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 $FFF0
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

Address Offset: $_07

RTICTL

	7	6	5	4	3	2	1	0
R	0	RTR6	RTR5	RTR4	RTR3	RTR2	RTR1	RTR0
W								
RESET:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

$003C **CRTINT**

Read:	RTIE	0	0	LOCKIE	0	0	SCMIE	0
Write:								

RTIE=1 Enable RTI interrupt

CRGFLG

RTIF	POR	0	LockI	Lock	TRA	SCM	SCM

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

PACTL
control port

0	PAEN	PAMOD	PEDGE	CI.	CI.	PA	PA

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

PAFLG
status por

0	0	0	0	0	0	PAOV	PAIF

PAOVF 1=Overflow; 0=No overflow detected
PAIF 1=Input edge; 0=No input edge

PACN0..3
8-bit data ports

$_03

	7	6	5	4	3	2	1	0
R	0	PCKB2	PCKB1	PCKB0	0	PCKA2	PCKA1	PCKA0
W								
RESET:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

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

$_00

	7	6	5	4	3	2	1	0
R	PWME7	PWME6	PWME5	PWME4	PWME3	PWME2	PWME1	PWME0
W								
RESET:	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

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 * PWMPER

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 parti
• Polarity = 0 (PPOLx=0)
Duty Cycle = [(PWMPERx-PWMDTYx)/PWMPERx] * 10
• Polarity = 1 (PPOLx=1)
Duty Cycle = [PWMDTYx / PWMPERx] * 100%

ATD Control Register 5 (ATDxCTL5)

- A write to ATDCTL5

$_05

	7	6	5	4	3	2	1	0
R W	DJM	DSGN	SCAN	MULT	0	CC	CB	CA

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 channelselection code (CC/CB/CA).
1= Sample across channels. Number of channels sampled: sequence length
(S8C, S4C, S2C, S1C). 1str channel examined: selection code (CC, CB, CA). Subsequent channels in sequence: increment Channel Select Code.

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)

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

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

	7	6	5	4	3	2	1	0
R W	0	S8C	S4C	S2C	S1C	FIFO	FRZ1	FRZ0
RESET:	0	0	1	0	0	0	0	

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*	$16x2^{10}$	$16x2^{11}$	$16x2^{12}$	$16x2^{13}$	$16x2^{14}$	$16x2^{15}$	$16x2^{16}$

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

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

Timer System Control Register (TSCR-**TSCR1**)

7	6	5	4	3	2	1	0
TEN	TSWAI	TSFRZ	TFFCA	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 (**TFLG1**)

	7	6	5	4	3	2	1	0
	C7F	C6F	C5F	C4F	C3F	C2F	C1F	C0F
value afterreset	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 (**TIE**)

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

Timer Interrupt MASK (**TMASK1**)

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

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

value after reset

(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