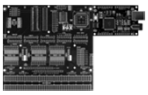
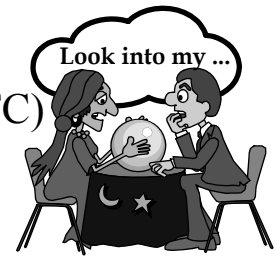


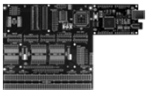
EEL 3744

Menu

- Main Timer System for
  - > 68HC11/12
  - > XMEGA Timer System
- Real-Time Interrupt/Counter (RTI/RTC)
  - >68HC11/12
    - – RTI Hardware and Registers
    - – RTI Programming Examples
      - ☞ Use RTI interrupt; use RTIF & polling
    - – Free-running Counter & Timer Overflow (TOF)
      - ☞ Example using TOF
  - >XMEGA
    - – RTC specs
    - – RTC Hardware
    - – RTC Registers
    - – RTC example



See docs/examples on web-site:
doc8331 (Sec 18), doc8385 (Sec 20), M&M: Ch 14, doc8047, RTC.asm
68HC11/12: S&HE (Ch 10), M68HC12B/D.pdf (Sec 10,12), RTIa*.asm, RTIb.asm, CoPrRTI.asm, TimeOver.asm, TimeOver_debug.asm

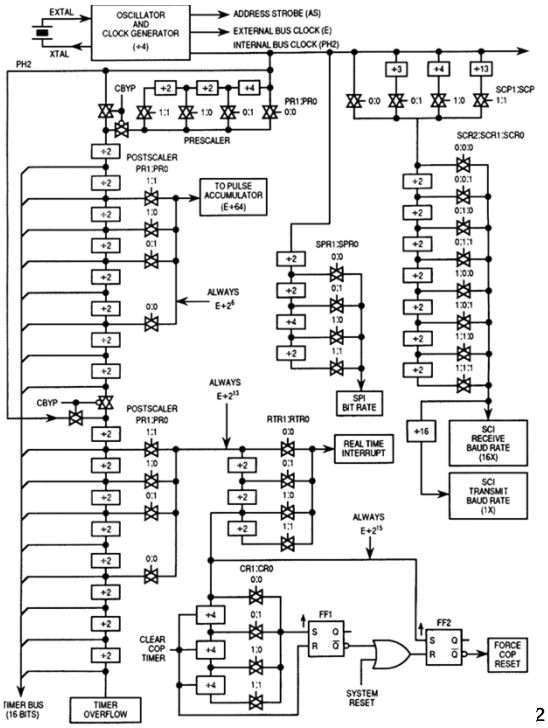


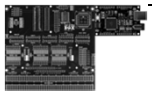
EEL 3744

68HC11 Clock Divider Circuit

- Clock Divider
  - >Major clock divider chains

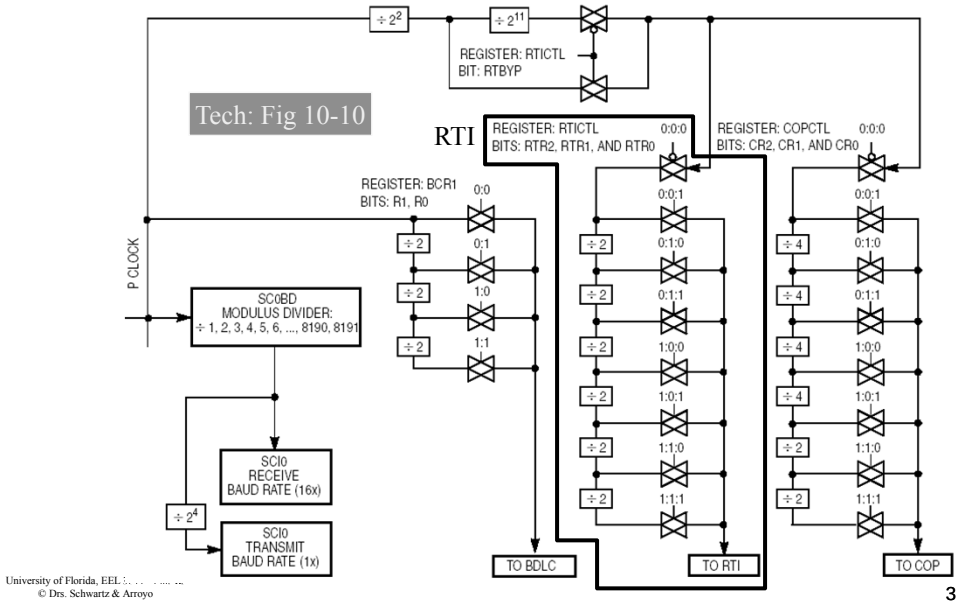
RM: Fig 10-3

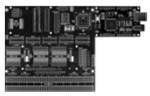




EEL 3744

68HC12 Clock Chain for SCI, BDLC, RTI & COP

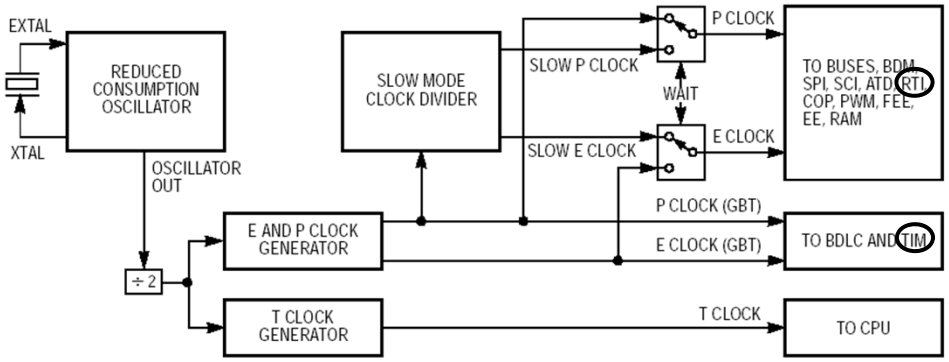




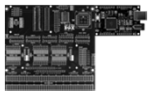
EEL 3744

68HC12 Clock Generation Module (CGM) Block Diagram

Tech: Fig 10-1



- P-Clock and E-clock are at same frequency, but 90° out of phase
- T-Clock and E-clock are at same frequency (when not in wait mode)

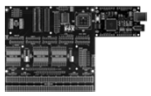


EEL 3744

XMEGA 16-bit Timer/Counter Type 0 and Type 1

See doc8331, Sec 14 & doc8385, Sec 16

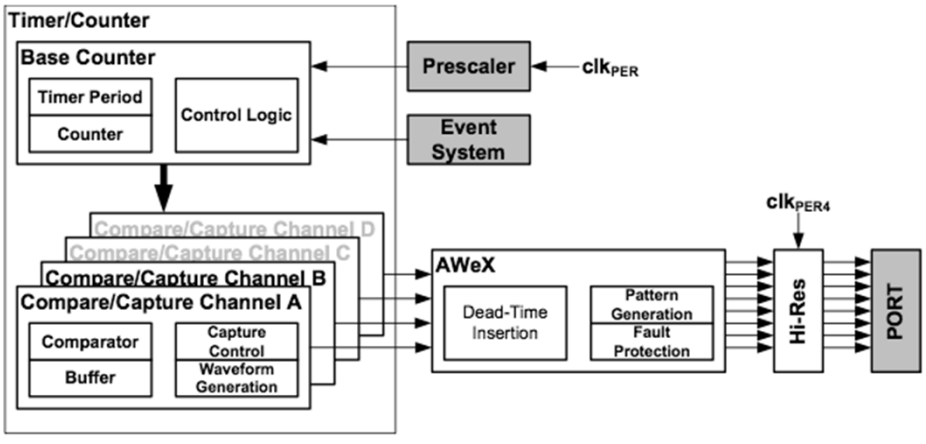
- XMEGA has a set of eight 16-bit timer/counters (TC)
- Two TCs can be combined to create a 32-bit TC
- A TC consists of a base counter and a set of compare or capture (CC) channels
  > Waveform generation available
- TC 0 has four CC channels; TC 1 has two CC channels
- TC 0 has the split mode feature that splits it into two 8-bit Timer/Counters with four compare channels each

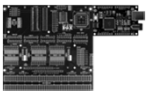


EEL 3744

XMEGA Timer/Counter type 0 and type 1

See doc8331, Fig 14-1

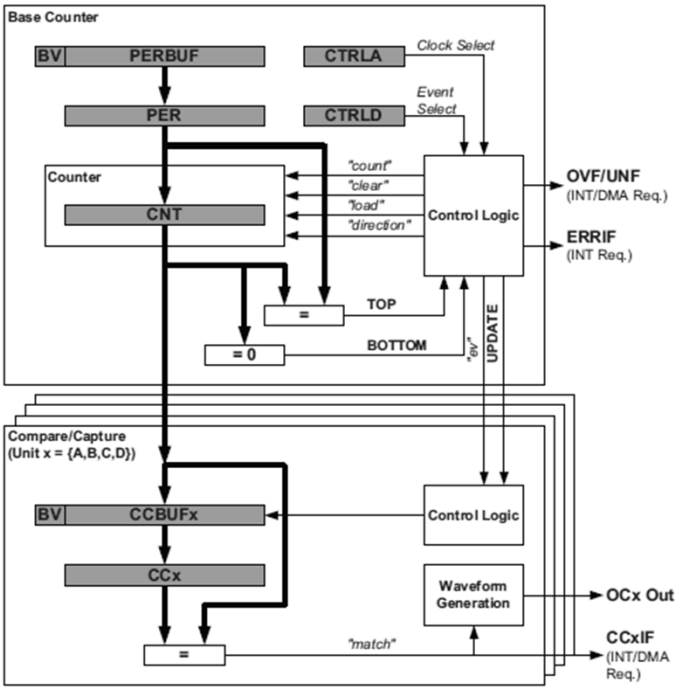


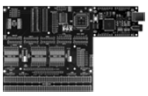


EEL 3

XMEGA Timer/ Counter Block Diagram

See doc8331, Fig 14-2

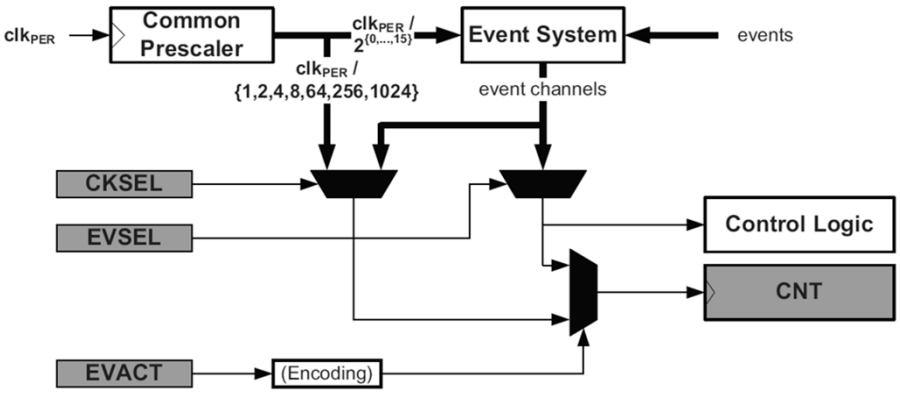


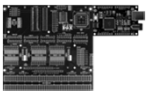


EEL 3744

XMEGA Clock and Event Selection

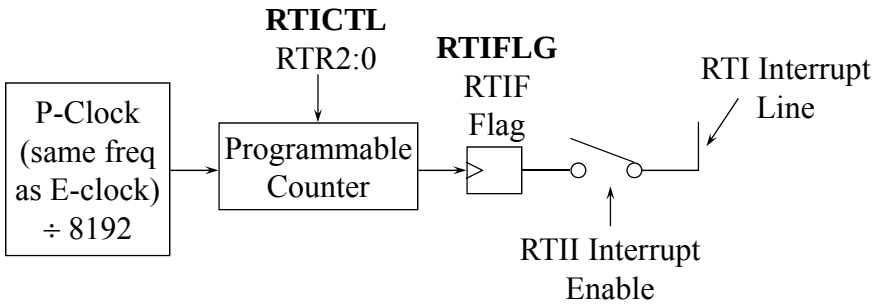
See doc8331, Fig 14-3



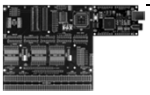


EEL 3744

68HC12 (& ~11) RTI (Real-Time Interrupt) Hardware



S&HE: Fig 10.7

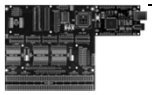


EEL 3744

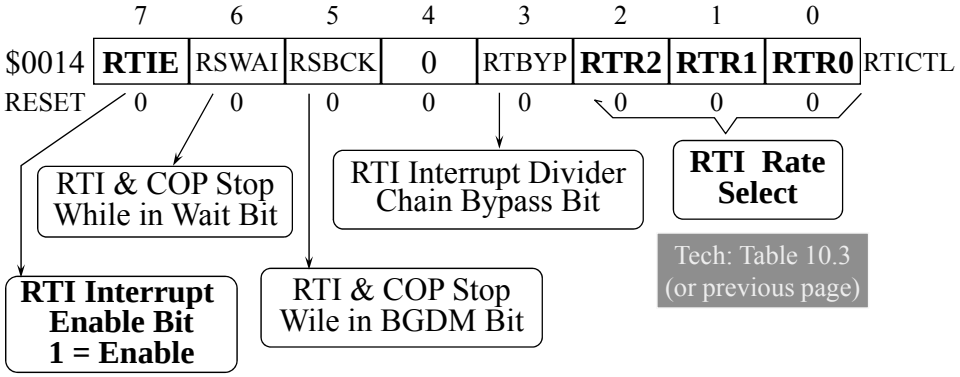
68HC12 RTI (Real-Time Interrupt) Rate Control

- RTR bits in register RTICTL

RTR2:0	Divide E by	E = 2MHz Timeout Period
000	OFF	OFF
001	2^13	4.096ms
010	2^14	8.192ms
011	2^15	16.384ms
100	2^16	32.768ms
101	2^17	65.536ms
110	2^18	131.072ms
111	2^19	262.144ms

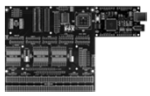


EEL 3744 RTICLT: Real-Time Interrupt Control Register

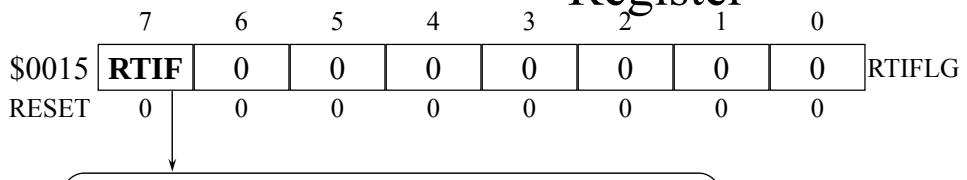


- Crystal Freq. vs. RTR2:0
  - > $f_{RTI} = (E/2^{13})/X$, where X=1, 2, 4, 8, 16, 32, 64

T_{RTI}(E=2MHz) ≈ 4.10ms, 8.19ms, 16.38ms, … 263.14ms
f_{RTI}(E=2MHz) ≈ 244.1Hz, 122.1Hz, 61.4Hz, … 3.815Hz

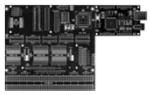


EEL 3744 **68HC12** RTIFLG: Real-Time Interrupt Flag Register



Real-Time Interrupt Flag:
- **Set automatically** at the end of every RTI period
- **Clear** by writing 1 to RTIF, bit-7 of RTIFLG

- Writing a 1 is a strange way to clear a bit!
  - > Direct clearing flag
  - > Show FF example of clearing flag



EEL 3744

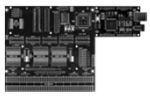
68HC12 RTI Programming Examples

- Running a single process
  - >See examples


`RTIa.asm` `RTIb.asm`

  - >Note that the time between interrupts is 32.768 ms assuming E=2MHz (and 8.196 ms if E=8MHz)
  - >A polling version of first example above

`RTIa_p.asm`



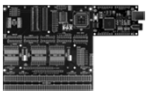
EEL 3744

68HC12 RTI Co-Procesing Example

- Running two processes
  - >See example

`Co_Proc_RTI.asm`

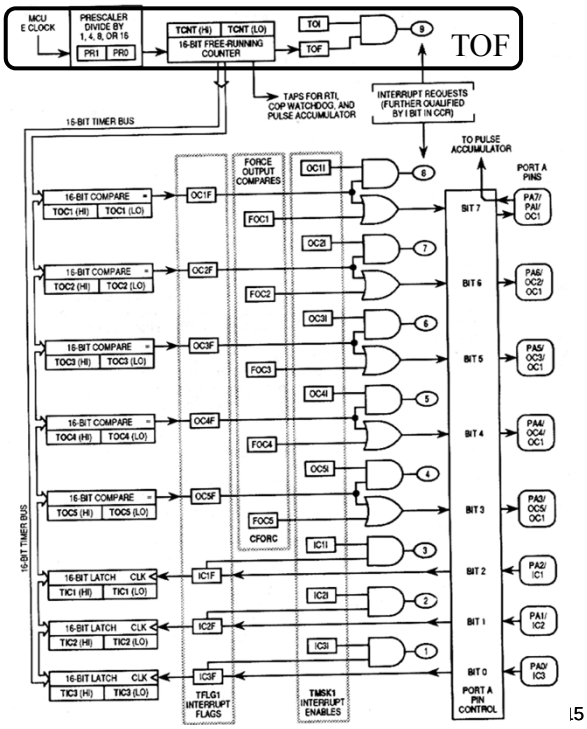
  - >We will get more on co-processing later in the semester (if time permits)

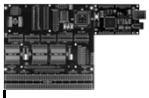


EEL 3744

68HC11 Main Timer System

RM: Fig 10-1



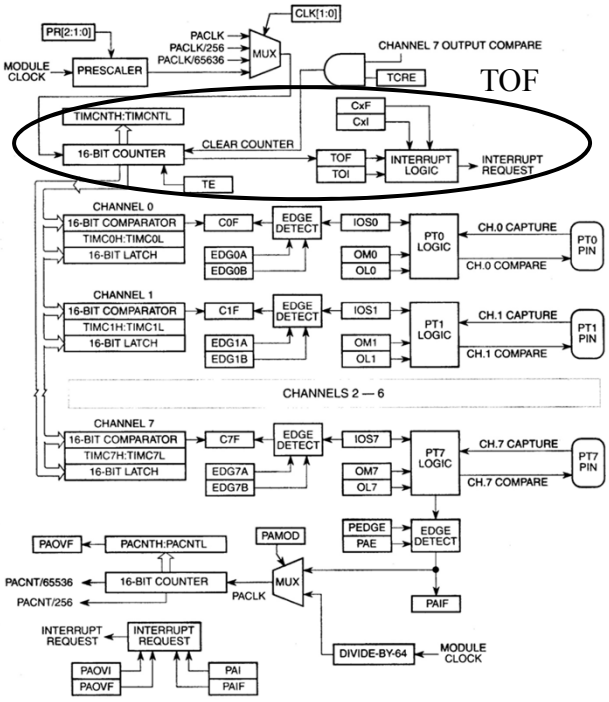


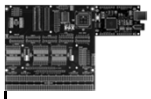
EEL 3744

68HC12 Timer System Block Diagram

S&HE: Fig 10.1

- No RTI on this figure



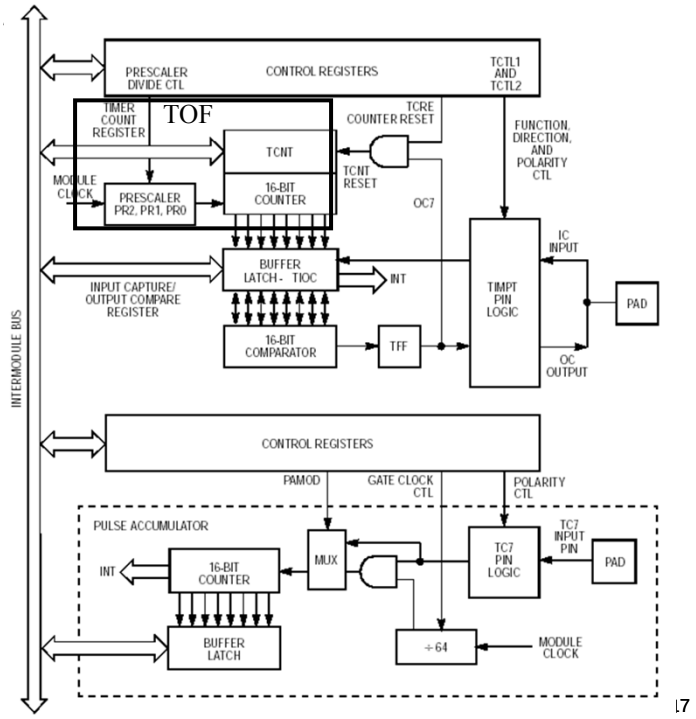


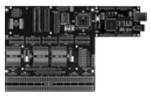
EEL 3

68HC12 Timer Block Diagram

Tech: Fig 12-1

- No RTI on this figure

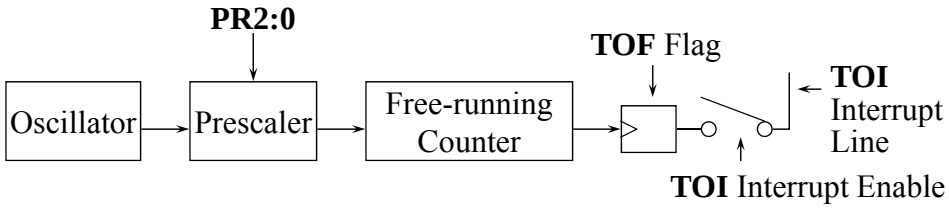




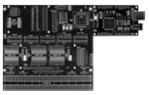
EEL 3744

68HC11/12 Free-Running Counter and TOI

- Free-running Counter and TOI (Timer Overflow Interrupt)



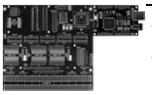
S&HE: Fig 10.2



EEL 3744 **68HC11/12** TCNT: Free-Running Counter

- TCNT - Free-Running Counter (@ E-rate/Prescaler)
  - >TCNT effects Output Compare (OC), Input Capture (IC), Timer Overflow (TOF), Pulse Accumulator (PA)
  - >TCNT does **not** effect RTI

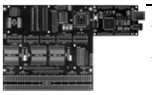
	7	6	5	4	3	2	1	0	
$0084	Bit 15	-	-	-	-	-	-	Bit 8	TCNT High
RESET	0	0	0	0	0	0	0	0	
	7	6	5	4	3	2	1	0	
$0085	Bit 7	-	-	-	-	-	-	Bit 0	TCNT Low
RESET	0	0	0	0	0	0	0	0	



EEL 3744 **68HC12** TSCR: Timer System Control Register

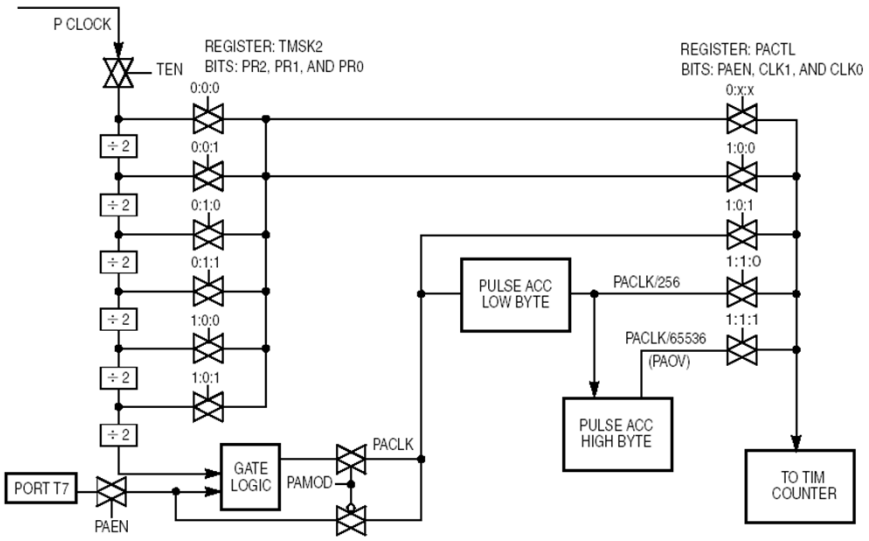
- TSCR: Timer System Control Register
  - >**TEN (Timer Enable): 1 = enable** (activates the timer)
  - >TSWAI (Timer Stops While in Wait)
  - >TSBCK (Timer Stops While in Background Mode)
  - >TFFCA (Timer Fast Flag Clear All): [S&HE: Sec 10.10]

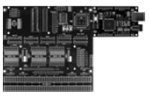
	7	6	5	4	3	2	1	0	
$0086	**TEN**	TSWAI	TSBCK	TFFCA	0	0	0	0	TSCR
RESET	0	0	0	0	0	0	0	0	



EEL 3744 **68HC12** Clock Chain for TIM (Standard Timer Module)

Tech: Fig 10-11





EEL 3744 **68HC12** TMSK2: Timer Interrupt Mask Register 2 (Prescaler bits)

- TMSK2 - Timer Interrupt Mask Register 2

	7	6	5	4	3	2	1	0	
$008D	**TOI**	0	TPU	TDRB	TCRE	**PR2**	**PR1**	**PR0**	TMSK2
RESET	0	0	0	0	0	0	0	0	

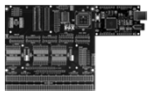
Timer Pull-up Resistor Enable
0: no pull-up
1: pull-up on all Port T pins when configured as inputs

Timer Drive Reduction

See next page

Timer Overflow Interrupt Enable

Timer Counter Reset Enable



EEL 3744

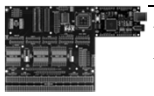
68HC12 Timer Prescaler bits in TMSK2

- Prescaler and resultant count period and overflow period

Tech: Table 12.3

RTI is **NOT** affected by the prescaler

PR2:0	Prescale Factor (2^{PR})	E = 2MHz Count/Overflow
000	1	500ns / 32.77ms
001	2	1μs / 65.54ms
010	4	2μs / 131.7ms
011	8	4μs / 262.1ms
100	16	8μs / 524.3ms
101	32	16μs / 1048.6ms
11X	---	--------



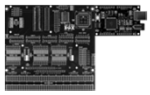
EEL 3744

68HC12 TFLG2: Timer Interrupt Flag 2

	7	6	5	4	3	2	1	0	
$008F	**TOF**	0	0	0	0	0	0	0	TFLG2
RESET	0	0	0	0	0	0	0	0	

Timer Overflow Interrupt Flag:
- **Set automatically** at TCNT = $FFFF → $0000
- **Clear** by writing 1 to TOF, bit-7 of TFLG2

- Notice that the flag bit is in same position as the corresponding interrupt enable bit
  - > Bit 7 of TFLG2 has the TOF flag and bit 7 of TMSK2 has the TOI interrupt enable
  - > Bit 7 of RTIFLG has the RTIF and bit 7 of RTICTL has the RTIE



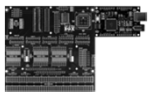
EEL 3744
68HC12 TOF Programming Examples

- Reverse all Port A bits every 32.768 ms
  - > See example

`TimeOver.asm`

  - > Note that the time between interrupts is 32.768 ms assuming E=2MHz (and 8.196 ms if E=8MHz)

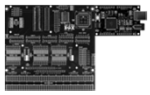
`TimeOver_debug.asm`



EEL 3744
XMEGA 16-bit Real-Time Counter (RTC)

See doc8331: Sec 18 & doc8385: Sec 20

- The 16-bit RTC typically runs continuously, **including** in low-power sleep modes
- The reference clock is typically the 1.024kHz output from a high-accuracy crystal of 32.768kHz
  - > With a 32.768kHz clock source, the maximum resolution is $1/32.768\text{kHz} = 1\text{s}/ 2^{15} \approx 30.5\mu\text{s}$
  - > With a 32.768kHz clock source, time-out periods can range up to 2 seconds (at max resolution) = $2^{16} \times 30.5\mu\text{s} = 2^{16} / 2^{15} = 2\text{s}$
- The RTC has a 10-bit programmable prescaler that can scale down the reference clock before it reaches the counter
  - > With maximum prescaler (1024) & 32kHz clock, range is ≈ 2000s

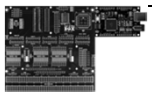


EEL 3744

XMEGA 16-bit Real-Time Counter (RTC)

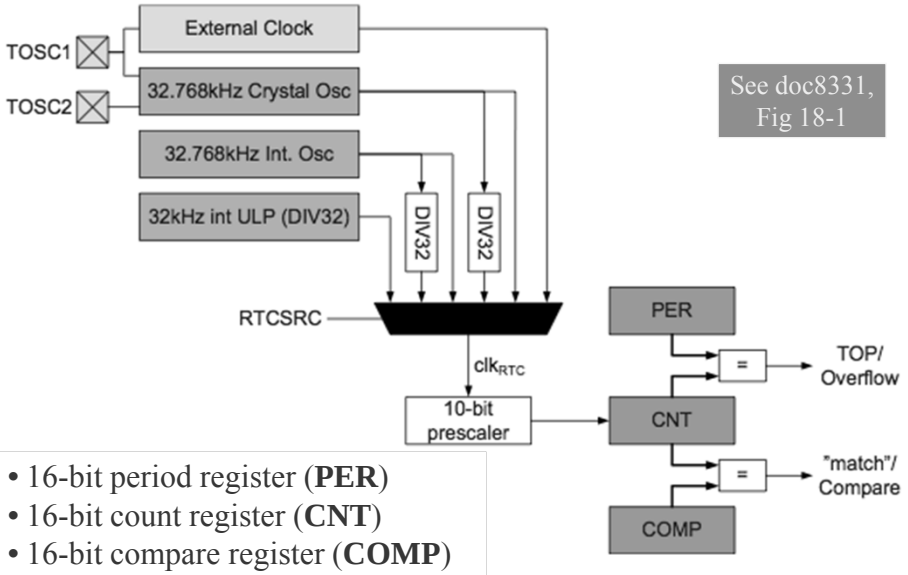
See doc8331, Sec 18.2

- With a resolution of 1s, the maximum time-out is 65,536s (≈18.2 hours)
- RTC can generate two types of interrupts
  - > The RTC can give a **compare interrupt** and/or event when the counter equals the compare register value
    - – Occurs at first count after the counter value equals **Compare** register value
  - > The RTC has an **overflow interrupt** and/or event when it equals the period register value
    - – Occurs at first count after the counter value equals the **Period** register value
    - – Overflow will also reset the counter value to zero
- RTC is asynchronous with respect to the main system clock



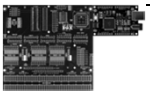
EEL 3744

XMEGA Real-Time Counter (RTC) Overview



See doc8331, Fig 18-1

- 16-bit period register (**PER**)
- 16-bit count register (**CNT**)
- 16-bit compare register (**COMP**)



EEL 3744 RTC CTRL – Control Register

- **PRESCALER[2:0]: Clock Prescaling factor**
  - > These bits define the prescaling factor for the RTC clock

See doc8331, Table 18-1

PRESCALER[2..0]	Group Config	TRC clock prescaling
000	OFF	No clock sours; RTC stopped
001	DIV1	RTC clock / 1 (no prescaling)
010	DIV2	RTC clock / 2
011	DIV8	RTC clock / 8
100	DIV16	RTC clock / 16
101	DIV64	RTC clock / 64
110	DIV256	RTC clock / 256
111	DIV1024	RTC clock / 1024

Bit	7	6	5	4	3	2	1	0
+0x00	–	–	–	–	–	PRESCALER[2:0]		
Read/Write	R	R	R	R	R	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

RTC_CTRL

© Drs. Schwartz & Arroyo

30

EEL 3744 RTC STATUS – Status Register

See doc8331, Sec 18.3.2

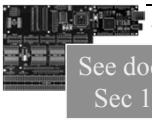
- **SYNCBUSY: Synchronization Busy Flag**
  - > Flag is set when the CNT, CTRL, PER, or COMP register is busy synchronizing between the RTC clock and system clock domains after writing any of these registers or when waking up from a sleep mode where the peripheral clock is stopped
  - > This flag is automatically cleared when the sync is complete

Bit	7	6	5	4	3	2	1	0
+0x01	–	–	–	–	–	–	–	SYNCBUSY
Read/Write	R	R	R	R	R	R	R	R
Initial Value	0	0	0	0	0	0	0	0

RTC_STATUS

University of Florida, EEL 3744 – File **15**
© Drs. Schwartz & Arroyo

31



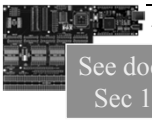
EEL 3744 RTC INTCTRL – Interrupt Control Register

- **COMPINTLVL: Compare Match Interrupt Enable**
  - >These bits enable the RTC compare match interrupt and select the interrupt level
  - >The enabled interrupt will trigger when COMPIF in the INTFLAGS register is set

Interrupt Level Config	Group Config	Description
00	Off	Interrupt disabled
01	Lo	Low-level interrupt
10	Med	Mid-level interrupt
11	Hi	High-level interrupt

Bit	7	6	5	4	3	2	1	0
+0x02	–	–	–	–	COMPINTLVL[1:0]		OVFINTLVL[1:0]	
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

RTC_INTCTRL



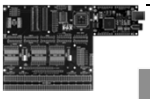
EEL 3744 RTC INTCTRL – Interrupt Control Register

- **OVFINTLVL[1:0]: Overflow Interrupt Enable**
  - >These bits enable the RTC overflow interrupt and select the interrupt level
  - >The enabled interrupt will trigger when OVFIF in the INTFLAGS register is set

Interrupt Level Config	Group Config	Description
00	Off	Interrupt disabled
01	Lo	Low-level interrupt
10	Med	Mid-level interrupt
11	Hi	High-level interrupt

Bit	7	6	5	4	3	2	1	0
+0x02	–	–	–	–	COMPINTLVL[1:0]		OVFINTLVL[1:0]	
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

RTC_INTCTRL



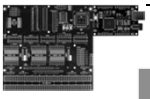
EEL 3744 INTFLAGS – Interrupt Flag Register

See doc8331, Sec 18.3.4

- **COMPIF: Compare Match Interrupt Flag**
  - >Flag is set on the next count after a compare match condition occurs
  - >Cleared automatically when the RTC compare match interrupt vector is executed
  - >Flag can also be cleared by writing a one to it

Bit	7	6	5	4	3	2	1	0
+0x03	–	–	–	–	–	–	COMPIF	OVFIF
Read/Write	R	R	R	R	R	R	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

RTC_INTFLAGS



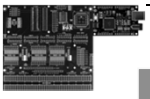
EEL 3744 RTC TEMP – Temporary Register

See doc8331, Sec 18.3.5

- **TEMP[7:0]: Temporary bits**
  - >Used for 16-bit access to the counter value, compare value, and TOP value registers
  - >The low byte of the 16-bit register is stored here when it is written by the CPU
  - >The high byte of the 16-bit register is stored when the low byte is read by the CPU
  - >See also doc8331, section 3.11

Bit	7	6	5	4	3	2	1	0
+0x04	TEMP[7:0]							
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

RTC_TEMP

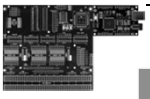


EEL 3744 RTC CNTL – Counter Register Low

See doc8331, Sec 18.3.6

- CNTH and CNTL represent the 16-bit value, CNT
- CNT counts positive clock edges on prescaled RTC clock
- Reading and writing 16-bit values requires special attention; see also doc8331, section 3.11
- Latency of two RTC clock cycles from write to effect
- **CNT[7:0]: Counter Value low byte**
  - >These bits hold the LSB of the 16-bit real-time counter value

Bit	7	6	5	4	3	2	1	0
+0x08	CNT[7:0]							
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

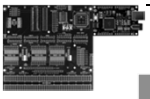


EEL 3744 RTC CNTH – Counter Register High

See doc8331, Sec 18.3.6

- **CNT[15:8]: Counter Value high byte**
  - >These bits hold the MSB of the 16-bit real-time counter value

Bit	7	6	5	4	3	2	1	0
+0x09	CNT[15:8]							
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0



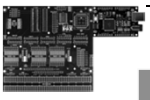
EEL 3744

See doc8331, Sec 18.3.6

RTC PERL – Period Register Low

- PERH and PERL represent the 16-bit value, PER
- PER is constantly compared with the counter value (CNT)
- A match will set OVFIF in the INTFLAGS register and clear CNT.
- Reading and writing 16-bit values requires special attention; see also doc8331, section 3.11
- Latency of two RTC clock cycles from write to effect
- **PER[7:0]: Period low byte**
  - >These bits hold the LSB of the 16-bit RTC TOP value

Bit	7	6	5	4	3	2	1	0
+0x0A	PER[7:0]							
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	1	1	1	1	1	1	1	1



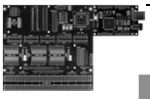
EEL 3744

See doc8331, Sec 18.3.6

RTC PERH – Period Register High

- **PER[15:8]: Period high byte**
  - >These bits hold the MSB of the 16-bit real-time counter value

Bit	7	6	5	4	3	2	1	0
+0x0B	PER[15:8]							
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	1	1	1	1	1	1	1	1

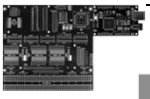


EEL 3744 RTC COMPL – Compare Register Low

See doc8331, Sec 18.3.6

- COMPH and COMPL represent the 16-bit value, COMP. COMP is constantly compared with the
- counter value (CNT). A compare match will set COMPIF in the INTFLAGS register
- Reading and writing 16-bit values requires special attention; see also doc8331, section 3.11
- Latency of two RTC clock cycles from write to effect
- Check that the SYNCBUSY flag (in RTC_STATUS) is cleared before writing to this register

RTC_COMP



EEL 3744 RTC COMPL/COMPH – Compare Register Low/High

See doc8331, Sec 18.3.6

- **COMP[7:0]: Period low byte**
  - >These bits hold the LSB of the 16-bit RTC compare value

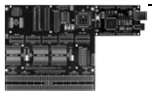
Bit	7	6	5	4	3	2	1	0
+0x0C	COMP[7:0]							
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

RTC_COMP

- **COMP[15:8]: Compare high byte**
  - >These bits hold the MSB of the 16-bit compare value

Bit	7	6	5	4	3	2	1	0
+0x0D	COMP[15:8]							
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

RTC_COMP



EEL 3744 **CLK** RTCCTRL – RTC Control Register

See doc8331, Sec 7.9.4

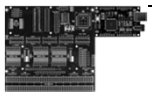
- **RTCSRC[2:0]: RTC Clock Source**
  - >These bits select the clock source for the real-time counter

RTCSRC[2:0]	Group Config	Description
000	ULP	1kHz from 32kHz internal oscillator
001	TOSC	1.024kHz from 32.768kHz crystal oscillator on TOSC
010	RCOSC	1.024kHz from 32.768kHz internal oscillator
011	-	-
100	-	-
101	TOSC32	32.768kHz from 32.768kHz crystal oscillator on TOSC
110	RCOSC32	32.768kHz from 32.768kHz internal oscillator
111	EXTCLK	External clock from TOSC1

CLK_RTCCTRL

Bit	7	6	5	4	3	2	1	0
+0x03	–	–	–	–	RTCSRC[2:0]			RTCEN
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

2



EEL 3744 **CLK** RTCCTRL – RTC Control Register

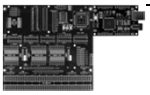
See doc8331, Sec 7.9.4

- **RTCEN: RTC Clock Source Enable**
  - >Setting the RTCEN bit enables the selected RTC clock source for the real-time counter

CLK_RTCCTRL

Bit	7	6	5	4	3	2	1	0
+0x03	–	–	–	–	RTCSRC[2:0]			RTCEN
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

3



EEL 3744 RTC: Register Summary

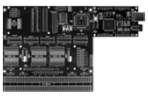
Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
+0x00	CTRL	–	–	–	–	–	PRESCALER[2:0]		
+0x01	STATUS	–	–	–	–	–	–	–	SYNCBUSY
+0x02	INTCTRL	–	–	–	–	COMPINTLVL[1:0]		OVFINTLVL[1:0]	
+0x03	INTFLAGS	–	–	–	–	–	–	COMPIF	OVFIF
+0x04	TEMP	–	–	–	–	–	–	COMPIF	OVFIF
+0x08	CNTL	TEMP[7:0]							
+0x09	CNTH	CNT[7:0]							
+0x0A	PERL	CNT[15:8]							
+0x0B	PERH	PER[7:0]							
+0x0C	COMPL	PER[15:8]							
+0x0D	COMPH	COMP[7:0]							

See doc8331, Sec 18.4

RTC Registers	
RTC_CTRL	RTC_TEMP
RTC_STATUS	RTC_CNT
RTC_INTCTRL	RTC_PER
RTC_INTFLAGS	RTC_COMP

And don't forget **CLK_RTCCTRL** and **PMIC_CTRL**

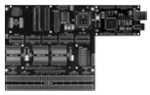
Also **OSC_CTRL** and **OSC_STATUS** (see doc8047)



EEL 3744 RTC Example for **uTinkerer** (not uPAD)

- Could do a demo with emulator **for uTinkerer**
  - >Extra credit:
    - – Change program for uPAD
    - – Send me program
    - – Send me video of it working on uPAD
- Change the interrupt periods

RTC.asm


EEL 3744

The End!