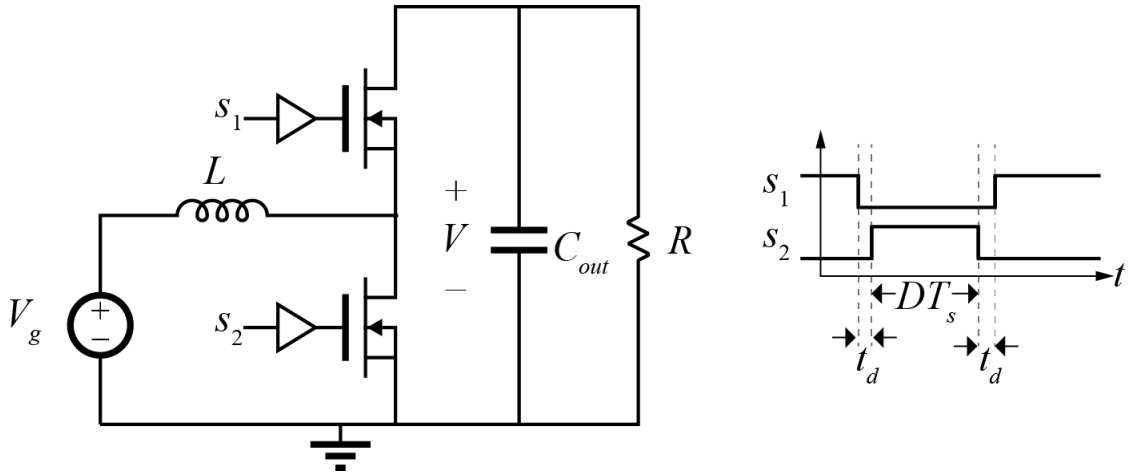
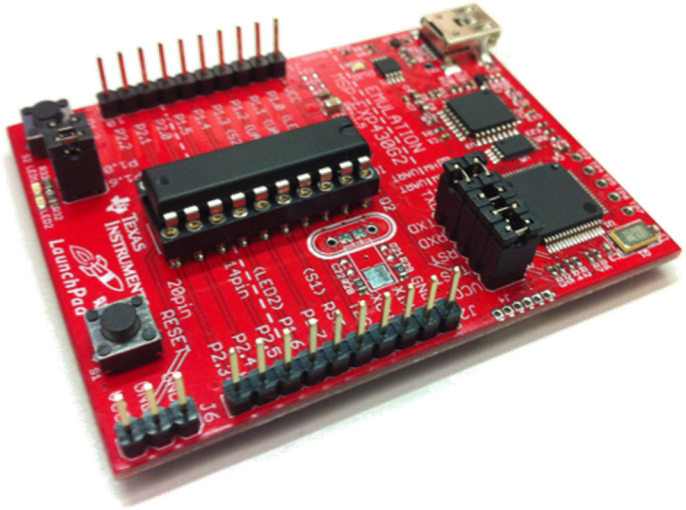
Experiment 2



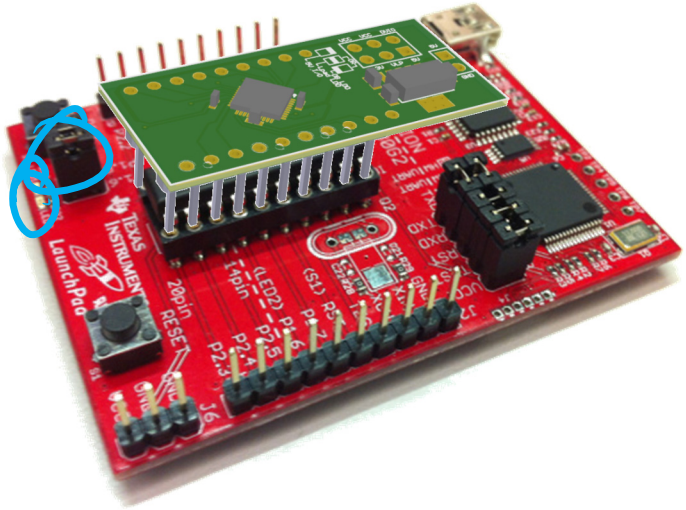
- Experiment 3 will build synchronous boost converter
- To operate open loop, need gate drive signals
- Experiment 2: brief introduction to MSP programming – Generate voltage-controlled PWM signals

Microprocessor: MSP430 Launchpad



- MSP430 microprocessors from Texas Instruments
- Programmable in C or ASM
- Ultra-low power (not a focus here)
- On-board USB bootloader
- Two LEDs, one switch
- Two timers, one 5-channel 10-bit ADC
- System clock up to 16 MHz

High Resolution PWM



MSP430G2553:

- 16 MHz clock
  - Max PWM resolution is 62.5ns

MSP430F2172:

- PWM 16x clock multiplier
  - Max PWM resolution is 4ns

- Final decision TBD; same programming approach applies in either case

Notes on Launchpad

- P1.1 and P1.2 are used as part of the digital communication for the debugger
- P1.0 and P1.6 can be tied to on-board LEDs for visual debugging
- Do not apply power to Vcc; it is generated on-board
- Launchpad **does not** break out all pins on MSP
  - User guide lists all functionality in family
  - Make sure to take note of what *your* chip can do
- Documentation contains both assembly and C code

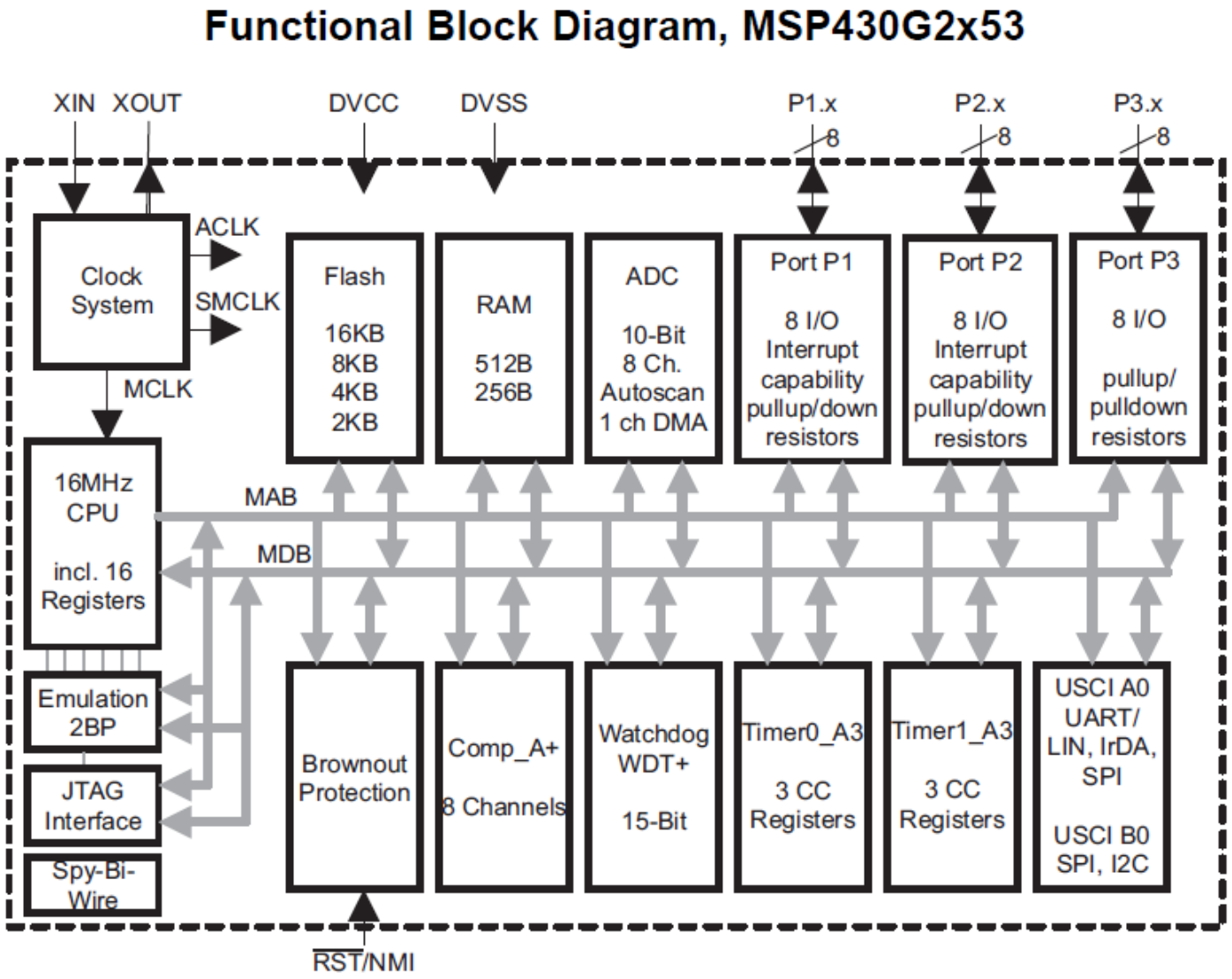
MSP430 Documentation

- User's Guide
  - http://www.ti.com/lit/ug/slau144j/slau144j.pdf
- Datasheet
  - http://www.ti.com/lit/ds/symlink/msp430g2553.pdf
- Errata
  - http://www.ti.com/lit/er/slaz440g/slaz440g.pdf

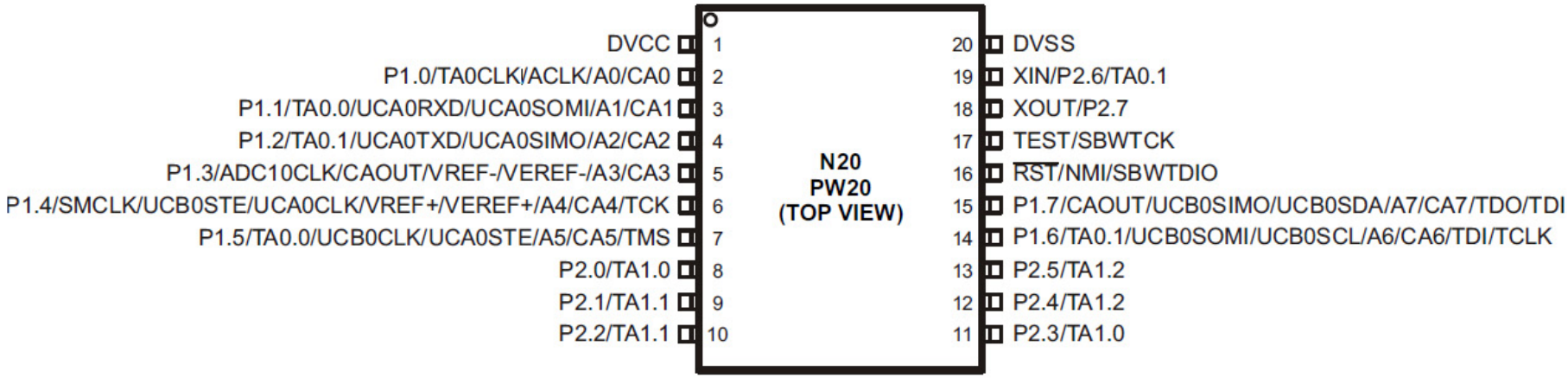
Example Today

- General Purpose I/O
- System Clock
- TimerA
- Interrupts

MSP430 Internal Block Diagram



Pin Assignments



NOTE: ADC10 is available on MSP430G2x53 devices only.

NOTE: The pulldown resistors of port P3 should be enabled by setting P3REN.x = 1.

Digital I/O Registers

8.2.1 Input Register PxIN

Each bit in each PxIN register reflects the value of the input signal at the corresponding I/O pin when the pin is configured as I/O function.

Bit = 0: The input is low

Bit = 1: The input is high

8.2.2 Output Registers PxOUT

Each bit in each PxOUT register is the value to be output on the corresponding I/O pin when the pin is configured as I/O function, output direction, and the pullup/down resistor is disabled.

Bit = 0: The output is low

Bit = 1: The output is high

If the pin's pullup/pulldown resistor is enabled, the corresponding bit in the PxOUT register selects pullup or pulldown.

Bit = 0: The pin is pulled down

Bit = 1: The pin is pulled up

8.2.3 Direction Registers PxDIR

Each bit in each PxDIR register selects the direction of the corresponding I/O pin, regardless of the selected function for the pin. PxDIR bits for I/O pins that are selected for other functions must be set as required by the other function.

Bit = 0: The port pin is switched to input direction

Bit = 1: The port pin is switched to output direction

8.2.4 Pullup/Pulldown Resistor Enable Registers PxREN

Each bit in each PxREN register enables or disables the pullup/pulldown resistor of the corresponding I/O pin. The corresponding bit in the PxOUT register selects if the pin is pulled up or pulled down.

Bit = 0: Pullup/pulldown resistor disabled

Bit = 1: Pullup/pulldown resistor enabled

8.2.5 Function Select Registers PxSEL and PxSEL2

Port pins are often multiplexed with other peripheral module functions. See the device-specific data sheet to determine pin functions. Each PxSEL and PxSEL2 bit is used to select the pin function - I/O port or peripheral module function.

Clock Module

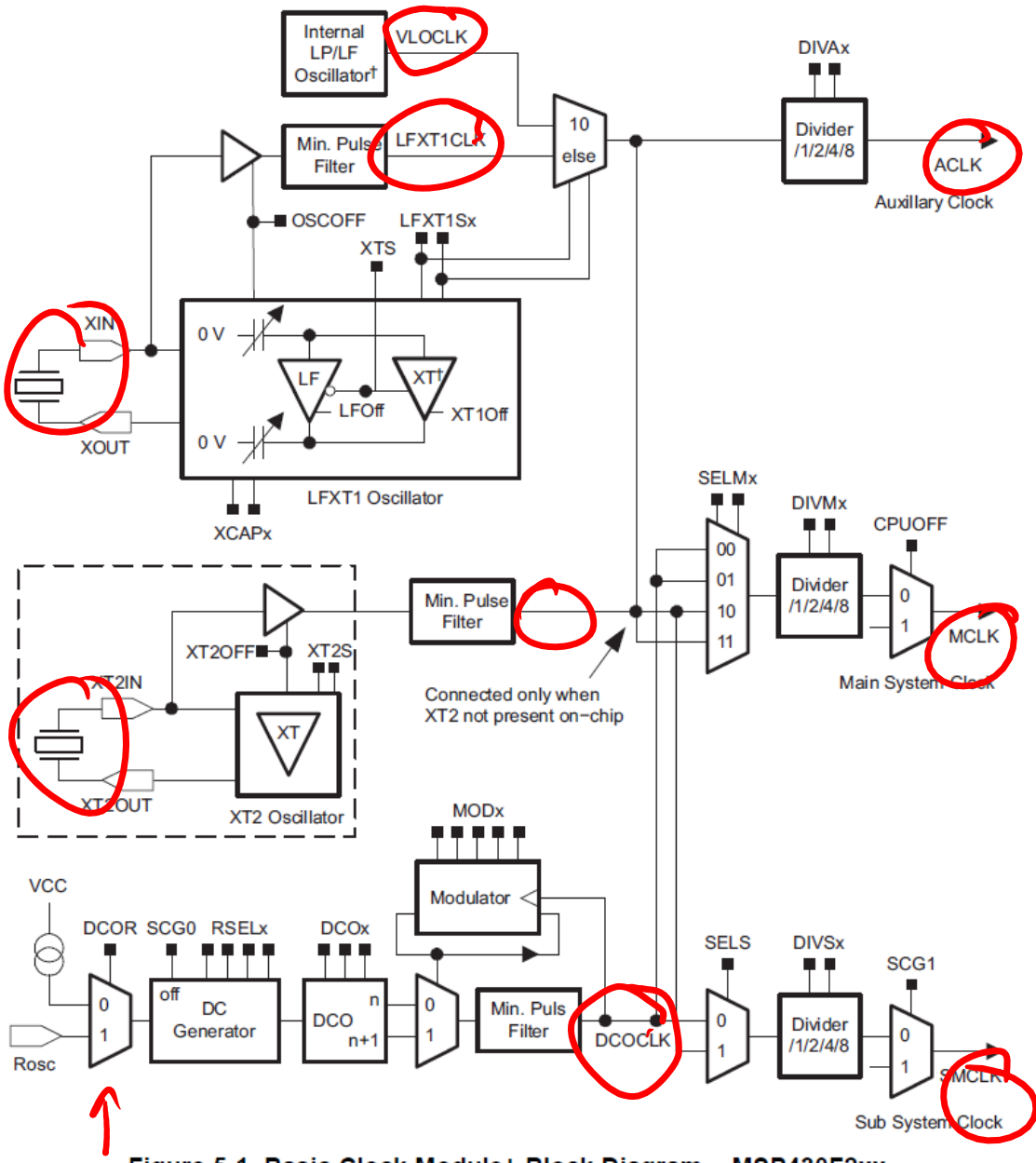


Figure 5-1. Basic Clock Module+ Block Diagram – MSP430F2xx

Clock Registers (1/2)

5.3.1 DCOCTL, DCO Control Register

7	6	5	4	3	2	1	0
DCOx			MODx				
rw-0	rw-1	rw-1	rw-0	rw-0	rw-0	rw-0	rw-0

DCOx	Bits 7-5	DCO frequency select. These bits select which of the eight discrete DCO frequencies within the range defined by the RSELx setting is selected.
MODx	Bits 4-0	Modulator selection. These bits define how often the f_{DCO+1} frequency is used within a period of 32 DCOCLK cycles. During the remaining clock cycles (32-MOD) the f_{DCO} frequency is used. Not useable when DCOx = 7.

5.3.2 BCSCTL1, Basic Clock System Control Register 1

7	6	5	4	3	2	1	0
XT2OFF	XTS(1)(2)	DIVAx		RSELx			
rw-(1)	rw-(0)	rw-(0)	rw-(0)	rw-0	rw-1	rw-1	rw-1

XT2OFF	Bit 7	XT2 off. This bit turns off the XT2 oscillator 0 XT2 is on 1 XT2 is off if it is not used for MCLK or SMCLK.
XTS	Bit 6	LFXT1 mode select. 0 Low-frequency mode 1 High-frequency mode
DIVAx	Bits 5-4	Divider for ACLK 00 /1 01 /2 10 /4 11 /8
RSELx	Bits 3-0	Range select. Sixteen different frequency ranges are available. The lowest frequency range is selected by setting RSELx = 0. RSEL3 is ignored when DCOR = 1.

Clock Registers (2/2)

5.3.3 BCSCTL2, Basic Clock System Control Register 2

7	6	5	4	3	2	1	0
SELMx		**DIVMx**		**SELS**	**DIVSx**		**DCOR**(1)(2)
rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0	rw-0

SELMx	Bits 7-6	Select MCLK. These bits select the MCLK source.
	00	DCOCLK
	01	DCOCLK
	10	XT2CLK when XT2 oscillator present on-chip. LFXT1CLK or VLOCLK when XT2 oscillator not present on-chip.
	11	LFXT1CLK or VLOCLK
DIVMx	Bits 5-4	Divider for MCLK
	00	/1
	01	/2
	10	/4
	11	/8
SELS	Bit 3	Select SMCLK. This bit selects the SMCLK source.
	0	DCOCLK
	1	XT2CLK when XT2 oscillator present. LFXT1CLK or VLOCLK when XT2 oscillator not present
DIVSx	Bits 2-1	Divider for SMCLK
	00	/1
	01	/2
	10	/4
	11	/8
DCOR	Bit 0	DCO resistor select. Not available in all devices. See the device-specific data sheet.
	0	Internal resistor
	1	External resistor

Timer A Block Diagram

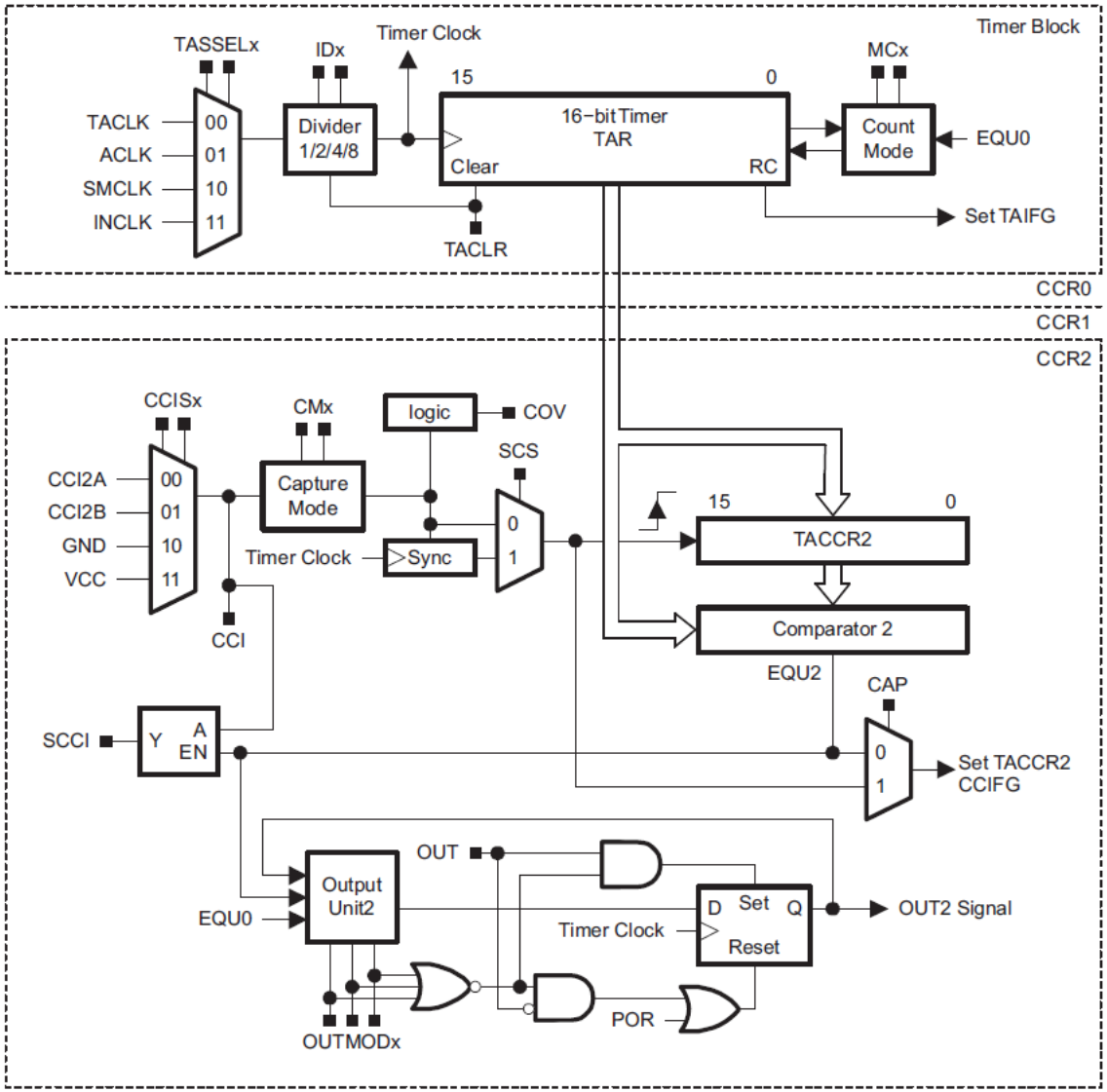


Figure 12-1. Timer_A Block Diagram

Timer A Operation – Up/Down Mode

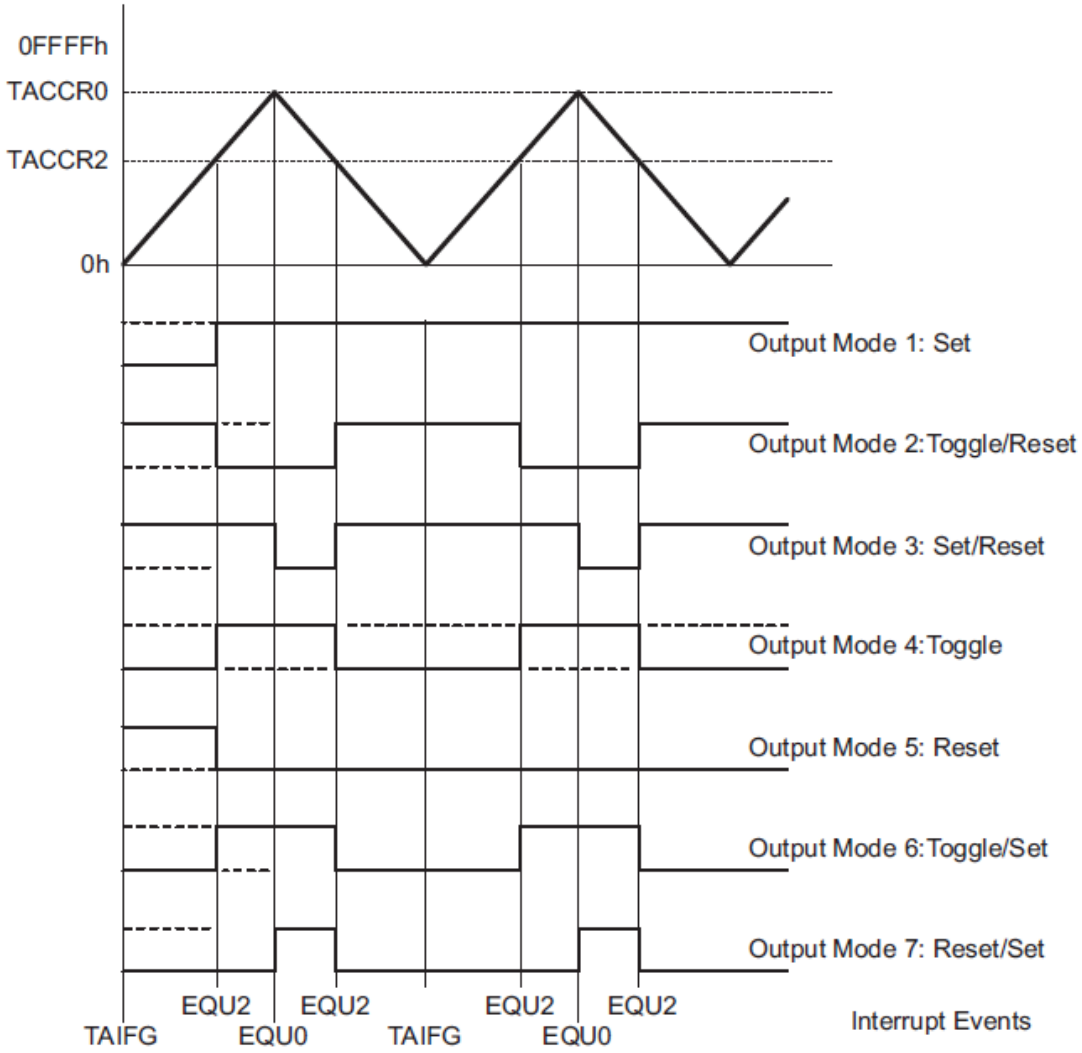


Figure 12-14. Output Example—Timer in Up/Down Mode

Timer A Registers (1/2)

12.3.1 TACTL, Timer_A Control Register

15	14	13	12	11	10	9	8
Unused						TASSELx	
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

7	6	5	4	3	2	1	0
IDx		MCx		Unused	TACLR	TAIE	TAIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)

Unused	Bits 15-10	Unused
TASSELx	Bits 9-8	Timer_A clock source select
		00 TACLK
		01 ACLK
		10 SMCLK
		11 INCLK (INCLK is device-specific and is often assigned to the inverted TBCLK) (see the device-specific data sheet)
IDx	Bits 7-6	Input divider. These bits select the divider for the input clock.
		00 /1
		01 /2
		10 /4
		11 /8
MCx	Bits 5-4	Mode control. Setting MCx = 00h when Timer_A is not in use conserves power.
		00 Stop mode: the timer is halted.
		01 Up mode: the timer counts up to TACCR0.
		10 Continuous mode: the timer counts up to 0FFFFh.
		11 Up/down mode: the timer counts up to TACCR0 then down to 0000h.
Unused	Bit 3	Unused
TACLR	Bit 2	Timer_A clear. Setting this bit resets TAR, the clock divider, and the count direction. The TACLR bit is automatically reset and is always read as zero.
TAIE	Bit 1	Timer_A interrupt enable. This bit enables the TAIFG interrupt request.
		0 Interrupt disabled
		1 Interrupt enabled
TAIFG	Bit 0	Timer_A interrupt flag
		0 No interrupt pending
		1 Interrupt pending

Timer A Registers (2/2)

12.3.4 TACCTLx, Capture/Compare Control Register

15	14	13	12	11	10	9	8
CMx		CCISx		SCS	SCCI	Unused	CAP
rw-(0)	rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	r0	rw-(0)
7	6	5	4	3	2	1	0
OUTMODx			CCIE	CCI	OUT	COV	CCIFG
rw-(0)	rw-(0)	rw-(0)	rw-(0)	r	rw-(0)	rw-(0)	rw-(0)

CMx Bit 15-14 Capture mode
- 00 No capture
- 01 Capture on rising edge
- 10 Capture on falling edge
- 11 Capture on both rising and falling edges

CCISx Bit 13-12 Capture/compare input select. These bits select the TACCRx input signal. See the device-specific data sheet for specific signal connections.
- 00 CCIxA
- 01 CCIxB
- 10 GND
- 11 V_{CC}

SCS Bit 11 Synchronize capture source. This bit is used to synchronize the capture input signal with the timer clock.
- 0 Asynchronous capture
- 1 Synchronous capture

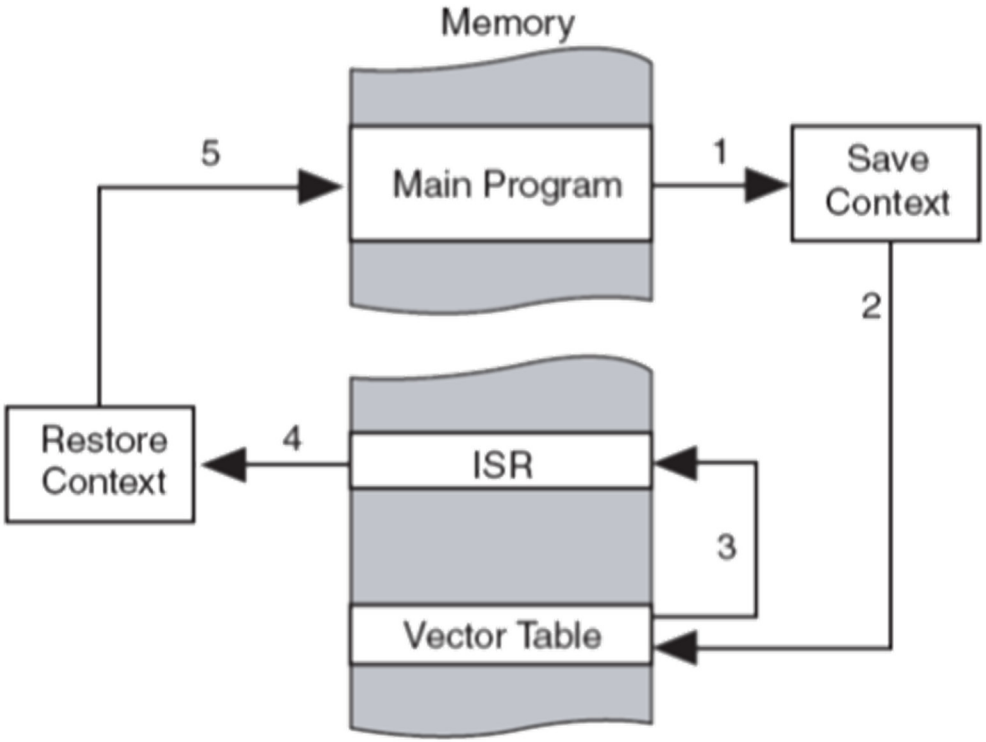
SCCI Bit 10 Synchronized capture/compare input. The selected CCI input signal is latched with the EQUx signal and can be read via this bit

Unused Bit 9 Unused. Read only. Always read as 0.

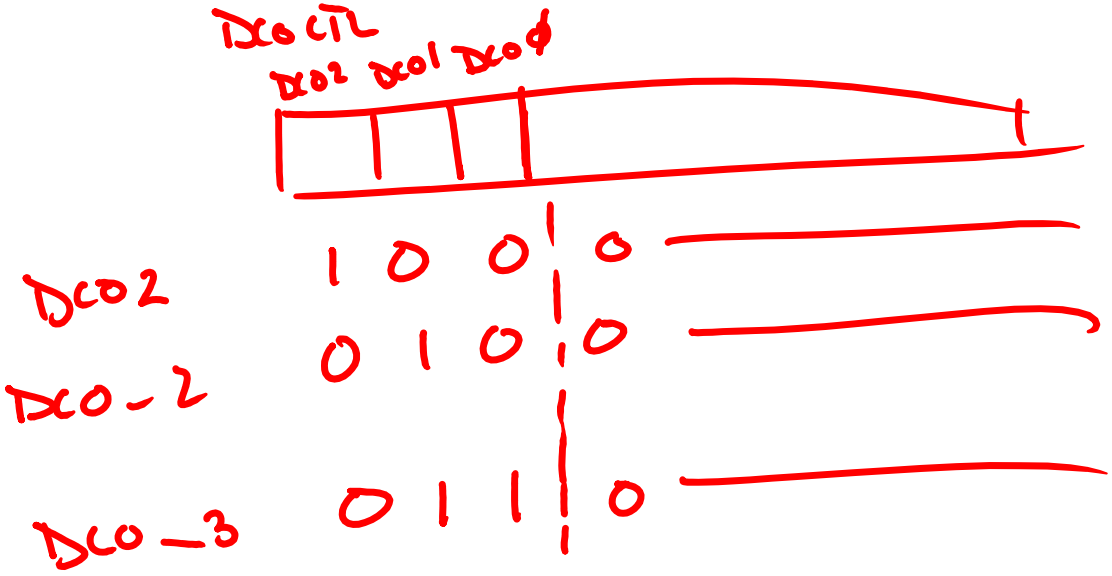
CAP Bit 8 Capture mode
- 0 Compare mode
- 1 Capture mode

OUTMODx Bits 7-5 Output mode. Modes 2, 3, 6, and 7 are not useful for TACCR0, because EQUx = EQU0.
- 000 OUT bit value
- 001 Set
- 010 Toggle/reset
- 011 Set/reset
- 100 Toggle
- 101 Reset
- 110 Toggle/set
- 111 Reset/set

Interrupts


Memory
Main Program
Save Context
Restore Context
ISR
Vector Table
1
2
3
4
5

Example Codes From Class



```
#include <msp430.h>
int main(void) {

int i;
   WDTCTL = WDTPW | WDTHOLD;// Stop watchdog timer

   //Set P1.0 and P1.6 high
   P1DIR = BIT0 + BIT6; // set both P1.0 and P1.6 to outputs
   P1OUT = BIT0 + BIT6; // set both outputs high

   //set P1.1 high
   P1DIR |= BIT1;
   //P1OUT |= BIT1; //set only P1.1 high
   P1OUT &= ~BIT1;

   //Boost clock frequency
   // DCOCTL = BIT7 + BIT6 + BIT5;
   DCOCTL = DCO0 + DCO1 +DCO2;
   BCSCTL1 = RSEL0 + RSEL1 + RSEL2 + RSEL3;
   BCSCTL2 = 0;
   BCSCTL3 = 0;

   //Configure Timer A for PWM
   TA0CTL = ID_0 + MC_1 + TASSEL_2; //No divider, Up mode, Source = SMCLK
   TA0CCTL1 = OUTMOD_2; // Toggle/reset
   TA0CCR0 = 20; // clocks per period
   TA0CCR1 = 15;// clocks before reset
   P1SEL |= BIT6;

   //Enable interrupts from Timer A
   TA0CCTL0 |= CCIE;
   _BIS_SR(GIE);

   while(1)
   {
            for (i=0; i<6; i++)
            {
                     __no_operation();
            }
   }
}
#pragma vector=TIMER0_A0_VECTOR
__interrupt void Timer_A(void)
{
         TA0CCR1 = TA0CCR1 + 1;
                  if (TA0CCR1 > 19)
                  {
                  TA0CCR1 = 1;
                  }
}
```