erase time is 20ms for the whole Flash memory. Erase is immediately followed by write (writing 1 intruction after the other). Typical write time is 60µs/word.
Code protection: The program memory content can not be read back, instead a checksum (CRC) is generated and compared with the programmer's CRC value.
Last Address read: The very last address of the program memory may be read back. (code identification)

Connection into the DIL connector is 1 to 1, DIL pin 1 goes to SO or TSSOP pin1 and so on. An adapter is needed for the SO and TSSOP packages.
On-board programming is possible by connecting the 5 programming terminals directly onto the PCB. This can be done with a DIL to PCB connecting cable (not furnished) or by using the on-board programming connector, which is present on some of the programmers.

Figure 2. On-board programming with DIL-adapter cable



The programming interface terminals PB5 and PB7 are automatically configured in input mode as soon as Test terminal goes high. This allows the programmer to download the programming setup into the circuit. As soon as a valid programming mode is recognized the circuit will enter a special state and allow only Flash programming and CRC check to be done.
During programming the PortA is configured as output driving V_{SS} level, PB[4:0] is in input state, PB6 is output.

3 Operating modes

The EM6812 has 3 main operation modes.

- Active Mode - CPU up and running, Instruction executing, Periphery clocked.
- Standby Mode - CPU stopped, No Instruction execution, Periphery clocked.
- Sleep Mode – CPU reset, No Instruction execution, Periphery stopped.

Within these operating modes different submodes exist with different clock selections to allow lowest power consumption for all given cases. Please refer to the clock-managing unit and the specif peripherals for all clock selections possibilities. When not activated the embedded peripherals are not clocked and therefore do not add any unnecessary power consumption.

Figure 3. Operating modes transition

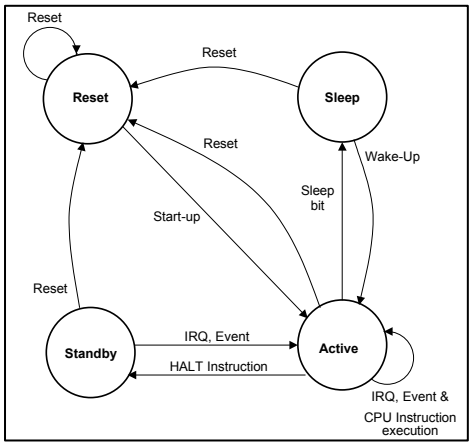


Table 2. Mode dependent peripheral status

Peripheral block	Active mode	Standby mode	Sleep mode
CPU	Running at defined Clock frequency	Stopped	Reset
Clock source	Running at defined Clock frequency	Running at defined Clock frequency	All internal clock sources are off
SVLD	Software selectable	Software selectable	Disabled
BrownOut	Software selectable	Software selectable	Disabled
POR	On	On	On
Prescalers	On	On	Off – no clock, retain value
Interrupts/ Events	Possible	Possible	Wake-up only
Watchdog timer	Software selectable	Software selectable	Off – no clock, retain value
Timer	Software selectable	Software selectable	Off – no clock, retain value
Ports	Software selectable	Software selectable	Retain state, input debouncers are by-passed.
RAM	Software selectable	Software selectable	Retains value

3.1 Active mode

The active mode is the default mode after poewr-up. The CPU executes instructions one after the other. All peripheral settings are performed in this mode before eventualy switching to low power modes. Any interrupt arriving will immediately at the next instruction branch into the interrupt vector.

3.2 Standby Mode

The standby mode is the commonly used low power dissipation mode. During standby the CPU instruction execution is halted but all peripheral circuitry is still clocked. Any interrupt or event will bring the circuit back into active mode on the next active CPU clock edge. Standby mode is entered with the CPU HALT instruction.

3.3 Sleep Mode

This is the lowest power possible mode. Circuit operation is stopped (no clock anymore) most peripheries retain their value. Exceptions are:

- The debouncer circuits are by-passed to allow reaset or wake-up.
- SVLD and Brownout function are disabled to have minimum power dissipation.
- CPU is in reset state.

To go into sleep mode one needs first to set bit **SleepEn** = ‘1’, then **Sleep** = ‘1’. While **SleepEn** =’0’ one can not write the **Sleep** bit.

Resume from sleep mode by either wake-up on pre-specified port A combination (refer to 7.4) or by any reset. Inspection of the reset status register allows determining the restart origin.

3.4 System registers

Functions	Register name	Basic function
SleepEn and Reset status	RegResStat	Reset Flags to identify reset source
Sleep mode external Resets and RC selections	RegSys1	Sleep bit and main CPU oscillator settings, reset pad configuration
Xtal and CPU clock selections	RegSys2	High and low frequency clock selections, Xtal enable

Table 3. System registers

RegResStat		0x12			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	ResetPadFlag	0	POR	RC	Flag for Reset terminal, clear by write'0'
6	ResetWDFlag	0	POR	RC	Flag for watchdog reset, clear by write'0'
5	ResInpPAFlag	0	POR	RC	Flag for PortA input reset ,clear by write'0' Flag is also set by BrownOut in case of DisResInp='1'
4	ResBwnOutFlag	0	POR	RC	Flag for BrownOut reset, clear by write'0'
3	SleepEn	0	POR	R/W	Enables to write the Sleep bit
2	EnDebResPad	0	RegSysSlp	R/W	1: Debounced reset input 0: Direct reset input
1	CkDebResPad	0	ResSys	R/W	1: high speed clock (Pr1Ck[13], 8kHz) 0: low speed clock (Pr1Ck[8], 256Hz)
0	DatOscOut	-	-	R	Read data on Oscout terminal if XTAL off

RegSys1		**0x10**			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	Sleep	0	ResSys	R/W	Put the circuit in sleep mode if = '1'
6	DisResetPad	0	POR	R/W	Disable the input pad reset if = '1'
5	DisResInp	0	POR	R/W	Disable the port A reset input if = '1'
4	FlagXtal	0	ResMain	R	Xtal cold start flag, Xtal ready if = '1'
3	OPTCldStart[1]	0	POR	R/W	Xtal cold start duration: '00' = 1s, '10' = 3/4s, '01'=1/2s, and '11'=1/4s.
2	OPTCldStart[0]	0	POR	R/W	
1	FreqRange	0	ResSys	R/W	RC osc. frequency range selection: '1'=10MHz '0'=1MHz
0	EnRC	1	ResSys	R/W	Enable RC oscillator if = '1'

RegSys2		**0x11**			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	EnXtal	0	ResMain	R/W	Enable Xtal oscillator if = '1'
6	SelExtHFck	0	ResMain	R/W	Enable external clock instead of RC if = '1'
5	SelHFckSource	0	ResMain	R/W	Select external clock PA4 if '0', PA5 if '1'
4	SelExtLFck	0	ResMain	R/W	Enable external clock instead of Xtal if = '1'
3	--	0	--	R/W	Not used, read always '0'
2	Sel32k	0	ResMain	R/W	CPU clock '1'=low freq (F1) '0'=high freq (F2)
1	RCDiv[1]	1	ResMain	R/W	HF domain division factor for F2: '00'=1, '01'=2 , '10'=4, '11'=8
0	RCDiv[0]	1	ResMain	R/W	

4 Program Memory

All instructions to be executed are stored in the Program Memory, all general purose data as well as peripheral registers values are stored in a separate data memory (see chapter 5). This special Hrvard-RISC like architecture gives the core the ability to read operands in the data memory simultaniously with one instruction fetch.

Maximum program memory size of the EM6812 is 22.5 kBytes. Each Instruction is 22 bits wide, which gives a total of 8192 Instruction words.

Program Memory is implemented as Flash memory (EM6812-Fx)

The device is delivered with different program memory sizes from 8192 down to 2048 instruction words. Please refer to the ordering information section for the different memory types and sizes.

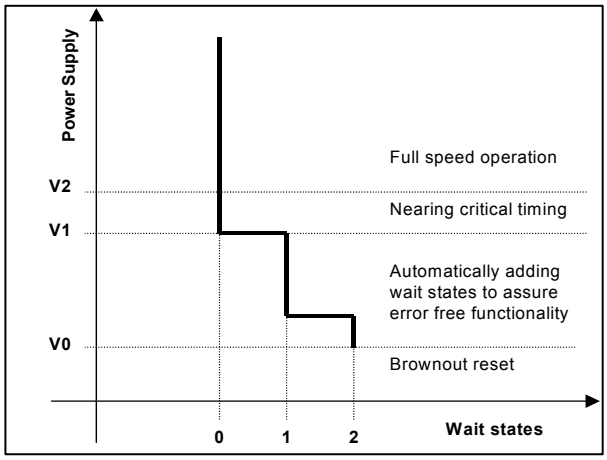
4.1 Memory miss

The unique Memory Miss feature of the EM CoolRISC products allows operating with high-speed peripheral clocks even at low voltage power supplys

The Memory Read Monitor (= memory miss) is an important feature ensuring correct program execution while allowing graceful performance reduction at low voltage conditions. The access times of memory cells are dependent mainly on the supply voltage and the temperature. A general case is shown Figure 4 showing wait states automatically added over the power supply.

By monitoring each program memory access it can be assured that all memory accesses are good. If necessary, wait- states will be added. As the supply voltage reaches V_2 an interrupt is generated to signal that the Memory Read Monitor will soon start adding wait states to ensure accurate program execution. To be sure to always be running error free a standard processor would have to stop its activity at this point. Using this warning the processor can, for example, turn off not absolutely required functions to reduce the power supply load. At V_1, the processor starts automatically adding the wait states necessary to ensure proper operation. The processor will then continue to operate flawlessly as the power supply voltage level continues to descend down to the Brownout voltage, V_0 in this diagram. As the voltage falls the number of wait states will increase as necessary to assure that the memory is read correctly at each access.

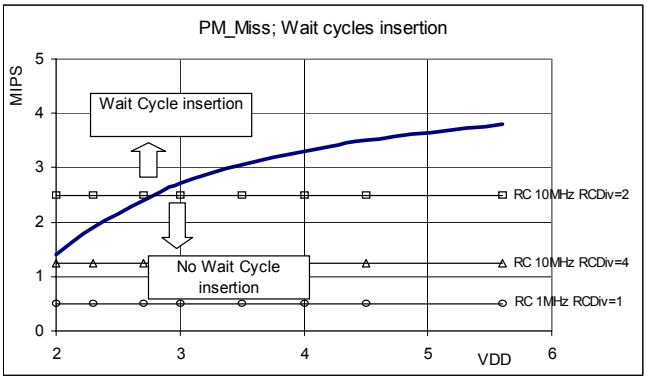
Figure 4. Read Monitor funtion



As the Memory Read Monitor is a hardware function that uses an actual memory cell as its standard it takes into account all of the factors that influence the real memory read time of the memory array. It works accurately for all combinations of voltage, frequency and temperature. This guarantees stable processor operation for graceful performance reduction.

The addition of the Memory Read Monitor allows extending the working range of the application from V_2 down to V_0 without compromising the operating security in any way.

Figure 5. Wait insertion versous Power supply (V2 voltage)



The program memory miss interrupt is asserted as long as the condition is true. Above 'Figure 5. Wait insertion versous Power supply (V2 voltage) ' shows actual values for memory miss interrupt generation.
Typically no PM Miss is generated if running on system frequencies of 2.5Mhz (RC 10MHz RC_div=4) or below on the Flash based circuits.
While running on RC=10MHz and RCDiv=1 (5Mips) the PM_Miss_skip interrupt may be triggered by every Flash access. If the high frequency 5MHz clock is needed for the peripheral blocks, the user should mask the PM_Miss_skip IRQ.

Table 4. Memory miss interrupt generation

Interrupt source	Priority	IntCtrl connection
PM_Miss_skip	0	Int0[0]

5 Data Memory

The data memory is connected to the CoolRisc core via an 8-bit wide bi-directional data bus. It contains:

- max 512Bytes of fully static RAM
- Dedicated peripheral data registers for timers ports, etc.
- 16 Bytes of general purpose registers. (12 Bytes LP RAM, 4 Bytes DPR)

Table 5. Data memory mapping

Data Memory Address ranges	Description	Data memory page
0x0257 (8 kInstr) 0x0157 (4 or 2k Instr)	SRAM 512 Byte (8k Instructions)	Page 1 Indexed addressing
0x0060	256 Byte (4 or 2k Instructions)	Page 0 (direct and indexed addressing)
0x0010	All Peripherals (Timers Ports, Configurations, Interfaces, etc)	
0x0000	Dual Port RAM Low Power RAM	

All peripherals and part of the SRAM are accessible with any addressing modes of the CoolRisc instruction set.
The portion of SRAM, which is on Page 1, is addressed with any of the indexed addressing modes.

5.1 SRAM

The SRAM size is adapted to the program memory size.

- 512 Byte SRAM: All Versions with 8k Instruction Memory
- 256 Byte SRAM: All Versions with 4k or 2k Instruction Memory.

The SRAM has no reset functions, therefore is should be initialized before storing any variables.

Table 6. SRAM mapping (4k or 2k Instructions Program Memory Version)

Name	Reset	Dec	Hex	
RAM 96 bytes	xx	351	0x015F	Indexed addressing
	:	:	:	
	xx	256	0x0101	
RAM 160 Bytes	xx	255	0x0100	Direct and indexed addressing
	:	:	:	
	xx	96	0x0060	

Table 7. SRAM mapping (8K Instructions Program Memory Version)

Name	Reset	Dec	Hex	
RAM 352 bytes	xx	607	0x025F	Indexed addressing
	:	:	:	
	xx	256	0x0101	
RAM 160 Bytes	xx	255	0x0100	Direct and indexed addressing
	:	:	:	
	xx	96	0x0060	

5.2 General Purpose Registers, 16 Bytes

A total of 16 general purpose 8-bit registers are available. 12 of these registers are realized as low power RAM to store and recall frequently accessed variables with minimum power whereas the power saving is realized by minimizing the parasitic capacitance which are inherent to large memories. The other 4 bytes are shared with the Dual Port RAM function. These registers are reset to '00' by POR only.

Table 8. General purpose register mapping

Name	Register Name	Res	Dec	Hex	
Low Power RAM (General purpose registers)	**LPRam0**	00	0000	0x0000	Direct and indexed addressing
	…	:	:	:	
	LPRAM11	00	0011	0x000B	
LP Ram shared with Dual Port RAM	**DPRam0**	00	0012	0x000C	
	…	:	:	:	
	DPRam3	00	0015	0x000F	

5.3 Dual Port RAM

The DP RAM is a memory block, which allows data, read and write, accesses from either the CPU core or an external processor in a total asynchronous way. The Dual Port RAM external data IO is mapped on Port B, the control signals on Port A. The occurrence of possible access conflicts is flagged to the CPU with 2 Interrupts. Priority is given to the external access.

Table 9. DPR Port mapping

DP RAM external connection	Port mapping	Function
ExtDat[7:0]	PB[7:0]	External bi-directional data bus
ExtAddr[1:0]	PA[3:2]	External address
ExtWen	PA[1]	External write or Read access selector
ExtCen	PA[0]	External chip enable, validates the access

Setting the bit **EnDualRam** = '1' in register RegCfgPB enables the DPR function. It is still possible to use the Port B as a standard port even if the DPR function is enabled. Only while input ExtWen is low and ExtCen is high the port B is forced as output. In all other cases the port B configuration is given as defined by the port b configuration registers. Please refer also to the Port B description.
The terminal configuration of the DPR control inputs on port A is totally free.

Note:
The port B terminals must not be left floating while EnDualRam='1'. One may use the integrated pull resistors id drive condition is not sure.

5.3.1 CPU R/W access to DPR

CPU Read and Write access are performed the same way as it does for all other general-purpose registers. No special precautions need to be done as long as EnDualRam is not set. When EnDualRam is set Conflicts with external access may occur. Such conflicts are flagged with interrupts to the CPU. Refer to 5.3.4 Conflict handling.

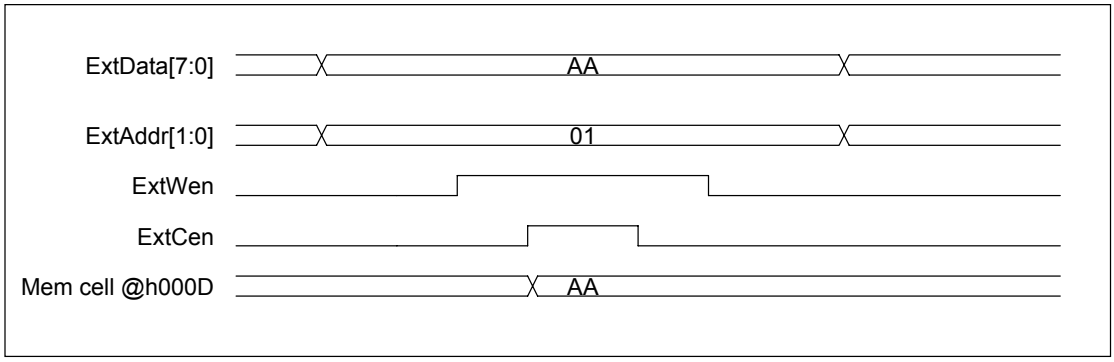
Table 10. DPR memory address mapping

Register	Bit_Names	ExtAddr[1:0] on port A	Res	Reset by	Adr(Dec)	Adr(Hex)
DPRam0	DPR0[7:0]	00	00	POR	12	0x0C
DPRam1	DPR1[7:0]	01	00	POR	13	0x0D
DPRam2	DPR2[7:0]	10	00	POR	14	0x0E
DPRam3	DPR3[7:0]	11	00	POR	15	0x0F

5.3.2 External Write Access to DPR

A write access from external uses the control signals ExtWen, ExtCen and the address signals ExtAddr[1:0] from the port A to write a value given on the port B into the DPR location inside the EM6812.
Setting **EnDualRam**='1' in RegCfgPB and then configure port A and port B to allow an external circuitry to drive the necessary control and data lines for DPR operation.

Figure 6. DPR, Write into the EM6812

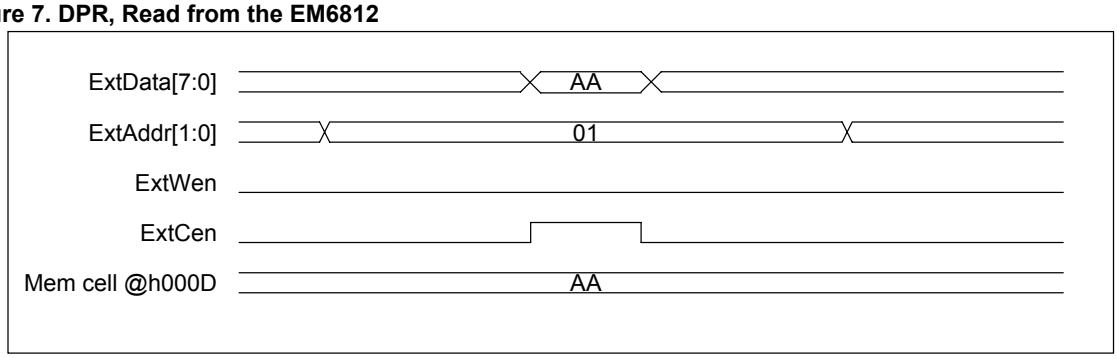


Data are written into the DPR on the rising edge of ExtCen. While ExtWen is high the selected address location is write protected against CPU writes.

5.3.3 Read Access from DPR

A Read access from external uses the control signals ExtWen, ExtCen and the address signals ExtAddr[1:0] from the port A to output the addressed DPR value on the port B.
First set the necessary port A configuration to allow an external circuitry to drive the necessary. As soon as ExtCen becomes high (and ExtWen is low) the port B will become output and drive the currently selected DPR data value.

Figure 7. DPR, Read from the EM6812



While ExtCen is high the selected DPR memory location is write protected to guarantee the value read by the external device.

5.3.4 Conflict handling

The external device always has the priority against the CPU read or write operations. 2 interrupts are used to flag the occurrence of possible conflicts. Conflicts may only occur by simultaneous access to the same memory location by the external device and the CPU.

Table 11. DPR interrupt flags for conflict handling

CPU operation	External operation	Conflict description,	Interrupt signification	IrqDR[1]	IrqDR[0]
Read	Write	External write during CPU read.	The data read by the CPU may be corrupted	**0**	**1**
Write	Read	External read during CPU write	The CPU write operation may have failed	**1**	**0**
Write	Write	Concurrent writes	The CPU write may have failed.	**1**	**1**

Table 12. DPR interrupt mapping

Interrupt source	Priority	Interrupt controller connection
IrqDR[1:0]	2	Int2[7:6]

5.3.5 Register overview

Table 13. DPR registers

RegCfgPB		0x32			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	EnDualRAM	0	ResSys	R/W	Enable the Dual Port RAM
6	*EnSPI*	*0*	*ResSys*	*R/W*	*Enable the Serial Interface function*
5	*EnSig1*	*0*	*ResSys*	*R/W*	*Connecting the internal Signal1 on PB[0]*
4	*EnSig2*	*0*	*ResSys*	*R/W*	*Connecting the internal Signal2 on PB[1]*
3	*EnSig3*	*0*	*ResSys*	*R/W*	*Connecting the internal Signal3 on PB[2]*
2	*EnSig4*	*0*	*ResSys*	*R/W*	*Connecting the internal Signal4 on PB[3]*
1	-	--	--	*R*	*Reads '0'*
0	-	--	--	*R*	*Reads '0'*

RegDPRAM0		0x0C			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	DPR0[7:0]	00	POR	R/W	Dual port RAM location 0

RegDPRAM1		0x0D			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	DPR1[7:0]	00	POR	R/W	Dual port RAM location 1

RegDPRAM2		0x0E			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	DPR2[7:0]	00	POR	R/W	Dual port RAM location 2

RegDPRAM3		0x0F			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	DPR3[7:0]	00	POR	R/W	Dual port RAM location 3

6 CPU Core, CR816L

The µProcessor Core CR816L is a true low power RISC Core including a 1 instruction cycle 8x8 hardware multiplier. Its main features are described below.

- 8 bits RISC register-memory processor based on a Harvard architecture
- 16 CPU internal registers (Accu, general purpose, Index, offset, status)
- 8b x 8b internal hardware multiplier
- 3 stage pipeline architecture (no delay slots or branch delays)
- 176 Kbytes max Program Memory size (64 Kinstruction, 22 bit wide) → EM6812 uses max 22k Instruction
- 64 Kbytes max Data Memory size (organized in 256 x 256 Kbytes pages) → EM6812 uses max 512Byte
- 8 max hardware subroutines and unlimited software subroutines → EM6812 uses max 4 hw subroutines
- 5 different addressing modes
  - direct addressing
  - indexed addressing with immediate offset
  - indexed addressing with register offset
  - indexed addressing with post-incrementation of the offset
  - indexed addressing with pre-decrementaion of the offset

.

Table 14: CR816L Instruction set

Mnemonic	ALU	Instruction
ADD	yes	Addition.
ADDC	yes	Addition with carry.
AND	yes	Logical AND.
CALL	no	Jump to subroutine.
CALLS	no	Jump to subroutine, using ip as return address.
CMP	yes	Unsigned compare.
CMPA	yes	Signed compare.
CMVD	yes	Conditional move, if carry clear.
CMVS	yes	Conditional move, if carry set.
CPL1	yes	One's complementation.
CPL2	yes	Two's complementation.
CPL2C	yes	Two's complementation with carry.
DEC	yes	Decrementation.
DECC	yes	Decrementation with carry.
HALT	no	no Halt mode selection
INC	yes	Increment.
INCC	yes	Increment with carry.
Jcc	no	Conditional jump.
MOVE	yes	Data move.
MUL	yes	Unsigned multiplication.
MULA	yes	Signed multiplication.
NOP	no	No operation.
OR	yes	Logical OR
POP	no	Pop ip index from hardware stack.
PUSH	no	Push ip index onto hardware stack.
RET	no	Return from subroutine.
RETI	no	Return from interrupt.
SFLAG	yes	Save flags.
SHL	yes	Logical shift left.
SHLC	yes	Logical shift left with carry.
SHR	yes	Logical shift right.
SHRA	yes	Arithmetic shift right.
SHRC	yes	Logical shift right with carry.
SUBD	yes	Subtraction (op1 - op2).
SUBDC	yes	Subtraction with carry (op1 - op2).
SUBS	yes	Subtraction (op2 - op1).
SUBSC	yes	Subtraction with carry (op2 - op1).
TSTB	yes	Test bit.
XOR	yes	Logical exclusive OR.

Please refer to the CoolRISC 816L 8 bit Microprocessor Core Hardware and Software Manual Version 1.1 dated Mai 2002

7 Reset Controller

7.1 Basic features

Internal and external reset sources are handled within the RESET controller. All these reset are used to put the device in a defined state at power-up, user request or system exceptions. The reset sources are flagged, and may be inspected by the CPU after the reset event to allow detection of the reset source.
Any Reset will keep the CPU in reset for state for approx. 300µs once the reset condition released.

Internal sources
- Power on Reset with powercheck at start-up
- BrownOut detection as voltage supervisory function
- Watchdog reset with protected disable key

External sources
- Reset Pad (User reset)
- Input reset combination on PortA

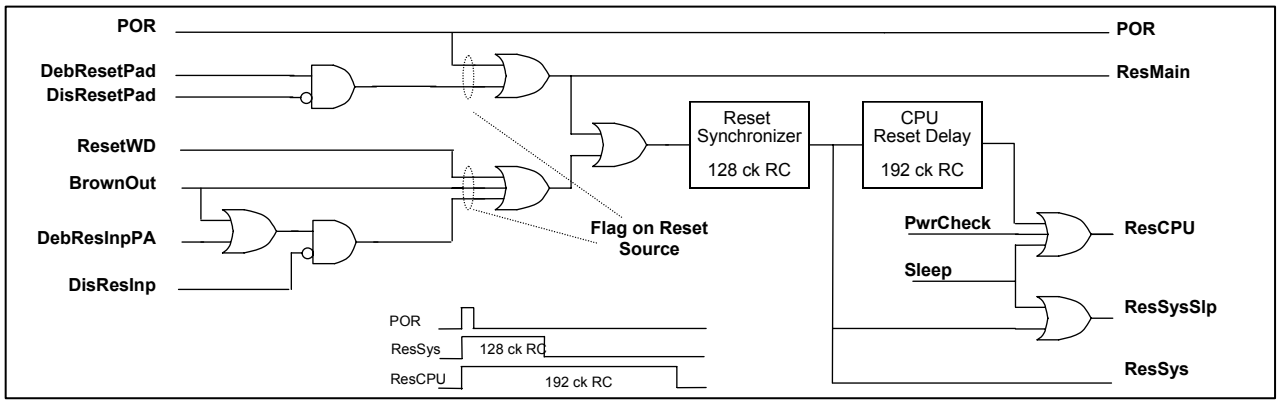
All these sources may initialize or re-initialize either the whole or part of the circuit. These sources are:

Table 15: Reset sources

Function	Signal	Activated reset signals
POR	POR	POR, ResMain, ResSys, ResSysSleep, ResCPU
Reset Pad	DebResetPad	ResMain, ResSys, ResSysSleep, ResCPU
PowerCheck	PwrCheck	ResCPU
Port A Input Reset	DebResInpPA	ResSys, ResSysSleep, ResCPU
Watchdog Reset	ResetWD	ResSys, ResSysSleep, ResCPU
Brownout Reset	BrownOut	ResSys, ResSysSleep, ResCPU
Sleep	Sleep	ResSysSleep, ResCPU

After every reset the circuit restart with RC clock 1MHz as the only clock selection. The periphery is released from reset after 128 RC clocks (Reset synchronizer) and the CPU 64 RC clocks later (CPU Reset Delay).

Figure 8. Reset controller architecture



POR: initializes the whole circuit;
ResMain: initializes the whole circuit except the ResetPad configuration
ResSys: initializes all internal registers, does nor reset the terminal configuration settings (I.e. pull resistors)
ResSysSleep: Enables the path to be able to come out of sleep.
ResCPU: Initializes the CPU
Signals POR, ResMain, ResSys, ResSysSleep, ResCPU are the actual reset signals which initialize the different latches and registers. Please refer to the different register tables to know the reset source for each register bit.

Note:
- BrownOut reset will set the BrownOut flag but also the ResInpPA flag if DisResInp is set.
- If the power up is faster than the BrownOut Filter (~10ms) no Reset Flag will be set. (allwing POR identification)
- In case of slow power up, both the BrownOut and the ResPAInp flag will show. The bit SleepEn (reset by POR only) may be used to distinguish between a slow power up and a ‘normal ‘ BrownOut condition.

7.1.1 Reset functions registers

Table 16: Reset register overview

Functions	Register name	Basic function
Reset status	RegResStat	Reset Flags to identify reset source
Reset Pad configuration	RegResStat	Direct or debounced input and debouncer clock selection
	RegSys1	Disable the reset pad input
BrownOut	RegAnaCfg	Enabling the Brownout function (default: On)
Reset PA input configuration Refer to PortA description	RegCfgPA	Reset and wake-Up system configuration,
	RegMskRstWkUp	Combination mask selection
	RegCmbKey	Reset or wake-up key
Watchdog reset	RegWDkey	Watchdog setup and keylock
	RegWDSys	

Table 17: Reset registers detail

RegResStat		0x12			Reset Flags to determine the reset source
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	ResetPadFlag	0	POR	RC	Flag for Reset terminal, clear by write'0'
6	ResetWDFlag	0	POR	RC	Flag for watchdog reset, clear by write'0'
5	ResInpPAFlag	0	POR	RC	Flag for PortA input reset ,clear by write'0' Flag is also set by BrownOut in case of DisResInp='1'
4	ResBwnOutFlag	0	POR	RC	Flag for BrownOut reset, clear by write'0'
3	SleepEn	0	POR	R/W	Enables to write the Sleep bit
2	EnDebResPad	0	ResSysSlp	R/W	'1': Debounced reset input '0': Direct reset input
1	CkDebResPad	0	ResSys	R/W	'1': high speed clock (Pr1Ck[13], 8kHz) '0': low speed clock (Pr1Ck[8], 256Hz)
0	DatOscOut	-	-	R	Read data on Oscout terminal if XTAL off

RegSys1		0x10			User reset handling
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	Sleep	0	ResSys	R/W	Put the circuit in sleep mode if = '1'
6	DisResetPad	0	POR	R/W	Disable the Input pad reset if = '1'
5	DisResInp	0	POR	R/W	Disable the port A reset input if = '1'
4	FlagXtal	0	ResMain	R	Xtal cold start flag, XTal ready if = '1'
3	OPTCldStart[1]	0	POR	R/W	Xtal cold start duration: '00' = 1s, '10' = 3/4s,
2	OPTCldStart[0]	0	POR	R/W	'01'=1/2s, and '11'=1/4s.
1	FreqRange	0	ResSys	R/W	RC osc. frequency range selection: '1'=10MHz 0=1MHz
0	EnRC	1	ResSys	R/W	Enable RC oscillator if = '1'

RegWDSys		0x3B			Watchdog setup register including key lock
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	WDClear	0	--	R/W	Clear the WD counterby writing '1' =Reset; '0'=no action. Read is always '0'
6	EnWD	1	ResSys	R/W	Enable watchdog = '1'. The key word must be loaded prior to force EnWD='0'
5	WDVal[1]	0	ResSys	R	WD counter status, MSB-bit
4	WDVal[0]	0	ResSys	R	WD counter status , LSB-bit
3	WDClkSel	0	ResSys	R/W	Clock selection. '0' = Pr1Ck[0] (typ 1Hz) '1' = Pr1Ck[7] (typ 128Hz)
2	WDKeyLock[2]	0	ResSys	R/W	Unlock the key work if = '111'
1	WDKeyLock[1]	0	ResSys	R/W	
0	WDKeyLock[0]	0	ResSys	R/W	

RegWDKey		0x3C			Watchdog key register
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	WDKey	00	ResSys	R*/W	WD key word to allow the software disabling the watchdog if = '10010110' bin

R*: The WDkey bits always read '00' if no valid WDKey or if WDKeyLock is locked

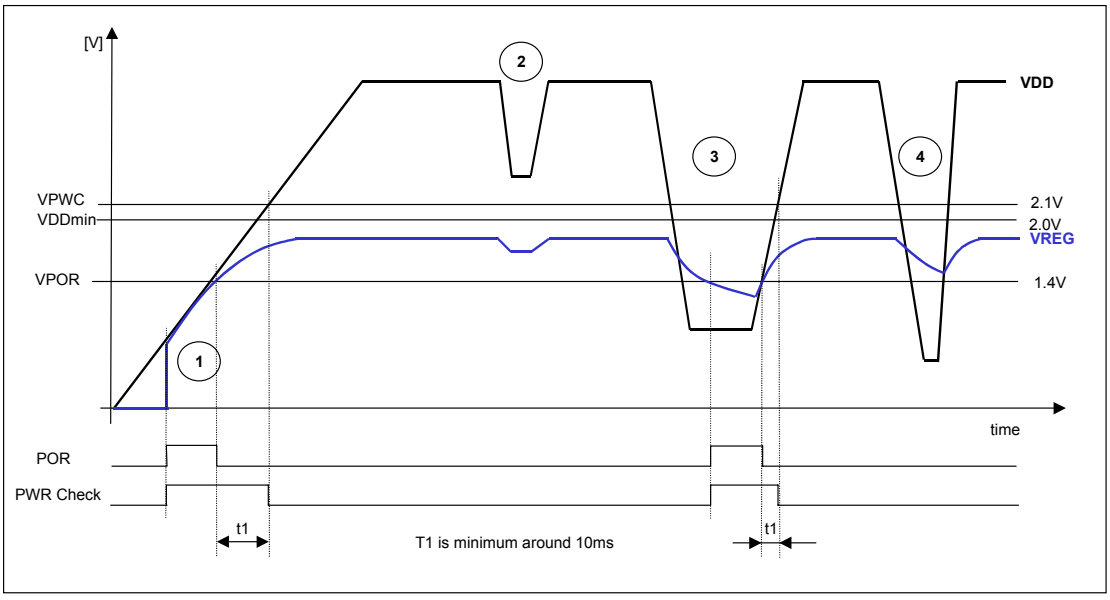
RegAnaCfg		0x20			BrownOut and SVLD handling
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	*EnSvld*	*0*	*ResSys*	*R/W*	*Enable the SVLD function*
6	EnBrownOut	1	POR	R/W	Enable the Brownout function
5	*SvldLevel1*	*0*	*ResSys*		*SVLD level selection (1 out of 8 levels)*
4	*SvldLevel2*	*0*	*ResSys*		
3	*SvldLevel3*	0	*ResSys*		
2	-	-	-	-	
1	-	-	-	-	
0	*SVLDStatus*	*0*		*R*	*Svld result '0' = V_{DD} > SVLD Level* *'1' = V_{DD} < SVLD Level*

7.2 POR and PowerCheck

At power-up the POR initializes the whole circuit and enables the power check function. The POR signal remains active and keeps until the supply voltage is above VPOR. The CPU is held in reset state until power supply reaches the power check level voltage VPWC but at minimum arround 10ms after POR releasing. Power Check is eliminating the grey zone between Vpor and V_{DD}min by releasing system operation not before the minimal specified supply voltage is reached.

The POR cell supervises the regulated voltage observable on VREG terminal. Vreg is also the supply voltage for the whole peripheral logic including the CPU core. The Voltage regulator output impedance together with the external capacitor on VREG terminal form a low pass filter which protect the core logic and the POR cell from noisy power supplies. Pulling VREG below VPOR voltage will also trigger a POR event and putting the circuit in reset state.

Figure 9. POR and Power Check behavior

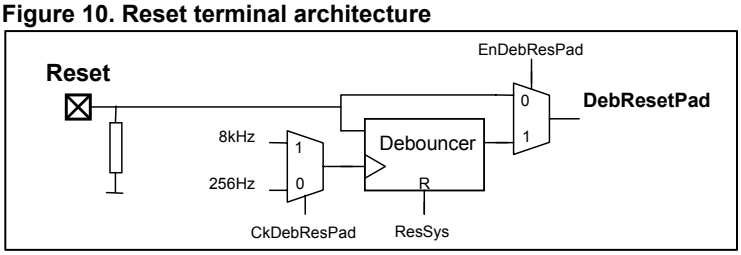


1: Power up phase, power on reset and Power check switched on
2: Noise on V_{DD} filtered on Vreg
3: Large and long drop on V_{DD}, Vreg terminal falls below VPOR voltage. POR and power check initiated.
4: Large but short drop in V_{DD} voltage. Vreg filter prevents to initiate POR.

7.3 Reset Pad

A high level on RESET terminal will trigger a system reset. All registers except the ones connected to POR will be initialized with the reset input. A pull-down resistor is connected on this terminal.
The input may be debounced with either a high or low frequency clock. This reset terminal may be disabled if not desired with bit **DisResInp** located in RegSys1. In the default configuration the reset input is directly routed to the reset controller. The Reset occurance is flagged with bit **ResetPadFlag** in register RegResStat. Write '0' to clear.

Figure 10. Reset terminal architecture

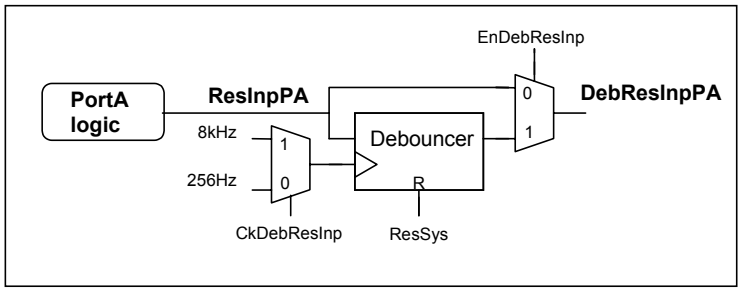


With the debouncer, **EnDebResPad** = '1', the reset input signal must remain high during 1 full debouncer clock cycle to pass and eventually create the ResetPad signal. The high and low frequency debouncing clocks are issued from Prescaler1 (Pr1Ck[13], Pr1Ck[8]). The selection is performed with bit **CkDebResPad** in register in register RegResStat

7.4 PortA Input Reset

Single port A input states or port A combinations can be defined to trigger a system reset. This function can be inhibited with bit **DisResInp** in register RegSys1. Please refer to the chapter PortA for the input reset combination set-up. The Reset occurance is flagged with bit **ResInpPAFlag** in register RegResStat. Write '0' to clear. The ResInpPAFlag will also show in case of BrownOut reset when the DisResInp was set '1'.

Figure 11. PortA input reset



The ResInpPA signal, which is the output of the combination matrix can be used debounced or straight as system reset. High and low debouncing clock frequencies are selectable, both are issued from Prescaler1 (Pr1Ck[13], Pr1Ck[8]).
Debouncer and clock selections are performed with bit **EnDebResInp** and bit **CkDebResInp** in register RegCfgPA.

7.5 BrownOut reset

The BrownOut is a voltage supervisory function. It monitors the main power supply and puts the circuit in reset state when the supply drops below a predefined value of voltage VBO.
BrownOut is enabled at power up **EnBrownOut=**'1' automatically. Afterwards the user can switch off the brown out function with the bit **EnBrownOut='0'** in register RegAnaCfg. During Sleep mode the function is temporarily disabled (most analog cells are switched off during sleep). After the sleep mode the brownout function which was selected before sleep will be reactivated.
The brown out reset occurrence is flagged with bit **ResBwnOutFlag** in register RegResStat. Write '0' to clear this flag. BrownOut reset will also set the ResInpPAFlag bit when DisResInp bit is set '1'.
If the brown out voltage VBO is reached faster than the Brown Out start-up delay time constant, then no reset condition will flag. In case of lower speed start up both ResBwnOutFlag and ResInpPAFlag will show. The user may distinguish between initial 'slow' power-up and 'normal' brown out by using the bit SleepEN. This bit is reset by POR only. In this case if after re-start the SleepEn and ResBwnOutFlag are set, then the circuit is coming from brown out condition. If SleepEn is reset the circuit comes from power-up condition.
While the Brownout function is enabled the circuit will draw additional ~6µA of IV_{DD} current (for Bandgap and Comparator) in all modes except sleep mode.

Figure 12. BrownOut and PowerCheck architecture

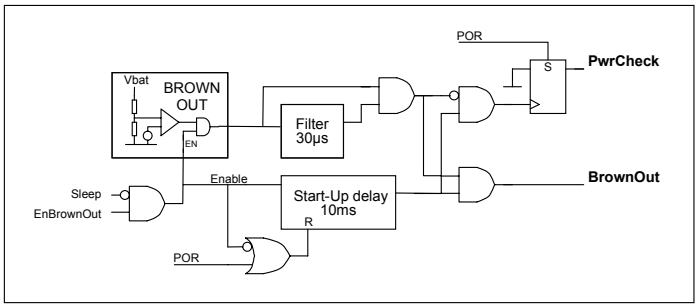
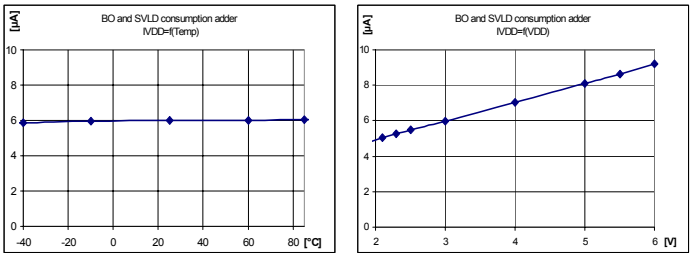


Figure 13. BO and SVLD consumption



7.5.1 BO Timings

7.5.1.1 BrownOut Startup delay ~10ms

The start-up delay allows the integrated Bandgap reference and the Comparator to stabilize after switching on the function. The start-up delay is switched on after power-up (voltage rises above VPOR), setting EnBrownOut='1' or resuming from Sleep mode. The start-up delay is independent of the current VDD voltage.
During the whole start-up delay phase no BrownOut reset will be generated.

7.5.1.2 BrownOut Filter (~30µs)

The BrownOut condition needs to be at least approx. 30µs present to initiate system reset. In case of VBAT undervoltage not reaching the VPOR, then CPU starts to operate approx. 330µs after the VDD voltage is again above VBO (= Re-start after BrownOut).
If the undervoltage reaches and switches on the POR function (< 1.5V), then BrownOut start-up delay as described in 7.5.1.1 applies in addition to meet the powercheck voltage VPWC before the CPU is able to operate again.

7.5.1.3 Re-start after BrownOut (~330µs)

V_{DD} rising again above the VBO voltage will allow restart of CPU operation. However this restart is delayed by approx. 330µs (cold start delay, reset synchronization and BrownOut Filter) Refer also to Figure 12. BrownOut and PowerCheck architecture

Note: Time constants are based on the Prescaler2 clock output that is closest to32kHz. It takes into account the prescaler clock selection, the RC Oscillator frequency range and the RC divider settings. If running on external clock input the time constants may change accordingly to the input frequency and clock management set-ups.

7.6 Watchdog

The digital watchdog is part of the integrated safety functions. Its main task is to supervise the good firmware execution flow. As such it may prevent being stuck in an unwanted endless loop or may allow system recovery in other cases. Its implementation is realised with a low speed counter, which asserts a reset signal on overrun.
Watchdog reset is flagged with bit **ResetWDFlag** in register RegResStat. It may be used to localize the reset source. Write '0' to clear.
The firmware must regularly clear the watchdog otherwise a watchdog reset will occur at timer overrun, this watchdog reset will itself trigger a system reset and forces the circuit to restart.
The watchdog function is always enabled after start-up and after any reset.

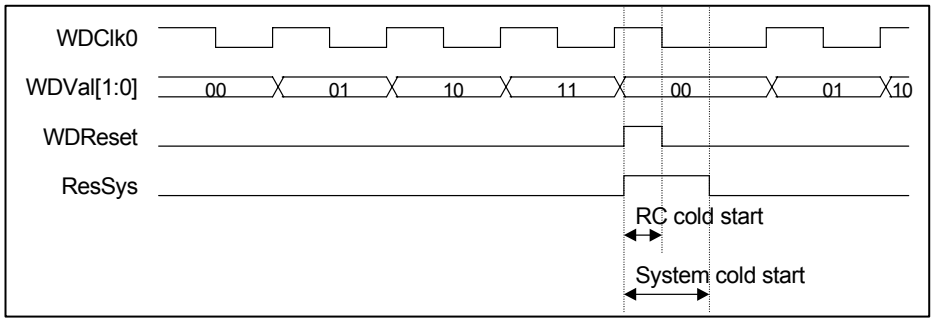
The watchdog is configurable

- 1Hz or 128Hz clock sources (timeouts of ~4s or ~30ms)
- Secure watchdog disabling (locking) with special watchdog
- Reading of the 2 bit watchdog counter value

7.6.1 Watchdog counter

To clear the watchdog counter the firmware must write the bit **WDClear** at '1' in the register RegWDSys. In this way the 2 bits counter is reset and the watchdog restart from 0. WDClear is always read at '0' and writing '0' has no effect. It is recommended to clear the WD counter frequently I.e. every 1s while working with the 1Hz counter clock source.

Figure 14. Watchdog timing diagram



The watchdog counter status is accessible by reading **WDVal**[1:0] in RegWDSys. These two bits are not accessible in write mode. The counter value is '0' if the WD is disabled.
With **WDClkSel** bit it is possible to select from 2 prescaler1 clock sources; Pr1Ck[0] (typ.1Hz) and Pr1ck[7] (typ.128Hz), meaning that the watchdog time out is respectively ~4s and ~30ms.
Changing the prescaler1 input clock selection might also change the watchdog counter frequency.

7.6.2 Lock/Unlock

It is possible to disable the watchdog by in a save way to prevent that system malfunction can itself inhibit the WD. Disabling needs to follow a strict protocol using key lock and key bits which at the end need to be confirmed with writing the **EnWD** at '0'.
The key lock system **WDKeyLock**[2:0] in RegWDSys allows the software writing the key word **WDKey** in the register RegWDKey. It is possible to write **EnWD** at '0' only when the watchdog key word has been loaded.

When the watchdog is disable, **WDKey** and **WDKeyLock** must not change; if one of them is modified or **EnWD** is written at '1' the watchdog is directly enabled.

Note:
EnWD can not be written '0' as long as the valid key lock number and the valid key word are not set.
When WDKeyLock is not the valid number, the WDKey is reset and can not be changed.

How to Unlock (Disable the watchdog)	**How to Lock (Enable the watchdog)**
1st: Write the valid key lock number hex47 in RegWDSys 2nd: Write the valid key word hex96 in register RegWDKey 3rd: Disable the WD by writing hex 07 in register RegWDSys	Locking by clearing the **WDKeyLock**. This action will automatically also clear **WDKey** and **EnWD** bits.

8 Clock management

8.1 Basic features

The EM6812 core and peripherals use several clock sources that can be involved in the same time. There are two main clock domains and both of them are split in two sub-sources:

- High frequency clock sources
  - RC oscillator 1MHz or 10MHz base frequency
  - External clock input from PA[4] or PA[5]
- Low frequency clock source
  - 32kHz Crystal oscillator (typical watch crystal)
  - External clock input on terminal CLKOUT instead of crystal

The main features of the clock management system are:

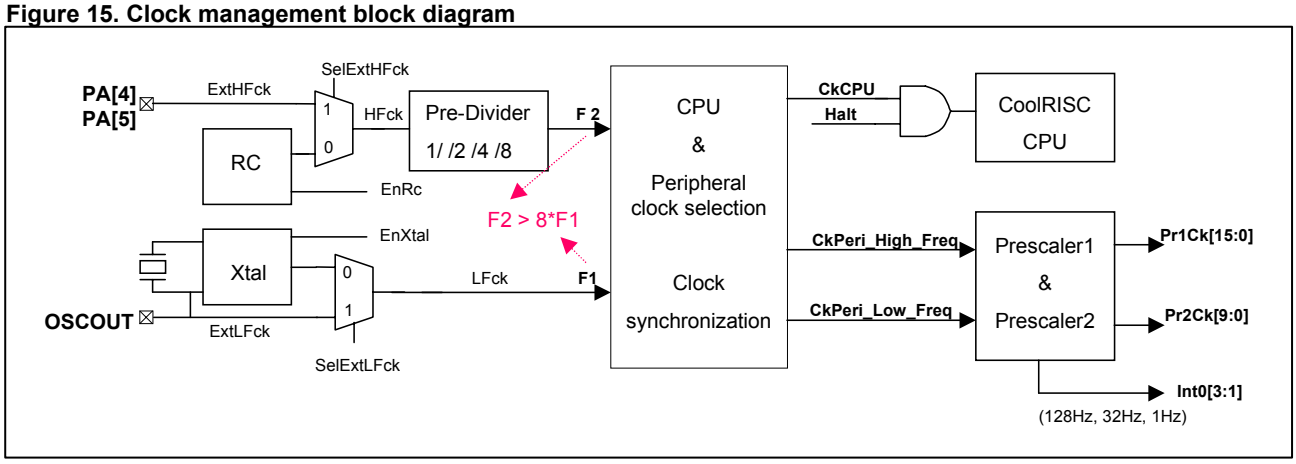
- In high frequency domain
  - Fully internal RC oscillator.
  - No external component needed.
  - Trimmable, continuous RC based frequencies from typically 75kHz up to 14MHz.
  - Pre-Division factors for CPU and Peripheral clocks of 1, 2, 4 or 8 are available.
  - Accurate frequency generation due to software FLL using RC trim and known timing.
  - Automatic clock selection (~32kHz) on Prescaler1 if the Xtal is not available.
  - Power saving switch in case of RC Oscillator is not used (i.e. CPU in Halt).

- In low frequency domain
  - Lowest power watch type Crystal oscillator on 32kHz.
  - RTC signal generation, division on Prescaler1.
  - 3 fix interval interrupts to the CPU.

- In both frequency domains
  - High and low frequency clock domains are fully synchronized for working together.
  - CPU can read registers on the fly thanks to the synchronization between both frequency domains.
  - Completely free of clock glitches, even when switching clocks while running.
  - Fully synchronous core operations.
  - Two clock prescalers (dividers) for the peripheral clock generation
  - Independent clock selection for both prescalers (high or low frequency domain).

8.1.1 Overview

Figure 15. Clock management block diagram



Note:

When both frequency domains are used, the minimum frequency after pre-division (F2) should be at least 8 times higher than the maximum low frequency (F1) to allow for proper system synchronization.

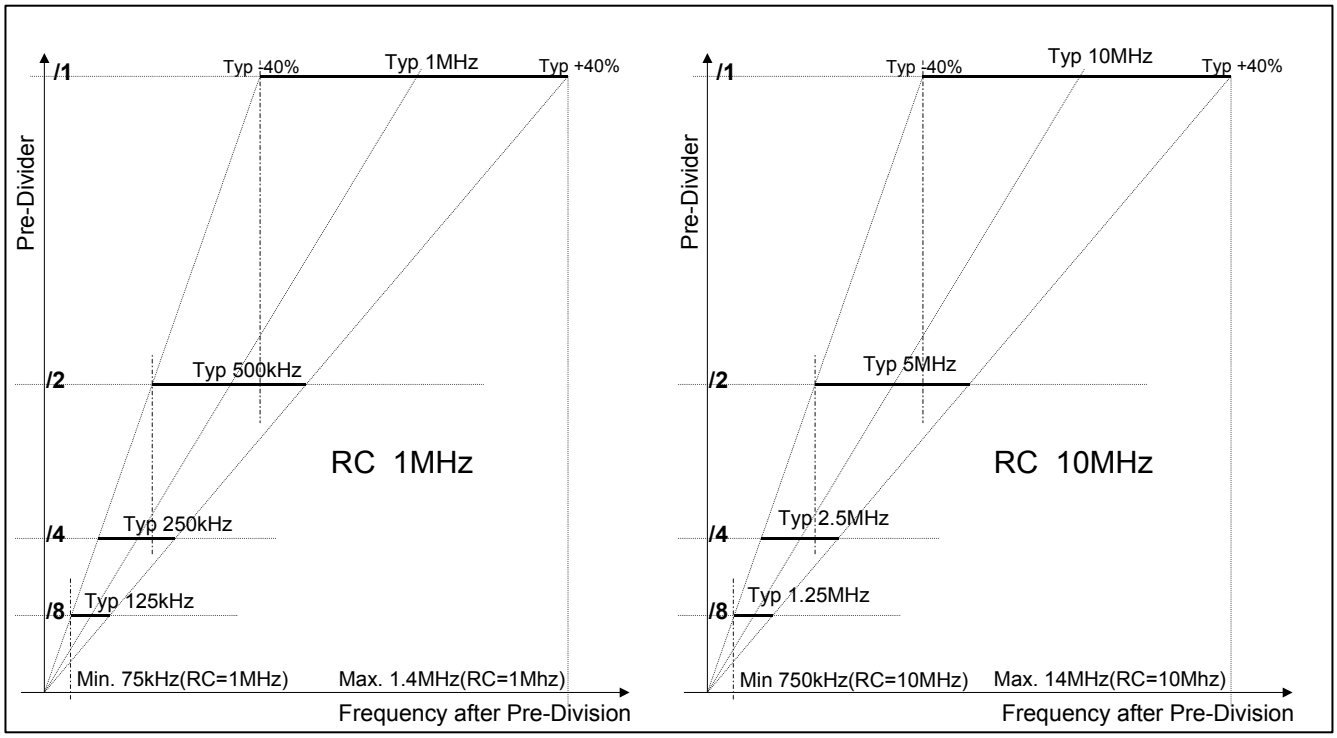
8.2 High frequency clock source

8.2.1 RC oscillator

The RC oscillator is the main clock generator of the high frequency domain. Its function is based on internal elements only. An 8-bit trim register is provided to allow precise frequency adjustment based on known timing functions (software Frequency Locked Loop FLL). The RC Osc is switched on by default on system startup and after any reset on its base frequency of 1MHz. The Pre-Divider is set to /8 which results in a CPU and peripheral clock frequency of 125KHz (typ).

- Frequency range: 1MHz or 10MHz. Selectable with bit **FreqRange** in register RegSys1.
- Frequency division by 1, 2, 4 or 8. Selectable with bits **RCDiv[1:0]** in register RegSys2.
- RC trimming: Range ± ~40%. Coded on 8 bits **Trim[7:0]** in register RegTrimRC.

Figure 16. Overlapping RC Oscillator trimming regions (75kHz to 14MHz)



The user may generate almost any frequency from typ. ~75kHz up to ~14MHz continuously with the 3 combinations mentioned above. The F2 clock mainly depends on the selected frequency range, divider setting and trimm value.

Table 18. Clock selection after Pre-Division (signal F2 in Figure 15)

RC oscillator @ 1 MHz FreqRange = 0 in RegSys1				**RC oscillator @ 10 MHz FreqRange = 1 in RegSys1**			
Divide by 8	Divide by 4	Divide by 2	Divide by 1	Divide by 8	Divide by 4	Divide by 2	Divide by 1
RCDiv[1:0] 11	RCDiv[1:0] 10	RCDiv[1:0] 01	RCDiv[1:0] 00	RCDiv[1:0] 11	RCDiv[1:0] 10	RCDiv[1:0] 01	RCDiv[1:0] 00
RC trimming: minimum frequency Trim[7:0] = 00							
75kHz	150kHz	300kHz	0.6MHz	0.75MHz	1.5MHz	3MHz	6MHz
RC trimming: nominal frequency Trim[7:0] = 7F (default)							
125 kHz	**250kHz**	**500kHz**	**1MHz**	**1.25MHz**	**2.5MHz**	**5MHz**	**10MHz**
RC trimming: maximum frequency Trim[7:0] = FF							
175kHz	350kHz	700kHz	1.4MHz	1.75MHz	3.5MHz	7MHz	14MHz

8.2.1.1 RC switch off

The RC oscillator may be switched off to save power consumption by setting bit **EnRC** = '0'. In this state the system clock must come from either the low frequency clock domain or the external high-speed clock input. While it is sufficient to switch on the RC Oscillator by setting bit **EnRC** = '1', it is not sufficient to clear this bit for switching off the RC. The RC oscillator will only stop if no peripheral circuitry (prescaler or CPU) has the RC clock as its input selection.

A special case occurs during CPU Halt mode. In this state, the RC will be switched off automatically until the CPU returns from Halt, either by IRQ or Reset. This automatic switch off will not take place if one of the prescalers have an RC based clock as an active input clock.

RC switch off procedure (running on XTAL):
1st: Enable the Crystal oscillator, if not yet enabled, by writing **EnXtal** = '1' in register RegSys2.
2nd: Switch the prescalers on the 32kHz Clock; **Pr1CkSel[2:0]=**'000', **AutoSel**='1', **Pr2CkSel**='0' in register RegPrCkSel
3rd: Switch the CPU to 32kHz operation; **Sel32k** ='1' in register RegSys2
4th: Once Crystal is ready, Flag **FlagXtal**='1', disable the RC oscillator with **EnRC**='0' in register RegSys1

The RC oscillator may also be switched off when running on the high speed external clock (SelExtHFck='1' and ENRC='0').

8.2.2 High frequency external clock

It is possible to use an external clock instead of the RC oscillator. There are two pads from the Port A which usable as an input clock. External clock selection is performed with bit **SelExtHFck** = '1' and the clock source is chosen with bit **SelHFckSource**, both in register RegSys2.

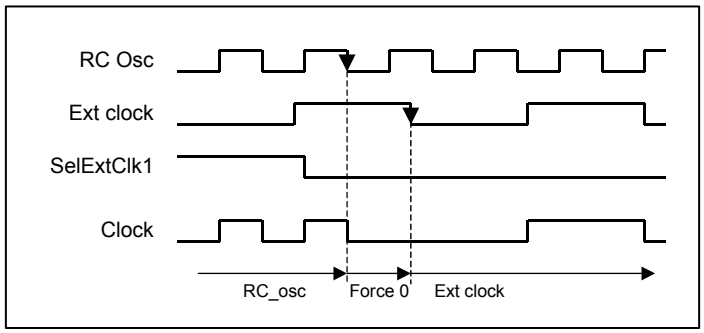
- PA[4] if **SelHFckSource** = '0' in RegSys2
- PA[5] if **SelHFckSource** = '1' in RegSys2

If using one of these pads as clock source, it must be configured as input. Pull resistor selection remains available.
Using the RC loop function, an external RC oscillator can be build using PA[4] as clock input. The clock switching is based on the scheme as shown in Figure 17. Synchronous Clock switching .

8.2.2.1 Switching from RC to external clock

A glitch free clock-switching scheme is implemented. Switch over procedure:
1st: CPU writes clock selection change bit **SelExtHFck**
2nd: on the next falling edge of the current clock the clock signal is forced '0'.
3rd: the newly selected input will become the clock source at its following falling edge.

Figure 17. Synchronous Clock switching



The clock selection output will switchover to the new clock on its next falling clock edge when the initial selected clock has been disabled.
After switchover to external clock, the RC oscillator can be stopped to save current consumption.

8.3 Low frequency clock source

Before switching on a low frequency clock source, make sure that the pre-devised high frequency clock is at least 8 times higher than the expected low frequency clock. Once the low frequency clock is up and running the high frequency clock source may be stopped if not needed anymore. The clock switching is based on the scheme as shown in Figure 17. Synchronous Clock switching.
There are several conditions which all activate the 2 possible low frequency clock sources, Crystal oscillator or external clock.

For Crystal oscillator:
- EnXTAL = '1' : Forces the XTAL on (must never be set if no XTAL present)
- Sel32k = '1' : CPU on low frequency clock source (F1)
- Pr1CkSel[2:0] = '000' : Prescaler 1 running on low frequency clock source (F1)
- Pr2CkSel = '0' : Prescaler 2 running on low frequency clock source (F1)

For external clock:
- EnXtal = '0' AND SelExtLFCk = '1':

To be able to run on external low frequency clock, above condition must be true before any of the crystal oscillator selection conditions is true.

Do not select a low frequency clock source if this source is not present or does not vehicule a clock.

8.3.1 Crystal oscillator

The Xtal oscillator is the main clock generator of the low frequency domain. It is off by default.
Writing **EnXtal** = '1' in the register RegSys2 enables the Xtal oscillator. Now the Xtal management system waits for a defined number of oscillation periods before it allows using this source as a CPU clock. This phase is called Xtal cold-start. It is possible to set this wait time using **OPTCldStart**[1:0] in RegSys1 between typically 1s and ¼ second. The peripheral Crystal derived clocks are not blocked during cold-start (Prescaler inputs). This cold start time is also active after every crystal oscillator re-start.
Whenever a peripheral block or the CPU get a low frequency clock selection – but not external low frequency clock (**SelExtLFCk=** '0') – then the crystal oscillator gets switched on also even if **EnXtal** is '0'. Again every start-up of the crystal oscillator is followed with a cold-start period. To avoid frequent cold start delays, one may permanantly switch on the Xtal with **EnXtal=**'1'.

Table 19. Table of Xtal cold-start duration and selection.

Number of cold-start pulses	Typical wait time	OPTCldStart[1]	OPTCldStart[0]
32768	1s	0	0
24576	3/4s	1	0
16384	1/2s	0	1
8192	1/4s	1	1

During the startup phase, the CPU can check if the Xtal cold start is done or not reading **FlagXtal** in RegSys1. The Xtal is not available as a CPU clock source while **FlagXtal** = '0'. After the cold-start time, this flag becomes '1' and thus allows to switchover.

Note:
Avoid high frequency operation and fast transitions on PA7 while the 32kHz Crystal Oscillator is running. PA7 induced crosstalk on OscOut terminal (PCB, Package) can influence the good Crystal operation.

8.3.2 Low frequency external clock

The low frequency external clock coming from OscOut terminal is used to replace the Xtal oscillator. **SelExtLFck** in RegSys2 controls the selection between Xtal and LF external clock. The same glitch-free clock switching scheme as shown in Figure 17. Synchronous Clock switching is implemented. While running on external low frequency clock, the cold start delay does not apply and the **FlagXtal** is forced '1'. Also the OscOut input must always be driven.

Low frequency external clock selection:

- **SelExtLFck** = '0' and **EnXtal** = '1'. The Xtal oscillator is selected.
- **SelExtLFck** = '1' and **EnXtal** = '0'. The LF external clock is selected.
- **SelExtLFck** = '0' and **EnXtal** = '0'. No active low frequency clock input (default state at startup)

It is not possible to have Xtal and low frequency clock active at the same time since both share the same circuit terminal OscOut. The crystal oscillator must be disabled **EnXtal**='0' to allow for external low frequency clock input. With **EnXtal**='1' the external clock input is blocked.
The low frequency external clock on OscOut terminal may only be selected if the crystal oscillator is not active. Therfore **EnXtal** must be '0', **Sel32k** = '0', **Pr1CkSel**[2:0] not equal to '000' , **Pr2CkSel**='1' prior to setting **SelExtLFCk** = '1'.

8.3.3 Data input on OscOut

The OscOut terminal status can be read when the Crystal Oscillator is not used (**EnXtal**='0'). The reading is performed with a read access to bit **DatOscOut** in register RegResStat. The OscOut input has no internal pull resistor. It may be left floating while not used.

8.4 Clock synchronization

Besides the already described clock synchronization schemes between internal and external clocks in their respective frequency domain, the EM6812 re-synchronizes internally the asynchronous F1 and F2 clocks (see 'Figure 15. Clock management block diagram') so the CPU and the periphery always get stable clock edge conditions. The implementation is done by synchronization of the low frequency clock with the higher speed one. For proper operation the rule F2 > 8* F1 applies.

An active peripheral clock edge issued from F1 or F2 will never occur during a CPU read or write cycle and thus allows the CPU to manage its peripherals while they are in a quiet state. Note that this does not apply for peripherals, which run on an asynchronous clock that has not been re-synchronized (i.e. undebounced timer clock sources). Maximum peripheral clock selection is half the high frequency pre-divided clock.

8.5 CPU clock selection

The CPU can run on the high frequency F2 or low frequency F1 clock domain. CPU clock selection and peripheral clock selection are independent. If the Xtal cold-start is not finished and the LF external clock is not selected, the CPU cannot switch on the LF clock domain, and it continuous to run on the HF clock domain until the end of the cold-start time. The bit **Sel32k** in RegSys2 controls the CPU clock selection:

- **Sel32k** = '0'. The CPU runs on HF clock domain.
- **Sel32k** = '1'. The CPU runs on LF clock domain. In this case one of the prescalers must also run with the LF clock (either Pr1CkSel='000' or Pr2CkSel='0')

Internal or external clock sources may be chosen within both the high and low frequency clock domains. The clock switching is based on the scheme as shown in Figure 17. Synchronous Clock switching.

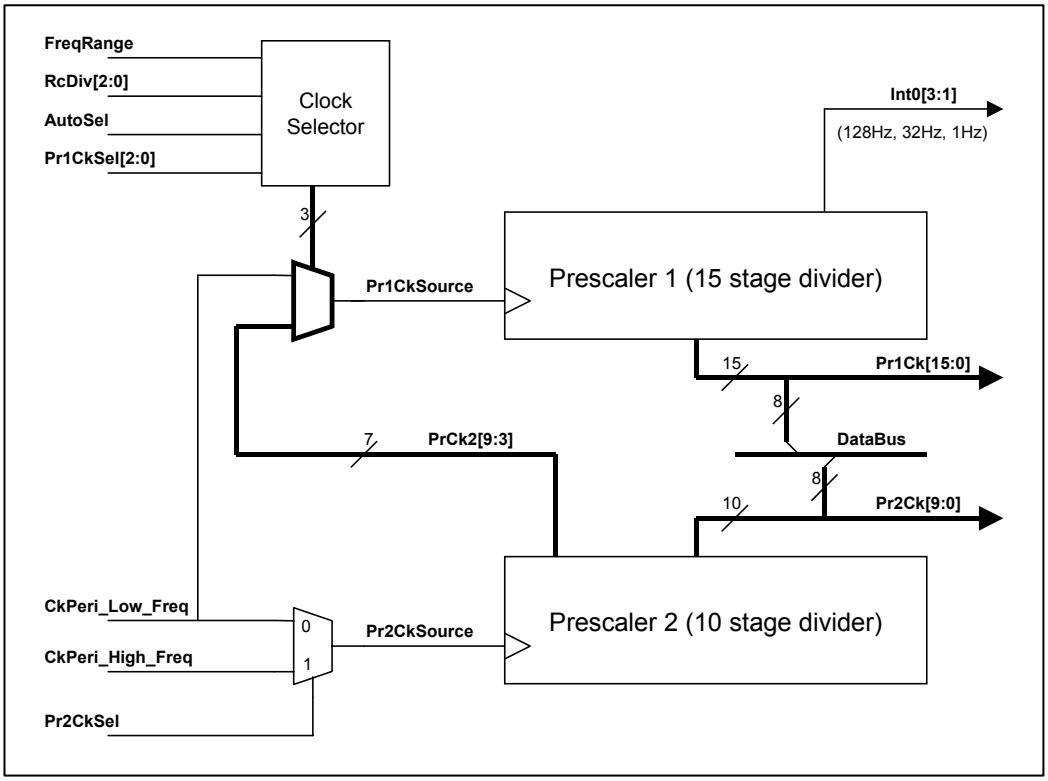
8.6 Peripheral clocks generation

There are two prescalers dedicated for the peripheral clocks and the input clock can be either issued from the high or low frequency domain. Their default setup is:

- Prescaler2: 10-stages of RC clock division; pre-division by 8 results in a typ 62.5kHz prescaler input clock.
- Prescaler1: 15-stage divider, 'auto selected' close to 32kHz input clock coming from prescaler2.

Each prescalers 8 most significant bit can be read and reset.

Figure 18. Prescaler clock selection architecture



8.6.1 Prescaler2 (10 stages)

The prescaler 2 has two selectable clock sources possible. By default it is running on RC oscillator.

- **Pr2CkSe**l = '0': LF clock domain (F1: Xtal oscillator or external clock).
- **Pr2CkSel** = '1': HF clock domain after pre-division (F2) divided by 2 (RC oscillator or external clock) F2/2.

Pr2CkSource is the clock source of the prescaler2. 10 different clocks get out from the prescaler2; Pr2Ck[9:0].
The shaded values are selected for prescaler1 clock input in case of autoselect and no active low frequency.
All clock duty cyles are 50% expecpt for the Pr2CkSource which is 25% high if issued from the high frequency input clock.

Table 20. Prescaler2 output frequencies when running on HF clock domain (CkPeri_High_Freq)

Prescaler2 output	RC Oscillator	10MHz	10MHz	10MHz	10MHz	1MHz	1MHz	1MHz	1MHz
	Pre-Division by	1	2	4	8	1	2	4	8
	Pr2CkSource (Duty: 25% high)	F2/2 5MHz	F2/2 2.5MHz	F2/2 1.25MHz	F2/2 625kHz	F2/2 500kHz	F2/2 250kHz	F2/2 125kHz	F2/2 62.5kHz
Pr2Ck[9]	Pr2CkSource / 2	2.5MHz	1.25MHz	625kHz	313kHz	250kHz	125kHz	62.5kHz	31.3kHz
Pr2Ck[8]	Pr2CkSource / 4	1.25MHz	625kHz	313kHz	156kHz	125kHz	62.5kHz	31.3kHz	15.6kHz
Pr2Ck[7]	Pr2CkSource / 8	625kHz	313kHz	156kHz	78kHz	62.5kHz	31.3kHz	15.6kHz	7.8kHz
Pr2Ck[6]	Pr2CkSource / 16	313kHz	156kHz	78kHz	39kHz	31.3kHz	15.6kHz	7.8kHz	3.9kHz
Pr2Ck[5]	Pr2CkSource / 32	156kHz	78kHz	39kHz	20kHz	15.6kHz	7.8kHz	3.9kHz	2.0kHz
Pr2Ck[4]	Pr2CkSource / 64	78kHz	39kHz	20kHz	10kHz	7.8kHz	3.9kHz	2.0kHz	1.0kHz
Pr2Ck[3]	Pr2CkSource / 128	39kHz	20kHz	10kHz	5kHz	3.9kHz	2.0kHz	1.0kHz	500Hz
Pr2Ck[2]	Pr2CkSource / 256	20kHz	10kHz	5kHz	2.4kHz	2.0kHz	1.0kHz	500Hz	250Hz
Pr2Ck[1]	Pr2CkSource / 512	10kHz	5kHz	2.4kHz	1.2kHz	1.0kHz	500Hz	250Hz	125Hz
Pr2Ck[0]	Pr2CkSource / 1024	5kHz	2.4kHz	1.2kHz	610Hz	500Hz	250Hz	125Hz	62Hz

The prescaler2 clock values, 8 MSB's, **Pr2CkStatus**[7:0] can be read in register RegPr2Status. These 8 most significative bits can be cleared by a simple write operation of any value to the RegPr2Status register.
The Pr2Ck sources are used as input clock sources for several other peripheries (SPI, Timers, Prescaler1, etc). Clearing the 8 MSB's may therefore influence the proper operation of these peripheries.

Table 21. Prescaler2 output frequencies when running on LF clock domain (CkPeri_Low_Freq)

Signal name	Division	Presacler 2 output frequency In case of XTAL 32kHz as active low frequency clock
Pr2CkSource	Pr2CkSource / 1	32768Hz
Pr2Ck[9]	Pr2CkSource / 2	16384Hz
Pr2Ck[8]	Pr2CkSource / 4	8192Hz
Pr2Ck[7]	Pr2CkSource / 8	4096Hz
Pr2Ck[6]	Pr2CkSource / 16	2048Hz
Pr2Ck[5]	Pr2CkSource / 32	1024Hz
Pr2Ck[4]	Pr2CkSource / 64	512Hz
Pr2Ck[3]	Pr2CkSource / 128	256Hz
Pr2Ck[2]	Pr2CkSource / 256	128Hz
Pr2Ck[1]	Pr2CkSource / 512	64Hz
Pr2Ck[0]	Pr2CkSource / 1024	32Hz

8.6.2 Prescaler1 (15 stages)

- 8 clock input selections from Xtal or Prescaler2 clocks.
- RTC real time clock function in case of 32kHz Xtal input.
- 'Autoselect' close to 32kHz input clock selection from prescaler2 in case of no Xtal. The autoselection is based on the RC oscillator settings (FreqRange and RCDiv) and switched on with bit **AutoSel** in register RegPrCkSel. In case of an active low frequency clock selection (Xtal or external low frequency enabled) the autoselect will select the low frequency input F1 as the prescaler clock source. The clock switching is based on the scheme as shown in Figure 17. Synchronous Clock switching.

Table 22. Table of the Prescaler1 automatic clock selection from prescaler2. (AutoSel = '1', Pr2CkSel ='1')

FreqRange	**RC = 10 MHz**				**RC = 1MHz**			
RCDiv	1	2	4	8	1	2	4	8
Pr2CkSource	5MHz	2.5MHz	1.25MHz	625kHz	500kHz	250kHz	125kHz	62.5kHz
Pr2Ck[9:0] selected bit	[3]	[4]	[5]	[6]	[6]	[7]	[8]	[9]
Pr1CkSource, AutoSel	39.1kHz	39.1kHz	39.1kHz	39.1kHz	31.3kHz	31.3kHz	31.3kHz	31.3 kHz

The low frequency clock gets automatic selected as Pr2CkSource in case of Autosel = '1' and Pr2CkSel = '0'.

The prescaler1 clock values, 8 MSB's, **Pr1CkStatus**[7:0] can be read in register RegPr1Status. These 8 most significative bits can be cleared by a simple write operation of any value to the RegPr1Status register.
The Pr1Ck sources are used as input clock sources for several other peripheries (SPI, Timer, etc). Clearing the 8 MSB's may therefore influence the proper operation of these peripheries.

It is also possible to select a specific prescaler1 input clock source by setting **AutoSel**='0'. The selection is done with bits **Pr1CkSel**[2:0] in register RegPrescCkSel.

Table 23. Prescaler1 clock selection, non-automatic mode. (AutoSel = '0')

Pr1CkSel[2:0]	**Clock selected in case of Pr2CkSel='1' (high freq.) on RC Oscillator**	**Clock selected in case of Pr2CkSel='0' (low freq.) with 32kHz Crystal**
000	Low frequency clock domain F1 (Xtal or ExtLFck)	Low frequency clock domain F1 (Xtal or ExtLFck)
001 (default)	Prc2Ck[3], refer to Table 20	Prc2Ck[3] = 16384Hz
010	Prc2Ck[4], refer to Table 20	Prc2Ck[4] = 8192Hz
011	Prc2Ck[5], refer to Table 20	Prc2Ck[5] = 4096Hz
100	Prc2Ck[6], refer to Table 20	Prc2Ck[6] = 2048Hz
101	Prc2Ck[7], refer to Table 20	Prc2Ck[7] = 1024Hz
110	Prc2Ck[8], refer to Table 20	Prc2Ck[8] = 512Hz
111	Prc2Ck[9], refer to Table 20	Prc2Ck[9] = 256Hz

Table 24. Prescaler1 output clock frequencies based on selected input clock source.

Prescaler1 output		XTAL	Pr2Source AutoSelect	RC 10MHz Pre-Div. 1	RC Clock Pre-Division 1 to 8	RC 1 MHz Pre-Div. 8	Xtal 32kHz
	Pr1CkSource origin	F1, Xtal	F2 based	F2 (10MHz) Highest value	F2:12 intermediate values from	F2 (125kHz) Lowest value	Xtal from Pr2
	Pr1CkSource frequency	F1 32kHz	Pr2cK auto ~32kHz	Pr2ck[9] 2.5MHz	Pr2ck[9:3] 1.25Mhz and 500Hz	Pr2Ck[3] 500Hz	Pr2Ck[3] 256Hz
Pr1Ck[14]	/ 2	16kHz	~16kHz	1.25MHz	625KHz to 500Hz	250Hz	128Hz
Pr1Ck[13]	/ 4	8kHz	~8kHz	625kHz	313kHz to 250Hz	125Hz	64Hz
Pr1Ck[12]	/ 8	4kHz	~4kHz	313kHz	156kHz to 125Hz	62Hz	32Hz
Pr1Ck[11]	/ 16	2kHz	~2kHz	156kHz	78kHz to 62Hz	31Hz	16Hz
Pr1Ck[10]	/ 32	1kHz	~1kHz	78kHz	39kHz to 31Hz	16Hz	8Hz
Pr1Ck[9]	/ 64	512Hz	~512Hz	39kHz	20kHz to 16Hz	8Hz	4Hz
Pr1Ck[8]	/ 128	256Hz	~256Hz	20kHz	10kHz to 8Hz	4Hz	2Hz
Pr1Ck[7]	/ 256	128Hz	~128Hz	10kHz	5kHz to 4Hz	2Hz	1Hz
Pr1Ck[6]	/ 512	64Hz	~64Hz	5kHz	2.4kHz to 2Hz	1Hz	0.5Hz (2s)
Pr1Ck[5]	/ 1024	32Hz	~32Hz	2.4kHz	1.2kHz to 1Hz	0.5Hz (2s)	0.25Hz (4s)
Pr1Ck[4]	/ 2048	16Hz	~16Hz	1.2kHz	600Hz to 0.5Hz	0.25Hz (4s)	0.125Hz (8s)
Pr1Ck[3]	/ 4096	8Hz	~8Hz	600Hz	300Hz to 0.25Hz	0.125Hz (8s)	0.062Hz (16s)
Pr1Ck[2]	/ 8192	4Hz	~4Hz	300Hz	150Hz to 0.125Hz	0.062Hz (16s)	0.031Hz (32s)
Pr1Ck[1]	/ 16384	2Hz	~2Hz	150Hz	75Hz to 0.062Hz	0.031Hz (32s)	0.016Hz (64s)
Pr1Ck[0]	/ 32768	1Hz	~1Hz	75Hz	38Hz to 0.031Hz	0.016Hz (64s)	0.008Hz (128s)

8.7 RC clock trimming with Xtal oscillator

The RC oscillator can be trimmed to a precise frequency using either internal 32kHz Xtal based frequencies or any other known timing as a reference. The base frequencies of the RC oscillator are trimmable to typ ± 40%, combining the trimming and the frequency divider gives a total of 8 overlapping frequency regions with each 256 possible frequencies. See also Figure 16. The frequency precision within the 75kHz to 10MHz range is better than 0.5%. Repeating the frequency adjustment regularly allows compensating for slow voltage and temperature changes.
The trim value can be obtained by successive approximation using the timer to count the RC clock during a given timing period (i.e. prescaler interrupts), then change the trim value based on the timer result until the result is within the desired precision window. See also the Application note for RC trimming.

8.8 Registers overview

Table 25. Clock management registers

RegSys1		0x10			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	Sleep	0	ResSys	R/W	Put the circuit in sleep mode if = '1'
6	DisResetPad	0	POR	R/W	Disable the input pad reset if = '1'
5	DisResInp	0	POR	R/W	Disable the port A reset input if = '1'
4	FlagXtal	0	ResMain	R	Xtal cold start flag, Xtal ready if = '1'
3	OPTCldStart[1]	0	POR	R/W	Xtal cold start duration: '00' = 1s, '10' = 3/4s, '01'=1/2s, and '11'=1/4s.
2	OPTCldStart[0]	0	POR	R/W	
1	FreqRange	0	ResSys	R/W	RC osc. frequency range selection: '1'=10MHz '0'=1MHz
0	EnRC	1	ResSys	R/W	Enable RC oscillator if = '1'

RegSys2		0x11			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	EnXtal	0	ResMain	R/W	Enable Xtal oscillator if = '1'
6	SelExtHFck	0	ResMain	R/W	Enable external clock instead of RC if = '1'
5	SelHFckSource	0	ResMain	R/W	Select external clock PA4 if '0', PA5 if '1'
4	SelExtLFck	0	ResMain	R/W	Enable external clock instead of Xtal if = '1'
3	--	0	--	R/W	Not used, read always '0'
2	Sel32k	0	ResMain	R/W	CPU clock '1'=low freq (F1) '0'=high freq (F2)
1	RCDiv[1]	1	ResMain	R/W	HF domain division factor for F2: '00'=1, '01'=2 , '10'=4, '11'=8
0	RCDiv[0]	1	ResMain	R/W	

RegPr1Status		0x15			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Pr1CkStatus[7:0]	01	ResSys	R/C*	Prescaler1 Clock status on 8 MSB

RegPr2Status		0x16			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Pr2CkStatus[7:0]	00	ResSys	R/C*	Prescaler2 Clock status on 8 MSB

C*: Write access resets the register value (8 MSB counter values)

RegTrimRC		0x13			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Trim[7:0]	7F	ResMain	R/W	RC oscillator trimming byte

RegPrCkSel		0x14			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	Pr1CkSel[2]	0	ResSys	R/W	Refer to the ' Table 23. Prescaler1 clock selection, non-automatic mode '
6	Pr1CkSel[1]	0	ResSys	R/W	
5	Pr1CkSel[0]	1	ResSys	R/W	
4	AutoSel	1	ResSys	R/W	Auto prescaler1 clock selection to ~32kHz
3	Pr2CkSel	1	ResSys	R/W	Prescaler2 clock selection '0'=low freq (F1), 1=high freq / 2 (F2 / 2).
2	--	1	--	R	Read always '1'
1	--	1	--	R	Read always '1'
0	--	0	--	R	Read always '0'

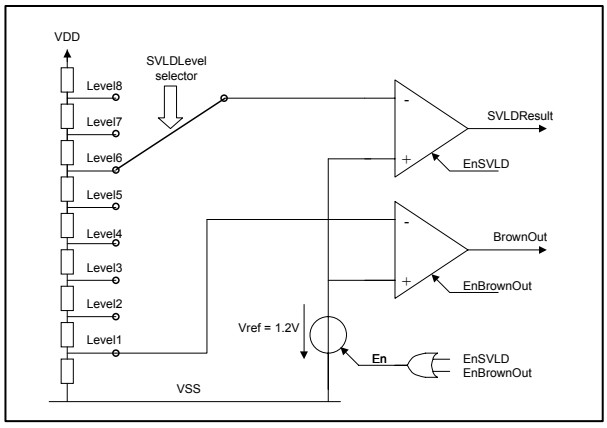
Table 26. Clock interrupts mapping

Interrupt source	Priority	IntCtrl connection
Pr1Ck[0] (1Hz if Xtal)	0	Int0[1]
Pr1Ck[5] (32Hz if Xtal	0	Int0[2]
Pr1Ck[7] (128Hz if Xtal)	0	Int0[3]

9 Supply Voltage Level Detector (SVLD)

The EM6812 has a built in 8 level supply voltage detector that compares the supply voltage against a predefined voltage level. The CPU can inspect the result of the comparison, it reads '0' if the supply voltage is higher than the compare level and '1' if lower.
The lowest compare level is equivalent to the brown out detection level. Obviously this level can only be measured if the brown out function is switched off. Also the SVLD function is temporarily disabled during Sleep mode.
The internal bandgap reference is shared between the SVLD and the BrownOut function. If active, it will consume an extra ~6µA during the whole measuring time

Figure 19. SVLD architecture



Timings:
- Internal voltage reference settling time: **8ms**
(From either EnSVLD='1' or EnBrownOut='1').
Must be respected when the voltage reference gets switched on.

- Comparator settling time:
(From EnSVLD='1' to 1st readout)
Must be respected for every measure before reading the result.

Table 27. Analogue configurations register

RegAnaCfg		**0x20**			**Analogue configurations**
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	EnSVLD	0	ResSys	R/W	Enable the SVLD function
6	EnBrownOut	1	POR	R/W	Enable the Brownout function
5	SVLDLevel1	0	ResSys		SVLD level selection (1 out of 8 levels)
4	SVLDLevel2	0	ResSys		
3	SVLDLevel3	0	ResSys		
2	-	-	-	-	
1	-	-	-	-	
0	SVLDStatus	0			SVLD result '0' = V_{DD} > SVLD Level '1' = V_{DD} < SVLD Level

Table 28. SVLD selection table

SVLDLevel3	**SVLDLevel2**	**SVLDLevel1**	**Typical Detection Level**
0	0	0	2.0
0	0	1	2.1
0	1	0	2.2
0	1	1	2.3
1	0	0	2.4
1	0	1	2.5
1	1	0	2.6
1	1	1	3.4

10 Port A

10.1 Basic features

The port A is an 8-bit general-purpose input/output port. The CPU can read the input state in all modes. All selections concerning the port A are bit-wise executable:

Bit-wise executable on PA[0] to PA[7]:
- Input / Output selection
- CMOS or NCH Open Drain Outputs
- Interrupt with rising or falling edge selection, direct or debounced.
- Pull resistor selection. Pull-up or Pull-down. When both are selected, pull-up has the priority.

Special features
- Input reset and wakeup capabilities on input pattern or single pin
- External system clock input on PA[4] or PA[5]
- RC Oscillation Loop on PA[6] , PA[4]
- Timer clock and start stop inputs.
- Dual Port Ram Control signals on PA[0] to PA[3]

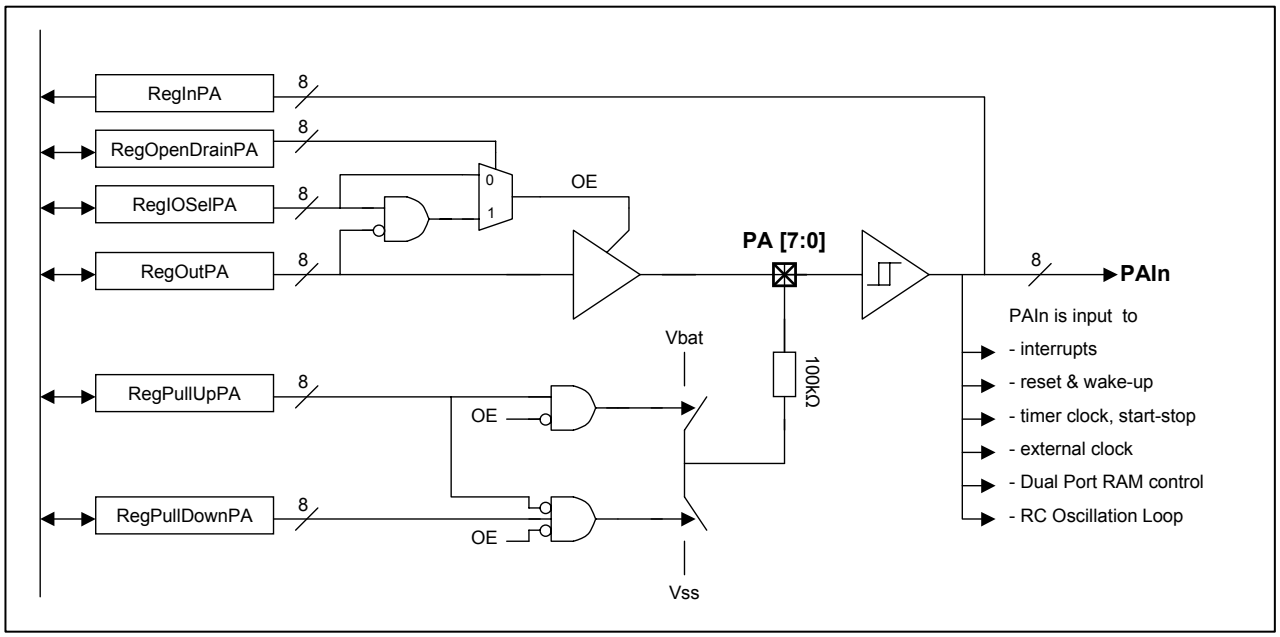
Table 29. Port A External Connectivity

Port A input connectivity									**PA output**
Always active	Enable with bits MskIRQPA[7:0]	Enable with bit SelExtClk1	Enable with bit RCLoop	Enable with bit EnDualRAM	Enable with corresponding Timer configuration bits				
PA input	External Interrupt	External system clock	Oscillation Loop	Dual Port RAM	Timer1	Timer2	Timer3	Timer4	Output drive
PA[7]	IRQPA[7]				Start[7]	Start[7]	Start[7]	Start[7] Clk[1]	Drive 2
PA[6]	IRQPA[6]		RCOut		Start[6]	Start[6]	Start[6] Clk[1]	Start[6]	Drive 2
PA[5]	IRQPA[5]	ExtClk1			Start[5]	Start[5] Clk[1]	Start[5]	Start[5]	Drive 1
PA[4]	IRQPA[4]	ExtClk1	RCIn		Start[4] Clk[1]	Start[4]	Start[4]	Start[4]	Drive 1
PA[3]	IRQPA[3]			ExtAdr[1]	Start[3]	Start[3]	Start[3]	Clk[0]	Drive 1
PA[2]	IRQPA[2]			ExtAdr[0]	Start[2]	Start[2]	Clk[0]	Start[3]	Drive 1
PA[1]	IRQPA[1]			ExtWEn	Start[1]	Clk[0]	Start[2]	Start[2]	Drive 1
PA[0]	IRQPA[0]			ExtCEn	Clk[0]	Start[1]	Start[1]	Start[1]	Drive 1

These input connections remain active also if the corresponding terminal is configured as output.

10.1.1 Overview

Figure 20. Port A IO and pull selection



10.1.2 Register map, PA IO functions

Table 30. Port A Registers overview

Functions	Register name	Basic function
Base	RegInPA	Direct read of input terminal state
	RegOutPA	Data output register
	RegIOSelPA	Direction selection
Pull resistor and open drain	RegPullUpPA	Pull-up resistor selection
	RegPullDownPA	Pull-down resistor selection
	RegOpenDrainPA	Enable n-channel open drain output
IRQ related	RegIntEdgPA	Interrupt edge selection
	RegEnDebPA	Debouncer selection for interrupt signal
Reset and Wake-up	RegCfgPA	Reset and wake up system configuration, RC Oscillation Loop selection
Reset and Wake-up	RegMskRstWkUp	Combination mask selection
	RegCmbKey	Reset or wake-up key

Table 31. Port A Registers

RegInPA		**0x21**			**Input register**
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	PAIn[7:0]	--	-	R	Direct read of input terminal state '0'=read low, '1'= read high

RegOutPA		**0x22**			**Output data**
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	OutPA[7:0]	00	ResSys	R/W	Data output register '0'= output low, '1'=output high

RegIOSelPA		**0x24**			**Direction setting**
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	IOSelPA[7:0]	00	ResSys	R/W	Direction selection; '1'=Output, '0'=Input

RegPullUpPA		**0x2A**			**Pull-up selection**
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	PullUpPA[7:0]	00	ResMain	R/W	Pull-up resistor selection '0'=no pull-up, '1'=pull-up enabled

RegPullDownPA		**0x2B**			**Pull-down selection**
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	PullDownPA[7:0]	FF	ResMain	R/W	Pull-down resistor selection '0'=no pull-down, '1'=pull-down enabled (if no pull-up)

RegOpenDrainPA		**0x29**			**N-channel Open drain selection**
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	OpenDrainPA[7:0]	00	ResMain	R/W	N-channel open drain selection (if output) '0'=CMOS mode, '1'= open drain enabled

10.1.3 IO Operation

Port A input terminal status can always be read directly. All registers influencing the IO modes are bit-wise selectable. The integrated switchable pull resistors and the selectable output drive mode allow a maximum of different terminal modes. Refer to Table 32. Port A IO mode for the details.

The default state after power up on all PA terminals is input mode with pull-down resistor.

Table 32. Port A IO modes

Modes	IOSelPA[i]*	OutPA[i]	OpenDrainPA[i]	PullUpPA[i]	PullDownPA[i]	PA[i] terminal	Notes
Input mode	0	X	X	0	0	High-Z	Needs external drive (PA[i] must never be floating)
Input mode with pull-up	0	X	X	1	X	Weak Hi	Pull-up has priority over pull-down
Input mode with pull-down	0	X	X	0	1	Weak Lo	Default state after Power-up
Output mode, CMOS high drive	1	1	0	X	X	1	Pull resistors disabled
Output mode, CMOS low drive	1	0	0	X	X	0	Pull resistors disabled
Output mode, open drain, high-Z	1	1	1	0	X	High-Z	Pull-down disabled Needs external drive (PA[i] must never be floating)
Output mode, open drain with pull-up	1	1	1	1	X	Weak Hi	Pull-up active
Output mode, open drain drive low	1	0	1	X	X	0	Pull-up disabled

Note:
Every port A input always needs at least one driver. A floating input can generate hazards and may induce cross current in the input amplifier.

Note:
Avoid high frequency operation and fast transitions on PA7 while the 32kHz Crystal Oscillator is running. PA7 induced crosstalk on OscOut terminal (PCB, Package) can influence the good Crystal operation.

10.2 Port A Interrupt requests

Each port A input is an interrupt request source active on rising or falling edge corresponding to the individual **IntEdgPA** bit setting. The interrupt source can be debounced or direct, bit-wise selection with bit **EnDebPA**.

Figure 21 Schematic view of Debouncer and Edge selection

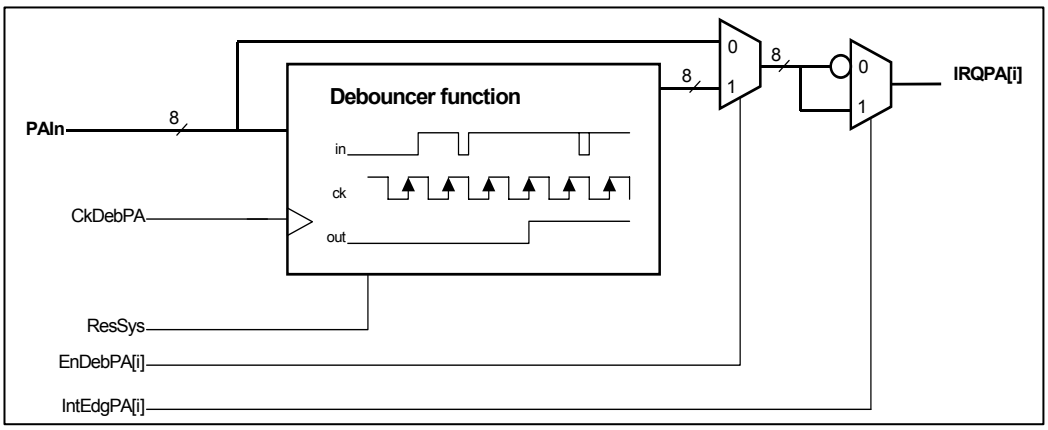


Table 33. Port A Registers for Debouncer and Interrupts

RegIntEdgPA		0x27			Debouncer edge selection
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	IntEdgPA[7:0]	00	ResMain	R/W	PA interrupt edge selection; 0=falling, 1=rising

RegEnDebPA		0x28			Debouncer selection
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	EnDebPA[7:0]	00	ResSysSlp	R/W	PA interrupt debouncer selection 0=direct, 1=debounced Pr1Ck[8] (256Hz).

Table 34. Port A Interrupt mapping

Interrupt source	Priority	IntCtrl connection
IRQPA[7:0]	1	Int1[7:0]

10.2.1 Debouncer

If the debouncer is selected the corresponding input signal must remain stable during 1 full debouncer clock cycle to pass and eventually create an interrupt request.

It is recommended to use the debouncer functionality on the port A interrupt inputs. The debouncer frequency CkDebPA is coming from the prescaler1, Pr1Ck[8] (256Hz).

Minimum input pulse length to pass the debouncer:

- Pulse length smaller 1 debouncer clock period : Pulse does not pass (filtered out)
- Pulse length in-between 1 and 2 debouncer clock periods : Pulse may pass
- Pulse length grater than 2 debouncer clock periods: Pulse always passes

Note:
The debouncer output is reset low with signal ResSys. On the second debouncer clock edge after reset an IRQ may be generated. Use the interrupt mask to overcome this.

Note:
Changing **RegIntEdgPA** may result in a transition interpreted as a valid IRQ. Avoid it by masking the IRQ while changing the edge selection.

10.3 Reset and Wake-up

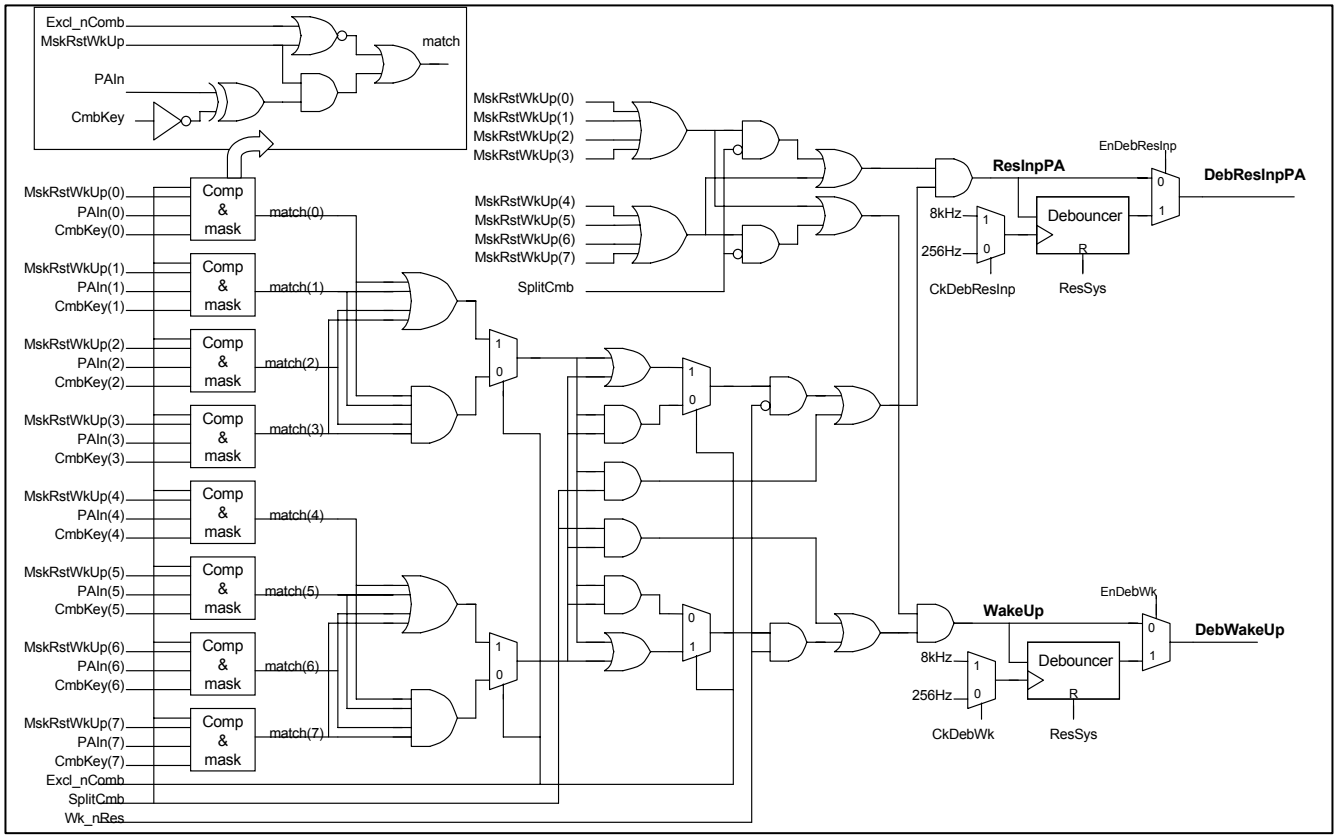
Port A has an input comparison register to detect an incoming reset and/or wakeup condition. On a compare match with the defined key pattern, the selected functionality will be activated. When working on all PA inputs, only reset or wake-up function is active. With the PA input split in 2 time's 4-pin input, the reset and wake-up functions can be used simultaneously. In this case, the upper 4 bits are attached to the reset and the lower to the wake-up. The 'don't care' condition allows masking specific inputs so their state is not taken into account.

The reset and the wake-up signal can be individually debounced. The input reset function can be disabled with bit **DisResInp** in register RegSys1 (Reset section).

Table 35. Port A Reset and wake-up selection

Wk_nRes	SplitCmb	Description
X	1	PAIn[7:4] used for reset function. PAIn[3:0] used for the wake up function.
0	0	PAIn[7:0] is used as reset function. (default)
1	0	PAIn[7:0] is used as wake-up function.

Figure 22; Reset and Wake-up diagram



10.3.1 Register map

Table 36. Port A Reset and wake-up registers

RegCfgPA		0x23		Configuration settings for reset and wake-up system.	
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	Excl_nComb	0	ResMain	R/W	Mode selection for reset and wake-up system '0'=Input combination, '1'=Individual bit action
6	Wk_nRes	0	ResMain	R/W	Selects either reset or wake-up function '0'=Reset, '1'=Wake-up (has no action if SplitCmb=0)
5	SplitCmb	0	ResMain	R/W	PA[7:4] = Reset combination PA[3:0] = Wake-up combination '0'=full port A, '1'=port A split,
4	EnDebResInp	0	ResSysSlp	R/W	Enable the reset debouncer function '0'=No debouncer, '1'= debouncer enabled
3	CkDebResInp	0	ResMain	R/W	Select the reset debouncer clock '0'=Pr1Ck[8] (256Hz); '1'=Pr1Ck[13] (8kHz)
2	EnDebWk	0	ResSysSlp	R/W	Enable the wake-up debouncer function '0'=No debouncer, '1'= debouncer enabled
1	CkDebWk	0	ResMain	R/W	Select the wake-up debouncer clock '0'=Pr1Ck[8] (256Hz); '1'=Pr1Ck[13] (8kHz)
0	RCLoop	0	ResSys	R/W	Enable the RC Osc Loop on PA[6], PA[4] '0'=RC Loop disabled, '1'= RC Loop enabled

RegMskRstWkUp		0x26		Input mask for reset and wake-up selection	
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	MskRstWkUp[7:0]	00	ResMain	R/W	Input selection for reset and wake-up system 1=input selected, 0= do not care

RegCmbKey		0x25		Reset and Wake-up key definition	
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	CmbKey[7:0]	00	ResMain	R/W	Reset and wake-up key 0=match if 0, 1=match if 1

10.3.2 Input splitting

Both reset and wake-up functions can be activated if the port A is split in 2 parts. PA[7:4] gets the reset function and PA[3:0] the wake-up. With the bit SplitCmb='0' (no split of port A) only the function defined by bit Wk_nRes is active.

10.3.3 Actions

Resetting the microcontroller on condition match.
Wake-up signal to the microcontroller on condition match.

Reset will trigger a system reset (ResSys)

Wake-up will resume from unclocked CPU low power modes. Its main actions are:

- In Halt mode: Sending Event to the CPU (resumes from HALT mode back to active mode). Refer to Interrupt section. Wake-up will activate signal DebWakeUP and give the CPUEvent0.
- In Sleep mode: Cancel Sleep mode (resets the Sleep bit, CPU restarts from adr 0).

10.3.4 Condition match

A match condition is obtained if either all selected bits match (AND-function for Combination trigger) or at least one of the selected bits match the input status (OR-function for Exclusive trigger).

Combination (AND-Type): The PA input, 4 or 8 bit, need to fully match the corresponding combination key pattern, except for the 'don't care' bits to trigger reset and/or wake-up.

Exclusive (OR-Type): At least 1 combination key bit matching a selected PA input pin status will trigger reset and/or wake-up if not inhibited by a 'don't care' condition.

10.3.5 Don't care bits

A don't care function is provided so that the input status is only taken into account if the corresponding mask selection bit is set. Special: If all relevant bits of a function are set as 'don't care', then this function is inhibited.

10.3.6 Debouncer

The Reset and the Wake-Up signal have a debouncer selection with either a fast or low speed clock. If the debouncer is selected, the corresponding input signal must remain stable during 1 full debouncer clock period to pass. In sleep mode (no clock) the debouncer is automatically by-passed.
The debouncer clock is coming from the prescaler1, and either Pr1ck[8] (256Hz) or Pr1ck[13] (8kHz) can be selected with **CkDebWk** for wake-up and **CkDebRes** for the reset function.

Note: To avoid detection spikes on input or mask changes it is strongly encouraged to always selecting the debouncer functionality.

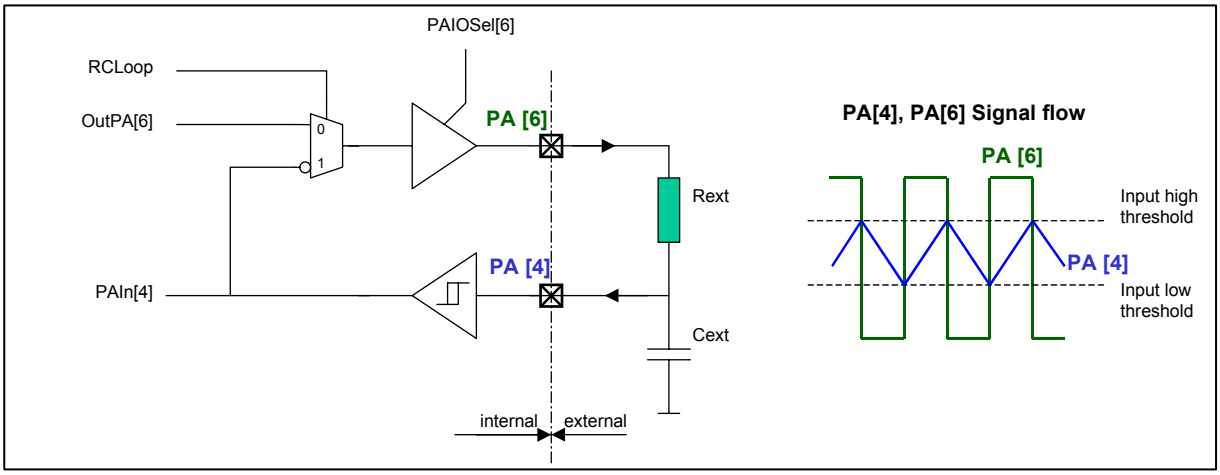
10.4 Oscillation Loop

On pad terminals PA[6] (RCOut) and PA[4] (RCIn) a simple oscillation system can be made using external components (i.e. RC oscillator with resistor from, PA[6] to PA[4] and capacitor from PA[4] to V_{SS}).

The Oscillation loop is configured with the bit **RCLoop** in RegCfgPA. With **RCLoop** set the inverted PA[4] input value is output on PA[6]. The input schmitt-trigger levels give the oscillation signal on PA[4].

For correct operation PA[6] needs to be configured as output and PA[4] as input. Refer to Table 32. Port A IO mode.

Figure 23. Port A oscillation loop



10.4.1 Inverter function

The RC oscillation loop can also be used as a signal inverter. Any signal connected to PA[4] will be feed out inverted on PA[6] if the **RCLoop** bit is set.

10.5 Dual Port RAM interface

The control signals and addresses for the embedded dual port RAM are mapped on port A. Refer to Table 29. Port A External Connectivity.

The involved port A terminals must be set in input mode. Pull selection will be active as specified in Table 32. Port A IO mode.

Please refer to chapter Dual Port Ram for full description.

11 Port B

11.1 Basic features

The port B is an 8-bit general-purpose input/output port. The CPU can read the input state in all modes. All selections concerning the port B are bit-wise executable on PB[0] to PB[7].
These are:
Input / Output selection
CMOS or NCH Open Drain Outputs
Pull resistor selection. Pull-up or Pull-down. When both are selected, pull-up has the priority.

Special features
SPI serial port interface
Frequency outputs
- Timer PWM and Frequency generation
- Divided RC and Crystal oscillator frequencies

Dual Port Ram data bus

Table 37. Port B External Connectivity

Port B output connectivity									
	Direction selection for normal IO function depending on OutEn bits	**Fix direction settings with EnSPI bit for SPI terminals**	**Fix output settings with corresponding EnSig[n] bit set**	**Fix output with bit EnDualRAM & ExtCEn & not ExtWEn**	**Signal1 to 4 selection on PB[3:0] with EnSig bit set** **Sig1Sel[1:0] --> selects special output source for PB[0]** **Sig2Sel[1:0] --> selects special output source for PB[1]** **Sig3Sel[1:0] --> selects special output source for PB[2]** **Sig4Sel[1:0] --> selects special output source for PB[3]**				
PortB	**Normal Output**	**Serial Interface**	**Freq Output**	**Dual Port RAM**	**SigXSel[1,0] '00'**	**SigXSel[1,0] '01'**	**SigXSel[1,0] '10'**	**SigXSel[1,0] '11'**	**Output drive**
PB[7]	PBOut[7]	SIN input	na	DPData[7] output	na	na	na	na	Drive 1
PB[6]	PBOut[6]	SOUT output		DPData[6] output					Drive 1
PB[5]	PBOut[5]	SCLK input/output		DPData[5] output					Drive 1
PB[4]	PBOut[4]			DPData[4] output					Drive 1
PB[3]	PBOut[3]	na	Signal4	DPData[3] output	PWM4	Not PWM3	Pr1Ck[11]	Pr1Ck[0]	Drive 2
PB[2]	PBOut[2]		Signal3	DPData[2] output	PWM3	Not PWM4	Pr2CkSource	Pr1CkSource	Drive 2
PB[1]	PBOut[1]		Signal2	DPData[1] output	PWM2	Not PWM1	Pr1Ck[11]	Pr1Ck[0]	Drive 2
PB[0]	PBOut[0]		Signal1	DPData[0] output	PWM1	Not PWM2	Pr2CkSource	Pr1CkSource	Drive 2

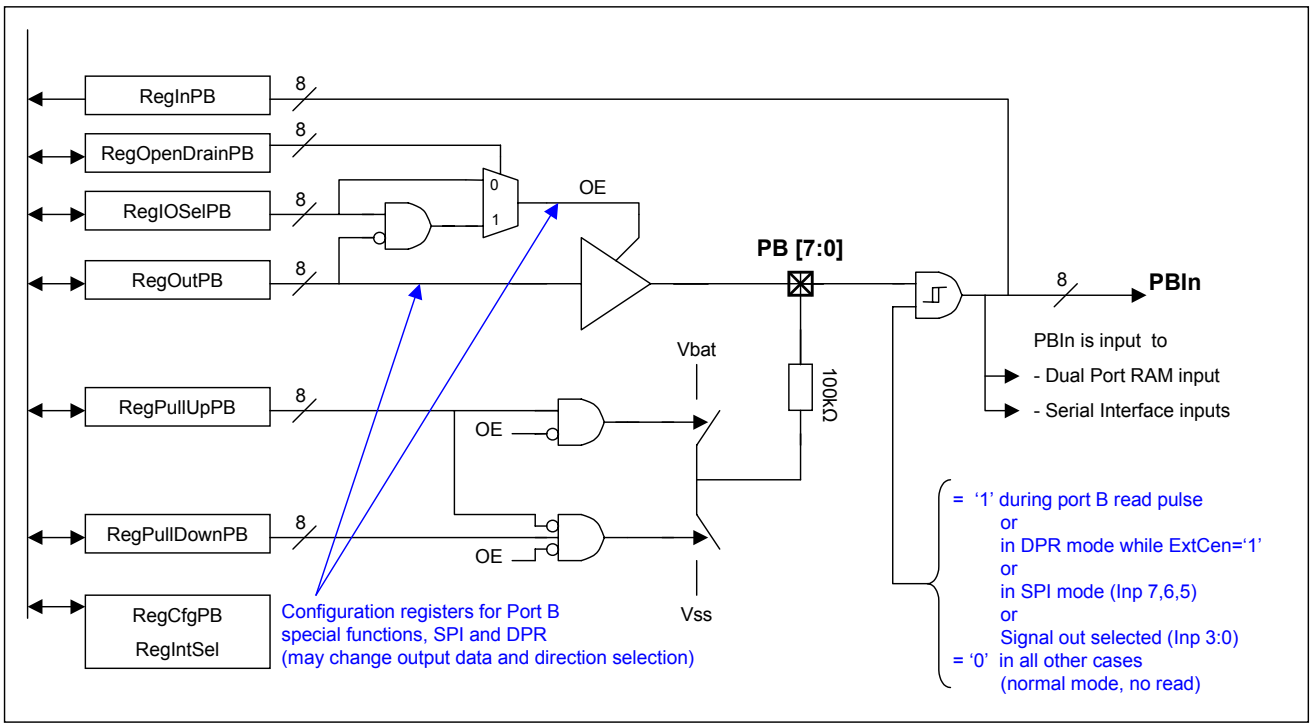
11.1.1 Special function priority handling

- Highest priority is on the Dual Port Ram function which takes full control of the output when the DPR is externally accessed in read mode (**EnDualRAM**='1', **ExtCEn**='1', **ExtWEn**='0').
- Second priority is the Serial interface settings and the Frequency outputs. The serial interface selection with bit **EnSPI** automatically configures the SIN, SOUT and SCLK terminals.
  On the same priority level is also the Frequency outputs. As soon as one of the **EnSig** bits is written the corresponding PB terminal becomes output.
- Last priority has the normal mode data output with **PBOut**[n] and direction with **IOSelPB**[n].

Please note that SPI, DPR and Frequency out selection selections do not change the normal port B output data or direction setting, but force their own IO requirement while used. Refer to chapter 11.4 Special IO operation.

11.1.2 Overview

Figure 24 Port B IO and pull selection in normal mode



11.2 Register map, PB IO functions

Table 38. Port B registers overview

Functions	Register name	Basic function
Base	RegInPB	Direct read of input terminal state
	RegOutPB	Data output register
	RegIOSelPB	Direction selection
Pull resistor and open drain	RegPullUpPB	Pull-up resistor selection
	RegPullDownPB	Pull-down resistor selection
	RegOpenDrainPB	Enable n-channel open drain output
Dual Port RAM selection Freq. output selection Serial Interface selection	RegCfgPB	Configures the corresponding bits accordingly the mode selection. See also 11.1.1 Special function priority handling
Signal selection	RegSigSel	Selection of the internal signal to put on port B

Table 39. Port B registers

RegInPB		**0x30**			**Input register**
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	PBIn[7:0]	--	-	R	Direct read of input terminal state '0'=read low, '1'= read high

RegOutPB		**0x31**			**Output data**
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	OutPB[7:0]	00	ResSys	R/W	Data output register '0'= output low, '1'=output high

RegIOSelPB		**0x34**			**Direction register**
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	IOSelPB[7:0]	00	ResSys	R/W	Direction selection; '1'=Output, '0'=Input

RegPullUpPB		**0x36**			**Pull-up selection**
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	PullUpPB[7:0]	00	ResMain	R/W	Pull-up resistor selection '0'=no pull-up, '1'=pull-up enabled

RegPullDownPB		**0x37**			**Pull-down selection**
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	PullDownPB[7:0]	FF	ResMain	R/W	Pull-down resistor selection '0'=no pull-down, '1'=pull-down enabled (if no pull-up)

RegOpenDrainPB		**0x35**			**N-channel Open drain selection**
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	OpenDrainPB[7:0]	00	ResMain	R/W	N-channel open drain selection (if output) '0'=CMOS mode, '1'= open drain enabled

RegCfgPB		**0x32**			**Port B configuration settings, DPR, SPI, Signals**
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	EnDualRAM	0	ResSys	R/W	Enable the Dual Port RAM
6	EnSPI	0	ResSys	R/W	Enable the Serial Interface function
5	EnSig1	0	ResSys	R/W	Connecting the internal Signal1 on PB[0]
4	EnSig2	0	ResSys	R/W	Connecting the internal Signal2 on PB[1]
3	EnSig3	0	ResSys	R/W	Connecting the internal Signal3 on PB[2]
2	EnSig4	0	ResSys	R/W	Connecting the internal Signal4 on PB[3]
1	-	--	--	R	Reads '0'
0	-	--	--	R	Reads '0'

RegSigSel		0x33			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-6	Sig1Sel[1:0]	0	ResSys	R/W	Select internal Signal1 for output on PB[0]
5-4	Sig2Sel[1:0]	0	ResSys	R/W	Select internal Signal2 for output on PB[1]
3-2	Sig3Sel[1:0]	0	ResSys	R/W	Select internal Signal3 for output on PB[2]
1-0	Sig4Sel[1:0]	0	ResSys	R/W	Select internal Signal4 for output on PB[3]

Internal signal selection for frequency output on port B when corresponding signal enable bit is set.

	SigXSel[1:0]='00'	**SigXSel[1:0]='01'**	**SigXSel[1:0]='10'**	**SigXSel[1:0]='11'**	**Description**
Sig4Sel[1:0]	PWM4	Not PWM3	Pr1Ck[11]	Pr1Ck[0]	Output on PB[3]
Sig3Sel[1:0]	PWM3	Not PWM4	Pr2CkSource	Pr1CkSource	Output on PB[2]
Sig2Sel[1:0]	PWM2	Not PWM1	Pr1Ck[11]	Pr1Ck[0]	Output on PB[1]
Sig1Sel[1:0]	PWM1	Not PWM2	Pr2CkSource	Pr1CkSource	Output on PB[0]

11.3 Normal IO operation

Port B input terminal status can always be read directly. All registers influencing the IO modes are bit-wise selectable. The integrated, switchable pull resistors and the selectable output drive mode allow a maximum of different terminal modes. Refer to Table 32. Port A IO mode for the details.
Default state after power up on all PB terminals is input mode with pull-down resistor.

Table 40. Port B settings in normal mode

Normal mode – No SPI selection – No Frequency out selection – No Dual Port Ram output mode	**IOSelPB[i]**	**OutPB[i]**	**OpenDrainPB[i]**	**PullUpPB[i]**	**PullDownPB[i]**	**PB[i] terminal**	**Notes**
Input mode	0	X	X	0	0	High-Z	Port may be left floating
Input mode with pull-up	0	X	X	1	X	Weak Hi	Pull-up has priority
Input mode with pull-down	0	X	X	0	1	Weak Lo	Default state after Power-up
Output mode, CMOS high drive	1	1	0	X	X	1	Pull resistors disabled
Output mode, CMOS low drive	1	0	0	X	X	0	Pull resistors disabled
Output mode, open drain, high-Z	1	1	1	0	X	High-Z	Pull-down disabled Port may be left floating
Output mode, open drain with pull-up	1	1	1	1	X	Weak Hi	Pull-up active
Output mode, open drain drive low	1	0	1	X	X	0	Pull-up disabled

11.4 Special IO operation

11.4.1 Frequency Output

Signal1 to Signal4, PWM and Prescaler frequencies

Table 41. Port B settings in Frequency out mode

Frequency output terminals - PWM - Pr1ck[n] - Pr1CkSource, Pr2CkSource	**IOSelPB[i]**	**OutPB[i]**	**OpenDrainPB[i]**	**PullUpPB[i]**	**PullDownPB[i]**	**PB[i] terminal**	**Notes**
Output mode, CMOS high drive	X	X	0	X	X	Signal	Selected signal output
Output mode, NCH open drain	X	X	1	0	X	0 / High-Z	Selected signal output, needs external pull-up
Output mode, NCH open drain, Pull-up	X	X	1	1	X	0 / Weak Hi	Signal output, pull-up intern

11.4.2 SPI outputs

SPI: SOUT, SCLK (in master mode)

Table 42. Port B settings for SPI outputs

SPI output terminals – SOUT – SCLK in master mode	**IOSelPB[i]**	**OutPB[i]**	**OpenDrainPB[i]**	**PullUpPB[i]**	**PullDownPB[i]**	**PB[i] terminal**	**Notes**
Output mode, CMOS high drive	X	X	0	X	X	SOUT SCLK	SPI outputs
Output mode, NCH open drain	X	X	1	0	X	0 / High-Z	SPI outputs, needs external pull-up
Output mode, NCH open drain, Pull-up	X	X	1	1	X	0 / Weak Hi	SPI outputs, pull-up intern

11.4.3 SPI inputs

SPI: SIN, SCLK (in slave mode)

Table 43. Port B settings for SPI inputs.

SPI input terminals – SIN – SCLK in slave mode	**IOSelPB[i]***	**OutPB[i]**	**OpenDrainPB[i]**	**PullUpPB[i]**	**PullDownPB[i]**	**PB[i] terminal**	**Notes**
SCLK, SIN input	X	X	X	0	0	High-Z	No pull, Terminals must be driven externally
SCLK, SIN input with pull-up	X	X	X	1	X	Weak Hi	Pull-up has priority
SCLK, SIN input with pull-down	X	X	X	0	1	Weak Lo	Default state after Power-up

11.4.4 Dual Port RAM terminals

Dual Port Ram output (while EnDualRAM='1', ExtCEn='1', ExtWrEn='0')

Table 44. Port B settings for DPR outputs

Dual Port RAM data output – DPData[7:0]	**IOSelPB[i]***	**OutPB[i]**	**OpenDrainPB[i]**	**PullUpPB[i]**	**PullDownPB[i]**	**PB[i] terminal**	**Notes**
Output mode, CMOS high drive	X	X	0	X	X	DPData	DP outputs
Output mode, NCH open drain	X	X	1	0	X	0 / High-Z	DP outputs, needs external pull-up
Output mode, NCH open drain, Pull-up	X	X	1	1	X	0 / Weak Hi	DP outputs, pull-up intern

In all other Dual Port Ram cases (not during read access), the port B terminals are configured based on the normal mode bit settings. The only difference is that the port B inputs must not be left floating.

12 Serial Port Interface

12.1 Basic features:

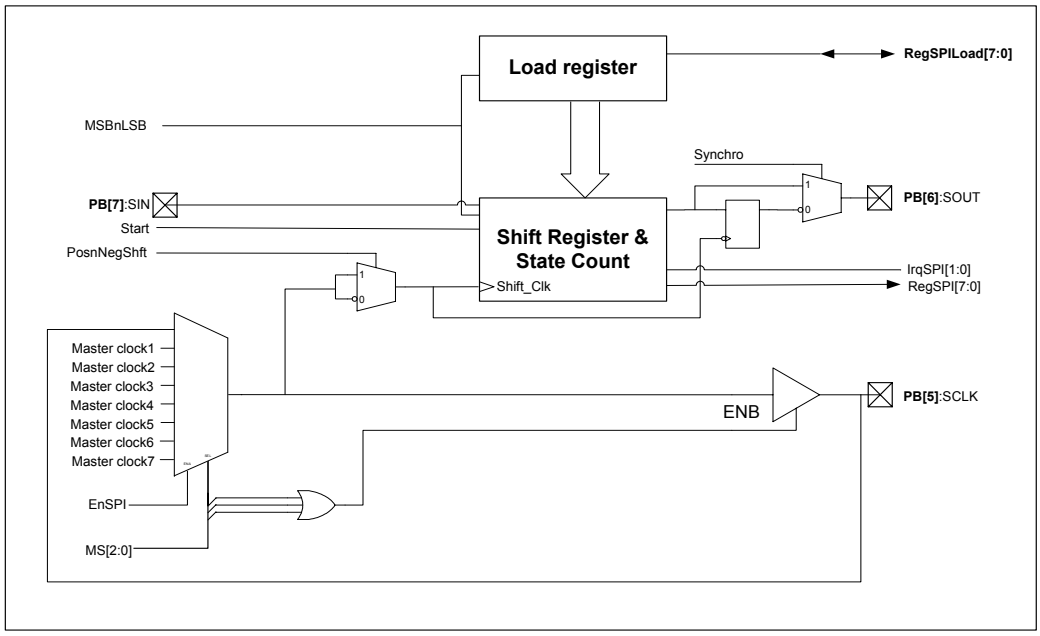
The SPI is a 3-wire serial interface, its inputs and outputs are mapped on the port B in following manner:

- SIN: Serial Input Data PB[7]
- SOUT: Serial Output Data PB[6]
- SCLK: Serial Clock PB[5]

- Master or Slave byte-wise serial communication.
  - Master mode: 6 internal clock sources (prescaler frequencies or from selected timer1 periods) and 1 external clock input on PA[5].
  - Slave Mode : External clock source from port B, PB[5]
- Transmission order, LSB or LSB first selection.
- The active edge of the serial interface is selectable (positive or negative edge)
- Data output synchronization with the opposite shift clock.
- Auto-Start mode; which allows to have a fix data stream output (UART support)
- Two maskable interruptions are generated.
  - Beginning of the transmission
  - End of the transmission.

12.1.1 Overview:

Figure 25 : SPI architecture



12.1.2 SPI terminal configuration

When the SPI is enabled **EnSPI=**'1', the pads used in the port B for the transmission are automatically set in input or output mode according to the configuration of the SPI. The pull resistors as well as the possible open-drain selections depend on port B settings for the terminal direction.

Table 45. Terminal configurations

Pads	SPI pins	Direction	Pull-down	Pull-up	Open-drain
PB[7]	SIN	Input	Selectable	Selectable	n.a.
PB[6]	SOUT	Output	Not selectable	Selectable in open-drain mode	Selectable
PB[5]	SCLK	Output in master mode. Input in slave mode	Selectable in slave mode	Selectable in slave or open-drain master mode	Selectable in master mode

12.2 Functionality

12.2.1 Master and Slave modes

In slave mode SCLK is given externally and input on PB[5]. In master mode, SCLK is generated by the SPI and output on PB[5].
Master/Slave selection as well as active clock selection in Master mode is done with the bits **MS[2:0]** in RegSPICfg. Maximum Clock frequency in master mode is ½ of the internal high-speed clock (= 5MHz in case of RC 10MHz). Different fix prescaler based clock frequencies and timer1 PWM frequencies are available as input clock.

- MS[2:0] = 000 : Slave mode. SCLK = PB[5]
- MS[2:0] = 001 : Master mode, SCLK = PWM1 (internal PWM signal coming for timer1)
- MS[2:0] = 010 : Master mode, SCLK = PA[5] (PA[5] input terminal)
- MS[2:0] = 011 : Master mode, SCLK = Prc2CkSource (clock source of prescaler2)
- MS[2:0] = 100 : Master mode, SCLK = Pr2Ck[9] (Output clock from prescaler2 bit[9])
- MS[2:0] = 101 : Master mode, SCLK = Pr2Ck[8] (Output clock from prescaler2 bit[8])
- MS[2:0] = 110 : Master mode, SCLK = Pr1CkSource (clock source of prescaler1)
- MS[2:0] = 111 : Master mode, SCLK = Pr1Clk[14] (Output clock from prescaler1 bit[14])

In master mode, SCLK is generated from the beginning until the end of the transmission. It is not necessary to enable and disable the SPI for each burst. After each transmission, **Start** is automatically reset after 8 SPI clocks, except in cases where the Load value is rewritten during the data transfer (AutoStart, fix data stream output).

12.2.2 Fix data stream Output (Auto-Start)

A new transmission will immediately follow the current transmission if the CPU writes in RegSPILoad while the Start = '1'. Load_nShift is a flag indicating that the SPI is actually loading a value from RegSPILoad (Load_nShift = '1') or if a transmission is in progress (Load_nShift = '0'). The new RegSPILoad value must be loaded while **LoadnShift** is '0' (during shift operation). A re-load after end of transmission, **Start** reads '0', will not trigger a new transmission. IrqSPI[1] can be used to determine the reload time.

Note: In Auto-Start mode, the SPI should run slower then the CPU especially when the CPU access RegSPIDat at the end of the transmission.

12.2.3 SPI Interruptions

There are to interrupts generated by the SPI:

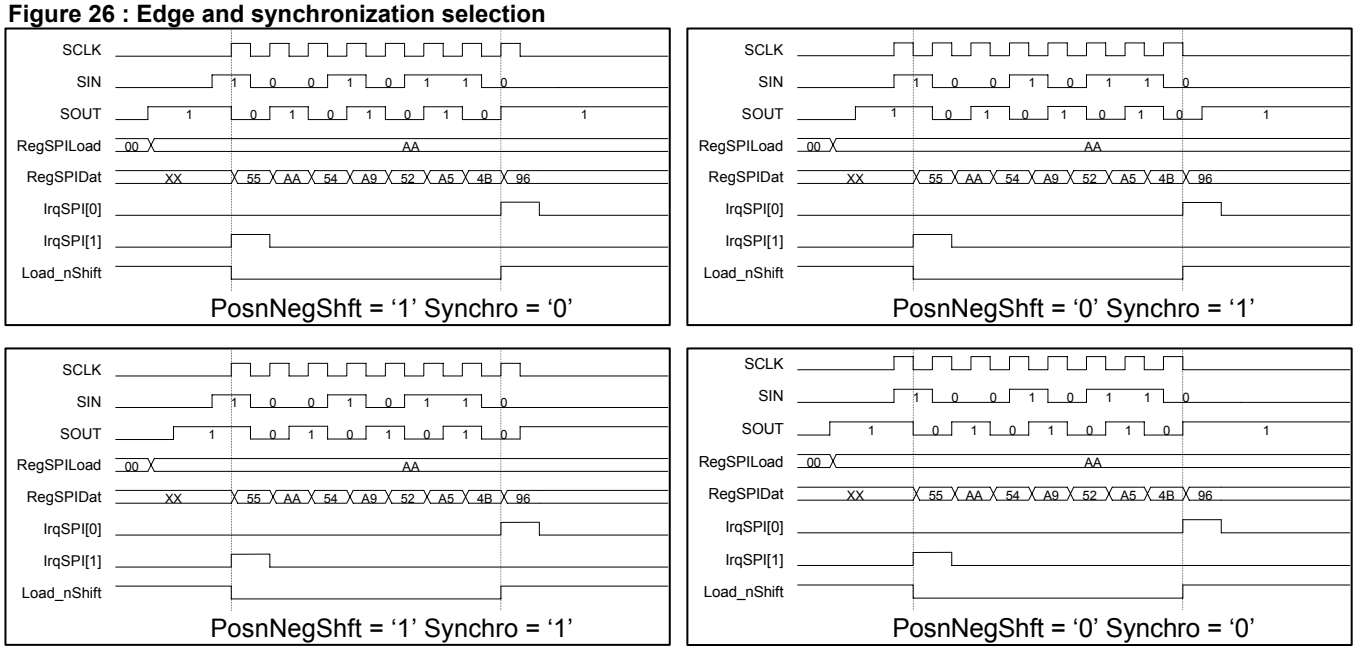
- IrqSPI[1]: Generated at the beginning of the transmission on the first active edge of SCLK. It can be used in Auto-Start mode to force the CPU to write the next value to transmit in RegSPILoad.
- IrqSPI[0]: Generated at the end of the transmission on the last active edge of SCLK.

Both interruptions are maskable with RegMsk20[5:4]. They are on priority 2 in RegInt20[5:4].

12.2.4 SPI edge and synchronization selection

Depending on the protocol, the SPI can shift the data's on falling or rising edge of SCLK with bit **PosnNegShft** in register RegSPICfg. It is possible to resynchronize SOUT on the opposite edge shift clock with bit **Synchro** in register RegSPICfg:

Figure 26 : Edge and synchronization selection



12.2.5 SPI start-up

The SPI is enabled by bit **EnSPI** in RegCfgPB. Before enabling the SPI its configuration must be set. Configuration means, the clock selection, the edge selection and the synchronization of SOUT selection. If one of this parameter change while **EnSPI** = '1', the transmitted data are not guaranteed. The last operation is to set **Start** to launch the transmission.

- 1st: Configuration settings : Clock, Edge, synch and MSB/LSB selection
- 2nd: Enabling the SPI: bit **EnSPI** in RegCfgPB
- 3rd: Write the load data in RegSPILoad
- 4th: Start the transmission with bit **Start** in RegSPICfg

When the transmission is finished, RegSPIDat contains the received value coming through SIN.

Note: This register is accessible in read mode only. It is possible to read it at any time but the data is not guaranteed during the transmission in slave mode and may not be the final value while Load_nShift = '0'.
The value is guaranteed when **Load_nShift** = '1'. It is possible to use the interrupt IrqSPI[0] to start to read this register.

The **Start** bit can be used to determine if a shift operation is ongoing or has finished. In Master mode the Start flag is synchronized with the next inversed active clock edge after writing the start bit. (The user writes the start bit but reads the start flag). In slave mode the start flag becomes active immediately after writing the start bit. After a full byte transfer the Start bit and also the start flag are reset. This may be used to determine end of transfer by software polling.

12.2.6 MSB or LSB first selection

By default, **MSBnLSB** = '0', the first transmission bit (receive or send) is the LSB. MSB first can be selected by writing **MSBnLSB**='1'. The selection must be performed at the SPI setup, before loading or reading the transmission value.

12.3 Registers overview:

Table 46. SPI registers

RegSPICfg		0x38			SPI configuration
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	Start	0	ResSysSlp	R/W	Start the transmission
6	MS2	0	ResSys	R/W	Selects master or slave mode. Used to select the clock in master mode as well.
5	MS1	0	ResSys	R/W	
4	MS0	0	ResSys	R/W	
3	PosnNegShft	0	ResSys	R/W	Select the active edge of the shift register
2	MSBnLSB	0	ResSys	R/W	Select MSB or LSB first. '0' = LSB first.
1	Synchro	0	ResSys	R/W	SOUT is synchronized on the opposite edge
0	Load_nShift	1	--	R	Flag indicating if the SPI is in load or transmit mode.

MS2	MS1	MS0	Mode	Clock source
0	0	0	Slave	External clock from SCLK PB[5]
0	0	1	Master	PWM1 from timer1 or timer12
0	1	0	Master	External clock from PA[5] input
0	1	1	Master	Pr2CkSource
1	0	0	Master	Pr2Ck[9]
1	0	1	Master	Pr2Ck[8]
1	1	0	Master	Pr1CkSource
1	1	1	Master	Pr1Ck[14]

RegSPIDat		0x39			SPI data register
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	SPIDat[7:0]	0	ResSys	R	Shift register status

RegSPILoad		0x3A			SPI load register
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	SPILoad[7:0]	0	ResSys	R/W	Buffer used to load the data to send through SOUT

RegCfgPB		0x32			Port B configuration
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	*EnDualRAM*	*0*	*ResSys*	*R/W*	*Enable the Dual port RAM*
6	EnSPI	0	ResSys	R/W	Enable the serial port interface
5	*EnSig1*	*0*	*ResSys*	*R/W*	*Enable the internal signal 1 driving PB[0]*
4	*EnSig2*	*0*	*ResSys*	*R/W*	*Enable the internal signal 1 driving PB[1]*
3	*EnSig3*	*0*	*ResSys*	*R/W*	*Enable the internal signal 1 driving PB[2]*
2	*EnSig4*	*0*	*ResSys*	*R/W*	*Enable the internal signal 1 driving PB[3]*
1	--	--	--	*R*	*Read always 0*
0	--	--	--	*R*	*Read always 0*

Table 47. SPI interrupt mapping

Interrupt source	Priority	IntCtrl connection
IrqSPI[1:0]	2	Int2[5:4]

13 Timers

13.1 Basic features:

The EM6812 contains 4 identical, individual configurable 8-bit timers. They may be used standalone or chained together as 2 16-bit timers. Possible configurations are

- 4 x 8-bit: Timer1, Timer2, Timer3, Timer4
- 2 x 8-bit, 1 x 16-bit Timer1, Timer2, Timer34
  Timer12, Timer3, Timer4
- 2 x 16-bit Timer12, Timer34

Each timer can work with input signals from port A (clock start, stop, input capture), output PWM or frequency signals on port B, and generating interrupt signals on ‘zero’ or 'compare' match conditions.

Each timer has its own configuration bits and setup registers.

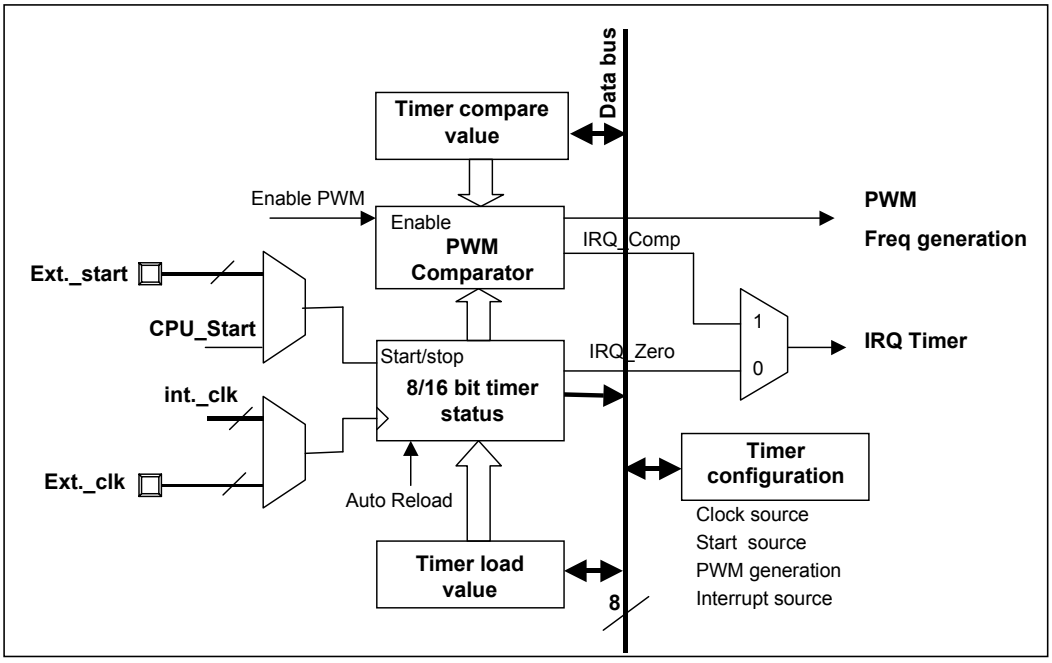
- Input clock source (external inputs and prescaler clock frequencies), select 1 out of 8.
- Start condition (software write or external condition); select 1 out of 8.
- Auto-Reload or Zero-Stop mode.
- Pulse Width Modulation (PWM) and Frequency generation.
- Interrupts source (compare or zero).
- Timer value read on the fly.

When chained together as a 16bit Timer34, the Timer4 will take over the full configuration of the Timer3. Same thing applies for Timer12.

The basic timer function is counting from the start value down to hex 00, then an interrupt is generated and the timer stops. With the autorelod bit set, the timer reloads the start value after reaching zero and so runs in an endless loop until stopped. At every zero detect an interrupt signal is generated.

With PWM bit set, a pulse width modulated signal can be output directly and inverted on port B. Pulse width and period depend on the timer compare and timer load value. Either the compare condition or zero condition can create an interrupt. Frequency generation is done in PWM mode by selecting the desired signal period (load value) and putting the compare value on half the period.

Figure 27. Timer architecture



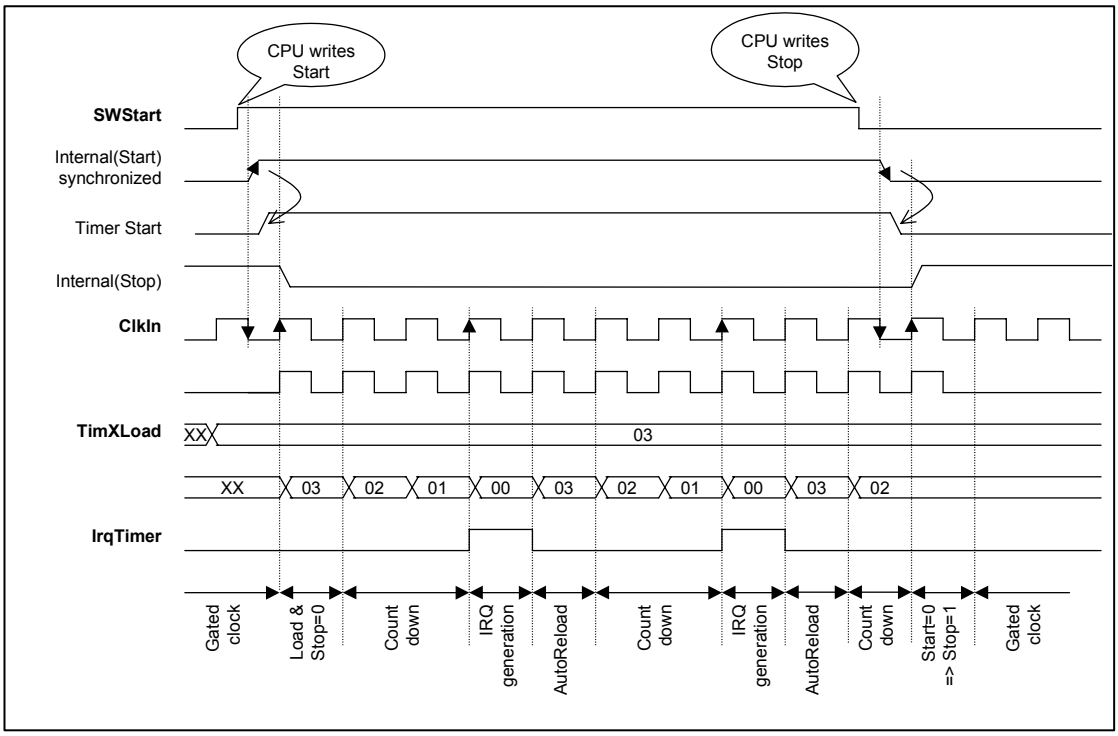
13.2 Functionality

13.2.1 Auto-Reload mode

"Auto-Reload" mode means that after having down-counted from the start value in **RegTimXLoad** to zero, the timer restart down-counting automatically from the start value. If the start value changes during the down counting, the timer waits until the end of the loop before loading the new start value. For Auto-Reload mode, the bit **AR**X in RegTimersCfg must be set. The period in Auto-Reload mode is equal to the **RegTimXLoad + 1** value.

Startup synchronization is based on the first negative edge of the selected clock source after the start condition was fulfilled. After the startup phase, the **RegTimXLoad** value is transferred into the timer and down counting starts on the next active clock edge. After every zero detection the timer value is loaded again, and if **RegTimXLoad** was altered, the new value will be loaded.
The timer stops at the first active clock edge following the removed start condition. Also, when the **AR** bit is cleared during down counting, the timer will stop when reaching zero (= Zero-Stop mode). At every zero crossing an interrupt **IRQTim**X will be generated.

Figure 28. Timing diagram in Auto-Reload mode (SWStart)



Stopping the timer during down-count
Will freeze the timer on the current value (positive timer input clock synchronization)

Restart of the stopped timer during down-count
is synchronized based on the negative edge of the selected timer input clock source. As such it acts like an initial timer start and will start by loading the TimXLoad value and start the down-count.
Special cases apply if the timer is stopped for short periods below 1½ timer-input clocks. In such cases the count value may be enlarged by one unit (stop seen, but restart just afterwards) or the TimXLoad may not take place (stop not seen)
For proper restart, the internal, synchronized Start signal must go low for at least 1 full clock period.

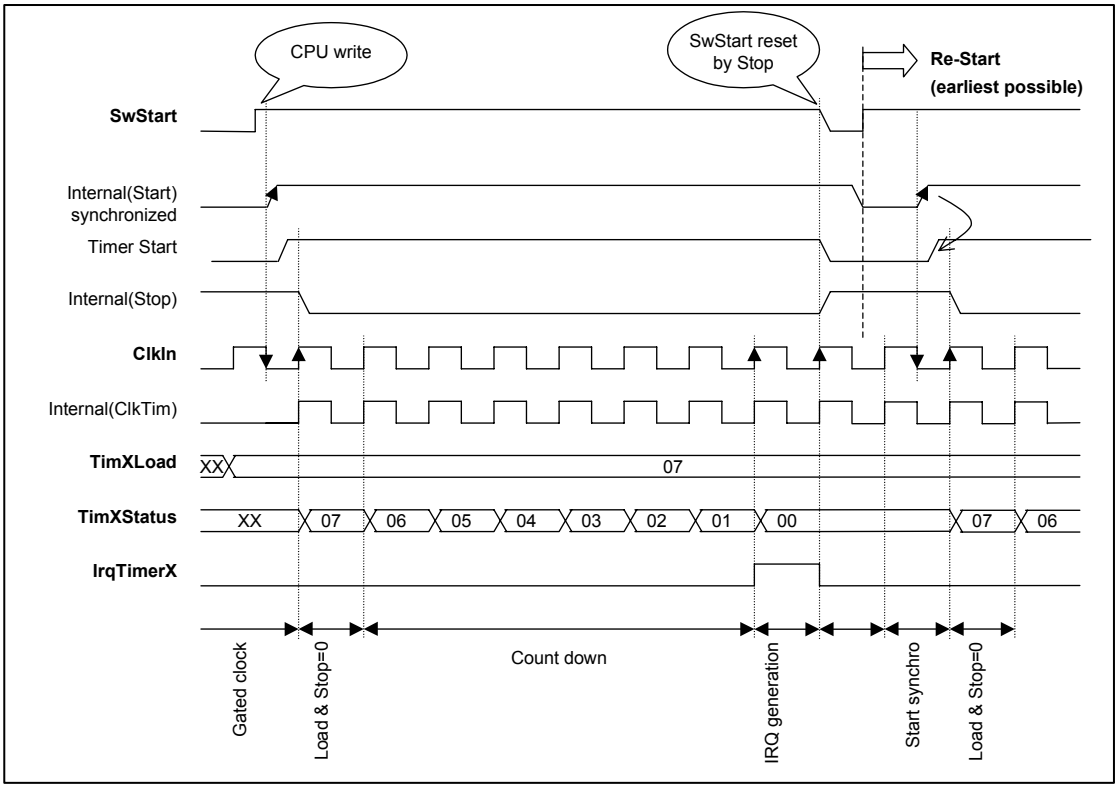
Note:
Above mentioned timer restart delay times can be drastically reduced when after going into the stop condition the timer frequency is temporarily set to the highest available frequency (i.e. setting Pr2CkSource, NOP, going back to original clock).

13.2.2 Zero-Stop mode

In “Zero-Stop” mode, the timer stops counting after reaching zero. It generates an interrupt **IRQTim**X and resets the Start bit.

Startup synchronization is based on the first negative edge of the selected clock source after the start condition was fulfilled. After the startup phase, the **RegTimXLoad** value is transferred into the timer and down counting starts on the next active clock edge. Counting is based on the positive edge of the selected timer input clock source.

Figure 29. Timing diagram in Zero-Stop mode (SWStart)



Restart of the stopped timer after 0 detect
is synchronized based on the negative edge of the selected timer input clock source. The timer stop condition must be valid for 2 full timer clock periods before being able to restart. This means that the timer can be restarted ½ clock period after the **SWStart**X bit was automatically cleared or 1½ clock timer clock cycles after the interrupt 0 detect.
Not respecting of this restart delay may prevent timer reload, its value will remain at 00 and no new IRQ will be generated independent of the fact that its start condition may be true.

Stopping the timer during down-count
Will freeze the timer on the current value (positive timer input clock synchronization)

Restart of the stopped timer during down-count
is synchronized based on the negative edge of the selected timer input clock source. As such it acts like an initial timer start and will start by loading the TimXLoad value and start the down-count.
Special cases apply if the timer is stopped for short periods below 1½ timer-input clocks. In such cases the count value may be enlarged by one unit (stop seen, but restart just afterwards) or the TimXLoad may not take place (stop not seen)
For proper restart, the internal, synchronized Start signal must go low for at least 1 full clock period.

Note:
Above mentioned timer restart delay times can be drastically reduced when after going into the stop condition the timer frequency is temporarily set to the highest available frequency (i.e. setting Pr2CkSource, NOP, going back to original clock).

13.2.3 Start control system

There are two principal ways to start the counter:

- Internal start:
  - Software writing the SWStart bit.
- External signals on port A input:
  - State condition: External start on input state (down-counting while condition true)
  - Pulse condition: External start on input pulse, stop on next pulse.
- The selection of the start source and the corresponding port A input is done in register RegTimXCfg bits StartXSel[2:0].
- CPU controlled software Start or State / Pulse Start condition in case of external start is defined in registers RegTimersStart.
- The **SwStart**X bits can be used as busy flags in case of software start. If the timer is started by port A input conditions, the SwStartX bits will remain '0', busy information can be derived from reading the timers status value or by detecting the start / stop conditions.

Each timer has 8 different start sources. Software start is selected by default, activated by writing bit **SWStart**X in register RegTimersStart. The selection of other start sources is done with **StartXSel**[2:0] in RegTimXCfg.

Table 48. Start selection

	Timer1	**Timer2**	**Timer3**	**Timer4**
StartXSel[2:0]	Selection with Start1Sel[2:0]	Selection with Start2Sel[2:0]	Selection with Start3Sel[2:0]	Selection with Start4Sel[2:0]
000	Soft start **SwStart1**	Software with **SwStart2**	Software with **SwStart3**	Software with **SwStart4**
001	Ext. start PA1	Ext. start PA0	Ext. start PA0	Ext. start PA0
010	Ext. start PA2	Ext. start PA2	Ext. start PA1	Ext. start PA1
011	Ext. start PA3	Ext. start PA3	Ext. start PA3	Ext. start PA2
100	Ext. start PA4	Ext. start PA4	Ext. start PA4	Ext. start PA4
101	Ext. start PA5	Ext. start PA5	Ext. start PA5	Ext. start PA5
110	Ext. start PA6	Ext. start PA6	Ext. start PA6	Ext. start PA6
111	Ext. start PA7	Ext. start PA7	Ext. start PA7	Ext. start PA7

Start selection must not be changed while the timer is running.

13.2.3.1 Software Start Condition

In Software start, the CPU writes the start condition **SWStart**X='1'. The software start gets synchronized on the next falling edge of the selected counter clock. The positive edge synchronization signal then enables the counter function.
The counter is stopped by either writing the SWStartX='0' or on Zero-Stop mode when the counter value reaches 0. Please refer to Figure 28. Timing diagram in Auto-Reload mode (SWStart) and Figure 29. Timing diagram in Zero-Stop mode (SWStart) for more details

13.2.3.2 External Signal State Condition

The Counter is active as long as the specified input pin reads '1'. In Zero-Stop mode it will stop as soon as the counter reaches 0.

The external state condition is first synchronized with the negative edge counter clock. The positive edge of this synchronized signal will enable the timer start. The basic restart conditions as described in 13.2.1 and 13.2.2 apply.

Figure 30. External start: State condition, Auto-Reload mode.

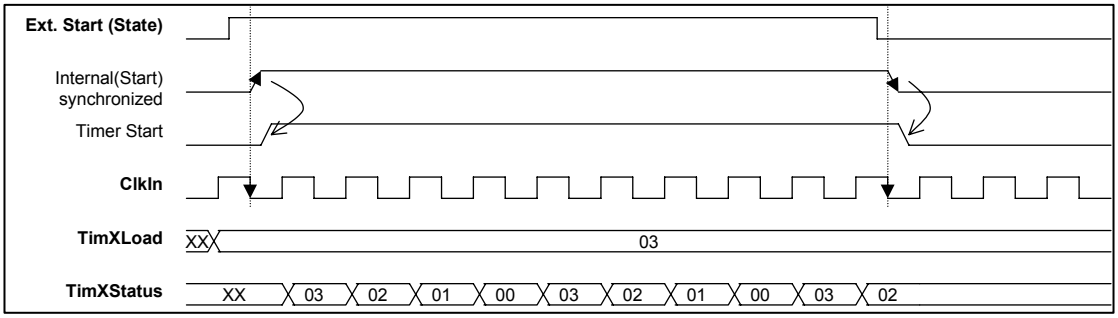
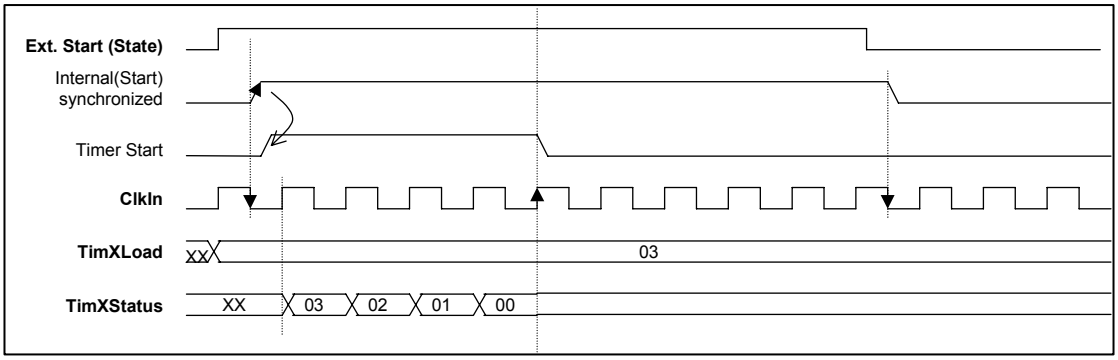


Figure 31. External start: State condition, Zero-Stop mode.



13.2.3.3 External Signal Pulse Condition

The Counter function gets activated by the first positive edge of the selected input signal, the following positive edge will stop the counter. In Zero-Stop mode it will stop as soon as the counter reaches 0.

The internal Pulse Start recognition signal is first synchronized with the negative edge counter clock. The positive edge of this synchronized signal will enable the timer start. The basic restart conditions as described in 13.2.1and 13.2.2 apply.

The pulse start recognition signal is set by the first positive edge of the external start condition and reset by either the following positive edge of the input or it will also be reset when the counter value reaches 0.

Figure 32. External start: Pulse condition, Auto-reload mode.

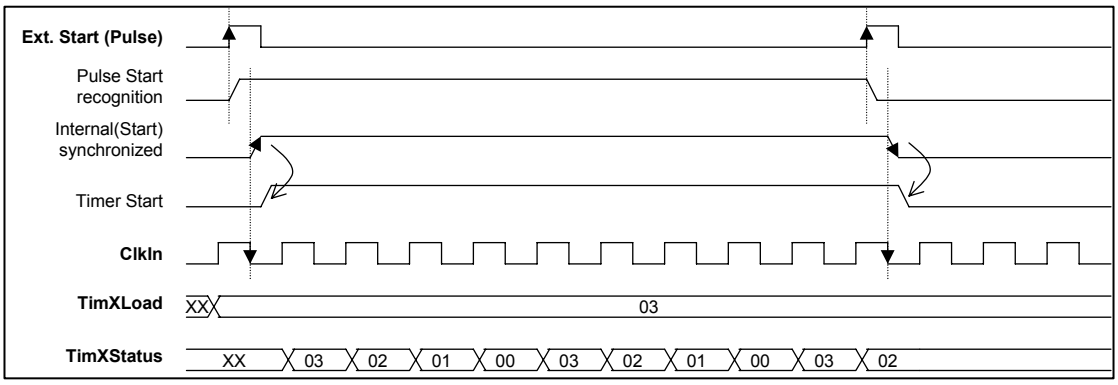
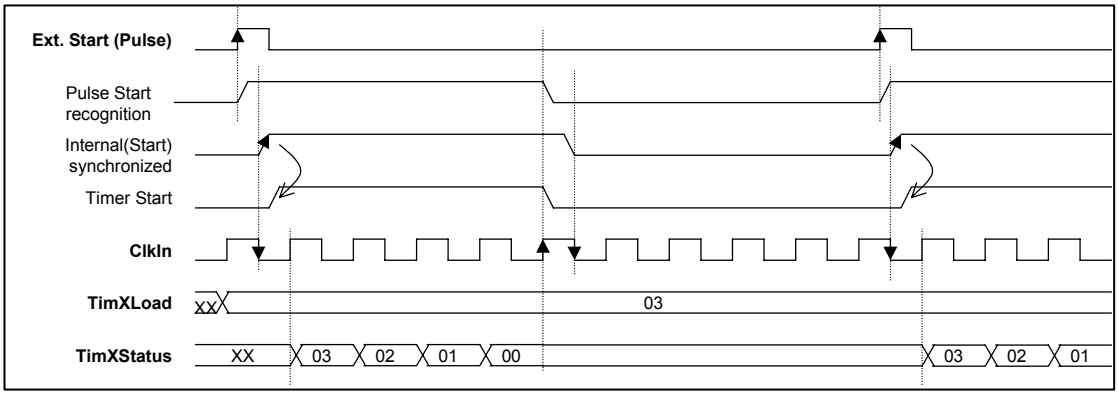


Figure 33. External start: Pulse condition, Zero-Stop mode.



13.2.4 Stopping the timer

The timer stops always at the next clock following a valid stop condition. These are:

- Reaching hex 00 in Zero-Stop mode.
- In software start control if **SWStart** is cleared.
- In external start control, on input state if the selected input value becomes 0.
- In external start control, on input pulse, the timer stops at the 2nd pulse.

Refer also to previous timing diagrams and hints for restart.

13.2.5 Clock selection

There are 8 different clock sources for each timer. Some of them come from external sources. The others are from the prescaler1 or prescaler2. The selection is done with **ClkXSel**[2:0] in RegTimXCfg.

Table 49. Input clock selection

	Timer1	Timer2	Timer3	Timer4
ClkXSel[2:0] value	**Selection with Clk1Sel[2:0]**	**Selection with Clk2Sel[2:0]**	**Selection with Clk3Sel[2:0]**	**Selection with Clk4Sel[2:0]**
000	PA0	PA1	PA2	PA3
001	PA4	PA5	PA6	PA7
010	Pr2CkSource	Pr2CkSource	Pr2CkSource	Pr2ClkSource
011	Pr2Ck[8]	Pr1CkSource	Pr2Ck[8]	Pr1ClkSource
100	Pr2Ck[6]	Pr1Ck[14]	Pr2Ck[4]	Pr1Ck[13]
101	Pr1CkSource	Pr1Ck[12]	Pr1CkSource	Pr1Ck[11]
110	Pr1Ck[13]	Pr1Ck[10]	Pr1Ck[13]	Pr1Ck[9]
111	Pr1Ck[11]	Pr1Ck[8]	Pr1Ck[9]	Pr1Ck[7]

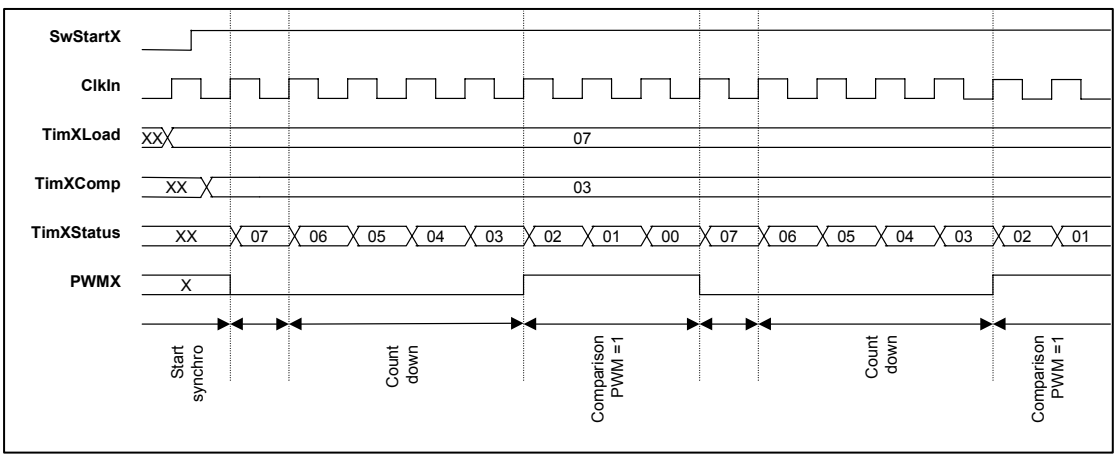
Clock selection should only be changed while the timer is stopped to avoid timer clock glitches, which may influence the actual counter value.

13.2.6 PWM and Frequency generation

With the pulse width modulation function, is possible to generate signals of a defined frequency and duty cycle. These signals can be output on the port B as frequencies or PWM. The function of the PWM is based on the comparison of the actual timer value and a compare value. PWM = '0' when the timer starts counting until it reaches the comparison value, then PWM='1' until the end of the loop when the timer reaches the value 0. See figure below for more details.

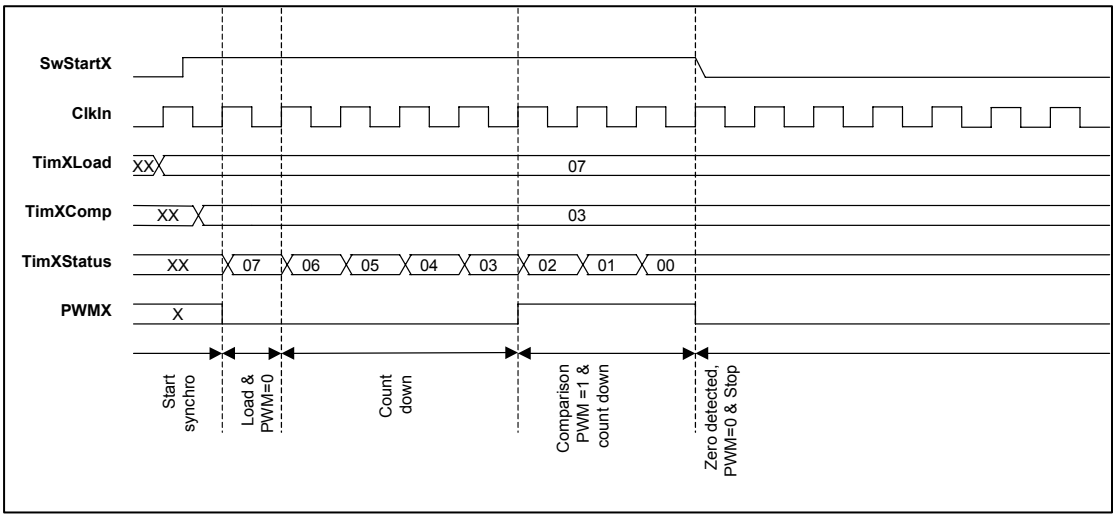
The PWM function is enabled with bit **EnPWM**X in RegTimXCfg. This bit enables the comparator and also routes the PWM signal to the port B. Compare match interrupt can be generated in PWM mode. The bit **TimEq**X allows to select between timer compare match IRQ or timer zero detect IRQ (default value). Also refer to chapter 13.2.8 Interrupts.

Figure 34. PWM or Frequency generation in Auto-Reload mode.



In Zero-Stop mode the PWM becomes '0' and stops after zero detection. See below.

Figure 35. PWM or Frequency generations in Zero-Stop mode.



13.2.6.1 Frequency and Duty cycle Calculation

$$F_{PWM} = \frac{F_{ClkTim}}{(TimXLoad + 1)}$$

$$DutyCyle = \frac{TimXComp}{TimXLoad + 1} \cdot 100$$

13.2.6.2 Frequency generation

Frequency generation is an extension of the PWM function. The only difference is that in Frequency generation the duty cycle and the signal period both change. The period adjustment is made with the autoreload load value. Whenever the load value is changed, it will be applied on the next following loop after the zero crossing.

13.2.7 16-bits configuration

Timer1 and timer2 can form together a 16-bit timer12. So can timer3 and timer4 which form timer34. In this configuration, timer1 (or timer3) becomes the master configuration for start source, clock source, the PWM mode, Auto-Reload or Zero-Stop mode and IRQ selection.
The LSB comes from the timer1 and timer3 and the MSB from timer2 and timer4.
The functionality remains identical to the standalone 8-bit timers but the load, compare and status values are split in two registers: RegTimXStatus, RegTimXLoad and RegTimXComp.

To merge timer1 and timer2 the bit **EnTim12** in RegTimersCfg must be set at '1'.
To merge timer3 and timer4 the bit **EnTim34** in RegTimersCfg must be set at '1'.

With timers merged the specific selection bits for timer2 or 4 have no function anymore and the interrupt source from the slave timers (2 or 4) will be inactive.

13.2.8 Interrupts

Each timer has 2 selectable interrupt sources:
- IrqTimer1 (zero detect or compare match)
- IrqTimer2 (zero detect or compare match)
- IrqTimer3 (zero detect or compare match)
- IrqTimer4 (zero detect or compare match)

Table 50. Timer interrupts selection

EnPWMX	**TimEq**X	**Interrupt**
0	0	Irq timer on 'Zero detect'
0	1	No timer Irq generated
1	0	Irq timer on 'Zero detect'
1	1	Irq timer on 'Compare match'

All these interrupt are in priority level 2 mapped in register RegInt20[3:0]. The source is individually selectable and each interrupt may be masked in register RegMsk20[3:0]. The **TimEq** selection is only valid when the corresponding **EnPWM**X bit is set.

In PWM mode the interruption can be generated when the timer reaches the value 0 or when it reaches the comparison value with **TimEq**X in RegTimXCfg. If needing both interrupts, the user may change the IRQ source after each event. Another solution consists of routing a PWM from port B output onto a port A input and configure this input as IRQ input (slave timer interrupt).

If configured as 16-bit timer, only the master timer (1 or 3) will generate interrupts.

Figure 36. Interrupts generation in PWM mode.

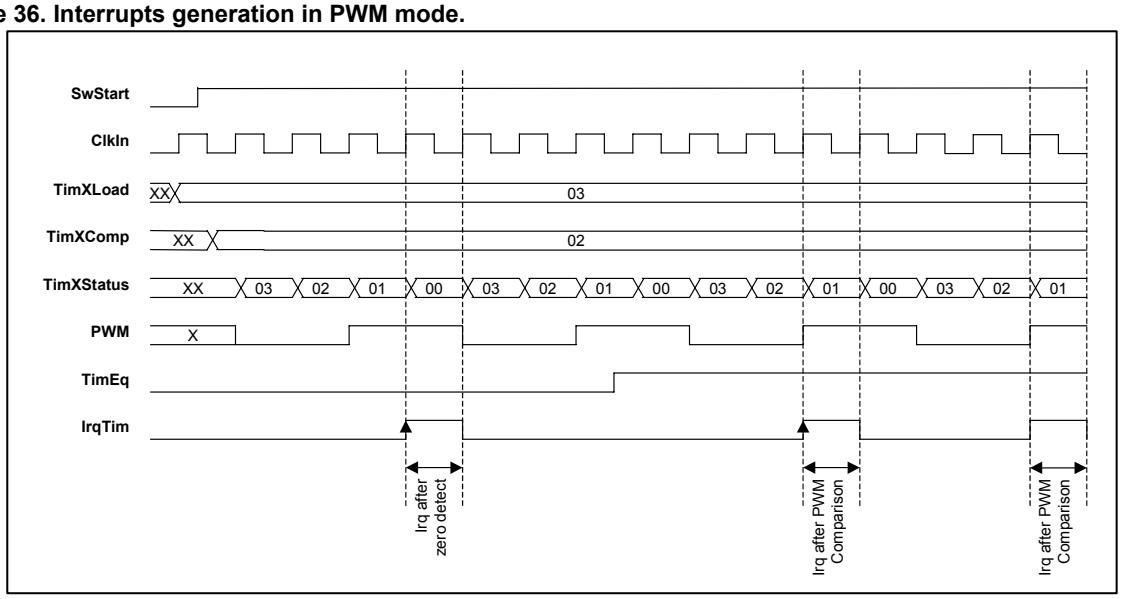


Table 51. Timer interrupts mapping

Interrupt source	**Priority**	**IntCtrl connection**
IrqTimer1	2	Int2[0]
IrqTimer2	2	Int2[1]
IrqTimer3	2	Int2[2]
IrqTimer4	2	Int2[3]

13.3 Recommended programming order

1st: Select the general timer configuration in RegTimersCfg.
2nd: Select the specific timer configuration RegTimXCfg.
3rd: Write the necessary load and compare values RegTimXLoad, RegTimXComp.
4th: Start the timer (software start) or enable the start condition in case of external start, RegTimerStart.

Note: Do not change the configuration while running. Clock glitches may occur and influence the result.

13.4 Registers overview:

13.4.1 General configuration registers

Table 52. General Timer configuration

RegTimersCfg		**0x4E**			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	EnTim12	0	ResSys	R/W	Timer1 and timer2 merge in 16bits timer if = '1'
6	EnTim34	0	ResSys	R/W	Timer3 and timer4 merge in 16bits timer if = '1'
5	AR1	0	ResSys	R/W	Set the timer1 in Auto-Reload mode if = '1'
4	AR2	0	ResSys	R/W	Set the timer2 in Auto-Reload mode if = '1'
3	AR3	0	ResSys	R/W	Set the timer3 in Auto-Reload mode if = '1'
2	AR4	0	ResSys	R/W	Set the timer4 in Auto-Reload mode if = '1'
1	--		--		
0	--		--		

RegTimersStart		**0x4F**			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	SWStart1	0	ResSys	R/W	Timer1 software start-bit
6	Tim1Pulse	0	ResSys	R/W	Timer1 start on state if = '0', on pulse if = '1')
5	SWStart2	0	ResSys	R/W	Timer2 software start-bit
4	Tim2Pulse	0	ResSys	R/W	Timer2 start on state if = '0', on pulse if = '1')
3	SWStart3	0	ResSys	R/W	Timer3 software start-bit
2	Tim3Pulse	0	ResSys	R/W	Timer3 start on state if = '0', on pulse if = '1')
1	SWStart4	0	ResSys	R/W	Timer4 software start-bit
0	Tim4Pulse	0	ResSys	R/W	Timer4 start on state if = '0', on pulse if = '1')

13.4.2 Timer1 configuration

Table 53. Timer1 configuration

RegTim1Cfg		0x50			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	EnPWM1	0	ResSys	R/W	Enable PWM on timer1 if = '1'
6	Tim1Eq	0	ResSys	R/W	IrqTimer1 on comparison in PWM mode if = '1'
5	Start1Sel[2]	0	ResSys	R/W	Start source selection for timer1
4	Start1Sel[1]	0	ResSys	R/W	
3	Start1Sel[0]	0	ResSys	R/W	
2	Clk1Sel[2]	0	ResSys	R/W	Clock source selection for timer1
1	Clk1Sel[1]	0	ResSys	R/W	
0	Clk1Sel[0]	0	ResSys	R/W	

Start1Sel[2:0]	Start source
000	Software start with bit **SWStart1**
001	External start on PA1
010	External start on PA2
011	External start on PA3
100	External start on PA4
101	External start on PA5
110	External start on PA6
111	External start on PA7

Clk1Sel[2:0]	Clock input
000	PA0
001	PA4
010	Pr2CkSource
011	Pr2Ck[8]
100	Pr2Ck[6]
101	Pr1CkSource
110	Pr1Ck[13]
111	Pr1Ck[11]

RegTim1Status		0x51			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Tim1Status	00	ResSys	R	Timer1 status

RegTim1Load		0x52			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Tim1Load	00	ResSys	R/W	Start value of the timer1

RegTim1Comp		0x53			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Tim1Comp	00	ResSys	R/W	Comparison value of the timer1 in PWM mode

13.4.3 Timer2 configuration

Table 54. Timer2 configuration

RegTim2Cfg		0x54			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	EnPWM2	0	ResSys	R/W	Enable PWM on timer2 if = '1'
6	Tim2Eq	0	ResSys	R/W	IrqTimer2 on comparison in PWM mode if = '1'
5	Start2Sel[2]	0	ResSys	R/W	Start source selection for timer2
4	Start2Sel[1]	0	ResSys	R/W	
3	Start2Sel[0]	0	ResSys	R/W	
2	Clk2Sel[2]	0	ResSys	R/W	Clock source selection for timer2
1	Clk2Sel[1]	0	ResSys	R/W	
0	Clk2Sel[0]	0	ResSys	R/W	

Start2Sel[2:0]	Start source
000	Software start with bit **SWStart2**
001	External start on PA0
010	External start on PA2
011	External start on PA3
100	External start on PA4
101	External start on PA5
110	External start on PA6
111	External start on PA7

Clk2Sel[2:0]	Clock input
000	PA1
001	PA5
010	Pr2CkSource
011	Pr1CkSource
100	Pr1Ck[14]
101	Pr1Ck[12]
110	Pr1Ck[10]
111	Pr1Ck[8]

RegTim2Status		0x55			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Tim2Status	00	ResSys	R	Timer2 status

RegTim2Load		0x56			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Tim2Load	00	ResSys	R/W	Start value of the timer2

RegTim2Comp		0x57			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Tim2Comp	00	ResSys	R/W	Comparison value of the timer2 in PWM mode

13.4.4 Timer3 configuration

Table 55. Timer3 configuration

RegTim3Cfg		0x58			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	EnPWM3	0	ResSys	R/W	Enable PWM on timer3 if = '1'
6	Tim3Eq	0	ResSys	R/W	IrqTimer3 on comparison in PWM mode if = '1'
5	Start3Sel[2]	0	ResSys	R/W	Start source selection for timer3
4	Start3Sel[1]	0	ResSys	R/W	
3	Start3Sel[0]	0	ResSys	R/W	
2	Clk3Sel[2]	0	ResSys	R/W	Clock source selection for timer3
1	Clk3Sel[1]	0	ResSys	R/W	
0	Clk3Sel[0]	0	ResSys	R/W	

Start3Sel[2:0]	Start source
000	Software start with bit **SWStart3**
001	External start on PA0
010	External start on PA1
011	External start on PA3
100	External start on PA4
101	External start on PA5
110	External start on PA6
111	External start on PA7

Clk3Sel[2:0]	Clock input
000	PA2
001	PA6
010	Pr2CkSource
011	Pr2Ck[8]
100	Pr2Ck[4]
101	Pr1CkSource
110	Pr1Ck[13]
111	Pr1Ck[9]

RegTim3Status		0x59			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Tim3Status	00	ResSys	R	Timer3 status

RegTim3Load		0x5A			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Tim3Load	00	ResSys	R/W	Start value of the timer3

RegTim3Comp		0x5B			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Tim3Comp	00	ResSys	R/W	Comparison value of the timers in PWM mode

13.4.5 Timer4 configuration

Table 56. Timer4 configuration

RegTim4Cfg		0x5C			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7	EnPWM4	0	ResSys	R/W	Enable PWM on timer4 if = '1'
6	Tim4Eq	0	ResSys	R/W	IrqTimer4 on comparison in PWM mode if = '1'
5	Start4Sel[2]	0	ResSys	R/W	Start source selection for timer4
4	Start4Sel[1]	0	ResSys	R/W	
3	Start4Sel[0]	0	ResSys	R/W	
2	Clk4Sel[2]	0	ResSys	R/W	Clock source selection for timer4
1	Clk4Sel[1]	0	ResSys	R/W	
0	Clk4Sel[0]	0	ResSys	R/W	

Start4Sel[2:0]	Start source
000	Software start with bit **SWStart4**
001	External start on PA0
010	External start on PA1
011	External start on PA2
100	External start on PA4
101	External start on PA5
110	External start on PA6
111	External start on PA7

Clk4Sel[2:0]	Clock input
000	PA3
001	PA7
010	Pr2ClkSource
011	Pr1ClkSource
100	Pr1Ck[13]
101	Pr1Ck[11]
110	Pr1Ck[9]
111	Pr1Ck[7]

RegTim4Status		0x5D			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Tim4Status	00	ResSys	R	Timer4 status

RegTim4Load		0x5E			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Tim4Load	00	ResSys	R/W	Start value of the timer4

RegTim4Comp		0x5F			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Tim4Comp	00	ResSys	R/W	Comparison value of the timer4 in PWM mode

14 Interruptions

14.1 Basic features

The EM6812 handles 20 independent Interrupt sources grouped into 3 priority levels.

- Highest Priority : Level 0 : Prescaler clocks, PM_miss_skip
- Medium Priority : Level 1 : Port A input changes
- Lowest Priority : Level 2 : Timer compare and full, SPI , Dual port RAM

As such the EM6812 contains

- 9 external Interrupts (Port A, SPI)
- 11 internal Interrupts (Prescaler, Timer, SPI, Dual port RAM, PM_miss_skip)

Interrupt from SPI and Timer may be initialized by either external or internal actions (i.e. timer running on external clock)

Interrupts force a CALL to a fixed interrupts vector, save the program counter (PC) onto the hardware stack and reset the general interrupt flag (GIE). If the CPU was in HALT mode prior to Interrupt then it will come back in active mode. Each priority level has its own interrupt vector.

- Level 1 → Address 1
- Level 2 → Address 2
- Level 0 → Address 3

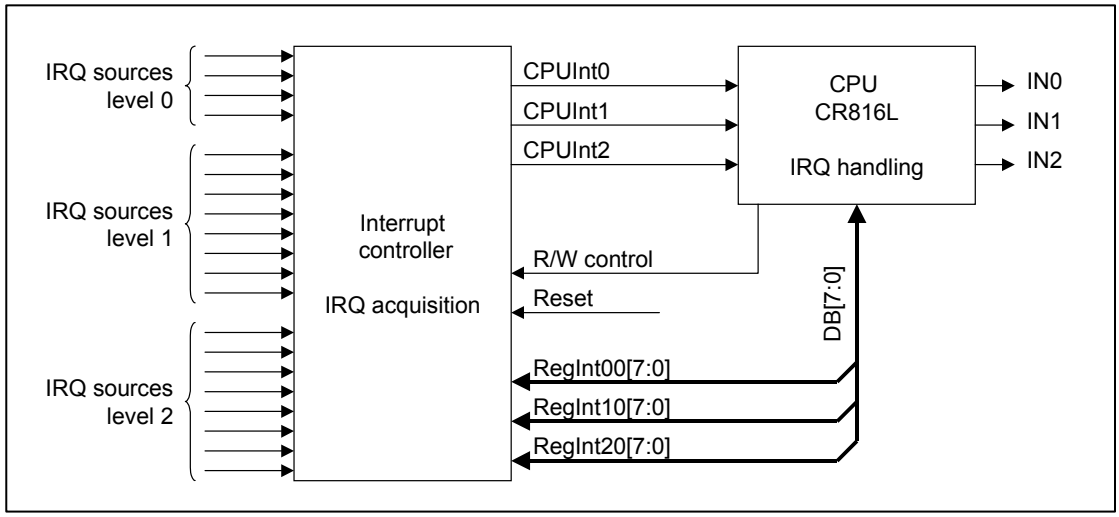
The GIE bit is restored when returning from interrupt with the RETI instruction. The RET instruction does not reinstall the GIE. Nested interrupts are possible by re-enabling the GIE bit within the interrupt routine. Special care needs to be taken when manipulating the GIE bit. See note on
Table 59. Interrupt acquisition registers.

Functions such as interrupt masking, enabling and clearing are available on different levels in the interrupt structure. At power up or after any reset all interrupt inputs are masked and the GIE is cleared.

The Interrupt handling is split into 2 parts.

- One part deals with the acquisition, masking and clearing of the interrupts outside of the CPU.
  → Interrupt acquisition, IRQ Controller
- The 2nd part covers all aspects of priority and interrupts enabling inside the CoolRISC core.
  → CPU interrupts handling

Figure 37, Interrupt top level diagram



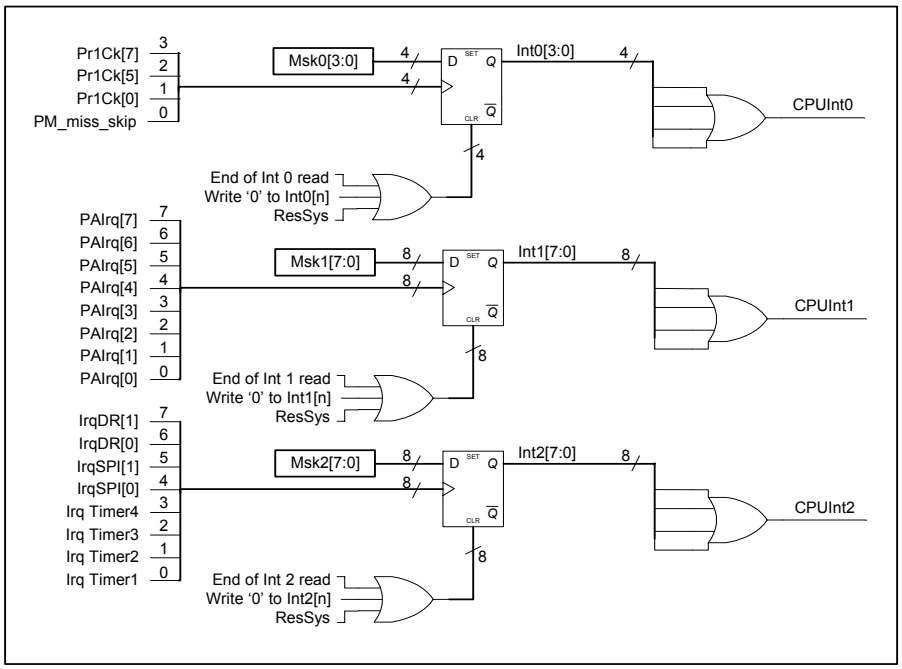
14.2 Interrupt acquisition

A positive edge on any of the unmasked interrupt source signals will set the corresponding interrupt register bit and activate the mapped CPU interrupt input. (I.e. Timer3 interrupt will set bit Int2[2] in register RegInt20 and activate the CPUInt2 interrupt input if mask bit Msk2[2] is '1'). The 3 priority branches for interrupt acquisition are totally independent of each other, masking and selective clear of interrupts on one interrupt vector input does not modify the others. Please refer also to Figure 38, Interrupt acquisition architecture.

Table 57. Interrupts signal sources and destination

Priority mapping	Interrupt source signal from the periphery	Interrupt request register bits	Interrupt mask register bits	Mapped on CPU Interrupt input (OR-function)
0 (high)	Pr1Ck[7]	Int0[3]	Msk0[3]	CPUInt0 *Interrupt source from register RegInt00* *Interrupt masking in register RegMsk00*
	Pr1Ck[5]	Int0[2]	Msk0[2]	
	Pr1Ck[0]	Int0[1]	Msk0[1]	
	PM_miss_skip	Int0[0]	Msk0[0]	
1 (medium)	IrqPA[7:0]	Int1[7:0]	Msk1[7:0]	CPUInt1 *Interrupt source from register RegInt10* *Interrupt masking in register RegMsk10*
2 (low)	IrqDR[1:0]	Int2[7:6]	Msk2[7:6]	CPUInt2 *Interrupt source from register RegInt20* *Interrupt masking in register RegMsk20*
	IrqSPI[1:0]	Int2[5:4]	Msk2[5:4]	
	IrqTimer4	Int2[3]	Msk2[3]	
	IrqTimer3	Int2[2]	Msk2[2]	
	IrqTimer2	Int2[1]	Msk2[1]	
	IrqTimer1	Int2[0]	Msk2[0]	

Figure 38, Interrupt acquisition architecture



14.2.1 Interrupt acquisition masking.

At start up or after any reset all interrupt sources are masked (mask bits are '0'). To activate a specific interrupt source input the corresponding mask bit must be set '1'. Masking does not clear an existing interrupt but will prevent future interrupts on the same input. Refer to Figure 38, Interrupt acquisition architecture.

14.2.2 Interrupt acquisition Clearing

A pending interrupt can be cleared in 3 ways

1. Reading the interrupt flag registers RegInt00, RegInt10 and RegInt20 will automatically clear all stored interrupts which were set prior to the read in the corresponding register. This read is normally done inside the interrupt subroutine to determine the source of the interrupt.
2. Each interrupt register bit can be individually cleared (set '0') by writing '0' to the corresponding RegInt00, RegInt10 and RegInt20 register bit. Write of '1' has no effect.
3. At power up or after any reset all interrupt registers are reset.

14.2.3 Register map, Interrupt acquisition

Table 58. Interrupt acquisition register overview

Functions	Register name	Basic function
Interrupt source inputs	RegInt00	Storage of incoming interrupts, priority 0 (highest)
	RegInt10	Storage of incoming interrupts, priority 1
	RegInt20	Storage of incoming interrupts, priority 2 (lowest)
Interrupt masking	RegMsk00	Masking of incoming interrupts on priority 0 level
	RegMsk10	Masking of incoming interrupts on priority 1 level
	RegMsk20	Masking of incoming interrupts on priority 2 level

Table 59. Interrupt acquisition registers

RegInt00		0x19			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Int0[7:0]	00	ResSys	R*/W*	Interrupt source flag register, priority 0

R*W*: auto-reset after read, only write of value '0' is performed write of '1' has no action

RegInt10		0x18			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Int1[7:0]	00	ResSys	R*/W*	Interrupt source flag register, priority 1

R*W*: auto-reset after read, only write of value '0' is performed write of '1' has no action

RegInt20		0x17			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Int2[7:0]	00	ResSys	R*/W*	Interrupt source flag register, priority 2

R*W*: auto-reset after read, only write of value '0' is performed write of '1' has no action

RegMsk00		0x1C			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Msk0[7:0]	00	ResSys	R/W	Interrupt mask register, priority 0

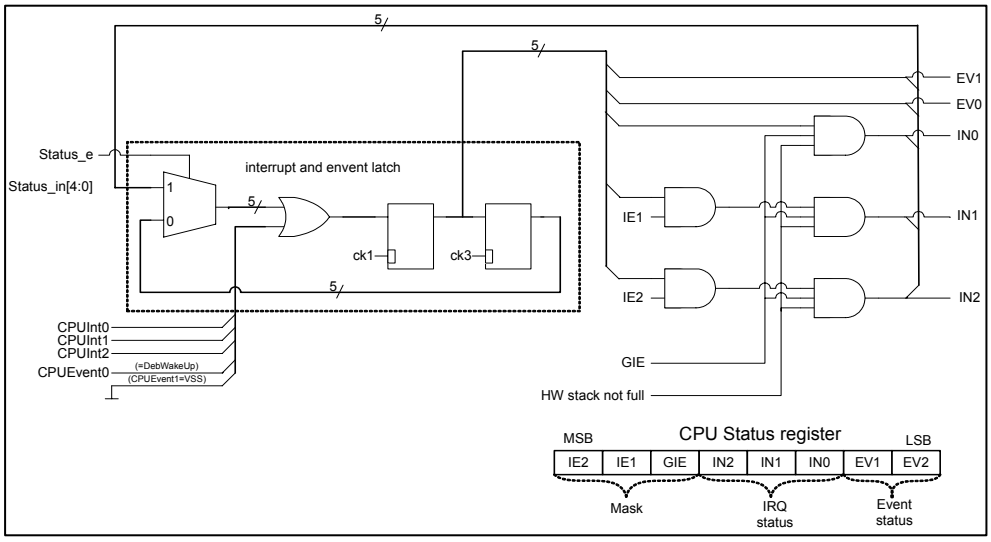
RegMsk10		0x1B			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Msk1[7:0]	00	ResSys	R/W	Interrupt mask register, priority 1

RegMsk20		0x1A			
Bit	**Name**	**Res**	**Reset by**	**R/W**	**Description**
7-0	Msk2[7:0]	00	ResSys	R/W	Interrupt mask register, priority 2

14.3 CPU Interrupt and Event handling

The CPU has three interrupt inputs of different priority. These inputs are directly connected to the peripheral interrupt acquisition block. Each of these inputs has its own interrupt vector. Individual interrupt enabling mechanism is provided for the 2 low priority inputs (IE1, IE2). The GIE acts as a master enable, if GIE is cleared no interrupt can reach the CPU, but may still be stored in the interrupt acquisition block. If the hardware stack of the EM6812 is full, all interrupt inputs are blocked. The number of implemented hardware stack levels is 4. Figure 39, CPU Interrupt architecture and Status register shows the architectural details concerning the interrupt and event latching and its enabling mechanism.

Figure 39, CPU Interrupt architecture and Status register block



An interrupt from the peripheral acquisition block i.e. CPUInt2 is synchronized in the CPU interrupt latch and fed to the CPU interrupt handler signal IN2 if enable bits IE2 and GIE are set and the hardware stack is not full.
Same thing applies to CPUInt1. CPUint0 is maskable only with GIE. As soon as the interrupt is latched, the GIE bit will be automatically cleared to avoid interleaved interrupts. Reading the interrupt acquisition register will clear the pending interrupt and at the end of the interrupt routine the RETI instruction will reinstall the GIE bit.

The CPU will loop in the interrupt routine so long as there is a CPU interrupt input active waiting and the corresponding IE1,IE2 and GIE are set. Refer to 14.2.2 for Interrupt acquisition Clearing.

An interrupt or Event will also clear the CPU Halt mode. The HALT mode disabling remains active as long as one of the EV0, EV1, IN0, IN1, IN2 signals are set.

Before leaving the interrupt service routine one needs to clear the active IRQ aquisition bit (inside RegIntxx) and the corresponding status bit (IN0, IN1, IN2) in the CoolRISC status register. Failure to do so will re-invoke the interrupt service routine just after the preceeding RTI.

Software Interrupts and Events
The above shown CPU Interrupt handling implementation is an extension to the base structure and as such allows software interrupts and software events to be written directly in the interrupt and event latches (write '1' to CPU status register bit 0 to 4, signals status_e and status_in). Software written interrupts and events remain stored in the interrupt latch until they get cleared again (write '0' to status register bit 0 to bit 4).
The CPUEvent1 input is not used on the EM6812 and therefore reads always '0' unless set by software.
The CPUEvent0 is connected to the Port A wake-up function (signal DebWakeUp), this event shows on EV0.

14.3.1 Interrupt priority

Interrupt priority is used only to select which interrupt will be processed when multiple interrupt requests occur simultaneously. In such case the higher priority interrupt is handled first. At the end of the interrupt routine RETI the processor will immediately go back into the interrupt routine to handle the next interrupt of highest priority.
If a high priority interrupt occurs while the CPU is treating a low priority interrupt, the pending interrupt must wait until the GIE is enabled, usually by the RETI instruction.

14.3.2 CPU Status register

The status register, used to control the interrupts and events, is an internal register to the CoolRISC CPU. It therefore does not figure in the peripheral memory mapping. All CPU enable bits for the interrupts and the current status of the events and the interrupts are part of this register.

Table 60. CPU status register description

Bit	Name	Reset	Reset by	R/W	Description
7	IE2	0	ResCPU	R/W	Level 2 Interrupt enable '1' = enabled, '0' = disabled
6	IE1	0	ResCPU	R/W	Level 1 Interrupt enable '1' = enabled, '0' = disabled
5	GIE	0	ResCPU	R/W*	General interrupt enable '1' = enabled, '0' = disabled
4	IN2	0	ResCPU	R/W	Interrupt request level 2 flag, shows CPUInt2 '1' = IRQ pending, '0' = no IRQ The IRQ may only take place if IN2, IE2, and GIE are set
3	IN1	0	ResCPU	R/W	Interrupt request level 1 flag, shows CPUInt1 '1' = IRQ pending, '0' = no IRQ The IRQ may only take place if IN1, IE1, and GIE are set
2	IN0	0	ResCPU	R/W	Interrupt request level 0 flag, shows CPUInt1 '1' = IRQ pending, '0' = no IRQ The IRQ may only take place if IN0 and GIE are set
1	EV1	0	ResCPU	R/W	Event request 1, input connected to V_{SS}
0	EV0	0	ResCPU	R/W	Event request 0, input connected to DebWakeUp

**Clear General Interrupt Enable bit GIE. Special care must be taken clearing the GIE bit. If an interrupt arrives during the clear operation the software may still branch into the interrupt routine and will set the GIE bit by the interrupt routine ending RETI instruction. This behavior may prevent from creating 'interrupt protected' areas within your code. A suitable workaround is to check if the GIE clearing took effect (Instruction) TSTB before executing the protected section.*

14.3.3 CPU Status register pipeline exception

Another consequence of the above interrupt implementation is that several instruction sequences work in a different way than expected. These instructions are mostly related to interrupt and event signals. For 'normal' instructions the pipeline is completely transparent.
If an interrupt is set by software (i.e. write into the status register with a MOVE stat) the pipeline causes the next instruction to be executed before the processor jumps to the interrupt subroutine. This allows one to supply a parameter to a 'trap' as in Code shown below.

```
SETB    stat,    #4            ; trap
MOVE    a        #parameter    ;
```

If an event bit is set by software (i.e. write into the CPU status register with a MOVE stat) and if a JEV (jump on event) instruction immediately follows the move, the jump on event will act as if the move has not been executed, since the write into the CPU status register will occur only once the JEV has been executed. The move takes 3 cycles to be executed and the JEV only one.

14.3.4 Processor vector table

Address 1,2 and 3 of the program memory are reserved for interrupt subroutine calls. Generally the first four addresses of the program memory are reserved for the processor vector table. The address 0 of the program memory contains the jump to the start-up routine

Table 61. Processor vector table

Address	Accessed by	Description	Priority
0	ResCPU	Any reset, start-up address	Maximal, above interrupts
1	IN1	Interrupt level 1	medium
2	IN2	Interrupt level 2	low
3	IN3	Interrupt level 0	high

14.3.5 Context Saving

Since an interrupt may occur any time during normal program execution, there is no way to know which processor registers are used by the user program. For this reason, all resources modified in the interrupt service routine have to be saved upon entering and restored when leaving the service routine. The flags(C,V) and the accumulator (a) must always be saved, since most instructions will modify them. Other registers need only to be saved when they are modified in the interrupt service routine. There is a particular way to follow when saving resources. The accumulator should be saved first, followed by the flags and then the other registers

15 Memory mapping

Page 1/3 of mapping

	Addr	Addr	Init	bit 7			bit 6			bit 5			bit 4			bit 3			bit 2			bit 1			bit 0		
	Hex	Dec		Name	R/W	Res	Name	R/W	Res	Name	R/W	Res	Name	R/W	Res	Name	R/W	Res	Name	R/W	Res	Name	R/W	Res	Name	R/W	Res
General purpose registers																											
LPRam0	0	0	00	LPR0[7]	R/W	POR	LPR0[6]	R/W	POR	LPR0[5]	R/W	POR	LPR0[4]	R/W	POR	LPR0[3]	R/W	POR	LPR0[2]	R/W	POR	LPR0[1]	R/W	POR	LPR0[0]	R/W	POR
...	...	...	00	...	R/W	POR	...	R/W	POR	...	R/W	POR	...	R/W	POR	...	R/W	POR	...	R/W	POR	...	R/W	POR	...	R/W	POR
LPRAM11	B	11	00	LPR11[7]	R/W	POR	LPR11[6]	R/W	POR	LPR11[5]	R/W	POR	LPR11[4]	R/W	POR	LPR11[3]	R/W	POR	LPR11[2]	R/W	POR	LPR11[1]	R/W	POR	LPR11[0]	R/W	POR
DPRAM0	C	12	00	DPR0[7]	R/W	POR	DPR0[6]	R/W	POR	DPR0[5]	R/W	POR	DPR0[4]	R/W	POR	DPR0[3]	R/W	POR	DPR0[2]	R/W	POR	DPR0[1]	R/W	POR	DPR0[0]	R/W	POR
DPRAM1	D	13	00	DPR1[7]	R/W	POR	DPR1[6]	R/W	POR	DPR1[5]	R/W	POR	DPR1[4]	R/W	POR	DPR1[3]	R/W	POR	DPR1[2]	R/W	POR	DPR1[1]	R/W	POR	DPR1[0]	R/W	POR
DPRAM2	E	14	00	DPR2[7]	R/W	POR	DPR2[6]	R/W	POR	DPR2[5]	R/W	POR	DPR2[4]	R/W	POR	DPR2[3]	R/W	POR	DPR2[2]	R/W	POR	DPR2[1]	R/W	POR	DPR2[0]	R/W	POR
DPRAM3	F	15	00	DPR3[7]	R/W	POR	DPR3[6]	R/W	POR	DPR3[5]	R/W	POR	DPR3[4]	R/W	POR	DPR3[3]	R/W	POR	DPR3[2]	R/W	POR	DPR3[1]	R/W	POR	DPR3[0]	R/W	POR
System Registers (Reset and Clock selections)																											
RegSys1	10	16	01	Sleep	RW	Sys	DisResetPad	RW	POR	DisResInp	RW	POR	FlagXtal	R	Main	OPTCldStart[1]	RW	POR	OPTCldStart[0]	RW	POR	FreqRange	RW	Sys	EnRC	RW	Sys
RegSys2	11	17	03	EnXtal	RW	Main	SelExtHFck	RW	Main	SelHFckSource	RW	Main	SelExtLFck	RW	Main	--	R	--	Sel32k	RW	Main	RCDiv[1]	RW	Main	RCDiv[0]	RW	Main
RegResStat	12	18	00	ResetPadFlag	RC	POR	ResetWDFlag	RC	POR	ResInpPAFlag	RC	POR	ResBwnOutFlag	RC	POR	SleepEn	RW	POR	EnDebResPad	RW	Slp	CkDebResPad	RW	Sys	DatOscOut	R	--
RegTrimRC	13	19	7F	Trim[7]	RW	Main	Trim[6]	RW	Main	Trim[5]	RW	Main	Trim[4]	RW	Main	Trim[3]	RW	Main	Trim[2]	RW	Main	Trim[1]	RW	Main	Trim[0]	RW	Main
RegPrCkSel	14	20	1E	Pr1CkSel[2]	RW	Sys	Pr1CkSel[1]	RW	Sys	Pr1CkSel[0]	RW	Sys	AutoSel	RW	Sys	Pr2CkSel	RW	Sys	--	R	--	--	R	--	--	R	--
RegPr1Status	15	21	01	Pr1CkStatus[7]	RC	Sys	Pr1CkStatus[6]	RC	Sys	Pr1CkStatus[5]	RC	Sys	Pr1CkStatus[4]	RC	Sys	Pr1CkStatus[3]	RC	Sys	Pr1CkStatus[2]	RC	Sys	Pr1CkStatus[1]	RC	Sys	Pr1CkStatus[0]	RC	Sys
RegPr2Status	16	22	00	Pr2CkStatus[7]	RC	Sys	Pr2CkStatus[6]	RC	Sys	Pr2CkStatus[5]	RC	Sys	Pr2CkStatus[4]	RC	Sys	Pr2CkStatus[3]	RC	Sys	Pr2CkStatus[2]	RC	Sys	Pr2CkStatus[1]	RC	Sys	Pr2CkStatus[0]	RC	Sys
Interupt Control																											
RegInt20	17	23	00	Int2[7]	RA	Sys	Int2[6]	RA	Sys	Int2[5]	RA	Sys	Int2[4]	RA	Sys	Int2[3]	RA	Sys	Int2[2]	RA	Sys	Int2[1]	RA	Sys	Int2[0]	RA	Sys
RegInt10	18	24	00	Int1[7]	RA	Sys	Int1[6]	RA	Sys	Int1[5]	RA	Sys	Int1[4]	RA	Sys	Int1[3]	RA	Sys	Int1[2]	RA	Sys	Int1[1]	RA	Sys	Int1[0]	RA	Sys
RegInt00	19	25	00	Int0[7]	RA	Sys	Int0[6]	RA	Sys	Int0[5]	RA	Sys	Int0[4]	RA	Sys	Int0[3]	RA	Sys	Int0[2]	RA	Sys	Int0[1]	RA	Sys	Int0[0]	RA	Sys
RegMsk20	1A	26	00	Msk2[7]	RW	Sys	Msk2[6]	RW	Sys	Msk2[5]	RW	Sys	Msk2[4]	RW	Sys	Msk2[3]	RW	Sys	Msk2[2]	RW	Sys	Msk2[1]	RW	Sys	Msk2[0]	RW	Sys
RegMsk10	1B	27	00	Msk1[7]	RW	Sys	Msk1[6]	RW	Sys	Msk1[5]	RW	Sys	Msk1[4]	RW	Sys	Msk1[3]	RW	Sys	Msk1[2]	RW	Sys	Msk1[1]	RW	Sys	Msk1[0]	RW	Sys
RegMsk00	1C	28	00	Msk0[7]	RW	Sys	Msk0[6]	RW	Sys	Msk0[5]	RW	Sys	Msk0[4]	RW	Sys	Msk0[3]	RW	Sys	Msk0[2]	RW	Sys	Msk0[1]	RW	Sys	Msk0[0]	RW	Sys
Analog Control																											
RegAnaCfg	20	32	40	EnSVLD	RW	Sys	EnBrownOut	RW	POR	SVLDLevel1	RW	Sys	SVLDLevel2	RW	Sys	SVLDLevel3	RW	Sys	--	R	--	--	R	--	SVLDStatus	R	--

Read and write acronyms

RW	Read and write access possible
R	Read access only
RC	Read access, clear by writing '0'
RA	Read access, clear automaticaly after reading or by writing '0'

Reset source

POR	Power on reset
Main	ResMain
Sys	ResSysSlp
Slp	ResSysSlp

Page 2/3 of mapping

	Addr Hex	Addr Dec	Init	bit 7			bit 6			bit 5			bit 4			bit 3			bit 2			bit 1			bit 0		
				Name	R/W	Res	Name	R/W	Res	Name	R/W	Res	Name	R/W	Res	Name	R/W	Res	Name	R/W	Res	Name	R/W	Res	Name	R/W	Res
Port A Settings																											
RegInPA	21	33	--	PAIn[7]	R	--	PAIn[6]	R	--	PAIn[5]	R	--	PAIn[4]	R	--	PAIn[3]	R	--	PAIn[2]	R	--	PAIn[1]	R	--	PAIn[0]	R	--
RegOutPA	22	34	00	PAOut[7]	RW	Sys	PAOut[6]	RW	Sys	PAOut[5]	RW	Sys	PAOut[4]	RW	Sys	PAOut[3]	RW	Sys	PAOut[2]	RW	Sys	PAOut[1]	RW	Sys	PAOut[0]	RW	Sys
RegCfgPA	23	35	00	Excl_nComb	RW	Main	Wk_nRes	RW	Main	SplitCmb	RW	Main	EnDebResInp	RW	Slp	CkDebResInp	RW	Main	EnDebWk	RW	Slp	CkDebWk	RW	Main	RCLoop	RW	Sys
RegIOSelPA	24	36	00	IOSelPA[7]	RW	Sys	IOSelPA[6]	RW	Sys	IOSelPA[5]	RW	Sys	IOSelPA[4]	RW	Sys	IOSelPA[3]	RW	Sys	IOSelPA[2]	RW	Sys	IOSelPA[1]	RW	Sys	IOSelPA[0]	RW	Sys
RegCmbKey	25	37	00	CmbKey[7]	RW	Main	CmbKey[6]	RW	Main	CmbKey[5]	RW	Main	CmbKey[4]	RW	Main	CmbKey[3]	RW	Main	CmbKey[2]	RW	Main	CmbKey[1]	RW	Main	CmbKey[0]	RW	Main
RegMskRstWkUp	26	38	00	MskRstWkUp[7]	RW	Main	MskRstWkUp[6]	RW	Main	MskRstWkUp[5]	RW	Main	MskRstWkUp[4]	RW	Main	MskRstWkUp[3]	RW	Main	MskRstWkUp[2]	RW	Main	MskRstWkUp[1]	RW	Main	MskRstWkUp[0]	RW	Main
RegIntEdgPA	27	39	00	IntEdgPA[7]	RW	Main	IntEdgPA[6]	RW	Main	IntEdgPA[5]	RW	Main	IntEdgPA[4]	RW	Main	IntEdgPA[3]	RW	Main	IntEdgPA[2]	RW	Main	IntEdgPA[1]	RW	Main	IntEdgPA[0]	RW	Main
RegEnDebPA	28	40	00	EnDebPA[7]	RW	Slp	EnDebPA[6]	RW	Slp	EnDebPA[5]	RW	Slp	EnDebPA[4]	RW	Slp	EnDebPA[3]	RW	Slp	EnDebPA[2]	RW	Slp	EnDebPA[1]	RW	Slp	EnDebPA[0]	RW	Slp
RegOpenDrainPA	29	41	00	OpenDrainPA[7]	RW	Main	OpenDrainPA[6]	RW	Main	OpenDrainPA[5]	RW	Main	OpenDrainPA[4]	RW	Main	OpenDrainPA[3]	RW	Main	OpenDrainPA[2]	RW	Main	OpenDrainPA[1]	RW	Main	OpenDrainPA[0]	RW	Main
RegPullUpPA	2A	42	00	PullUpPA[7]	RW	Main	PullUpPA[6]	RW	Main	PullUpPA[5]	RW	Main	PullUpPA[4]	RW	Main	PullUpPA[3]	RW	Main	PullUpPA[2]	RW	Main	PullUpPA[1]	RW	Main	PullUpPA[0]	RW	Main
RegPullDownPA	2B	43	FF	PullDownPA[7]	RW	Main	PullDownPA[6]	RW	Main	PullDownPA[5]	RW	Main	PullDownPA[4]	RW	Main	PullDownPA[3]	RW	Main	PullDownPA[2]	RW	Main	PullDownPA[1]	RW	Main	PullDownPA[0]	RW	Main
Port B settings																											
RegInPB	30	48	--	PBIn[7]	R	--	PBIn[6]	R	--	PBIn[5]	R	--	PBIn[4]	R	--	PBIn[3]	R	--	PBIn[2]	R	--	PBIn[1]	R	--	PBIn[0]	R	--
RegOutPB	31	49	00	OutPB[7]	RW	Sys	OutPB[6]	RW	Sys	OutPB[5]	RW	Sys	OutPB[4]	RW	Sys	OutPB[3]	RW	Sys	OutPB[2]	RW	Sys	OutPB[1]	RW	Sys	OutPB[0]	RW	Sys
RegCfgPB	32	50	00	EnDualRAM	RW	Sys	EnSPI	RW	Sys	EnSig1	RW	Sys	EnSig2	RW	Sys	EnSig3	RW	Sys	EnSig4	RW	Sys	--	R	--	--	R	--
RegSigSel	33	51	00	Sig1Sel[1]	RW	Sys	Sig1Sel[0]	RW	Sys	Sig2Sel[1]	RW	Sys	Sig2Sel[0]	RW	Sys	Sig3Sel[1]	RW	Sys	Sig3Sel[0]	RW	Sys	Sig4Sel[1]	RW	Sys	Sig4Sel[0]	RW	Sys
RegIOSelPB	34	52	00	IOSelPB[7]	RW	Sys	IOSelPB[6]	RW	Sys	IOSelPB[5]	RW	Sys	IOSelPB[4]	RW	Sys	IOSelPB[3]	RW	Sys	IOSelPB[2]	RW	Sys	IOSelPB[1]	RW	Sys	IOSelPB[0]	RW	Sys
RegOpenDrainPB	35	53	00	OpenDrainPB[7]	RW	Main	OpenDrainPB[6]	RW	Main	OpenDrainPB[5]	RW	Main	OpenDrainPB[4]	RW	Main	OpenDrainPB[3]	RW	Main	OpenDrainPB[2]	RW	Main	OpenDrainPB[1]	RW	Main	OpenDrainPB[0]	RW	Main
RegPullUpPB	36	54	00	PullUpPB[7]	RW	Main	PullUpPB[6]	RW	Main	PullUpPB[5]	RW	Main	PullUpPB[4]	RW	Main	PullUpPB[3]	RW	Main	PullUpPB[2]	RW	Main	PullUpPB[1]	RW	Main	PullUpPB[0]	RW	Main
RegPullDownPB	37	55	FF	PullDownPB[7]	RW	Main	PullDownPB[6]	RW	Main	PullDownPB[5]	RW	Main	PullDownPB[4]	RW	Main	PullDownPB[3]	RW	Main	PullDownPB[2]	RW	Main	PullDownPB[1]	RW	Main	PullDownPB[0]	RW	Main
Serial interface																											
RegSPICfg	38	56	01	Start	RW	Slp	MS2	RW	Sys	MS1	RW	Sys	MS0	RW	Sys	PosnNegShft	RW	Sys	MSBnLSB	RW	Sys	Synchro	RW	Sys	Load_nShift	R	--
RegSPIDat	39	57	00	SPIDat[7]	R	Sys	SPIDat[6]	R	Sys	SPIDat[5]	R	Sys	SPIDat[4]	R	Sys	SPIDat[3]	R	Sys	SPIDat[2]	R	Sys	SPIDat[1]	R	Sys	SPIDat[0]	R	Sys
RegSPILoad	3A	58	00	SPILoad[7]	RW	Sys	SPILoad[6]	RW	Sys	SPILoad[5]	RW	Sys	SPILoad[4]	RW	Sys	SPILoad[3]	RW	Sys	SPILoad[2]	RW	Sys	SPILoad[1]	RW	Sys	SPILoad[0]	RW	Sys
Logic Watchdog																											
RegWDSys	3B	59	40	WDClear	RW	--	EnWD	RW	Sys	WDVal[1]	R	Sys	WDVal[0]	R	Sys	WDClkSel	RW	Sys	WDKeyLock[2]	RW	Sys	WDKeyLock[1]	RW	Sys	WDKeyLock[0]	RW	Sys
RegWDKey	3C	60	00	WDKey[7]	RW	Sys	WDKey[6]	RW	Sys	WDKey[5]	RW	Sys	WDKey[4]	RW	Sys	WDKey[3]	RW	Sys	WDKey[2]	RW	Sys	WDKey[1]	RW	Sys	WDKey[0]	RW	Sys

Read and write acronyms	**RW**	Read and write access possible	Reset source	**POR**	Power on reset
	R	Read access only		**Main**	ResMain
	RC	Read access, clear by writing '0'		**Sys**	ResSysSlp
	RA	Read access, clear automaticaly after reading or by writing '0'		**Slp**	ResSysSlp

Page 3/3 of mapping

	Addr	Addr	Init	bit 7			bit 6			bit 5			bit 4			bit 3			bit 2			bit 1			bit 0		
	Hex	Dec		Name	R/W	Res	Name	R/W	Res	Name	R/W	Res	Name	R/W	Res	Name	R/W	Res	Name	R/W	Res	Name	R/W	Res	Name	R/W	Res
Timer settings																											
RegTimersCfg	4E	78	00	EnTim12	RW	Sys	EnTim34	RW	Sys	AR1	RW	Sys	AR2	RW	Sys	AR3	RW	Sys	AR4	RW	Sys	--	--	--	--	--	--
RegTimersStart	4F	79	00	SWStart1	RW	Sys	Tim1Pulse	RW	Sys	SWStart2	RW	Sys	Tim2Pulse	RW	Sys	SWStart3	RW	Sys	Tim3Pulse	RW	Sys	SWStart4	RW	Sys	Tim4Pulse	RW	Sys
RegTim1Cfg	50	80	00	EnPWM1	RW	Sys	Tim1Eq	RW	Sys	Start1Sel[2]	RW	Sys	Start1Sel[1]	RW	Sys	Start1Sel[0]	RW	Sys	Clk1Sel[2]	RW	Sys	Clk1Sel[1]	RW	Sys	Clk1Sel[0]	RW	Sys
RegTim1Status	51	81	00	Tim1Status[7]	R	Sys	Tim1Status[6]	R	Sys	Tim1Status[5]	R	Sys	Tim1Status[4]	R	Sys	Tim1Status[3]	R	Sys	Tim1Status[2]	R	Sys	Tim1Status[1]	R	Sys	Tim1Status[0]	R	Sys
RegTim1Load	52	82	00	Tim1Load[7]	RW	Sys	Tim1Load[6]	RW	Sys	Tim1Load[5]	RW	Sys	Tim1Load[4]	RW	Sys	Tim1Load[3]	RW	Sys	Tim1Load[2]	RW	Sys	Tim1Load[1]	RW	Sys	Tim1Load[0]	RW	Sys
RegTim1Comp	53	83	00	Tim1Comp[7]	RW	Sys	Tim1Comp[6]	RW	Sys	Tim1Comp[5]	RW	Sys	Tim1Comp[4]	RW	Sys	Tim1Comp[3]	RW	Sys	Tim1Comp[2]	RW	Sys	Tim1Comp[1]	RW	Sys	Tim1Comp[0]	RW	Sys
RegTim2Cfg	54	84	00	EnPWM2	RW	Sys	Tim2Eq	RW	Sys	Start2Sel[2]	RW	Sys	Start2Sel[1]	RW	Sys	Start2Sel[0]	RW	Sys	Clk2Sel[2]	RW	Sys	Clk2Sel[1]	RW	Sys	Clk2Sel[0]	RW	Sys
RegTim2Status	55	85	00	Tim2Status[7]	R	Sys	Tim2Status[6]	R	Sys	Tim2Status[5]	R	Sys	Tim2Status[4]	R	Sys	Tim2Status[3]	R	Sys	Tim2Status[2]	R	Sys	Tim2Status[1]	R	Sys	Tim2Status[0]	R	Sys
RegTim2Load	56	86	00	Tim2Load[7]	RW	Sys	Tim2Load[6]	RW	Sys	Tim2Load[5]	RW	Sys	Tim2Load[4]	RW	Sys	Tim2Load[3]	RW	Sys	Tim2Load[2]	RW	Sys	Tim2Load[1]	RW	Sys	Tim2Load[0]	RW	Sys
RegTim2Comp	57	87	00	Tim2Comp[7]	RW	Sys	Tim2Comp[6]	RW	Sys	Tim2Comp[5]	RW	Sys	Tim2Comp[4]	RW	Sys	Tim2Comp[3]	RW	Sys	Tim2Comp[2]	RW	Sys	Tim2Comp[1]	RW	Sys	Tim2Comp[0]	RW	Sys
RegTim3Cfg	58	88	00	EnPWM3	RW	Sys	Tim3Eq	RW	Sys	Start3Sel[2]	RW	Sys	Start3Sel[1]	RW	Sys	Start3Sel[0]	RW	Sys	Clk3Sel[2]	RW	Sys	Clk3Sel[1]	RW	Sys	Clk3Sel[0]	RW	Sys
RegTim3Status	59	89	00	Tim3Status[7]	R	Sys	Tim3Status[6]	R	Sys	Tim3Status[5]	R	Sys	Tim3Status[4]	R	Sys	Tim3Status[3]	R	Sys	Tim3Status[2]	R	Sys	Tim3Status[1]	R	Sys	Tim3Status[0]	R	Sys
RegTim3Load	5A	90	00	Tim3Load[7]	RW	Sys	Tim3Load[6]	RW	Sys	Tim3Load[5]	RW	Sys	Tim3Load[4]	RW	Sys	Tim3Load[3]	RW	Sys	Tim3Load[2]	RW	Sys	Tim3Load[1]	RW	Sys	Tim3Load[0]	RW	Sys
RegTim3Comp	5B	91	00	Tim3Comp[7]	RW	Sys	Tim3Comp[6]	RW	Sys	Tim3Comp[5]	RW	Sys	Tim3Comp[4]	RW	Sys	Tim3Comp[3]	RW	Sys	Tim3Comp[2]	RW	Sys	Tim3Comp[1]	RW	Sys	Tim3Comp[0]	RW	Sys
RegTim4Cfg	5C	92	00	EnPWM4	RW	Sys	Tim4Eq	RW	Sys	Start4Sel[2]	RW	Sys	Start4Sel[1]	RW	Sys	Start4Sel[0]	RW	Sys	Clk4Sel[2]	RW	Sys	Clk4Sel[1]	RW	Sys	Clk4Sel[0]	RW	Sys
RegTim4Status	5D	93	00	Tim4Status[7]	R	Sys	Tim4Status[6]	R	Sys	Tim4Status[5]	R	Sys	Tim4Status[4]	R	Sys	Tim4Status[3]	R	Sys	Tim4Status[2]	R	Sys	Tim4Status[1]	R	Sys	Tim4Status[0]	R	Sys
RegTim4Load	5E	94	00	Tim4Load[7]	RW	Sys	Tim4Load[6]	RW	Sys	Tim4Load[5]	RW	Sys	Tim4Load[4]	RW	Sys	Tim4Load[3]	RW	Sys	Tim4Load[2]	RW	Sys	Tim4Load[1]	RW	Sys	Tim4Load[0]	RW	Sys
RegTim4Comp	5F	95	00	Tim4Comp[7]	RW	Sys	Tim4Comp[6]	RW	Sys	Tim4Comp[5]	RW	Sys	Tim4Comp[4]	RW	Sys	Tim4Comp[3]	RW	Sys	Tim4Comp[2]	RW	Sys	Tim4Comp[1]	RW	Sys	Tim4Comp[0]	RW	Sys

Read and write acronyms

RW	Read and write access possible
R	Read access only
RC	Read access, clear by writing '0'
RA	Read access, clear automaticaly after reading or by writing '0'

Reset source

POR	Power on reset
Main	ResMain
Sys	ResSysSlp
Slp	ResSysSlp

16 Typical V and T dependencies

16.1 IVDD Currents

Figure 40. RC_10MHz, Active and Standby mode

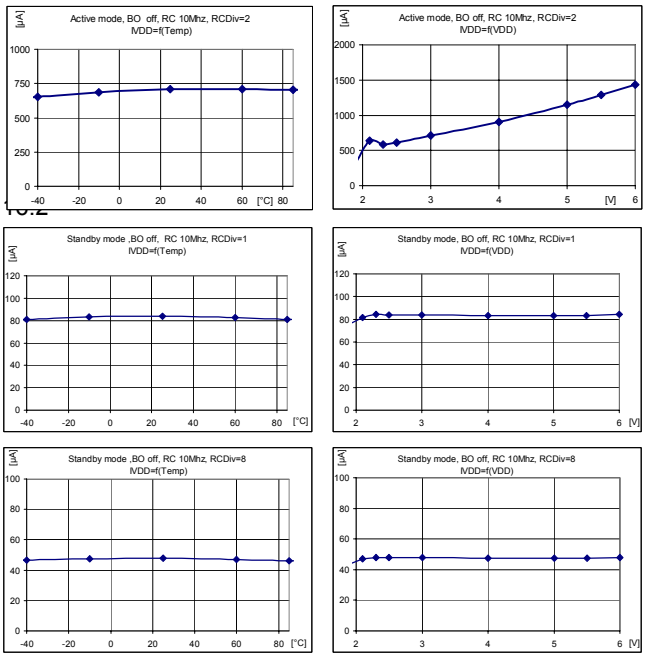


Figure 41. RC_1MHz, Active and Standby mode

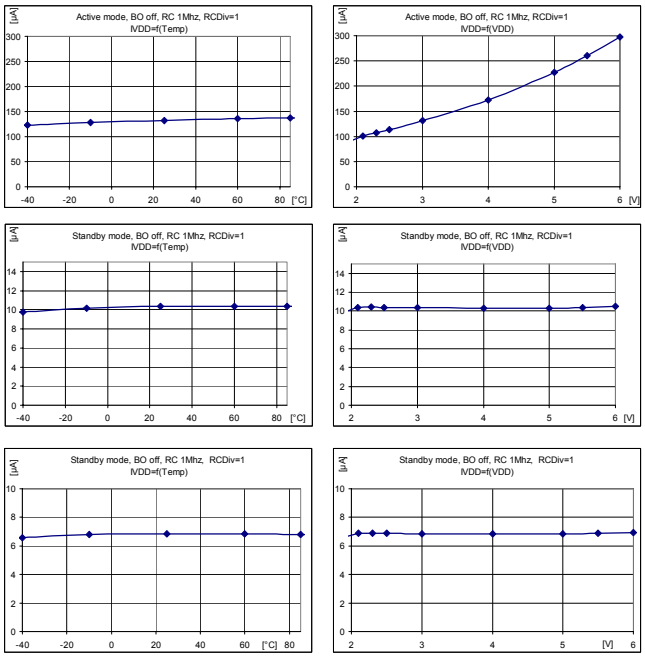


Figure 43. Xtal 32kHz Active and Standby mode

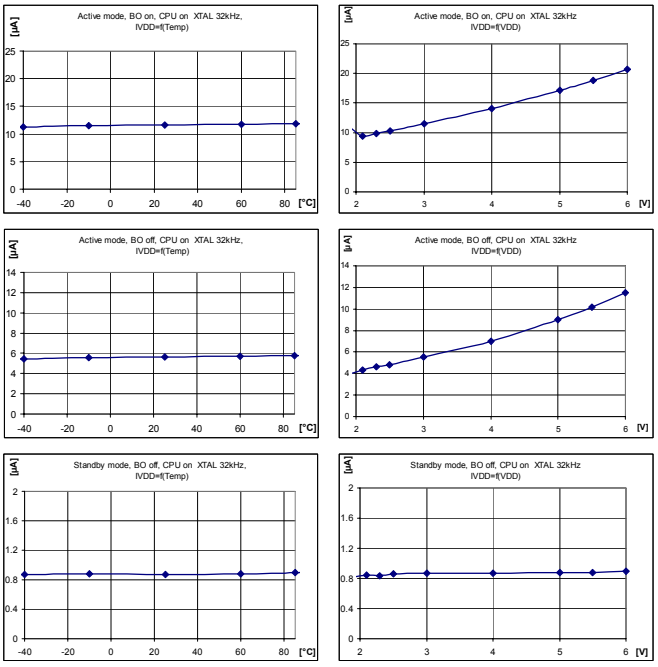
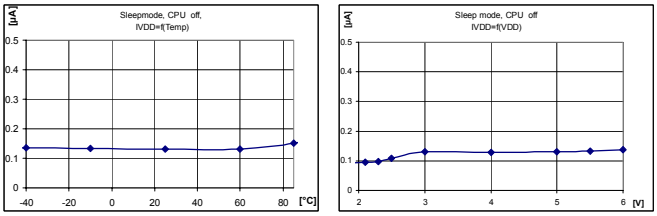
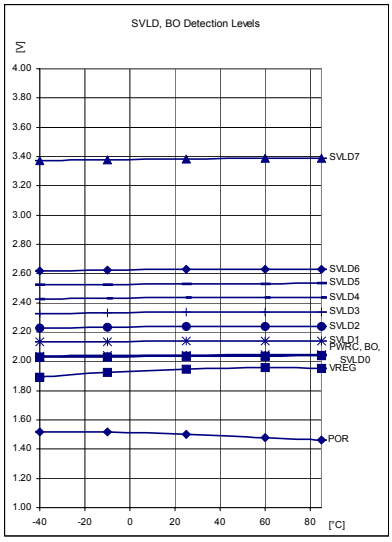


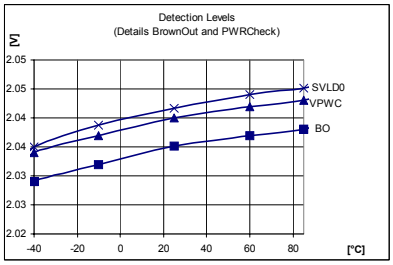
Figure 42. Sleep mode



16.3 SVLD, BO Detection levels

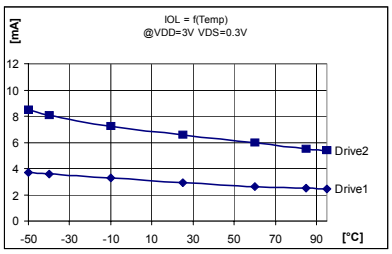
Figure 44. BO and SVLD detection levels

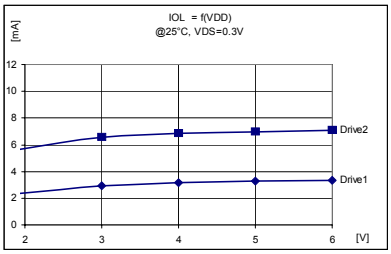


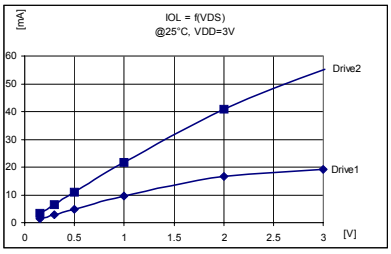


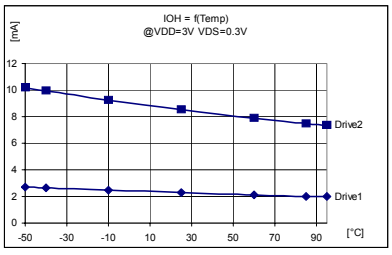
16.4 IOL and IOH drives

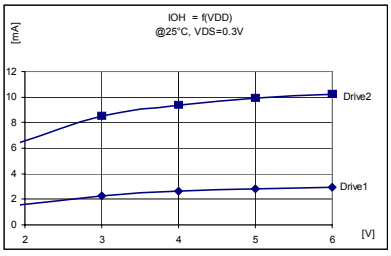
Figure 45. Output Current drives

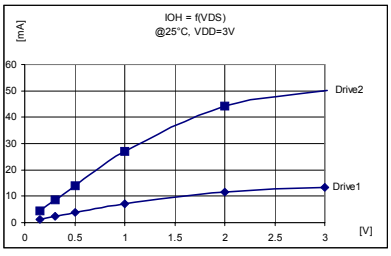






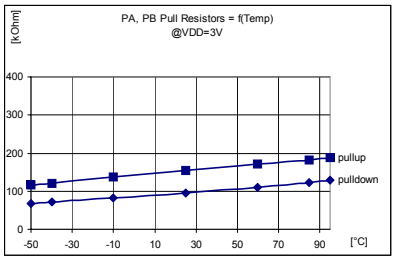


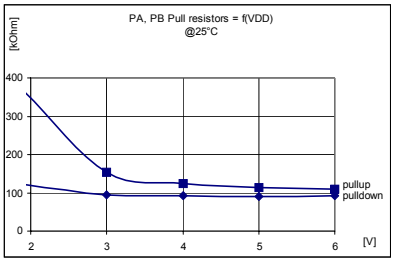


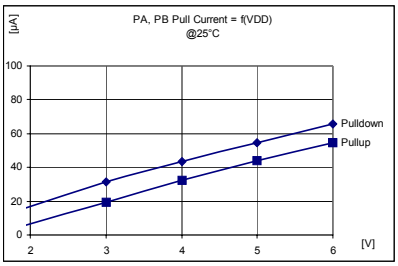


16.5 Pullup and Pulldown

Figure 46. Pullup and pulldown resitances and current







17 Electrical Specification

17.1 Absolute Maximum Ratings

	Min.	Max.	Units
Power supply V_{DD}-V_{SS}	- 0.2	+6.0	V
Input voltage	V_{SS} – 0.2	V_{DD}+0.2	V
Storage temperature	- 40	+ 125	°C
Electrostatic discharge to Mil-Std-883C method 3015.7 with ref. to V_{SS}	-2000	+2000	V
Maximum soldering conditions Packages are Green-Mold and Lead-free	As per Jedec J-STD-020C		

Stresses above these listed maximum ratings may cause permanent damage to the device.
Exposure beyond specified electrical characteristics may affect device reliability or cause malfunction

17.2 Handling Procedures

This device has built-in protection against high static voltages or electric fields; however, anti-static precautions should be taken as for any other CMOS integrated circuit.
Unless otherwise specified, proper operation can only occur when all terminal voltages are kept within the supply voltage range.

17.3 Standard Operating Conditions

Parameter	MIN	TYP	MAX	Unit	Description
Temperature	-40	25	85	°C	
V_{DD} _Range	2	3.0	5.5	V	
I_{VSS} max			80	mA	Maximum current out of V_{SS} Pin
I_{VDD} max			80	mA	Maximum current into V_{DD} Pin
V_{SS}		0		V	Reference terminal
C_{VREG} **(note 1)**	470			nF	regulated voltage capacitor
Flash data retention	10 yrs				Read and Erase state retention

Note 1: This capacitor filters switching noise from V_{DD} to keep it away from the internal logic cells.
In noisy systems the capacitor should be chosen bigger than minimum value.

17.4 Typical Crystal specification

Fq		32768		Hz	nominal frequency
Rqs		35		KOhm	typical quartz serial resistance
CL		8.2		pF	typical quartz load capacitance
df/f		± 30		ppm	quartz frequency tolerance

Watch type crystal oscillator (i.e Microcrystal DS15), connected between QIN and Qout terminal.

17.5 DC Characteristics - Power Supply Currents

Parameter	Conditions	Symbol	Min.	Typ.	Max.	Unit
ACTIVE Supply Current CPU on RC=10MHz	V_{DD} =3V, 25°C, RCDiv=2	I_{VDDa10}		800	1200	µA
	V_{DD} =3V, -40 to 85°C, RCDiv=2	I_{VDDa10}		800	1400	µA
Standby Supply Current RC=10MHz enabled	V_{DD} =3V, 25°C, RCDiv=1	I_{VDDh10}		80	110	µA
	V_{DD} =3V, -40 to 85°C, RCDiv=1	I_{VDDh10}		80	110	µA
Active Supply Current CPU on RC=1MHz	V_{DD} =3V, 25°C, RCDiv=1	I_{VDDa1}		120	140	µA
	V_{DD} =3V, -40 to 85°C, RCDiv=1	I_{VDDa1}		120	140	µA
	V_{DD} =3V, 25°C; RCDiv=8	I_{VDDa1}		45		µA
Standby Supply Current RC=1MHz enabled	V_{DD} =3V, 25°C, RCDiv=1	I_{VDDh1}		10	13.7	µA
	V_{DD} =3V, -40 to 85°C, RCDiv=1	I_{VDDh1}		10	15	µA
	V_{DD} =3V, 25°C, RCDiv=8	I_{VDDh1}		6		µA
Active Supply Current CPU on Xtal 32KHz	V_{DD} =3V, 25°C, RC off	I_{VDDa32}		8	9	µA
	V_{DD} =3V, -40 to 85°C, RC off	I_{VDDa32}		8	10	µA
Standby Supply Current CPU on Xtal 32KHz	V_{DD} =3V, 25°C, RC off	I_{VDDh32}		0.8	1	µA
	V_{DD} =3V, -40 to 85°C, RC off	I_{VDDh32}		0.8	2	µA
BrownOut or SVLD consumption	V_{DD} =3V, -40 to 85°C	I_{VDDa32}		6		µA
SLEEP Supply Current	V_{DD} =3V, 25°C			0.16	0.2	µA
	V_{DD} =3V, -40 to 85°C	I_{VDDs1}		0.16	2	µA

Active supply currents are measured using a checkerboard read-write loop in the low power RAM.

17.6 DC Characteristics – Voltage detection levels

Parameter	Conditions	Symbol	Min.	Typ.	Max.	Unit
POR static level	-40 to 85°C	V_{POR}		1.5		V
Power-Check level VBAT increasing	-40 to 85°C	V_{PWC}	1.81	2.05	2.26	V
BrownOut level	-40 to 85°C	V_{BO}	1.81	2.0	2.26	
SVLD0, VBAT decreasing	-40 to 85°C	V_{SVLD0}	1.81	2.0	2.26	
SVLD1, VBAT decreasing	-40 to 85°C	V_{SVLD1}	1.90	2.1	2.37	
SVLD2, VBAT decreasing	-40 to 85°C	V_{SVLD2}	1.99	2.2	2.47	
SVLD3, VBAT decreasing	-40 to 85°C	V_{SVLD3}	2.08	2.3	2.58	
SVLD4, VBAT decreasing	-40 to 85°C	V_{SVLD4}	2.16	2.4	2.69	
SVLD5, VBAT decreasing	-40 to 85°C	V_{SVLD5}	2.24	2.5	2.81	
SVLD6, VBAT decreasing	-40 to 85°C	V_{SVLD6}	2.35	2.6	2.91	
SVLD7, VBAT decreasing	-40 to 85°C	V_{SVLD7}	2.99	3.4	3.73	
SVLD & BrownOut temperature dependency	-40 to 85°C			± 50		ppm/°C

17.7 DC Characteristics – Oscillators

Parameter	Conditions	Symbol	Min.	Typ.	Max.	Unit
XTAL Integrated Input capacitor	Reference on V_{SS} T=25°C	C_{IN}		7		pF
Xtal Integrated Output capacitor	Reference on V_{SS} T=25°C	C_{OUT}		14		pF
Xtal Oscillator start time	$V_{DD} > V_{DD}$Min T=25°C	t_{dosc}		0.5	3	s
RC Oscillator 10MHz	Trimm reg 0#7F	$F_{RC10MHz}$	7	10	13	MHz
Trimm range 10MHz				± 39		%
RC Oscillator 1MHz	Trimm reg 0#7F	F_{RC1MHz}	0.7	1	1.3	MHz
Trimm range 1MHz				± 46		%

17.8 DC Characteristics - I/O Pins

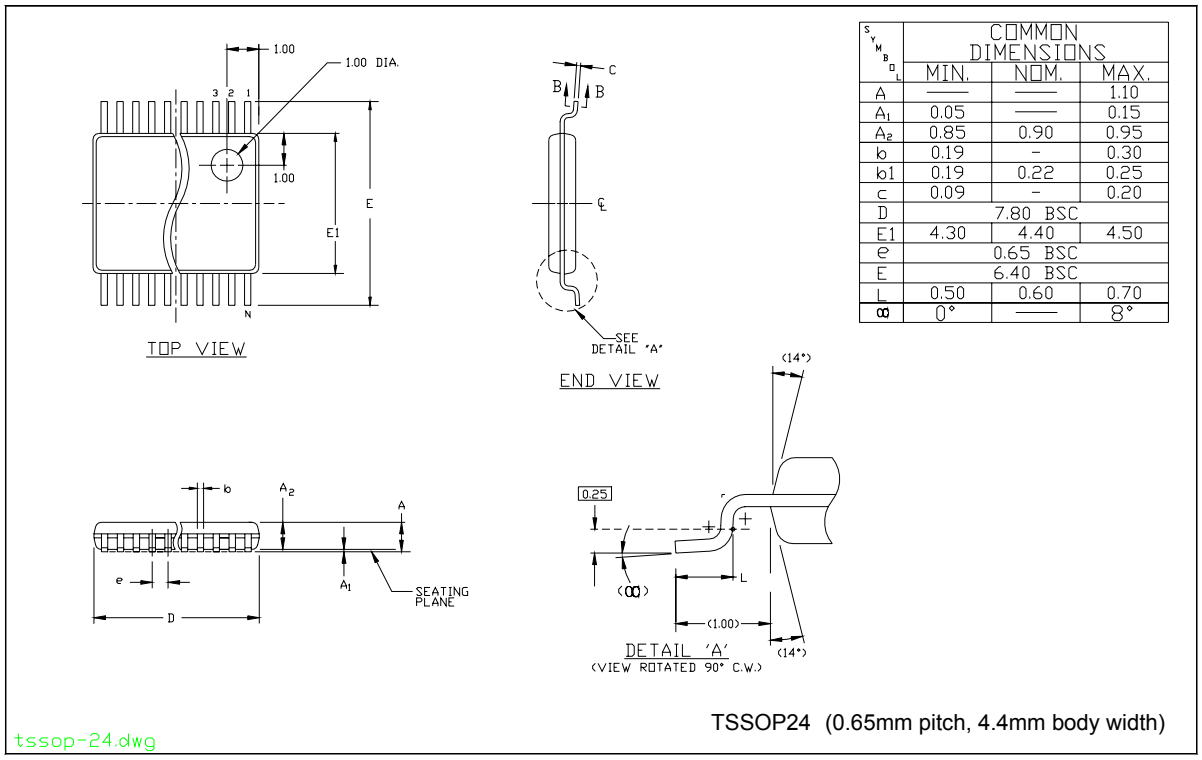
Conditions: T= -40 to 85°C, V_{DD}=3.0V (unless otherwise specified)

Parameter	Conditions	Symb.	Min.	Typ.	Max.	Unit
Input Low voltage						
Ports A,B, Reset		V_{IL}	V_{SS}		0.2* V_{DD}	V
OscOut	Note 1	V_{IL}	V_{SS}		0.1*Vreg	V
Input High voltage						
Ports A,B, Reset,		V_{IH}	0.7* V_{DD}		V_{DD}	V
OscOut	Note 1	V_{IH}	0.9*Vreg		Vreg	V
Input Hysteresis PA[7:0], PB[7:0]	Temp=25°C	V_{Hyst}		0.4		V
IOL drive 1 PA[5:0], PB[7:4]	V_{DD} =3.0V , V_{OL}=0.30V	I_{OL}	1.75	2.8		mA
	V_{DD} =3.0V , V_{OL} =1.0V	I_{OL}		7.40		mA
IOL drive 2 PA[7:6], PB[3:0]	V_{DD} =3.0V , V_{OL} =0.15V	I_{OL}	1.8	3.3		mA
	V_{DD} =3.0V , V_{OL} =0.30V	I_{OL}		6.6		mA
	V_{DD} =3.0V , V_{OL} =1.0V	I_{OL}		21.4		mA
IOH drive 1 PA[5:0], PB[7:4]	V_{DD} =3.0V, V_{OH}= V_{DD} - 0.30V	I_{OH}		-2.1	-1.4	mA
	V_{DD} =3.0V, V_{OH} = V_{DD} - 1.0V	I_{OH}		-7.0		mA
IOH drive 2 PA[7:6], PB[3:0]	V_{DD} =3.0V, V_{OH} = V_{DD} - 0.15V	I_{OH}		-3.8	-2.9	mA
	V_{DD} =3.0V, V_{OH} = V_{DD} - 0.30V	I_{OH}		-8.6		mA
	V_{DD} =3.0V, V_{OH} = V_{DD} - 1.0V			-26.5		mA
Input Pull-down Port A,B	V_{DD} =3.0V, Pin at 3.0V, 25°C	R_{PD2}	80k	100k	120k	Ohm
Input Pull-up Port A,B	V_{DD} =3.0V, Pin at 0.0V, 25°C	R_{PU2}	120k	160k	200k	Ohm
Input Pull-down Test, Reset	V_{DD} =3.0V, Pin at 3.0V, 25°C	R_{PD1}	20k	40k	60k	Ohm

Note 1; OscOut is only usable if no XTAL connected, its input is referenced to the regulated voltage Vreg.

17.9 Package drawings

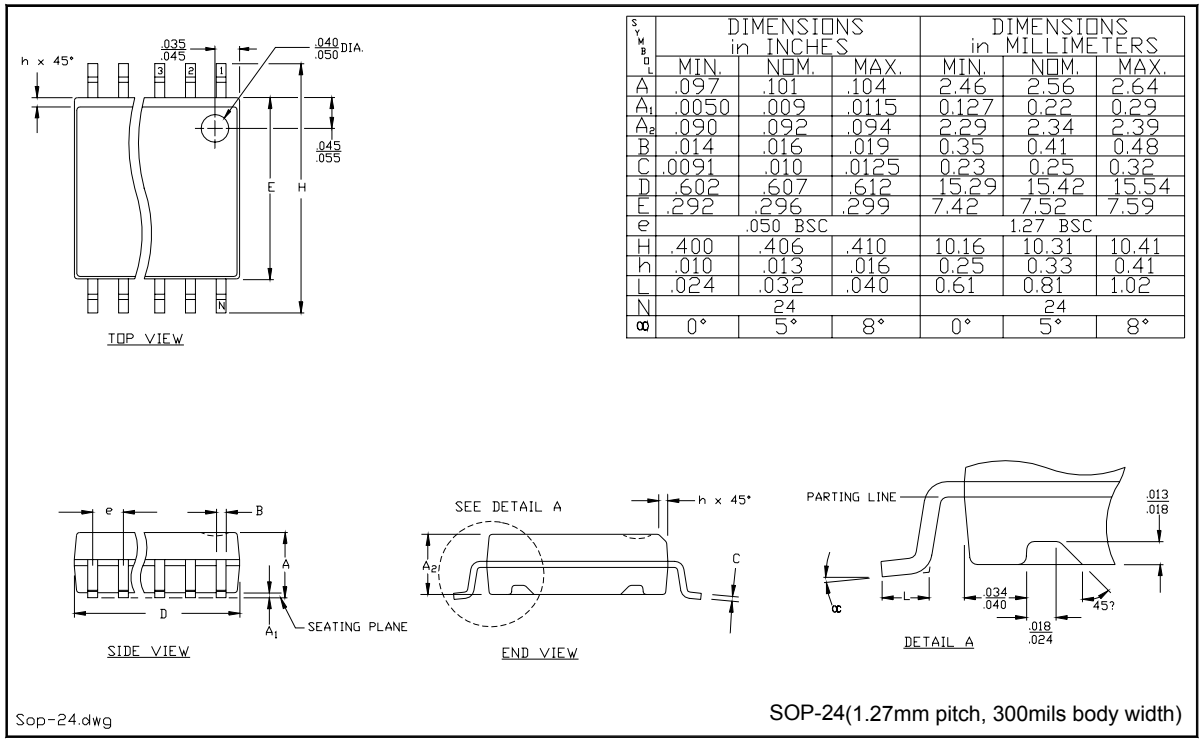
Figure 47. Dimensions of TSSOP24 Package



SYMBOL	COMMON DIMENSIONS		
	MIN.	NOM.	MAX.
A	——	——	1.10
A_1	0.05	——	0.15
A_2	0.85	0.90	0.95
b	0.19	-	0.30
b1	0.19	0.22	0.25
c	0.09	-	0.20
D	7.80 BSC		
E1	4.30	4.40	4.50
e	0.65 BSC		
E	6.40 BSC		
L	0.50	0.60	0.70
θ	0°	——	8°

TSSOP24 (0.65mm pitch, 4.4mm body width)

Figure 48. Dimensions of SOIC24 Package



SYMBOL	DIMENSIONS in INCHES			DIMENSIONS in MILLIMETERS		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	.097	.101	.104	2.46	2.56	2.64
A_1	.0050	.009	.0115	0.127	0.22	0.29
A_2	.090	.092	.094	2.29	2.34	2.39
B	.014	.016	.019	0.35	0.41	0.48
C	.0091	.010	.0125	0.23	0.25	0.32
D	.602	.607	.612	15.29	15.42	15.54
E	.292	.296	.299	7.42	7.52	7.59
e	.050 BSC			1.27 BSC		
H	.400	.406	.410	10.16	10.31	10.41
h	.010	.013	.016	0.25	0.33	0.41
L	.024	.032	.040	0.61	0.81	1.02
N	24			24		
θ	0°	5°	8°	0°	5°	8°

SOP-24(1.27mm pitch, 300mils body width)

18 Ordering information Flash device

Packaged Device:

EM6812 F8 TP24 A +

Flash Memory Size:
F2 = 3k x 22 Bit (5.6 kByte)
F4 = 4k x 22 Bit (11.2 kByte)
F8 = 8k x 22 Bit (22.5 kByte)

Package:
SO24 = 24 pin SOIC
TP24 = 24 pin TSSOP

Delivery Form:
A = Stick
B = Tape&Reel

Device in DIE Form:

EM6812 F8 WS 11

Flash Memory Size:
F2 = 3k x 22 Bit (5.6 kByte)
F4 = 4k x 22 Bit (11.2 kByte)
F8 = 8k x 22 Bit (22.5 kByte)

Die form:
WW = Wafer
WS = Sawn Wafer/Frame
WP = Waffle Pack

Thickness:
11 = 11 mils (280um), by default
27 = 27 mils (686um), not backlapped

In its packaged form, EM6812 is available in **green mold / leadfree** (symbolized by a "+" at the end of the part number).

Note: Please contact EM Microelectronic for availability of other die thicknesses.

Ordering Part Number (selected examples)

Part Number	Memory Size	Package/Die Form	Delivery Form/Thickness
EM6812F2TP24B+	2k x 22 bit Flash	24 pin TSSOP	Tape&Reel, 3000 pieces
EM6812F4TP24A+	4k x 22 bit Flash	24 pin TSSOP	Stick, 50 pieces
EM6812F8SO24B+	8k x 22 bit Flash	24 pin SOIC	Tape&Reel, 2000 pieces
EM6812F8WS11	8k x 22 bit Flash	Sawn wafer	11 mils

Please make sure to give the complete Part Number when ordering.

Package Marking

SOIC marking:

First line:	E	M	6	8	1	2		%	%	%	Y
Second line:	P	P	P	P	P	P	P	P	P	P	P
Third line:	F	&	D								

TSSOP marking:

E	M	6	8	1	2	%	%
P	P	P	P	P	P	P	P
		F	&	D		Y	P

Where: %%% = specific number assigned by EM
Y = Year of assembly
PP…P = Production identification (date & lot number) of EM Microelectronic
& = memory size (2,4, 8 k Instruction)