

An instruction summary

Addressing modes for accessing *memory* :

opr8a, opr16a, oprx0_xysp, oprx5_xysp, oprx16_xysp, [D, xysp],[oprx16,xysp]

Transfer instructions

LDAA(or B) #opr8a LDAA(or B) *memory* STAA(or B) *memory*
LD(D/S/X/Y) #opr16a LD(D/S/X/Y) *memory* STA(D/S/X/Y) *memory*
TFR source, dest where source,dest = abcdxys
MOV(B/W) #opr(8a/16a), *memory* MOV(B/W) *memory, memory*
BCLR or BSET *memory*, mask8

Compare instructions : **CBA** Compares A to B (A) – (B)

Compare a register to *memory* CMPA, CMPB, CPD, CPX, CPY, CPS

Simple Branches : **BCC BCS BEQ BNE BMI BPL BVC BVS**

Unsigned Branches : **BHI BHS BLO BLS**

Signed Branches : BGE BGT BLE BLT

Bit Condition Branches : BRSET or BRCLR *memory, mask8, relative*

Loop Primitives: DBNE/DBEQ abdxys, relative

IBNE/IBEQ abdxys, relative

TBNE/TBEQ abdxys, relative

Logical instructions : ANDA(or B) #opr ANDA(or B) *memory*

ORA(or B) #opr ORA(or B) *memory*

EORA(or B) #opr EORA(or B) *memory*

Arithmetic Instructions : ADDA #opr ADDA *memory* (similar for SUBA)

ADDB #opr ADDB *memory* (similar for SUBB)

ADDD #opr ADDD *memory* (similar for SUBD)

ABA (A=A+B) ABX (X=X+B) ABY (Y=Y+B)

INCA INCB INX INY

MUL A*B->A:B EMUL(S) D*Y=>Y:D

EDIV(S) (Y:D) / X ->Y remainder D

IDIV(S) D / X -> X remainder D

Shift Instructions:

Arithmetic : ASL(A/B/D) ASR(A/B/D) ASL *memory* ASR *memory*

Logical LSL(A/B/D) LSR(A/B/D) LSL *memory* LSR *memory*

Interrupts Enable CLI Disable SEI

Vector Table Summary:

RTI at \$FFFF

PAI A Overflow at \$FFFC and PAI A Input Edge at \$FFFA

Timer0 ... Timer 7 at \$FFEE ... \$FFE0

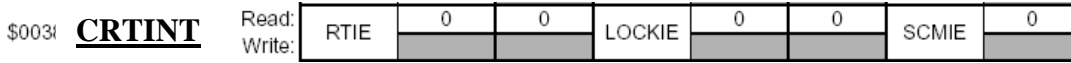
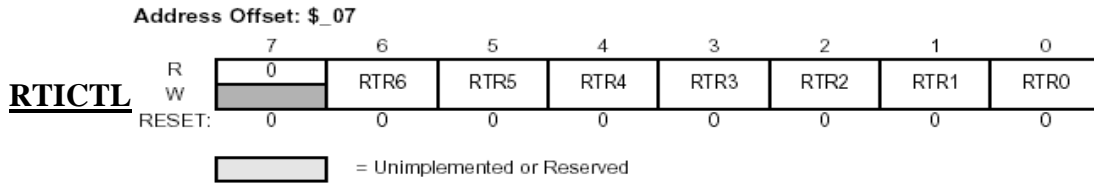
ATD0 and ATD1 at \$FFD2 and \$FFD0, respectively

For C Programs:

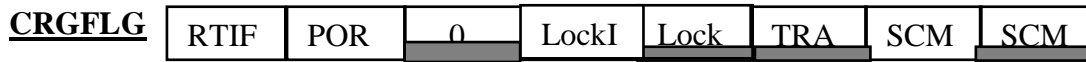
```
#pragma interrupt_handler nameISR()  
void nameISR(void) { ... }
```

An port summary: **All Data Direction Registers (DDRx):** 1 = output, 0 = input

See Table **Last page**,
Examples:
Clock scale factor
\$7F gives 1/8 sec
\$70 gives 1/128 sec



RTIE=1 Enable RTI interrupt

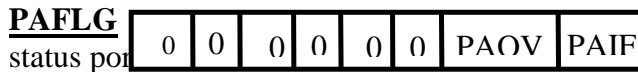


RTIF=1 Timeout
RTIF=0 No timeout
Write 1 to RTIF to clear

PACTL
control port



PAEN – Enable(1)/Disable (0) PAI
PAMOD – Event counter (0) or Gated Time mode(1)
PEDG- Falling(0)/Rising(1) Edge for counting



PAOVF 1=Overflow; 0=No overflow detected
PAIF 1=Innute edge; 0=No innute edge

PACN0..3
8-bit data ports

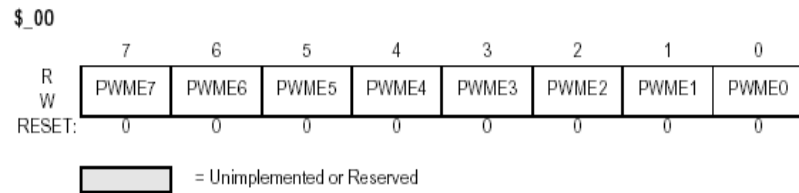
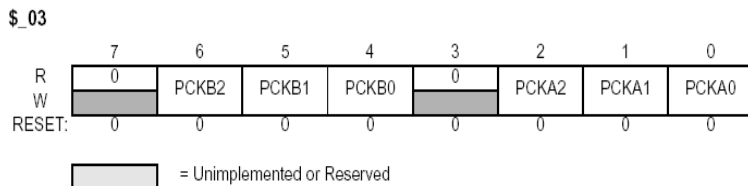


Figure 3-4 PWM Prescale Clock Select Register (PWMPRCLK)

Figure 3-1 PWM Enable Register (PWME)

PWM Channel Period Registers (PWMPERx, x = 0..7 – One register per channel)

Each channel: Dedicated period register

To calculate output period, take selected clock source period for the channel (A, B, SA, or SB) and multiply it by the value in the period register for that channel:

Left: $PWMxPeriod = Channel\ Clock\ Period * PWMPERx$
Centre : $PWMx\ Period = Channel\ Clock\ Period * (2 * PWMPERx)$

| PCKx2 | PCKx1 | PCKx0 | PWM Clock |
|-------|-------|-------|-----------------|
| 0 | 0 | 0 | bus clock |
| 0 | 0 | 1 | bus clock / 2 |
| 0 | 1 | 0 | bus clock / 4 |
| 0 | 1 | 1 | bus clock / 8 |
| 1 | 0 | 0 | bus clock / 16 |
| 1 | 0 | 1 | bus clock / 32 |
| 1 | 1 | 0 | bus clock / 64 |
| 1 | 1 | 1 | bus clock / 128 |

PWM Channel Duty Registers (PWMDTYx)

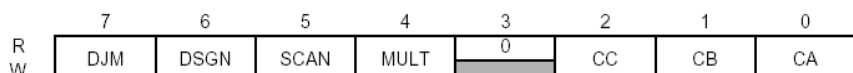
To calculate output duty cycle (high time as % of period) for a part

- Polarity = 0 (PPOLx=0)
Duty Cycle = $[(PWMPERx - PWMDTYx) / PWMPERx] * 100\%$
- Polarity = 1 (PPOLx=1)
Duty Cycle = $[PWMDTYx / PWMPERx] * 100\%$

ATD Control Register 5 (ATDxCTL5)

A write to ATDCTL5

\$_05



SCAN — Continuous Conversion Sequence Mode

1 = Continuous conversion sequences (scan mode)

0 = Single conversion sequence

MULT — Multi-Channel Sample Mode

0 = Sample only from specified channel. Channel selected by channel selection code (CC/CB/CA).

1 = Sample across channels. Number of channels sampled: sequence length

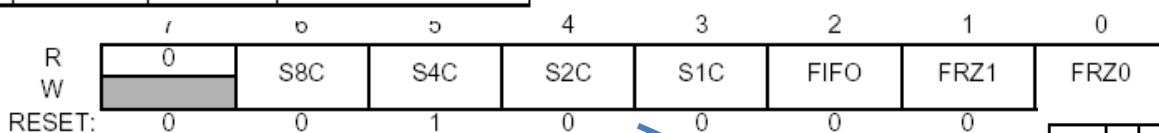
(S8C, S4C, S2C, S1C). 1st channel examined: selection code (CC, CB, CA). Subsequent channels in sequence: increment Channel Select Code.

| CC | CB | CA | Analog Input Channel |
|----|----|----|----------------------|
| 0 | 0 | 0 | AN0 |
| 0 | 0 | 1 | AN1 |
| 0 | 1 | 0 | AN2 |
| 0 | 1 | 1 | AN3 |
| 1 | 0 | 0 | AN4 |
| 1 | 0 | 1 | AN5 |
| 1 | 1 | 0 | AN6 |
| 1 | 1 | 1 | AN7 |

| SRES8 | DJM | DSGN | Result Data Formats |
|-------|-----|------|--|
| 1 | 0 | 0 | 8-bit/left justified/unsigned – bits 8-15 |
| 1 | 0 | 1 | 8-bit/left justified/signed – bits 8-15 |
| 1 | 1 | x | 8-bit/right justified/unsigned – bits 0-7 |
| 0 | 0 | 0 | 10-bit/left justified/unsigned – bits 6-15 |
| 0 | 0 | 1 | 10-bit/left justified/signed – bits 6 – 15 |
| 0 | 1 | x | 10-bit/right justified/unsigned – bits 0 - 9 |

Can choose to use either 8-bit or 10-bit conversion mode. -10-bit mode (left justified): H register holds upper eight bits of the conversion; L result register hold lowest two bits of conversion result in 2 MSB (7 and 6). Note: right justified: H holds 2 upper bits; L the lowest 8 bits)

ATD Result Registers ADR0H/L..ADR7H/L, ADR00H/L..ADR07H/L, ADR10H/L..ADR17H/L



ATDxCTL3

Table 3-2 RTI Frequency Divide Rates

| RTR[3:0] | RTR[6:4] = | | | | | | | |
|------------|------------|------------------------|------------------------|------------------------|------------------------|------------------------|------------------------|------------------------|
| | 000 (OFF) | 001 (2 ¹⁰) | 010 (2 ¹¹) | 011 (2 ¹²) | 100 (2 ¹³) | 101 (2 ¹⁴) | 110 (2 ¹⁵) | 111 (2 ¹⁶) |
| 0000 (+1) | OFF* | 2 ¹⁰ | 2 ¹¹ | 2 ¹² | 2 ¹³ | 2 ¹⁴ | 2 ¹⁵ | 2 ¹⁶ |
| 0001 (+2) | OFF* | 2x2 ¹⁰ | 2x2 ¹¹ | 2x2 ¹² | 2x2 ¹³ | 2x2 ¹⁴ | 2x2 ¹⁵ | 2x2 ¹⁶ |
| 0010 (+3) | OFF* | 3x2 ¹⁰ | 3x2 ¹¹ | 3x2 ¹² | 3x2 ¹³ | 3x2 ¹⁴ | 3x2 ¹⁵ | 3x2 ¹⁶ |
| 0011 (+4) | OFF* | 4x2 ¹⁰ | 4x2 ¹¹ | 4x2 ¹² | 4x2 ¹³ | 4x2 ¹⁴ | 4x2 ¹⁵ | 4x2 ¹⁶ |
| 0100 (+5) | OFF* | 5x2 ¹⁰ | 5x2 ¹¹ | 5x2 ¹² | 5x2 ¹³ | 5x2 ¹⁴ | 5x2 ¹⁵ | 5x2 ¹⁶ |
| 0101 (+6) | OFF* | 6x2 ¹⁰ | 6x2 ¹¹ | 6x2 ¹² | 6x2 ¹³ | 6x2 ¹⁴ | 6x2 ¹⁵ | 6x2 ¹⁶ |
| 0110 (+7) | OFF* | 7x2 ¹⁰ | 7x2 ¹¹ | 7x2 ¹² | 7x2 ¹³ | 7x2 ¹⁴ | 7x2 ¹⁵ | 7x2 ¹⁶ |
| 0111 (+8) | OFF* | 8x2 ¹⁰ | 8x2 ¹¹ | 8x2 ¹² | 8x2 ¹³ | 8x2 ¹⁴ | 8x2 ¹⁵ | 8x2 ¹⁶ |
| 1000 (+9) | OFF* | 9x2 ¹⁰ | 9x2 ¹¹ | 9x2 ¹² | 9x2 ¹³ | 9x2 ¹⁴ | 9x2 ¹⁵ | 9x2 ¹⁶ |
| 1001 (+10) | OFF* | 10x2 ¹⁰ | 10x2 ¹¹ | 10x2 ¹² | 10x2 ¹³ | 10x2 ¹⁴ | 10x2 ¹⁵ | 10x2 ¹⁶ |
| 1010 (+11) | OFF* | 11x2 ¹⁰ | 11x2 ¹¹ | 11x2 ¹² | 11x2 ¹³ | 11x2 ¹⁴ | 11x2 ¹⁵ | 11x2 ¹⁶ |
| 1011 (+12) | OFF* | 12x2 ¹⁰ | 12x2 ¹¹ | 12x2 ¹² | 12x2 ¹³ | 12x2 ¹⁴ | 12x2 ¹⁵ | 12x2 ¹⁶ |
| 1100 (+13) | OFF* | 13x2 ¹⁰ | 13x2 ¹¹ | 13x2 ¹² | 13x2 ¹³ | 13x2 ¹⁴ | 13x2 ¹⁵ | 13x2 ¹⁶ |
| 1101 (+14) | OFF* | 14x2 ¹⁰ | 14x2 ¹¹ | 14x2 ¹² | 14x2 ¹³ | 14x2 ¹⁴ | 14x2 ¹⁵ | 14x2 ¹⁶ |
| 1110 (+15) | OFF* | 15x2 ¹⁰ | 15x2 ¹¹ | 15x2 ¹² | 15x2 ¹³ | 15x2 ¹⁴ | 15x2 ¹⁵ | 15x2 ¹⁶ |
| 1111 (+16) | OFF* | 16x2 ¹⁰ | 16x2 ¹¹ | 16x2 ¹² | 16x2 ¹³ | 16x2 ¹⁴ | 16x2 ¹⁵ | 16x2 ¹⁶ |

| | Number of Conversions per Sequence | | | | | | | | |
|-----|------------------------------------|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| S1C | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | X |
| S2C | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | X |
| S4C | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | X |
| S8C | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

* Denotes the default value out of reset. This value should be used to disable the RTI to ensure future backwards compatibility.

Timer System Control Register (TSCR-TSCR1)

| | | | | | | | |
|-----|-------|-------|-------|---|---|---|---|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| TEN | TSWAI | TSFRZ | TFPCA | 0 | 0 | 0 | 0 |

TEN -- timer enable bit
 0 = disable timer; this can be used to save power consumption
 1 = allows timer to function normally

Timer Input Output Select Register (TIOS)

| | | | | | | | |
|------|------|------|------|------|------|------|------|
| IOS7 | IOS6 | IOS5 | IOS4 | IOS3 | IOS2 | IOS1 | IOS0 |
|------|------|------|------|------|------|------|------|

IOS[7:0] -- Input capture or output compare channel configuration bits
 0 = The corresponding channel acts as an input capture
 1 = The corresponding channel acts as an output compare

Timer Interrupt Flag Register (TFLG1)

| | | | | | | | | |
|-------------|-----|-----|-----|-----|-----|-----|-----|-----|
| value | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| after reset | C7F | C6F | C5F | C4F | C3F | C2F | C1F | C0F |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

C7F-C0F: input capture/output compare interrupt flag bits
 0 = interrupt condition has not occurred
 1 = interrupt condition has occurred

Figure 8.7 Timer interrupt flag 1 register

Timer Interrupt Enable Register (TIE)

| | | | | | | | | |
|-------|-----|-----|-----|-----|-----|-----|-----|-----|
| value | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | C7I | C6I | C5I | C4I | C3I | C2I | C1I | C0I |
| | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

C7I-C0I: input capture/output compare interrupt enable bits
 0 = interrupt disabled
 1 = interrupt enabled

Figure 8.6 Timer interrupt mask 1 register (TMSK1)

(a) TCTL1 register value after reset

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| OM7 | OL7 | OM6 | OL6 | OM5 | OL5 | OM4 | OL4 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

(b) TCTL2 register

| | | | | | | | |
|-----|-----|-----|-----|-----|-----|-----|-----|
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| OM3 | OL3 | OM2 | OL2 | OM1 | OL1 | OM0 | OL0 |

OMn OLn : output level
 0 0 no action (timer disconnected from output pin)
 0 1 toggle OCn pin
 1 0 clear OCn pin to 0